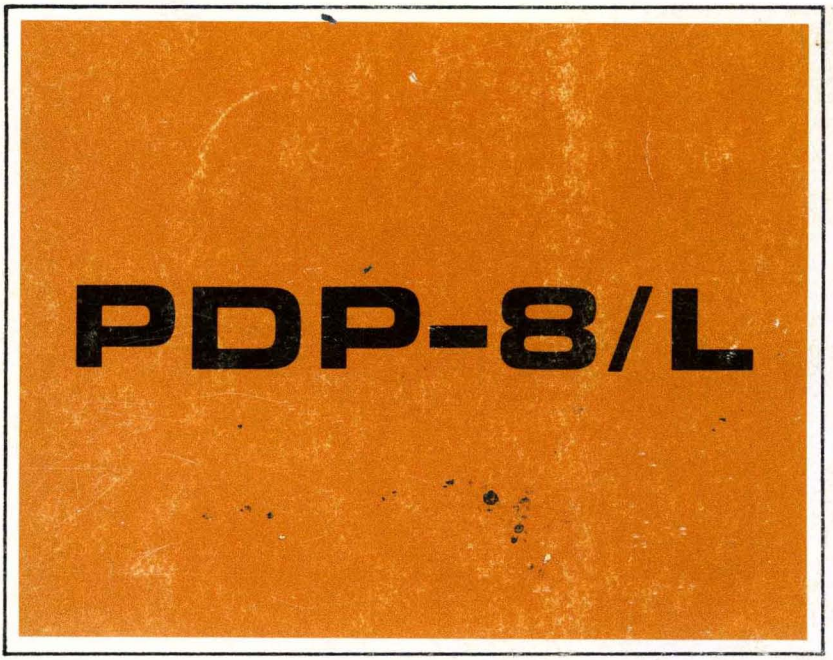
USERS HANDBOOK



DIGITAL EQUIPMENT CORPORATION

THE

digital

PDP-8/L USERS HANDBOOK

TABLE OF CONTENTS
PDP-8/L USERS HANDBOOK

APPENDICES

CHAPTER 1
SYSTEM INTRODUCTION

The Digital Equipment Corporation Programmed Data Processor-8/L (PDP-8/L) is a small-scale general-purpose computer. TTL monolithic integrated circuit modules are used throughout thereby providing efficient packaging, high reliability and noise immunity. The PDP-8/L is a one-address, fixed word length, parallel computer using 12 bit, two's complement arithmetic. Cycle time of the 4096-word random-address magnetic-core memory is 1.6 microseconds. Standard features of the system include indirect addressing and facilities for instruction skipping and program interruption as functions of input-output device conditions.

The 1.6 microsecond cycle time of the machine provides a computation rate of 312,500 additions per second. Addition is performed in 3.2 microseconds (with one number in the accumulator) and subtraction is performed in 6.4 microseconds (with the subtrahend in the accumulator). Multiplication is performed in approximately 336 microseconds by a subroutine that operates on two signed 12-bit numbers to 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 474 microseconds by a subroutine that produces a 12-bit quotient in the accumulator and a 12-bit remainder in core memory.

Flexible, high-capacity, input-output capabilities of the computer allow it to operate a variety of peripheral equipment. In addition to standard Teletype and perforated tape equipment, the system is capable of operating in conjunction with a number of optional devices such as high-speed perforated tape readers and punches, card equipment, a line printer, analog-to-digital converters, cathode-ray-tube displays, magnetic-tape equipment, and a 32,000 word random access disc file. Equipment of a special design is easily adapted for connection into the PDP-8/L system. The computer does not have to be modified when peripheral devices are added.

The PDP-8/L is completely self-contained, requiring no special power sources or environmental conditions. A single source of 115-volt, 50-60 cycle, single-phase power is required to operate the machine. Internal power supplies produce all of the operating voltages required. Modules utilizing TTL monolithic integrated circuits and built-in provisions for marginal checking insure reliable operation in ambient temperatures between 50 and 132 degrees Fahrenheit.

COMPUTER ORGANIZATION

The PDP8/L system is organized into a processor, core memory, and input/output facilities. All arithmetic, logic, and system control operations of the standard PDP-8/L are performed by the processor. Permanent (longer than one instruction time) local information storage and retrieval operations are performed by the core memory. The memory is continuously cycling, automatically performing a read and write operation during each computer cycle. Input and output address and data buffering for the core memory is per-

formed by registers of the processor, and operation of the memory is under control of timing signals produced by the processor. Due to the close relationship of operations performed by the processor and the core memory, these two elements are described together in this chapter of this handbook.

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

Standard peripheral equipment provided with each PDP-8/L 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 ten characters per second. The Teletype provides a means of supplying data to the computer from perforated tape or by means of a keyboard, and supplies data as an output from the computer in the form of perforated tape 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 and all optional input/output equipment are discussed in Chapter 7 of this handbook.

SYMBOLS

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

Symbol	Explanation
A => B	The content of register A is transferred into register B
0 => A	Register A is cleared to contain all binary zeros
Aj	Any given bit in A
A5	The content of bit 5 of register A
A5(1)	Bit 5 of register A contains a 1
A6 — 11	The content of bits 6 through 11 of register A
A6 — 11 => B0 — 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
∧	AND
$\overline{A}$	Ones complement of the content of A

MEMORY AND PROCESSOR FUNCTIONS

Major Registers

To store, retrieve, control, and modify information and to perform the required logical, arithmetic, and data processing operations, the core memory and the processor employ the logic complement shown in Figure 2 and described in the following paragraphs.

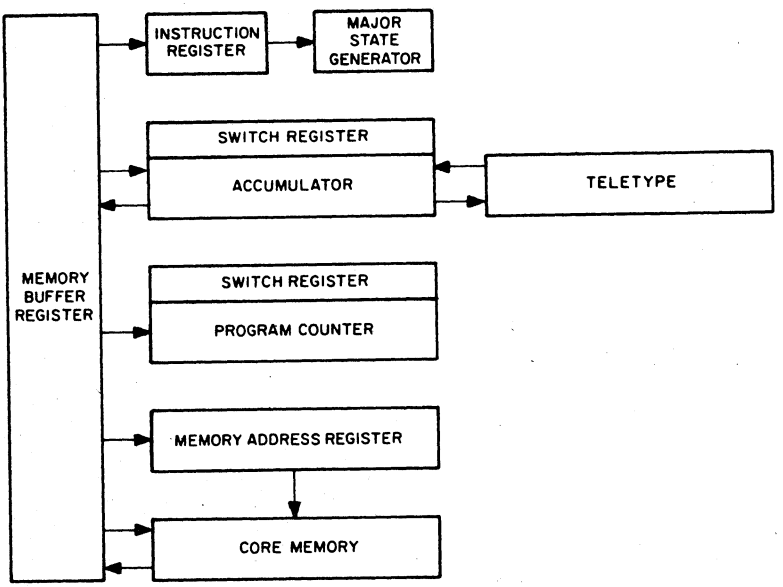


Figure 2 Simplified Block Diagram

ACCUMULATOR (AC)

Arithmetic and logic operations are performed in this 12-bit register. Under program control the AC can be cleared or complemented, its 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 the logical operation AND, the result remaining in the AC. The memory buffer register and the AC also have gates which allow them to be used together as the shift register and buffer register of a successive approximation analog-to-digital converter. The inclusive OR may be performed between the AC and the switch register on the operator console and the result left in the AC.

The accumulator also serves as an input-output register. All programmed information transfers between core memory and an external device pass through the accumulator.

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 complemented, and it can be rotated as part of the accumulator.

PROGRAM COUNTER (PC)

The program sequence, that is the order in which instructions are performed, is determined by the PC. This 12-bit register contains the address of the core memory location from which the next instruction will be taken. Information enters the PC from the core memory, via the memory buffer register, and from the switch register on the operator console. Information in the PC is transferred into the memory address register to determine the core memory address from which each instruction is taken. Incrementation of the content of the PC establishes the successive core memory locations of the program and provides skipping of an instruction based upon a programmed test of information or conditions.

MEMORY ADDRESS REGISTER (MA)

The address in core memory which is currently selected for reading or writing is contained in this 12-bit register. Therefore, all 4096 words of core memory can be addressed directly by this register. Data can be set into it from the memory buffer register, from the program counter, or from an I/O device using the data break facilities.

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 AC as data to be stored in core memory by means of the DEPOSIT key.

CORE MEMORY

The core memory provides storage for instructions to be performed and information to be processed or distributed. This random address magnetic core memory holds 4096 12-bit words in the standard PDP-8/L. Optional equipment extends the storage capacity by 4096 words or expands the word length to 13 bits to provide parity checking. Memory location 0_8 is used to store the content of the PC following a program interrupt, and location 1_8 is used to store the first instruction to be executed following a program interrupt. (When a program interrupt occurs, the content of the PC is stored in location 0_8, and program control is transferred to location 1 automatically.) Locations 10_8 through 17_8 are used for auto-indexing. All other locations can be used to store instructions or data.

The core memory contains numerous circuits such as read-write switches, address decoders, inhibit drivers, and sense amplifiers. These circuits perform the electrical conversions necessary to transfer information into or out of the core array and perform no arithmetic or logic operations upon the data. Since their operation is not discernible by the programmer or operator of the PDP-8/L these circuits are not described here in detail.

MEMORY BUFFER REGISTER (MB)

All information transfers between the processor registers and the core memory are temporarily held in the MB. Information can be transferred into the MB from the accumulator or memory address register. The MB can

be cleared, incremented by one or two, or shifted right. Information can be set into the MB from an external device during a data break or from core memory, via the sense amplifiers. Information is read from a memory location in 0.75 microsecond and rewritten in the same location in another 0.85 microsecond of one 1.6 microsecond memory cycle.

INSTRUCTION REGISTER (IR)

This 3-bit register contains the operation code of the instruction currently being performed by the machine. The three most significant bits of the current instruction are loaded into the IR from the memory buffer register during a Fetch cycle. The content of the IR is decoded to produce the eight basic instructions, and affect the cycles and states entered at each step in the program.

MAJOR STATE GENERATOR

One or more major states are entered serially to execute programmed instructions or to effect a data break. The major state generator establishes one state for each computer timing cycle. The Fetch, Defer, and Execute states are entered to determine and execute instructions. Entry into these states is produced as a function of the current instruction and the current state. The Word Count, Current Address, and Break states are entered during a data break. The Break state or all three of these states are entered based upon request signals received from peripheral I/O equipment.

Fetch (F)

During this state an instruction is read into the MB from core memory at the address specified by the content of the PC. The instruction is restored in core memory and retained in the MB. The operation code of the instruction is transferred into the IR to cause enactment, and the content of the PC is incremented by one.

If a multiple-cycle instruction is fetched, the following major state will be either Defer or Execute. If a one-cycle instruction is fetched, the operations specified are performed during the last part of the Fetch cycle and the next state will be another Fetch.

Defer (D)

When a 1 is present in bit 3 of a memory reference instruction, the Defer state is entered to obtain the full 12-bit address of the operand from the address in the current page or page 0 specified by bits 4 through 11 of the instruction. The process of address deferring is called indirect addressing because access to the operand is addressed indirectly, or deferred, to another memory location.

Execute (E)

This state is entered for all memory reference instructions except jump. During an AND, two's complement add, or increment and skip if zero instruction, the content of the core memory location specified by the address portion of the instruction is read into the MB and the operation specified by bits 0 through 2 of the instruction is performed. During a deposit and clear accumulator instruction the content of the AC is transferred into the MB and is stored in core memory at the address specified in the instruction. During a jump to subroutine instruction this state occurs to write the content of the PC into the core memory address designated by the instruction and to transfer this address into the PC to change program control.

Word Count (WC)

This state is entered when an external device supplies signals requesting a data break and specifying that the break should be a 3-cycle break. When this state occurs, a transfer word count in a core memory location designated by the device is read into the MB, is incremented by 1, and is rewritten in the same location. If the word count overflows, indicating that the desired number of data break transfers will be enacted at completion of the current break, the computer transmits a signal to the device. The Current Address state immediately follows the Word Count state.

Current Address (CA)

As the second cycle of a 3-cycle data break, this cycle establishes the address for the transfer that takes place in the following cycle (Break state). Normally the location following the word count is read from core memory into the MB, is incremented by 1 to establish sequential addresses for the transfers, and is transferred into the MA to determine the address selected for the next cycle. When the word count operation is not used, the device supplies an inhibit signal to the computer so that the word read during this cycle is not incremented. Transfers occur at sequential addresses due to incrementing during the Word Count state. During this sequence the word in the MB is rewritten at the same location and the MB is cleared at the end of the cycle. The Break state immediately follows the Current Address state.

Break (B)

This state is entered to enact a data transfer between computer core memory and an external device, either as the only state of a 1-cycle data break or as the final state of a 3-cycle data break. When a break request signal arrives and the cycle select signal specifies a 1-cycle break, the computer enters the Break state at the completion of the current instruction. Information transfers occur between the external device and a device-specified core memory location, through the MB. When this transfer is complete, the program sequence resumes from the point of the break. The data break does not affect the content of the AC, L, and PC.

OUTPUT BUS DRIVERS

Output signals from the computer processor are power amplified by output bus driver modules of the Standard PDP-8/L; allowing these signals to drive a heavy circuit load.

FUNCTIONAL SUMMARY

Operation of the computer is accomplished on a limited scale by keys on the operator console. Operation in this manner is limited to address and data storage by means of the switch register, core memory data examination, the normal start/stop/continue control, and the single step operation that allows a program to be monitored visually as a maintenance operation. Most of these manually initiated operations are performed by executing an instruction in the same manner as by automatic programming, except that the gating is performed by special pulses rather than by the normal clock pulses. In automatic operation, instructions stored in core memory are loaded into the memory buffer register and executed during one or more computer cycles. Each instruction determines the major control states that must be entered for its execution. Each control state lasts for one 1.6 microsecond computer cycle and is divided into distinct time states which can be used to perform sequential logical operations. Performance of any function of the computer is controlled by gating of a specific instruction during a specific major control state and a specific time state.

TIMING AND CONTROL ELEMENTS

The circuit elements that determine the timing and control of the operation of the major registers of the PDP-8/L are added to Figure 2 to form Figure 3.

Figure 3 shows the timing and control elements described in the succeeding paragraphs and indicates their relationship to the major registers. These elements can be grouped categorically into timing generators, register controls, and program controls.

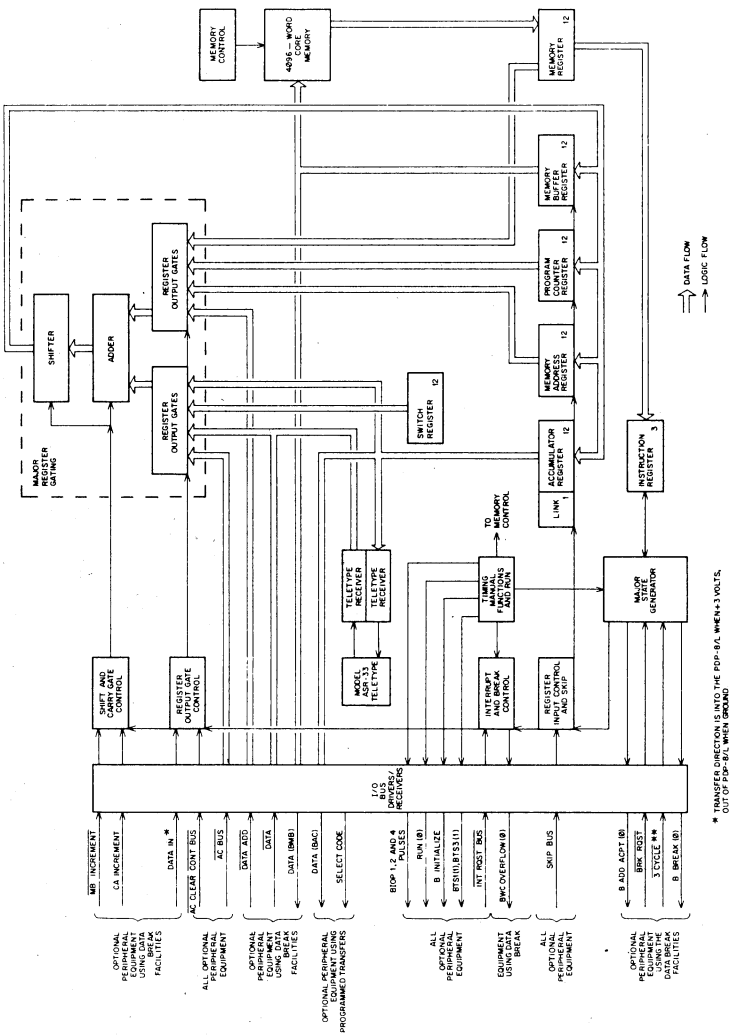


Figure 3. PDP-8/L Timing and Control Element Block Diagram

TIMING GENERATORS

Timing pulses used to determine the computer cycle time and used to initiate sequential time-synchronized gating operations are produced by the timing signal generator. Timing pulses used during operations resulting from the use of the keys and switches on the operator console are produced by the special pulse generator. Pulses that reset registers and control circuits during power turn on and turn off operations are produced by the power clear pulse generator. Several of these pulses are available to peripheral devices using programmed or data break information transfers.

Register Controls

The AC, MA, MB and PC each have gated inputs and gated outputs. The gated input bus of each register is tied to a common register bus that is the output of the major register gating circuit. The data on the common register bus originates from the various outputs of each register and can be modified by the ADDER or SHIFTER in the major register gating circuit. When the contents of a register are to be transferred to another register, its contents are gated by the register output gate control onto the common register bus and strobed into the appropriate register by the register input control. Data can therefore be transferred between registers directly by disabling the ADDER and SHIFTER or can be modified during transfer to provide SHIFT, CARRY and SKIP operations. Operations such as incrementing a register are accomplished simply by gating the output of the register onto the register bus, enabling the ADDER, and strobing the results back into the same register.

PROGRAM CONTROLS

Cirucits are also included in the PDP-8/L that produces the IOP pulses which initiate operations involved in input-output transfers, determine the advance of the computer program, and allow peripheral equipment to cause a program interrupt of the main computer program to transfer program control to a subroutine which performs some service for the I/O device.

Interface

The input/output portion of the PDP-8/L is extremely flexible and interfaces readily with special equipment, especially in real time data processing and control environments.

The PDP-8/L utilizes positive logic within the computer and on the input/output "bus" system. This makes the PDP-8/L compatible with new peripheral equipment offered by Digital and other manufacturers, and simplifies interface to specialized systems which may be constructed of DTL and TTL circuitry. An external option (DW08-A) converts the positive bus to a negative level bus, expanding the system flexibility by making direct interface to all 8 Family peripherals with negative logic. With this option both positive and negative peripherals may be connected.

The PDP-8/L 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/L need not be modified when adding new peripheral devices.

Data transfers may also be made directly with core memory at a high speed using the data break facility. This is a completely separate I/O system from the one described previously. It is optional in the PDP-8/L and is ordinarily used with fast I/O devices such as magnetic drums or tapes. Transfers through the data break facility are interlaced with the program in progress. They are initiated by a request from the peripheral device and not by programmed instruction. Thus, the device may transfer a word with memory whenever it is ready and does not have to wait for the program to issue an instruction. Computation may proceed on an interlaced basis with these transfers.

Interface signal characteristics are indicated in Chapter 11.

CHAPTER 2
STANDARD PDP-8/L OPERATION

Controls and Indicators

Manual control of the PDP-8/L 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 1, and their location is shown in Figure 4. The functions of all controls and indicators of the Model 33 ASR Teletype unit are described in Table 2, as they apply to operation of the computer. The Teletype console is shown in Figure 5.

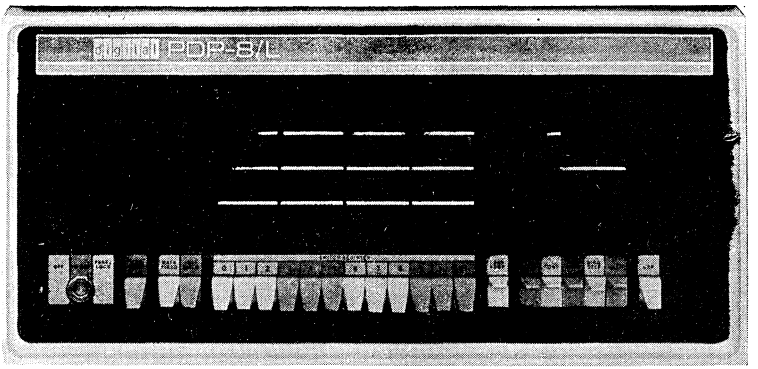


Figure 4. PDP-8/L Operator Console

TABLE 1. OPERATOR CONSOLE CONTROLS AND INDICATORS

Control or Indicator	Function
POWER/PANEL LOCK SWITCH	In the counterlockwise position this key operated switch removes primary power from the computer. In the center position it applies power and enables all manual control. In the clockwise position it maintains power but all keys and switches except the SWITCH REGISTER switches on the operator console are disabled. In this condition the program cannot be disturbed by inadvertent key operation. The program can, however, monitor the content of the SR by execution of the OSR instruction.

TABLE 1. OPERATOR CONSOLE CONTROLS AND INDICATORS (continued)

Control or Indicator	Function
START key	Starts the computer program by turning off the program interrupt circuits; clearing the AC, L, MB, and IR; setting the Fetch state, transferring the content of the PC into the MA; and setting the RUN flip-flop. 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 PC, sets the content of the INST FIELD switch into the IF, and sets the content of the DATA FIELD switch into the DF.
DEPOSIT key	Lifting this key sets the content of the SR into the MB and core memory at the address specified by the current content of the PC. 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 sets the content of core memory at the address specified by the content of the PC 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 cycle in progress at the time the key is pressed.
SINGLE STEP switch	The switch is off in the up position. In the down position the switch causes the RUN flip-flop to be cleared to disable the timing circuits at the end of one cycle of operation. Thereafter, repeated operation of the CONTINUE key steps the program one cycle at a time so that the content of registers can be observed in each state.

TABLE 1. OPERATOR CONSOLE CONTROLS AND INDICATORS (continued)

Control or Indicator	Function
SWITCH REGISTER switches	Provide a means of manually setting a 12-bit word into the machine Switches in the up position; corresponds to binary ones, down to zeros. 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 SR can be set into the AC under program control by means of the OSR instruction.
DATA FIELD SWITCH*	The DF switch serves 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 SWITCH*	The IF switch serves 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.
EA INDICATOR	This indicates the content of the extended address field being addressed. If the current memory address is for data, it is the content of the DF flip flop. If the current memory address is an instruction, it is the content of the IF flip flop.
MEMORY PROTECT Switch & Indicator	The switch is off in the down position. In the up position the switch causes the top page, 200_8 (128 decimal) locations, of memory field 0 to be protected from any memory modifying instruction. When protected the top page may be referenced by instructions but any change to the content will be inhibited and the machine will halt with the PROT indicator on. Manual restart will clear indicator and error halt.
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. If the machine is in a fetch cycle, the MA indicates the address of the current instruction (the content of the PC at the beginning of the fetch cycle).

*Activated only on systems containing the Type MC8/L Memory Extension option.

TABLE 1. OPERATOR CONSOLE CONTROLS AND INDICATORS (continued)

Control or Indicator	Function
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	Indicates the content of the three bit instruction register and therefore the basic instruction being executed.
FETCH, EXECUTE DEFER, WORD COUNT, CURRENT ADDRESS, BREAK indicators	Indicate the primary control state of the machine and that the current memory cycle is a Fetch (F), Execute (E), defer (D) or Break (B) cycle, respectively. Word Count (WC) and Current Address (CA) indicate the first and second cycles of a 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.
RUN indicator	When lit, the internal timing circuits are enabled and the machine performs instructions.
PARITY INDICATOR*	Indicates the 1 status of the Parity Error flip flop. When lit a parity error has been detected.

*Activated only on systems containing the MP-8/L Memory Parity Option.

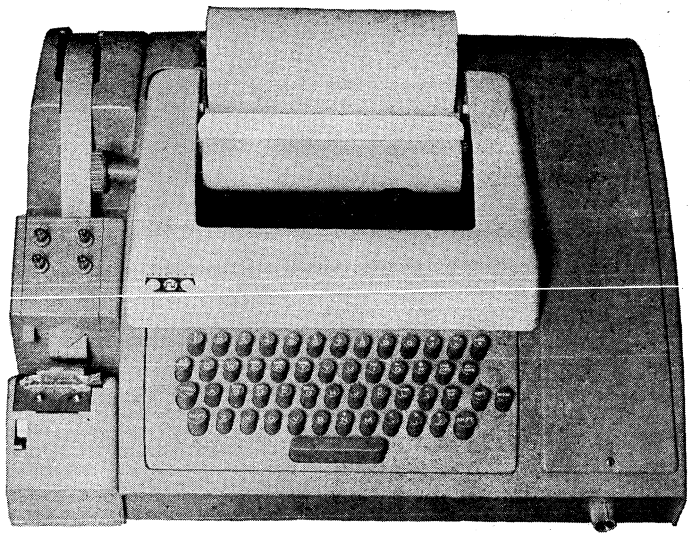


Figure 5. Teletype Model 33 ASR Console

TABLE 2. 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.

Control or Indicator	Function
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/L 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 PDP8/L, 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 leader program into the computer core memory. The readin mode (RIM) loader is a program used to automatically load programs into PDP-8/L from perforated tape in RIM format. This program and the RIM tape format are described in Appendix 4 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/L core memory:

1. Turn the POWER/PANEL LOCK switch clockwise to center position.
2. Set the MEMORY PROTECT switch down.
3. 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 MA, as designated by lighted MEMORY ADDRESS indicators corresponding to switches in the 1 (up) position and unlighted indicators corresponding to switches in the 0 (down) position.

4. Set the SR to correspond with the data or instruction word to be stored at the address just set into the MA. 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 MA 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 MA as in step 2, then press the EXAMINE key. The content of the address is then designated by the MEMORY BUFFER indicators. The content of the MA 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 POWER/PANEL LOCK switch clockwise to center position.

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. Set the MEMORY PROTECT switch down.

5. Load the starting address of the RIM loader program (not the address of the program to be loaded) into the MA and PC by means of the SR and the LOAD ADDRESS key.

6. 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 MA and 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/L for typing, punching tape, or duplicating tapes. To use the Teletype in this manner:

1. Assure that the computer POWER/PANEL LOCK switch is turned clockwise to center position.

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 3 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 push-

button 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.

Program Control

If the program is stopped at the end of an instruction by raising the SINGLE STEP key, then the LOAD ADDRESS, EXAMINE, and DEPOSIT keys may be used without changing the AC. The program may then be resumed by resetting the PC using LOAD ADDRESS and by pressing CONTINUE.

CHAPTER 3

MEMORY AND PROCESSOR BASIC PROGRAMMING*

MEMORY ADDRESSING

The following terms are used in memory address programming:

Term	Definition
Page	A block of 128 core memory locations (200_8 addresses). The page containing the instruction being executed; as determined by bits 0 through 4 of the program counter.
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

*See Appendix I for Program Abstracts.

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 Location	Content	Page 1 Location	Content	Page 2 Location	Content	Remarks
				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:

Address	Purpose
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 10_8 and 17_8 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 12_8 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-index 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/L 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 core 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/L 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. In PDP-8/L 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.

PROGRAMMING SYSTEM

The programming system for the PDP-8/L includes: the Symbolic Assemblers, FORTRAN System Compiler, Symbolic Tape Editor, Floating Point Package, DISC/DECtape Keyboard monitor, mathematical function subroutines, and utility and maintenance programs. All operate with the basic computer. The programming system was designed to simplify and accelerate the process of learning to program. At the same time, experienced programmers will find that it incorporates many advanced features. The system is intended to make immediately available to each user the full, general-purpose data processing capability of the computer and to serve as the operating nucleus for a growing library of programs and routines to be made available to all installations. New techniques, routines, and programs are constantly being developed, field-tested, and documented in the Digital Program Library for incorporation in users' systems.

Assemblers

The use of an assembly program has become standard practice in programming digital computers. This process allows the programmer to code his instructions in a symbolic language, one he can work with more conveniently than the 12-bit binary numbers which actually operate the computer. The assembly program then translates the symbolic language program into its machine code equivalent. The advantages are significant: the symbolic language is more meaningful and convenient to a programmer than a numeric code; instructions or data can be referred to by symbolic names without concern for, or even knowledge of, their actual addresses in core memory; decimal and alphabetical data can be expressed in a form more convenient than binary numbers; programs can be altered without extensive changes; and debugging is considerably simplified.

Two Assemblers Are Available:

1. PAL III is a basic assembler allowing symbolic references, symbolic origins, and expressions. The output is in a form suitable for input to the binary loader. High or low-speed paper tape input is accepted.
2. MACRO-8 is an advanced assembler which has the same basic features of PAL III and in addition, MACRO capability, literals, off-page references, and high/low-speed paper tape input and output.

DISC/DECtape Keyboard Monitor

A Keyboard Monitor is available to users with a DISC or DECtape System which allows the user to save core images on the DISC or DECtape System device and restore these core images to memory. Programs modified to work under the Monitor include: FORTRAN, EDITOR, DDT LOADER, and an Assembler. In addition, the user may save his own core images and restore them and use the remainder of the available device storage for temporary storage of source or binary data.

FORTRAN Compiler (4K)

The FORTRAN (for FORmula TRANslation) compiler lets the user express the problem he is trying to solve in a mixture of English words and mathematical statements that is close to the language of mathematics and is also intelligible to the computer. In addition to reducing the time needed for program preparation, the compiler enables users with little or no knowledge of the computer's organization and operating language to write effective programs for it. The FORTRAN Compiler contains the instructions the computer requires to perform the clerical work of translating the FORTRAN version of the problem statement into an object program in machine language. It also produces diagnostic messages. After compilation, the object program, the operating system and the data it will work with, are loaded into the computer for solution of the problem.

The FORTRAN language consists of four general types of statements: arithmetic, logic, control, and input/output. FORTRAN functions include addition, subtraction, multiplication, division, sine, cosine, arctangent, square root, natural logarithm, and exponential.

FORTRAN Compiler (8K)

The 8K FORTRAN compiler is an extension of the 4K FORTRAN compiler which features the following additions:

1. U.S.A. Standard FORTRAN Syntax
2. Subroutines
3. Two levels of subscripting
4. Function subprograms
5. I/O supervisor
6. Relocatable link loadable output
7. Common
8. I, E, F, H, A, X, format specification
9. Arithmetic and trigonometric library

Symbolic On-Line Debugging Program

On-line debugging with DDT-8 gives the user dynamic printed program status information. It gives him close control over program execution, preventing errors ("bugs") from destroying other portions of his program. He can monitor the execution of single instructions or subsections, change instructions or data in any format, and output a corrected program at the end of the debugging session.

Using the standard Teletype keyboard/reader and teleprinter/punch, the user can communicate conveniently with the PDP-8/L in the symbols of his source language. He can control the execution of any portion of his object program by inserting breaks, or traps, in it. When the computer reaches a break, it transfers control of the object program to DDT. The user can then examine and modify the content of individual core memory registers to correct and improve his object program.

Symbolic Tape Editor

The Symbolic Tape Editor program is used to edit, correct, and update symbolic program tapes using the PDP-8/L the teletype unit and/or the high-speed reader. With the editor in core memory, the user reads in portions of his symbolic tape, removes, changes, or adds instructions or operands, and gets back a complete new symbolic tape with errors removed. He can work through the program instruction by instruction, spot check it, or concentrate on new sections. A character string search is available. The user can move one or more lines of text from one place to another.

Floating Point Package

The Floating Point Package permits the PDP-8/L to perform arithmetic operations that many other computers can perform only after the addition of costly optional hardware. Floating point operations automatically align the binary points of operands, retaining the maximum precision available by discarding leading zeros. In addition to increasing accuracy, floating point operations relieve the programmer of scaling problems common in fixed point operations. This is of particular advantage to the inexperienced programmer.

Mathematical Function Routines

The programming system also includes a set of mathematical function routines to perform the following operations in both single and double precision: addition, subtraction, multiplication, division, square root, sine, cosine, arctangent, natural logarithm, and exponential.

Utility and Maintenance Programs

PDP-8/L utility programs provide printouts or punchouts of core memory content in octal, decimal, or binary form, as specified by the user. Subroutines are provided for octal or decimal data transfer and binary-to-decimal, decimal-to-binary, and Teletype tape conversion.

A complete set of standard diagnostic programs is provided to simplify and expedite system maintenance. Program descriptions and manuals permit the user to effectively test the operation of the computer for proper core memory functioning and proper execution of instructions. In addition, diagnostic programs to check the performance of standard and optional peripheral devices are provided with the devices.

CHAPTER 4

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 through 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. Memory reference instruction execution times are multiples of the 1.6 microsecond memory cycle. Indirect addressing increases the execution time of a memory reference instruction by 1.6 microseconds. The augmented instructions: input-output transfer and operate, are performed in 4.25 and 1.6 microseconds, respectively. (All computer times are ±12%.)

MEMORY REFERENCE INSTRUCTIONS

Since the PDP-8/L 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, 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.

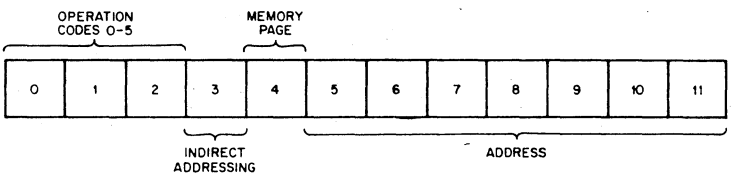


Figure 6. Memory Reference Instruction Bit Assignments

Word format of memory reference instructions is shown in Figure 6 and the instructions perform as follows:

Logical AND (AND Y)

Octal Code: 0
Indicators: FETCH, EXECUTE, IR=0
Execution Time: 3.2 microseconds with direct addressing, 4.8 microseconds with indirect addressing.
Operation: The AND operation is performed between the content of memory location Y and the content 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:

Original ACj	Yj	Final ACj
0	0	0
0	1	0
1	0	0
1	1	1

Symbol: ACj ∧ Yj = > ACj

Two's Complement Add (TAD Y)

Octal Code: 1
Indicators: FETCH, EXECUTE, IR=1
Execution Time: 3.2 microseconds with direct addressing, 4.8 microseconds with indirect addressing.
Operation: The content 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 AC0, the link is complemented. This feature is useful in multiple precision arithmetic.
Symbol: AC0 − 11 + Y0 − 11 = > AC0 − 11

Increment and Skip If Zero (ISZ Y)

Octal Code: 2
Indicators: FETCH, EXECUTE, IR=2
Execution Time: 3.2 microseconds with direct addressing, 4.8 microseconds with indirect addressing.
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 = > Y
If resultant Y0 − 11 = 0, then PC + 1 = > PC

Deposit and Clear AC (DCA Y)

Octal Code: 3
Indicators: FETCH, EXECUTE, IR=3
Execution Time: 3.2 microseconds with direct addressing, 4.8 microseconds with indirect addressing.
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 = > Y
then 0 = > AC

Jump to Subroutine (JMS Y)

Octal Code: 4
Indicators: FETCH, EXECUTE, IR=4
Execution Time: 3.2 microseconds with direct addressing, 4.8 microseconds with indirect addressing.
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 = > Y
Y + 1 = > PC

Jump to Y (JMP Y)

Octal Code: 5
Indicators: JMP, FETCH, IR=5
Execution Time: 1.6 microseconds with direct addressing, 3.2 microseconds with indirect addressing.
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 core memory. They are the input-output transfer, which has an operation code of 6, and the operate which has an operation code of 7. Bits 3 through 11 within these instructions function as an extension of the operation code and can be microprogrammed to perform several operations within one instruction. Augmented instructions are one-cycle (Fetch) instructions that initiate various operations as a function of bit microprogramming.

Input/Output Transfer Instruction

Microinstructions of the input-output transfer (IOT) initiate operation of peripheral equipment and effect information transfers between the processor and an I/O device. Specifically, upon recognition of the operation code 6 as an IOT instruction, the computer enters a 4.25 μsec expanded computer FETCH cycle by setting the PAUSE flip-flop and enabling the IOP generator to produce IOP 1, IOP 2 and IOP 4 pulses as a function of the three least significant bits of the instruction (bits 9 thru 11). These pulses occur at 1 microsecond intervals designated as event times 3, 2 and 1 as follows:

Instruction Bit	IOP Pulse	IOT Pulse	Event Time
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 7. Operations performed by IOT microinstructions are explained in Chapter 7.

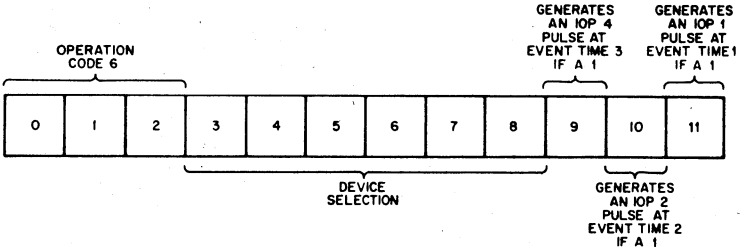


Figure 7. IOT Instruction Bit Assignments

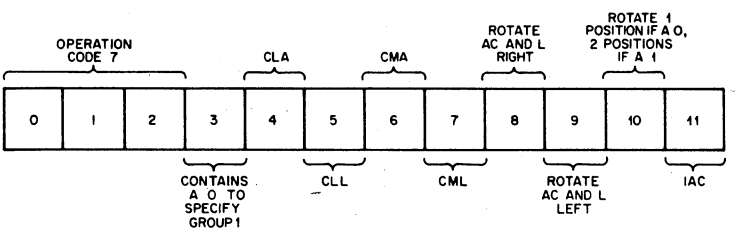
Operate Instruction

With operate instructions, the programmer can consider logical sequences occurring during one computer FETCH cycle. These sequences provide a logical method of forming microinstructions.

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

The Group 1 operate microinstruction format is shown in Figure 8. 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; although it is not logical to assign ones to bits 8 and 9 simultaneously since they specify conflicting operations. (The most frequently used combinations are listed in Appendix 2.)



LOGICAL SEQUENCE:

1 — CLA, CLL
2 — CMA, CML
3 — IAC
4 — RAR, RAL, RTR, RTL,

Figure 8. Group 1 Operate Instruction Bit Assignments

No Operation (NOP)

Octal Code: 7000
Sequence: None
Indicators: FETCH, IR=7
Execution Time: 1.6 microseconds
Operation: This command causes a 1-cycle delay in the program 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
Sequence: 3
Indicators: FETCH, IR=7
Execution Time: 1.6 microseconds
Operation: The content of the AC is incremented by one in two's complement arithmetic.
Symbol: AC + 1 => AC

Rotate Accumulator Left (RAL)

Octal Code: 7004
Sequence: 4
Indicators: FETCH, IR=7
Execution Time: 1.6 microseconds
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 ACO is shifted into the L, and the content of the L is shifted into AC11.
Symbol: ACj => ACj — 1
ACO => L
L => AC11

Rotate Two Left (RTL)

Octal Code: 7006
Sequence: 4
Indicators: FETCH, IR=7
Execution Time: 1.6 microseconds
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: ACj => ACj — 2
AC1 => L
ACO => AC11
L => AC10

Rotate Accumulator Right (RAR)

Octal Code: 7010
Sequence: 4
Indicators: FETCH, IR=7
Execution Time: 1.6 microseconds
Operation: The content of the AC is rotated one binary position to the right with the content of the link. The content of bits ACO — 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 ACO.
Symbol: ACj => ACj + 1
AC11 => L
L => ACO

Rotate Two Right (RTR)

Octal Code: 7012
Sequence: 4
Indicators: FETCH, IR=7
Execution Time: 1.6 microseconds
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: ACj $=>$ Acj + 2
AC10 $=$ L
AC11 $=$ ACO
L $=>$ AC1

Complement Link (CML)

Octal Code: 7020
Sequence: 2
Indicators: FETCH, IR=7
Execution Time: 1.6 microseconds
Operation: The content of the L is complemented.
Symbol: $\overline{L} => L$

Complement Accumulator (CMA)

Octal Code: 7040
Sequence: 2
Indicators: FETCH, IR=7
Execution Time: 1.6 microseconds
Operation: The content of the AC is set to the one's complement of the current content of the AC. The content of each bit of the AC is complemented individually.
Symbol: $\overline{ACj} => ACj$

Complement and Increment Accumulator (CIA)

Octal Code: 7041
Sequence: 2, 3
Indicators: FETCH, IR=7
Execution Time: 1.6 microseconds
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 content of the AC is complemented during sequence 2 and is incremented by one during sequence 3.
Symbol: $\overline{ACj} => Acj$,
then AC + 1 $=>$ AC

Clear Link (CLL)

Octal Code: 7100
Sequence: 1
Indicators: FETCH, IR=7
Execution Time: 1.6 microseconds
Operation: The content of the L is cleared to contain a 0.
Symbol: 0 $=>$ L

Set Link (STL)

Octal Code: 7120
Sequence: 1, 2
Indicators: FETCH, IR=7
Execution Time: 1.6 microseconds
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
Sequence: 1
Indicators: FETCH, IR=7
Execution Time: 1.6 microseconds
Operation: The content of each bit of the AC is cleared to contain a binary 0.
Symbol: 0 = > AC

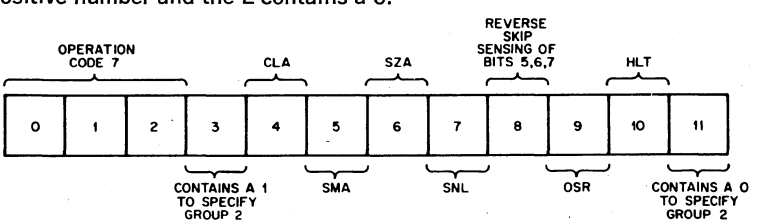
Set Accumulators (STA)

Octal Code: 7240
Sequence: 1, 2
Indicators: FETCH, IR=7
Execution Time: 1.6 microseconds
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

The Group 2 operate microinstruction format is shown in Figure 9. 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 2.)

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.



Logical Sequence:

1 (Bit 8 is a zero) — Either SMA or SZA or SNL
1 (Bit 8 is a one) — Both SPA and SNA and SZL
2 — CLA
3 — OSR, HLT

Figure 9. Group 2 Operate Instruction Bit Assignments

Halt (HLT)

Octal Code: 7402
Sequence: 3
Indicators: not RUN, IR=7
Execution Time: 1.6 microseconds
Operation: Clears the RUN flip-flop at Sequence 3, so that the program stops at the conclusion of the current machine cycle. This command can be combined with others in the OPR 2 group that are executed during either sequence 1, or 2, and so are performed before the program stops.
Symbol: 0 = > RUN

OR with Switch Register (OSR)

Octal Code: 7404
Sequence: 3
Indicators: FETCH, IR=7
Execution Time: 1.6 microseconds
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 into the AC.
Symbol: ACj V Srj = > ACj

Skip, Unconditional (SKP)

Octal Code: 7410
Sequence: 1
Indicators: FETCH, IR=7
Execution Time: 1.6 microseconds
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
Sequence: 1
Indicators: OPR, FETCH, IR=7
Execution Time: 1.6 microseconds
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
Sequence: 1
Indicators: FETCH, IR=7
Execution Time: 1.6 microseconds
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.
Symbol: If L = 0, then PC + 1 = > PC

Skip on Zero Accumulator (SZA)

Octal Code: 7440
Sequence: 1
Indicators: FETCH, IR=7
Execution Time: 1.6 microseconds
Operation: The content of each bit of the AC is sampled, and if any bit contains a 0 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 $=$ 0, then PC $+$ 1 $=>$ PC

Skip on Non-Zero Accumulator (SNA)

Octal Code: 7450
Sequence: 1
Indicators: FETCH, IR=7
Execution Time: 1.6 microseconds
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 $=>$ PC

Skip on Minus Accumulator (SMA)

Octal Code: 7500
Sequence: 1
Indicators: FETCH, IR=7
Execution Time: 1.6 microseconds
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 $=>$ PC

Skip on Positive Accumulator (SPA)

Octal Code: 7510
Sequence: 1
Indicators: OPR, FETCH, IR=7
Execution Time: 1.6 microseconds
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 $=>$ PC

Clear Accumulator (CLA)

Octal Code: 7600
Sequence: 2
Indicators: FETCH, IR=7
Execution Time: 1.6 microseconds
Operation: Each bit of the AC is cleared to contain a binary 0.
Symbol: 0 $=>$ AC

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. These instructions are:

Interrupt Turn On (ION)

Octal Code: 6001
Event Time: Not applicable
Indicators: FETCH, ION, IR=6
Execution Time: 4.25 microseconds
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 subroutine before allowing another interrupt to occur. This instruction has no effect 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: FETCH, IR=6
Execution Time: 4.25 microseconds
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 next instruction is taken from core memory location 1. 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 instruc-

tion immediately precedes the last instruction in the routine. A delay of one instruction (regardless of the execution time of the following instruction) 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 Z 0 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

DATA BREAK

Peripheral equipment connected to the data break facility can cause a temporary suspension in the program in progress to transfer information with the computer core memory, via the MB. One I/O device can be connected directly to the data break facility or up to seven devices can be connected to it through the Type DM01 Data Multiplexer. This cycle stealing mode of operation provides a high-speed transfer of individual words or blocks of information at core memory addresses specified by the I/O device. Since program execution is not involved in these transfers, the program counter, accumulator, and instruction register are not disturbed or involved in these transfers. The program is merely suspended at the conclusion of an instruction execution, and the data break is entered to perform the transfer, then the Fetch state is entered to continue the main program.

Data breaks are of two basic types: single-cycle and three-cycle. In a single-cycle data break, registers in the device (or device interface) specify the core memory address of each transfer and count the number of transfers to determine the end of data blocks. (See discussion of Data Break Transfers in Chapter 10.) In the three-cycle data break two computer core memory locations perform these functions, simplifying the device interface by omitting two hardware registers.

