digital

pdp8/e & pdp8/m small computer handbook

1972

digital equipment corporation

USERS THUMB INDEX

FOREWORD

Minicomputers, from Digital Equipment Corporation, are changing your world—in banks and hospitals, supermarkets and factories. Everywhere people are realizing that computers don't have to be large and expensive to get the job done. A Computer is no longer a multi-million dollar giant that can only survive in spotlessly clean rooms. Minicomputers are going where the job is, because they are rugged, dependable, and inexpensive.

You should know about minicomputers. The PDP-8/E Story shows our computers at work; designing, producing and testing new computers, saving time and money. Other industries, such as oil refineries and automobile manufacturers, are also using the power and speed of computers to produce better products. Minicomputers are not just for big business; hospitals, schools, laboratories and factories are using minis just as effectively. New and old companies are exploring minicomputers.

How large a computer should you buy? Most enterprises begin small, After the computer requirements are completely defined, a decision is then made to either continue with the existing system or to expand. The basic PDP-8/E can be expanded without having to sacrifice your initial investment.

Right now, there are more than 16,000 minicomputers serving in almost every field of endeavor and embracing every discipline known to man. The PDP-8/E and PDP-8/M are DEC's newest models of the PDP-8 family. We invite you to explore the advantages of owning this small machine with big ideas.

INTRODUCTION

This handbook is another in a series intended to familiarize the user with the Digital Equipment Corporation (DEC) PDP-8 family of small general-purpose computers. It explains the newest member of the family, the PDP-8/E Programmed Data Processor and how it is interfaced with the wide variety of peripheral equipment available.

Major topics are: The Programmed Data Processor, PDP-8/E Options, Interface & Installation, Support Services, and Appendices.

Another member of this series of handbooks is titled: An Introduction to Programming. The programming handbook familiarizes the user with the principles of programming the PDP-8 family of general-purpose computers. Together this handbook and the programming handbook describe the complete hardware and software aspects of the PDP-8 family. A newly released handbook called Programming Languages has been added to the handbook series which contains information on the primary languages for the PDP-8 family such as FOCAL, BASIC, PAL III, MACRO, PAL D, SABR, and 4K FORTRAN.

How to Use This Book

This book is neither a text book on computers nor a novel. It contains a wealth of information divided into specific areas of interest. For instance, Chapter 1 defines the basic processor and Chapter 3 defines the basic instructions. These two chapters offer information that allow the reader to compare a PDP-8/E with other processors.

A very important area to the reader who is buying a PDP-8/E is the extent of available, useable programs. Chapter 4 describes many of our commonly used programs, ranging from loaders to complete operating systems. Many more programs are further defined in Appendix A. However, these programs represent only a small fraction of the more than 1000 operating programs available to the PDP-8/E user.

Chapter 2 provides operating instructions. Because the operation of the processor requires similiar steps, learning to operate the processor can be compared with learning to drive a car.

To add additional capability (such as peripherals) Chapter 7 defines the complete line of options. Thousands of these options are presently operating in customer facilities.

For the customer requiring a special application, Chapters 9 and 10 illustrate in detail how a customer can design an interface to allow the PDP-8/E or the PDP-8/M to control his particular application.

The PDP-8/E Computer requires operating and programming skills.

Chapter 12 explains our courses of instruction designed to qualify customers in the areas of operation, maintenance, and programming. Chapter 12 also defines other various types of services including maintenance, depot level repair, software support, application support, etc.

THE COMPANY

In a little over thirteen years, Digital has become a major force in the electronics industry. The company has grown from three employees and 8,500 square feet of production floor space in a converted woolen mill in Maynard, Massachusetts, to an international corporation employing more than 6,000 people with well over two million square feet of floor space in more than 60 manufacturing, sales, and service facilities around the world. In addition to the corporate headquarters in Maynard, Massachusetts, other manufacturing facilities are located in Westfield and Westminster, Massachusetts. Internationally and outside the continental United States, Digital has manufacturing plants in England, Canada and Puerto Rico.

From its beginnings as a manufacturer of digital modules, the company has now grown to the point where it is the world's largest manufacturing supplier of logic modules and the third largest computer-manufacturer, by number of installations, in the industry. Digital's rise as a leader in the electronics industry began in 1957 with the introduction of the company's line of electronic circuit modules. These solid-state modules were used to build and test other manufacturers' computers. Two years later, Digital introduced its first computer, the PDP-1. The PDP-1 heralded a new concept for the industry—the small, on-line computer. And the PDP-1 was inexpensive—it sold for $120,000 while competitive machines with similar capabilities were selling at over $1 million. But the PDP-1 was more than a data processor; more than just a tool to manipulate data. It was a system that could be connected to all types of instrumentation and equipment for on-line, real-time monitoring control, and analysis. It was a system with which people and machines could interact.

Also, in 1958, Digital introduced the Systems Modules, high-quality, low-cost, solid-state, digital logic circuits on a single printed circuit card.

Today, electronic modules like the ones Digital introduced are used in most electronic equipment, from computers to television sets.

In 1965, Digital announced the first of the FLIP CHIP® module lines. These highly reliable modules include cards for internal computer logic, interfacing, control and analog-to-digital conversion.

In 1963, Digital Equipment Corporation introduced the PDP-5 computer, predecessor of the PDP-8 series. This was followed by the first PDP-8/I, and PDP-8/L. Over this seven year period, considerable improvement has been made, many options have been developed, over 60 peripherals and a variety of programs developed. As each new application need arises, Digital Equipment engineering responds with new equipment; each time further increasing the capability of the PDP-8 Family and making available a wider range of equipment.

Throughout the life span of the PDP-8 Family, DEC has developed more than 1,000 programs for a wide variety of applications. New programs are constantly in development by Digital's Programming Department and the PDP-8 Users. This means that each PDP-8/E user will have a wide variety of programs immediately available to him.

To further enhance the user's capability, the DECUS library contains a wide variety of programs developed by the PDP-8 users. This library is operated by DEC exclusively for customer use. Programs are available for as little as $1.00 each.

The PDP-8/E is designed for the inexperienced as well as the most sophisticated user. Digital Equipment Corporation provides training as well as maintenance.

PDP-8/E FEATURES

Digital's all new PDP-8/E is the most powerful, most expandable and most versatile 12-bit computer available today. Its low price and high performance makes it the ideal system for a variety of uses, extending all the way from minimal control units to fully expanded general purpose systems. It is fast, compact and easy to interface.

PDP-8/E offers features such as a unique internal bus system called OMNIBUS™, which allows the user to plug memory and processor options into any available slot location: the availability of 256 words of Read-Only or Read/Write memory; a 1.2 microsecond memory cycle time; the use of TTL integrated circuitry with MSI technology; expansion to 32,768 12-bit words of core storage; low-cost mass storage expansion with DECdisk or DECtape; and a space and money saving packaging design.

PDP-8/E Features at a Glance:

- A unique internal bus design called OMNIBUS which eliminates the need for back panel wiring. Processor options can be inserted in any available slot.
- Increased speed-memory cycle time of 1.2 microseconds.
- A new packaging scheme which makes PDP-8/E physically smaller than its predecessor, the PDP-8/I. And, with no predetermined locations needed for options, there is no wasted space in the logic panel.
- A full line of over 60 options and peripherals immediately available.
- More than 1000 programs immediately available to the user.
- Availability of 256 word increments of Read-Only memory and/or Read/Write memory.
- A Standard General Purpose register in the basic machine which becomes the MQ register when the EAE option is implemented.
- Six additional Processor IOT instructions which make flag manipulation and interrogation faster and easier.

- A six bit byte swap instruction allowing faster and more convenient character handling.
- TTL integrated circuit modules utilizing MSI technology.
- Over 11,000 compatible PDP-8 Family computers in use for sharing programs through Digital's users group, DECUS.
- Low-cost core memory expansion to 32,768 words and low-cost mass storage expansion with DECdisk, DECtape and IBM-compatible magnetic tape.
- Hardware Bootstrap Loader option.
- Provision for multiple (up to 17 total) teletypes.
- Worldwide, dependable service.
- Program and maintenance training included.
- Fully parallel processor.
- Link feature to facilitate multiple precision arithmetic.
- Full range of turnkey and applications-oriented systems available.
- Over seven years of software development by Digital.
- Expanded hardware multiply/divide.
- Eight auto-index registers.
- **FO**rmula **CA**lculator **L**anguage (FOCAL)
- **DI**gital Equipment Corporation **B**usiness **O**riented **L**anguage (DIBOL)
- FORTRAN
- BASIC
- Assemblers
- Editors
- Debugging Aids
- Operating Systems