The computer receives the following signals from the device during a data break:

Signal	+3 Volts	0 Volts
Break Request	No break request	Break request
Cycle Select	One-cycle break	Three-Cycle break
Transfer Direction	Data into PDP-8/I	Data out of PDP-8/I
Increment CA Inhibit	CA incremented	CA not incremented
Increment MB (pulse)	MB not incremented	MB incremented
Address (12 bits)	Binary 0	Binary 1
Data (12 bits)	Binary 0	Binary 1

The computer sends the following signals to the device during a data break:

Signal	Characteristics
Data (12 Bits)	+3 volts = binary 1, 0 volts = binary 0
Address Accepted	.35—.45 microsecond negative pulse (+3 volts to 0 volts) beginning at TP4 of the break cycle.
WC Overflow	+3 volt to 0 volt level change occurring at TP2 time of the WC state and lasting for one machine cycle.
Buffered Break	0 volt when in Break state.

To initiate a data break an I/O device must supply four signals simultaneously to the data break facility. These signals are the Break Request signal, which sets the BRK SYNC flip-flop in the major state generator to control entry into the data break states (Word Count for a three-cycle data break or Break for a single-cycle data break); a Transfer Direction signal, supplied to the MB control element to allow data to be strobed into the MB from the peripheral

equipment and to inhibit reading from core memory; a Cycle Select Signal which controls gating in the major state generator to determine if the one-cycle or three-cycle data break is to be selected; and a core memory address of the transfer which is supplied to the input of the MA. When the break request is made, the data break replaces entry into the Fetch state of an instruction. Therefore the data break is entered at the conclusion of the Execute state of most memory reference instructions and at the conclusion of a Fetch state of augmented instructions. Having established the data break, each machine cycle is a Word Count, Current Address, or Break cycle until all data transfers have taken place, as indicated by removal of the Break Request signal by the peripheral equipment.

More exactly, the Break Request signal enables the BRK SYNC flip-flop. At TP1 time, the BRK SYNC flip-flop is set if the Break Request signal has been received, and is cleared otherwise.

At TP4 time of each machine cycle, the major state generator is set to establish the state for the cycle. At this time, the status of the BRK SYNC flip-flop is sampled and if in the 1 state, the Word Count or Break state is set into the major state generator and a data break commences.

Therefore, to initiate a data break, the Break Request must be at ground potential for at least 500 nanoseconds preceeding TP1 of the cycle preceeding the data break cycle. A Break Request signal should be supplied to the computer when the address, data, transfer direction, and cycle select signals are supplied to the computer.

When a data break occurs, the address designated by the device is loaded into the MA at TP4 time of the last cycle of the current instruction, and the major state generator is set to the World Count state if the Cycle Select signal is at ground, or is set to the Break state if this signal is at +3 volts. The program is delayed for the duration of the data break, commencing in the following cycle. A break request is granted only after completion of the current instruction as specified by the following conditions:

1. At the end of the Fetch cycle of an OPR or IOT instruction, or a directly addressed JMP instruction.
2. At the end of the Defer cycle of an indirectly addressed JMP instruction.
3. At the end of the Execute cycle of a JMS, DCA, ISZ, TAD, or AND instruction.

At the beginning of the Word Count cycle of a three-cycle data break or the Break cycle of a one-cycle data break, the address supplied to the input of the MA is strobed into the MA and the computer supplies an Address Accepted signal to the device. Entry into the Break cycle is indicated to the peripheral equipment by a Buffered Break signal and by an Address Accepted signal that can be used to enable gates in the device to perform tasks associated with the transfers. The Address Accepted signal is the most convenient control to be used by I/O equipment to disable the Break Request signal, since this signal must be removed at TP4 time to prevent continuance of the data break into the next cycle. If the Transfer Direction signal establishes the direction as out of the computer, the content of the core memory register at the address specified is transferred into the MB and is immediately available for strobing by the peripheral equipment. If the Transfer Direction signal specifies a data direction into the PDP-8/L, reading from core memory is inhibited and data is transferred into the MB from peripheral equipment.

The status of the BRK SYNC flip-flop is sensed at the beginning of a Break cycle to determine if an additional Break cycle is required. If a Break Request signal has been received since TP4, the Break state is maintained in the major state generator; if the Break Request signal has not been received by this time; the Fetch state is set into the major state generator to continue the program. The Break Request signal should be removed by the end of the Address Accepted signal if additional Break cycles are not required.

SINGLE-CYCLE DATA BREAK

One-cycle breaks transfer a data word into the computer core memory from the device, transfer a data word into a device from the core memory, or increment the content of a device-specified core memory location. In each of these types of data break one computer cycle is stolen from the program by each transfer; Break cycles occur singly (interleaved with the program steps) or continuously (as in a block transfer), depending upon the timing of the Break Request signal at rates of up to 625 khz.

During the memory strobe portion of the Break cycle, the content of the addressed cell is read into the MB if the transfer direction is out of the computer (into the I/O device). If the transfer direction is into the computer, generation of the Memory strobe pulse is inhibited. Information is transferred from the output data register at I/O device into the MB and is written into core memory during TS3 and TS4 times of the Break cycle. In an outward transfer, the write operation restores the original content of the address cell to memory.

If there is a further break request, another Break cycle is initiated. If there is no break request, the content of the PC is transferred into the MA, the IR is cleared, and the major state generator is set to Fetch. The program then executes the next instruction.

The increment MB facility is useful for counting iterations or events by means of a data break, so that the PC and AC are not disturbed. Within one Break cycle of 1.6 microseconds, a word is fetched from a device-specified core memory location, is incremented by one, and is restored to the same memory location. The Increment MB signal input must be supplied to the computer only during a Break cycle in which the direction of transfer is out of the PDP-8/L. These restrictions can be met by a simple AND gate in the device; an Increment MB signal is generated only when an event occurs, the Buffered Break signal from the computer is present, and the Transfer Direction signal supplied to the computer is at ground potential.

THREE-CYCLE DATA BREAK

The three-cycle data break provides an economical method of controlling the transfer of data between the computer core memory and fast peripheral devices. Transfer rates in excess of 208 khz are possible using this feature of the PDP-8/L.

The three-cycle data break differs from the one-cycle break in that a ground-level Cycle Select signal is supplied so that when the data break conditions are fulfilled the program is suspended and the Word Count state is entered. The Word Count state is entered to increment the fixed core memory location containing the word count. The device requesting the break supplies this address as in the one-cycle break, except that this is a fixed address supplied by wired ground and +3v signals rather than from a register.

Following the Word Count state a Current Address state occurs in which the location following the Word Count address is read, incremented by one, restored to memory, and loaded into the MA to be used as the transfer address. Then the normal Break state is entered to effect the transfer between the device and the computer memory cell specified by the MA.

Word Count State

When this state is entered, the contents of the core memory address specified by the external device plus 1 is loaded into the MB at TP2 time. The word count, established previously by instructions, is the 2's completment negative number equal to the required number of transfers. If the word becomes 0 when incremented, the computer generates a WC overflow signal and supplies it to the device. During TS3 and TS4 times, the incremented word count is rewritten in memory, the contents of the MA is incremented by 1 to establish the next location as the address for the following memory cycle, and the major state generator is set to the Current Address state.

Current Address State

Operations during the second cycle of the three-cycle data break depend upon the condition of the Increment CA Inhibit ($+1 \rightarrow$ CA Inhibit) signal supplied to the computer from the I/O device. At TP2 time, the MB is loaded with either the contents of the memory cell following the word count (Current Address register) or the incremented contents of the current address register (i.e. if CA Inhibit is at ground, the contents are loaded; if CA Inhibit is at $+3$ volts, the incremented contents are loaded). The Current Address register may be incremented to advance the address of the transfer to the next sequential location. During TS3 and TS4 times, the contents of the MB is rewritten into core memory, the address word in the MB is transferred into the MA to designate the address to be used in the succeeding memory cycle, and the major state generator is set to Break state.

Break State

The actual transfer of data between the external device and the core memory, through the MB, occurs during the Break state as during a single-cycle data break, except that the address is determined by the current content of the MA rather than directly by the device.

CHAPTER 6

OPTIONAL MEMORY AND PROCESSOR EQUIPMENT AND INSTRUCTIONS

MEMORY EXTENSION CONTROL AND MEMORY MODULE (MC8/L)

Note: Two types of Memory Extension Control and Memory Module are available for use with PDP-8/L. The MC-8/L-A is designed for operation when the MP-8/L parity option is not installed. When MP-8/L is installed, the MC-8/L-B should be used.

Extension of the storage capacity of the standard 4096-word core memory is accomplished by adding one field of 4096-word core memory. Field select control and extended address control for up to 8192 words is provided by the MEMORY EXTENSION CONTROL (MC8/L). Direct address of 8,192 words requires 13 bits ($2^{13} = 8{,}192$). However, since programs and data need not be directly addressed for execution of each instruction, a field can be program-selected, and all 12-bit addresses are then assumed to be within the current memory field. Program interrupt of a program in either field automatically specifies field 0, address 0 for storage of the program count. The memory extension control consists of several 1-bit flip-flop registers that extend addresses to 13 bits to establish or select a field.

The functional circuit elements which comprise the memory extension control perform as follows:

Instruction Field Register (IF): The IF is a 1-bit register that serves as an extension of the PC. The content of the IF determines the field from which all instructions are taken and the field from which operands are taken in directly-addressed AND, TAD, ISZ, or DCA instructions. Operating the LOAD ADDRESS key JAM transfers the contents of the INSTRUCTION FIELD switch register on the operator console into the IF register. During a JMP or JMS instruction the IF is set by a transfer of information contained in the instruction buffer register. When a program interrupt occurs, the content of the IF is automatically stored in bit 0 of the save field register for restoration to the IF from the instruction buffer register at the conclusion of the program interrupt subroutine.

Data Field Register (DF): This 1-bit register determines the memory field from which operands are taken in indirectly-addressed AND, TAD, ISZ, or DCA instructions. Operating the LOAD ADDRESS key JAM transfers the contents of the DATA FIELD switch register on the operator console into the DF register. The DF is set by a transfer of information from bit 8 of the MB during a CDF microinstruction to establish a microprogrammed data field. When a program interrupt occurs, the content of the DF is automatically stored in the save field register. The DF is set by a transfer of information from bit 1 of the save field register by the RMF microinstruction to restore the data field at the conclusion of the program interrupt subroutine.

Instruction Buffer Register (IB): The IB serves as a 1-bit input buffer for the instruction field register. All field number transfers into the instruction field register are made through the instruction buffer, except transfers from the operator console switches. The IB is set by operation of the LOAD ADDRESS key in the same manner as the instruction field register. A CIF microinstruction loads the IB with the programmed field number contained in MB 8. An RMF microinstruction transfers the content of bit 0 of the save field register into the IB to restore the instruction field to the conditions that existed prior to a program interrupt.

Save Field Register (SF): When a program interrupt occurs, this 2-bit register is loaded from the instruction field and data field registers. The RMF microinstruction can be given immediately prior to the exit from the program interrupt subroutine to restore the instruction field and data field by transferring the content of the SF into the instruction buffer and the data field register. The SF is cleared during the cycle in which the program count is stored at address 0000 of the JMS instruction forced by a program interrupt request, then the instruction field and data field are strobed into the SF.

Break Field Register (BF): This 1-bit register receives the ADDRESS EXTEND signal from any I/O device using the data break facility. When the B SET signal arrives from the processor, this register is loaded with the bit combination of the three inputs.

Extended Address Signal Generator: When the PDP-8/L core memory capacity is extended, the standard memory is designated as field 0. This circuit produces the EXTEND ADDRESS FIELD 0 signal when data field 0 is selected, or instruction field 0 is selected. This circuit will produce the other EXTEND ADDRESS FIELD signal determined by the bit applied to its input.

Accumulator Transfer Gating: This gating allows the contents of the save field register, instruction field register, or the data field register to be strobed into the accumulator. A portion of the accumulator is transferred back upon itself so that certain AC bits are not changed when a memory control read function is executed. Other bits are unconditionally cleared (for program compatibility reasons). During an RIF or RDF instruction AC 0-5 and 9-11 are fed back by the computer, bits 6 and 7 are unconditionally cleared, and bit 8 is controlled by the appropriate field flip-flop. Similarly, an RIB instruction causes AC 0-5 to be fed back, AC 6-7 and 9-10 to be cleared, and AC 8 and 11 to be controlled by the SF.

Device Selector: Bits 3 through 5 of the IOT instruction are decoded to produce the IOT command pulses for the memory extension control. Bits 6 through 8 of the instruction are not used for device selection since they specify a field number in some commands. Therefore, the select code for this device selector is designated as 2X.

Instructions

The instructions for the Type MC 8/L option use the IOP generator and extend the IOT instruction list to include the following:

Change to Data Field N (CDF)

Octal Code: 62N1
Event Time: Not applicable
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
Operation: The data field register is loaded with the program-selected field number (N = 0 or 1). All subsequent memory requests for operands are automatically switched to that data field until the data field number is changed by a new CDF command, or during a program interrupt.
Symbol: MB8 => DF

Change Instruction Field (CIF)

Octal Code: 62N2
Event Time: Not applicable
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
Operation: The instruction buffer register is loaded with the program-selected field number (N = 0 or 1). The next JMP or JMS instruction causes the new field to be entered.
Symbol: MB8 => IB

Read Data Field (RDF)

Octal Code: 6214
Event Time: Not applicable
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
Operation: The content of the data field register is transferred into bit 8 of the AC. Bits 6 and 7 are cleared. All other bits of the AC are unaffected.

Symbol: DF => AC8

Read Instruction Field (RIF)

Octal Code: 6224
Event Time: Not applicable
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
Operation: The content of the instruction field register is transferred into bit 8 of the AC. Bits 6 and 7 are cleared. All other bits of the AC are unaffected.
Symbol: IF => AC8

Read Interrupt Buffer (RIB)

Octal Code: 6234
Event Time: Not applicable
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
Operation: The instruction field and data field held in the save field register during a program interrupt are transferred into bit 8, and 11 of the AC respectively.
Symbol: SF 0 => AC8
SF 1 => AC11

Restore Memory Field (RMF)

Octal Code: 6244
Event Time: Not applicable
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
Operation: This command is used upon exit from the program interrupt subroutine. The data and instruction fields that were interrupted by the subroutine are restored by transferring the content of the save field register into the instruction buffer and data field registers.
Symbol:

SF 0 => IB
SF 1 => DF

Programming

Instructions and data are accessed from the currently assigned instruction and data fields, where instructions and data may be stored in the same or different memory fields. When indirect memory references are executed, the operand address refers first to the instruction field to obtain an effective address, which in turn, refers to a location in the currently assigned data field. All instructions and operands are obtained from the field designated by the content of the instruction field register, except for indirectly-addressed operands which are specified by the content of the data field register. In other words, the DF is effective only in the Execute cycle that is directly preceded by the Defer cycle of a memory reference instructions, as follows:

Indirect (Bit 3)	Page or Z Bit (Bit 0)	Field In IF	Field In DF	Effective Address
0	0	m	n	The operand is in page 0 of field m at the page address specified by bits 5 through 11.
0	1	m	n	The operand is in the current page of field m at the page address specified by bits 5 through 11.
1	0	m	n	The absolute address of the operand in field n is taken from the content of the location in page 0 of field m designated by bits 5 through 11.
1	1	m	n	The absolute address of the operand in field n is taken from the content of the location in the current page of field m designated by bits 5 through 11.

Both fields of extended memory contain eight autoindex registers in addresses 10 through 17. For example, assume that a program in field 1 is running (IF = 1) and using operands in field 0 (DF = 0) when the instruction TAD I 10 is fetched. The Defer cycle is entered (bit 3 = 1) and the content of location 10 in field 1 is read, incremented, and rewritten. If address 10 in field 1 originally contained 4321, it now contains 4322. In the Execute cycle the operand is fetched from location 4322 of field 0.

Program control is transferred between memory fields by the CIF commands. The instruction does not change the instruction field directly, since this would make it impossible to execute the next sequential instruction. The CIF instruction sets the new instruction field into the IB for automatic transfer into the IF when either a JMP or JMS instruction is executed. The DF is unaffected by the JMP and JMS instructions. The 12-bit program counter is set in the normal manner and, since the IF is an extension on the most significant end of the PC, program sequence resumes in the new memory field following a JMP or JMS. Entry into a program interrupt is inhibited after the CIF instruction until a JMP or JMS is executed.

To call a subroutine that is out of the current field, the data field register is set to indicate the field of the calling JMS, which establishes the location of the operands as well as the identity of the return field. The instruction field is set to the field of the starting address of the subroutine. The following sequence returns program control to the main program from a subroutine that is out of the current field.

```
/PROGRAM OPERATIONS IN MEMORY FIELD 0
/INSTRUCTION FIELD = 0; DATA FIELD = 2
/CALL A SUBROUTINE IN MEMORY FIELD 1
/INDICATE CALLING FIELD LOCATION BY THE CONTENT OF THE DATA FIELD

          CIF      10          /CHANGE TO INSTRUCTION
                               /FIELD 1 = 6212
          JMS      1  SUBRP    /SUBRP = ENTRY ADDRESS
          CDF      0           /RESTORE DATA FIELD
SUBRP,    SUBR                 /POINTER
/CALLED SUBROUTINE
          0                    /SUBR = PC + 1 AT CALLING POINT
          RDF                  /READ DATA FIELD INTO AC
          TAD RETURN           /CONTENT OF THE AC = 6202 + DATA
                               /FIELD BITS
          DCA EXIT             /STORE INSTRUCTION SUBROUTINE
            .
            .
            .
            .
EXIT,     0                    /A CIF INSTRUCTION
          JMP I SUBR           /RETURN
RETURN,   CIF
```

When a program interrupt occurs, the current instruction and data field numbers are automatically stored in the 2-bit save field register, then the IF and DF are cleared. The 12-bit program count is stored in location 0000 of field 0 and program control advances to location 0001 of field 0. At the end of the program interrupt subroutine the RMF instruction restores the IF and DF from the content of the SF. The following instruction sequence at the end of the program interrupt subroutine continues the interrupted program after the interrupt has been processed:

```
.                    /RESTORE MQ IF REQUIRED
.
.
.
.                    /RESTORE L IF REQUIRED
.
.
.
CLA
TAD   AC             /RESTORE AC
RMF                  /LOAD IB FROM SF
ION                  /TURN ON INTERRUPT SYSTEM
JMP   I 0            /RESTORE PC WITH CONTENT OF
                     /LOCATION 0 AND LOAD IF FROM IB
```

A device using the computer data break facility supplies a 12-bit address to the MA and a 1-bit address extension to the Memory Extension Control Type MC 8/L. The address extension is received by a break field decoder which selects the memory field used for the data break.

MEMORY PARITY (MP-8/L) (The logic for this option is housed within the PDP-8/L central processor.)

Note: Two types of Memory Extension Control and Memory Module are available for use with PDP-8/L. The MC-8/L-A is designed for operation when the MP-8/L parity option is not installed. When MP-8/L is installed, the MC-8/L-B should be used.

Data transmission checking of each word written in and read from core memory is provided by this option. The option replaces the 12-bit core memory with a 13-bit system (driving, inhibiting, sensing circuits as well as a core array constructed of 13 planes) and includes a parity generator and a parity checking circuit. The parity generator produces the 13th bit for each 12-bit data word written in core memory so that the entire word contains an odd number of binary ones. The parity checking circuit monitors each word read from core memory to assure that the odd parity is maintained. If a word read contains an even number of ones a transmission error is indicated by setting a parity error flag. This flag is connected to the program interrupt synchronization element of the computer to initiate a program interrupt subroutine. This routine sequentially checks all equipment error flags to determine the option causing the interrupt and initiates an appropriate service and returns to the main program, or provides a suitable error printout and halts programmed operations. Upon determining that a memory parity error has occurred the program interrupt subroutine can repeat the main program step that caused the error to check the reliability of the error condition, can perform a simple write/read/check routine at the error address, or can determine the status of the machine when the error was detected and re-establish or print out these conditions and halt.

Instructions

Two instructions are associated with the Type PDP-8/L option. They are:

Skip on No Memory Parity Error (SMP)

Octal Code: 6101
Event Time: 1
Indicator: FETCH, IR = 6
Execution Time: 4.25 microseconds

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: 3
Indicator: FETCH, IR = 6
Execution Time: 4.25 microseconds
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 the memory parity option as described previously. The CMP command is used to initialize the memory parity option in preparation for normal programmed operation of the computer.

POWER FAILURE (KP-8/L) (The logic for this option is housed within the PDP-8/L central processor.)

This prewired option protects an operating program in the event of failure of the source of computer primary power. If a power failure occurs, this option causes a program interrupt and enables continued operation for 1 millisecond, allowing the interrupt routine to detect the power low condition as initiator of the interrupt, and to store the content of active registers (AC, L, etc.) and the program count in known core memory locations. When power is restored, the power low flag clears and a routine beginning in address 0000 starts automatically. This routine restores the content of the active registers and program counter to the conditions that existed when the interrupt occurred, then continues the interrupted program.

The KP8/L option consists of three logic circuits:

A power interrupt circuit monitors the status signal of the computer power supply, and sets a power low flag when power is interrupted (due to a power failure or due to the operation of the POWER lock on the operator console). This flag causes a program interrupt when an interruption in computer power is detected.

A restart circuit assures that when a power interrupt occurs the logic circuits of the computer continue operation for 1 millisecond to allow a program subroutine to store the content of the active registers; maintains the inoperative condition of the computer during periods of power fluctuation; and clears the power low flag and restarts the program when power conditions are suitable for computer operation. A manual RESTART switch enables or disables the automatic restart operation. With this switch in the ON (down) position, the option clears the program counter immediately and produces a signal to simulate operation of the START key on the operator console 200 milliseconds after power conditions are satisfactory. The MA is cleared so that operation restarts by executing the instruction in address 0000. This instruction must

be a JMP to the starting address of the subroutine which restores the content of the active registers and the program counter to the conditions that existed prior to the power low interrupt. The 200-millisecond delay assures that slow mechanical devices, such as Teletype equipment, have come to a complete stop before the program is resumed. Simulation of the manual START function causes the processor to generate a Power Clear pulse to clear internal controls and I/O device registers. With the RESTART switch in the OFF (up) position, the power low flag is cleared but the program must be started manually, possibly after resetting peripheral equipment or by starting the interrupted program from the beginning.

A skip circuit provides programmed sensing of the condition of the power low flag by adding the following instruction to the computer repertoire:

Skip on Power Low (SPL)
Octal Code: 6102
Event Time: 2
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds

Operation: The content of the power low flag is sampled, and if it contains a 1 (indicating a power failure has been detected) the content of the PC is incremented by one so the next sequential instruction is skipped.

Symbol: If Power Low flag = 1, then PC +1 => PC

Since the time that operation of the computer can be extended after a power failure is limited to 1 millisecond, the condition of the power low flag should be the first status check made by the program interrupt subroutine. The beginning of the program interrupt subroutine, containing the SPL microinstruction and the power fail program sequence can be executed in 27 microseconds on a basic PDP-8/L. The power fail program sequence stores the content of the active register and program count in designated core memory location, then relocates the calling instruction of the power restore subroutine to address 0000, as follows:

```
Address   Instruction        Remarks
-------   -----------        -------
0000          —              /STORAGE FOR PC AFTER PROGRAM INTERRUPT
0001      JMP FLAGS          /INSTRUCTION EXECUTED AFTER PROGRAM
                             /INTERRUPT
FLAGS,    SPL                /SKIP IF POWER LOW FLAG = 1
          JMP OTHER          /INTERRUPT NOT CAUSED BY POWER LOW,
                             /CHECK OTHER FLAGS
          DCA AC             /INTERRUPT WAS CAUSED BY POWER LOW,
                             /SAVE AC
          RAR                /GET LINK
          DCA LINK           /SAVE LINK
          TAD 0000           /GET PC
          DCA PC             /SAVE PC
          TAD RESTRT         /GET RESTART INSTRUCTION
          DCA 0000           /DEPOSIT RESTART INSTRUCTION IN 0000
          HLT
RESTRT    JMP ABCD           /ABCD IS LOCATION OF RESTART ROUTINE
```

Automatic program restart begins by executing the instruction stored in address 0000 by the power fail routine. This instruction must be a JMP (either direct or indirect). The power restore subroutine restores the content of the active registers, enables the program interrupt facility, and continues the interrupted program from the point at which it was interrupted, as follows:

Address	Instruction	Remarks
0000	JMP ABCD	
ABCD,	TAD LINK	/GET LINK
	CLL RAL	/RESTORE LINK
	TAD AC	/RESTORE AC
	ION	/TURN ON INTERRUPT
	JMP I PC	/RETURN TO INTERRUPTED PROGRAM

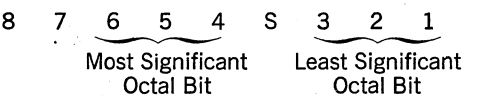
CHAPTER 7
INPUT/OUTPUT EQUIPMENT INSTRUCTIONS

TELETYPE AND CONTROL
Teletype Model 33 ASR

(The control circuitry for this device is located in the PDP-8/L central processor.)

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 = Teletype). This code is read in the reverse of the normal octal form used in the PDP-8/L 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 3 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 code of a teletype character is loaded into the TTI so that spaces correspond with binary zeros and holes (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 a binary one, and causes a program interrupt when an 8-bit computer character has been assembled in the TTI from a Teletype character. The program must sense the condition of this flag with a KSF microinstruction, and if the flag is set, issue 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: FETCH, IR = 6
Execution Time: 4.25 microseconds
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: FETCH, IR = 6
Execution Time: 4.25 microseconds

Operation: The AC is cleared in preparation for another microinstruction to transfer a character from the TTI into the AC. The keyboard flag is also cleared, this allows the hardware to begin assembling the next input charac-

ter in the TTI. If there is tape in the reader and the reader is on, the character over the read head will be loaded into the TTI and the tape advanced one frame. If there is no tape or the reader is turned off (STOP or FREE) the character struck on the keyboard will be assembled into the TTI. In either case, when the character is completely assembled in the TTI the hardware causes the keyboard flag to be set to a binary 1.

Symbol: 0 = > AC
0 = > Keyboard flag allowing the hardware to cause:
Keyboard/Tape Character = > TTI
1 > Keyboard flag when done

Read Keyboard Buffer Static (KRS)

Octal Code: 6034
Event Time: 3
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
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: FETCH, IR = 6
Execution Time: 4.25 microseconds
Operation: This microinstruction combines the functions of the KCC and KRS. The AC and keyboard flag are both cleared and the content of the TTI is transferred into bits 4-11 of the AC. Clearing the keyboard flag allows the hardware to begin assembling the next input character into the TTI (as discussed with the KCC). When the character is completely assembled in the TTI, the hardware causes the flag to be set indicating it again has a character ready for transfer.

Symbol: 0 = > AC C(TTI) V C(AC 4-11) = > AC 4-11
0 = > Keyboard Flag allowing the hardware to cause:
Keyboard/Tape Character = > TTI
1 = > Keyboard flag when done.

The following are examples of possible sequences of instruction to read a character into the AC from the teletype:

```
        .
        .
LOOK,   KSF          /SKIP IF FLAG = 1
        JMP LOOK     /JMP BACK & TEST FLAG AGAIN
        KRB          /TRANSFER TTI CONTENTS INTO AC
```

This sequence waits for the TTI to set its flag, indicating that it has a character ready to be transferred. It then skips to the KRB command which causes the character to be read into the AC from the TTI.

By making this sequence of instructions a subroutine of a larger program, it can be accessed each time an input character is desired.

```
READ,   KCC              /CLEAR TTI FLAG
        .
        .
        .
        0                /STORE DC HERE FOR RETURN ADDRESS
        KSF              /SKIP IF /FLAG = 1
        JMP.-1           /TEST FLAG AGAIN
        KRB              /READ CHARACTER INTO AC
        JMP I READ       /EXIT TO MAIN PROGRAM
```

Teleprinter/Punch

On program command a character is sent in parallel from the accumulator (AC) to the TTO shift register for transmission to the teleprinter/punch unit. The control generates the start space, then shifts the eight character bits serially into the printer selector magnets of the teletype unit, and then generates the stop marks. This transfer of information from the TTO into the teleprinter/punch unit is accomplished at the normal teletype rate and requires 100 milliseconds for completion. The flag in the teleprinter control is again set to a 1 when the last of the character 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 instruction skip element. Unless using the interrupt, the program must check the flag and, upon detecting the ready or set (binary 1) condition of the flag by means of the TSF microinstruction, the program must issue a TLS microinstruction which clears the flag and sends a new character from the AC to the TTO to be shifted out to the teleprinter/punch. The process of sending a character to the TTO from the AC is a great deal shorter than that of shifting the character out to the teleprinter/punch, therefore, the program must account for the time differential by waiting for flag to be set (1) before issuing a TLS.

Instructions for use in outputting data to the teletype are as follows:

Skip on Teleprinter Flag (TSF)

Octal Code: 6041
Event Time: 1
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
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: FETCH, IR = 6
Execution Time: 4.25 microseconds
Operation: The teleprinter flag is cleared to 0.
Symbol: 0 = > Teleprinter Flag

Load Teleprinter and Print (TPC)

Octal Code: 6044
Event Time: 3
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
Operation: The contents of bits 4-11 of the AC are sent to the TTO, then the hardware starts shifting the character out to the printer/punch unit. This microinstruction does not clear the teleprinter flag.

Symbol: C(AC 4-11) = > TTO causing:
C(TTO) = > printed and (if punch on) punched

Load Teleprinter Sequence (TLS)

Octal Code: 6046
Event Time: 2, 3
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds

Operation: This microinstruction combines the functions of the TCF and the TPC. The teleprinter flag is cleared (set to 0) then the contents of bits 4-11 of the AC are sent to the TTO, where the hardware shifts the character out to the printer/punch unit. When the printer/punch has finished outputting the character and is ready for another character, the hardware has again raised the teleprinter flag (set it to a 1) to indicate this free condition. The whole operation, from the time at which the TLS has cleared the flag and sent out the character until the time at which the hardware finishes with the character and sets the flag to a 1 again, requires 100 milliseconds with the time required for the character to travel from the TTO to the paper being considerably greater than that required for it to be sent from the computer to the TTO.

Symbol: 0 = > Teleprinter flag
C(AC 4-11) = > TTO causing:
C(TTO) = > Printed and (if punch on) punched
1 = > Teleprinter flag when done

The following are examples of possible ways to use these instructions to output a character to the teletype. The last is recommended:

```
        .
        .
        .
        CLA
        TAD X           /PUT CHARACTER CODE INTO AC FROM
                        /LOCATION X
        TLS             /LOAD TTO FROM AC & PRINT/PUNCH
FREE,   TSF             /TEST FLAG TO SEE IF DONE PRINTING,
                        /SKIP IF = 1
        JMP FREE        /TEST FLAG AGAIN
        CLA             /CLEAR CHARACTER CODE FROM AC
        .
        .
        .               continue program
```

This sequence sends one character code to the TTO and waits for it to finish printing/punching before continuing program. It does not require that the flag to be set, in order to output the character. By making this sequence of instructions a subroutine of a larger program, it can be accessed (by a JMS) each time a character is to be output. Assume that the subroutine is entered with the character code in the AC:

```
TYPE,   0
        TLS             /LOAD TTO FROM AC AND PRINT/PUNCH
        TSF             /TEST FLAG, SKIP IF = 1
        JMP.-1          /JMP BACK & TEST FLAG AGAIN
        CLA             /CLEAR CHARACTER FROM AC
        JMP I TYPE      /EXIT TO MAIN PROGRAM
        .
        .
        .
```

By rearranging this subroutine the present time spent waiting for the character to be output and the flag to be set to 1 (100 milliseconds) can be used to continue the calculations, etc., of the main program thus making more efficient use of time.

```
TYPE,   0
        TSF             /TEST FLAG TO SEE IF PRINTER FREE,
                        /SKIP IF YES OR . . .
        JMP.-1          /WAIT TIL IT IS BY TESTING AGAIN AND
                        /AGAIN
        TLS             /OUTPUT CHARACTER
        CLA
        JMP I TYPE      /EXIT TO CONTINUE PROGRAM
        .
        .
        .
```

This subroutine tests the flag first and waits only if a previous character is still being output. It clears the AC and exits immediately after sending the character to the TTO and is continuing to run the user's program instead of waiting while the teletype (a much slower device) is off typing/punching the last character. The PDP-8/L clears all flags which are on the clear flag bus (this includes teletype flags) when key START is depressed. This means that the user program must account for setting the teleprinter flag initially and after each TCF if (any) or else the program will hang up in the wait loop of the print routine. The only way to set the flag to a 1 is through issuing a microinstruction which leaves the flag set when alone. This instruction should appear among the first few executed and must appear before any attempt to output a character.

The following example initializes the flag with a TLS as the first instruction of the program and makes optimum use of the time that would be spent waiting for the teletype to finish.

```
BEGIN,    TLS              /INITIALIZE TELEPRINTER FLAG
          .
          .
          .
TYPE,     0
          TSF              /SKIP IF FLAG = 1 or . . .
          JMP.-1           /WAIT UNTIL IT IS LOAD TTO &
          TLS              /TYPE CHARACTER
          CLA
          JMP I TYPE       /EXIT & CONTINUE PROGRAM WHILE
          .                /TELETYPE IS FINISHING CHARACTER
          .
          .
```

TELETYPE OPTION (TYPE PT08)

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 with the PT08 option. A PT08 option allows a Teletype to be interfaced to the PDP-8/L. Each Teletype line added contains logic elements that are functionally identical to those of the basic Teletype control. Therefore, instructions and programming for each PT08 are similar to those described previously for the basic Teletype unit. The following device select codes have been assigned for 5 PT08 options.

Line Unit	Select Codes
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 PT08 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 "PT" and the line number. For example, the following instructions can be defined for line 3:

Mnemonic	Octal	Operation
TSFPT3	6441	Skip if teleprinter 3 flag is a 1.
TCPPT3	6442	Clear teleprinter 3 flag.
TPCPT3	6444	Load teleprinter 3 buffer (TT03) from the content of AC4-11 and print and/or punch the character.
TLSPT3	6446	Load TT03 from the content of AC4-11, clear teleprinter 3 flag, and print and/or punch the character
KSFPT3	6451	Skip if keyboard 3 flag is a 1.
KCCPT3	6452	Clear AC and clear keyboard 3 flag.
KRSPT3	6454	Read keyboard 3 buffer (TTI3) static. The content of TTI3 is loaded into AC4-11 by an OR transfer.
KRBPT3	6456	Clear the AC, clear keyboard 3 flag, and read the content of TTI3 into AC4-11.

HIGH-SPEED PERFORATED TAPE READER AND CONTROL (TYPE PR8/L)

(The control circuitry for this device is located in the PDP-8/L central processor.)

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: FETCH, IR = 6
Execution Time: 4.25 microseconds
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: FETCH, IR = 6
Execution Time: 4.25 microseconds
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 V AC 4-11 = > AC 4-11
0 = > Reader Flag

Reader Fetch Character (RFC)

Octal Code: 6014
Event Time: 3
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
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
```

HIGH-SPEED TAPE PUNCH AND CONTROL (TYPE PP8/L)

(The control circuitry for this device is located in the PDP-8/L central processor.)

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 as described for the Teletype unit. The punch instructions are:

Skip on Punch Flag (PSF)

Octal Code: 6021
Event Time: 1
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
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: FETCH, IR = 6
Execution Time: 4.25 microseconds
Operation: Both the punch flag and the punch buffer are cleared in preparation for receiving a new character from the computer.
Symbol: 0 = > Punch Flag, PB

Load Punch Buffer and Punch Character (PPC)

Octal Code: 6024
Event Time. 3
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
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: FETCH, IR = 6
Execution Time: 4.25 microseconds
Operation: The punch flag and punch buffer are both cleared, the content 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
```

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 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-buffered 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: FETCH, IR = 6
Execution Time: 4.25 microseconds
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_8 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 ±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 megahertz.

OSCILLOSCOPE DISPLAY CONTROL (TYPE VC8/I)

The oscilloscope available for use with the Type VC8/I Control is a Tektronix Oscilloscope Model RM503 with 10 bits per axis.
Type VC8/I control is a two axis digital-to-analog converter and an intensifying circuit, which provides the Deflection and Intensify signals needed to plot data on an oscilloscope. Coordinate data is loaded into an "X" buffer (XB) or a "Y" buffer (YB) from bits 2 through 11 of the accumulator. The binary data in these buffers is converted to a 0 to +2 volt Analog Deflection signal. The Intensify signal is connected to the grid of the oscilloscope CRT. The voltage of this signal, and hence the intensity of the point displayed, is determined by a 2-bit brightness register (BR). The content of the BR controls circuits that establish nominal voltages of −7, −13, or −23 volts for the Intensify signal. The BR is loaded from a number contained in the appropriate IOT instruction. Application of power to the computer or pressing of the START key resets the BR to the maximum brightness. Points can be plotted at approximately a 30 kilohertz rate. The instructions for this display are:

Clear X Coordinate Buffer (DCX)

Octal Code: 6051
Event Time: 1
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
Operation: The X coordinate buffer is cleared in preparation for receiving new X-axis display data.
Symbol: 0 => XB

Clear and Load X Coordinate Buffer (DXL)

Octal Code: 6053
Event Time: 1, 2
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
Operation: The X coordinate buffer is cleared, then loaded with new X-axis data from bits 2 through 11 of the AC.
Symbol: 0 => XB
AC2-11 => XB

Clear Y Coordinate Buffer (DCY)

Octal Code: 6061
Event Time: 1
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
Operation: The Y coordinate buffer is cleared in preparation for receiving new Y-axis display data.
Symbol: 0 = > YB

Clear and Load Y Coordinate Buffer (DYL)

Octal Code: 6063
Event Time: 1, 2
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
Operation: The Y cordinate buffer is cleared then loaded with new Y-axis data from bits 2 through 11 of the AC.
Symbol: 0 = > YB
AC 2-11 = > YB

Intensify (DIX)

Octal Code: 6054
Event Time: 3
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
Operation: Intensify the point defined by the content of the X and Y coordinate buffers. This command can be combined with the DXL command.
Symbol: None

Intensify (DIY)

Octal Code: 6064
Event Time: 3
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
Operation: Intensify the point defined by the content of the X and Y coordinate buffers. This command is identical to the DIX command except that it can be combined with the DYL command.
Symbol: None

X Coordinate Sequence (DXS)

Octal Code: 6057
Event Time: 1, 2, 3
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
Operation: This command executes the combined functions performed by the DXL and DIX commands. The X coordinate buffer is cleared then loaded from the content of AC2 through AC11, then the point defined by the content of the X and Y buffers is intensified.
Symbol: 0 = > XB
AC 2-11 = > XB
then intensify

Y Coordinate Sequence (DYS)

Octal Code: 6067
Event Time: 1, 2, 3
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
Operation: This command executes the combined functions performed by the DYL and DIY commands. The Y coordinate buffer is cleared, then loaded from the content of bits AC2 through 11, then the point defined by the content of the X and Y coordinate buffers is intensified.
Symbol: 0 => YB
AC 2-11 => YB
then intensify

Set Brightness Control (DSB)

Octal Code: 607X
Event Time: 3
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
Operation: The brightness register (BR) is loaded from the content of bits 10 and 11 of the instruction. When the instruction is 6075 the minimum brightness is set, when 6076 the medium brightness is set, and when 6077 the maximum brightness is set. 6074 Instruction sets zero brightness.
Symbol: MB10-11 => BR

The following program sequence to display a point assumes that the coordinate data is stored in known addresses X and Y.

```
X,
Y,
BEG,   CLA
       TAD X    /LOAD AC WITH X
       DXL      /CLEAR AND LOAD XB
       CLA
       TAD Y    /LOAD AC WITH Y
       DYS      /CLEAR AND LOAD YB, DISPLAY POINT
```

INCREMENTAL PLOTTER AND CONTROL (TYPE VP8/I)

Four models of California Computer Products Digital Incremental Recorder can be operated from a Digital Type VP8/I Incremental Plotter Control. Characteristics of the four recorders are:

CCP Model	Step Size (inches)	Speed (steps/minute)	Paper Width (inches)
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 each of the four 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 incremen-

tal 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: FETCH, IR = 6
Execution Time: 4.25 microseconds
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: FETCH, IR = 6
Execution Time: 4.25 microseconds
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: FETCH, IR = 6
Execution Time: 4.25 microseconds
Operation: The plotter pen is raised from the surface of the paper.
Symbol: None

Pen Right (PLPR)

Octal Code: 6511
Event Time: 1
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
Operation: The plotter pen is moved to the right in either the raised or lowered position.
Symbol: None

Drum Up (PLDU)

Octal Code: 6512
Event Time: 2
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
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: FETCH, IR = 6
Execution Time: 4.25 microseconds
Operation: The plotter paper drum is moved downward.
Symbol: None

Pen Left (PLPL)

Octal Code: 6521
Event Time: 1
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
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: FETCH, IR = 6
Execution Time: 4.25 microseconds
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: FETCH, IR = 6
Execution Time: 4.25 microseconds
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/L can track the location of the pen on the paper by counting the instructions that increment position of the pen and the drum.

CARD READER AND CONTROL (TYPE CR8/I)

The Card Reader and Control Type CR8/I reads 12 row, 80 column punched cards at a nominal rate of 200 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 photo-electrically sensed. 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 3 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 1.4 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 immediately 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 check for data loss. The reader-not-ready flag can be sensed by a skip command to provide indication of card reader power off, pick failure, a dark check indication, a stacker failure, hopper empty, stacker full, Sync failure or light check indication. 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 CR8/I are:

Skip on Data Ready (RCSF)

Octal Code: 6631
Event Time: 1
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
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: FETCH, IR = 6
Execution Time: 4.25 microseconds
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: FETCH, IR = 6
Execution Time: 4.25 microseconds
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 AC11.
Symbol: AC V Binary Code = > AC
0 = > Data Ready Flag

Skip on Card Done Flag (RCSD)

Octal Code: 6671
Event Time: 1
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
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: FETCH, IR = 6
Execution Time: 4.25 microseconds
Operation: The status of the card reader is sensed.
If the reader is ready, the PC is incremented to skip the next sequential instruction, a card is started toward the read station from the input hopper and the card done flag is cleared.
Symbol: If reader flag = 1, then PC + 1 + > PC
0 = > Card Done Flag

Clear Card Done Flag (RCRD)

Octal Code: 6674
Event Time: 3
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
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
         JUMP NOT RDY    /JUMP TO SUBROUTINE THAT TYPES OUT
                         /"CARD READER MANUAL INTERVENTION
                         /REQUIRED" OR HALTS
NEXT,    RCSF            /DATA READY?
         JMP DONE        /NO, CHECK FOR END OF CARD
         RCRA or RCRB    /YES, READ ONE CHARACTER OR ONE
                         /COLUMN AND CLEAR DATA READY
         DCA 1 STR       /STORE DATA
DONE,    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 checking is performed by the CR8/I. A programmed validity check can be made by reading each card column in both the aphanumeric and binary mode (within the 1.4 millisecond time limitation), then performing a comparison check.

Before commencing a card reading program, load the input hopper with cards and press Motor Start and Read Start pushbuttons. The function of the manual controls and indicators are as follows (as they appear from left to right on the card reader):

Control or Indicator	Function
A — POWER switch	On-Off toggle switch. Applies power to all circuits except drive motor.
B — MOTOR START	Momentary action pushbutton, with separate indicator. Applies power to main drive motor. Motor start is also used as a reset to clear error indicators and therefore will not operate if there is an unremedied condition such as: 1. Input hopper is empty. 2. Output hopper is full. 3. All photo cells are not lit. 4. Internal power supplier is not operational.
C — READ START	Momentary action pushbutton, with separate indicator. Causes ready line to go high, which enables card reading under control of the external read command. If read command is open or high, card reading begins immediately at full rated speed.
D — READ STOP	Momentary action pushbutton, with separate indicator. Inhibits further card reading until READ START switch is pressed again. Ready line goes low and READ STOP condition is indicated. Does not stop drive motor. However, a READ STOP condition is indicated anytime the drive motor is stopped.
E — INDICATORS	Several detection circuits are incorporated in the card reader. Whenever any red indicator lights, the drive motor is stopped after allowing the completion of the current card cycle.
1. PICK FAIL Indicator	Lights when a card fails to enter the read station after two successive pick attempts.
2. DARK CHECK Indicator	After the card enters the read station, a check is made at the hypothetical 0th and 81st hole positions to be sure all photocells are dark. If not, the DARK CHECK indicator lights and data outputs are inhibited immediately.
3. STACKER FAIL Indicator	When three cards have passed the read station and none have been stacked, a STACK FAIL is indicated. Prevents more than three cards from being in the track at once.
4. HOPPER EMPTY Indicator	Indicates input hopper is empty.
5. STACKER FULL Indicator	Switch closure detects when approximately 400 cards are in the stacker hopper.

Control or Indicator	Function
6. SYNC FAIL Indicator	Internal timing signals are derived from an oscillator which is sync'ed to the track speed. If the sync signal is lost, a SYNC FAIL is indicated.
7. LIGHT CHECK Indicator	All photo cells must always be lit except during the time a card is being read. The detector is inhibited each time a card enters the read station until position (count of) 84. If a card fails to leave the read station by this time, a LIGHT CHECK is indicated.

RANDOM ACCESS DISK FILE (TYPE DF32)

The Type DF32 Disk File is a fast, low-cost, random-access, bulk-storage device and control for the PDP-8/L. Operating through the 3-cycle data-break channel, the DF32 provides 32,768 13-bit words (12 bits plus parity) of storage, and is economically expandable to 131,072 using Expander Disc Type DS32.

Transfer rate of the DF32 is 66 μsec per word; average access time is 16.67 msec for 60-cycle power (20 msec with 50-cycle power).

Two basic assemblies comprise the DF32: the storage unit with read/write electronics, and computer interface logic. The storage unit contains a nickel-cobalt plated disc driven by a hysteresis synchronous motor. Data is recorded on a single disc surface by 16 read/write heads which are in a fixed position. A photo-reflective marker is used on the disc's outer perimeter to denote beginning and end of timing and address tracks.

Disk motor and shaft, read/write data heads, timing and address heads, and photocell assembly are mounted on a rack assembly which permits sliding the unit in and out of a standard Digital Equipment Corporation cabinet.

The disk is designed for rack mounting in a 19 inch relay rack.