CONTENTS

CHAPTER 2 STANDARD SYSTEM OPERATION

CHAPTER 3 MEMORY AND PROCESSOR INSTRUCTIONS

CHAPTER 6 DATA BREAK

PART II PDP-8/E OPTIONS

CHAPTER 7 PDP-8/E OPTIONS

PART III INTERFACING & INSTALLATION

CHAPTER 8 DIGITAL LOGIC CIRCUITS

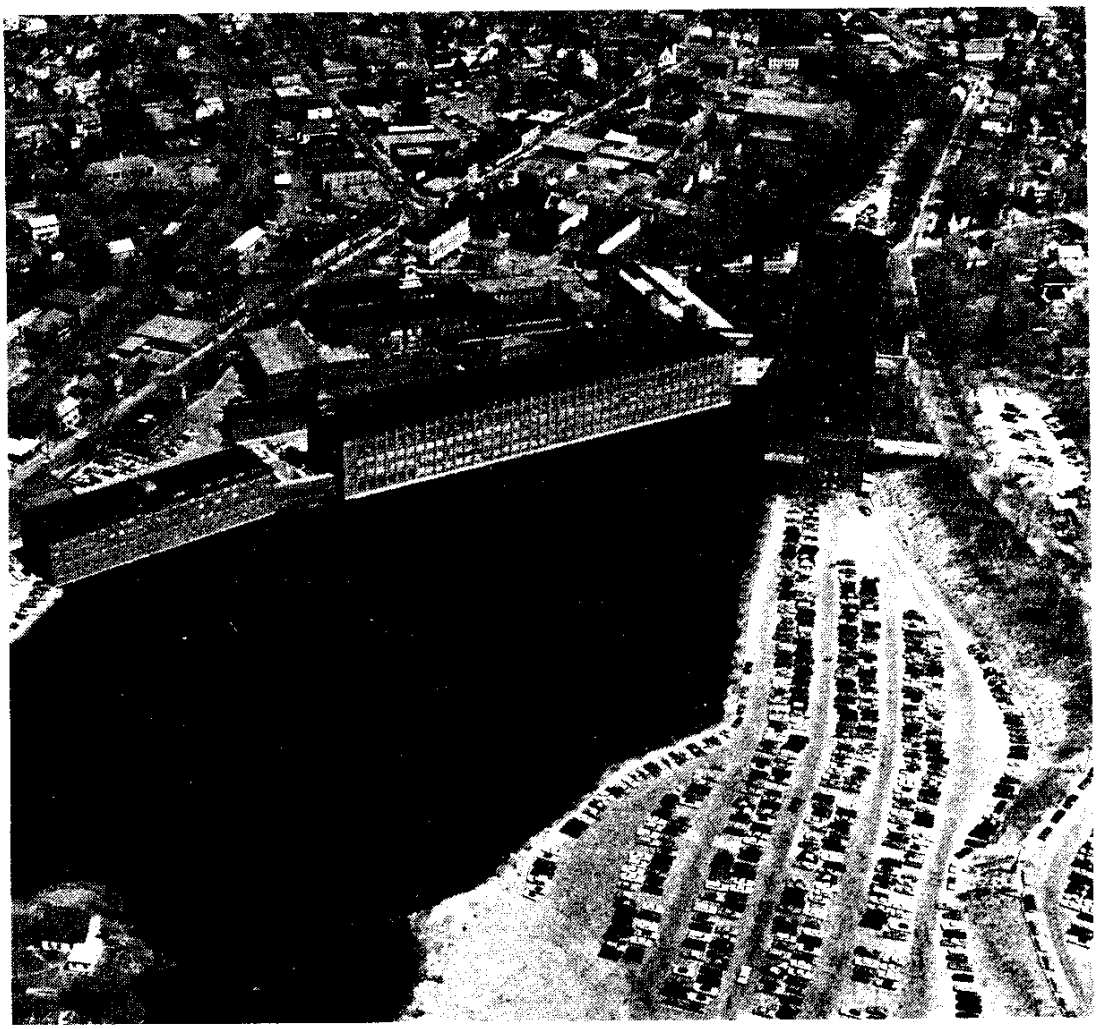
CHAPTER 9 THE OMNIBUS INTERFACING

APPENDIX A PROGRAM ABSTRACTS

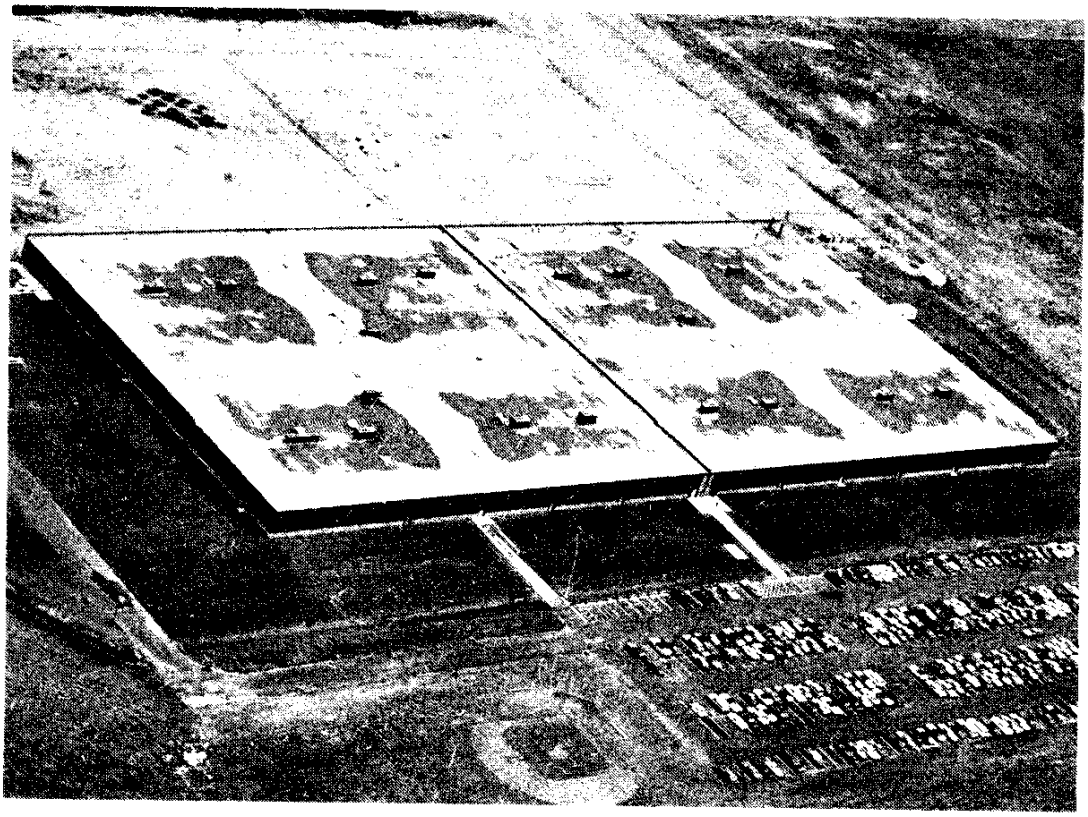
THE PDP-8/E STORY

The PDP-8/E story is a guided tour, using pictures and descriptions, of Digital Equipment Corporation. We want you to see the skilled people, the manufacturing processes, the scores of test stations, and the wide variety of DEC products—all of which contribute to produce the finest, most cost effective computers and related products on the market.

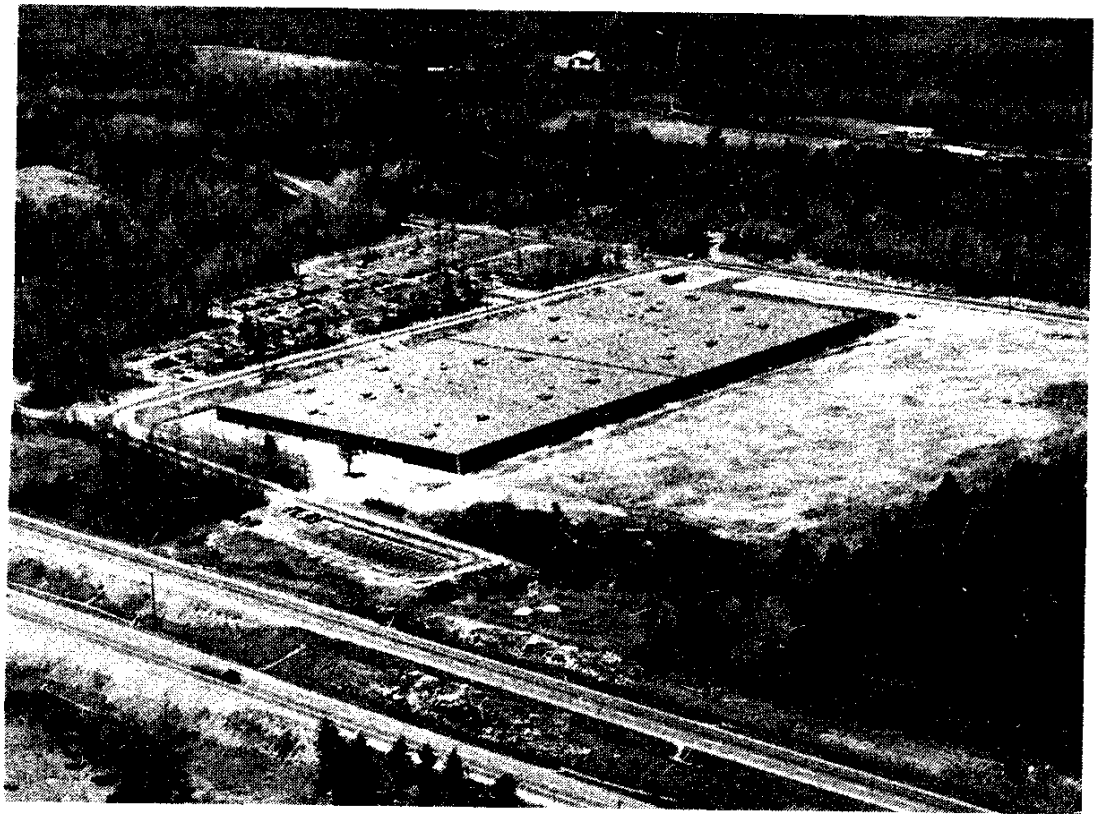
At our production facilities in Maynard, Westminster, and Westfield, Massachusetts; Carlton Place, Ontario, Canada; San German, Puerto Rico; and Galway Bay, Ireland; computers are used extensively to design, produce and qualify new computers. After each computer has passed all of its qualifying tests and is accepted by DEC's quality control and field service groups, it is shipped with the full assurance and guarantee that the computer will provide the outstanding performance and dependability that our customers expect.