INSTRUCTIONS

The command for the disk system are as follows:

Clear Disk Memory Address Register (DCMA)

Octal Code: 6601
Event Time: 1
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
Operation: Clears Memory Address Register, parity error and completion flags. This instruction clears the disk memory request flag and interrupt flags.
Symbol: 0 = > completion flag
0 = > error flag

Load Disk Memory Address Register and Read (DMAR)

Octal Code: 6603
Event Time: 1, 2
Indicators: FETCH, IR = 6
Execution Time: 42 microseconds
Operation: The contents of the AC are loaded into the disk memory address register and the AC is cleared. Begin to read information from the disk into the specified core location. Clears parity error and completion flags. Clears interrupt flags.
Symbol: $AC_{0\text{-}11} \rightarrow DMA_{0\text{-}11}$
$0 = >$ completion flag
$0 = >$ error flag

Load Disk Memory Address Register and Write (DMAW)

Octal Code: 6605
Event Time: 1, 3
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
Operation: The contents of the AC are loaded into the disk memory address register and the AC is cleared. Begin to write information into the disk from the specified core location. Clears parity error and completion flags. Clears interrupt flags. Data break must be allowed to occur within 66 μsec after issuing this instruction.
Symbol: $AC_{0\text{-}11} \rightarrow DMA_{0\text{-}11}$
$0 = >$ completion flag
$0 = >$ error flag

Clear Disk Extended Address Register (DCEA)

Octal Code: 6611
Event Time: 1
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
Operation: Clears the Disk Extended Address and memory address extension register.
Symbol: $0 = >$ Disk Extended Address Register
$0 = >$ Memory Address Extension Register

Skip on Address Confirmed Flag (DSAC)

Octal Code: 6612
Event Time: 2
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
Operation: Skips next instruction if address confirmed Flag is a 1. Flag is set for 16 μsec (AC is cleared)
Symbol: If address confirmed flag = 1, then
PC + 1 = > PC

Load Disk Extended Address (DEAL)

Octal Code: 6615
Event Time: 1, 3
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
Operation: The Disk extended address and memory address extension registers are cleared and loaded with the track address data held in the AC.
Symbol: $AC_{6-8} = >$ Core Memory Extension
$AC_{1-5} = >$ Disk Address Extension 32K, 64K, 96K, 128K
$AC_{0,\ 9-11}$ used in DEAC instruction
(SEE NOTE)

Read Disk Extended Address Register (DEAC)

Octal Code: 6616
Event Time: 2, 3
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
Operation: Clear the AC then loads the contents of the disk extended address register into the AC to allow program evaluation. Skip next instruction if address confirmed flag is a 1.
Symbol: Disk Address Extension 32K, 64K, 96K, 128K $= > AC_{1-5}$
Core Memory Extension $= > AC_{6-8}$
Photo-cell sync mark $= > AC_0$ (Available 200 μsec)
Data Request Late flag $= > AC_9$
Non-existent or write Lock switch on $= > AC_{10}$
Parity Errors $= > AC_{11}$
(SEE NOTE)

Skip Old Zero Error Flag (DFSE)

Octal Code: 6621
Event Time: 1
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
Operation: Skips next instruction if partial error, data request late, or write lock switch flag is a zero. Indicates no errors.
Symbol: If parity error flag = 1, then PC + 1 = > PC
If Data Request late flag = 1, then PC + 1 = > PC
If Write lock switch Flag = 1, then PC + 1 = > PC

Skip on Data Completion Flag (DFSC)

Octal Code: 6622
Event Time: 2
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
Operation: Skips next instruction if the completion flag is a 1. Indicates data transfer is complete.
Symbol: If completion flag = 1, PC + 1 = > PC

Read Disk Memory Address Register (DMAC)

Octal Code: 6626
Event Time: 2, 3
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
Operation: Clears the AC then loads contents of the Disk Memory Address Register into the AC to allow program evaluation. During read, the final address will be the last one transferred.
Symbol: $DMA_{0-11} = > AC_{0-11}$

NOTE: For the DEAL and DEAC Instructions, refer to the diagrams shown below:

BITS 1-5 (DEAL INST)	ACCUMULATOR (LOW ORDER 12 BITS) 0-11 of DMAW or DMAR	DISC ADDRESS (17 BIT)

FIELD BITS 6-8 (DEAL INST)	CELL 7751 (CURRENT ADDRESS)	CURRENT ADDRESS (MEMORY) ADDRESS (15 BIT)

The computer can handle 12 bits, therefore, the high order bits for Disk and Memory addresses are manipulated by the DEAL and DEAC instructions. Low order bits are manipulated in the accumulator (AC).

The Disk address is a 17 bit value. Bit 1 of the DEAL and DEAC instructions is the most significant bit. The Memory Address is a 15 bit value. Bit 6 of the DEAL and DEAC instructions is the most significant bit.

Note that the Word Count 7750 is the 2's complement of the number of words to be transferred and that the Disk Address is the desired starting address. The memory or current address (7751) is the desired address —1.

Note: Write lock switch status is true only when disc module contains write command. The non-existent disc condition will appear following the completion of a data transfer during read, where the address acknowledged was the last address of a disc and the next word to be addressed falls within a non-existent disc. The completion flag for this data transfer is set by the non-existent disc condition 16 microseconds following the data transfer.

SOFTWARE

DF32 Disk System, available with PDP-8/L, is a fast convenient keyboard oriented monitor which will enable the user to efficiently control the flow of programs through his PDP-8/L. This system is modular and open ended, allowing the user to build the software components required in his environment. The user may specify the system device (Disc or DECtape), the amount of core, number of discs available and the number, name and size of his resident system programs.

GENERAL PURPOSE MULTIPLEXED ANALOG-TO-DIGITAL CONVERTER SYSTEM (TYPE AF01A)

The Type AF01A General-Purpose Multiplexed Analog-to-Digital Converter combines a versatile, multi-purpose 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. The Type AF01A option is used with the PDP-8/L to multiplex up to 64-analog signals and to convert the signals to binary numbers. Analog data on each of 64

channels can be accepted and converted into 12-bit digital numbers 420 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 1420 times each second.**

Switching point accuracy in this case is 98.4 per cent, again with the additional quantization error of half the digital value of the LSB.

*Conversion rate $= [(35 + 2)(10^{-6})(64)]^{-1} = 420$ cycles/sec.
**Conversion rate $= [(9 + 2)(10^{-6})(64)]^{-1} = 1420$ cycles/sec.

A/D CONVERTER SPECIFICATIONS

The Type AF01A has 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. The characteristics of the A/D converter are as follows:

Accuracy and Conversion Times:	See Table 2 (includes all linearity and temperature errors)
Converter Recovery Time:	Zero
Input and Input Impedance:	0 to −10V at 10 megohms standard. Input scaling may be specified using the amplifier or sample and hold options (see Table 1)
Input Loading:	±1 μA and 125 pf for 0 to −10V input signal.
Output:	Binary number of 6 to 12 bits, with negative numbers represented in 2's complement notation. A 0V input gives a 4000_8; a −5V input a 0000_8 and a −10V (minus 1 LSB*) input gives 3777_8 number.

Provision is made for using the Type A400 Sample and Hold Amplifier (AH02 option) between the multiplexer output and A/D converter input to reduce the effective aperture to less than 150 nsec. The Type A400 may also be used to scale the signal input to accept ±10v, ±5v, or 0 to +10v. The Type A200 amplifier (AH03 option) may be substituted for the Type A400 to accomplish the same signal scaling without reducing the effective aperture. Both the AH02 and AH03 options may be used to obtain high input impedance and small aperture.

MULTIPLEXER SPECIFICATIONS

The multiplexer can include from 1 to 16 Type A121 Switch Modules. Each module contains four single-pole, high speed, insulated gate FET switches with appropriate gating. The Type A121 Switches are arranged as a 64-channel group of series-switch single-pole switches with a separate con-

tinuous ground wire for each signal input. The switched signal input wire and the continuous ground for each channel are run as twisted pairs to the input connectors mounted on the rear panel. The continuous grounds for all channels are terminated at the high quality ground of the AF01A System. Specifications (measured at input connector) are as follows:

Input Operating Signal Voltages:	+10V to −10V
Current:	1 mA
On Resistance	450 ohm (max)
Voltage Offset	0
"Off Leakage"	10 nA (max)
Capacitance	10 pf (max)
Speed	
10% Input to within 1 LSB* of output	2 μs
Operate Time	The time required to switch from one channel to another is 2 μs to within 1 LSB* of the final voltage. This time is preset within the control and starts when a set or index command is received.

OPERATION

The Type AF01 System may be operated in either the random or sequential address modes. In the random address mode, the control routes the analog signal from any selected channel to the A/D converter input. In the sequential address mode, the multiplexer control advances its channel address by one each time an index command is received. After indexing through the maximum number of channels implemented, the address is returned to 0. When using sequential operation, the conditioning levels for random addressing are ignored.

The multiplexer switch settling time is preset within the control to initiate the conversion process automatically after a channel has been selected in either the random or sequential address mode. Two separate A/D Convert I/O Transfer Commands may also initiate one or more conversions on a currently selected channel.

A/D conversion times are increased by 2 μsec when multiplexer channels are switched to allow for settling time of the analog signal at the multiplexer output. Conversion times are increased an additional 3 μsec when AH03 is used. These times are added to the conversion times shown in Table 2 under selected channel conversion time, which is the only time required for each successive conversion on a selected channel.

When the Type AH02 Sample and Hold option is required, the multiplexer switch settling time and the sample and hold acquisition time are overlapped. The total conversion and switching time is increased by 10 μsec. (See A400 specifications.)

*LSB—Least Significant Bit.

A/D CONVERTER/MULTIPLEXER CONTROLS

DESIGNATION	FUNCTION
WORD LENGTH:	Rotary switch used to select digital word length or conversion accuracy. Refer to Table 2 for corresponding conversion times.
POWER ON/OFF:	Applies 117 Vac power to internal power supplies.
CLR:	Clear multiplexer channel-address registers; i.e., selects analog channel 0 for conversion.
INDEX:	Advances multiplexer channel-address register by one each time it is depressed, enabling manual addressing of channels (up to 64) in sequential mode. Returns address to zero when maximum value is reached.
ADC:	Starts conversion of the analog voltage on the selected channel to a binary number when depressed.
A/D CONVERTER:	Indicates binary contents of A/D converter register.
MULTIPLEXER:	Indicates binary contents of multiplexer channel-address register.
POWER:	Indicates ON/OFF status.

TABLE 3. INPUT SIGNAL SCALING

CONFIGURATION	GAIN	INPUT SIGNAL	INPUT IMPEDANCE	BINARY OUTPUT	OPTION DESIGNATION
Standard		0	10 meg.	4000_8	STD
		−5	10 meg.	0000_8	
		−10	10 meg.	3777_8	
Sample & Hold	−1	+5	10 K	3777_8	AHO2
	−1	0	10 K	0000_8	
	−1	−5	10 K	4000_8	
Sample & Hold	−½	+10	10 K	3777_8	AHO2
	−½	0	10 K	0000_8	
	−½	−10	10 K	4000_8	
Amplifier	+1	+5	>100 meg.	4000_8	AHO3
	+1	0	>100 meg.	0000_8	
	+1	−5	>100 meg.	3777_8	
Amplifier	+½	+10	>100 meg.	4000_8	AHO3
	+½	0	>100 meg.	0000_8	
	+½	−10	>100 meg.	3777_8	
Amplifier and Sample & Hold	−1 or −½	+5 or 0 −5	+10 or 0 −10	>100 meg. >100 meg. >100 meg.	3777_8 0000_8 4000_8	AHO3 & AHO2

Note: Unipolar signals (0 to +5, or 0 to +10v) may also be specified with either the AHO3 or AHO2 option.

TABLE 4. SYSTEM CONVERSION CHARACTERISTICS**

WORD LENGTH (NO. OF BITS)	MAX SWITCHING POINT ERROR*	SELECTED CHANNEL (A/D) CONVERSION TIME (μSEC)	RANDOM OR SEQUENTIAL (MPX. & A/D) CONVERSION TIME (μSEC)**	W/AHO3 AMP. (MPX. & A/D) CONVERSION TIME (μSEC)**	W/AHO2 SAMPLE & HOLD (MPX. & A/D) CONVERSION TIME (μSEC)**	W/AHO3 & AHO2 (MPX. & A/D) CONVERSION TIME (μSEC)**
6	±1.6%	9.0	11.0 (9.5)	14.0 (11.0)	19.0 (14.0)	21.0 (18.0)
7	±0.8%	10.5	12.5 (11.0)	15.5 (12.5)	20.5 (15.5)	22.5 (19.5)
8	±0.4%	12.0	14.0 (12.5)	17.0 (14.0)	22.0 (17.0)	24.0 (21.0)
9	±0.2%	13.5	15.5 (14.0)	18.5 (15.5)	23.5 (18.5)	25.5 (22.5)
10	±0.1%	18.0	20.0 (18.5)	23.0 (20.0)	28.0 (23.0)	30.0 (27.0)
11	±0.05%	25.0	27.0	30.0	35.0	37.0
12	±0.025%	35.0	37.0	40.0	45.0	47.0

*±½ LSB for quantizing error.
**If system is to operate at less than 10 bits continuously, conversion times may be reduced to times shown in parentheses.

Programming

Programmed control of the converter/multiplexer by the PDP-8/L is accomplished with the IOT instructions listed below. PDP-8/L selects the converter/multiplexer with two device selection codes, depending upon whether conversion of multiplexing functions are being selected; 53_8 and 54_8. The converter/multiplexer interprets the device selection code to enable execution of the IOP command pulse generated by the IOT instruction.

Skip on A-D Flag (ADSF)

Octal Code: 6531
Event Time: 1
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
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: FETCH, IR = 6
Execution Time: This time is a function of the accuracy and word length switch setting as listed in Table 2.
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: FETCH, IR = 6
Execution Time: 4.25 microseconds
Operation: The converted number contained in the converter buffer (ADCB) is transferred into the AC left justified; unused bits of the AC are left in a clear state, and the A-D converter flag is cleared. This command must be preceded by a CLA instruction.
Symbol: ADCB = > AC
0 = > A-D Converter Flag

Clear Multiplexer Channel (ADCC)

Octal Code: 6541
Event Time: 1
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
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: FETCH, IR = 6
Execution Time: 4.25 microseconds
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 input channels can be used.
Symbol: AC 6-11 = > CAR

Increment Multiplexer Channel (ADIC)

Octal Code: 6544
Event Time: 3
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
Operation: The content of the channel address register of the multiplex 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

The converter/multiplexer may be operated by the computer program in either the random or sequential addressing mode. In the random addressing mode, the analog channel is selected arbitrarily by the program for digitizing and the resultant binary word is read into the accumulator. A sample program for the random addressing mode is as follows:

```
TAD ADDR        /YES — GET CHANNEL ADDRESS
ADSC            /AND SEND TO MULTIPLEXER
ADCV            /CONVERT A TO D
ADSF            /SKIP ON A/D DONE FLAG
JMP. —1         /WAIT FOR FLAG
ADRB            /AND READ INTO AC
```

In the sequential address mode, the program advances the multiplexer channel-address register to the next channel each time an analog value is converted and read into the accumulator.

Should the converter/multiplexer be operated in the interrupt mode, the computer will be signaled each time that a binary word is ready, enabling the system to use processor time more efficiently.

AMPLIFIER, SAMPLE AND HOLD OPTIONS FOR AF01A

AH03 AMPLIFIER OPTION

The AH03 consists of a DEC amplifier (part #1505379) mounted on an A990 Amplifier Board with appropriate scaling networks and gain trim and balance potentiometers.

Open loop gain	2×10^6
Rated output voltage	(@ 10 ma) ±11v
Frequency response	
Unity Gain, small signal	10 mc
Full output voltage	300 kc
Slewing rate	30v/μsec
Overload recovery	200 μsec
Input voltage offset	Adjustable to 0
Avg vs temp	20 μv/°C
Vs supply voltage	15 μv/%
Vs time	10 μv/day
Input current offset	±2 na
Avg vs temp	0.4 na/°C
Vs supply voltage	0.15 na/%
Input impedance	
Between inputs	6 meg
Common mode	500 meg
Input voltage	±15v
Max common mode	±10v
Common mode rejection	20,000
Power	
Voltage	±15 to 16v
Current at rated load	35 ma

AH02 SAMPLE AND HOLD OPTION

A400 (standard gain options)	
Acquisition time to 0.01% (full-scale step)	<12 μsec
Aperture time	<150 nsec
Hold inaccuracy (droop)	<1 mv/msec
Temperature coefficient	0.1 mv/msec/°C
Gain (negative)	1.0 0.5
Input range (volts)	±5.0 ±10.0
Impedance	10K 10K
Output voltage	0 to −10v
Impedance	<1.0 ohm

GUARDED SCANNING DIGITAL VOLTMETER (TYPE AF04A)

DESCRIPTION

The Type AF04A is a guarded scanning digital voltmeter system, with wide dynamic range and high common-mode rejection, and fully capable of expansion to 1000 channels. The Type AF04A is used with a PDP-8/L computer to multiplex up to 1000 3-wire analog channels into a 6-decimal-digit (BCD) integrating digital voltmeter. Full scale ranges are from ±10mv to ±300v, with automatic ranging, 300 percent over ranging, and a usable 5 μv resolution. Guarded input construction and active integration assist in attaining an effective common-mode rejection of greater than 140 db at all frequencies. (Normal-mode rejection is infinite at multiples of power line frequency.)

This system is ideally suited for data acquisition or process monitoring where a wide range of signals requires large dynamic range. The 10-mv range has 0.001 percent resolution and, coupled with excellent noise rejection, allows accurate direct measurement of thermocouples, strain gauges, load cells, and other low-level transducers without additional amplification.

The AF04A Voltmeter, operated under program control, is capable of either random channel selection or sequential channel selection. The computer selects either program controlled ranging (for fastest speed) or autoranging, as well as the integration time of the integrating digital voltmeter (IDVM). The digitized data, as well as the current channel address, is read by the computer in either two or three bytes.

A decimal display of the digitized value, including sign and decimal location, is continuously displayed on the front panel. The current channel number is also displayed. Front-panel controls on the digital voltmeter allow manual setting of all the programmed functions. A front-panel control allows continuous display of the internal secondary standard, which can be prewired to a particular channel for reference checking during normal operation. The AF04A Voltmeter System may be manually controlled, completely independent of the computer.

SPECIFICATIONS

Full scale ±	10mv, 100mv, 1v, 10v, 100v, 300v, and automatic ranging
Over ranging	300% on all but highest range
Resolution	5 μv (usable), 0.1 μv (LSB)
Accuracy (overall worst case with daily calibration at calibration temperature)	±0.004% of reading ±0.01% of full scale±5μv
Stability (RMS full scale and zero drift)	±0.006%/day
Temperature coefficient	±0.003% of reading/°C

Full scale Zero	±0.002% of full scale/°C (±0.006% of full scale/°C on 10mv and 1v range)
Line voltage stability	±0.0005%/10% change
Maximum common-mode voltage	±300v from power line ground
Common-mode rejection (166.6 msec integration period and 1000 ohm-source unbalance)	> 140 db at all frequencies
Normal-mode rejection	Infinite at multiples of line frequency
Input impedance 10, 100, 1000 mv ranges	1000 meg/v
10, 100, 300v ranges	10 meg
Internal secondary standard	
Value	±1.000v
Accuracy	±0.002% traceable to N.B.S.
Stability	±0.005%/month
Temperature coefficient	negligible

SELECTED RESOLUTION

	0.001%		0.01%		0.1%	
DC Voltage Range	Maximum Reading	Resolution	Maximum Reading	Resolution	Maximum Reading	Resolution
10 mv	30.0000 mv	0.1 μv	030.000 mv	1 μv	0030.00 mv	10 μv
100 mv	300.000 mv	1 μv	0300.00 mv	10 μv	00300.0 mv	100 μv
1000 mv	3000.00 mv	10 μv	03000.0 mv	100 μv	003000. mv	1 mv
10v	30.0000v	100 μv	030.000v	1 mv	0030.00v	10 mv.
100v	300.000v	1 mv	0300.00v	10 mv	00300.0v	100 mv
1000v*	0300.00v	10 mv	00300.0v	100 mv	000300.v	1v

*1000v range is scanner-limited to 300v peak maximum

SCANNING SPEED

(Programmed Range)

Resolution	Integration Time	Total Time	Scanning Speed
0.1%	1.6 msec	20 msec	50 ch/sec.
0.01%	16.6 msec	40 msec	25 ch/sec.
0.001%	166.6 msec	188 msec	5 ch/sec.

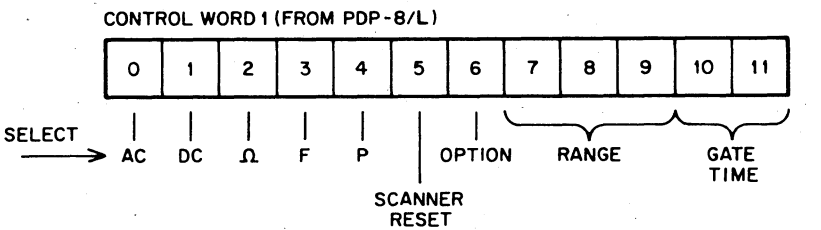
Scanning Speed (Auto Range)—Add 6-36 msec depending on per-channel voltage span.

INSTRUCTIONS

The I/O transfer (IOT) commands associated with the scanning digital voltmeter system are designed to minimize the computer overhead associated with this option while retaining maximum program controlled flexibility. The IOT instructions are:

Select Range and Gate (VSEL)

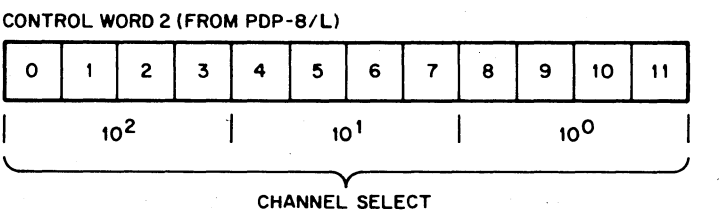
Octal Code: 6542
Event Time: 2
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
Operation: The contents of the accumulator are transferred to the AF04A control register as shown below:
Symbol: C(AC) = > C(VCR)



Control Word 1 only used if a range change is required.

Select Channel and Convert (VCNV)

Octal Code: 6541
Event Time: 1
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
Operation: The contents of the accumulator are transferred to the AF04A channel address register as shown below. The analog signal on the selected channel is automatically digitized.
Symbol: C(AC) = > C(VAR)



Index Channel and Convert (VINX)

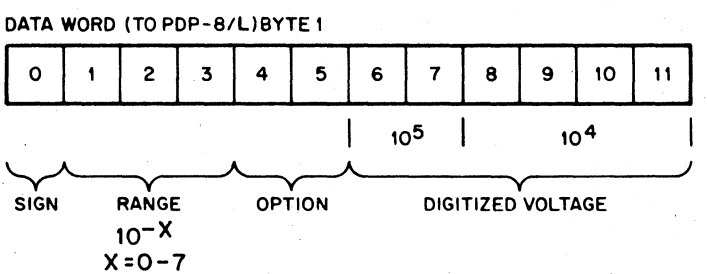
Octal Code: 6544
Event Time: 3
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
Operation: The last channel address is incremented by one and the analog signal on the selected channel is automatically digitized. The contents of the control register is unchanged.
Symbol: VAR + 1 = > VAR

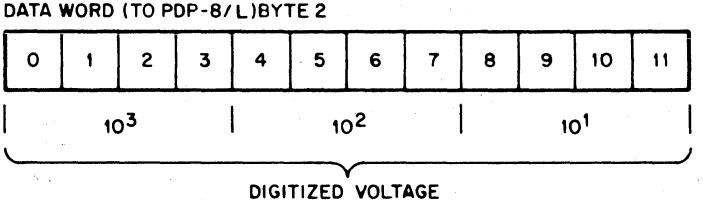
Skip on Data Ready (VSDR)

Octal Code: 6561
Event Time: 1
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
Operation: When the scanning voltmeter has selected a channel and digitized the analog signal, a data ready flag is set. This instruction is used to test for the data ready flag.
Symbol: If Flag = 1, then PC + 1 = > PC

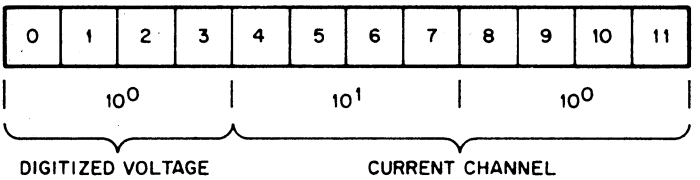
Read Data and Clear Flag (VRD)

Octal Code: 6562
Event Time: 2
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
Operation: The contents of the selected byte of the voltmeter output word is transferred to the accumulator and the data ready flag is cleared. The first data flag after the flag is set; is always byte 1 (see below). Subsequent bytes are program selected using the byte advance command.
Symbol: C(VOR) = > C(AC)





DATA WORD (TO PDP-8/L) BYTE 3



Data word 3 seldom required, all address and digitized data are in 8-4-2-1 BCD format.

Byte Advance (VBA)

Octal Code: 6564
Event Time: 3
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
Operation: The total data word from the AF04A is 36-bits long. The first data word after the flag is set, is always the twelve most significant bits. The BYTE ADVANCE command requests the next twelve most significant bits. When the data is available, the data ready flag is set again. To select the twelve least significant bits, a second BYTE ADVANCE command is required. When the data is available, the data ready flag is set again.
Symbol: $C(VOR_{0-12}) = > C(VOR_{13-23})$
or $C(VOR_{13-23}) = > C(VOR_{24-35})$

Sample Current Channel (VSCC)

Octal Code: 6571
Event Time: 1
Indicators: FETCH, IR = 6
Execution Time: 4.25 microseconds
Operation: The analog signal on the current channel is digitized. This command is **not** required except when multiple samples are required on any channel. (Using this command on a preselected channel saves up to 10 milliseconds per sample.)
Symbol: None.

FREQUENCY AND PERIOD MEASUREMENT OPTIONS FOR AF04A

A separate input permits the IDVM to be used as a frequency counter capable of counting to 2mHz with selectable gate times of 1, 10, and 100 milliseconds, providing measurement resolution of 10Hz. Increased accuracy at low frequencies (to 10kHz with automatic 250% overranging) is accomplished with the period-measurement mode. This mode counts an internal frequency source for 1 , 10, or 100 periods of the frequency being measured, thereby providing increased full-scale accuracy. Period readout is in milliseconds.

Frequency and voltage measurements may be made within one scanning cycle by grouping all frequency inputs in one master or slave scanner and all voltage inputs in another master or slave scanner. The output of one scanner may then be connected to the frequency-input connector of the

IDVM and the output of the other scanner to the voltage input. One of the optional control word bits is used to program the IDVM for frequency or period measurements.

SPECIFICATIONS

Frequency Measurements

Range: 10Hz to 2mHz

Sensitivity: 100mv rms or −1v pulses, at least 0.3 μsec wide at 50% points. 100v rms maximum working voltage.

Input Impedance: 22k ohms shunted by less than 1000 pf, including internal cabling.

Accuracy: ±1 count + time base accuracy

Time Base: 100 KHz crystal oscillator with initial accuracy of ±0.0005%, long-term stability ±0.0001%/wk; temp. coefficient ±0.0002%/°C.

Period Measurements

Range: 1, 10, and 100 period average. Input frequency from 10Hz to 25kHz sine wave or 0.1 pps. to 25,000 pps.

Sensitivity: 100 mv rms or −1v pulses, at least 0.3 μsec wide at 50% points. 100v rms maximum working voltage.

Input Impedance: 22k ohms shunted by less than 1000pf, including internal cabling.

Accuracy: ±1 count + time base accuracy + trigger error. Trigger error < ± 0.03% for 100mv rms sine wave with 40db signal-to-noise ratio.

Time Base: 100kHz crystal oscillator with initial accuracy of ±0.0005%, long-term stability ±0.0001%/wk; temp. coefficient ±0.0002%/°C.

Selected Resolution	0.001%		0.01%		0.1%	
Function	Maximum Reading	Resolution	Maximum Reading	Resolution	Maximum Reading	Resolution
Frequency	2000.00kHz	10Hz	02000.0kHz	100Hz	002000kHz	1kHz
Period	99.9999msec	0.1μs	999.999msec	1.0μs	9999.99msec	10μs

Additional AF04A Options

Information on the following options may be had from your nearest DIGITAL EQUIPMENT CORPORATION Office:

Frequency (period) measurements.
AC/ohms/DC Converter
Time-of-day clock.
Thumb-wheel data entry panel.
Thermocouple reference junctions.
Extended scanner for more than 1000 channels.
Special cabinet with roll-out drawer chassis accessibility.

CHAPTER 8
PDP-8/L INPUT/OUTPUT FACILITIES

Since the processing power of the computer depends largely upon the range and number of peripheral devices that can be connected to it, the PDP-8/L has been designed to interface readily with a broad variety of external equipment. The following chapters of this handbook define the interface characteristics of the computer to allow the reader to design and implement any electrical interfaces required to connect devices to the PDP-8/L. Chapters 9 and 10 functionally describe the logic circuit elements involved in programmed data transfers and data break transfers, respectively. Chapter 11 gives detailed information on digital logic circuits used for computer interfacing. Chapter 12 describes the design and construction of interface equipment. Chapter 13 lists connection point, module type, and module location, etc., for each interface signal; gives detailed loading and driving characteristics for each module in the computer interface; then presents some general rules and characteristics to be considered in selecting or designing electrical circuits to be connected to the PDP-8/L. Chapter 14 presents information for planning the installation of a basic PDP-8/L and the available standard optional equipment.

The simple I/O technique of the PDP-8/L, the availability of DEC's FLIP CHIP logic circuit modules, and DEC's policy of giving assistance wherever possible allow inexpensive, straightforward 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/L contains a processor and core memory composed of Digital's M Series TTL circuit modules. These circuits have an operating temperature exceeding the limits of 50°F to 130°F, so no air-conditioning is required at the computer site. Standard 115V, 50/60-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/L 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 a random access disc file, card equipment, line printers, magnetic tape transports, magnetic drums, analog-to-digital converters, CRT displays, and digital plotters. The PDP-8/L 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 183 devices requiring one programmed command pulse can be connected to the computer. One machine using the optional data break facility can be connected directly to the PDP-8/L or up to seven such machines can be connected through a Data Multiplex Type DM01. Interfacing of any devices to the computer requires no modifications to the processor and can be achieved in the field.

Control of some kind is needed to determine when an information exchange is to take place between the PDP-8/L and peripheral equipment and to indicate the location(s) in the computer memory which will accept or yield the data. Either the computer program or the device external to the computer can exercise this control. Transfers controlled by the computer, hence under control of its stored program, are called programmed data transfers. Transfers made at times controlled by the external devices through the optional data break facility are called data break transfers.

Programmed Data Transfers

The majority of I/O transfers occur under control of the computer program. To transfer and store information under program control requires about six times as much computer time as under data break control. In terms of real time, the duration of a programmed transfer is rather small, due to the high speed of the computer, and is well beyond that required for laboratory or process control instrumentation.

To realize full benefit of the built-in control features of the PDP-8/L programmed I/O transfers should be used in most cases. Controls for devices using programmed data transfers are usually simpler and less expensive than controls for devices using data break transfers. Using programmed data transfer facilities, simultaneous operation of devices is limited only by the relative speed of the computer with respect to the device speeds, and the search time required to determine the device requiring service. Analog-to-digital converters, digital-to-analog converters, digital plotters, line printers, message switching equipment, and relay control systems typify equipment using only programmed data transfers.

Data Break Transfers

Devices which operate at very high speed or which require very rapid response from the computer use the optional data break facilities. Use of these facilities permits an external device, almost arbitrarily, to insert or extract words from the computer core memory, bypassing all program control logic. Because the computer program has no cognizance of transfers made in this manner, programmed checks of input data are made prior to use of information received in this manner. The data break is particularly well-suited for devices that transfer large amounts of data in block form, e.g., random access disc file, high-speed magnetic tape systems, high-speed drum memories, or CRT display systems containing memory elements.

Logic Symbols

Figure 20 defines the symbols used in the following chapters of this handbook to express signals and digital logic circuits.

The PDP-8/L uses TTL logic internally. In order to discuss some of the internal logic pertaining to interfacing, it is necessary to understand the TTL symbology used in the 8/L. The figures are as follows:

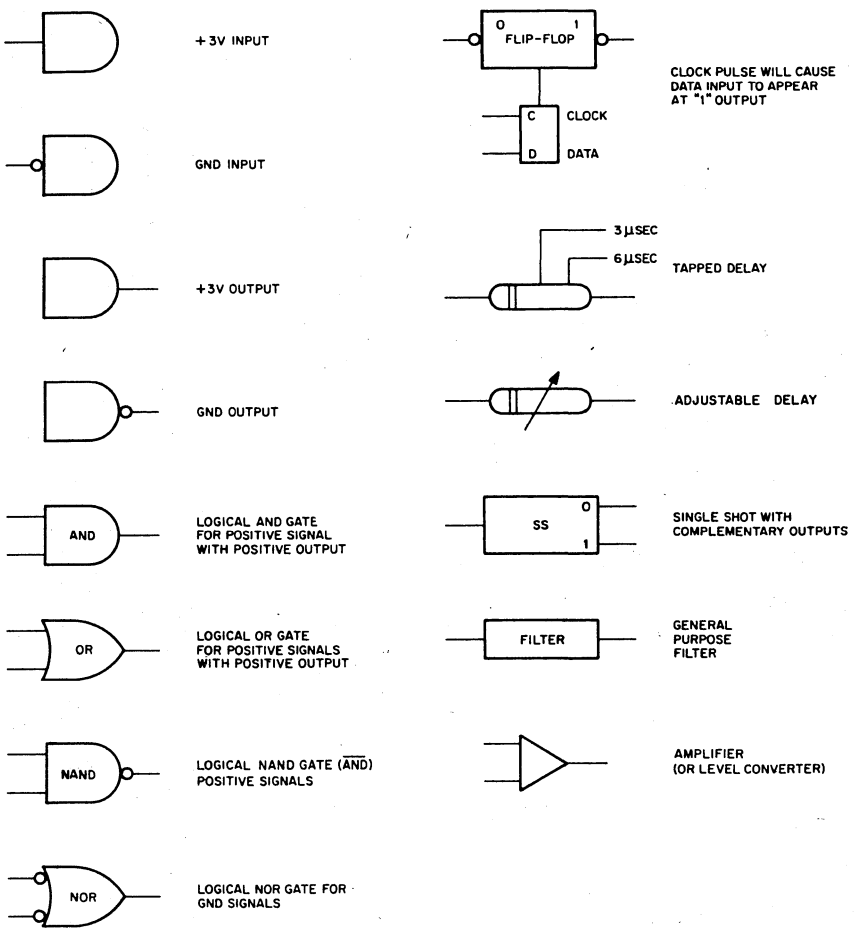


Figure 20. Logic Symbols

SIGNAL NAMES

All signals not originating at a flip-flop output are true when the line is high. Thus a line labeled IOP 1 is positive when the pulse is being generated and is otherwise ground. Similarly, $\overline{\text{AC CLEAR}}$ is at ground for assertion, and positive otherwise.

Signals originating at flip-flops are defined in terms of the flip-flop state. The following table illustrates the convention.

Signal Name	State of MB Flip-Flop	Signal Voltage
MB03(0)	0	+3V
	1	0V
MB03(1)	0	0V
	1	+3V

CHAPTER 9
PROGRAMMED DATA TRANSFERS

The majority of I/O transfers take place under control of the PDP-8/L program, taking advantage of control elements built into the computer. Although programmed transfers take more computer and actual time than do data break transfers, the timing discrepancy is insignificant, considering the high speed of the computer with respect to most peripheral devices. The maximum data transfer rate for programmed operations of 12-bit words is 148 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/L 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. Gating is also recommended at the input to the peripheral device in order to minimize loading on the BAC signal lines.

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 21 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 a 4.25 μsec expanded computer 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 22 indicates the timing of programmed data transfers and Figure 23 shows the decoding of the IOT instruction.

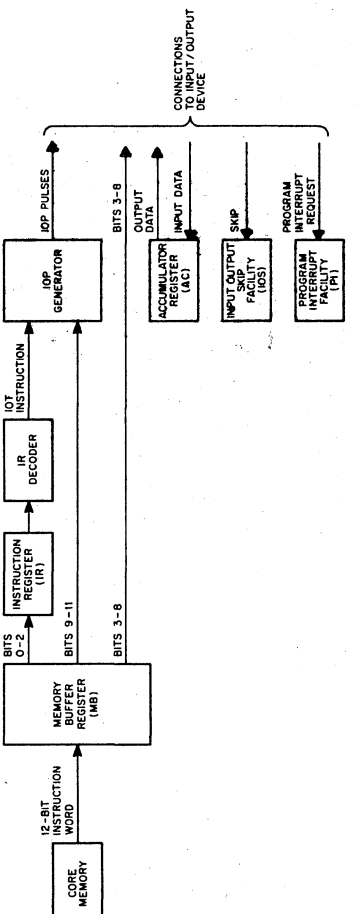


Figure 21. Programmed Data Transfer Interface Block Diagram

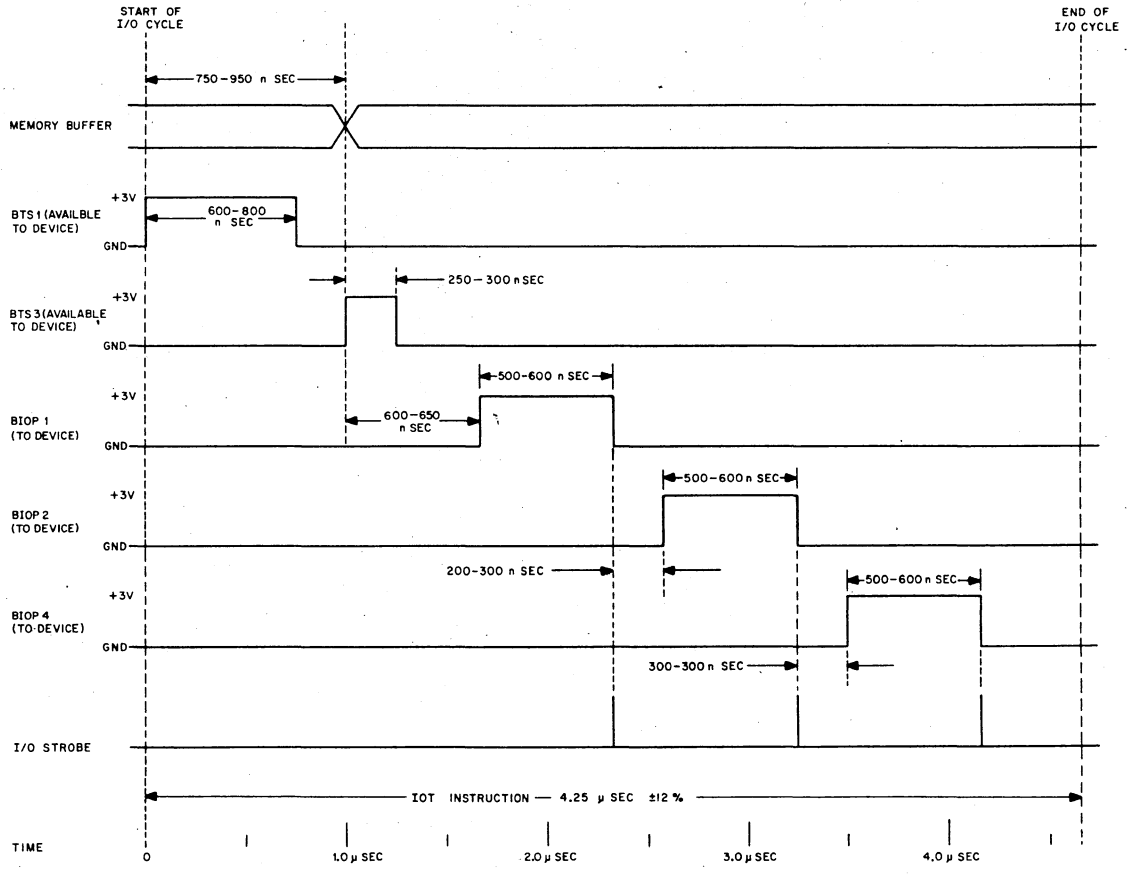


Figure 22. Programmed Data Transfer Timing

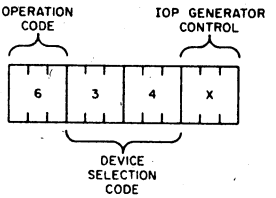


Figure 23. Typical IOT Instruction Decoding

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 6_8, it identifies an IOT instruction and the computer generates a slow cycle pulse. The Slow Cycle signal ANDS with TP3 to generate I/O Start and sets the PAUSE flip-flop. The logic of the IOP generator consists of a re-entrant delay chain which generates three time states. These time states are gated with MB bits 11, 10 and 9 to gnerate IOP 1, IOP 2 and IOP 4 respectively. Note that an IOP is generated only if the corresponding MB bit is set, although the I/O timing remains constant. At the end of each IOP, the state of the I/O interface is sampled by an I/O strobe pulse.

Following the end of IOP 4 time, the PAUSE flip-flop is reset and the normal timing chain is restarted.

Unlike PDP-8/I, the PDP-8/L makes no timing distinction between internal I/O functions and normal I/O. Thus all I/O instructions cause the slow cycle.

Note: All cycle times of the PDP/8L have a tolerance of ± 12%.

Instruction Bit	IOP Pulse	IOT Pulse	Event Time	Used Primarily For, But Not Restricted To
11	IOP 1	IOT 1	1	Sampling Flags, Skipping.
10	IOP 2	IOT 2	2	Clearing Flags, Clearing AC.
9	IOP 4	IOT 4	3	Reading Buffers, Loading Buffers and Clearing Buffers.

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 (M103 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 25 shows generation of command pulses by several DC channels.

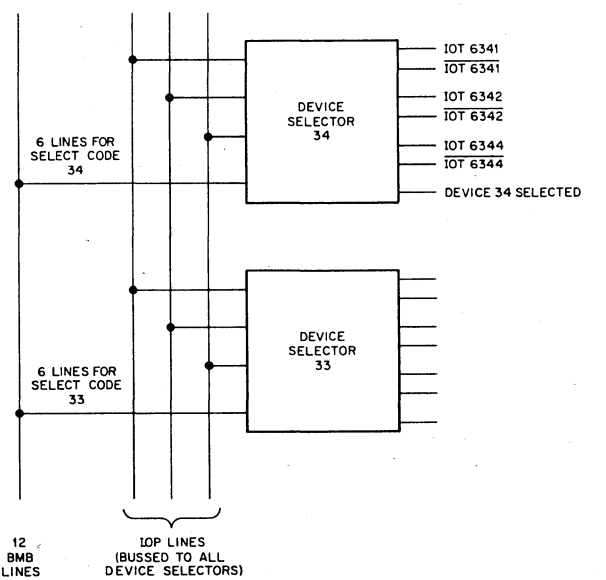


Figure 25. Generation of IOT Command Pulses by Device Selectors

The logical representation for a typical channel of the DS, using channel 34, is shown in Figure 26. A 6-input NAND gate wired to receive the appropriate signal outputs from the MB3-8 for select code 34 activates the channel. In the DS module, the NAND gate contains 8 input terminals; 6 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 connected to 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 power NAND gates, each of which produces an output pulse if the corresponding IOP pulse occurs. The positive output from each gate 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. An amplifier module can be connected in each channel of the DS to provide greater output drive.

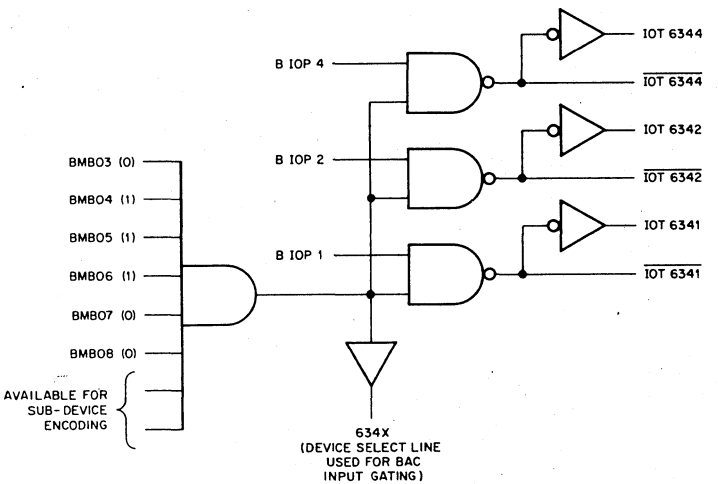


Figure 26. Typical Device Selector (Device 34)

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 27. 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 skip (designated $\overline{\text{SKIP BUS}}$) to ground when the skip conditions are fulfilled. The state of the skip bus is sampled at the end of each IOT. If the bus has been 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.

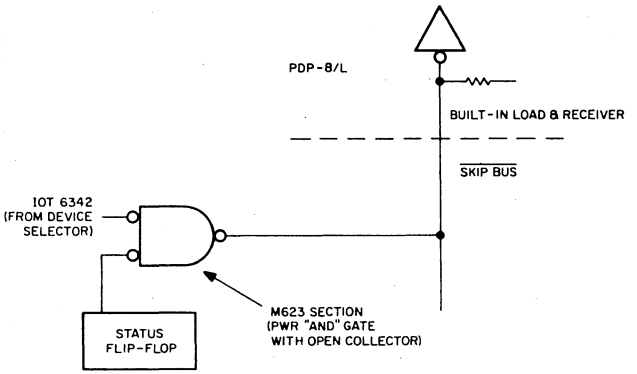


Figure 27. Use of IOS to Test the Status of an External Device

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

Address	Instruction	Remarks
.		
.		
.		
.		
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 this 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 a bus driver, 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/L. A 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/L. An IOP that drives the AC input bus terminal to ground causes setting of the corresponding AC flip-flop to the binary 1 state. Output and input connections to the accumulator appear in Figure 28.

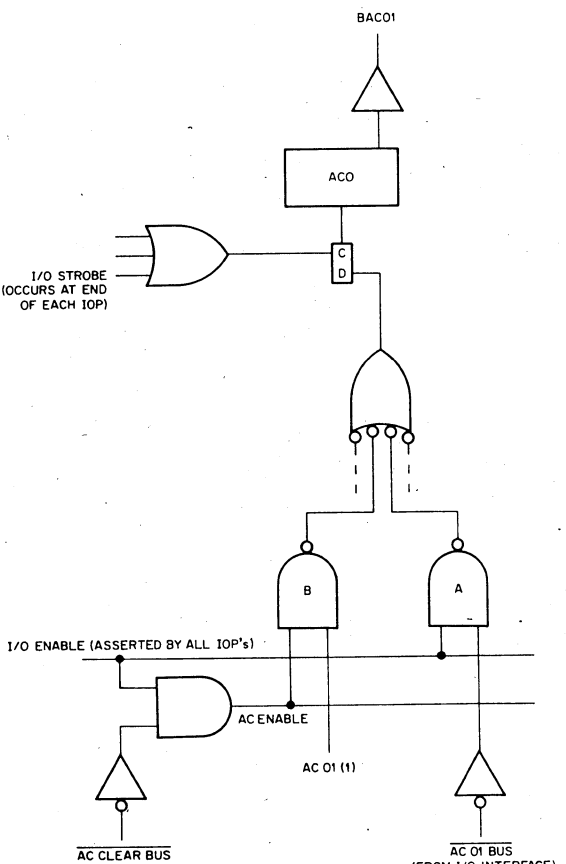


Figure 28. Accumulator Input and Output

The status of the link bit is not available to enter into transfers with peripheral equipment (unless it is rotated into the AC). A bus 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/L 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 $\overline{\text{AC CLEAR BUS}}$ is not asserted, 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. $\overline{\text{AC CLEAR BUS}}$ may also be used as an I/O AC clear by activating only this line from a separate IOT.

The illustration in Figure 29 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 interface gate. 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 transfer into the accumulator. (The program need not include a clear AC command prior to loading in this manner.)

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.

The BAC lines should be gated by the select code at each device to prevent excessive loading. A special module, the M101, is provided for this purpose. See Chapter 11 for more details.

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 transfer to an external 6-bit data register. Since this is a jam transfer, it is not necessary to clear the external data register. The 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 strobe pulse shown in Figure 30 must occur when the data to be placed in the external data register is held in the accumulator. This pulse therefore must be under computer control to effect synchronization with the operation or program of the computer.

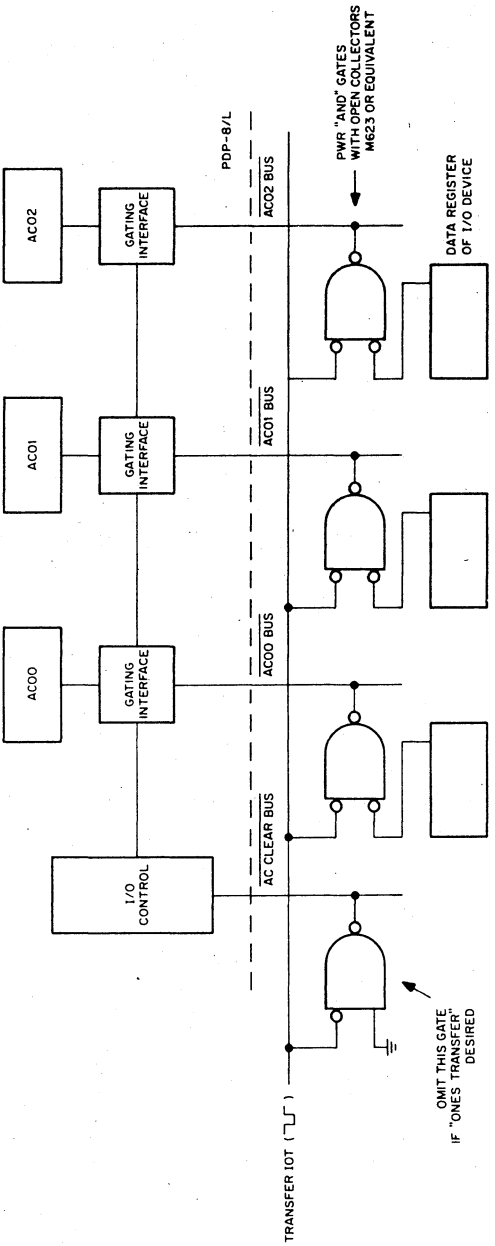


Figure 29. Loading Data into the Accumulator from an External Device

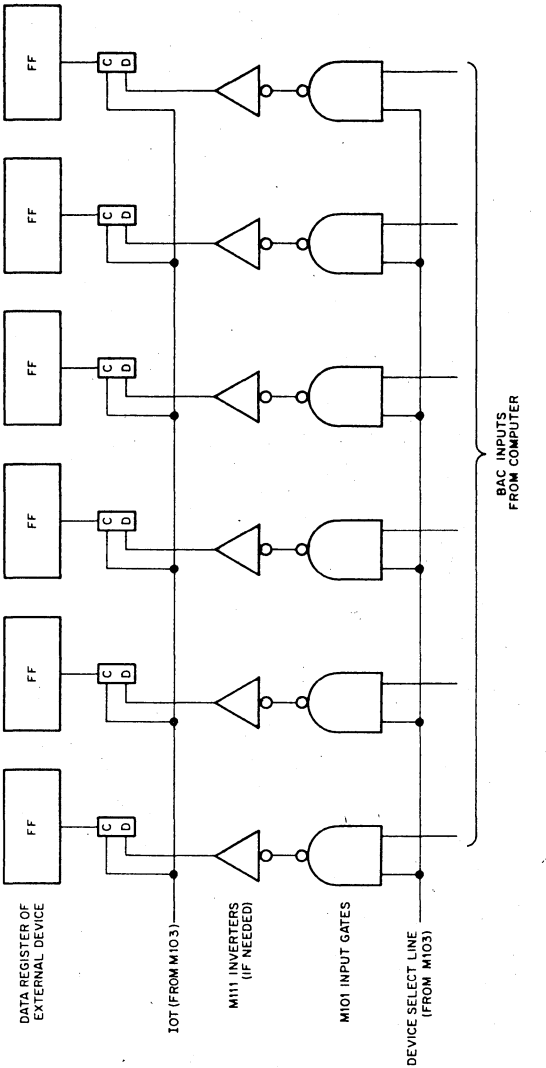


Figure 30. Loading a 6-Bit Word into an External Device from the Accumulator

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 32, a flag signal from a status flip-flop operates a standard gate with no internal 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.

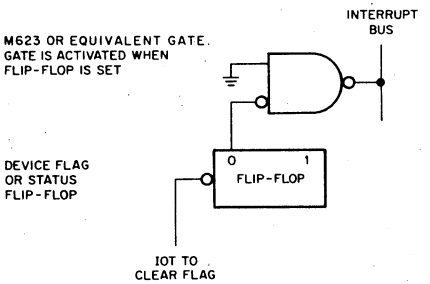


Figure 32. Program Interrupt Request Signal Origin

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:

```
Tag     Address   Instruction        Remarks
        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 AC, AND
        3001          .        /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/L 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. In the basic PDP-8/L, the teletype flags are already connected. 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 33 shows IOS and PI connections for two typical devices.

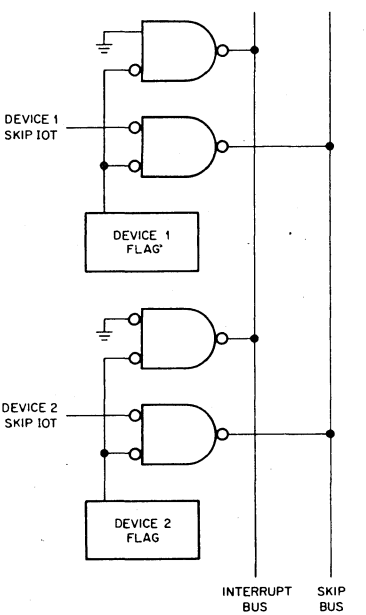


Figure 33. Multiple Inputs to IOS and PI Facilities

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