The home office and main manufacturing facilities for DEC, the third largest computer manufacturer in the world, are located in this mill complex in Maynard, Massachusetts. We have 1,000,000 square feet here, about 100 times more than when the company started producing digital modules 14 years ago.

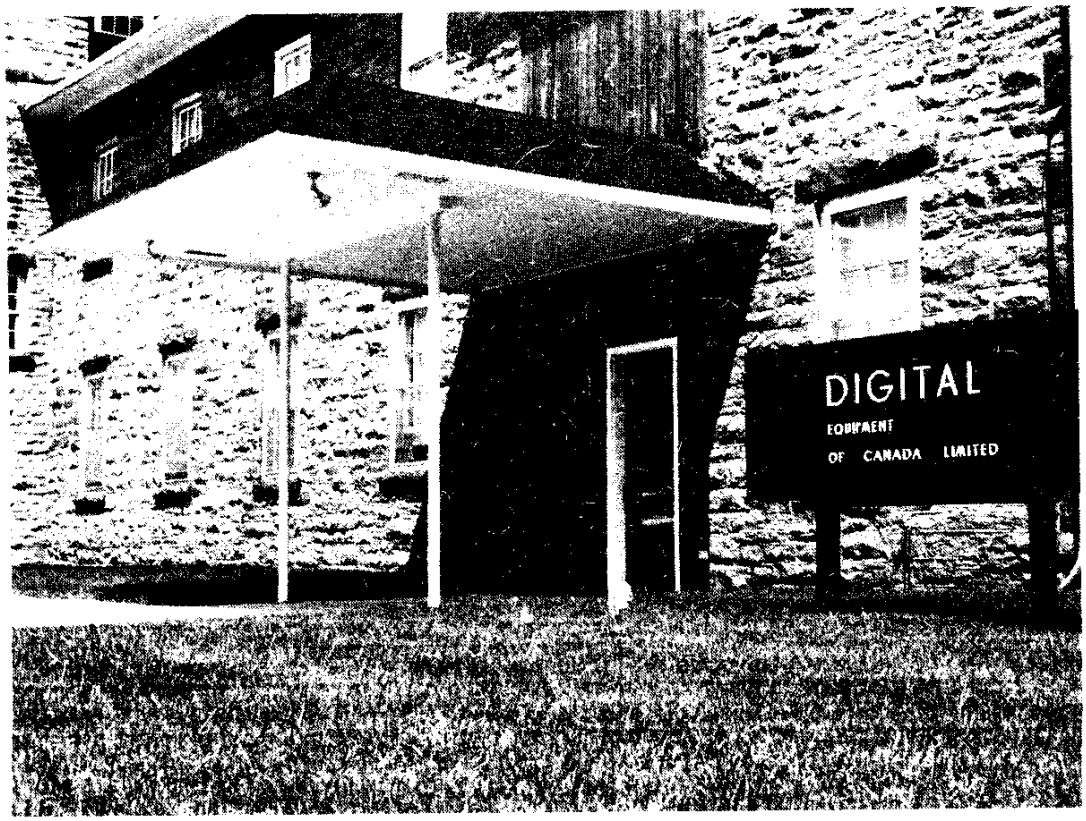


Westfield Plant — Westfield, Massachusetts, U.S.A.

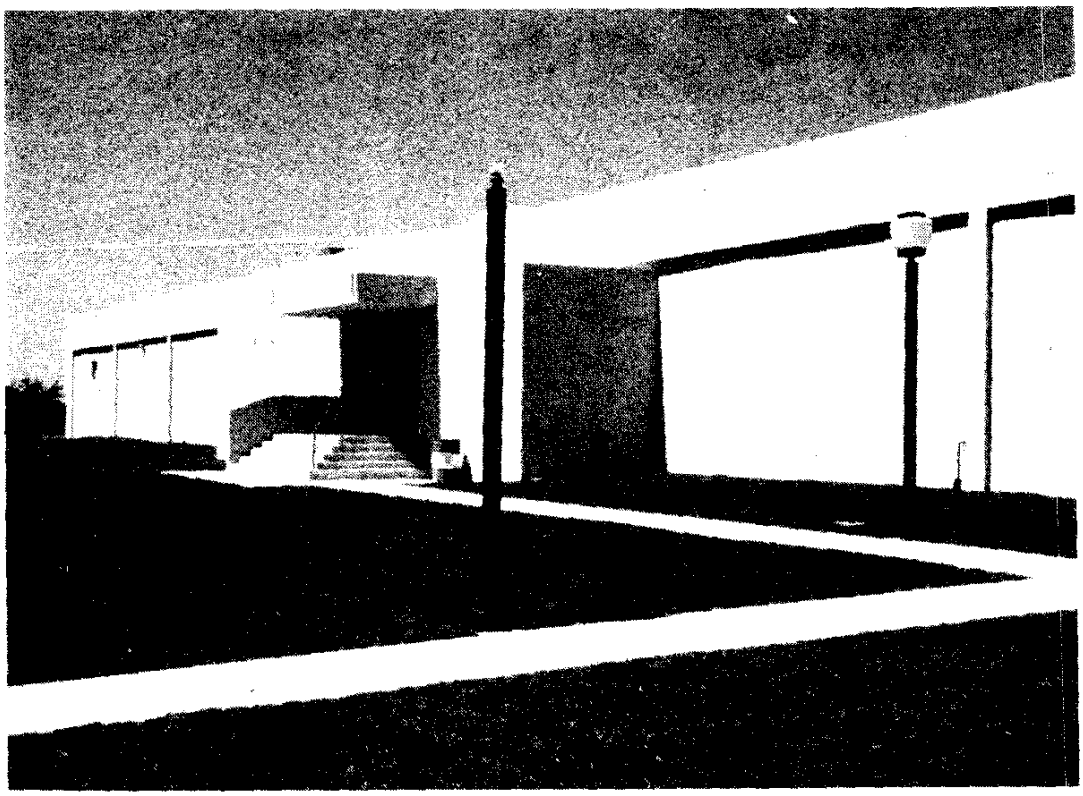


Westminster Plant — Westminster, Massachusetts, U.S.A.