```
Tag       Address   Instruction          Remarks
          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 SR44          /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:

```
Tag               Instruction          Remarks
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.

CHAPTER 10
DATA BREAK TRANSFERS

The data break facility allows I/O device to transfer information directly with the PDP-8/L core memory on a cycle-stealing basis. Up to seven devices can connect to the data break facility through the optional Data Multiplexer Type DM01. The data break is particularly well-suited for devices which transfer large amounts of information in block form.

Peripheral I/O equipment operating at high speeds can transfer information with the computer through the data break facility more efficiently than through programmed means. The combined maximum transfer rate of the data break facility is 7.5 million bits per second. Information flow to effect a data break transfer with an I/O device appears in Figure 34.

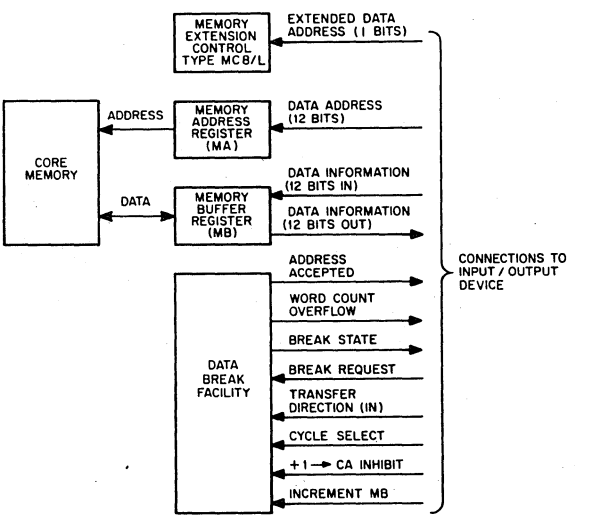


Figure 34. Data Break Transfer Interface Block Diagram

In contrast to programmed operations, the data break facilities permit an external device to control information transfers. Therefore, data-break device interfaces require more control logic circuits, causing a higher cost than programmed-transfer interfaces.

Data breaks are of two basic types: single-cycle and three-cycle. In a single-cycle data break, registers in the device (or device interface) specify the core memory address of each transfer and count the number of transfers to determine the end of data blocks. In the three-cycle data break two computer core memory locations perform these functions, simplifying the device interface by omitting two hardware registers.

In general terms, to initiate a data break transfer of information, the interface control must do the following:

a. Specify the affected address in core memory.

b. Provide the data word by establishing the proper logic levels at the computer interface (assuming an input data transfer), or provide readin gates and storage for the word (assuming an output data transfer).

c. Provide a logical signal to indicate direction of data word transfer.

d. Provide a logical signal to indicate single-cycle or three-cycle break operation.

e. Request a data break by supplying a proper signal to the computer data break facility.

Single-Cycle Data Breaks

Single-cycle breaks are used for input data transfers to the computer, output data transfers from the computer, and memory increment data breaks. Memory increment is a special data break in which the content of a memory address is read, incremented by 1, and rewritten at the same address. It is useful for counting iterations or external events without disturbing the computer program counter (PC) or Accumulator AC registers.

Input Data Transfers

Figure 35 illustrates timing of an input transfer data break. The address to be affected in core is normally provided in the device interface in the form of a 12-bit flip-flop register (data break address register) which has been preset by the interface control by programmed transfer from the computer.

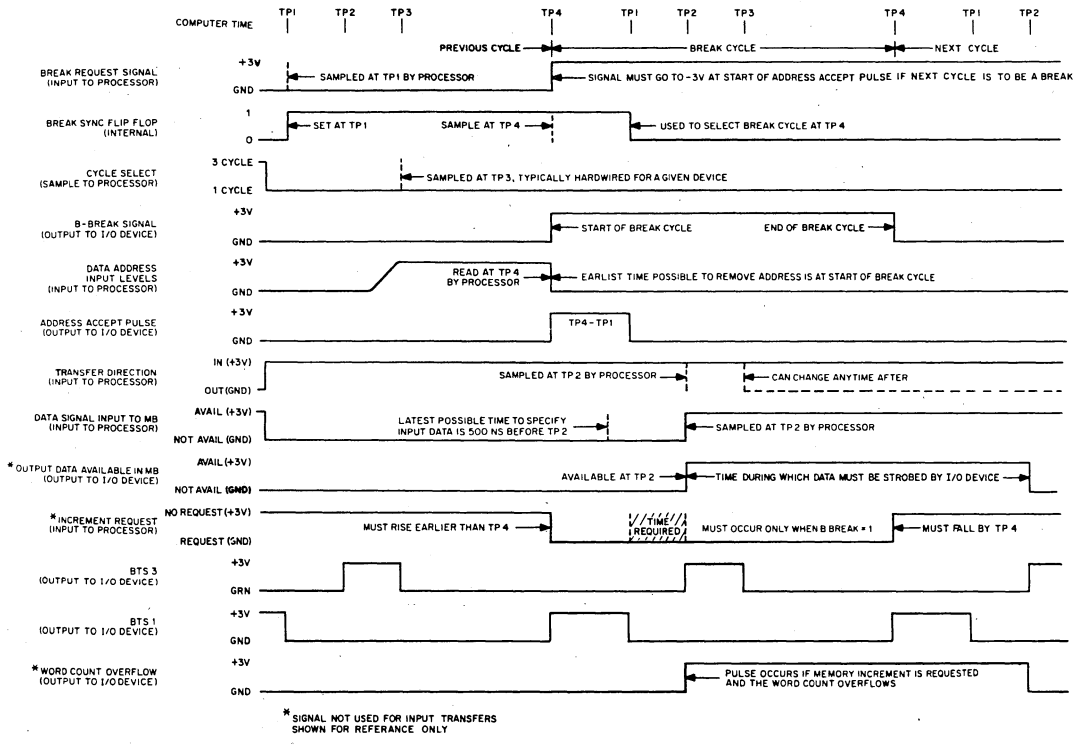


Figure 35. Single-Cycle Data Break Input Transfer Timing Diagram

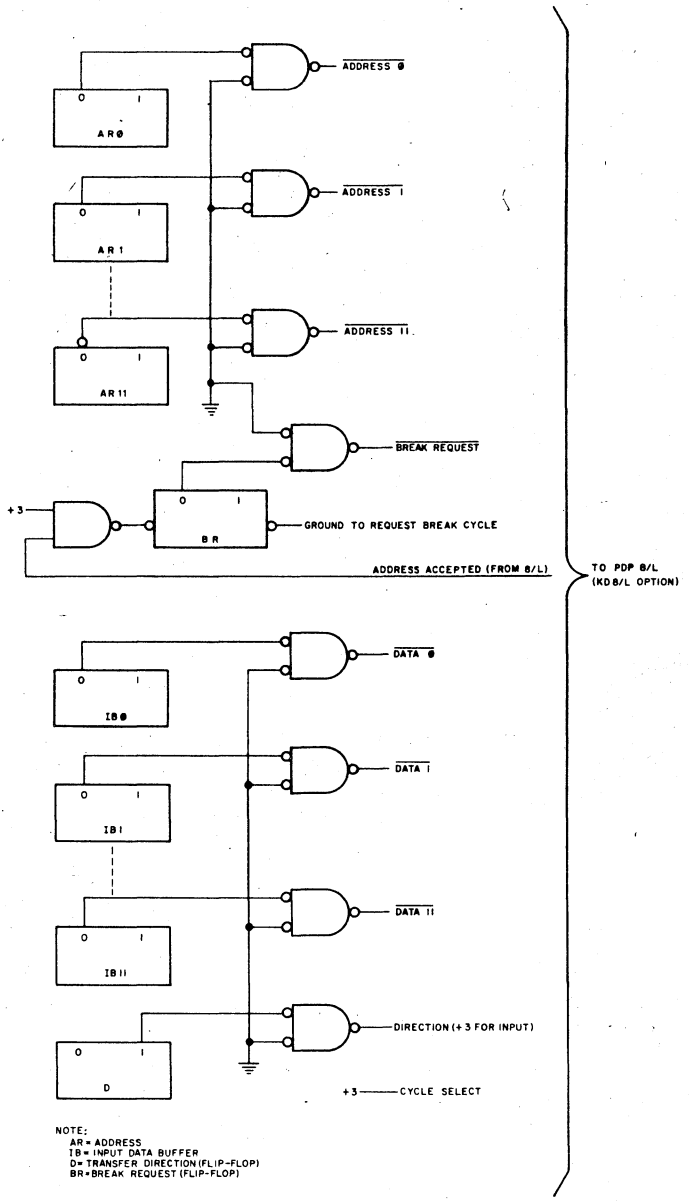


Figure 36. Device Interface Logic for Single-Cycle Data Break Input Transfer

External registers and control flip-flops supplying information and control signals to the data break facility and other PDP-8/L interface elements are shown in Figure 36. The input buffer register (IB in Figure 36) holds the 12-bit data word to be written into the computer core memory location specified by the address contained in the address register (AR in Figure 36).

Appropriate output terminals of these registers are connected to the computer to supply ground potential to designate binary 1's. Since most devices that transfer data through the data break facility are designed to use either single-cycle or three-cycle breaks, but not both, the Cycle Select signal can usually be supplied from a stable source (such as a ground connection or a + 3v clamped load resistor) rather than from a bistable device as shown in Figure 36.

Other portions of the device interface, not shown in Figure 36, establish the data word in the input buffer register, set the address into the address register, set the direction flip-flop to indicate an input data transfer, and control the break request flip-flop. These operations can be performed simultaneously or sequentially, but all transients should occur before the data break request is made. Note that the device interface need supply only static levels to the computer, minimizing the synchronizing logic circuits necessary in the device interface.

When the data break request arrives, the computer completes the current instruction, generates an Address Accepted pulse (at TP4 time of the cycle preceding the data break) to acknowledge receipt of the request, then enters the Break state to effect the transfer (see Chapter 5 of this handbook for more details on data break operations performed by the computer). The Address Accepted pulse can be used in the device interface to clear the break request flip-flop, increment the content of the address register, etc. If the Break Request signal is removed before TP1 time of the data break cycle, the computer performs the transfer in one 1.6-μsec cycle and returns to programmed operation.

Output Data Transfers

Timing of operations occurring in a single-cycle output data break is shown in Figure 37. Basic logic circuits for the device interface used in this type of transfer are shown in Figure 38. Address and control signal generators are similar to those discussed previously for input data transfers, except that the

Transfer Direction signal must be at ground potential to specify the output transfer of computer information. An output data register (OB in Figure 38) is usually required in the device interface to receive the computer information. The device, and not the PDP-8/L, controls strobing of data into this register. The device must supply strobe pulses for all data transfers out of the computer (programmed or data break) since circuit configuration and timing characteristics differ in each device.

When the data break request arrives, the computer completes the current instruction and generates an Address Accepted pulse as in input data break transfers. At TP4 time the address supplied to the PDP-8/L is loaded into the MA, and the Break state is entered. Not more than 900 nsec after TP4 (at TP2 time), the content of the device-specified core memory address is read and available in the MB. (This word is automatically rewritten at the same address during the last half of the Break cycle and is available for programmed operations when the data break is finished.) Data Bit signals are available as static levels of ground potential for binary 0's and +3v for binary 1's. The MB is changed at TP2 time of each computer cycle, so the data word is available in the MB for approximately 1.5 μsec to be strobed by the device interface.

Generation of the strobe pulse by the device interface can be synchronized with computer timing through use of timing pulses BTS1 or BTS3, which are avaliable at the computer interface. In addition to a timing pulse (delayed or used directly from the computer), generation of this strobe pulse should be gated by condition signals that occur only during the Break cycle of an output transfer. Figure 38 shows typical logic circuits to effect an output data transfer. In this example, BTS3 and B BREAK set the BREAK ENABLE flip-flop which remains set for one computer cycle (unless successive cycles are requested). This enabling signal samples the buffered MB lines into the data inputs of a D type flop. At BTSI time the data will be clocked into the Output Buffer flip-flops. Note that BTSI can generate a strobe pulse only during a BREAK ENABLE cycle. Interface input gates are M101; output bus drivers are M623.

By careful design of the input and output gating, one register can serve as both the input and the output buffer register. Most DEC options using the data break facility have only one data buffer register with appropriate gating to allow it to serve as an output buffer when the Transfer Direction signal is at ground potential or as an input buffer when the Transfer Direction signal is +3v.

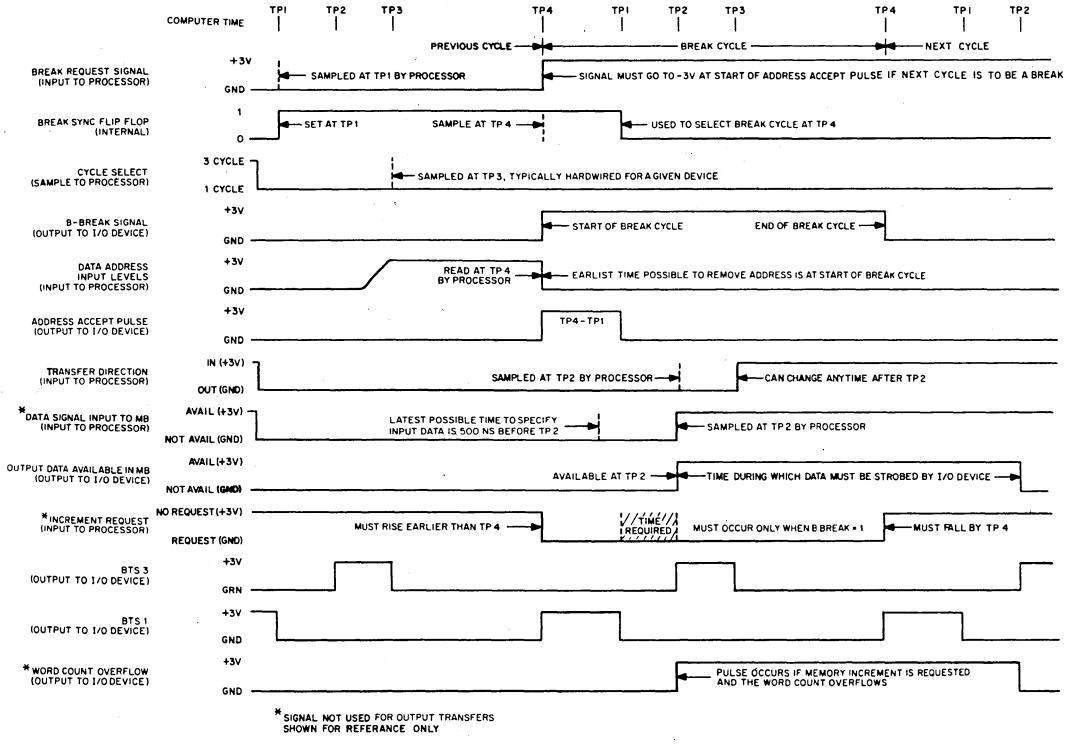


Figure 37. Single-Cycle Data Break Output Transfer Timing Diagram

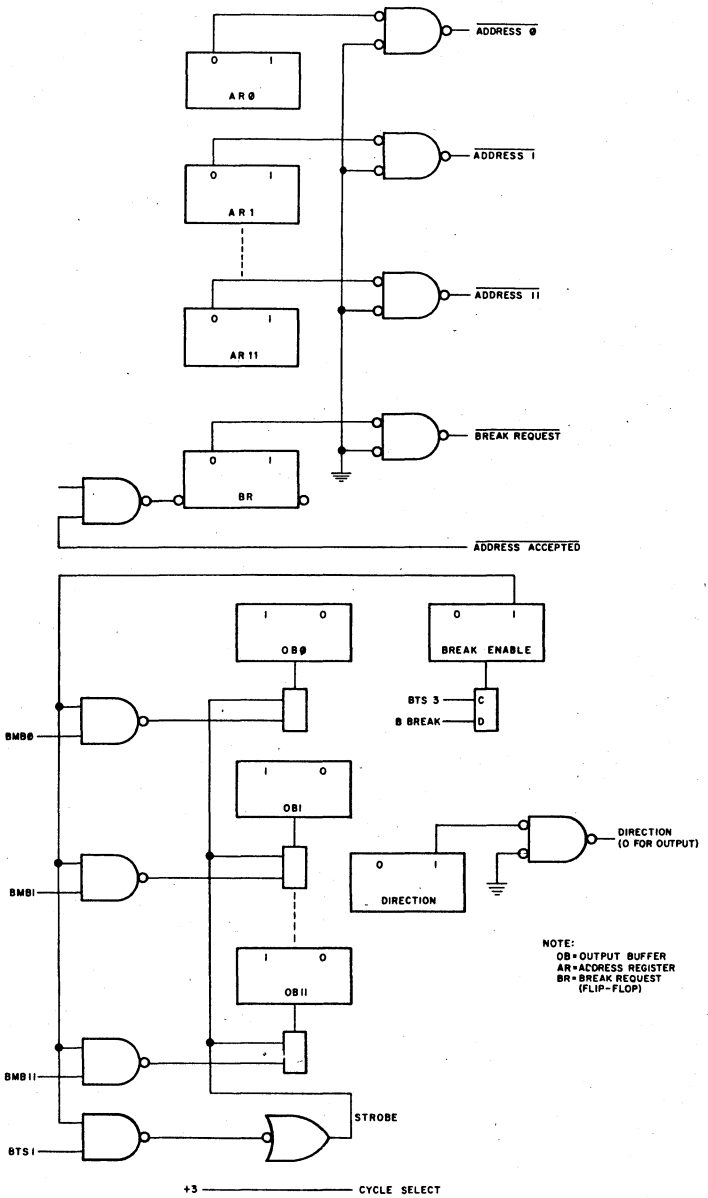


Figure 38. Device Interface Logic
for Single-Cycle Data Break Output Transfer

Memory Increment

In this type of data break the content of core memory at a device-specified address is read into the MB, is incremented by 1, and is rewritten at the same address within one 1.6-μsec cycle. This feature is particularly useful in building a histogram of a series of measurements, such as in pulse-height analysis applications. For example, in a computer-controlled experiment that counts the number of times each value of a parameter is measured, a data break can be requested for each measurement, and the measured value can be used as the core memory address to be incremented (counted).

Signal interface for a memory increment data break is similar to an output transfer data break except that the device interface generates an Increment MB signal and does not generate a strobe pulse (no data transfer occurs between the PDP-8/L and the device). Timing of memory increment operations appear in Figure 40.

An interface for a device using memory increment data breaks must supply twelve Data Address signals, a Transfer Direction signal, a Cycle Select signal, and a Break Request signal to the computer data break facility as in an output transfer data break. In addition, a ground potential increment MB signal must be provided at least 250 nsec before TP2 time of the Break cycle. This signal can be generated in the device interface by AND combining the B Break computer output signal, the output transfer condition of the Transfer Direction signal, and the condition signal in the device that indicates that an increment operation should take place. When the computer receives this Increment MB signal, it forces the MB control element to generate a Carry Insert signal at TP2 time to increment the content of the MB.

Three-Cycle Data Breaks

Timing of input or output 3-cycle data breaks is shown in Figure 42. The 3-cycle break uses the block transfer control circuits of the computer. The block transfer control provides an economical method of controlling the flow of data at high speeds between PDP-8/L core memory and fast peripheral devices, e.g., drum, disc, magnetic tape and line printers, allowing transfer rates in excess of 208 kc.

The three-cycle data break facility provides separate current address and word count registers in core memory for the connected device, thus eliminating the necessity for flip-flop registers in the device control. When several devices are connected to this facility, each is assigned a different set of core locations for word count and current address, allowing interlaced operations

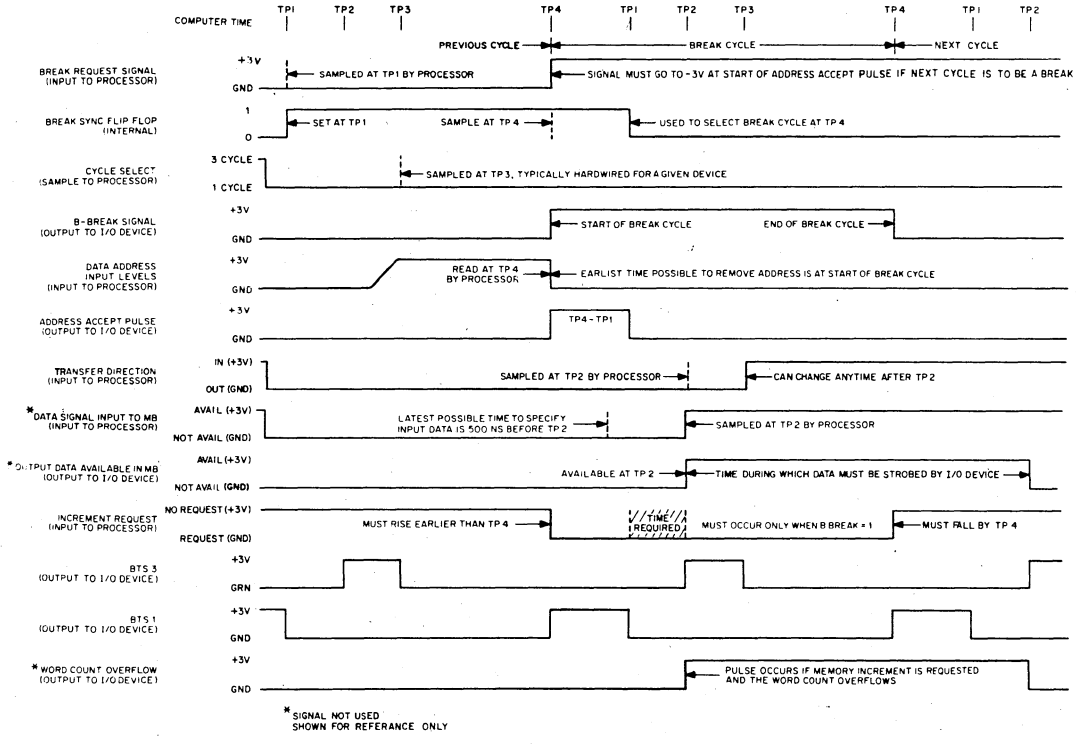


Figure 40. Memory Increment Data Break Timing Diagram

of all devices as long as their combined rate does not exceed 208 kc. The device specifies the location of these registers in core memory, and thus the software remains the same regardless of what other equipment is connected to the machine. Since these registers are located in standard memory, they may be loaded and unloaded directly without the use of IOT instructions. In a procedure where a device requests to transfer data to or from core memory, the three-cycle data break facility performs the following sequence of operations:

a. An address is read from the device to indicate the location of the word count register. This address is always the same for a given device; thus it can be wired in and does not require a flip-flop register.

b. The content of the specified address is read from memory and 1 is added to it before rewriting. If the content of this register becomes 0 as a result of the addition, a WC Overflow pulse will be transmitted to the device. To transfer a block of N words, this register is loaded with − N during programmed initialization of the device. After the block has been fully transferred this pulse is generated to signify completion of the operation.

c. The next sequential location is read from memory as the current address register. Although the content of this register is normally incremented before being rewritten, an increment CA Inhibit (+1 → CA Inhibit) signal from the device may inhibit incrementation. To transfer a block of data beginning at location A, this register is program initialized by loading with A-1.

d. The content of the previously read current address is transferred to the MA to serve as the address for the data transfer. This transfer may go in either direction in a manner identical to the single-cycle data break system.

The three-cycle data break facility uses many of the gates and transfer paths of the single-cycle data break system, but does not preclude the use of standard data break devices. Any combination of three-cycle and single-cycle data

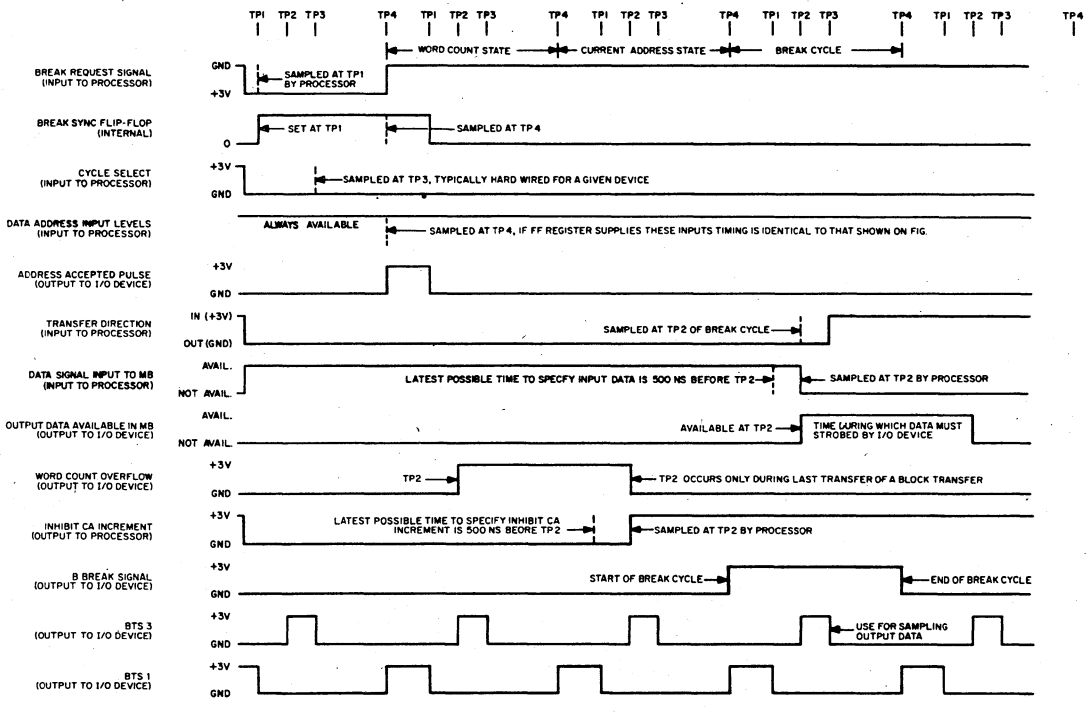


Figure 42. Three-Cycle Data Break Timing Diagram

break devices can be used in one system, as long as a multiplexer channel is available for each. Two additional control lines are provided with the three-cycle data break. These are:

a. Word Count Overflow. A level change from GND to +3V, from TP2 to TP2, is transmitted to the device when the word count becomes equal to zero.

b. Increment CA Inhibit. When ground potential, this device-supplied signal inhibits incrementation of the current address word.

In summary, the three-cycle data break is entered similarly to the single-cycle data break, with the exception of supplying a ground-level Cycle Select signal to allow entry of the WC (Word Count) state to increment the fixed core memory location containing the word count. The device requesting the break supplies this address as in the one-cycle data break, except that this address is fixed and can be supplied by wired ground and +3v signals, rather than from a register. The sole restriction on this address is that it must be an even number (bit 11 = 0). Following the WC state a CA (Current Address) state is entered in which the core memory location following the WC address (bit 11 = 1) is read, incremented by one, restored to memory, and used as the transfer address (by MB = > MA). Then the normal B (Break) state is entered to effect the transfer.

CHAPTER 11
DIGITAL LOGIC CIRCUITS

DIGITAL LOGIC CIRCUITS

PDP-8/L interfacing is constructed of Digital FLIP CHIP modules. The Digital Logic Handbook describes more than 150 of these modules, all of their component circuits, and the associated accessories; i.e., power supplies and mounting panels. The user should study this catalog carefully before beginning the design of a special interface.

The interface modules of the PDP-8/L are the M111, M906, M516, M660 and M623 modules. Interface signals to the computer use either a combination of the M111 and M906 modules or the M516 module. Interface signals from the computer will originate from a combination of M623 and M906 modules for data signals, and M660 modules for timing signals.

M111/M906 Positive Input Circuit

The M111 Inverter module is used in conjunction with the M906 Cable Terminator module which clamps the input to prevent excursions beyond + 3 volts and ground. The M906 also provides the pullup resistors to + 5 volts.

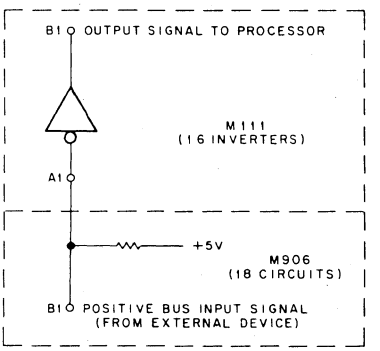


Fig. 43 Typical M111/M906 Positive Input Circuit

M516 Positive Bus Receiver Input Circuit

Six four input NAND gates with overshoot and undershoot clamp on one input of each gate. Pullup resistors to + 5 v are also provided.

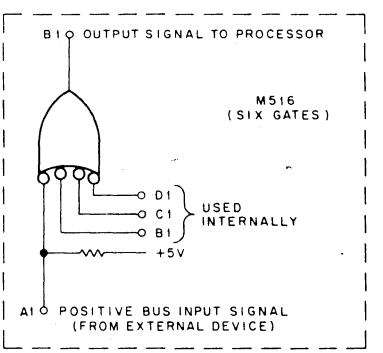


Fig. 44 Typical M516 Positive Bus Receiver Input Circuit

M623/M906 Positive Output Circuit

The M623 Bus Driver module contains twelve circuits with negative NAND's. Used in conjunction with the M906 Cable Terminator module, the output is clamped to prevent excursions beyond + 3 volts and ground. Output can drive + 5 milliamperes at the high level and sink 20 milliamperes at the low level.

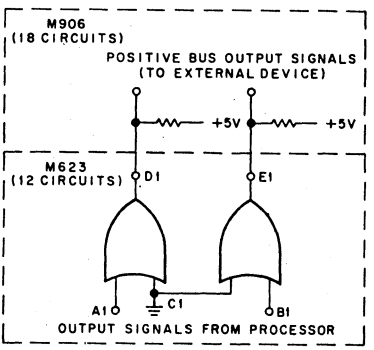


Fig. 45 Typical M623/M906 Positive Output Circuit

M660 Bus Driver Output Circuit

Three circuits which provide low impedance 100 ohm terminated cable driving capability using M Series levels or pulses of duration greater than 100 nsec. Output can drive + 5 ma at the high level and sink 20 ma at the low level, in addition to termination current required by the G717 termination module. The M660 module is used in the PDP-8/L for the following output signals:

IOP 1, IOP 2, IOP 4, TS 3, TS 1

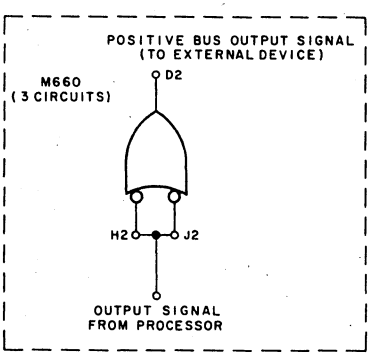


Fig. 46 Typical M660 Bus Driver Output Circuit

Module Selection for Interface Circuits of Peripheral Equipment
Two FLIP CHIP modules are of particular interest in the design of equipment to interface to the PDP-8/L. Complete details on these and other FLIP CHIP modules can be found in the Digital Logic Handbook.

M103 Device Selector
The M103 selects an input/output device according to the code in the instruction word (being held in the memory buffer during the IOT cycle). M103 module includes diode protection clamps on input lines so that it may be used directly on the PDP-8/L positive bus.

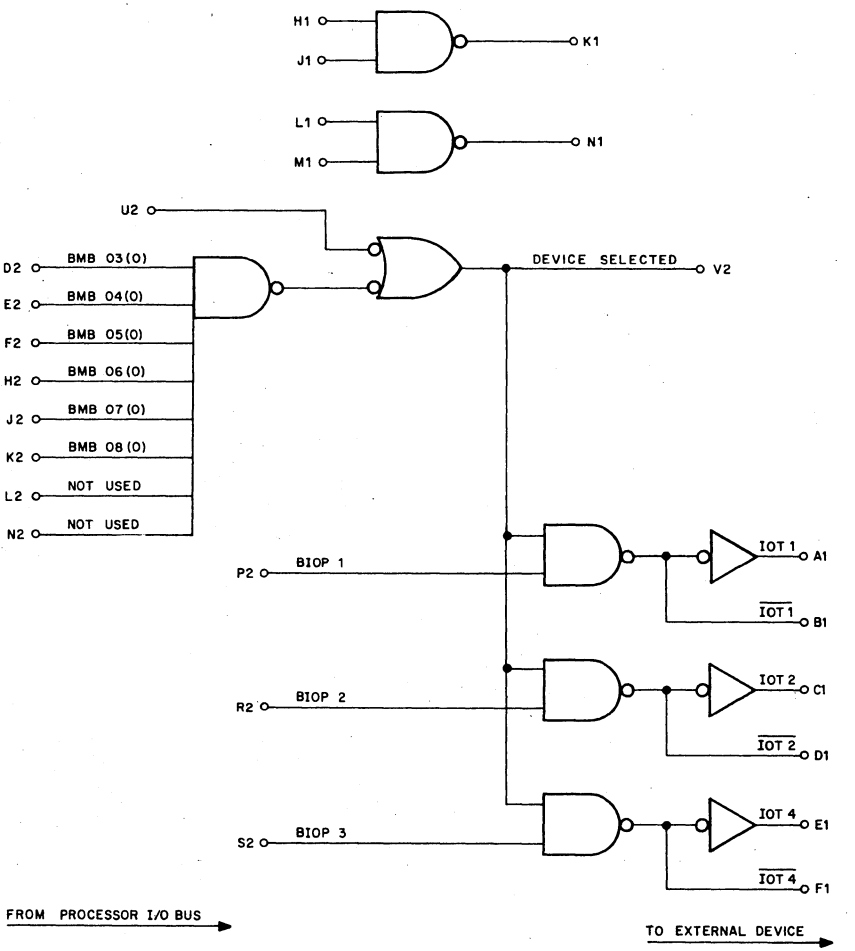


Fig. 47 M103 Device Selector Logic Circuit

M101 — Bus Data Interface
Fifteen two-input NAND gates with one input of each gate tied to a common line. For use in strobing data off of the PDP-8/L I/O bus. M101 module includes diode protection clamps on input lines so that it may be used directly on the PDP-8/L positive bus.

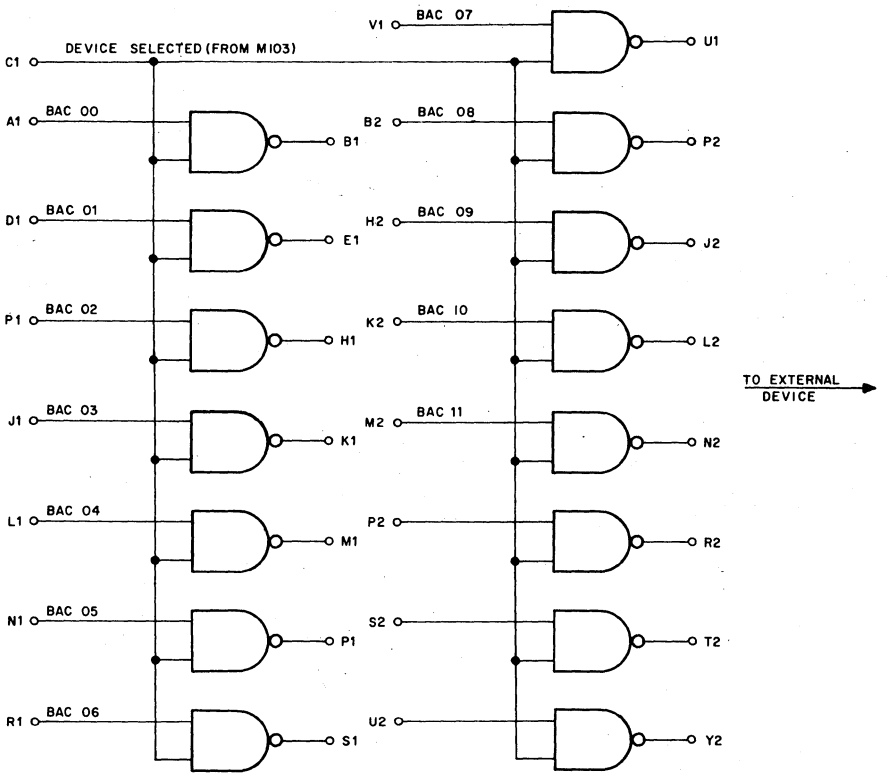


Fig. 48 M101 — Bus Data Interface Logic Circuit

The following is a list of M Series modules available from Digital Equipment Corporation that can be used in designing special interfaces and special devices. The majority of these modules are described in the Digital Logic Handbook. For those that cannot be found in the Handbook, contact the nearest Digital representative.

M SERIES MODULE SUMMARY

Type		Description
M002	15 Loads	Fifteen +3 volt sources each capable of driving ten unit loads. Can be used for tying off unused inputs.
M040	Solenoid Driver	Output ratings of −70 volts and 0.6 amp allow these two drivers to be used with a variety of medium current loads.
M050	Indicator Driver	Output ratings of −20 volts and 50 ma. Allow any of the twelve circuits on this module to drive a variety of incandescent lamps. These drivers can also be used as slow speed open collector PNP level shifters to −3 volt systems.
M101	Bus Data Interface	Fifteen two-input NAND gates with one input of each gate tied to a common line. For use in strobing data off of the PDP8/I or 8/L I/O bus. Pin compatible with M111.
M103	Device Selector	Similar to W103 only for use with PDP8/I and 8/L options. Output pulse are not regenerated but only buffered.
M111	Inverters	Sixteen inverter circuits with a fan-in of one unit load and fan-out of ten unit loads.
M112	NOR Gates	Ten positive NOR gates with a fan-in of one unit load and fan-out of ten unit loads.
M113	NAND Gates	Ten two input positive NAND gates with a fan-in of one unit load and fan-out of ten unit loads.
M115	NAND Gates	Eight three-input positive NAND gates with a fan-in of one unit load and a fan-out of ten unit loads.
M117	NAND Gates	Six four-input positive NAND gates with a fan-in of one unit load and a fan-out of ten unit loads.
M119	NAND Gates	Three eight-input positive NAND gates with a fan-in of one unit load and a fan-out of ten unit loads.
M121	AND/NOR Gates	Six gates which perform the positive logic function AB + CD. Fan-in on each input is one unit load and gate fan-out is ten unit loads.
M141	NAND/OR Gates	Twelve two input positive NAND gates which can be used in a wired OR manner. Gates are grouped in a 4-4-3-1 configuration with a fan-in of one unit load and a fan-out which depends on the number of gates ORed together.

M160	AND/NOR GATES	Three general purpose multi-input gates which can be used for system input selection. Fan-in is one unit load and fan-out is ten unit loads.
M161	Binary to Octal/ Decimal Decoder	A binary to eight line or BCD to ten line decoder. Gating is provided so that up to six binary bits can be decoded using only M161's. Accepts a variety of BCD codes.
M162	Parity Circuit	Two circuits each of which can be used to generate even or odd parity signals for four bits of binary input.
M169	Gating Module	Four circuits which can be used for input selection. Each circuit is of an AND/OR configuration with four two input AND gates.
M202	Triple J-K Flip Flop	Three J-K flip flops with multiple input AND gates on J and K. Versatile units for many control or counter purposes. All direct set and clear inputs are available at module pins.
M203	Set-Reset-Flip-Flops	Eight single-input set-reset flip-flops for use as buffer storage. Each circuit has a fan-in of one unit load and a fan-out of ten unit loads.
M204	General Purpose Buffer and Counter	Four JK flip-flops which can be interconnected as a ripple or synchronous counter or used as general control elements.
M206	General Purpose Flip-Flops	Six D type flip-flops which can be used in shift registers, counters, buffer registers and general purpose control functions.
M207	General Purpose Flip-Flops	Six single input J and K type flip-flops for use in shift-registers, ripple counters, and general purpose control functions.
M208	8-Bit Buffer/ Shift Register	An internally connected 8-bit buffer or shift register. Provisions are made for gated single-ended parallel load, bipolar parallel output, and serial input.
M211	Binary Up/Down Counter	A six bit binary up/down ripple counter with control gates for direction changes via a single control line.
M212	Left-Right Shift Register	An internally connected left-right shift register. Provisions are made for gated single-ended parallel load, bipolar parallel output, and serial input.
M213	BCD Up/Down Counter	One decade of 8421 up or down counting is possible with this module. Provisions are made for parallel loading, bipolar output and carry features.
M220	Major Register	PDP8/I major registers with gating for selected input single-ended parallel loading of each of the four registers, left-right shifting, and adding. Two bits of each register are provided.

M230	Binary to BCD Shift Register Converter	One decade of a modified shift register which allows high speed conversion (100nsec per binary bit) of binary data to 8421 BCD code. System use of this module requires additional modules.
M302	Dual Delay Multivibrator	Two pulse or level triggered one-shot delays with output delay adjustable from 50 nsec to 7.5 msec. Fan-in is 2.5 unit loads and fan-out is 25 unit loads.
M310	Delay Line	Fixed tapped delay line with delay adjustable in 50 nsec increments from 50 nsec to 500 nsec. Two digital output amplifiers and one driver are included.
M360	Variable Delay	Continuously variable delay line with a range of 50 nsec to 500 nsec. Module includes delay line drivers and digital output amplifiers.
M401	Variable Clock	A gateable RC clock with both positive and negative pulse outputs. The output frequency is adjustable from 10MHz to below 100Hz.
M405	Crystal Clock	Stable system clock frequencies from 5KHz to 10MHz are available with this module. Frequency drift at either the positive or negative pulse output is less than .01% of the specified frequency.
M410	Resonant Reed Clock	A stable low frequency reed controlled clock similar to the M452. Stability in the range 0°C to 70°C is better than .15%. For use with communications systems and available with only standard teletype and dataset frequencies.
M452	Teletype Clock	Provides 880Hz, 440Hz, and 220Hz square waves necessary for clocking the M706 and M707 in an 110 baud teletype system.
M501	Schmitt Trigger	Provides the regenerative characteristics necessary for switch filtering, pulse shaping, and contact closure sensing. This circuit can be AND/OR expanded.
M502	Negative Input Converter	Pulses as short as 35 nsec can be level shifted from —3 volt systems to standard M Series levels by the two circuits in this converter. This module can also drive low impedance terminated cables.
M506	Negative Input Converter	This converter will level shift pulses as short as 100 nsec from —3 volt systems to M Series levels. Each of the six circuits on this module provide a low impedance output for driving unterminated long lines.
M507	Bus Converter	Six inverting level shifters which accept —3 and GRD, as inputs and have an open collector NPN transistor at the output. Output rise is delayed by 100nsec. for pulse spreading.

M516	Positive Bus Receiver	Six four input NAND gates with overshoot and undershoot clamps on one input of each gate. In addition, one input of each gate is tied to +3 volts with the lead brought out to a connector pin.
M602	Pulse Amplifier	The two pulse amplifiers in this module provide standard 50 nsec or 110 nsec pulses for M Series systems.
M617	Four Input Power NAND Gate	Six four-input positive NAND gates with a fan-in of one unit load and a fan-out of 30 unit loads.
M623	Bus Driver	Twelve circuits organized in a manner similar to two R123 gates. Input gates are negative NAND's, and the open collector NPN outputs can drive 100ma at ground.
M627	NAND Power Amplifier	Six four-input high speed positive NAND gates with a fan-in of 2.5 unit loads and a fan-out of 40 unit loads.
M650	Negative Output Converter	The three non-inverting level shifters on this module can be used to interface the positive levels or pulses (duration greater than 100 nsec) of K and M Series to —3 volt logic systems.
M652	Negative Output Converter	These two circuits provide high-speed non-inverting level shifting for pulses as short as 35 nsec or levels from M Series to —3 volt systems. The output can drive low impedance terminated cables.
M660	Bus Driver	Three circuits which provide low impedance 100 ohm terminated cable driving capability using M Series levels or pulses of duration greater than 100 nsec. Output drive capability is 50ma at +3 volts or ground.
M661	Positive Level Driver	Three circuits which provide low impedance unterminated cable driving. Characteristics are similar to M660 with the exception that +3 volts drive is 5ma.
M700	Manual Timing Generator	Five phase clocking signals for manual system operation are provided by this module. Also included is a switch filter.
M701	Display Control	Interface logic and Z axis control for DIGITAL display types VC8/I and 30 N is provided by the circuitry on the modules. It does not include any D-A's or deflection voltage amplifiers.
M703	Power Fail	Loss of power conditions on the PDP8/I are detected by the circuitry on this module. The necessary signals for memory protection and machine restart are also included.
M704	Plotter Control	Interface logic to the PDP8/I timing signals and control logic for several models of Calcomp Incremental Plotters are provided by this module.

M705	Byte Input Control	Interface logic to the PDP8/I, timing logic and output buffering to DIGITAL's high speed paper tape reader are included in this module.
M706	Teletype Receiver	Converts asynchronous five or eight bit character teletype code to parallel form. All control less basic timing is provided.
M707	Teletype Transmitter	Converts parallel five or eight bit data to asynchronous teletype code. All control less basic timing is provided.
M710	Byte Output Control	Interface logic for the PDP8/I, timing logic, and an input buffer for an 8 level paper tape punch or similar devices are included in this module.
M715	Reader Clock	This module provides the basic timing signals necessary for use with the M705 Reader Control.
M730	8/I Bus Positive Output Interfacer	General purpose positive bus output module for use in interfacing many positive level (0 to +20 volt) systems to the PDP8/I or 8/L. Module includes device selector, 12 bit parallel output buffer, and adjustable timing pulses.
M731	8/I Bus Negative Output Interfacer	Identical to M730 except outputs are level shifted for 0 to —20 volt negative level systems.
M732	8/I Bus Positive Input Interfacer	General purpose positive bus input module for use in interfacing many positive level (0 to +20 volt) systems to the PDP8/I or 8/L. Module includes device selector, 12 bit parallel input buffer, and adjustable timing pulses.
M733	8/I Bus Negative Input Interfacer	Identical to M732 except inputs are level shifted from negative voltage systems.
M901	Flexprint Cable Connector	Double-sided 36 pin flexprint cable connector. All pins are available for signals or grounds. Pins A2, B2, U1, and V1 have 10Ω resistors in series.
M902	Resistor Terminator	Double-sided 36 pin terminator module with 100Ω terminations on signal leads. Alternate grounds are provided as in the M903 and M904.
M903	Connector	Double-sided 36 pin flexprint cable connector with alternate grounds for I/O bus cables.
M904	Connector	Double-sided pin coaxial cable connector with alternate grounds for I/O bus cables.
M906	Cable Terminator	18 load resistors clamped to prevent excursions beyond +3 volts and ground. It may be used in conjunction with the M623 to provide cable driving ability.

CHAPTER 12
DESIGNING AND CONSTRUCTING INTERFACE EQUIPMENT

This section will provide the interface designer with additional information on design procedures, module layout, wiring, and cable selection. Additional help may be obtained from local DEC sales offices.

PHYSICAL:

The PDP-8/L was designed to provide the user maximum ease and flexibility in implementing special interfaces. External devices and interfaces are constructed and mounted outside of the basic machine, thereby eliminating the necessity for modifications to the basic processor. All signals to and from the computer are carried on coaxial or flexprint cables.

To implement several devices, the cables parallel connect each peripheral in a serial type form (see Figure 49). Three dual cables are used for program interrupt cable connections in (or out). Two additional dual cables are used for a total of five, when Data Break devices are implemented. The PDP-8/L requires the KD8/L Data Break option when a Data Break device is connected on the I/O bus.

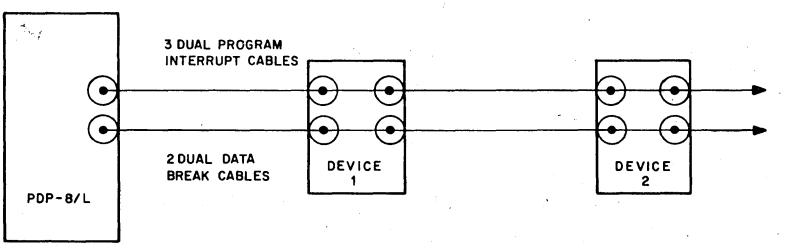


Fig. 49 I/O Bus Configuration

MODULE LAYOUT

In general, module layout is done based on the functional elements within a system and is primarily a matter of common sense.

Digital has, however, layout conventions for I/O cabling to extend devices. The interface designer may wish to use these conventions as a guide. The general rule is DO NOT DEAD END THE I/O BUS. This means that parallel connections should always be made at each device to handle possible future expansion.

Figure 50 shows the I/O cable connections in an option mounting panel. Module slot locations 1 through 3 (looking at the wiring pin side) in an option mounting panel) are reserved for program interrupt cable connections in (or out). Module slot locations 4 to 5 are reserved for Data Break cable connections in (or out).

Module slot locations 1 through 5 in the bottom half of the option mounting panel are wired in parallel with the top module slot locations 1 through 5. To continue the I/O cabling to the next device, the bottom slots are used and the I/O cable connections are exactly the same as mentioned above.

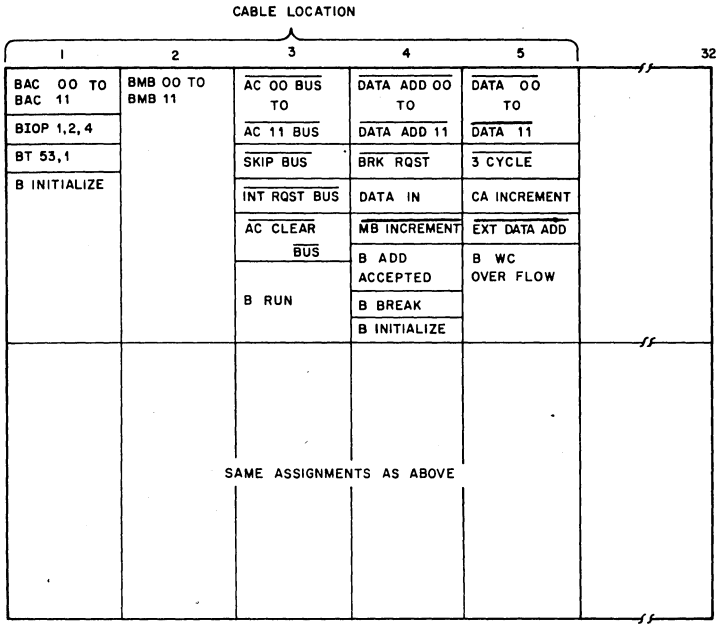


Fig. 50 I/O Cable Connections

CABLE SELECTION

Two types of cables are recommended for I/O interface connections.
One is 9 conductor coax cable. This cable protects systems from radiated noise and cross talk between individual lines. Coax cable used and sold by Digital has the following nominal specs:

Z_o = 0.095 ohm/foot nominal
C = 95 ± 5
L = 13.75 pf/foot approx. (unterminated)
R = 124 Nhy/foot approx.
Y = 79% of velocity of light, approx. (1.5 nsec/ft.

The other cable is 19 conductor (9 signals and 10 grounds), # 30 gauge flat copper flexprint cable.
The total length of I/O cabling, from the PDP-8/L to the last device, can be a maximum of 50 feet. This can be 50 feet of coax or a combination of coax and flexprint, in which case the flexprint cannot exceed a total of 15 feet.

Connector Selection

Of the many connectors available in the module product line, several have particular application to I/O connectors. Price and ordering information is available on these and other connectors in the Digital Logic Handbook. Of particular interest are the following:

M903 Connector

Double-sided 36 pin flexprint cable connector with alternate grounds for I/O bus cables. (Two flexprint cables are utilized with this connector module.)

M904 Connector
Double-sided 36 pin coaxial cable connector with alternate grounds for I/O bus cables. (Two coax cables are utilized with this connector module.)

Signals:
B1, D1, E1, H1, J1, L1, M1, P1, S1,
D2, E2, H2, K2, M2, P2, S2, T2, V2

Grounds:
A1, C1, F1, K1, N1, R1, T1,
C2, F2, J2, L2, N2, R2, V2

Signal Terminating
The G717 module is used for terminating the following signals:
IOP 1, IOP 2, IOP 4, TS 3, TS 1.
This module contains five 100 ohm terminating resistors and should be located in the last device of the I/O cabling scheme.

Wiring Hints
These suggestions may help reduce mounting panel wiring time. They are not intended to replace any special wiring instructions given on individual module data sheets or in application notes. For fastest and neatest wiring, the following order is recommended.

(1) All power wiring (Pins A2, B2, C2, T1) and any horizontally bussed signal wiring. Use Horizontal Bussing Strips, Type 933. (Pin B2 is bussed with —15 V for modules requiring —15 V.)
(2) Vertical grounding wires interconnecting each chassis ground with pins C2 & T1 grounds. Start these wires at the uppermost mounting panel and continue to the bottom panel. On the first and last blocks of the mounting panel, connect the grounds to the chassis.
(3) All other ground wires. Always use the nearest ground pin, unless a special grounding pin has been provided in the module.
(4) Wire all signal wires in convenient order. Point-to-point wiring produces the shortest wire lengths, goes in the fastest, is easiest to trace and change, and generally results in better appearance and performance than cabled wiring. Point-to-point wiring is strongly urged.

The recommended wire size for use with the H803 mounting blocks and H911 mounting panel is #30. Larger or smaller wire may be used depending on the number of connections to be made to each lug. Solid wire and a heat resistant insulation (Kynar) is recommended. The H803 mounting blocks are only available with wire wrap pins which necessitates the use of a wire wrap tool. (Digital can supply #30 gauge wire in 1000 foot rolls.)

ADEQUATE GROUNDING IS ESSENTIAL. IN ADDITION TO THE CONNECTIONS BETWEEN MOUNTING PANELS MENTIONED ABOVE, THERE MUST BE CONTINUITY OF GROUNDS BETWEEN CABINETS AND BETWEEN THE LOGIC ASSEMBLY AND ANY EQUIPMENT WITH WHICH THE LOGIC COMMUNICATES.

When wire wrapping is done on a mounting panel containing modules, the wire wrap tool should be grounded except when all modules are removed from the mounting panel. This procedure should be followed, because even with completely isolated tools, such as those operated by batteries or compressed air, a static charge can often build up and burn out semiconductors.

Cooling
The low power consumption of M Series modules results in a total of about 15 watts dissipation in a typical H911 mounting panel with 64 modules. Convection cooling is sufficient for a few mounting panels, but forced air cooling should be used when a very large system is built.

IOT ALLOCATIONS

IOT	OPTION
*00	Interrupt
01	High Speed Reader Type PR8/L
02	High Speed Punch Type PP8/L
*03	Teletype Keyboard/Reader
*04	Teletype Teleprinter/Punch
05	Displays, Types VC8/I and KV8/I
06	Displays, Types VC8/I and KV8/I
07	Displays, Types VC8/I, KV8/I, and Light Pen Type 370
10	Memory Parity Option MP8/L and Power Fail Option KP8/L
11	Teletype System Type PT08
12	Teletype System Type PT08
13	Rea Time Clock Type KW8/I
14	
15	
16	
17	
20	Memory Extension Control Option Type MC8/L
21	Memory Extension Control Option Type MC8/L
22	Memory Extension Control Option Type MC8/L
23	Memory Extension Control Option Type MC8/L
24	Memory Extension Control Option Type MC8/L
25	Memory Extension Control Option Type MC8/L
26	Memory Extension Control Option Type MC8/L
27	Memory Extension Control Option Type MC8/L
30	
31	
32	
33	
34	
35	
36	
37	
40	Teletype System Type PT08 and 680 Communications System
41	Teletype System Type PT08 and 680 Communications System
42	Teletype System Type PT08 and 680 Communications System
43	Teletype System Type PT08 and 680 Communications System
44	Teletype System Type PT08 and 680 Communications System
45	Teletype System Type PT08
46	Teletype System Type PT08
47	Teletype System Type PT08
50	Incremental Plotter Type VP8/I
51	Incremental Plotter Type VP8/I
52	Incremental Plotter Type VP8/I
53	General Purpose A/D Converters and Multiplexers, Types AF01A, AM08, AM02A, AM03A and AF04A Scanning Digital Voltmeter
54	General Purpose A/D Converters and Multiplexers, Types AF01A, AM08, AM02A, AM03A and AF04A Scanning Digital Voltmeter
55	D/A Converter Type AA01A
56	D/A Converter Type AA01A
57	D/A Converter Type AA01A, Sample and Hold Control Type AC01A and AF04A Scanning Digital Voltmeter

IOT	OPTION
60	Random Access Disk File and Control Type DF32 and Synchronous Modem Interface Type DPØ1A
61	Random Access Disk File and Control Type DF32 and Synchronous Modem Interface Type DPØ1A
62	Random Access Disk File and Control Type DF32 and Synchronous Modem Interface Type DPØ1A
63	Card Reader Type CR8/I
64	Synchronous Modem Interface Type DPØ1A
65	Synchronous Modem Interface Type DPØ1A
66	Synchronous Modem Interface Type DPØ1A
67	Card Reader Type CR8/I and Synchronous Modem Interface Type DPØ1A
70	Automatic Mag Tape Type TC58
71	Automatic Mag Tape Type TC58
72	Automatic Mag Tape Type TC58
73	Automatic Mag Tape Type TC58
74	Automatic Mag Tape Type TC58
75	
76	DECtape Control TC01
77	DECtape Control TC01

*These devices are not available for customer options.

CHAPTER 13

INTERFACE CONNECTIONS

All interface connections to the PDP-8/L are made at assigned module receptacle connectors in the mounting frame. Capital letters designate horizontal rows of modules within a mounting frame from top to bottom. Module receptacles are numbered from left to right as viewed from the wiring side (right to left from the module side). Terminals of a connector or module are assigned capital letters from top to bottom, omitting G, I, O, and Q. Double-sided connectors or modules are used with the suffix number "1" used to designate the left side and suffix number "2" used to designate the right side.

The module receptacles and assigned use for interface signal connections are:

RECEPTACLE	SIGNAL USE
D34	$\overline{\text{AC 0-11}}$, $\overline{\text{SKIP}}$ $\overline{\text{Interrupt Request}}$, $\overline{\text{Clear AC}}$ inputs, and B RUN output
D35	BMB 0-11 outputs
D36	BAC 0-11, IOT's, BTS1, BTS3, and B Initialize outputs
C35	$\overline{\text{DATA BIT 0-11}}$, $\overline{\text{3-Cycle Select}}$, $\overline{\text{Increment CA}}$, $\overline{\text{EXT DATA ADD}}$ inputs, and BWC overflow output
C36	$\overline{\text{DATA ADDRESS 0-11}}$, $\overline{\text{Break Request}}$, $\overline{\text{Transfer Out}}$, $\overline{\text{Increment MB}}$ inputs, B Break, B Address accepted and B Initialize outputs

Terminals A1, C1, F1, K1, N1, R1, T1, C2, F2, J2, L2, N2, R2. and U2 of these receptacles are grounded within the computer and terminals B1, D1, E1, H1, J1, L1, M1, P1, S1, D2, E2, H2, K2, M2, P2, S2, T2, and V2 carry signals. Terminal A2 is +5 VDC and terminal B2 is −15 VDC. These terminals mate with either M903 or M904 Cable Connectors.

Interface connection to the PDP-8/L can be established for all peripheral equipment by making series cable connections between devices. In this manner only one set of cables is connected to the computer and two sets are connected to each device: one receiving the computer connection from the computer itself or the previous device, and one passing the connection to the next device. Where physical location of equipment does not make series bus connections feasible, or when cable length becomes excessive, additional interface connectors can be provided near the computer. All logic signals passing between the PDP-8/L and input/output equipment are positive voltage levels, allowing direct TTL logic interface with appropriate diode clamp protection.

Positive level for a low logic state is 0 to 0.4 volts. Positive level for a high logic state is +2.4 to +3.6 volts.

The following tables present cable connections for the PDP-8/L interface signals, with the noted logic states true when the signal level is high.

TABLE
POSITIVE 8/L BUS OUTPUT SIGNALS

Signals	Interface Connection
BAC00 (1)	D36B1
BAC01 (1)	D36D1
BAC02 (1)	D36E1
BAC03 (1)	D36H1
BAC04 (1)	D36J1
BAC05 (1)	D36L1
BAC06 (1)	D36M1
BAC07 (1)	D36P1
BAC08 (1)	D36S1
BAC09 (1)	D36D2
BAC10 (1)	D36E2
BAC11 (1)	D36H2
BIOP1	D36K2
BIOP2	D36M2
BIOP4	D36P2
BTS3 (1)	D36S2
BTS1 (1)	D36T2
B Initialize	D36V2
B Initialize	C36V2
BMB00 (1)	D35B1
BMB01 (1)	D35D1
BMB02 (1)	D35E1
BMB03 (0)	D35H1
BMB03 (1)	D35J1
BMB04 (0)	D35L1
BMB04 (1)	D35M1
BMB05 (0)	D35P1
BMB05 (1)	D35S1
BMB06 (0)	D35D2
BMB06 (1)	D35E2
BMB07 (0)	D35H2
BMB07 (1)	D35K2
BMB08 (0)	D35M2
BMB08 (1)	D35P2
BMB09 (1)	D35S2
BMB10 (1)	D35T2
BMB11 (1)	D35V2
B Run (0)	D34S2
BWC Overflow (0)	C35P2
B Add Accepted (0)	C36S2
B Break (0)	C36P2

TABLE
POSITIVE 8/L BUS INPUT SIGNALS

Signals	Interface Connection
AC00 BUS	D34B1
AC01 BUS	D34D1
AC02 BUS	D34E1
AC03 BUS	D34H1
AC04 BUS	D34J1
AC05 BUS	D34L1
AC06 BUS	D34M1
AC07 BUS	D34P1
AC08 BUS	D34S1
AC09 BUS	D34D2
AC10 BUS	D34E2
AC11 BUS	D34H2
SKIP BUS	D34K2
INT RQST BUS	D34M2
AC CLEAR BUS	D34P2
*	D34T2
*	D34V2
DATA ADD 00	C36B1
DATA ADD 01	C36D1
DATA ADD 02	C36E1
DATA ADD 03	C36H1
DATA ADD 04	C36J1
DATA ADD 05	C36L1
DATA ADD 06	C36M1
DATA ADD 07	C36P1
DATA ADD 08	C36S1
DATA ADD 09	C36D2
DATA ADD 10	C36E2
DATA ADD 11	C36H2
BRK RQST	C36K2
DATA IN	C36M2
MB INCREMENT	C36T2
DATA 00	C35B1
DATA 01	C35D1
DATA 02	C35E1
DATA 03	C35H1
DATA 04	C35J1
DATA 05	C35L1
DATA 06	C35M1
DATA 07	C35P1
DATA 08	C35S1
DATA 09	C35D2
DATA 10	C35E2
DATA 11	C35H2
3 CYCLE	C35K2
CA INCREMENT	C35M2
EXT DATA ADD	C35S2
*	C35T2
*	C35V2

* Not used in PDP-8/L but reserved for specific use in PDP-8/I.

The following input and output signal connections are available for use with DEC equipment options or for use in special interface equipment designed by the customer. Refer to Tables 1 and 2 for interface connections.

EXTENDED DATA ADDRESS INPUT AND DATA FIELD OUTPUT

When the Memory Extension Control Type MC-8/L is in the computer system, devices using the data break facility must supply a 12-bit data address and a 1-bit extended data address. Conversely, the programmed transfer of an address to a register in a device that uses the data break occurs as a 12-bit word from the accumulator and 1-bit data field extension from the MC-8/L. The Extended Data Address signal must be at ground potential to designate a binary 1 and+3V to designate a binary 0. If the MC8/L is incorporated in the system and no EXT DATA ADD is supplied, then field zero will always be selected.

B RUN OUTPUT SIGNAL

The output of the RUN flip-flop flows to external equipment through the interface circuits. This signal is at ground potential when the computer is performing instructions and is at +3V when the program halts. Magnetic tape and DEC tape equipment use this signal to stop transport motion when the PDP-8L halts, preventing the tape from running off the end of the reel.

BTS1 AND BTS3 OUTPUT PULSES

Two buffered timing pulse signals, designated BTS1 and BTS3, are supplied to I/O devices. These signals can synchronize operations in external equipment with those in the computer. The BTS1 and BTS3 pulse signals are derived from the TS1 and TS3 signals generated by the timing signal generator of the PDP-8/L. The Type M660 Bus Driver standardizes the TS1 and TS3 pulses as positive pulses.

INITIALIZE OUTPUT PULSES

The Initialize pulses generated and used within the PDP-8/L are made available at the interface connections. External equipment uses these pulses to clear registers and control logic during the power turn-on period. Use of Initialize pulses in this manner is valid only when the logic circuits cleared by the pulses are energized before or at the same time the PDP-8/L POWER switch is turned on. Operating the KEY START switch also generates the Initialized pulses.

Note that two "B Initialize" signals are provided at the Interface connections. The load on these two lines should be equalized, but the two lines should not be connected together.

CHAPTER 14
INSTALLATION AND PLANNING

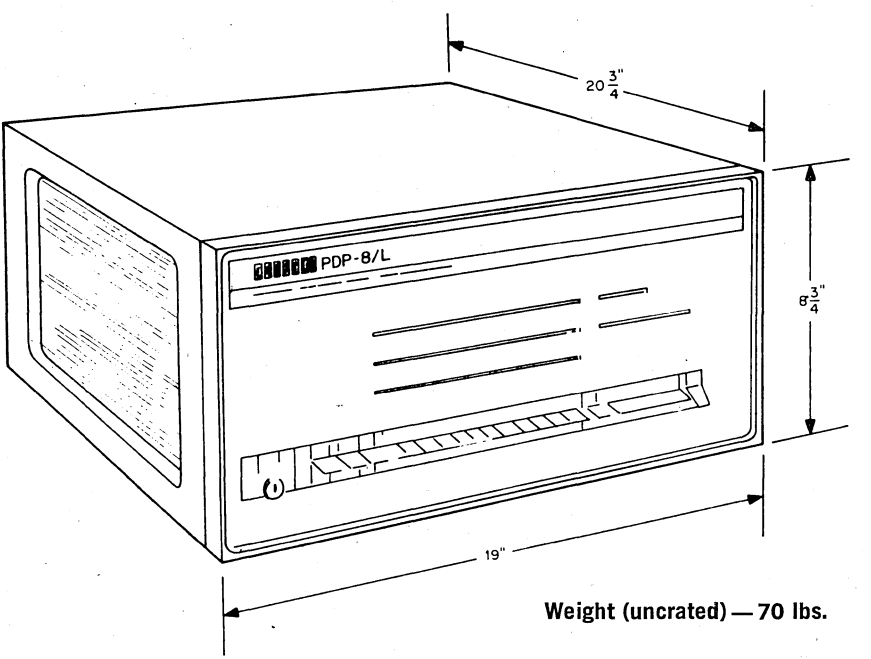
Space Requirements

Access space must be provided at the installation site to accommodate the PDP-8/L and peripheral equipment and to allow access to all doors and panels for maintenance.

The PDP-8/L is available in either a table top configuration or a rack mounted configuration. The rack mounted configuration and peripherals may be purchased completely installed in DEC cabinets or may be purchased unmounted for installation into the customer cabinet.

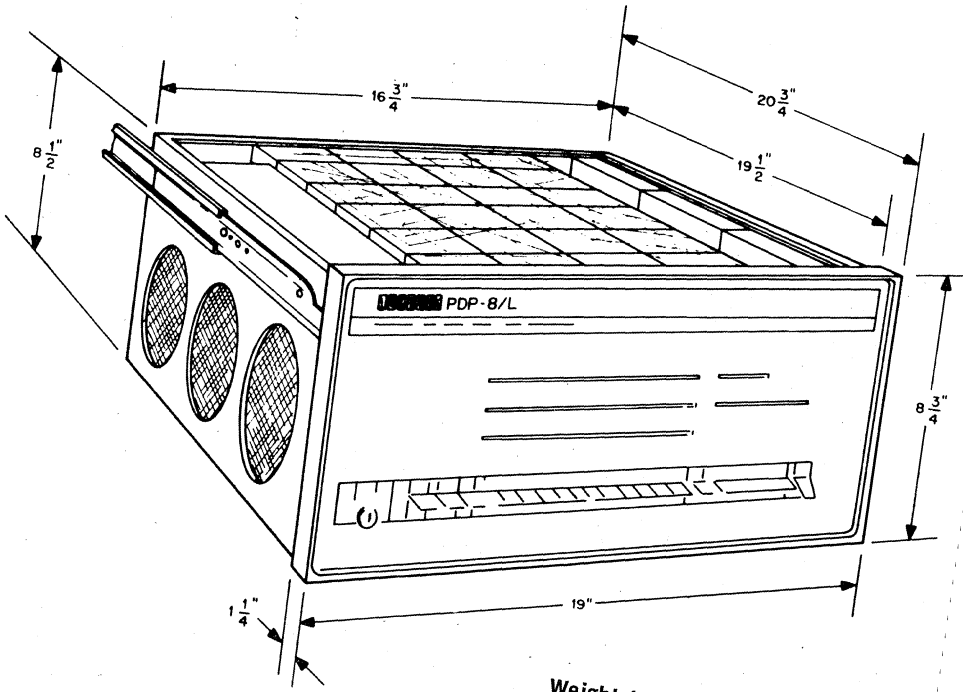
Minimum service clearance on all standard DEC computer cabinets is 8¾ inches at the front and 14⅞ inches at the back.

The standard Teletype automatic send receive set requires floor space approximately 22¼ inches wide by 18½ inches deep. Signal cable length restricts the location of the Teletype to within 8 feet of the side of the computer.



Weight (uncrated) — 70 lbs.

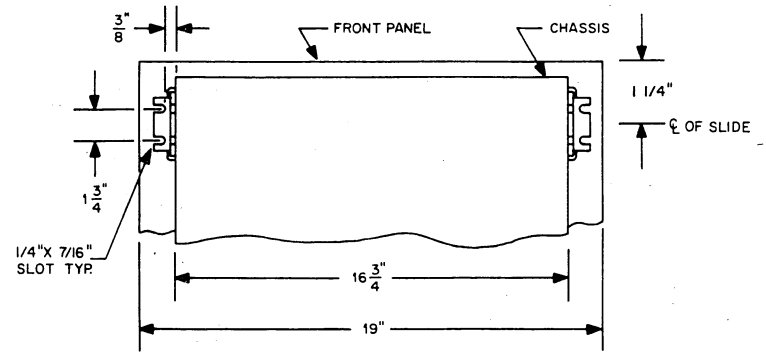
Table Top PDP-8/L Dimensions

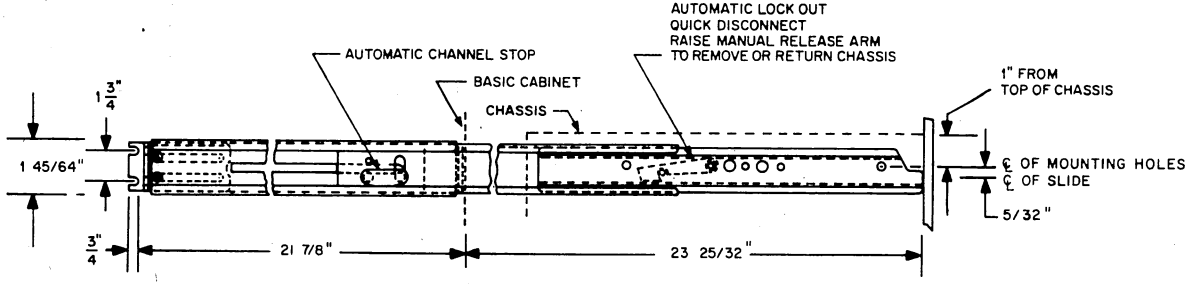


Weight (uncrated) — 70 Lbs.

Rack Mountable PDP/8L Dimensions

REAR VIEW OF
MOUNTING HARDWARE





SIDE VIEW OF
MOUNTING HARDWARE

ENVIRONMENTAL REQUIREMENTS

Ambient temperature at the installation site can vary between 50 and 130F (between 10 and 55C). During shipping or storing of the system, the ambient temperature may vary between —4 and 150F (between —20 and 65C). Humidity can vary between 10 and 90%.

POWER REQUIREMENTS (PDP-8/L)

Line Voltage Input — 115 VAC (Single Phase) +15 VAC, —10 VAC
Line Frequency — 47-63 Hertz
Line Current Drain — 2.5 Amperes
Power Dissipation — 250 Watts

A 15 ampere 3 prong, U-ground male connector is supplied on the rear of the PDP-8/L chassis for connection to the power source. For operation of the PDP-8/L on a 220V power source, a step-down transformer is required and can be supplied as an option.

POWER REQUIREMENTS (TELETYPE — MODEL ASR-33)

Line Voltage Input — 115 VAC (Single Phase) ± 10%
Line Frequency — 60 Hz ± 0.45 Hz (50 Hz ± 0.75 Hz)
Line Current Drain — 2.0 Amperes
Power Dissipation — 150 Watts

The Teletype plugs into the rear of the PDP-8/L chassis and turns ON or OFF by operation of the power ON/OFF switch on the front panel of the PDP-8/L.

CABLE REQUIREMENTS

Eighteen conductor coaxial cables or flexprint cables with either M904 or M903 cable connectors respectively provide signal connection between the computer and optional equipment. These cables are connected by plugging the connectors into standard FLIP-CHIP module receptacles.

On large systems, operation of the ON/OFF switch on the PDP-8/L front panel operates a power control panel that connects or disconnects 115 VAC power to the peripheral equipment.

Cables connect to cabinets through a drop panel in the bottom of cabinets. Subflooring is not necessary because casters elevate the cabinets from the floor to afford sufficient cable clearance.

INSTALLATION PROCEDURE

During system check-out, customers are invited to visit the Maynard manufacturing facility to inspect and become familiar with their equipment. Computer customers may also send personnel to instruction courses on computer operation, programming, and maintenance conducted regularly in Maynard, Massachusetts.

DEC's engineers are available during installation and test for assistance or consultation. Further technical assistance in the field is provided by home office design engineers or branch office application engineers.

APPENDIX 1
PROGRAM ABSTRACTS

FAMILY-OF-8 PROGRAMS

The PDP-8/L is delivered to the user complete with an extensive selection of system programs and routines making the full data processing capability of the new computer immediately available to each user, eliminating many commonly experienced initial programming delays.

The programs described in these abstracts come from two sources, past programming efforts on the PDP-5, 8, 8/S, 8/I, and present and continuing programming effort on these same machines plus the PDP-8/L. Thus the programming system takes advantage of the many man-years of program development and field testing by Digital computer users. There are over 2500 Family-of-8 systems in the field already.

Although in many cases PDP-8/L programs orginated from previous Family-of-8 computers, all utility and functional program documentation is issued anew, recursive format introduced with the Family-of-8 computers. Programs, written by users of Family-of-8 computers and submitted to the DECUS library (DECUS—Digital Equipment Corporation User's Society) are immediately available to Family-of-8 users. Consequently, users of all Family-of-8 computers can take advantage of continuing program developments.

SYSTEM PROGRAMS

DEC-08-AJAB-D
FOCAL

FOCAL (for FOrmula CALculator) is an on-line, conversational, service program for the PDP-8-family of computers, designed to help scientists, engineers, and students solve numerical problems. The language consists of short imperative English statements which are relatively easy to learn. Mathematical expressions are typed, for the most part, in standard notation. No previous programming experience is needed either to understand this manual or to use FOCAL at the Teletype console. However, the best way to learn the FOCAL language is to sit at the Teletype and try the commands, starting with the examples given in the manual.

DEC-08-AZAO-D
FORTRAN II MANUAL

FORTRAN (8K) (acronym for FORmula TRANslation) is used interchangeably to designate both the FORTRAN (8K) language and translator or compiler. The FORTRAN (8K) compiler is a computer program that enables the programmer to express his problem using English words and mathematical statements similar to the language of mathematics and acceptable to the computer. The compiler translates the programmer's source program into symbolic language, and then the symbolic version of the program is translated into relocatable binary code, that is, machine language, the language of the computer. The relocatable binary code, which is output on paper tape, is then loaded into the computer for solution of the problem.

The FORTRAN (8K) system has the following features:

a. USA Standard FORTRAN syntax
b. Subroutines
c. Two levels of subscripting

d. Function subprograms
e. Input-output supervisors
f. Relocatable output loaded by the Linking Loader
g. COMMON statements
h. I, F, E, A, X, and H format specification
i. Arithmetic and trigonometric library subroutines

The FORTRAN (8K) system (hereafter referred to as FORTRAN) consists of the two-pass FORTRAN compiler, Linking Loader, Run-Time Monitor, and a library of subprograms (see the appendices.)
This FORTRAN system requires a PDP-8/I or 8/L computer with two fields of core memory, an ASR-33 teleprinter, and a high-speed paper tape reader and punch.

DEC-08-AFCO-D
FORTRAN Compiler and Operating System
The one pass FORTRAN Compiler and Operating System compiles FORTRAN source language statements into an object program tape. The operating system executes the program. This operating system contains the interpreter, arithmetic function subroutines and input/output packages.

DEC-08-AFA2
Symbol Print
Loaded over the FORTRAN Compiler, this program lists the variables used and where they will be located in core. It also indicates the section of core not used by the compiled program and data.

DEC-08-ASAB-D
PAL III Symbolic Assembler
PAL III is a two pass symbolic machine language assembler which converts programs coded in symbolic machine language to binary machine language. It has an optional third pass to produce a side-by-side octal/symbolic assembly listing. The basic process performed by the assembler is the substitution of numeric values for symbols, according to associations defined in the symbol table. In addition, the user may request that the assembler itself assign values to the user's own symbols at assembly time. These symbols are normally used to name memory locations, which may then be referenced by name.

DEC-08-CDDA-D
DDT-8 (Dynamic Debugging Tape)
DDT-8 provides a means for on-line program debugging at the symbolic or mnemonic level. By typing commands on the console teleprinter, memory locations can be examined and changed, program tapes can be inserted, selected program tapes can be inserted, selected portions of the program can be run, and the updated program can be punched.

DEC-08-CMAA-D
MACRO-8 Assembler
The MACRO-8 symbolic assembler accepts source programs written in symbolic language and translates them into binary form in two passes. MACRO-8 produces an object program tape (binary), a SYMBOL table (for use with DDT), an Octal/Symbolic assembly listing, and useful diagnostic messages. MACRO-8 is compatible with PAL III, and has the following additional features: user defined macros; double precision integers, floating point constants; arithmetic and Boolean operators, literals, text facilities and automatic link generation.

DEC-08-COCO-D
ODT-8 (Octal Debugging Technique)
ODT-8 (Octal Debugging Technique) is a debugging aid, which facilitates communication with, and alteration of, the program being run. Communication between operator and program occurs via the Teletype, using defined commands and octal numbers. ODT-8 is a subset of DDT-8 and occupies three pages of core storage.

The program may be relocated to occupy any three consecutive pages of core.

DEC-08-ESAB-D
Symbolic Editor
The Symbolic Editor allows the user to prepare and edit symbolic tapes on-line in ASCII code with the Teletype and/or high-speed reader/punch. The tedious task of correcting symbolic program tapes using the Teletype off-line is thereby avoided. Proper use of the Symbolic Editor can substantially ease the labor and reduce the number of passes necessary to correct symbolic program tapes. The Editor reads a page, or section, of symbolic tape into a buffer in core storage, where it is available for examination and correction upon keyboard command. The page buffer occupies all of core not taken up by the Editor itself and has a capacity of approximately 6000_{10} characters. When the Editor has finished reading a page into the buffer, a bell rings to signal the user that he may begin editing. The user may then call for a listing of individual (numbered) lines, in any order, and insert desired changes and corrections. In addition, text may be added to the buffer, or inserted between specified lines. Groups of lines or individual lines may be moved or deleted by a single command, or the entire page may be erased if desired. Searches may be made and parts of lines changed without retyping the entire line. Upon keyboard command, the Editor will then either list or punch out the corrected lines or page on paper tape. The Editor can also be used to generate a new symbolic tape by typing new text directly on the keyboard. Errors in typing may be corrected simply by typing a rubout.

DEC-08-YQYA-D
Floating-Point System
Includes Floating-Point Interpreter and I/O subsystems. Allows the programmer to code his problem in floating-point machine language.
Floating-point operations automatically align the binary points of operands, retaining the maximum precision available by discarding leading 0s. In addition to increasing accuracy, floating-point operations relieve the programmer of scaling problems common in fixed-point operations. This system includes elementary function subroutines programmed in floating-point. These subroutines are sine, cosine, square root, logarithm, arc tangent, and exponential functions. Data being processed in floating-point is maintained in three words of memory (12-bit exponent, 24-bit mantissa). An accuracy of seven decimal places is maintained.

Digital-8-16-S-D
Master Tape Duplicator
This program will duplicate and verify 8-channel paper tapes using a high-speed reader and high-speed punch. The program uses the program interrupt and allows both the reader and the punch to operate at maximum speed.
The program accumulates two types of checksums while reading and while punching: (1) the number of nonzero characters on the tape, and (2) the sum of characters on the tape (both are taken modulo 4096).
When duplicating, the program compares the checksums at the end of the tape with the checksums accumulated by the read routine. If these differ, a reader error has occurred and a message is typed.

Digital-08-USBO-P
Multianalyzer Display and Analysis
The two-dimensional pulse-height analysis reads in and analyzes two-parameter energy and spectra data. The program receives and executes commands from the keyboard. These commands start and stop data taking, control the displays, and control writing and punching of data. The displays available are: isometric, vertical and horizontal slicing, differential and integral contours, and "twinkle box." The program is flexible with respect to the dimensions of the data matrix.

Digital-8-15-S-P
Oceanographic Analysis
This program represents the basic accepted physical oceanography method for the reduction of data concerning depth, temperature, and salinity measurements of the water column.
This program allows the field oceanographer a rapid means of immediately calculating Sigma-T, anomaly of specific volume, and sound velocity following a Nansen cast whereby he may examine results in detail to determine the structure of the environment he has just sampled and to check the validity of his measurements.
The program also contains an interpolation routine as well as a depth integration of the anomaly of specific volume.

ELEMENTARY FUNCTION ROUTINES

The following routines are described in the Program Library Math Routines Manual (DEC-08-FFAA-D).

Square Root Subroutine-Single Precision
Forms the square root of a single-precision number. An attempt to take the square root of a negative number will be given 0 for a result.

Signed Multiply Subroutine-Single Precision
Forms a 22-bit signed product from 11-bit signed multiplier and multiplicand.

Signed Divide Subroutine-Single Precision
This routine divides a signed 11-bit divisor into a signed 23-bit dividend giving a signed 11-bit quotient and a remainder of 11 bits with the sign of the dividend.

Double-Precision Multiply Subroutine-Signed
This subroutine multiplies a 23-bit signed multiplicand by a 23-bit signed multiplier and returns with a 46-bit signed product.

Double-Precision Divide Subroutine-Signed
This routine divides a 23-bit signed divisor into a 47-bit signed dividend and returns with a 23-bit signed quotient and a remainder of 23 bits with the sign of the dividend.

Sine Routine-Double Precision
The Double-Precision sine subroutine evaluates the function Sin (X) for $-4<X<4$ (X is in radians). The argument is a double-precision word, 2 bits representing the integer part and 21 bits representing the fractional part. The result is a 23-bit signed fraction $-1<\text{Sin}(X)<1$.

Cosine Routine-Double Precision
This subroutine forms the cosine of a double-precision argument (in radians). The input range is $-4<X<4$.

Four-Word Floating-Point Package
This is a basic floating-point package that carries data as three words of mantissa and one word of exponent. Common arithmetic operations are included as well as basic input/output control. No functions are included.

Logical Subroutines
Subroutines for performing the logical operations of inclusive and exclusive OR are presented as a package.

Shift Right, Shift Left Subroutines (Single and Double Precision)
Four basic subroutines, shift right and shift left, each at both single and double precision, are presented as a package.

Logical Shift Routines
Two basic subroutines, shift right at both single and double precision, are presented as a package. The shifts are logical in nature.

LOADERS

DEC-08-LRAA-D
Read-In Mode (RIM Loader)
The RIM Loader is a minimum routine for reading and storing information contained in read-in-mode coded tapes via the ASR33 Perforated Tape Reader.

DEC-08-LBAA-D
Binary Loader
The Binary Loader is a short routine for reading and storing information contained in binary-coded tapes, using the ASR33 Perforated-Tape Reader and the Type PR8/L High-Speed Perforated Tape Reader.
The Binary Loader accepts tapes prepared by the use of PAL (Program Assembly Language) or MACRO-8. Diagnostic messages may be included on tapes produced when using either PAL or MACRO. The Binary Loader will ignore all diagnostic messages.

DEC-08-LHAA-D
"HELP" Loader
The "HELP" Loader loads the standard version of the RIM and BIN Loaders into the PDP-8/L, in less than 90 s, replacing manual procedures which required several minutes.

DEC-08-LUAA-D
TC01 Bootstrap Loader
This is a bootstrap for loading the PDP-8 DECtape Library System designed for use with DECtape Control Type TC01 with TU55 Tape Transports.

UTILITY PROGRAMS AND SUBROUTINES

DEC-08-PMPO-D
Read-In Mode (RIM) Punch
The RIM Punch program provides a means of punching out information contained in selected blocks of core memory as RIM-coded tape via the ASR33 Perforated Tape Punch or High-Speed Punch. The punch program may occupy either low or high memory depending on the version used.

Digital-8-5-U-Sym-D
Binary Punch (ASR33 or PP8/L)
This program provides a means of punching out information contained in selected blocks of core memory as binary-coded tape via the ASR33 Perforated Tape Punch or via the high-speed punch.

Digital-8-6-U-Sym-D
Octal Memory Dump
This routine will read the console switches twice to obtain the upper and lower limits of an area of memory, then type on the Teletype an absolute address plus the octal contents of the first four words specified and repeat this until the block is exhausted, at which time the user may repeat the operation.

Digital-8-10-U-Sym-D
BCD to Binary Conversion Subroutine
A basic subroutine for converting binary-coded-decimal numbers to their equivalent binary value. Conversion is accomplished by "radix deflation."

Digital-8-11-U-D
Double Precision BCD to Binary Conversion Subroutine
This subroutine converts a 6-digit BCD number to its equivalent binary value in two computer words.

Digital-8-12-U-D
Incremental Plotter Subroutine
This subroutine moves the pen of an incremental plotter to a new position along the best straight line. The pen may be raised or lowered during the motion.

Digital-8-14-U-Sym-D
Binary to BCD Conversion Subroutine
This subroutine provides the basic means of converting binary data to binary-coded-decimal (BCD) data for typeout, magnetic tape recording, etc.

Digital-8-15-U-Sym-D
Binary to BCD Conversion (Four Digit)
This subroutine extends the method used in Digital-8-14-U-Sym so that binary integers from 0 to 4095 contained in a single computer word may be converted to four binary-coded-decimal characters packed in two computer words.

Digital-8-18-U-Sym-D
Alphanumeric Message Typeout
A basic subroutine to type messages packed in computer words. Two 6-bit characters are packed internally in a single word. All ASR33 codes from 301 to 337 and from 240 to 277 (excepting 243 and 245) can be typed. The typing of line-feed (code 212) and carriage-return (code 215) are made possible by arbitrarily assigning internal codes of 43 and 45, respectively, to represent these characters, thus preventing the output of ASCII codes 243 (#1) and 245 (%).

Digital-8-19-U-Sym-D
Teletype Output Subroutines
A group of subroutines useful in controlling ASR33 output is presented as a package. Provision is made for the simulation of tabulation stops. The distance "tabbed" may be controled by the user. Characters whose ASR33 codes are in groups 241 and 277, inclusive, and 300 through 337, inclusive, are legal. Space, carriage return then line feed, and tabulation are provided via subroutines.

Digital-8-20-U-Sym-D
Character String Typeout
A basic subroutine to type messages stored internally as a string of coded characters. All ASR33 characters are legal.

Digital-8-21-U-Sym-D
Symbolic Tape Format Generator
The format generator allows the user to create symbolic tapes with formatting. It may be used to condense tapes with spaces by inserting tabs, or merely to align tabs, instructions, and comments.

Digital-8-22-U-Sym-D
Unsigned Decimal Print
This subroutine permits the typeout of the contents of a computer word as a 4-digit, positive, decimal integer.

Digital-8-23-U-Sym-D
Signed Decimal Print—Single Precision
This subroutine permits the typeout of the contents of a computer word as a signed 2s complement number. If bit 0 of the computer word is a 1, the remaining bits represent a negative integer in 2s complement form; if bit 0 equals 0, the remaining bits represent a positive integer. If the number is negative, a minus sign is printed; if positive, a space.

Digital-8-24-U-Sym-D
Unsigned Decimal Print, Double Precision
This subroutine permits the typeout of a double-precision integer stored in the usual convention for double-precision numbers. The one exception is that all 24 bits are interpreted as magnitude bits (i.e., the bit 0 of the high-order word is not a sign bit). The typeout is in the form of a 7-digit, positive, decimal integer.

Digital-8-25-U-Sym-D
Signed Decimal Print, Double Precision
This subroutine permits the typeout of the contents of two consecutive computer words as one signed, double-precision, 2s complement number. If bit 0 of the high-order word is a 1, the remaining 23 bits represent a negative integer in 2s complement form; if bit 0 equals 0, the remaining bits represent a positive integer. If the number is negative, a minus sign is printed; if positive, a space.

Digital-8-28-U-Sym-D
Single Precision Decimal to Binary Conversion and Typeout ASR333, Signed or Unsigned
This routine accepts a string of up to four decimal digits (single precision for the PDP-8/L) from the Teletype keyboard and converts it to the corresponding 2s complement binary number.
The string may contain as legal characters a sign (+, —, or space) and the digits from 0 through 9. If the first legal character is not a sign, the conversion is unsigned. A back arrow (←) at any point in the string erases the current string and allows the operator to reenter the correct value. Any character after the first, other than another digit or back arrow causes the conversion to terminate and is found in location SISAVE within the subroutine.

Digital-8-29-U-Sym-D
Double Precision Decimal to Binary Conversion and Typeout (ASR33), Signed or Unsigned
This routine accepts a string of up to eight decimal digits (double-precision for the PDP-8/L) from the Teletype keyboard and converts it to the corresponding 2's complement binary number.
The string may contain as legal characters a sign (+, —, or space) and the digits 0 through 9. If the first legal character is not a sign, the conversion is unsigned. A back-arrow (←) at any point in the string erases the current string and allows the operator to reenter the value. Any character after the first, other than another digit or "back-arrow," causes the conversion to terminate and is found in location DIDSAV within the subroutine.

DECTAPE SYSTEM SOFTWARE

DEC-08-SUBO-D
DECtape Programming Manual
The DECtape Library System is loaded by a 17_{10} instruction bootstrap routine that starts at 7600_8. This loader calls a larger program into the last memory page, whose function is to preserve on tape the contents of memory from 6000_8-7577_8, and then to load the INDEX program and the directory into those same locations. Since the information in this area of memory has been preserved, it can be restored when operations have been completed. The skeleton system tape contains the following programs:

INDEX—Typing this causes the names of all programs currently on file to be typed out.

UPDATE—Allows the user to add a new program to the files. UPDATE queries the operator about the program's name, its starting address, and its location in core memory.

GETSYS—Generates a skeleton library tape on a specified DECtape unit.

DELETE—Causes a named file to be deleted from the tape.

Starting the skeleton library tape, the user can build up a complete file of his active programs and continuously update it.

DEC-08-FUBO-D
TC01 DECtape Subroutines
These subroutines provide the user with the ability to read, write and search using the TC01 tape system. The read and write subroutines transfer 128_{10} (one memory page) of the specified block (or blocks) although the standard block length is 129_{10} 12-bit words. Successive blocks are read (written) from (into) successive 128 word blocks of core. Provision is made for transfers to and from extended memories.

DEC-08-EUFA-D
TC01/TU55 DECtape Formatter
The purpose of this system is to record the required timing and mark tracks on a DECtape mounted on the TC01-TU55 DECtape unit.
The program, which never stops, obtains the variable information it needs by communication with the operator via the ASR33 Teletype.
Two full passes are required to complete one DECtape. Upon completion of a sequence, another tape may be mounted and formatted, as the last, without renewed communication between the operator and program. Therefore, marked tape may be produced in great numbers with little operator intervention, at a rather rapid rate. One tape, excluding tape setup time, requires two minutes from start to finish (see also Disc Software).

DISK/DECTAPE SOFTWARE

DEC-D8-SDAA-D
DISK MONITOR SYSTEM
This system consists of a keyboard-oriented Monitor, which enables the user to efficiently control the flow of programs through his PDP-8/L, and a comprehensive software package, which includes a FORTRAN Compiler, Program Assembly Language (PAL-D), Edit program (EDITOR), Peripheral Interchange Program (PIP) and Dynamic Debugging Technique (DDT-D) program. Also provided is a program (BUILDER) for generating a customized monitor according to the user's particular machine configuration (amount of core, number of discs or DECtapes, etc.).
The system is modular and open ended, permitting the user to construct the software required in his environment, and allows the user full access to his disc (or DECtape)—referred to as the **system device**—for storage and retrieval of his programs. By typing appropriate commands to the Monitor, the user can **load** a program (construct it from one or more units of binary coding previously punched out on paper tape or written on the disc by the Assembler, and assign it core), **save** it (write it out, with an assigned starting address, on the system device), and later **call** it (read it back into core from the system device) for execution.
In order to have a complete DISC/DECtape package, the user may order the following in addition to DEC-D8-SDAA-D above:

1. Disc System Builder — DEC-D8-SBAC-PB
2. Disc Editor — DEC-D8-ESAB-PB
3. PIP — DEC-D8-PDAA-PB
4. Disc DDT — DEC-D8-CDDO-PB
5. Disc DDT Driver (ASCII) — DEC-D8-CDDO-PA
6. Disc/DECtape FORTRAN — DEC-D8-AFA(1-6)-PB
7. PAL-D Assembler — DEC-D8-ASAA-D (Manual)
   DEC-D8-ASAA-PB

DEC-D8-ASAA-D
PAL-D DISK ASSEMBLER
PAL-D is the symbolic assembly program designed primarily for the 4K PDP-8 family of computers with disc or DECtape. The PAL-D Assembler makes machine language programming easier, faster, and more efficient. Basically, the Assembler processes the programmer's source program statements by translating mnemonic operation codes to the binary codes needed in machine instructions, relating symbols to numeric values, assigning absolute core addresses for program instructions and data, and preparing an output listing of the program, which includes notification of any errors detected during the assembly process.

MAINTENANCE PROGRAMS

Maindec-8/I-D01B-D
Instruction Test 1
This is a diagnostic program for testing the AND, TAD, and OPERATE instructions of the PDP-8/L.

Maindec-8I-D02B-D
Instruction Test 2
This program is an extensive test of autoindexing, indirect addressing, and the DCA instruction for the PDP-8/L. It also offers minimal testing for interrupt and the AND, TAD, ISZ, JMP and JMS instructions.

Maindec 08-D02B-D
Instruction Test Part 2B
This program is a test of the 2's complement add (TAD) and rotate logic (RAL, RTL, RAR, RTR). Random numbers are used in the Two's Add portion of the test and sequential numbers are used in the Rotate portion. Program control depends on operator manipulation of four switches in the SWITCH REGISTER (bits 0, 1, 2, 3). Error information is normally printed out on the keyboard printer.

Maindec 81-D4CA-D(L)
PDP-8/L Memory Parity IOT Test
The PDP-8L Memory Parity IOT Test is designed to exercise and detect errors on the memory parity control logic. A routine is also included which writes random numbers in memory field 0, and then checks for data parity errors. Manual intervention after the start of the test is required in order to test the parity IOT's. Printed instructions are given on the TTY printer.

Maindec-08-D04B-D
Random JMP Test
This program tests the JMP instruction of the PDP-8/L. Most of memory is used as a JUMP field with a random number generator selecting each "JUMP FROM" and "JUMP TO" location.

Maindec-08-D05B-D
Random JMP-JMS Test
This is a diagnostic program to test the JMS instruction of the PDP-8/L. Random "FROM" and "TO" addresses are selected for each test. The JMP instruction is tested in that each test requires a JMP to reach the JMS.

Maindec-08-D07B-D
Random ISZ Test
This program is written to test the ISZ instruction of the PDP-8/L. An ISZ instruction is placed in a FROM location, and a TO location contains the OPERAND. Part 1 of the program selects FROM, TO, and OPERAND from a random number generator, with the option of holding any or all constant. Part 2 uses a fixed set of FROM, TO, and OPERAND numbers.

Maindec-08-D1AC-D
PDP-8/L Memory Power On/Off Test
This program is a Memory Data Validity Test to be used after a simulated power failure.

Maindec-08-D1B0-D
Memory Address Test
The Memory Address Test checks for proper memory address selection on the PDP-8/L.

Maindec-08-D1EB-D
Extended Memory Checkerboard
The PDP-8/L Extended Memory Checkerboard diagnostic is designed to provide worst case half-select noise conditions in order to determine the operational status of core memory. Four data patterns, and their complements, are written and checked for error. The patterns provided will generate the worst case noise conditions for a PDP-8/L equipped with standard or specially purchased core stacks, and will test systems equipped with 8K words of core memory. Automatic program relocation is provided in order to test all memory stacks from each stack.
Teletype print-outs are provided for error identification. Also, the operator is given a degree of control over the program by various SR settings.

Maindec-08-D1GB-D
Extended Memory Control
This program tests the Extended Memory Control logic for proper operation. It may be used with a PDP-8/L equipped with a minimum of 4K of extended memory. The program exercises and tests the control IOT's; the ability to reference all fields from field 0; program interrupt and interrupt inhibit; auto-indexing in each field, and a special test for the PDP-8/L which tests the presence of a false memory pulse when a non-existent memory field is referenced. Errors encountered during running will result in a program halt. The halt locations are labeled, and the error may be identified by referencing the program listing or table of error halts.

Maindec-08-D1HA-D(D)
Extended Memory Address Test
The PDP-8/L Extended Memory Address Test tests all of memory not occupied by the program to make sure that each location can be uniquely addressed. This is performed by a series of four tests. The first two tests write the address and complement address of each memory location into itself, and then checks the contents of each location to make sure each is correct. The third test first sets all of memory not occupied by the program to all ones, and then writes a word of all zeroes, except for one bit, into each location and checks for error. The fourth test is similar except that a word of all ones, except for one bit, is written into each location and checks for error.

Maindec-08-D1LO
Basic Memory Checkerboard
The Memory Checkerboard diagnostic tests memory for core failure on half-selected lines under worst case conditions. Its use is intended for basic 4K memory systems.

Maindec-08-D4AO-D
Memory Parity Checkerboard
The PDP-8/L Memory Parity Checkerboard diagnostics tests the parity bit plane for core failure on half-selected lines under worst case conditions. Its use is intended for basic 4K memory systems.

Maindec-08-D4BA-D
Extended Memory Parity Test
The PDP-8/L Extended Memory Parity Test is designed to provide worst case half-select noise conditions within the parity bit plane. Four data patterns, and their complements are written, and checks for parity errors after writing each pattern are made. The program will test systems equipped with 8K words of core memory.
Operation of the program is similar to the Extended Memory Checkerboard test except that program relocation is not included, and error halts are provided for error identification.

Maindec-08-D2EA-D
High Speed Reader Test
This program tests the performance of the Speed Perforated Tape Reader and Control by scanning a closed-loop test tape for accuracy of transmission. The reader control is tested for correct operation with the PDP-8/L interrupt system.
An auxiliary program included with the test punches a tape from which the test loop can be made.

Maindec-08-D2PD-D
Family of 8 ASR 33/35
Teletype Tests Part 1
The family-of-8 ASR33/35 Teletype Tests Part 1 is the first part of a 2 part package used to test the ASR33, or ASR35 Teletype when attached to a Family-of-8 system.
Part 1 contains nine selectable programs numbered from 0 to 10 (octal). The programs are selected by means of the Switch Register (SR).
The programs available are:

PRG0	Basic Input Logic Tests
PRG1	Basic Output Logic Tests
PRG2	Reader Test
PRG3	Test Tape Generator. Punches tape with characters stored in locations 0021 and 0022.
PRG4	Test Tape Generator. Punches Binary Count Pattern test tape.
PRG5	Reader Exerciser. Reads Binary Count pattern tape in random length blocks, and with fixed stalls between characters. The stall is determined at random.
PRG6	Reader Exerciser. Reads Binary Count pattern tape. Fixed stall between characters. Stall count is taken from LOC 0023.
PRG7	Reader Exerciser. Reads tape punched with any 2 test characters. Random length blocks and fixed stall between characters. The stall is determined at random.
PRG10	Reader Exerciser. Reads tape punched with any 2 test characters. Fixed stall between characters. Stall count taken from LOC 0023.

Maindec-08-D2QD-D
Family of 8 ASR33/35
Teletype Tests, Part 2
The Family-of-8 ASR33/35 Teletype Tests, Part 2 is the second part of a 2 part package used to test the ASR33 or ASR35 Teletype when attached to a Family-of-8 system.
Part 2 contains nine selectable programs numbered from 0 to 10 (octal). The programs are selected by means of Switch Register (SR).
The available programs are:

PRG0	Printer Test
PRG1	Punch Test
PRG2	Keyboard Test
PRG3	Combined Reader, Printer, Punch Test
PRG4	Printer Exerciser. Prints lines of characters stored in LOC 0021 and 0022. No stalls.
PRG5	Same as PRG4, but stalls between characters.
PRG6	Punch Exerciser. Punches and read checks data blocks of data stored in LOC 0021 and 0022. No stalls.
PRG7	Same as PRG6, but random stalls between characters punched.
PRG10	Punch Exerciser. Punches and read checks blocks of Binary Count pattern. Random stalls between characters punched.

Maindec-08-D3BB-D
TC01 Basic Exerciser
The TC01 Basic Exerciser is a series of test programs that may be used to gain a high degree of confidence in the data handling ability of a TC01 DECtape Control and one to eight TU55 DECtape Transports. The Basic Exerciser consists of several basic routines that may be individually selected; each routine will operate on any configuration of one to eight drives. These routines include a Basic Motion Routine, Search Find All Blocks Test, Basic Search Routine, Start/Stop/Turnaround Test, Basic Write/Read Data Test with eight selectable patterns, and a Parity Generation and Checking Test. The operation of the Basic Motion Routine and the Basic Search Routine are controlled by keyboard input. Also, a Write Data Scope Loop, Read Data Scope Loop, and a Search Scope are provided to keep the tape moving from end zone to end zone.

Maindec-08-D3EB-D
TC01 Extended Memory Exerciser
TC01 EXTENDED MEMORY EXERCISER is a test program for the PDP-8/L Computer which tests the transfer to data between the TC01 DECtape Control and extended memory fields (more than 4K). It does this by storing a data pattern in an extended memory field, transferring the data onto DECtape and then reading the data back into the field and checking it for correct transfer.

Maindec-08-D3RA-D
DECTREX 1
TC01 Random Exerciser
DECTREX 1 is a DECtape Random Exerciser for the TC01 DECtape control and any configuration of one to eight TU55 DECtape transports. Drive selection, tape direction, number of blocks, sequence of operation and patterns generated are by random selection. The DECtape functions exercised are search, read data and write data in normal and continuous modes, read all in continuous mode, and move.
Also included are a short series of processor tests that are executed while waiting for interrupts and during data breaks while searching, reading, and writing from DECtape.

MAINDEC-8/1-D5BB-D
DF32 DISKLESS
Logic Test, MiniDisk
Diskless is a test of the DF32 disk logic and its computer interface. This program does not test the disk, nor associated analog interface circuits.
(The disk is not needed for these routines; if the disk is connected, the disk motor should be turned off. For a complete test of the disk system, use DF32 Disk Data Test.)

MAINDEC-08-D5CC-D
DF32 DISK DATA
Mini Disk, Interface, Address, Data Test
The DF32 Disk Data is a complete test of the disk system. Also included is a short processor test that is executed while waiting for interrupts, and during data breaks.

MAINDEC-08-D5DA-D
Exerciser for Master and Slaves Units
"Multi Disk" is a high speed exerciser intended for multi disk configuration and can control one (1) to four (4) disks.

MAINDEC-08-D6CB-D
Calcomp Plotter Test
This program tests the CALCOMP Plotter and its control. All control and plotting functions are tested.

MAINDEC-08-D6HA-D
AF04A Diagnostic & Demonstration
The DIAGNOSTIC & DEMONSTRATION program for the AF04A allows the operator to type in up to 1000_8 pseudo instructions and cause analog to digital conversions via the AF04A. The pseudo-instructions which make up the individual pseudo-program will be executed when a "$" (dollar sign) is inputed from the keyboard. The operator may specify all parameters of the conversion instruction and specify any order of instructions.

MAINDEC-08-D6JA-D
AD08 Diagnostic
For the AD08A this is an I/O INSTRUCTION and calibration check. For the AD08B it is also a limited test of multiplexer selection and A/D repeatability.

MAINDEC-81-D6AA-D
AX08 Diagnostic
This unit is tested in three sections (a) an instruction test of the logic, (b) a display test for the scope, (c) and a calibration section for the A/D Converter.

MAINDEC-8/I-D8AB-D
KW8I Real Time Clock
The KW8I Real Time Clock, diagnostic program, is designed to thoroughly test all IOT and DATA transfer instructions used in the M708 Clock Control and M709 Clock Counter. The program consists of two routines; the first routine which starts at address 200, tests all flags, enables, etc., to ascertain if initialize has cleared them, and the second phase of the clock control program tests each of the IOT's to determine if they will set and/or clear each of the controllable flip-flops, it test for proper skips, program interrupt and for proper operation of any of the three clocks. The error type out for an error in the clock control test is as follows:

ERROR 0001

NOTE

Any clock system which supplys a clock at a frequency of less than one clock pulse per 10 seconds will cause a failure Error 0003.

The second routine starts at address 400, and is used only when a M709 Clock Counter module is connected to the clock control. The clock i.e. crystal, adjustable or line, must be removed from the computer in order for this test to run. The error typeout for an error in the clock control test is as follows:

CLOCK COUNTER FAILED
SENT RXED
0001 0000

The SENT refers to a 12-digit number which was loaded into the clock control counter, and the RXED refers to the number which was transferred back to the computer from the counter.

MAINDEC 08-D8FA-D
DP01A Bit Syncronous Data Communication System
IOT and Data Test
The DP01A test consists of two sequences intended to verify correct operation of all IOT instructions and associated control logic with the DP01A Bit synchronous data communication system.

MAINDEC 08-D8HB-D
DP01A Bit Synchronous Data Communication System
IOT and Data Test for IOT's 6301 through 6354
The DP01A test consists of two test sequences intended to verify correct operation of all IOT instructions and associated control logic with the DP01A Bit synchronous data communication system.

MAINDEC-08-D8KA-D
DP01A IOT and Data Tests (60-67)
The DP01A test consists of two independent test sequences intended to verify correct operation of the IOT instructions and control logic associated with the DP01A Bit Synchronous Data Communication System. Although the tests are treated separately, they may be in memory at the same time.

MAINDEC-08-D8LA-D
DP01A Bit Synchronous Data Communication System IOT and Data Test for IOT 6501 through 6564
The DP01A test consists of two test sequences intended to verify correct operation of all IOT instructions and associated control logic with the DP01A Bit synchronous data communication system.

MAINDEC-08-D9AB-D
TC58 Data Reliability Test (7 Track)
The TC58 Data Reliability Test is primarily designed for the collection of statistical information pertaining to the data reliability of the tape drives that may be associated with the TC58 Magnetic Tape Control. The program is also designed to be usable as an aid to the hardware debugging and maintenance of the TC58 Magnetic Tape Control and its associated magnetic tape drives. This program may also be used as an extended data reliability acceptance test.

MAINDEC 08-D9BA-DL
TC-58 Drive Function Timer
The TC58 Drive Function Timer program is designed to be an aid in the hardware debugging and maintenance of the TC58 Magnetic Tape Control and its associated magnetic tape drives. The program will operate on any configuration of 1 to 8,45 or 75 inch per second 7, or 9 track drives.
Selected operations are initiated, timed and the times are then typed in decimal milliseconds. There is no limit checking on times by the program, the decisions on the validity of times typed must be made external to the program or by the person operating this test.

MAINDEC-08-D9CB-D
TC58 Random Exerciser
The TC58 Random Exerciser Test is a test program designed to simulate tape system usage. Any configuration of 1 through 8 TU20 (or similar) 7- and/or 9-track drives may be concurrently tested.

MAINDEC-08-D9DB-D
TC58 Instruction Test — Part 1
The TC58 Instruction Test is a series of incremental subtests designed to aid in the checkout and maintenance of the TC58 magnetic tape system.

MAINDEC-08-D9EA
TC58 Instruction Test — Part 2
The TC58 Instruction Test is a series of incremental subtests designed to aid in the checkout and maintenance of the TC58 magnetic tape system.

DECUS No. 5/8-1.1
BPAK — A Binary Input-Output Package

A revision of the binary package originally written by A. D. Hause of Bell Telephone Laboratories. With BPAK the user can read in binary tapes via the photoreader and punch them out via the Teletype punch. It may be used with any in-out device, but is presently written for the photoreader and Teletype punch. A simple modification converts BPAK so that it reads from the Teletype reader if the photoreader is disabled. In its present form it occupies locations 7600-7777.

DECUS No. 5-2.1
OPAK — An On-line Debugging Program for the PDP-5

A utility program which enables the user to load, examine, and modify computer programs by means of the Teletype. This program is a revision of the program written by A. D. Hause, Bell Telephone Laboratories. Extensive use of the program has suggested many refinements and revisions of the original program, the most significant additions being the word search and the break point. The standard version of OPAK is stored in 6200 to 7577 and also 0006. An abbreviated version is available (7000 to 7577, 0006) which is identical to the other except that it has no provision for symbolic dump. Both programs are easily relocated. Control is via Teletype, with mnemonic codes, (e.g. "B" for inserting breakpoint, "P" for proceed, etc.).

DECUS No. 5-3
BRL — A Binary Relocatable Loader with Transfer Vector Options for the PDP-5 Computer.

A binary loader program occupying 4640_8 to 6177_8 registers, also 160 to 177. It has two main functions:

1. It allows a PDP-5 operator to read a suitably prepared binary program into any page location in memory except the registers occupied by BRL.
2. It greatly simplifies the calling of programmed subroutines by allowing the programmer to use an arbitrary subroutine calling sequence when writing his program, instead of having to remember the location of the subroutines.

DECUS No. 5-4
Octal Typeout of Memory Area with Format Option

(Write-up and Listing Only)

DECUS No. 5-5
Expanded Adding Machine

Expanded Adding Machine is a minimum-space version of Expensive Adding Machine (DEC-5-43-D) using a table lookup method including an error space facility.

This is a basic version to which additional control functions can easily be added. Optional vertical or horizontal format, optional storage of intermediate result without reentry, fixed-point output of results within reason, and other features that can be had in little additional space under switch register control. (Write-up and Listing Only)

DECUS No. 5-6
BCD to Binary Conversion of 3-Digit Numbers

This program is based on DEC-5-4 and is intended to illustrate the use of alternative models in program construction.

While not the fastest possible, this program has one or two interesting features. It converts any 3-digit BCD-coded decimal number, $D_1D_2D_3$ into binary in the invariant time of 372 microseconds. Efficient use is made of BCD positional logic to work the conversion formula $(10D_1 + D_2)\ 10 + D_3)$ by right shifts in the accumulator. In special situations, it could be profitable to insert and initial test/exit on zero, adding 12 microseconds to the time for non-zero numbers.

(Write-up and Listing Only)

DECUS No. 5/8-7
Decimal to Binary Conversion by Radix Deflation on PDP-8

(Write-up and Listing Only)

DECUS No. 5-8
PDP-5 Floating Point Routines

Consists of the following routine:

1. Square Root — Binary Tape and Symbolic Listing
2. Sine-Cosine — Binary Tape only
3. Exponential — Binary Tape only

DECUS No. 5/8-9
Analysis of Variance PDP-5/8

An analysis of variance program for the standard PDP-5/8 configuration.

The output consists of:

A. For each sample:
   1. sample number
   2. sample size
   3. sample mean
   4. sample variance
   5. sample standard deviation

B. The grand mean

C. Analysis of Variance Table:
   1. The grand mean.
   2. The weighted sum of squares of class means about the grand mean..
   3. The degrees of freedom between samples.
   4. The variance between samples.
   5. The pooled sum of squares of individual values about the means of their respective classes.
   6. The degrees of freedom within samples.
   7. The variance within samples.
   8. The total sum of squares of deviations from the grand mean.
   9. The degrees of freedom.
   10. The total variance.
   11. The ratio of the variance between samples to the variance with samples.

This is the standard analysis of variance table that can be used with the F

test to determine the significance, if any, of the differences between sample means. The output is also useful as a first description of the data.

All arithmetic calculations are carried out by the Floating Point Interpretive Package (Digital-8-5-S)

DECUS No. 5-10
Paper Tape Reader Test

A test tape can be produced and will be continuously read as an endless tape. Five kinds of errors will be detected and printed out. The Read routine is in 6033-6040. Specifications: Binary with Parity Format — Length: registers in locations (octal): 10, 11, 4 through 67 (save 63, 64), and 6000-7777.

DECUS No. 5-11
PDP-5 Debug System

Purpose of this program is to provide a system capable of:

1. Octal dump 1 word per line.
2. Octal dump 10_8 words per line.
3. Modifying memory using the typewriter keyboard.
4. Clearing to zero parts of memory.
5. Setting to HALT codes part of memory.
6. Entering breakpoints into a program.
7. Initiating jumps to any part of memory.
8. Punching leader on tape.
9. Punching memory on tape in RIM format.
10. Punching memory on tape in PARITY format.
11. Load memory from tape in PARITY format.

DECUS No. 5-12
Pack-Punch Processor and Reader for the PDP-5

The processor converts a standard binary-format tape into a more compressed format, with two twelve-bit words contained on every three lines of tape. Checksums are punched at frequent intervals, with each origin setting or at least every 200 words.

The reader, which occupies locations 7421 to 7577 in the memory will load a program which is punched in the compressed format. A test for checksum error is made for each group of 200 or less and the program will halt on error detection. Only the most recent group of words need be reloaded. Read-in time is about ten per cent less than for conventional binary format, but the principal advantage is that little time is lost when a checksum error is detected, no matter how long the tape.

DECUS No. 5-13
PDP-5 Assembler for use in IBM 7094/7044

This program accepts IBM 7094/7044 symbolic programs punched on cards and assembles them for the PDP-5. An assembly listing is produced, and a magnetic tape is generated containing the program. This magnetic tape can be converted to paper tape and then read into the PDP-5 or it can be read directly into a PDP-5 with an IBM compatible tape unit. Cards are available.

DECUS No. 5/8-14
DICE Game for the PDP-5/8

Enables a user to play the game, DICE, on either the PDP-5 or PDP-8.

DECUS No. 5/8-15
ATEPO (Auto Test in Elementary Programming and Operation of a PDP-5/8 Computer)
The program will type questions or instructions to be performed by the operator of a 4K PDP-5/8. The program will check to see if the operator has answered the questions correctly. If this is the case, it will type the next question or instruction.

DECUS No. 5-16
Paper Tape Duplicator for PDP-5
The tape duplicator for the PDP-5 is a single buffered read and punch program utilizing the program interrupt. It computes a character count and checksum for each tape and compares with checks at the end of the tape. Checks are also computed and compared during punching.

DECUS No. 5/8-17
Type 250 Drum Transfer Routine For Use on PDP-5/8
Transfers data from drum to core (Read) or core to drum (Write) via ASR-33 Keyboard Control.

DECUS No. 5/8-18a
Binary Tape Disassembly Program
Dissassembles a PDP-5 or 8 program, which is on tape in BIN format. It prints the margin setting, address, octal contents, mnemonic interpretation (PAL) of the octal contents. A normal program or a program which uses Floating Point may be dissassembled

DECUS No. 5/-18B
Binary Tape Disassembler
An extension of the Disassembler which enables double spacing and paging of output.

DECUS No. 5/8-18C
Disassembler with Symbols
This disassembler accepts a binary tape of standard format and produces a listing of the tape in PAL III mnemonics, and a cross-reference table of all addresses referenced by any memory-reference instruction. A symbol table may be entered to produce a listing similar to a PAL III Pass 3 listing. A patch to produce only a cross reference table is included.

Minimum Hardware:	PDP-8 (4K), ASR-33, high-speed reader, EAE
Storage:	20-1773$_8$ for program, 1774-7577$_8$ for scratch

DECUS No. 8-19a
DDT-UP Octal-Symbolic Debugging Program
DDT-UP is an octal-symbolic debugging program for the PDP-8 which occupies locations 5600 through 7677. It is able to read a symbol table punched by PDPSYM and stores symbols, four locations per symbol, from 5577 down towards 0000. The mnemonics for the eight basic instructions and standard OPR and IOT group instructions are initially defined and the highest available location for the user is initially 5363.

From the Teletype, the user can symbolically examine and modify the contents of any memory location. DDT-UP allows the user to punch a corrected program in CBL format.

DDT-UP has a breakpoint facility to help the user run sections of his program. When this facility is used, the debugger also uses location 0005.

DECUS No. 5/8-20
Remote Operator FORTRAN System

Program modification and instructions to make the FORTRAN OTS version dated 2/12/65 operate from remote stations.

DECUS No. 5/8-21
Triple Precision Arithmetic Package for the PDP-5 and the PDP-8

An arithmetic package to operate on 36-bit signed integers. The operations are add, subtract, multiply, divide, input conversion, and output conversion. The largest integer which may be represented is $2^{35} - 1$ or 10 decimal digits. The routines simulate a 36-bit (3 word) accumulator in core location 40, 41, and 42 and a 36-bit multiplier quotient register in core locations 43, 44, and 45. Aside from the few locations in page 0, the routines use less core storage space than the equivalent double-precision routines.

DECUS No. 5-22
DECtape Duplicate

A DECtape routine for the PDP-5 to transfer all of one reel (transport 1) to another (transport 2). Occupies one page of memory beginning at 7400. The last page of memory is not used during the operation of the program, however, the memory from 1 to 7436 is used to set the DECtape reels in the proper starting attitude and is then destroyed during duplication. Duplication will commence after which both reels will rewind. Parity error will cause the program to halt with 0040 in the accumulator.

DECUS No. 5/8-23A
PDP-5/8 Oscilloscope Symbol Generator

The subroutine may be called to write a string of characters, a pair of characters, or a single character on an oscilloscope. Seventy (octal) symbols in ASCII Trimmed Code and four special "format" commands are acceptable to this routine. The program is operated in a fashion similar to the DEC Teletype Output Package.

DECUS No. 5/8-23B
PDP-5/8 Oscilloscope Symbol Generator (5 x 7 Matrix)

This subroutine may be called to write a string of characters, a pair of characters, or a single character on a 34D Oscilloscope. Twenty-six alphabetic characters and 0-9 numeric characters are acceptable. However, there is space available to include any symbol the user desires. The program is operated in a fashion similar to the DEC Teletype Output Package.