Mountainview, California

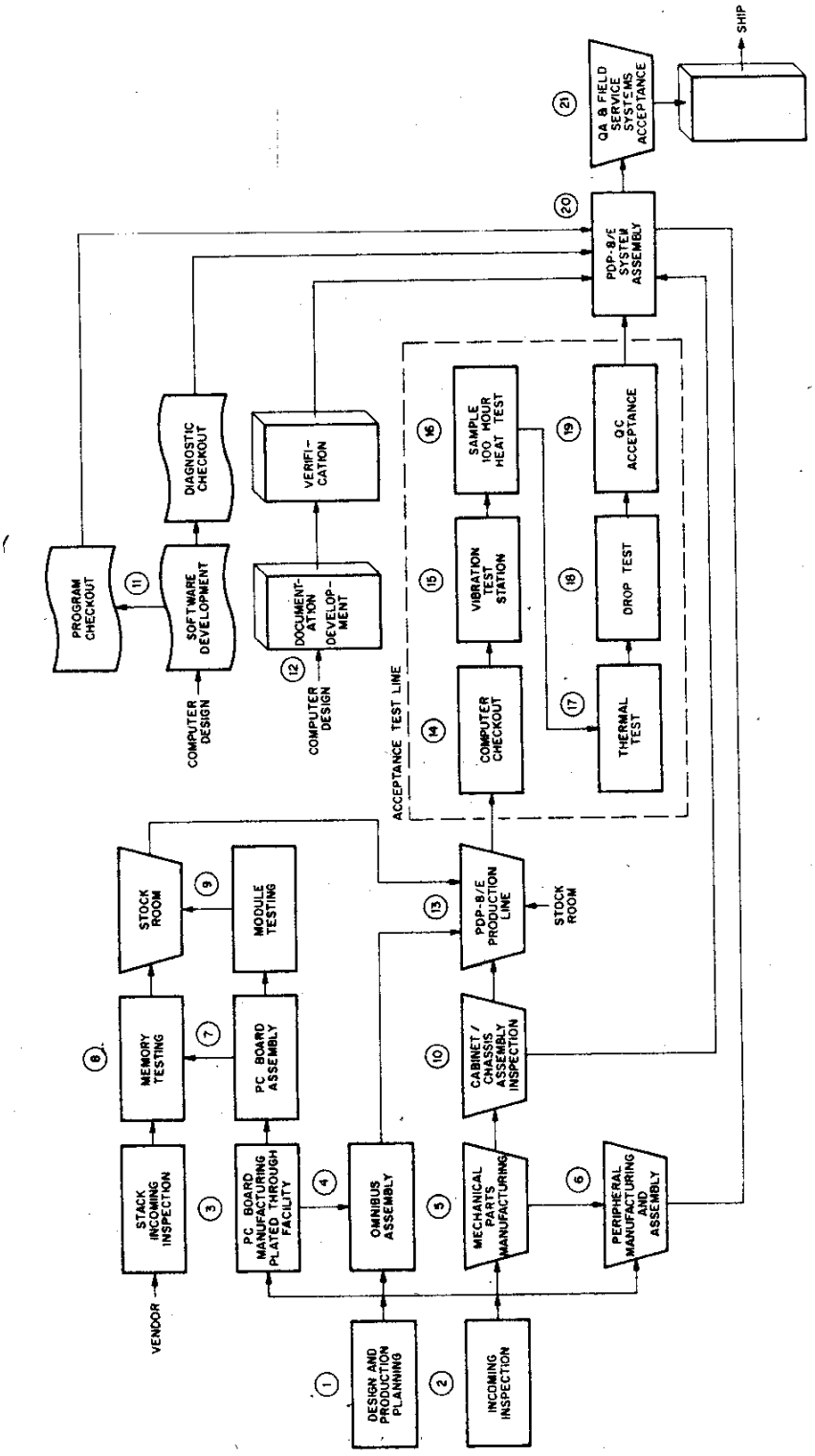


Carlton Place Plant — Carlton Place, Ontario, Canada



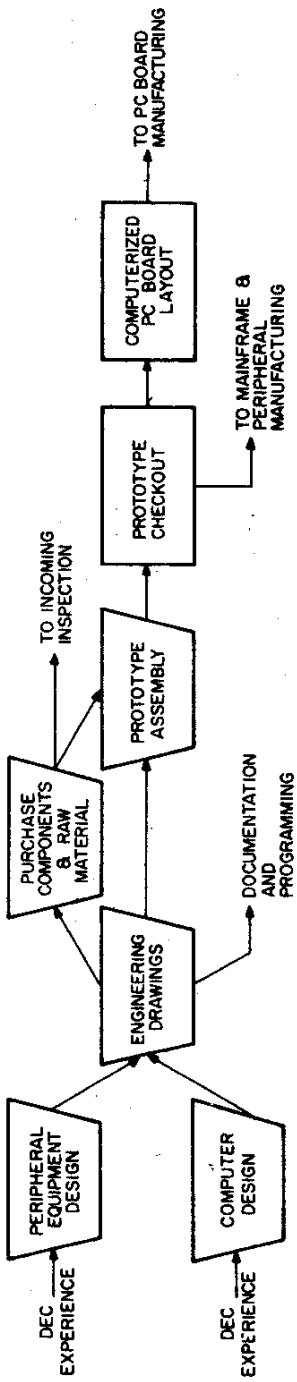
Galway Bay Plant — Galway Bay, Ireland

Puerto Rico Plant — San German, Puerto Rico



THE STORY

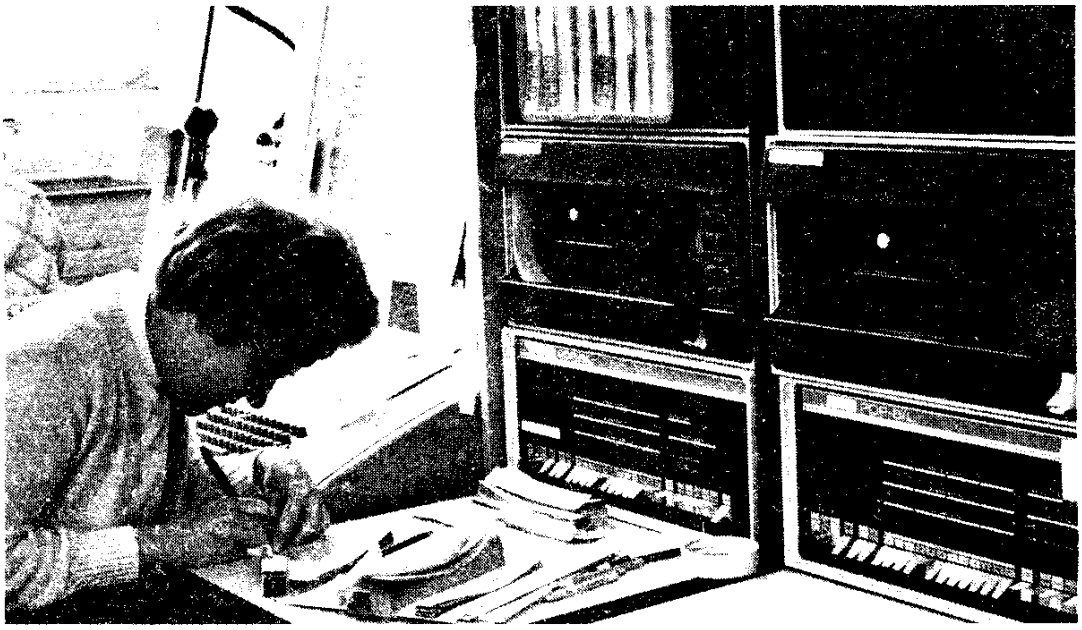
A diagram is provided for you to relate the photo-story to the actual events that occur here every day at DEC. It begins with design, evolves to incoming inspection of components and raw materials, then progresses to a finished computer system ready for shipment to a customer. The numbers in each block refer to the part of the story about that particular process or test.



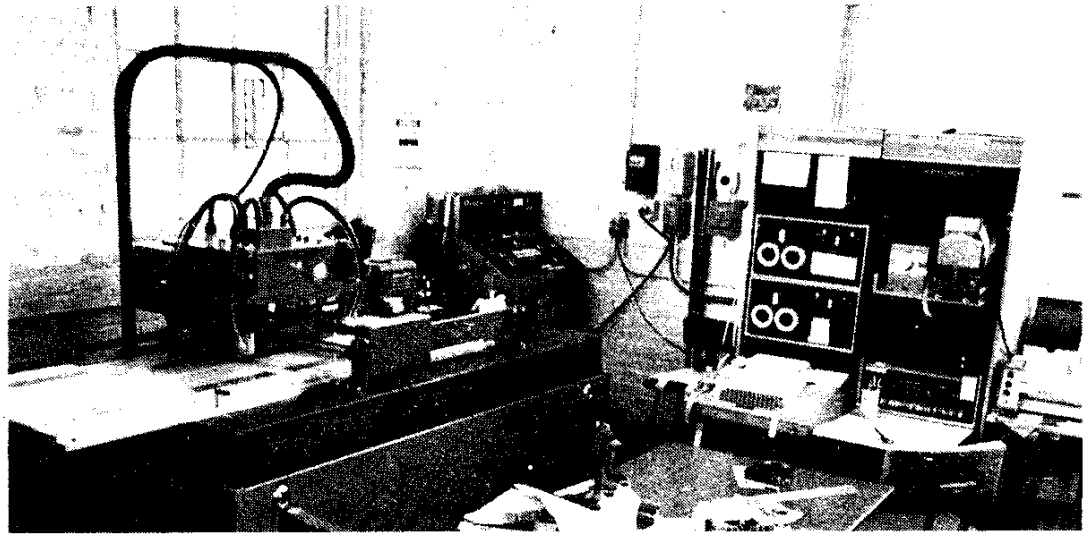
(1) Design and Production Planning

The experience that DEC has acquired from many years of computer and peripheral manufacturing goes into DEC's newest computers and peripherals. No equipment is manufactured until the prototype has undergone full evaluation by engineering, quality assurance, and field service.

After evaluation, production planning begins. New test stations to accommodate high volume testing are designed and produced. DEC's programming department immediately goes to work on new programs for all computerized testing.

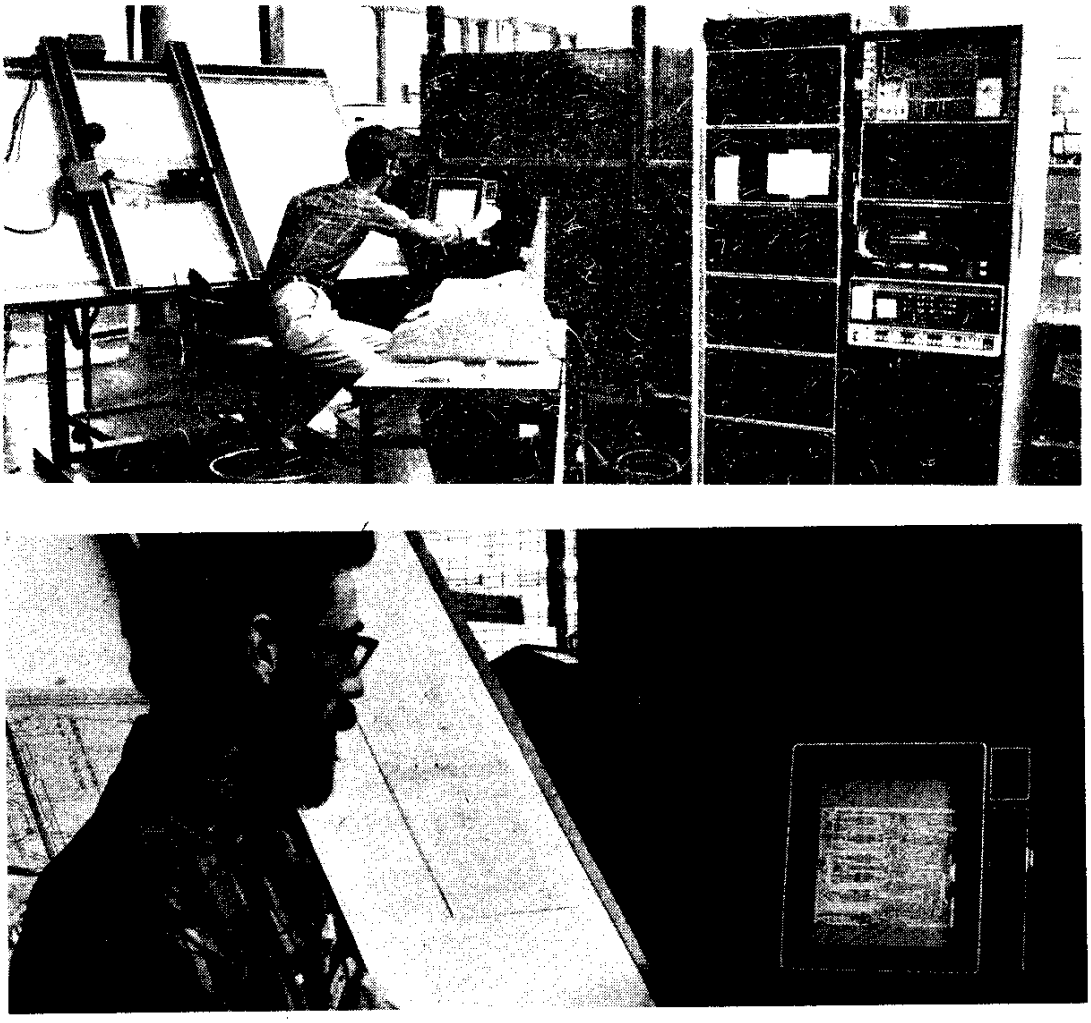


DEC uses a PDP-8 Computer with a Digitizer to prepare for highly accurate automated drilling operations on PDP-8 logic module boards. Drilling coordinates are retrieved from a layout of a module (shown on the drating table). The information is stored in core memory, and the computer generates a paper tape that contains digitized information about the location of the holes to be drilled in the module boards.



The paper tape containing the digitized information is then taken to another PDP-8 computer for post processing to produce another paper tape with all of the various control signals to run the drilling machine. Thus, a PDP-8 Computer is actively involved in producing new PDP-8 Computers.

The X- and Y-coordinate information is first plotted out on an automatic plotter to check its accuracy and then post processed in the larger PDP-8 Computer. Next comes a test run on the drilling machine to see the results.



DEC uses computers to design more computers.

The PC board layout system (using a PDP-8 Computer, a KV Graphics System, and a Digitizer) is another example of computers being used to design more computers.

The computer is used to design and lay out each circuit board and obtain drilling coordinates.

The system provides the layout of a PC board from hand-drawn sketches by inputting X- and Y-coordinate information into the computer in digitized form. When the operator wants a connector to be placed at a particular location, he locates the digitizer cursor at the starting point and commands the computer via the Teletype®. The appropriate connector appears on the graphics display.

This information, in digitized form, is available to lay out the PC board, drill holes for the various components, operate the computerized component insertion machine, and other specialized functions. With this system, DEC is able to computerize a large part of the process of laying out and producing printed circuit boards.

® Teletype is a registered trademark of Teletype Corporation.

QUALITY

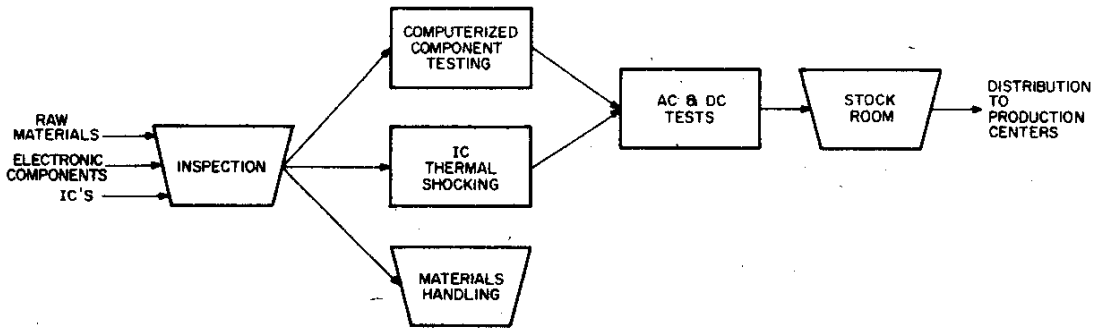
We have built more than 12,000 computers. Naturally, at DEC computerized testing techniques play a major role. Dynamic testing controlled by computers begins as each component is received at the plant and continues through most all production phases. As the major components of the computer progress through the assembly lines, the testing becomes more and more complex, and culminates with the final acceptance test of the finished system. Before a unit progresses to the next assembly or test station, it must meet the rigid standards imposed by DEC.

Computerized testing is ideal for quality control. Many similar tests are continually being run. By automating the tests, all results are calculated the same way and printed out in a standard format, thereby increasing test reliability and accuracy. The cost of quality control tests is drastically reduced by cutting manhours required for other test methods. The computer can control the tests, as well as acquire data and calculate results, and the system is flexible enough to make real-time "decisions" as the test progresses.

The advantages of using small computers during design, production, and testing are mainly economical. Small computers are inexpensive and can be located in the shop, right where the action is.

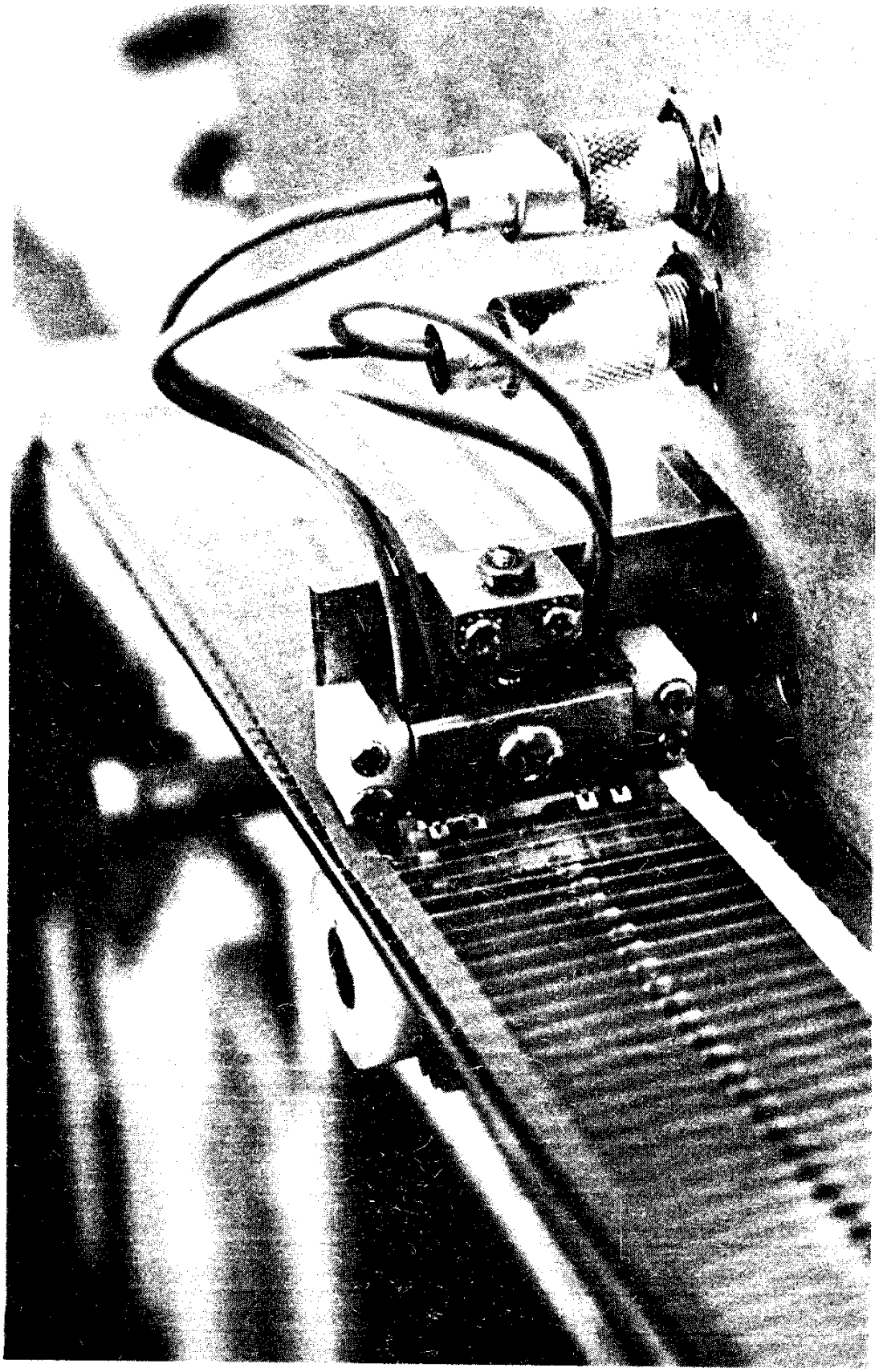
In manufacturing, computers provide on-the-spot testing. When repeatability of testing is important, computers make certain that all components meet the required standards. A computer can do the same task identically again and again; human variation in performing tests is virtually eliminated. The result is a test that is identical for each component being tested.