Source Language: MACRO-8

Storage 200_8-777_8 registers

DECUS No. 5-24
Vector Input/Edit

Accepts input to a PDP-5 and allows both time-of-entry and post time-of-entry corrections.

DECUS No. 5-25
A Pseudo Random Number Generator

The random number generator subroutine, when called repeatedly, will return a sequence of 12-bit numbers which, though deterministic, appears to be drawn from a random sequence uniform over the interval 0000_8 to 7777_8. Successive numbers will be found to be statistically uncorrelated. The sequence will not repeat itself until it has been called over 4 billion times.

DECUS No. 8-26a
Compressed Binary Loader (CBL)

The CBL (Compressed Binary Loader) format in contrast to BIN format utilizes all eight information channels of the tape, thus achieving nearly 25% in time savings.

Whereas BIN tapes include only one checksum at the end of the tape, CBL tapes are divided into many independent blocks, each of which includes its own checksum. Each block has an initial loading address for the block and a word count of the number of words to be loaded.

The CBL loader occupies locations 7700 through 7777.

DECUS No. 8-26b
CBC (BIN to CBL) and CONV (CBL to BIN)

Two conversion programs which use the PDP-8 on-line Teletype to read a binary tape in one format and punch a binary tape in the other format. The conversion programs both ignore memory field characters so that the output is a tape for memory field 0.

DECUS No. 8-26c
XCBL — Extended Memory CBL Loader

XCBL is used to load binary tapes punched in CBL format into a PDP-8 with more than standard 4K memory. This loader occupies locations 7670 through 7777 of any memory field.

DECUS No. 8-26d
XCBL Punch Program

This program permits a user to prepare an XCBL tape of portions of a PDP-8 extended memory through the control of the keyboard of the on-line Teletype.

The program is loaded by the XCBL Loader.

There are two versions of the program so that any section of memory may be punched:

LOW XCBL occupies 00000 - 00377 and its starting address is 00000

HIGH XCBL occupies 17200 - 17577 and its starting address is 17200.

The program may be restarted at the starting address at any time.

One option is provided according to the setting of bit 0 of the Switch Register. If bit 0 is a ONE, the operation of XCBL PUNCH is similar to that of DDT-UP (DECUS No. 8-19a).

DECUS No. 5/8-27 and 5/8-27a
Bootstrap Loader and Absolute Memory Clear

Bootstrap Loader inserts a bootstrap loading program in page 0 from a minimum of toggled instructions.

Absolute Memory Clear leaves the machine in an absolutely clear state and, therefore, cycling around memory obeying an AND instruction with location zero. Should not be used unless one plans to re-insert the loader program.

DECUS No. 5/8-28a
PAL III Modifications-Phoenix Assembler

This modification of the PAL III Assembler speeds up assembly on the ASR-33/35 and operates only with this I/O device. Operation is essentially the same as PAL III, except that an additional pass has been added, Pass 0. This pass, started in the usual manner but with the switches set to zero, reads the symbolic tape into a core buffer area. Subsequent passes then read the tape image from storage instead of from the Teletype.

DECUS No. 5/8-29
BCD to Binary Conversion Subroutines

These two subroutines improve upon the DEC supplied conversion routine. Comparison cannot be made to the DECUS-supplied fixed-time conversions. DECUS No. 5-6, because it is specified only for the PDP-5. One routine is designed for minimal storage, the other for minimal time. Both are fixed-time conversions; time specified is for a 1.5-μsec machine.

Minimal time routine: 73.6 μsec/32 locations

Minimal storage routine: 85 μsec/29 locations

DEC Routine: 64-237 μsec/37 locations

DECUS No. 5-30
GENPLOT—General Plotting Subroutine

This self-contained subroutine is for the PDP-5 with a 4K memory and a CALCOMP incremental plotter. The subroutine can move (with the pen in the up position) to location (x,y), make an "x" at this location, draw a line from this present position to location (x,y) and initialize the program location counters.

DECUS No. 5-31
FORLOT—FORTRAN Plotting Program for PDP-5

FORPLOT is a general-purpose plotting program for the PDP-5 computer in conjunction with the CALCOMP 560 Plotter. It is self-contained and occupies memory locations 0000_8 to 4177_8. FORPLOT accepts decimal data inputted on paper tape in either fixed or floating point formats. Formats can be mixed at will. PDP-5 FORTRAN output tapes are acceptable directly and any comment on these are filtered out.

DECUS No. 5/8-32a
Program to Relocate and Pack Program in Binary Format

Provides a means to shuffle machine language programs around in memory to make the most efficient use of computer store.

DECUS No. 5/8-33
Tape to Memory Comparator

Tape to Memory Comparator is a debugging program which allows comparison of the computer memory with a binary tape. It is particularly useful for detecting reader problems, or during stages of debugging a new program. Presently, uses high-speed reader, but may be modified for TTY reader.

DECUS No. 5-34
Memory Halt—A PDP-5 Program to Store Halt in Most of Memory

With Memory Halt and Opak, (DECUS No. 5-2.1), in memory, it is possible to store halt (7420) in the following memory locations:

0001 to 0005
0007 to 6177
7402 to 7403

DECUS No. 5/8-35
BCD to Binary Conversion Subroutine and Binary to BCD Subroutine (Double Precision)

This program consists of a pair of relatively simple and straightforward double-precision conversions.

DECUS No. 5-36
Octal Memory Dump Revised

The Octal Memory Dump on Teletype is a DEC routine (DEC-5-8-U) which dumps memory by reading the switch register twice; once for a lower limit and again for an upper limit. It then types an address, the contents of the program and the next three locations, issues a CR/LF, then repeats the process for the next four locations. This leaves the right two-thirds of the Teletype page unused. The 78_{10} instructions occupy two pages.

This revised routine uses the complete width of the Teletype page and occupies only one memory page, using less paper and two less instructions. Now an address and the contents of 15 locations are typed out before a carriage return.

Octal Memory Dump Revised has proved its value as a subroutine and/or a self-contained dump program when it is necessary to dump large sections of DECtape, magnetic tape (IBM compatible) or a binary formatted paper tape.

DECUS No. 5-37
Transfer II

For users who have more than one memory bank attached the PDP-5/8, Transfer II may prove valuable in moving information from one field to another. When debugging, Transfer II enables a programmer to make a few changes in a new program and test it without reading in the original program again. Transfer II enables more extensive use of memory banks.

DECUS No. 5/8-38
FTYPE—Fractional Signed Decimal Type-In

Enables a user to type fractions of the form: .582, —.73, etc., which will be interpreted as sign plus 11 bits (e.g., $0.5 = 2000_8$). Subroutine reads into 300-3177 and is easily relocated, as it will work on any page without modifications.

DECUS No. 5/8-39
DSDPRINT, DDTYPE—Double-Precision Signed Decimal Input-Output Package

DSDPRINT, when given a signed 24-bit integer, types a space or minus sign, and then a 7-digit decimal number in the range —8388608 to +8388607. DDTYPE enables user to type in a signed decimal number in either single or double precision. These routines are already separately available, but the present subroutine package occupies only one memory page and allows for more efficient memory allocation. Located in 3000-3177, but will work on any page.

DECUS No. 5-40
ICS DECtape Routines (One-Page)

The routines will read or write from the specified DECtape unit and delay the program until all I/O is completed. The last block read will overflow the specified region and destroy one core location. Only standard 129 word DECtape blocks will be read or written. The routines will halt if an error occurs with the status bits in the AC.

DECUS No. 5-41
Breakpoint

This debugging routine has been reduced to a minimum operation. It is a mobile routine which can operate around any program that leaves an extra 30 cells of memory space.

Its function is to insert break points in any given location of the program being debugged, and to hold the contents of AC and Link. The programmer may examine any locations desired and then continue to the next breakpoint. It is presently located in 140_8-170_8, but may be easily relocated.

DECUS No. 5-42
Alphanumeric Input

With the Alphanumeric Input Package, any character may be read into the PDP-5 through either the Teletype or the high-speed reader. The characters are packed two/cell and stored in the address indicated in the switch register.

DECUS No. 5/8-43
Unsigned Octal-Decimal Fraction Conversion

This routine accepts a four-digit octal fraction in the accumulator and prints it out as an N-digit decimal fraction where N = 12 unless otherwise specified. After N digits, the fraction is truncated. Programs are included for use on the PDP-5 with Type 153 Automatic Multiply-Divide and the PDP-8 with Type 182 Extended Arithmetic Element.

Storage requirements: 55 Octal locations for the PDP-5. 47 Octal locations for the PDP-8.

DECUS No. 8-44
Modifications to the Fixed Point Output in the PDP-8 Floating Point Package (Digital 8-5-S)

The Floating Point Package (Digital 8-5-S) includes an Output Controller which allows output in fixed point as well as floating point format. This Output Controller takes the form of a certain number of patches to the "Floating Output E Format" routine, plus an additional page of coding.

Using the Calculator program (Digital 8-10-S), which includes the Floating Point Package, certain deficiencies were noted in the fixed-point output format, particularly the lack of any automatic rounding off. For example, the number 9, if outputted as a single digit, appears as 8. Modification attempts to provide automatic rounding off resulted in the Output Controller being completely rewritten with minor changes in the format.

This new version of the Output Controller is also in the form of patches to the Floating Output with an additional page of coding, thereby not increasing the size of the Floating Point Package.

The following summarizes this new version:

1. The number output is automatically rounded off to the last digit printed, or the sixth significant digit, whichever is reached first. Floating point output is rounded off to six figures since the seventh is usually meaningless.
2. A number less than one is printed with a zero preceding the decimal point (e.g., "+0.5" instead of "+.5").
3. A zero result, after rounding off, is printed as "+0" instead of "+".

4. The basic Floating Point Package includes the facility to specify a carriage return/line feed after the number using location 55 as a flag for this purpose. The patches for the Output Controller caused this facility to be lost. This version restores this facility.

DECUS No. 5/8-45
PDP-5/8 Remote & Time-Shared System

A time-shared programming system which allows remote stations immediate access to the computer and a wide selection of programs.

DECUS No. 5/8-46A
PDP-5/8 Utility Programs

Consists of four programs (listed below) each of which may be selected via the teletypewriter. When the program is started, either by a self-starting binary loader or by manually starting the computer in address $200_{(8)}$, it is in its executive mode. In this mode, it will respond only to five keys and perform the following functions:

B—go to BIN to QK Converter Program
E—go to Editor Program
L—type a section of leader and stay in executive
P—go to Page Format Program
Q—go to QK to BIN Converter Program

DECUS No. 8-47
ALBIN—A PDP-8 Loader for Relocatable Binary Programs

ALBIN is a simple method for constructing relocatable binary formatted programs, using the PAL III Assembler. Allocation of these programs can be varied in units of one memory page (128_{10} registers.). When loading an ALBIN program, the actual absolute addresses of indicated program elements (e.g., the keypoint of subroutines) are noted down in fixed program-specified location on page zero. In order to make a DEC symbolic program suitable for translation into its relocatable binary equivalent, minor changes are required, which, however, do not influence the length of the program. Due to its similarity to the standard DEC BIN loader, the ALBIN loader is also able to read-in normal DEC binary tapes. ALBIN requires 122_{10} locations, RIM loader included. Piling-up in core memory of ALBIN programs stored on conventional or DECtape can be achieved using the same method with some modifications.

DECUS No. 5/8-48
Modified Binary Loader MKIV

The Mark IV Loader was developed to accomplish four objectives:

1. Incorporate the self-starting format described in DECUS 5/8-27, ERC Boot.
2. Selected the reader in use, automatically, without switch register settings.
3. Enables a newly-prepared binary tape to be checked prior to loading by calculating the checksum.
4. Reduce the storage requirements for the loader so that a special program would fit on the last page of memory with it.

DECUS No. 8-49
Relativistic Dynamics

Prints tables for relativistic particle collisions and decay in the same format as the Oxford Kinematic Tables. It can be used in two ways:

1. Two-particle Collisions—Given the masses of incident, target, and emitted particles, the incident energy and centre-of-mass angles, the program calculates angles and energies of the emitted particles in the Lab frame. If the process is forbidden energetically, program outputs "E" allowing the threshold energy to be found.
2. Single Particle Decays—By specifying M2 = 0 (target), the problem will be treated as a decay, and similar tables to the above will be printed.

DECUS No. 5/8-50
Additions to Symbolic Tape Format Generator (DEC-8-21-U)

Performs further useful functions by the addition of a few octal patches. By making the appropriate octal patches via the toggles, the Format Generator can also format FORTRAN tapes, shorten tape by converting space to tabs, and convert the type of tape.

A short binary tape may be made and added on to the end of 8-21-U to "edit" an original tape that was punched off-line.

The rubout character will cause successive deletion of the previous characters until the last C. R. is reached but not removed. The use of "←" will cause the current line to be restarted. Thus an input tape may be prepared off-line without attention to format spacing, mistakes corrected as they occur, and finally passed through the Format Generator to create a correctly formatted, edited, and line-fed on either rolled or fanfold paper tape.

DECUS No. 5/8-51
Character Packing and Unpacking Routines

ASCII characters may be packed two to a word and recovered. Control characters are also packable but are preceded by a 37 before being packed into the buffer. The two programs total 63_{10} words.

DECUS No. 8-52
Tiny Tape Editor

This Tiny Tape Character Editor fits in core at the same time as the PAL III or MACRO-8 assemblers. A tape may be duplicated at three speeds and stopped at any character for insertion or deletion. The toggle switches control the speed and the functions desired.

The program occupies 72_{10} registers.

DECUS No. 5/8-53
COPCAT

COPCAT is a tape to tape copy routine for PDP-5 and PDP-8 DECtape.

DECUS No. 5/8-54
Tic-tac-toe Learning Program—T^3

This program plays Tic-tac-toe basing its moves on stored descriptions of previously lost games. The main program is written in FORTRAN. There is a short subroutine written in PAL II used to print out the Tic-tac-toe board. The program comes already educated with about 32 lost games stored. Requires FORTRAN Object Time System.

DECUS No. 5/8-55
PALEX—An On-Line Debugging Program for PDP-5 and PDP-8

One problem with programs written in Program Assembly Language (PAL) for operation on a PDP-5/8 computer is the danger of an untested program being self-destructive, running wild, destroying other programs residing in memory such as loading programs. PALEX prevents any of the above unwanted operations from occurring while it gives the operator-programmer valuable debugging information and enables him to make changes in his program and try out the modified program. Once running, PALEX cannot be destroyed by any program or instruction in memory, the operator need not touch any manual console controls, and all required information is printed in easy-to-read format on the Teletype console.

DECUS No. 5/8-56
Fixed Point Trace No. 1

A minimum size monitor program which executes the users program one instruction at a time and reports the contents of the program counter, the octal instruction, the contents of the accumulator and link and the contents of the effective address by means of the ASR-33 Teletype. Storage Requirements: two pages.

DECUS No. 5/8-57
Fixed Point Trace No. 2

Similar to Fixed Point No. 1 except that the symbolic tape provided has a single origin setting instruction of (6000). Any four consecutive memory pages can be used, with the exception of page zero, by changing this one instruction.

DECUS No. 8-58
One-Page DECtape Routine (522 Control)

A general-purpose program for reading, writing, and searching of magnetic tape. This program was written for the Type 552 Control. It has many advantages over both the standard DEC routines and also over the DECUS No. 5-46. The routines are one-page long and can be operated with the interrupt on or off. The DEC program delays the calling program while waiting for the unit and movement delays to time-out. This routine returns control to the calling program. This saves ¼ second every time the tape searches forward and half that time when it reverses. In addition, it will read and write block 0,. This program is an advantage over the previous one-page routines in that it allows interrupt operations, doesn't overflow by one location, interrupts the end zone correctly and not as an error, and provides a calling sequence identical to the DEC program.

DECUS No. 8-59
PALDT—PAL Modified for DECtape (552 Control)

When assembling programs, PALDT requires that the symbolic tape be read in only once. The program writes on the library tape itself after finding the next available block from the directory. During pass 0 the tape is read in using the entire user's symbol table. During passes 1, 2, 3, as much

of the symbol table is used as possible. This means the fewest tape passes as possible. As an added advantage pass 0 ignores blank tape, leader-trailer, line feeds, form feeds, and rub outs; saving space. The whole program decreases the users symbol table by only three pages: one for the DECtape program above, one for pass 0, and one for the minimal length read in buffer.

DECUS No. 8-60
Square Root Function by Subtraction Reduction

A single precision square routine using EAE. This routine is usually faster than the DEC routine and can easily be modified for double precision calculation at only twice the computation time.

DECUS No. 8-61
Improvement to Digital 8-9-F Square Root

An improved version of the DEC Single Precision Square Root Routine (without EAE). Saves a few words of storage and execution is speeded up 12 per cent.

DECUS No. 5-63
SBUG-4

SBUG-4 allows the PDP-5 to execute one instruction of any given program at a time, returning to SBUG-4 following each instruction and printing out the contents of various registers. This permits following the path of a program which has gone astray or examining some defective operation.

DECUS No. 5/8-64
DECtape Programming System

This program provides rapid access to DEC software and utilizes routines through the use of DECtape. Programs may be stored, edited, assembled, listed, or executed without reliance upon paper tape.

May be used with both TC01 and 552 DECtape Controls.

DECUS No. 8-65
A Programmed Associative Multichannel Analyser

The program describes the use of a small computer as an associate analyser with special reference to the PDP-8. The advantages and limitations of the method are discussed in the write-up, and general program algorithms are presented.

DECUS No. 8-66
Editor Modified for DECtape

This program consists of modifications to the Digital 8-1-S Symbolic Editor to enable reading and writing on DECtape. This results in considerable time savings in assembling PAL programs since PAL has also been modified to accept the symbolic program directly from DECtape. The DECtape compatibility is also useful for storing text for later use and for regaining Editor memory space lost due to delete and change commands.

In addition, the overflow detection routine is now foolproof and results in a HALT.

Storage: Editor <0, 1461>

Modifications: <1462, 1502> <6376, 7177>

DECtape Routines: <7200, 7577>

Equipment: PDP-8 with EAE, ASR-33, DECtape

DECUS No. 8-67
PAL Modified for DECtape Input

This program consists of modifications to the Digital 8-3L-S PAL Assembly Program to enable it to obtain the symbolic program to be assembled from DECtape (in addition to paper tape), outputting the assembled program in the usual manner. (The symbolic program is written onto DECtape by use of the "Editor Modified for DECtape" Program.) The modification also makes it possible to assemble sections of programs in any order, and to intersperse sections or commands from the keyboard with those from DECtape. The resulting assembly is limited in speed mainly by the punching of the assembled program during Pass 2, and Pass 1 is speeded considerably. The modifications also include a tabulator interpreter, so that Pass 3 listings are produced in tabulated format.

Storage: PAL III <0, 3561> plus symbol table

Modifications: <6555, 7177>

DECtape Routines: <7200, 7577>

Equipment: PDP-8 with EAE, ASR-33, DECtape

DECUS No. 8-68A
LABEL Program (Teletype Punch)

The LABEL Program punches labels for paper tapes on the Teletype punch. When a key on the Teletype keyboard is depressed, no echo is performed, but a few characters of tape are punched which form the outline of the character associated with the key. Outlines are punched for all characters whose code is between 240 and 337.
All characters have a width of 5 lines of tape and a height of 6 columns. Column 7 of the tape is never punched and column 8 is always punched, so that the labels are underlined. Three extra lines are normally punched between each character to provide separation. The user may alter the number of separation lines by depressing one of the control keys at any time.
The program occupies locations 200 through 677 of any memory field, (locations 400-677 of Readable Punch, DECUS No. 8-106).

DECUS No. 5/8-69
LESQ29 and LESQ11

The purpose of the program is to fit the best sequences of parabolas to a given 400 point data curve in order to remove extraneous noise; rather than rely on a single 400 point parabola least squares fit to approximate a given data curve. Approximately 400 individual parabolas are computed as follows.

LESQ29

Data values 1 through 29 are subjected to a second order Least Squares fit. The median point of the resulting parabola (point #15) is then substituted for the original data value #15.*

A second parabola is then computed using data values 2 through 30. The median point of this parabola (point #16) is then substituted for point #16 of the original data curve.

This procedure is repeated until all data values have been replaced (except for the first and last 14 points which are excluded by the mechanics of the operation).

LESQ11

Process identical to LESQ29 except that an 11 rather than a 29 point smooth interval is used. First point replaced is point #6, and only the first and last 5 points are excluded from smoothing.

LESQ11 will preserve higher frequency data than LESQ29 for a given data curve with constant time between data points.

Minimum Hardware: 4K Memory PDP-5 or PDP-8, Teletypewriter (plotter, DECtape optional)

Other Programs Needed: Floating Point Package and appropriate data handling routines.

Storage Requirements: (LESQ11: 400-564; 700-716)
(LESQ29: 400-564; 700-751)

Execution Time: (PDP-5) LESQ11: 1 minute.
LESQ29: 2.5 minutes.

Restrictions: Positive integer data $<3777_8$; time between data points constant.

*See B. J. Power, R. N. Hagen, S. O. Johnson, "SPORT, A System for Processing Reactor Transient Data on the IBM-7040 Computer," pp. 4-8, AEC Research and Development Report (IDO-17078), Available from: The Clearinghouse for Federal Scientific and Technical Information, National Bureau of Standards, U. S. Department of Commerce, Springfield, Virginia.

DECUS No. 8-70
EAE Routines for FORTRAN Operating System (DEC-08-CFA3)

These are two binary patches to the FORTRAN Operating System which utilizes the Type 182 EAE hardware for single precision multiplication and normalization, replacing the software routines in FOSSIL (the operating system). The binary tape is loaded by the BIN Loader after FOSSIL has been loaded. Execution time of a Gauss-Jordan matrix inversion is reduced by approximately 30%.

Minimum Hardware: PDP-8 with Type 182 EAE

Other Programs Needed: FORTRAN Operating System (DEC-08-CFA3-PB) dated March 2, 1967.

DECUS No. 8-71
Perpetual Calendar

The program is designed as a computer demonstration. When a valid date is fed into the computer, the corresponding day of the week is typed out. If an invalid date is given, "YOU GOOFED, TRY AGAIN" is typed out. The program is based on the Gregorian Calendar and is, therefore, limited to years between 1500 and 4095. The upper limit being due to the computer's capacity.

Minimum Hardware: 4K storage, ASR-33 Teletype

Storage: 20-1333

DECUS No. 8-72
Matrix Inversion — Real Numbers

The program inverts a matrix, up to size 12 x 12, of real numbers. The algorithm used is the Gauss-Jordan method. A unit vector of appropriate size is generated internally at each stage. Following the Gauss sweep-out, the matrix is shifted in storage, another unit vector is generated, and the calculation proceeds.

Other Programs Needed: FORTRAN Compiler and FORTRAN Operating System.

Storage: This program uses essentially all core not used by the FORTRAN Operating System.

Execution Time: Actual computation takes less than 10 seconds. Data read-in and read-out may take up to five minutes.

DECUS No. 8-73
Matrix Inversion — Complex Numbers

The program inverts a matrix, up to size 6 x 6, of complex numbers. The algorithm used is the Gauss-Jordan method, programmed to carry out complex number calculations. A unit vector of appropriate size is generated internally. Following the Gauss sweep-out, the matrix is shifted, another unit vector is generated, and the calculation proceeds. The print-out of the matrices uses the symbol J to designate the imaginary part, e.g. A = a + jb.

Other Programs Needed: FORTRAN Compiler and FORTRAN Operating System.

Storage: This program uses essentially all core not used by the FORTRAN Operating System.

Execution Time: Actual computation takes less than 10 seconds. Data read-in and read-out may take up to five minutes.

DECUS No. 8-74
Solution of System of Linear Equations: AX = B, by Matrix Inversion and Vector Multiplication

This program solves the set of linear algebraic equations AX = B by inverting matrix A using a Gauss-Jordan method. When the inverse matrix has been calculated, it is printed out. At that point, the program requests the B-vector entries. After read-in of the B-vector, the product is computed and printed out. The program then loops back to request another B-vector, allowing the system to solve many sets of B-vectors without the need to

invert matrix A again. Maximum size is 8 x 8.

Other Programs Needed: FORTRAN Compiler and FORTRAN Operating System.

Storage: This program uses essentially all core not used by the FORTRAN Operating System.

Execution Time: Actual computation is less than 10 seconds. Data read-in and read-out may take up to five minutes.

DECUS No. 8-75
Matrix Multiplication — Including Conforming Rectangular Matrices

This program multiplies two matrices, not necessarily square but which conform for multiplication.

Other Programs Needed: FORTRAN Compiler and FORTRAN Operating System.

Storage: This program uses essentially all core not used by the FORTRAN Operating System.

Execution Time: Actual computation takes less than 10 seconds. Data read-in and read-out may take up to five minutes.

Author's comments regarding the four matrix routines: DECUS Nos. 8-72, -73, -74, -75.

"Each program has been written in FORTRAN for use on the basic PDP-8. The printed output of each program has been organized to efficiently and effectively use the Teletype, yet maintain a readable and meaningful output format. This has been done at the expense of optimizing storage requirements. Thus, each program may be changed to handle slightly larger cases by cutting down on the print-out information. Since the source programs are in FORTRAN, a user may readily change the program to suit his own requirements.

"Another common feature of each program is a print-out of the input data from core. This has been found desirable for checking that data was read in properly, and for purposes of having an accurate record of a given calculation. Also, since each program requires a large amount of data input, it is suggested that a data tape be made prior to running the program.

"The matrix inversion scheme used is straightforward and gives good results on run-of-the-mill matrices. However, error build-up is quite rapid on an ill-conditioned matrix. This is partly due to errors in the fifth and sixth decimal place caused by the floating point conversion at read-in time, and also to the limited mantissa carried in the PDP-8 floating point word."

DECUS No. 8/8S-76
PDP NAVIG 2/2

This program utilizes the output of the U. S. Navy's AN/SRN-9 satellite navigation receiver to obtain fixes on a PDP-8 or PDP-8/S. This program, except for some details of input and output, follows very closely NAVIG2 written for the IBM 1620 which in turn is derived from the TRIDON program written at the Applied Physics Laboratory of Johns Hopkins University for the IBM 7090.

PDP NAVIG 2/2 is written in PAL III for a 4096 core machine using the ASR-33. Floating point numbers using two 12-bit words as mantissa and

one 12-bit word as exponent are employed. The accuracy is slightly less than that using 7 decimal digits per word.

DECUS No. 8/8S-77
PDP-8 Dual Process System

The purpose of this system is to expedite the programming of multiprocessing problems on the PDP-8 and PDP-8/S. It maximizes both the input speed and the portion of real time actually used for calculations by allowing the program to run during the intervals between issuing I/O commands and the raising of the device flag to signal completion of the command. The technique also allows queuing of input data or commands so that the user need not wait while his last line is being processed, and so that each line of input may be processed as fast as possible regardless of its length. The system uses the interrupt facilities and has less than a 3% overhead on the PDP-8/S (about .1% on the PDP-8).

This method is especially useful for a slower machine where the problem may easily be calculation limited but would, without such a system, become I/O bound.

The program may also be easily extended to handle input from an A/D converter. Here, the input would be buffered by groups of readings terminated either arbitrarily in groups of N or by zero crossings.

The system requires 600_8 registers for two TTY's plus buffer space. Several device configurations are possible.

This program can increase the I/O to computation efficiency of some programs by 100%. It can do this even for a single Teletype. Each user will probably want to tailor the program to his individual needs.

DECUS No. 8-78
Diagnose: A Versatile Trace Routine for the PDP-8 Computer with EAE

This trace routine will track down logical errors in a program (the "sick" program). Starting at any convenient location in the "sick" program, instructions are executed, one at a time, and a record of all operations is printed out via the Teletype. To avoid tracing proven subroutines, an option is provided to omit subroutine tracing. The present routine is significantly more versatile than two other trace routines in the DECUS library (DECUS Nos. 8-56 and 8-57 — Biavati) for the PDP-8 in that it is able to trace "sick" programs containing floating-point, extended arithmetical, and a variety of input-output instructions. Diagnose is, however, at a disadvantage compared with Biavati's first routine (DECUS No. 8-56) in requiring more memory space (five pages as opposed to two); and compared with his second routine (DECUS No. 8-57) in not possessing the trace-suppression features of the latter. The mode of operation of Diagnose is quite different from that of the trace routines of Biavati.

Minimum Hardware: PDP-8 with EAE

Other Programs Needed: Floating Point Package needed for floating point tracing.

Storage: 5(4) pages of memory.

Miscellaneous: Program is relocatable.

DECUS No. 8-79
TIC-TAC-TOE (Trinity College Version)

This TIC-TAC-TOE game is programmed, using internal logic, so that the computer will either win or stalemate, but not lose a game. Either the player or the computer may choose to go first. At the termination of a game, the program restarts for the next game by typing anew the grid code to be followed.

DECUS No. 8-80
Determination of Real Eigenvalues of a Real Matrix

This is a two-part program for determining the real eigenvalues of a real-valued matrix. The matrix does not have to be symmetric. Part I uses the power method of iterating on an eigenvector to determine the largest eigenvalue of the matrix. Part II then deflates the matrix using the results of Part I so as to produce a matrix of order one less than that solved for in Part I. Part I can then be reloaded, and the next eigenvalue in line may be calculated. In this, all the real eigenvalues may be computed in order.

DECUS No. 8-81
A BIN or RIM Format Data or Program Tape Generator

This program enables a PDP-8 operator to generate tapes under Teletype control in RIM or PAL BIN format without formal assembly, assuming the operator knows the octal codes corresponding to each instruction. This is particularly useful when one is dealing with small programs for testing interface equipment or when making small modifications to large programs when one does not wish to spend time reassembling the whole program. Often during program debugging, changes are repeatedly toggled into core manually, which leaves no permanent record of the changes made and is prone to error. Tapes generated using this program can be appended to existing BIN or RIM tapes and can then be loaded with the original tape into core with the appropriate loader. Another use of this program is in the preparation of data tapes in RIM or BIN format so that data can be loaded straight into PDP-8 core via the usual leaders. The program also generates leader/trailer code and a checksum under program control.

Storage: Program occupies locations 6000-6077.

DECUS No. 8-82
Library System for 580 Magnetic Tape (Preliminary Version)

The system provides for storing program files (or other files) on the 580 Magnetic Tape with PDP-8, and recalling them at will without altering the state of the rest of the computer. In general principle, it is similar to the DECtape Library System, and the only effective storage requirement is the last page of memory.

At present, the system consists of three programs known as BOOTSTRAP 1, BOOTSTRAP 2, and the LIBRARY Routines.

Bootstrap 1 is a minimal loader program which resides in the last page of memory. Its function is to rewind the tape and load Bootstrap into the last page, automatically transferring control to it. Bootstrap saves the area of core to be used by the system as a record on the magnetic tape, loads the Library Routines into core, and transfers control to them.

The Library Routines comprise a Directory of the files on tape, an Input-Output package, enabling communication with the Teletype, and four

system programs:

LIst: Types out the names of files in the Directory

CAll: Transfers a file into core and exits

DUmp: Writes a file on tape, rewrites the Directory, and exits

EXit: Restores the computer to its original state, with Bootstrap 1 and BIN on the last page.

The magnetic tape subroutines and some control functions are included in Bootstrap 2. Each entry in the directory consists of three words: the name of the file, its first location in core, and the number of words it occupies. The capacity of the directory is 22_{10} entries.

DECUS No. 8/8S-83A and B
Octal Debugging Package (With and Without Floating Point)

This program is an on-line debugger which will communicate with the operator through the ASR-33 Teletype. It allows register examination and modification, octal dumping, binary punching, multiple and simultaneous breakpoints, starting a program, and running at a particular location with preset AC and link. ODP is completely relocatable at the beginning of all pages except page zero, and is compatible with the PDP-5, the PDP-8, and the PDP-8/S.

Requirements: The high version of ODP requires locations 7000-7577. The low version requires locations 0200-0777. All versions will require three pages. Also, location 0002 is used for a breakpoint pointer to ODP.

Equipment: The standard PDP-8 with ASR-33 Teletype is required. A high-speed punch is optional.

DECUS No. 8-84
One-Pass PAL III

This is a modification to Digital 8-3L-S. It is for use on an 8K PDP-8 with ASR 33. The principle of the modification is to store the incoming characters during Pass I into the memory extension and to take them from there during Pass 2 and 3. Source programs must be limited to 4095 characters. This modification can save about 40% of assembly time.

Operation of the program is the same as for PAL III except that the reading of the source program for Pass 2 and 3 need not be repeated. For these passes, one simply pressses CONTINUE after setting the correct switches.

Restriction: The program does not work with high-speed reader and punch.

DECUS No. 5/8-85
Set Memory Equal to Anything

This program will preset all locations to any desired settings. Thus, combining a memory clear, set memory equal to HALT, etc. into a single program. The program is loaded via the switch registers into core.

DECUS No. 8-87
XMAP
This program types on TTY keyboard the contents of the DECtape directory. This list includes the name of the program, its initial block number, the amount of blocks used, the starting address and the location(s) of the program in core. The above restriction is only a format restriction due to the line length on the TTY unit. At present, this program is operational only with the TC01 control; however, the symbolic version may be modified for use with the 552 control.

Storage: 0000-1232, 6000-6577 (directory)

Restrictions: Each program on tape is assumed to occupy no more than three successive sequences of memory pages.

DECUS No. 8-88
DECtape Symbolic Format Generator
These are DECtape versions of the Symbolic Tape Format Generator, Digital 8-21-U, that operate under the DECtape Programming System, DECUS 5/8-64. They provide neat formats for symbolic files generated with XEDIT, and a means to get symbolic programs out on paper. They compact a program containing extra spaces and give the number of blocks actually used in the output file. The library tape is executable on TC01 equipment only, but the write-up gives instructions for altering it for 552 equipment.

Other programs needed: XRDCT, XWDCT, XBUFF
(DECUS 5/8-64)

Storage: 0-4777(8)

DECUS No. 8-89
XOD — Extended Octal Debugging Program
XOD is an octal debugging program for a PDP-8 with extended memory which preserves the status of program interrupt system at breakpoints. The program occupies locations 6430 through 7577 of any memory field.
From the on-line Teletype, the user can examine and modify the contents of any memory location. Positive and negative block searches with a mask may also be performed.
XOD includes an elaborate breakpoint facility to help the user run sections of his program. When this facility is used, the debugger also uses locations 0005, 0006, and 0007 of every memory field.
The ability to punch binary tapes is not included in XOD.

DECUS No. 5/8-90
Histogram on Teletype
This routine provides a means of plotting histograms on the Teletype when there is no CRT display available or of making a permanent copy of a CRT display. Input to the routine consists of a vertical scaling factor, the size of the table to be plotted (limited only by the size of the Teletype print line), the starting address of two core areas: one containing the data to be plotted, and one for use as temporary storage by the machine.
Storage: 128_{10} words plus tables

DECUS No. 8-91
MICRO-8: An On-Line Assembler
Micro-8 is a short assembler program for the PDP-8 that translates typed mnemonic instructions into the appropriate binary code and places them in specified memory locations immediately ready to function. It processes the typed instructions by a table-lookup procedure.

It is especially useful for programs of less than one page which are to be run immediately. Only octal (not symbolic) addresses may be specified, but the user has control of the zero page and indirect addressing bits. An octal type-out routine permits examination of any memory location.
Storage: 3200 - 4200
Restrictions: Micro-8 is quite capable of modifying itself.

DECUS No. 8-92
Analysis of Pulse-Height Analyzer Test Data With a Small Computer
This PDP-8 computer program is used in the evaluation of test data for multi-channel pulse-height analyzers. The program determines integral and differential non-linearities and examines smooth spectra of radioactive decay.

DECUS No. 8-93
CHEW — Convert Any BCD to Binary — Double Precision
This subroutine converts a double precision (6 digit) unsigned-integral-binary coded decimal (BCD) number with bit values of 4, 2, 2, and 1 to its integral-positive-binary equivalent in two computer words. It is possible to change the bit values to any desired values and thereby convert any BCD number to binary.
Storage: 0109_{10}

DECUS No. 8-94A
BLACKJACK
This program enables a person to play Blackjack with the computer. The computer acts as dealer and keeps track of bets, cards played, etc.
Storage: $0 - 3777_{8}$

DECUS No. 8-94B
Patch for BLACKJACK
This patch contains two overlays for the Blackjack Program (DECUS No. 8-94A). The first eliminates the need for the EAE hardware, the second allows one to "double down" on any two cards with the instruction "D" (0 response to "HIT?" is made invalid).
With these overlays the Blackjack Program will run on any 4K PDP-5, PDP-8, 8/S, or 8/I.

DECUS No. 8-95.
TRACE for EAE
Trace interpretively executes a PDP-8 program. At the same time a printout is provided of the contents of the program counter, the instruction, the link, accumulator, and multiplier-quotient registers, and where applicable the effective address, and the contents of the effective address. This printout may be for all or a selected type of instruction within selected memory bounds. The program is capable of handling any PDP-8 instruction including IOT, two-word EAE, and interrupt instructions. Trace cannot be destroyed by the program being traced while Trace is in control.
Minimum Hardware: PDP-8 with Type 182 EAE, ASR-33 Teletype
Storage: 400_{8} or 500_{8} Locations

DECUS No. 8-96
J Bessel Function (FORTRAN)
This program computes the J Bessel Function for a given argument and order. It is complete PDP-8 FORTRAN program that operates in a conversational mode.
Other Programs Needed: FORTRAN Compiler/Operating System

DECUS No. 8-97
GOOF
A one-page program which allows insertion of instruction (xxxx) in location (nnnn) by means of the TTY keyboard. A feature of automatically incrementing the current address permits rapid insertion of blocks of data or instructions. Typing "RUB-OUT" reinitializes the program.
Storage: 175_8 locations (1 page)

DECUS No. 8-98
3D DRAW for 338
This program is a demonstration of the capabilities of the 338 system. The program allows the user to sketch three dimensional objects on the scope and rotate them in real time. The equipment required consists of a basic 338.

DECUS No. 8-99A
Kaleidoscope
The program creates pictures on the PDP-8 or PDP-8/S with 34D Display. They are varied by manipulating the sense switches (within the range 0000 - 0007) The program was submitted without comments by an anonymous donor.

DECUS No. 8-99B
Kaleidoscope - 338
The program creates varied pictures by manipulating the buttons of the 338 Display pushbutton bank (within the range 0000 - 0007).
Storage: 200_8 - 274_8.

DECUS No. 8-100
Double Precision Binary Coded Decimal Arithmetic Package
Consists of the following routines:
BCDADD — The single precision BCD addition routine is the basic component of the BCD arithmetic package. This routine functions simply by masking out and adding together corresponding BCD digits (i.e., four bits) and checking for carry (i.e., when the sum of two four-bit numbers is greater than 9 (1001)).
MPYBCD — This routine multiplies a single precision (three digit) number times a double precision one to produce another double precision number. Overflow is indicated in the link; the arguments are not affected.
SUBBCD — One double precision BCD number is subtracted from a second by this routine. It uses a 9's complement routine and the double precision add routine.
DOLOUT — Special formats: ("$XXXX : YY "); ("XXXXXX "); (3 non-printing data codes); ("XXX ").

DECUS No. 8-101
Symbolic Editor With View
This program is an extended version of the standard PDP-8 Symbolic Editor (high-speed I/O) program. One extra command has been added, "V", which takes the lines specified by the arguments and displays them on the CRT (338). The program, otherwise, operates in the same way as the Editor.
The following pushbutton options are provided:

0: Count Up Scale
1: Count Down Scale
2: Count Up Intensity
3: Count Down Intensity

Minimum Hardware: 8K PDP-8 and VC-38 character generator.

DECUS No. 8-102
A LISP Interpreter For The PDP-8
LISP is a programming language for list manipulation. The system is particularly suitable for conversational use and teaching. There are very few restrictions to the language apart from the total storage space. The system is designed to operate on a basic 4K PDP-8 and an ASR-33 Teletype. More than half of the storage is used as list space.

DECUS No. 8-103 A
Four Word Floating Point Function Package
This program package, written for use with Digital's Four Word Floating Point Package (DEC-08-FMHA-PB), includes subroutines to evaluate square, square root, sine, cosine, arctangent, natural logarithm, and exponential functions.

DECUS No. 8-103 B
Four Word Floating Point Rudimentary Calculator
This is a minimum space program to perform calculations with the 10.5 decimal place precision of Digital's Four Word Floating Point Package (DEC-08-FMHA-PB), and uses the Four Word Floating Point Function Packages (DECUS 8-103A). Operations are performed in the sequence in which they are entered. One storage register is provided. Up to five user-defined operation routines may be called.

DECUS No. 8-103 C
Four Word Floating Point Output Controller With Rounding
This subprogram is almost identical to the output controller for the Three Word Floating Point Package (Digital 8-5-S) with the rounding addition (DECUS 8-44) except that the Four Word Floating Point Package (DEC-08-FMHA-PB) is used.

DECUS No. 8-103 D
Additional Instructions for Use With Four Word Floating Point Packages
These subroutines allow the Four Word Floating Point Interpreter to perform the operations: read a floating point number, skip positive floating point accumulator, skip zero floating point accumulator, no operation, unconditional jump, negate floating point accumulator, and halt. The two skip instructions and the jump instruction allow forward or backward jumping up to 15 locations from location of instruction.

DECUS No. 8-104
Card Reader Subroutine for PDP-8 FORTRAN Compiler
Modifications and additions which allow the PDP-8 FORTRAN Compiler to read source programs from cards. The standard FORTRAN card format is used with only minor modifications.
Minimum Hardware: 8K PDP-8 and a Type CR01-C Card Reader

DECUS No. 8-105
D-BUG
D-BUG is an aid used in debugging PDP-8 programs by facilitating communication with the program being run. Communication between operator and program is via the ASR-33 Teletype. D-BUG is similar to DEC's program ODT II (DEC-08-COA1-PB); however, it uses the DEC Floating Point Interpreter (Digital 8-5-S).
Two modes of operation are possible, fixed and floating point. D-BUG features include register examination and modification, control transfer, octal dumping, and instruction trap-outs to D-BUG control. Registers containing floating point numbers may also be examined, and breaktraps can be inserted in floating point programs.

DECUS No. 8-106
Readable Punch
This program enables the user to type a character on the keyboard and produce the character in readable form on paper tape. The program uses the high-speed punch. The readable characters on tape are produced by means of a table which contains the format of a 6 x 5 matrix using three words of storage per character to be punched. In addition, channel 8 is punched throughout. The program is terminated by typing a carriage return which generates 6 inches of tape. (Reference DECUS No. 8-68A)

DECUS No. 8-107
CHESSBOARD for the PDP-8/338
This program displays a chessboard on the screen of a DEC 338 Display with all thirty-two chessmen set up on their initial board positions. There is no provision to move them about the board; it is just a demonstration picture. The program occupies 03000 throught 04230.