A written record of test results is often necessary. In computerized testing, the record is available the instant the test results are available. This is particularly important, especially on an assembly line where the unit must be qualified at one station before moving on to the next station. The test operator can press a button and instantly receive a printout of test results.

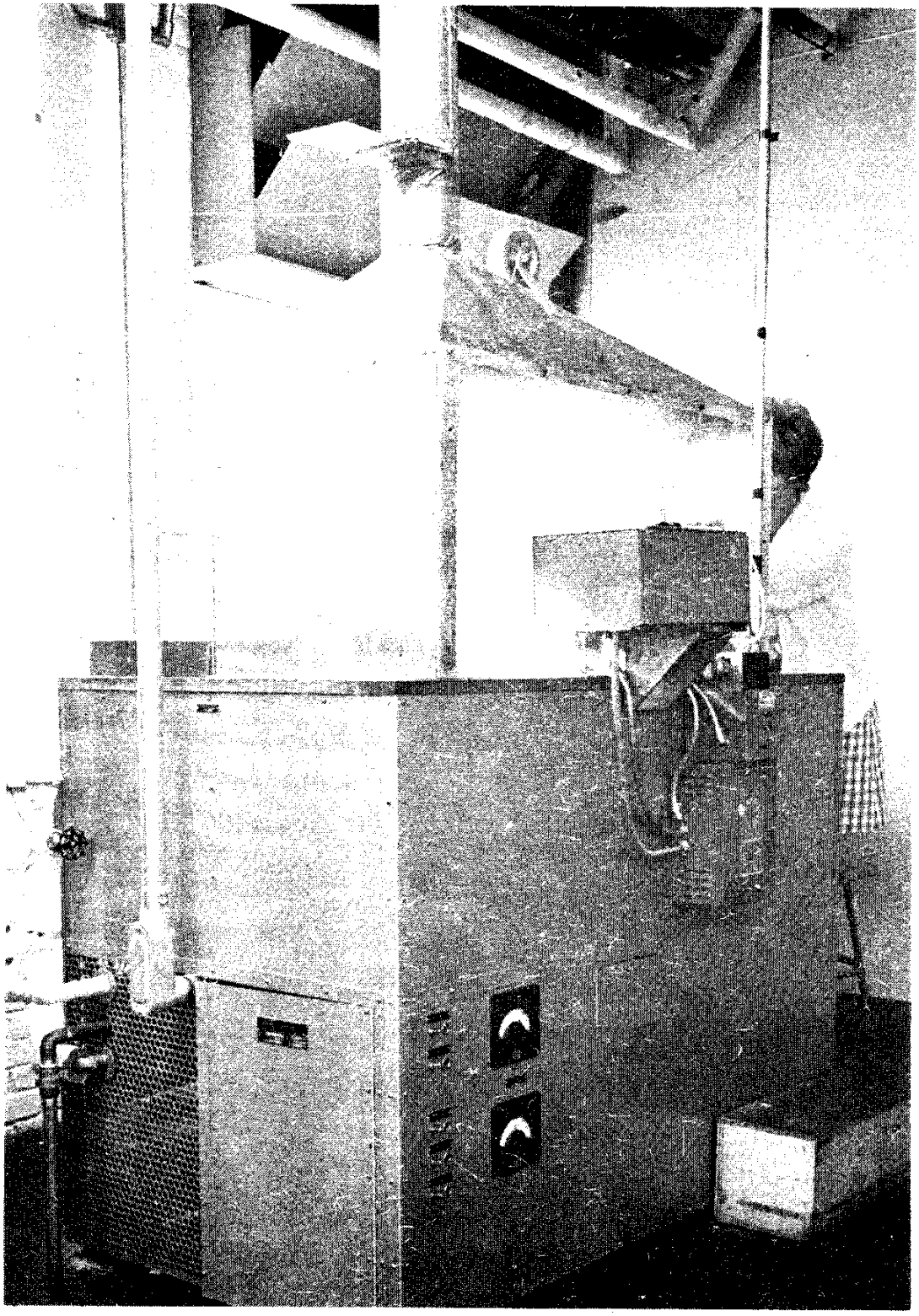


(2) *Incoming Inspection*

Inspection, testing and more testing, right from the beginning, is a major factor in the PDP-8 family success story. All material, components, and integrated circuits (ICs) must pass rigorous inspections before being placed in DEC's stock room.

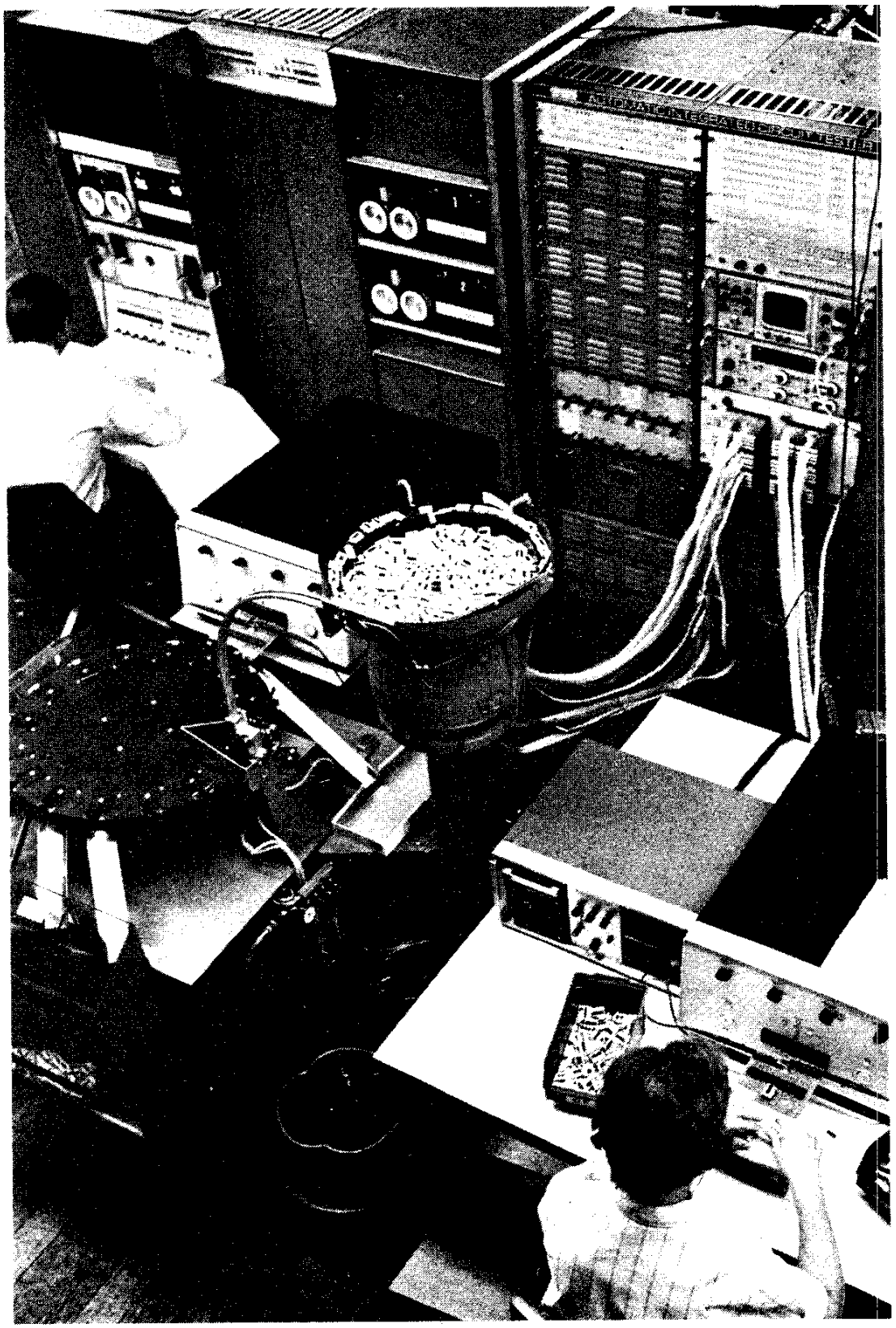


All incoming components are 100% tested. Here, diodes are being tested automatically.