DECUS No. 8-108
INCMOD — Increment Mode Compiler (338)
The INCMOD program for the DEC 338 Display allows the user to build a display subroutine composed of increments only. The user inputs information by pointing with the light pen. The program displays the figure he is constructing in each of the four available scale settings. The program is of value as a demonstration and may be of help for maintenance purposes. It occupies locations 00000-01231 and builds the increment mode display file beginning at location 01232.

DECUS No. 8-109
SEETXT Subroutine
SEETXT is a subroutine for the DEC 338 Display which can be called instead of the normal typeout subroutine. In addition to typing, it displays all printed characters on the screen corresponding to the last twenty lines which have been typed out.
The program includes the option of suppressing the typing, so that output can occur at a much higher rate than ten characters per second. The user has the option of controlling the length of a delay loop in the subroutine, so the output rate may range from nearly immediate to Teletype rate.
The maximum number of lines displayed, the scale, and intensity may be altered at any time. There is also the option of clearing the screen or displaying a blinking marker at the current typing position.
SEETXT is a subroutine for a standard 338, and does not depend upon the VC38 Character Generator. It occupies locations 0016 through 1500 (approximately) of any memory field and its entry point (for a JMS) is location 0200.

DECUS No. 8-110
DIREC — Directory Print
DIREC is a system program to be used with the PDP-8 Disk Monitor System. The program lists, on the on-line Teletype, an index of the file directory for the disk. The user has the option of seeing the index to system files or users files, or both.
DIREC can also be used in conjunction with the SEETXT Subroutine for the 338 Display to obtain a listing of the directory on the display screen.

DECUS No. 8-111
DISKLOOK
DISKLOOK is a small utility program for a PDP-8 with a 32K DF32 Disk. Using the on-line Teletype, the user may examine and alter any location (in octal) on the disk. Masked searches are also available. DISKLOOK occupies locations 200 through 777 of any memory field.

DECUS No. 8-112
Sentence Generator
This program generates random English language sentences, using a dictionary (provided by the user) of ten basic word groups (A - J). The dictionary is used in conjunction with a random number generator and a syntactical algorithm to provide an output of randomly constructed English language sentences.
The program is an execellent vehicle for computer demonstration purposes. It may also be used to advantage in English teaching programs to aid students in perceiving sentence structure and errors in the use of words.

DECUS No. 8-113
Conversion of Friden (EIA) to ASCII
This program will translate tapes prepared on a Friden Flexowriter (EIA) into ASCII for direct assembly, further editing, or feeding into the FORTRAN program. Alphabetic characters may be in either upper or lower case. The program uses a table lookup and comparison with the negative complement of the EIA character, then outputs the corresponding ASCII character.
Storage: 213_8 including 2 auto-index registers

DECUS No. 8-114
Rounded Decimal Output Modification for PDP-8 FORTRAN
The program loads over the PDP-8 FORTRAN Operating System (DEC-08-AFA3-PB) and provides output in conventional decimal form: rounded, aligned, and with plus sign, leading zeros (other than one, in the case of fractional numbers), and trailing decimal point replaced by spaces. The FORTRAN trigonometrical routines are over-written. The source program must begin with two statements assigning integer variables representing, respectively, the number of digits required to the right of the decimal point, and the total number of digits (these can be reassigned, by program or manually). Output is called in the normal way, i.e. by TYPE statements referring to FORMAT statements containing the symbol E. If output of a number is not possible in the format requested, the decimal point is shifted to the right in the field; if formatted output is still impossible, or if zero or negative total digits were requested, output reverts to "E" format.
Restrictions: FORTRAN source language programs must begin with two special statements defining format required.

DECUS No. 8-115
Double Precision Integer Interpretive Package
This program is a Double Precision Integer Interpretive Package similar in operation to the Floating Point Package (Digital 8-5-S). It consists of addition, subtraction, multiplication, division, load, store, jump and branch subroutines coupled to an interpreter. It allows direct and indirect addressing in the normal assembly language manner. The operation is faster and more compact than the collected individual double precision subroutines. The program requires fourteen words on page zero and an additional two pages of memory.
Minimum Hardware: Basic PDP-5, -8, -8/S or -8/I

DECUS No. 8-116
PDP-8 Automatic Magnetic Tape Control (Type 57A) Library System
The PDP-8 Automatic Magnetic Tape Control (Type 57A) Library System is a series of bootstrap programs which load library programs into memory from an IBM-compatible magnetic tape read using a Type 57A Automatic Magnetic Tape Control. A program is selected by entering the appropriate code number into the switch register on the computer console.
A copy of the IBM-compatible library tape may be obtained by sending a ½" magnetic tape to the authors.

DECUS No. 8-117
A PDP-8 Interface for a Charged-Particle Nuclear Physics Experiment
Documentation (only) describing an interface constructed to use a PDP-8 computer with a charged particle detector system employing three solid-state detectors and flight-time analysis. Up to 48 bits from each randomly-occurring event are transferred through the data (break) channel to a hardware-selected buffer region in the core of a PDP-8 computer. Designed for use as a magnetic tape analyzer for the most complex cases, the system assumes that the 48 bits originate in flag bits set by fast logic and in (presently four) amplitude digitizers, all of which are assumed to contain information for the same event. The system includes some limited capability for controlling the course of the experiment, and provides for readout through the computer of a series of external fast counters. The report summarizes the design concepts, shows schematic flow diagrams, defines the computer instructions associated with the interface system, and gives simple model programs to illustrate methods of application.

DECUS No. 8-118
General Linear Regression
The major section of this program is the "Main Arithmetic IX" which consists of four initializing statements; an input section; a weighting section; a section which cumulates means, sums of squares, etc.; a section which calculates the relevant regression coefficients, etc.; and a section which calculates confidence limits as variances.
The section which calculates the relevant regression co-efficients allows for both cases of linear regression, and in the computation of standard error of the intercept, uses (N-2) degrees of freedom to provide a better estimate for small values of N while providing negligible differences from conventional calculation when N is large.
The section which calculates confidence limits as variances provides a calculation of the variance of the error of the estimate of the dependent variable again using (N-2) degrees of freedom for the general case. This calculation is fully corrected for both random variance within the tested population of data and for the difference between the independent variable and the mean of the independent variable for the population of data.

DECUS No. 8-119
Off-Line TIC-TAC-TOE
TIC-TAC-TOE is a self-learning program which will improve its game as it plays. Whenever its human opponent wins, the program changes its strategy such that it can never be beaten again in the same way. Thus, the program gains "experience" every time it loses. The program will punch its experience on paper tape in binary format on request. This experience tape can be reread by the program at any time and will reset the program to the level of experience it had when the tape was punched. The program will notify the operator if any error is made in reading the experience tape and gets very upset if the player tries to cheat. The program occupies locations 10-4000 (approximately) and will operate with low or high speed tape input/output equipment.

DECUS No. 8-120
Disk/DECtape FAILSAFE
This program will punch the contents of the disk (or DECtape) onto paper tape which can be loaded back onto the disk using the same program. The paper tape is punched in 200_8 word blocks in binary format, with a checksum for each block. FAILSAFE simplifies and speeds the process of rebuilding the Disk System Monitor after running disk tests.
Minimum Hardware: PDP-8, -8/S, -8/I with 32K Disk or DECtape

DECUS No. 8-121
DECtape Handler
This program allows quick, controlled data-block transfers between the PDP-8 and DECtape. It reads, writes and searches in minimum time (interrupt mode), requires minimum place (overlay with last page BIN, RIM, DECSYS Loaders) and occupies only two blocks on tape (block 0 = System, block 1 = Return-System). It is protected against destruction and gives, after the transfer, the status levels for testing purposes. It is usuable as a Switch Register controlled program or as a subroutine with or without interrupt, giving the possibility of quick data storage, program shuffling and overlay technique with PDP-8 and DECtape.
Minimum Hardware: PDP-8, DECtape 552 Control

DECUS No. 8-122
SNAP—Simplified Numerical Analysis Program
SNAP is a computer language for real-time interactive computation which can be learned in less than one hour. It is particularly useful in teaching programming to beginners.
A unique feature of SNAP is its ability to interact on-line with other laboratory instruments. SNAP can accept electrical inputs directly and can read inputs from a real-time clock. Both of these functions are incorporated in a single SNAP instruction.
Another feature particularly useful for biological problems is Table Instructions. A list of 100 numbers may be entered from the keyboard or from punched paper tape.

DECUS No. 8-123
UNIDEC Assembler
The UNIDEC Assembler runs on the Univac 1108 and passes assembled PDP-8 code over the electronic link between the 1108 and PDP-8. The source statements are punched on cards for input into the 1108 in a format nearly identical to that of MACRO-8. A printed listing and the object code are produced as fast as the cards can be read.
Note: Source deck and documentation only available.

DECUS No. 8-124
PDP-8 Assembler for IBM 360/67
The 360/PDP-8 Assembler is a collection of programs written mostly in FORTRAN IV (G) which operate on the IBM 360/67. It assembles programs for PDP-5 and PDP-8 computers. Once a program has been assembled, it may be punched on cards, saved in a file, or transmitted through the Data Concentrator over data lines. It is also possible to obtain binary paper tapes by use of the Data Concentrator.
The Assembler follows the PAL III operation code and addressing conventions. The input format and program listing conventions are slightly different from those of PAL III, because it is organized around a line format, while PAL III is organized around a paper tape format.
Note: Source deck and documentation only available.

DECUS No. 8-125
PDP-8 Relocatable Assembler for IBM 360/67
The documentation available describes a method for segmenting PDP-8 programs for the purpose of facilitating program maintenance and residence in MTS (Michigan Terminal System) files. The method provides for program storage on a page-relocatable basis with relocation information contiguous to but not necessarily integral with text information. Linkages between separately assembled program segments are provided in a form very similar to those used in IBM System/360 systems.
Currently available utilities within MTS provide assembly and link-editing facilities, using programs stored either as punched card decks or in MTS files. Utilities are also included for the purpose of paper tape transcription either in PAL-compatible format or in a special format useful for dynamic loading via a data link to a remote machine. In addition to these MTS utilities, two relocating PDP-8 loaders are available which operate using the special dynamic-loading format. Each of these programs occupies one dedicated page of PDP-8 memory and operates in a multicore-bank environment. One of these programs is designed to operate as a stand-alone utility, while the other is designed to operate within the RAMP system.

DECUS No. 6/8-12
PDP-8 Assembler for PDP-6
Assembles PDP-8 programs written in PAL on a PDP-6 using any I/O devices.

APPENDIX 2
TABLES OF INSTRUCTIONS
PDP-8/L MEMORY REFERENCE INSTRUCTIONS

Mnemonic Symbol	Operation Code	Direct Addr. States Entered	Direct Addr. Execution Time (μsec)	Indirect Addr. States Entered	Indirect Addr. Execution Time (μsec)	Operation
AND Y	0	F, E	3.2	F, D, E	4.8	Logical AND. The AND operation is performed between the content of memory location Y and the content 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. ACj ∧ Yj = > ACj
TAD Y	1	F, E	3.2	F, D, E	4.8	Two's complement add. The content 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 AC0, the link is complemented. AC + Y = > AC
ISZ Y	2	F, E	3.2	F, D, E	4.8	Increment and skip if zero. The content of memory location Y is incremented by one. If the resultant content of Y equals zero, the content of the PC is incremented and the next instruction is skipped. If the resultant content of Y does not equal zero, the program proceeds to the next

PDP-8/L MEMORY REFERENCE INSTRUCTIONS (continued)

Mnemonic Symbol	Operation Code	Direct Addr. States Entered	Direct Addr. Execution Time (μsec)	Indirect Addr. States Entered	Indirect Addr. Execution Time (μsec)	Operation
						instruction. The incremented content of Y is restored to memory. If resultant Y = 0, PC + 1 = > PC.
DCA Y	3	F, E	3.2	F, D, E	4.8	Deposit and clear AC. 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. AC = > Y 0 = > AC
JMS Y	4	F, E	3.2	F, D, E	4.8	Jump to subroutine. The content of the PC is deposited in core memory location Y and the next instruction is taken from core memory location Y + 1. PC + 1 = > Y Y + 1 = > PC
JMP Y	5	F	3.2	F, D	4.8	Jump to Y. 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. Y = > PC

PDP-8/L GROUP 1 OPERATE MICROINSTRUCTIONS

Mnemonic Symbol	Octal Code	Sequence	Operation
NOP	7000	—	No operation. Causes a 1.5 μsec program delay.
IAC	7001	3	Increment AC. The content of the AC is incremented by one in two's complement arithmetic.
RAL	7004	4	Rotate AC and L left. The content of the AC and the L are rotated left one place.
RTL	7006	4	Rotate two places to the left. Equivalent to two successive RAL operations.
RAR	7010	4	Rotate AC and L right. The content of the AC and L are rotated right one place.
RTR	7012	4	Rotate two places to the right. Equivalent to two successive RAR operations.
CML	7020	2	Complement L.
CMA	7040	2	Complement AC. The content of the AC is set to the one's complement of its current content.
CIA	7041	2, 3	Complement and increment accumulator. Used to form two's complement.
CLL	7100	1	Clear L.
CLL RAL	7104	1, 4	Shift positive number one left.
CLL RTL	7106	1, 4	Clear link, rotate two left.
CLL RAR	7110	1, 4	Shift positive number one right.
CLL RTR	7112	1, 4	Clear link, rotate two right.
STL	7120	1, 2	Set link. The L is set to contain a binary 1.
CLA	7200	1	Clear AC. To be used alone or in OPR 1 combinations.
CLA IAC	7201	1, 3	Set AC = 1.
GLK	7204	1, 4	Get link. Transfer L into AC 11.
CLA CLL	7300	1	Clear AC and L.
STA	7240	2	Set AC = −1. Each bit of the AC is set to contain a 1.

Note: EAE microinstructions used on the PDP-8 & PDP-8/I are NOP's to the PDP-8/L.

PDP-8/L GROUP 2 OPERATE MICROINSTRUCTIONS

Mnemonic Symbol	Octal Code	Sequence	Operation
HLT	7402	3	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	3	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	1	Skip, unconditional. The next instruction is skipped.
SNL	7420	1	Skip if L $\neq$ 0.
SZL	7430	1	Skip if L = 0.
SZA	7440	1	Skip if AC = 0.
SNA	7450	1	Skip if AC $\neq$ 0.
SZA SNL	7460	1	Skip if AC = 0, or L = 1, or both.
SNA SZL	7470	1	Skip if AC $\neq$ 0 and L = 0.
SMA	7500	1	Skip on minus AC. If the content of the AC is a negative number, the next instruction is skipped.
SPA	7510	1	Skip on positive AC. If the content of the AC is a positive number, the next instruction is skipped.
SMA SNL	7520	1	Skip if AC $<$ 0, or L = 1, or both.
SPA SZL	7530	1	Skip if AC $>$ 0 and if L = 0.
SMA SZA	7540	1	Skip if AC $<$ 0.
SPA SNA	7550	1	Skip if AC $>$ 0.
CLA	7600	2	Clear AC. To be used alone or in OPR 2 combinations.
LAS	7604	1, 3	Load AC with SR.
SZA CLA	7640	1, 2	Skip if AC = 0, then clear AC.
SNA CLA	7650	1, 2	Skip if AC $\neq$ 0, then clear AC.
SMA CLA	7700	1, 2	Skip if AC $<$ 0, then clear AC.
SPA CLA	7710	1, 2	Skip if AC $>$ 0, then clear AC.

Note: EAE microinstructions used on the PDP-8 & PDP-8/I are NOP's to the PDP-8/L.

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		
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		
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.
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.

BASIC IOT MICROINSTRUCTIONS (continued)

Mnemonic	Octal	Operation
Oscilloscope Display Type VC8/I		
DCX	6051	Clear X coordinate buffer.
DXL	6053	Clear and load X coordinate buffer. AC 2-11 = > YB
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.
DSB	6075	Set minimum brightness.
DSB	6076	Set medium brightness.
DSB	6077	Set maximum brightness.
DSB	6074	Zero brightness.
Light Pen Type 370		
DSF	6071	Skip if display flag is a 1.
DCF	6072	Clear the display flag.
Memory Parity Type MP8/L		
SMP	6101	Skip if memory parity error flag = 0.
CMP	6104	Clear memory parity error flag.
Automatic Restart Type KP8/L		
SPL	6102	Skip if power is low.
Memory Extension Control Type MC8/L		
CDF	62N1	Change to data field N. The data field register is loaded with the selected field number (0 or 1). All subsequent memory requests for operands are automatically switched to that data field until the data field number is changed by a new CDF command.
CIF	62N2	Prepare to change to instruction field N. The instruction buffer register is loaded with the selected field number (0 or 1). The next JMP or JMS instruction causes the new field to be entered.
RDF	6214	Read data field into AC 8. Bits 0-5 and 9-11 of the AC are not affected. Bits 6 & 7 are cleared.

BASIC IOT MICROINSTRUCTIONS (continued)

Mnemonic	Octal	Operation
RIF	6224	Same as RDF except reads the instruction field.
RIB	6234	Read interrupt buffer. The instruction field and data field stored during an interrupt are read into AC 8 and 11 respectively.
RMF	6244	Restore memory field. Used to exit from a program interrupt.
Incremental Plotter and Control Type VP8/I		
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.
Random Access Disc File (Type DF32)		
DCMA	6601	Clears memory address register, parity error and completion flags. This instruction clears the disc memory request flag and interrupt flags.
DMAR	6603	The contents of the AC are loaded into the disc memory address register and the AC is cleared. Begin to read information from the disc into the specified core location. Clears parity error and completion flags. Clears interrupt flags.
DMAW	6605	The contents of the AC are loaded into the disc memory address register and the AC is cleared. Begin to write information into the disc from the specified core location. Clears parity error and completion flags.
DCEA	6611	Clears the disc extended address and memory address extension register.
DSAC	6612	Skips next instruction if address confirmed flag is a 1. (AC is cleared.)
DEAL	6615	The disc extended address extension registers are cleared and loaded with the track data held in the AC.
DEAC	6616	Clear the AC then loads the contents of the disc extended address register into the AC to allow program evaluation. Skip next instruction if address confirmed flag is a 1.
DFSE	6621	Skips next instruction if parity error, data request late, or write lock switch flag is a zero. Indicates no errors.
DFSC	6622	Skip next instruction if the completion flag is a 1. Indates data transfer is complete.
DMAC	6626	Clear the AC then loads contents of disc memory address register into the AC to allow program evaluation.

BASIC IOT MICROINSTRUCTIONS (continued)

Mnemonic	Octal	Operation
Card Reader and Control Type CR8/I		
RCSF	6631	Generates an IOP pulse (IOP 1) to test the data-ready flag output. If the data ready flag is 1, the next sequential program instruction is skipped.
RCRA	6632	Generates an IOP pulse (IOP 2) to read the alpha-numeric data at the control-logic buffer register and clear the data ready flag.
RCRB	6634	Generates an IOP pulse (IOP 4) to read the BCD data at the control logic buffer register and clear the data ready flag.
RCSD	6671	Generates an IOP pulse (IOP 1) to test the card-done flag output. If the card done flag is 1, the next sequential program instruction is skipped.
RCSE	6672	Generates an IOP pulse (IOP 2) to advance the card, clear the card done flag, and produce a skip flag if reader is ready. If skip flag is generated, the next sequential program instruction is skipped.
RCRD	6674	Generates an IOP pulse (IOP 4) to clear the card done flag.
General Purpose Converter and Multiplexer Control Type AF01A		
ADSF	6531	Skip if A/D converter flag is a 1.
ADVC	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.

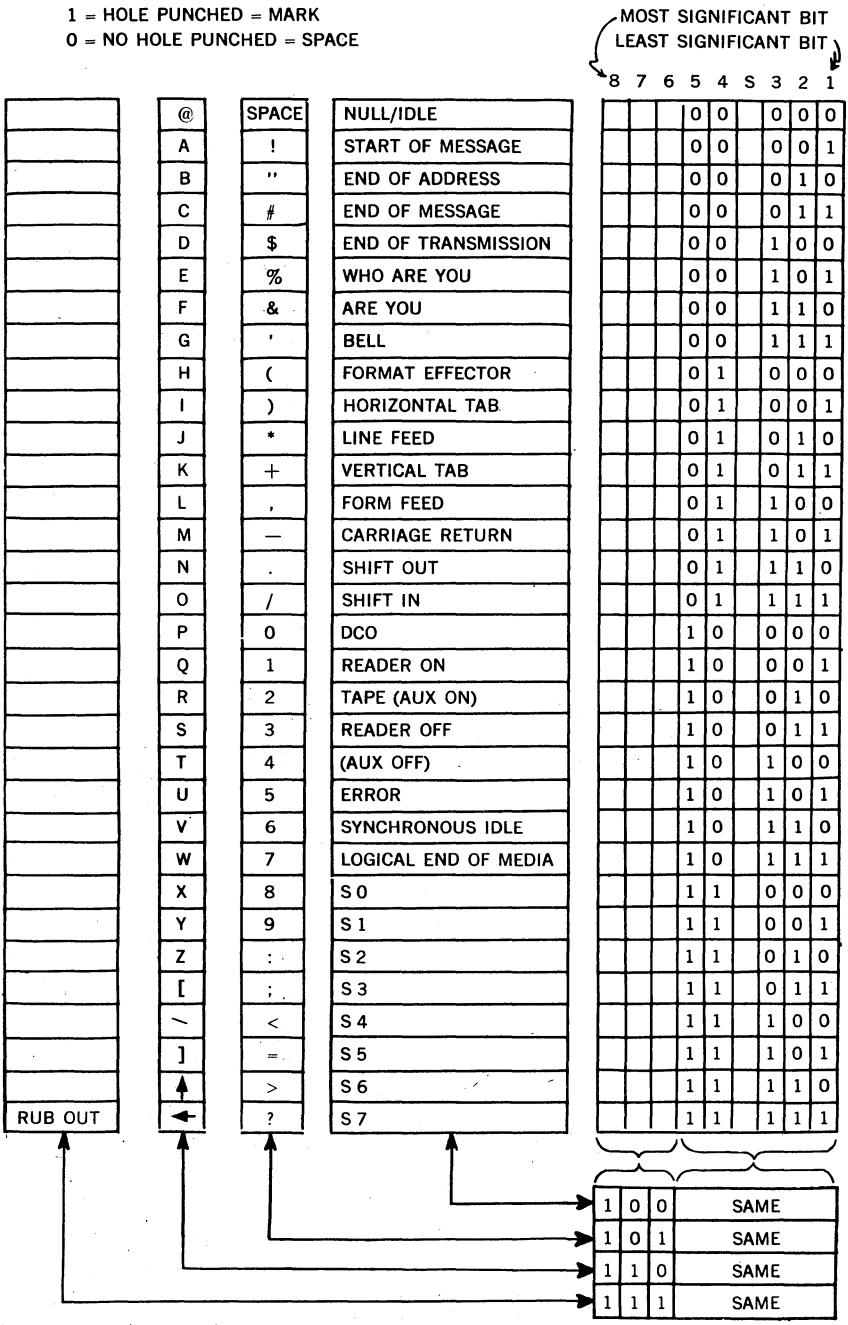
Mnemonic	Octal	Operation
Guarded Scanning Digital Voltmeter Type AF04A		
VSEL	6542	The contents of the accumulator are transferred to the AF04A control register.
VCNV	6541	The contents of the accumulator are transferred to the AF04A channel address register. Analog signal on selected channel is automatically digitized.
VINX	6544	The last channel address is incremented by one and the analog signal on the selected channel is automatically digitized.
VSDR	6561	Skip if data ready flag is a 1.
VRD	6562	Selected byte of voltmeter is transferred to the accumulator and the data ready flag is cleared.
VBA	6564	BYTE ADVANCE command requests next twelve bits, data ready flag is set.
VSCC	6571	SAMPLE CURRENT CHANNEL when required to digitize analog signal on current channel repeatedly.

APPENDIX 3
TABLES OF 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	305	%	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	act-mode	375
8	270	escape	233
9	271		

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



CARD READER CODE

INTERNAL CODE	CARD ZONE	CODE NUM.	IBM 26 CHARACTER	IBM 29 CHARACTER
00	NONE		SPACE	SPACE
01	—	1	1	1
02	—	2	2	2
03	—	3	3	3
04	—	4	4	4
05	—	5	5	5
06	—	6	6	6
07	—	7	7	7
10	—	8	8	8
11	—	9	9	9
12	—	8-2	ASSIGNABLE	
13	—	8-3	#	=
14	—	8-4	@	'
15	—	8-5	ASSIGNABLE	
16	—	8-6	ASSIGNABLE	
17	—	8-7	ASSIGNABLE	
20	0	—	0	0
21	0	1	/	/
22	0	2	S	S
23	0	3	T	T
24	0	4	U	U
25	0	5	V	V
26	0	6	W	W
27	0	7	X	X
30	0	8	Y	Y
31	0	9	Z	Z
32	0	8-2	ASSIGNABLE	
33	0	8-3	,	,
34	0	8-4	%	(
35	0	8-5	ASSIGNABLE	
36	0	8-6	ASSIGNABLE	
37	0	8-7	ASSIGNABLE	
40	11	—	—	—
41	11	1	J	J
42	11	2	K	K
43	11	3	L	L
44	11	4	M	M

INTERNAL CODE	CARD ZONE	CODE NUM.	IBM 26 CHARACTER	IBM 29 CHARACTER
45	11	5	N	N
46	11	6	O	O
47	11	7	P	P
50	11	8	Q	Q
51	11	9	R	R
52	11	8-2	ASSIGNABLE	
53	11	8-3	$	$
54	11	8-4	*	*
55	11	8-5	ASSIGNABLE	
56	11	8-6	ASSIGNABLE	
57	11	8-7	ASSIGNABLE	
60	12	—	&	+
61	12	1	A	A
62	12	2	B	B
63	12	3	C	C
64	12	4	D	D
65	12	5	E	E
66	12	6	F	F
67	12	7	G	G
70	12	8	H	H
71	12	9	I	I
72	12	8-2	ASSIGNABLE	
73	12	8-3	•	•
74	12	8-4	¤	)
75	12	8-5	ASSIGNABLE	
76	12	8-6	ASSIGNABLE	
77	12	8-7	ASSIGNABLE	

WILL NOT DETECT INVALID PUNCH COMBINATIONS

APPENDIX 4

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 of page 37.

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

Tape Channel 8 7 6 5 4 S 3 2 1	Format
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/L diagnostic program tapes are in RIM format.)

The complete PDP-8/L 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 AC0
7764,	7510		SPA	/CHECKING FOR LEADER
7765,	5357		JMP BEG+1	/FOUND LEADER
7766,	7006		RTL	/OK, CHANNEL 7 IN LINK
7767,	6031		KSF	

Absolute Address	Octal Content	Tag	Instruction I Z	Comments
7770,	5367		JMP .−1	
7771,	6034		KRS	/ READ, DO NOT CLEAR
7772,	7420		SNL	/ CHECKING FOR ADDRESS
7773,	3776		DCA I TEMP	/ STORE CONTENT
7774,	3376		DCA TEMP	/ STORE ADDRESS
7775,	5356		JMP BEG	/ NEXT WORD
7776,	0	TEMP,	0	/ TEMP STORAGE
7777,	5XXX		JMP X	/ JMP START OF BIN LOADER

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 DEC-08-LRAA-D 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 more desirable format to use with the 10 cps 33 ASR reader or the Type PR8/L 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:

Tape Channel 8 7 6 5 4 S 3 2 1	Memory Location	Content	Comments
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 DEC-08-LBAA-D for additional information on the Binary Loader program.

APPENDIX 5

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

$10^{\pm n}$ IN OCTAL

10^n	n	10^{-n}	10^n	n	10^{-n}
1	0	1.000 000 000 000 000 000 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 $\log_{10} 2$, n $\log_2 10$ 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

$0 + 0 = 0$, $0 + 1 = 1$, $1 + 0 = 1$, $1 + 1 = 10$

$0 \times 0 = 0$, $0 \times 1 = 1$, $1 \times 0 = 0$, $1 \times 1 = 1$

Octal Scale

+	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

×	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 6

POWERS OF TWO

2^n	n	2^{-n}
1	0	1.0
2	1	0.5
4	2	0.25
8	3	0.125
16	4	0.062 5
32	5	0.031 25
64	6	0.015 625
128	7	0.007 812 5
256	8	0.003 906 25
512	9	0.001 953 125
1 024	10	0.000 976 562 5
2 048	11	0.000 488 281 25
4 096	12	0.000 244 140 625
8 192	13	0.000 122 070 312 5
16 384	14	0.000 061 035 156 25
32 768	15	0.000 030 517 578 125
65 536	16	0.000 015 258 789 062 5
131 072	17	0.000 007 629 394 531 25
262 144	18	0.000 003 814 697 265 625
524 288	19	0.000 001 907 348 632 812 5
1 048 576	20	0.000 000 953 674 316 406 25
2 097 152	21	0.000 000 476 837 158 203 125
4 194 304	22	0.000 000 238 418 579 101 562 5
8 388 608	23	0.000 000 119 209 289 550 781 25
16 777 216	24	0.000 000 059 604 644 775 390 625
33 554 432	25	0.000 000 029 802 322 387 695 312 5
67 108 864	26	0.000 000 014 901 161 193 847 656 25
134 217 728	27	0.000 000 007 450 580 596 923 828 125
268 435 456	28	0.000 000 003 725 290 298 461 914 062 5
536 870 912	29	0.000 000 001 862 645 149 230 957 031 25
1 073 741 824	30	0.000 000 000 931 322 574 615 478 515 625
2 147 483 648	31	0.000 000 000 465 661 287 307 739 257 812 5
4 294 967 296	32	0.000 000 000 232 830 643 653 869 628 906 25
8 589 934 592	33	0.000 000 000 116 415 321 826 934 814 453 125
17 179 869 184	34	0.000 000 000 058 207 660 913 467 407 226 562 5
34 359 738 368	35	0.000 000 000 029 103 830 456 733 703 613 281 25
68 719 476 736	36	0.000 000 000 014 551 915 228 366 851 806 640 625
137 438 953 472	37	0.000 000 000 007 275 957 614 183 425 903 320 312 5
274 877 906 944	38	0.000 000 000 003 637 978 807 091 712 951 660 156 25
549 755 813 888	39	0.000 000 000 001 818 989 403 545 856 475 830 078 125
1 099 511 627 776	40	0.000 000 000 000 909 494 701 772 928 237 915 039 062 5
2 199 023 255 552	41	0.000 000 000 000 454 747 350 886 464 118 957 519 531 25
4 398 046 511 104	42	0.000 000 000 000 227 373 675 443 232 059 478 759 765 625
8 796 093 022 208	43	0.000 000 000 000 113 686 837 721 616 029 739 379 882 812 5
17 592 186 044 416	44	0.000 000 000 000 056 843 418 860 808 014 869 689 941 406 25
35 184 372 088 832	45	0.000 000 000 000 028 421 709 430 404 007 434 844 970 703 125
70 368 744 177 664	46	0.000 000 000 000 014 210 854 715 202 003 717 422 485 351 562 5
140 737 488 355 328	47	0.000 000 000 000 007 105 427 357 601 001 858 711 242 675 781 25
281 474 976 710 656	48	0.000 000 000 000 003 552 713 678 800 500 929 355 621 337 890 625
562 949 953 421 312	49	0.000 000 000 000 001 776 356 839 400 250 464 677 810 668 945 312 5
1 125 899 906 842 624	50	0.000 000 000 000 000 888 178 419 700 125 232 338 905 334 472 656 25
2 251 799 813 685 248	51	0.000 000 000 000 000 444 089 209 850 062 616 169 452 667 236 328 125
4 503 599 627 370 496	52	0.000 000 000 000 000 222 044 604 925 031 308 084 726 333 618 164 062 5
9 007 199 254 740 992	53	0.000 000 000 000 000 111 022 302 462 515 654 042 363 166 809 082 031 25
18 014 398 509 481 984	54	0.000 000 000 000 000 055 511 151 231 257 827 021 181 583 404 541 015 625
36 028 797 018 963 968	55	0.000 000 000 000 000 027 755 575 615 628 913 510 590 791 702 270 507 812 5
72 057 594 037 927 936	56	0.000 000 000 000 000 013 877 787 807 814 456 755 295 395 851 135 253 906 25
144 115 188 075 855 872	57	0.000 000 000 000 000 006 938 893 903 907 228 377 647 697 925 567 626 953 125
288 230 376 151 711 744	58	0.000 000 000 000 000 003 469 446 951 953 614 188 823 848 962 783 813 476 562 5
576 460 752 303 423 488	59	0.000 000 000 000 000 001 734 723 475 976 807 094 411 924 481 391 906 738 281 25
1 152 921 504 606 846 976	60	0.000 000 000 000 000 000 867 361 737 988 403 547 205 962 240 695 953 369 140 625
2 305 843 009 213 693 952	61	0.000 000 000 000 000 000 433 680 868 994 201 773 602 981 120 347 976 684 570 312 5
4 611 686 018 427 387 904	62	0.000 000 000 000 000 000 216 840 434 497 100 886 801 490 560 173 988 342 285 156 25
9 223 372 036 854 775 808	63	0.000 000 000 000 000 000 108 420 217 248 550 443 400 745 280 086 994 171 142 578 125
18 446 744 073 709 551 616	64	0.000 000 000 000 000 000 054 210 108 624 275 221 700 372 640 043 497 085 571 289 062 5
36 893 488 147 419 103 232	65	0.000 000 000 000 000 000 027 105 054 312 137 610 850 186 320 021 748 542 785 644 531 25
73 786 976 294 838 206 464	66	0.000 000 000 000 000 000 013 552 527 156 068 805 425 093 160 010 874 271 392 822 265 625
147 573 952 589 676 412 928	67	0.000 000 000 000 000 000 006 776 263 578 034 402 712 546 580 005 437 135 696 411 132 812 5
295 147 905 179 352 825 856	68	0.000 000 000 000 000 000 003 388 131 789 017 201 356 273 290 002 718 567 848 205 566 406 25
590 295 810 358 705 651 712	69	0.000 000 000 000 000 000 001 694 065 894 508 600 678 136 645 001 359 283 924 102 783 203 125
1 180 591 620 717 411 303 424	70	0.000 000 000 000 000 000 000 847 032 947 254 300 339 068 322 500 679 641 962 051 391 601 562 5
2 361 183 241 434 822 606 848	71	0.000 000 000 000 000 000 000 423 516 473 627 150 169 534 161 250 339 820 981 025 695 800 781 25
4 722 366 482 869 645 213 696	72	0.000 000 000 000 000 000 000 211 758 236 813 575 084 767 080 625 169 910 490 512 847 900 390 625

APPENDIX 7

OCTAL-DECIMAL CONVERSION

OCTAL-DECIMAL INTEGER CONVERSION TABLE

0000 to 0777 (Octal) | 0000 to 0511 (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
0000	0000	0001	0002	0003	0004	0005	0006	0007
0010	0008	0009	0010	0011	0012	0013	0014	0015
0020	0016	0017	0018	0019	0020	0021	0022	0023
0030	0024	0025	0026	0027	0028	0029	0030	0031
0040	0032	0033	0034	0035	0036	0037	0038	0039
0050	0040	0041	0042	0043	0044	0045	0046	0047
0060	0048	0049	0050	0051	0052	0053	0054	0055
0070	0056	0057	0058	0059	0060	0061	0062	0063
0100	0064	0065	0066	0067	0068	0069	0070	0071
0110	0072	0073	0074	0075	0076	0077	0078	0079
0120	0080	0081	0082	0083	0084	0085	0086	0087
0130	0088	0089	0090	0091	0092	0093	0094	0095
0140	0096	0097	0098	0099	0100	0101	0102	0103
0150	0104	0105	0106	0107	0108	0109	0110	0111
0160	0112	0113	0114	0115	0116	0117	0118	0119
0170	0120	0121	0122	0123	0124	0125	0126	0127
0200	0128	0129	0130	0131	0132	0133	0134	0135
0210	0136	0137	0138	0139	0140	0141	0142	0143
0220	0144	0145	0146	0147	0148	0149	0150	0151
0230	0152	0153	0154	0155	0156	0157	0158	0159
0240	0160	0161	0162	0163	0164	0165	0166	0167
0250	0168	0169	0170	0171	0172	0173	0174	0175
0260	0176	0177	0178	0179	0180	0181	0182	0183
0270	0184	0185	0186	0187	0188	0189	0190	0191
0300	0192	0193	0194	0195	0196	0197	0198	0199
0310	0200	0201	0202	0203	0204	0205	0206	0207
0320	0208	0209	0210	0211	0212	0213	0214	0215
0330	0216	0217	0218	0219	0220	0221	0222	0223
0340	0224	0225	0226	0227	0228	0229	0230	0231
0350	0232	0233	0234	0235	0236	0237	0238	0239
0360	0240	0241	0242	0243	0244	0245	0246	0247
0370	0248	0249	0250	0251	0252	0253	0254	0255

	0	1	2	3	4	5	6	7
0400	0256	0257	0258	0259	0260	0261	0262	0263
0410	0264	0265	0266	0267	0268	0269	0270	0271
0420	0272	0273	0274	0275	0276	0277	0278	0279
0430	0280	0281	0282	0283	0284	0285	0286	0287
0440	0288	0289	0290	0291	0292	0293	0294	0295
0450	0296	0297	0298	0299	0300	0301	0302	0303
0460	0304	0305	0306	0307	0308	0309	0310	0311
0470	0312	0313	0314	0315	0316	0317	0318	0319
0500	0320	0321	0322	0323	0324	0325	0326	0327
0510	0328	0329	0330	0331	0332	0333	0334	0335
0520	0336	0337	0338	0339	0340	0341	0342	0343
0530	0344	0345	0346	0347	0348	0349	0350	0351
0540	0352	0353	0354	0355	0356	0357	0358	0359
0550	0360	0361	0362	0363	0364	0365	0366	0367
0560	0368	0369	0370	0371	0372	0373	0374	0375
0570	0376	0377	0378	0379	0380	0381	0382	0383
0600	0384	0385	0386	0387	0388	0389	0390	0391
0610	0392	0393	0394	0395	0396	0397	0398	0399
0620	0400	0401	0402	0403	0404	0405	0406	0407
0630	0408	0409	0410	0411	0412	0413	0414	0415
0640	0416	0417	0418	0419	0420	0421	0422	0423
0650	0424	0425	0426	0427	0428	0429	0430	0431
0660	0432	0433	0434	0435	0436	0437	0438	0439
0670	0440	0441	0442	0443	0444	0445	0446	0447
0700	0448	0449	0450	0451	0452	0453	0454	0455
0710	0456	0457	0458	0459	0460	0461	0462	0463
0720	0464	0465	0466	0467	0468	0469	0470	0471
0730	0472	0473	0474	0475	0476	0477	0478	0479
0740	0480	0481	0482	0483	0484	0485	0486	0487
0750	0488	0489	0490	0491	0492	0493	0494	0495
0760	0496	0497	0498	0499	0500	0501	0502	0503
0770	0504	0505	0506	0507	0508	0509	0510	0511

1000 to 1777 (Octal) | 0512 to 1023 (Decimal)

	0	1	2	3	4	5	6	7
1000	0512	0513	0514	0515	0516	0517	0518	0519
1010	0520	0521	0522	0523	0524	0525	0526	0527
1020	0528	0529	0530	0531	0532	0533	0534	0535
1030	0536	0537	0538	0539	0540	0541	0542	0543
1040	0544	0545	0546	0547	0548	0549	0550	0551
1050	0552	0553	0554	0555	0556	0557	0558	0559
1060	0560	0561	0562	0563	0564	0565	0566	0567
1070	0568	0569	0570	0571	0572	0573	0574	0575
1100	0576	0577	0578	0579	0580	0581	0582	0583
1110	0584	0585	0586	0587	0588	0589	0590	0591
1120	0592	0593	0594	0595	0596	0597	0598	0599
1130	0600	0601	0602	0603	0604	0605	0606	0607
1140	0608	0609	0610	0611	0612	0613	0614	0615
1150	0616	0617	0618	0619	0620	0621	0622	0623
1160	0624	0625	0626	0627	0628	0629	0630	0631
1170	0632	0633	0634	0635	0636	0637	0638	0639
1200	0640	0641	0642	0643	0644	0645	0646	0647
1210	0648	0649	0650	0651	0652	0653	0654	0655
1220	0656	0657	0658	0659	0660	0661	0662	0663
1230	0664	0665	0666	0667	0668	0669	0670	0671
1240	0672	0673	0674	0675	0676	0677	0678	0679
1250	0680	0681	0682	0683	0684	0685	0686	0687
1260	0688	0689	0690	0691	0692	0693	0694	0695
1270	0696	0697	0698	0699	0700	0701	0702	0703
1300	0704	0705	0706	0707	0708	0709	0710	0711
1310	0712	0713	0714	0715	0716	0717	0718	0719
1320	0720	0721	0722	0723	0724	0725	0726	0727
1330	0728	0729	0730	0731	0732	0733	0734	0735
1340	0736	0737	0738	0739	0740	0741	0742	0743
1350	0744	0745	0746	0747	0748	0749	0750	0751
1360	0752	0753	0754	0755	0756	0757	0758	0759
1370	0760	0761	0762	0763	0764	0765	0766	0767

	0	1	2	3	4	5	6	7
1400	0768	0769	0770	0771	0772	0773	0774	0775
1410	0776	0777	0778	0779	0780	0781	0782	0783
1420	0784	0785	0786	0787	0788	0789	0790	0791
1430	0792	0793	0794	0795	0796	0797	0798	0799
1440	0800	0801	0802	0803	0804	0805	0806	0807
1450	0808	0809	0810	0811	0812	0813	0814	0815
1460	0816	0817	0818	0819	0820	0821	0822	0823
1470	0824	0825	0826	0827	0828	0829	0830	0831
1500	0832	0833	0834	0835	0836	0837	0838	0839
1510	0840	0841	0842	0843	0844	0845	0846	0847
1520	0848	0849	0850	0851	0852	0853	0854	0855
1530	0856	0857	0858	0859	0860	0861	0862	0863
1540	0864	0865	0866	0867	0868	0869	0870	0871
1550	0872	0873	0874	0875	0876	0877	0878	0879
1560	0880	0881	0882	0883	0884	0885	0886	0887
1570	0888	0889	0890	0891	0892	0893	0894	0895
1600	0896	0897	0898	0899	0900	0901	0902	0903
1610	0904	0905	0906	0907	0908	0909	0910	0911
1620	0912	0913	0914	0915	0916	0917	0918	0919
1630	0920	0921	0922	0923	0924	0925	0926	0927
1640	0928	0929	0930	0931	0932	0933	0934	0935
1650	0936	0937	0938	0939	0940	0941	0942	0943
1660	0944	0945	0946	0947	0948	0949	0950	0951
1670	0952	0953	0954	0955	0956	0957	0958	0959
1700	0960	0961	0962	0963	0964	0965	0966	0967
1710	0968	0969	0970	0971	0972	0973	0974	0975
1720	0976	0977	0978	0979	0980	0981	0982	0983
1730	0984	0985	0986	0987	0988	0989	0990	0991
1740	0992	0993	0994	0995	0996	0997	0998	0999
1750	1000	1001	1002	1003	1004	1005	1006	1007
1760	1008	1009	1010	1011	1012	1013	1014	1015
1770	1016	1017	1018	1019	1020	1021	1022	1023

OCTAL-DECIMAL INTEGER CONVERSION TABLE (continued)

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

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

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

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

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

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

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

OCTAL-DECIMAL INTEGER CONVERSION TABLE (continued)

4000 to 4777 (Octal) | 2048 to 2559 (Decimal)

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

	0	1	2	3	4	5	6	7
4000	2048	2049	2050	2051	2052	2053	2054	2055
4010	2056	2057	2058	2059	2060	2061	2062	2063
4020	2064	2065	2066	2067	2068	2069	2070	2071
4030	2072	2073	2074	2075	2076	2077	2078	2079
4040	2080	2081	2082	2083	2084	2085	2086	2087
4050	2088	2089	2090	2091	2092	2093	2094	2095
4060	2096	2097	2098	2099	2100	2101	2102	2103
4070	2104	2105	2106	2107	2108	2109	2110	2111
4100	2112	2113	2114	2115	2116	2117	2118	2119
4110	2120	2121	2122	2123	2124	2125	2126	2127
4120	2128	2129	2130	2131	2132	2133	2134	2135
4130	2136	2137	2138	2139	2140	2141	2142	2143
4140	2144	2145	2146	2147	2148	2149	2150	2151
4150	2152	2153	2154	2155	2156	2157	2158	2159
4160	2160	2161	2162	2163	2164	2165	2166	2167
4170	2168	2169	2170	2171	2172	2173	2174	2175
4200	2176	2177	2178	2179	2180	2181	2182	2183
4210	2184	2185	2186	2187	2188	2189	2190	2191
4220	2192	2193	2194	2195	2196	2197	2198	2199
4230	2200	2201	2202	2203	2204	2205	2206	2207
4240	2208	2209	2210	2211	2212	2213	2214	2215
4250	2216	2217	2218	2219	2220	2221	2222	2223
4260	2224	2225	2226	2227	2228	2229	2230	2231
4270	2232	2233	2234	2235	2236	2237	2238	2239
4300	2240	2241	2242	2243	2244	2245	2246	2247
4310	2248	2249	2250	2251	2252	2253	2254	2255
4320	2256	2257	2258	2259	2260	2261	2262	2263
4330	2264	2265	2266	2267	2268	2269	2270	2271
4340	2272	2273	2274	2275	2276	2277	2278	2279
4350	2280	2281	2282	2283	2284	2285	2286	2287
4360	2288	2289	2290	2291	2292	2293	2294	2295
4370	2296	2297	2298	2299	2300	2301	2302	2303

	0	1	2	3	4	5	6	7
4400	2304	2305	2306	2307	2308	2309	2310	2311
4410	2312	2313	2314	2315	2316	2317	2318	2319
4420	2320	2321	2322	2323	2324	2325	2326	2327
4430	2328	2329	2330	2331	2332	2333	2334	2335
4440	2336	2337	2338	2339	2340	2341	2342	2343
4450	2344	2345	2346	2347	2348	2349	2350	2351
4460	2352	2353	2354	2355	2356	2357	2358	2359
4470	2360	2361	2362	2363	2364	2365	2366	2367
4500	2368	2369	2370	2371	2372	2373	2374	2375
4510	2376	2377	2378	2379	2380	2381	2382	2383
4520	2384	2385	2386	2387	2388	2389	2390	2391
4530	2392	2393	2394	2395	2396	2397	2398	2399
4540	2400	2401	2402	2403	2404	2405	2406	2407
4550	2408	2409	2410	2411	2412	2413	2414	2415
4560	2416	2417	2418	2419	2420	2421	2422	2423
4570	2424	2425	2426	2427	2428	2429	2430	2431
4600	2432	2433	2434	2435	2436	2437	2438	2439
4610	2440	2441	2442	2443	2444	2445	2446	2447
4620	2448	2449	2450	2451	2452	2453	2454	2455
4630	2456	2457	2458	2459	2460	2461	2462	2463
4640	2464	2465	2466	2467	2468	2469	2470	2471
4650	2472	2473	2474	2475	2476	2477	2478	2479
4660	2480	2481	2482	2483	2484	2485	2486	2487
4670	2488	2489	2490	2491	2492	2493	2494	2495
4700	2496	2497	2498	2499	2500	2501	2502	2503
4710	2504	2505	2506	2507	2508	2509	2510	2511
4720	2512	2513	2514	2515	2516	2517	2518	2519
4730	2520	2521	2522	2523	2524	2525	2526	2527
4740	2528	2529	2530	2531	2532	2533	2534	2535
4750	2536	2537	2538	2539	2540	2541	2542	2543
4760	2544	2545	2546	2547	2548	2549	2550	2551
4770	2552	2553	2554	2555	2556	2557	2558	2559

5000 to 5777 (Octal) | 2560 to 3071 (Decimal)

	0	1	2	3	4	5	6	7
5000	2560	2561	2562	2563	2564	2565	2566	2567
5010	2568	2569	2570	2571	2572	2573	2574	2575
5020	2576	2577	2578	2579	2580	2581	2582	2583
5030	2584	2585	2586	2587	2588	2589	2590	2591
5040	2592	2593	2594	2595	2596	2597	2598	2599
5050	2600	2601	2602	2603	2604	2605	2606	2607
5060	2608	2609	2610	2611	2612	2613	2614	2615
5070	2616	2617	2618	2619	2620	2621	2622	2623
5100	2624	2625	2626	2627	2628	2629	2630	2631
5110	2632	2633	2634	2635	2636	2637	2638	2639
5120	2640	2641	2642	2643	2644	2645	2646	2647
5130	2648	2649	2650	2651	2652	2653	2654	2655
5140	2656	2657	2658	2659	2660	2661	2662	2663
5150	2664	2665	2666	2667	2668	2669	2670	2671
5160	2672	2673	2674	2675	2676	2677	2678	2679
5170	2680	2681	2682	2683	2684	2685	2686	2687
5200	2688	2689	2690	2691	2692	2693	2694	2695
5210	2696	2697	2698	2699	2700	2701	2702	2703
5220	2704	2705	2706	2707	2708	2709	2710	2711
5230	2712	2713	2714	2715	2716	2717	2718	2719
5240	2720	2721	2722	2723	2724	2725	2726	2727
5250	2728	2729	2730	2731	2732	2733	2734	2735
5260	2736	2737	2738	2739	2740	2741	2742	2743
5270	2744	2745	2746	2747	2748	2749	2750	2751
5300	2752	2753	2754	2755	2756	2757	2758	2759
5310	2760	2761	2762	2763	2764	2765	2766	2767
5320	2768	2769	2770	2771	2772	2773	2774	2775
5330	2776	2777	2778	2779	2780	2781	2782	2783
5340	2784	2785	2786	2787	2788	2789	2790	2791
5350	2792	2793	2794	2795	2796	2797	2798	2799
5360	2800	2801	2802	2803	2804	2805	2806	2807
5370	2808	2809	2810	2811	2812	2813	2814	2815

	0	1	2	3	4	5	6	7
5400	2816	2817	2818	2819	2820	2821	2822	2823
5410	2824	2825	2826	2827	2828	2829	2830	2831
5420	2832	2833	2834	2835	2836	2837	2838	2839
5430	2840	2841	2842	2843	2844	2845	2846	2847
5440	2848	2849	2850	2851	2852	2853	2854	2855
5450	2856	2857	2858	2859	2860	2861	2862	2863
5460	2864	2865	2866	2867	2868	2869	2870	2871
5470	2872	2873	2874	2875	2876	2877	2878	2879
5500	2880	2881	2882	2883	2884	2885	2886	2887
5510	2888	2889	2890	2891	2892	2893	2894	2895
5520	2896	2897	2898	2899	2900	2901	2902	2903
5530	2904	2905	2906	2907	2908	2909	2910	2911
5540	2912	2913	2914	2915	2916	2917	2918	2919
5550	2920	2921	2922	2923	2924	2925	2926	2927
5560	2928	2929	2930	2931	2932	2933	2934	2935
5570	2936	2937	2938	2939	2940	2941	2942	2943
5600	2944	2945	2946	2947	2948	2949	2950	2951
5610	2952	2953	2954	2955	2956	2957	2958	2959
5620	2960	2961	2962	2963	2964	2965	2966	2967
5630	2968	2969	2970	2971	2972	2973	2974	2975
5640	2976	2977	2978	2979	2980	2981	2982	2983
5650	2984	2985	2986	2987	2988	2989	2990	2991
5660	2992	2993	2994	2995	2996	2997	2998	2999
5670	3000	3001	3002	3003	3004	3005	3006	3007
5700	3008	3009	3010	3011	3012	3013	3014	3015
5710	3016	3017	3018	3019	3020	3021	3022	3023
5720	3024	3025	3026	3027	3028	3029	3030	3031
5730	3032	3033	3034	3035	3036	3037	3038	3039
5740	3040	3041	3042	3043	3044	3045	3046	3047
5750	3048	3049	3050	3051	3052	3053	3054	3055
5760	3056	3057	3058	3059	3060	3061	3062	3063
5770	3064	3065	3066	3067	3068	3069	3070	3071

OCTAL-DECIMAL INTEGER CONVERSION TABLE (continued)

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

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

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

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

	0	1	2	3	4	5	6	7
7400	3840	3841	3842	3843	3844	3845	3846	3847
7410	3848	3849	3850	3851	3852	3853	3854	3855
7420	3856	3857	3858	3859	3860	3861	3862	3863
7430	3864	3865	3866	3867	3868	3869	3870	3871
7440	3872	3873	3874	3875	3876	3877	3878	3879
7450	3880	3881	3882	3883	3884	3885	3886	3887
7460	3888	3889	3890	3891	3892	3893	3894	3895
7470	3896	3897	3898	3899	3900	3901	3902	3903
7500	3904	3905	3906	3907	3908	3909	3910	3911
7510	3912	3913	3914	3915	3916	3917	3918	3919
7520	3920	3921	3922	3923	3924	3925	3926	3927
7530	3928	3929	3930	3931	3932	3933	3934	3935
7540	3936	3937	3938	3939	3940	3941	3942	3943
7550	3944	3945	3946	3947	3948	3949	3950	3951
7560	3952	3953	3954	3955	3956	3957	3958	3959
7570	3960	3961	3962	3963	3964	3965	3966	3967
7600	3968	3969	3970	3971	3972	3973	3974	3975
7610	3976	3977	3978	3979	3980	3981	3982	3983
7620	3984	3985	3986	3987	3988	3989	3990	3991
7630	3992	3993	3994	3995	3996	3997	3998	3999
7640	4000	4001	4002	4003	4004	4005	4006	4007
7650	4008	4009	4010	4011	4012	4013	4014	4015
7660	4016	4017	4018	4019	4020	4021	1022	4023
7670	4024	4025	4026	4027	4028	4029	4030	4031
7700	4032	4033	4034	4035	4036	4037	4038	4039
7710	4040	4041	4042	4043	4044	4045	4046	4047
7720	4048	4049	4050	4051	4052	4053	4054	4055
7730	4056	4057	4058	4059	4060	4061	4062	4063
7740	4064	4065	4066	4067	4068	4069	4070	4071
7750	4072	4073	4074	4075	4076	4077	4078	4079
7760	4080	4081	4082	4083	4084	4085	4086	4087
7770	4088	4089	4090	4091	4092	4093	4094	4095

OCTAL-DECIMAL FRACTION CONVERSION TABLE

Octal	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

NOTES

NOTES

digital

WORLD-WIDE SALES AND SERVICE

MAIN OFFICE AND PLANT

DIGITAL EQUIPMENT CORPORATION
146 Main Street, Maynard, Massachusetts 01754
Telephone: From Metropolitan Boston: 646-8600
Elsewhere: (617)-897-5111
TWX: 710-347-0212 Cable: Digital Mayn. Telex: 94-8457

DOMESTIC

NORTHEAST

NORTHEAST OFFICE:
146 Main Street, Maynard, Massachusetts 01754
Telephone: (617)-646-8600 TWX: 710-347-0212

BOSTON OFFICE:
899 Main Street, Cambridge, Massachusetts 02139
Telephone: (617)-491-6130 TWX: 710-320-1167

ROCHESTER OFFICE:
480 Clinton Avenue So., Rochester, New York 14620
Telephone: (716)-454-2000 TWX: 510-253-3078

NEW HAVEN OFFICE:
129 College Street, New Haven, Connecticut 06510
Telephone: (203)-777-5797 TWX: 710-465-0692

MID-ATLANTIC

NEW YORK OFFICE:
Suite #1
71 Grand Avenue, Palisades Park, New Jersey 07650
Telephone: (201)-941-2016 or (212)-594-6955 TWX: 710-992-8974

PARSIPPANY OFFICE:
1259 Route 46, Parsippany, New Jersey 07054
Telephone: (201)-335-3300 or (212)-279-4735 TWX: 710-987-8319

PRINCETON OFFICE:
3 Ninianne Boulevard, Princeton, New Jersey 08540
Telephone: (609)-452-2940 TWX: 510-685-2337

LONG ISLAND
1919 Middle Country Road, Centereach, L.I., New York 11720
Telephone: (516)-585-5410 TWX: 510-228-6505

PHILADELPHIA OFFICE:
1100 West Valley Road, Wayne, Pennsylvania 19087
Telephone (215)-687-1405 TWX: 510-668-4401

WASHINGTON OFFICE:
Executive Building
7100 Baltimore Ave., College Park, Maryland 20740
Telephone: (301)-779-1100 TWX: 710-826-9662

SOUTHEAST

HUNTSVILLE OFFICE:
Suite 41 — Holiday Office Center
3322 Memorial Parkway S.W., Huntsville, Ala 35801
Telephone: (205)-881-7730 TWX: 810-726-2122

COCOA OFFICE:
412 High Point Drive, Cocoa, Florida 32922
Telephone: (305)-632-9283 TWX: 510-957-1448

CENTRAL

PITTSBURGH OFFICE:
400 Penn Center Boulevard, Pittsburgh, Pennsylvania 15235
Telephone: (412)-243-8500 TWX: 710-797-3657

CHICAGO OFFICE:
69 North Broadway, Des Plaines, Illinois 60016
Telephone: 312-299-0144 TWX: 910-233-0894

ANN ARBOR OFFICE:
3853 Research Park Drive, Ann Arbor, Michigan 48104
Telephone: (313)-761-1150 TWX: 810-223-6053

MINNEAPOLIS OFFICE:
Digital Equipment Corporation
15016 Minnetonka Industrial Road
Minnetonka, Minnesota 55343
Telephone: 612-935-1744 TWX: 910-576-2818

CLEVELAND OFFICE:
Park Hill Bldg., 35104 Euclid Ave., Willoughby, Ohio 44094
Telephone: (216)-946-8484

DAYTON OFFICE:
3110 South Kettering Blvd., Dayton, Ohio 45439
Telephone: (513)-299-7377 TWX: 810-459-1676

HOUSTON OFFICE:
3417 Milam Street, Suite A, Houston, Texas 77002
Telephone: (713)-523-2529 TWX: 910-881-1651

WEST

LOS ANGELES OFFICE:
801 E. Ball Road, Anaheim, California 92805
Telephone: (714)-776-6932 or (213)-625-7669 TWX: 910-591-1189

SAN FRANCISCO OFFICE:
560 San Antonio Road, Palo Alto, California 94306
Telephone: (415)-326-5640 TWX: 910-373-1266

SEATTLE OFFICE:
McAusland Building, 10210 N.E. 8th Street
Bellevue, Washington 98004
Telephone: (206)-454-4058 TWX: 910-443-2306

ALBUQUERQUE OFFICE:
6303 Indian School Road, N.E., Alburquerque, N.M. 87110
Telephone: (505)-296-5411 TWX: 910-989-0614

DENVER OFFICE:
2305 South Colorado Blvd., Suite #5, Denver, Colorado 80222
Telephone: 303-757-3332 TWX: 910-931-2650

SALT LAKE CITY OFFICE
431 South 3rd East, Salt Lake City, Utah 84111
Telephone: 801-328-9638

INTERNATIONAL

CANADA

Digital Equipment of Canada, Ltd.
150 Rosamond Street, Carleton Place, Ontario, Canada
Telephone: (613)-257-2615 TWX: 610-561-1651

Digital Equipment of Canada, Ltd.
230 Lakeshore Road East, Port Credit, Ontario, Canada
Telephone: (416)-279-6111 TWX: 610-492-4306

Digital Equipment of Canada, Ltd.
640 Cathcart Street, Suite 205, Montreal, Quebec, Canada
Telephones: (514)-861-6394 TWX: 610-421-3960

Digital Equipment of Canada, Ltd.
5531-103 Street
Edmonton, Alberta, Canada
Telephone: (403)-434-9333 TWX: 610-831-2248

GERMANY

Digital Equipment GmbH
5 Koeln, Neue Weyerstr. 10, West Germany
Telephone: 23 55 01 Telex: 841-888-2269
Telegram: Flip Chip Koeln

Digital Equipment GmbH
8 Muenchen, 2 Theresienstr. 29, West Germany
Telephone: 28 30 53 Telex: 841-24226

ENGLAND

Digital Equipment Co. Ltd.
Arkwright Road, Reading, Berkshire, England
Telephone: Reading 85131 Telex: 851-84327

Digital Equipment Co. Ltd.
13/15 Upper Precinct
Bolton Road,Walkden,Worsley, Manchester, M285AZ England
Telephone: (061) 790 4591

FRANCE

Equipement Digital
22, rue du Champ de l'Alouette, Paris 13e, France
Telephone: 336-0384 Telex: 842-26705

SWEDEN

Digital Equipment Aktiebolag
Vretenvagen 2, Sotna 1, Stockholm, Sweden
Telephone: 98 13 90
Telex: Digital Stockholm 17050
Cable: Digital Stockholm

AUSTRALIA

Digital Equipment Australia Pty. Ltd.
75 Alexander Street, Crows Nest, N.S.W. 2065, Australia.
Telephone: 439-2566 Telex: AA20740
Cable: Digital, Sydney

Digital Equipment Australia Pty. Ltd.
445 St. Kilda Road, Melbourne, Victoria 3004
Telephone: 26 6542 Telex: AA30700

JAPAN

Rikei Trading Co., Ltd. (sales only)
Kozato-Kaikan Bldg.
No. 18-14, Nishishimbashi 1-chome
Minto-ku, Tokyo, Japan
Telephone: 5915246 Telex: 7814208

Digital Equipment Corporation (service only)
c/o Azabu, P.O. Box 23, Tokyo, Japan
Telephone: 431-1554