All incoming IC's are tested—The IC Thermal Shocking Process

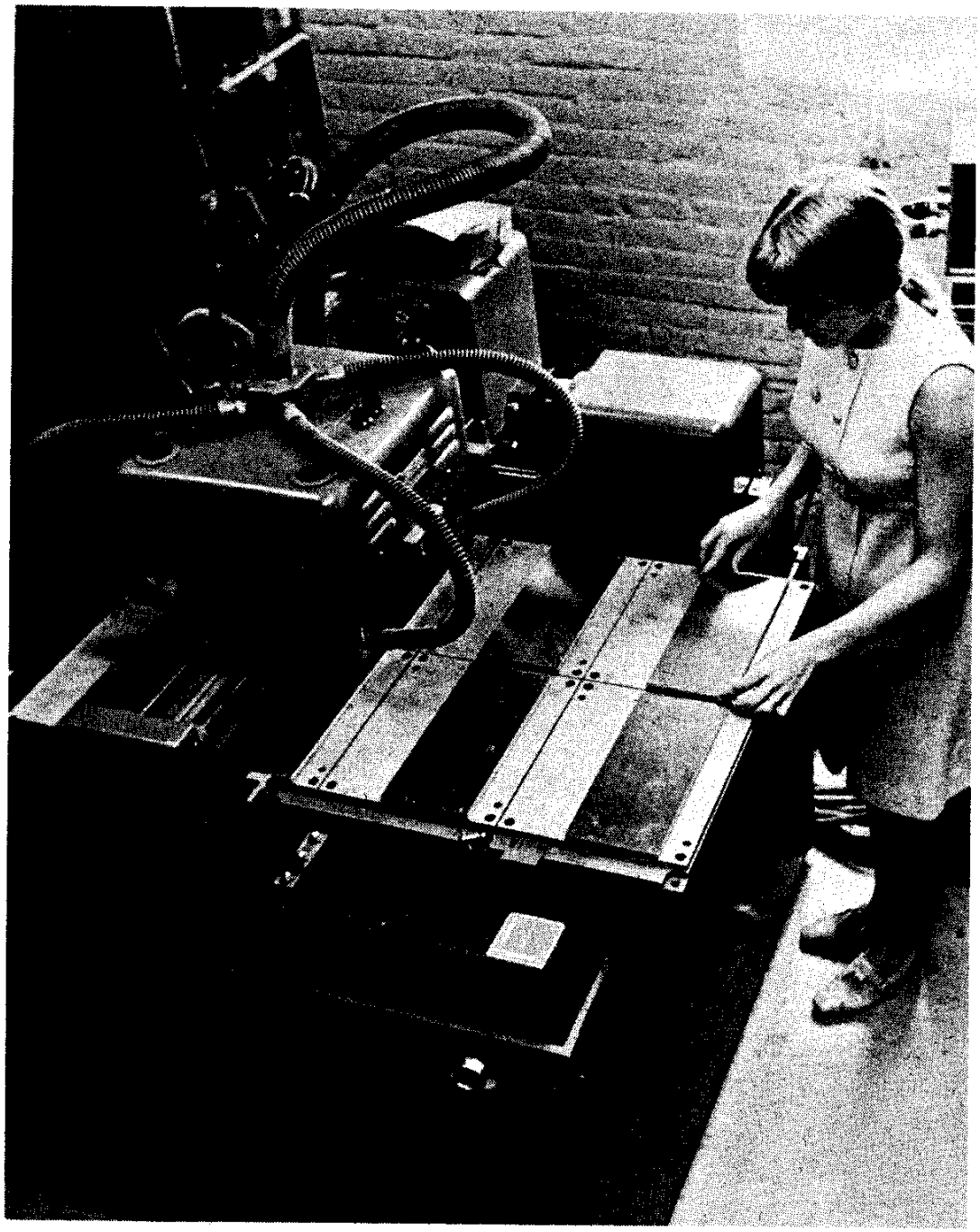
IC's are first given a cold test by placing the IC's into a bath at 32°F for 2 minutes. The IC's are then cycled into another chamber at a temperature of 212° F to force any possible fault to appear. Then testing for faults begins.



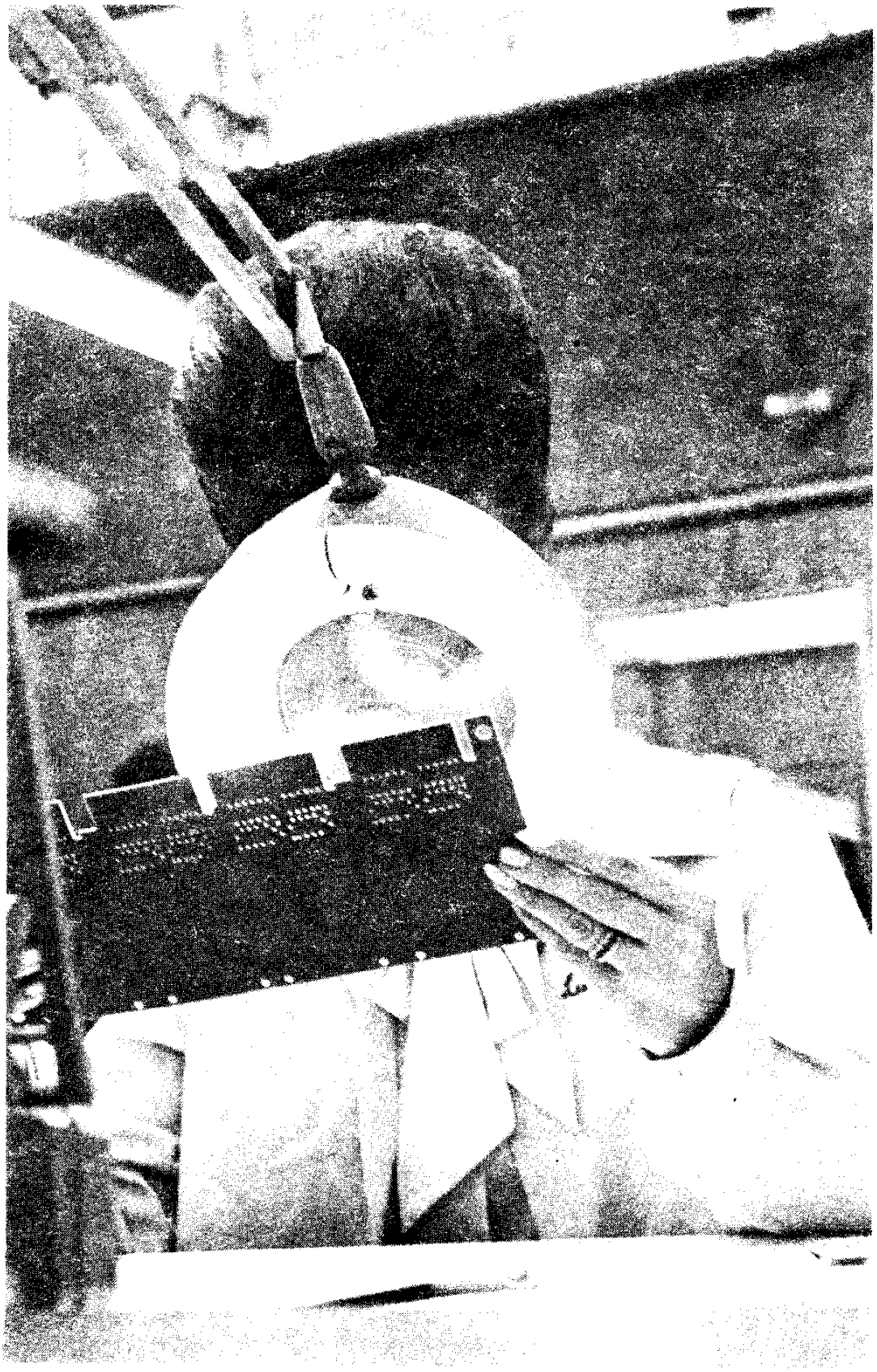
All incoming integrated circuits under computer controlled testing, with 40 dc and 16 ac tests performed in 1.1 seconds. This 100% inspection speeds production by minimizing the diagnosis of component failures in module test.

(3) PC Board Manufacturing Plated Through Facility

The manufacturing of printed circuit (PC) boards requires a facility that provides a controlled process and rigorous quality control. DEC is a world leader in the manufacturing of logic modules. We produce more than 3,000,000 modules per year and have been producing logic modules since 1957.

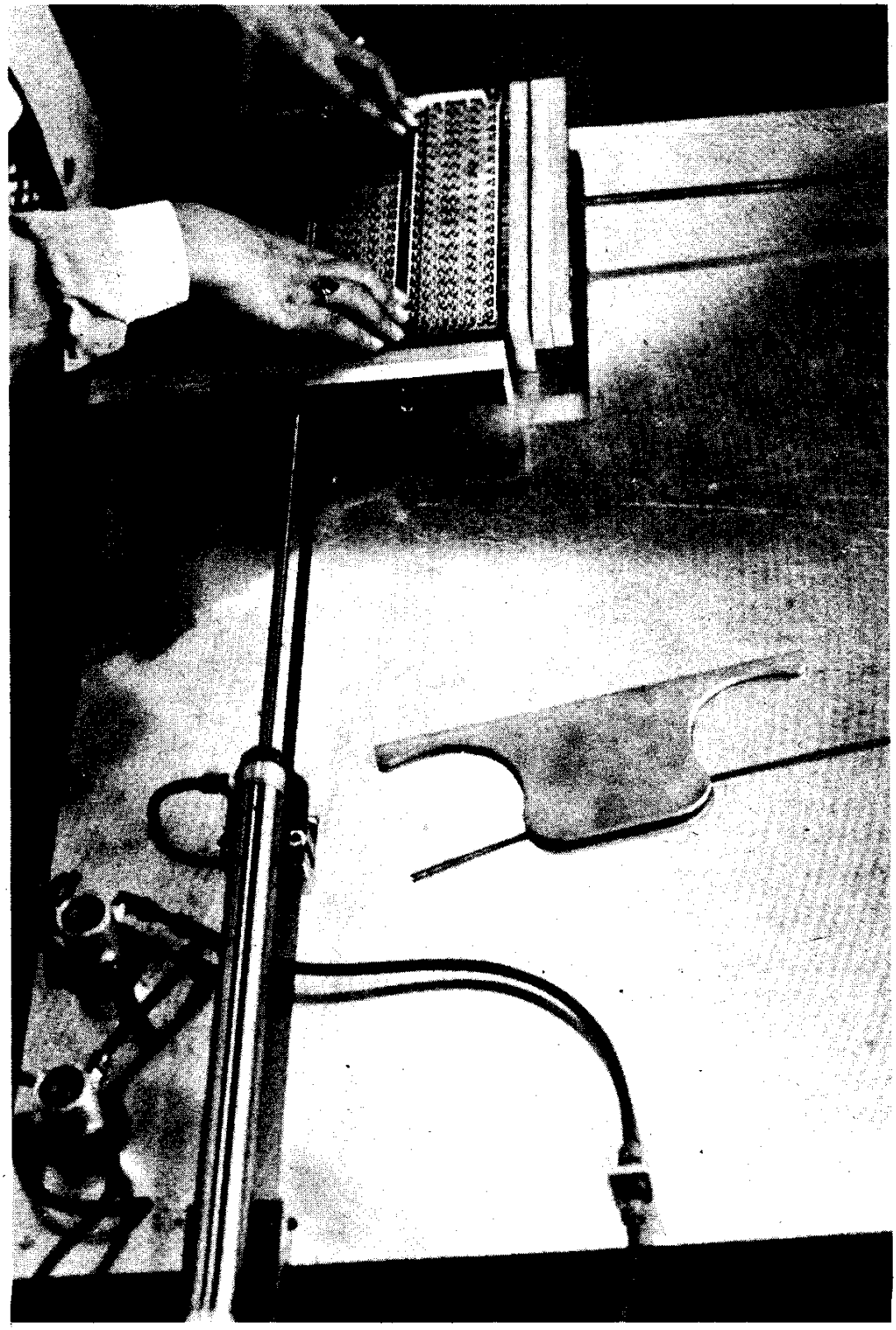


Twenty module boards are drilled simultaneously from a PDP-8 computer-generated coordinate tape. Other pantograph-controlled machines drill up to 200 boards simultaneously from a PDP-8 computer-generated template.



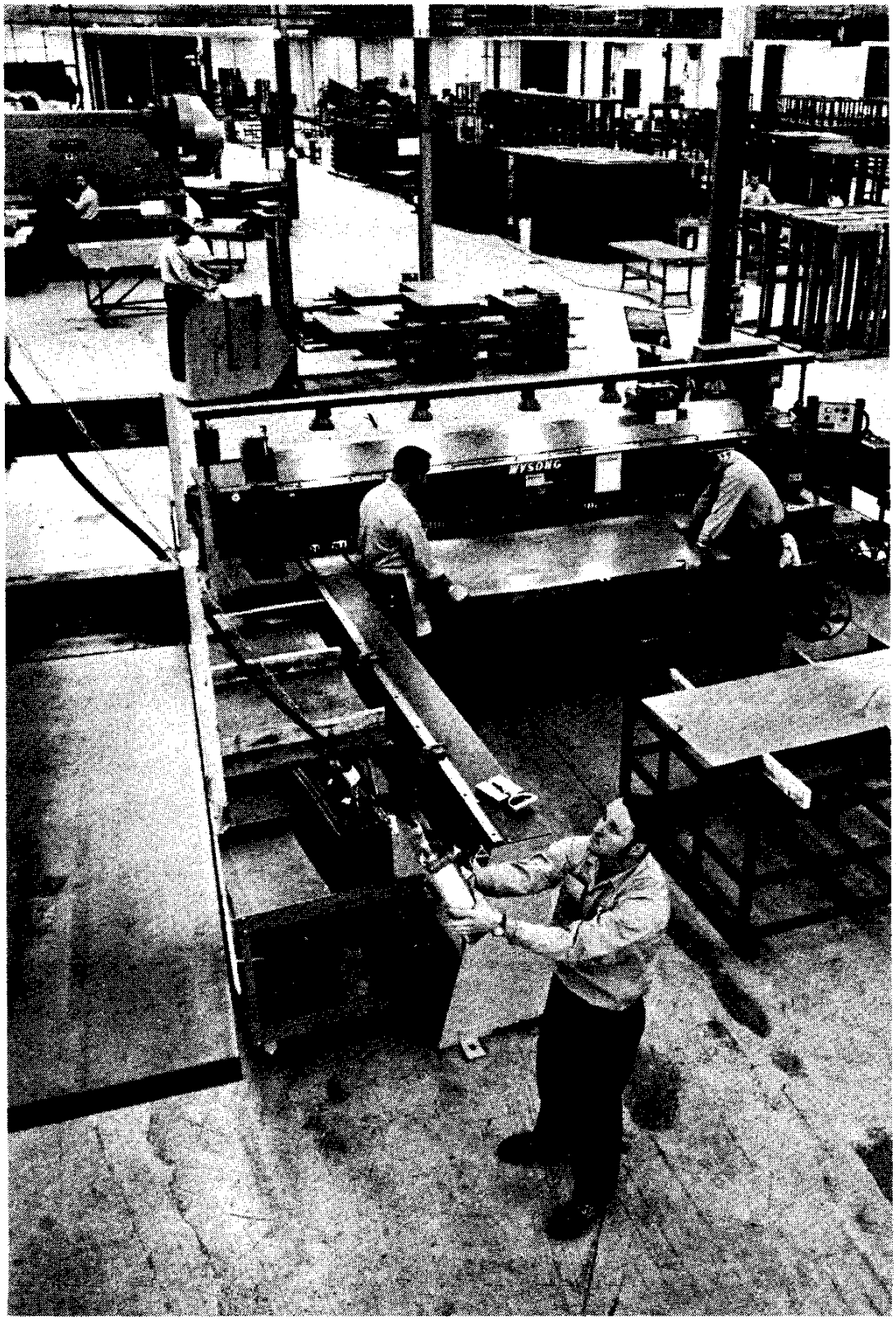
Quality of plated-thru holes is checked in our new electrochemical facility before boards go to the module assembly area.

(4) OMNIBUS Assembly

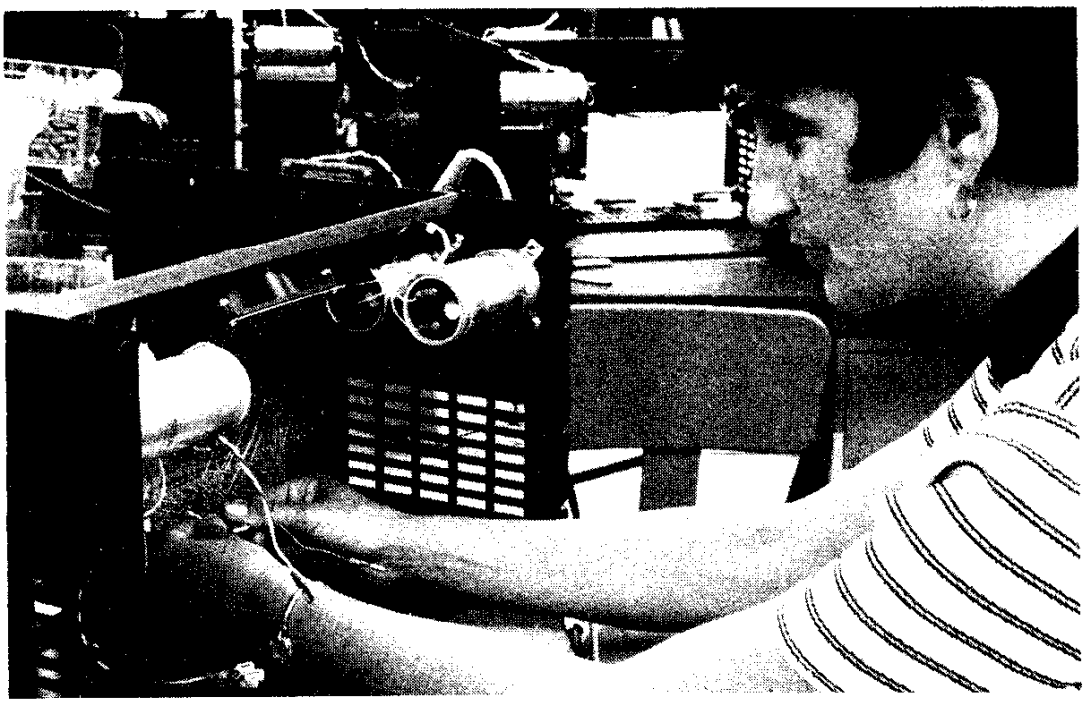


OMNIBUS Assembly—A PC Board and Connector Block are assembled here.

(5) Cabinet Assembly



Cabinets for DEC systems are manufactured in this portion of DEC's Westfield, Massachusetts production facility.

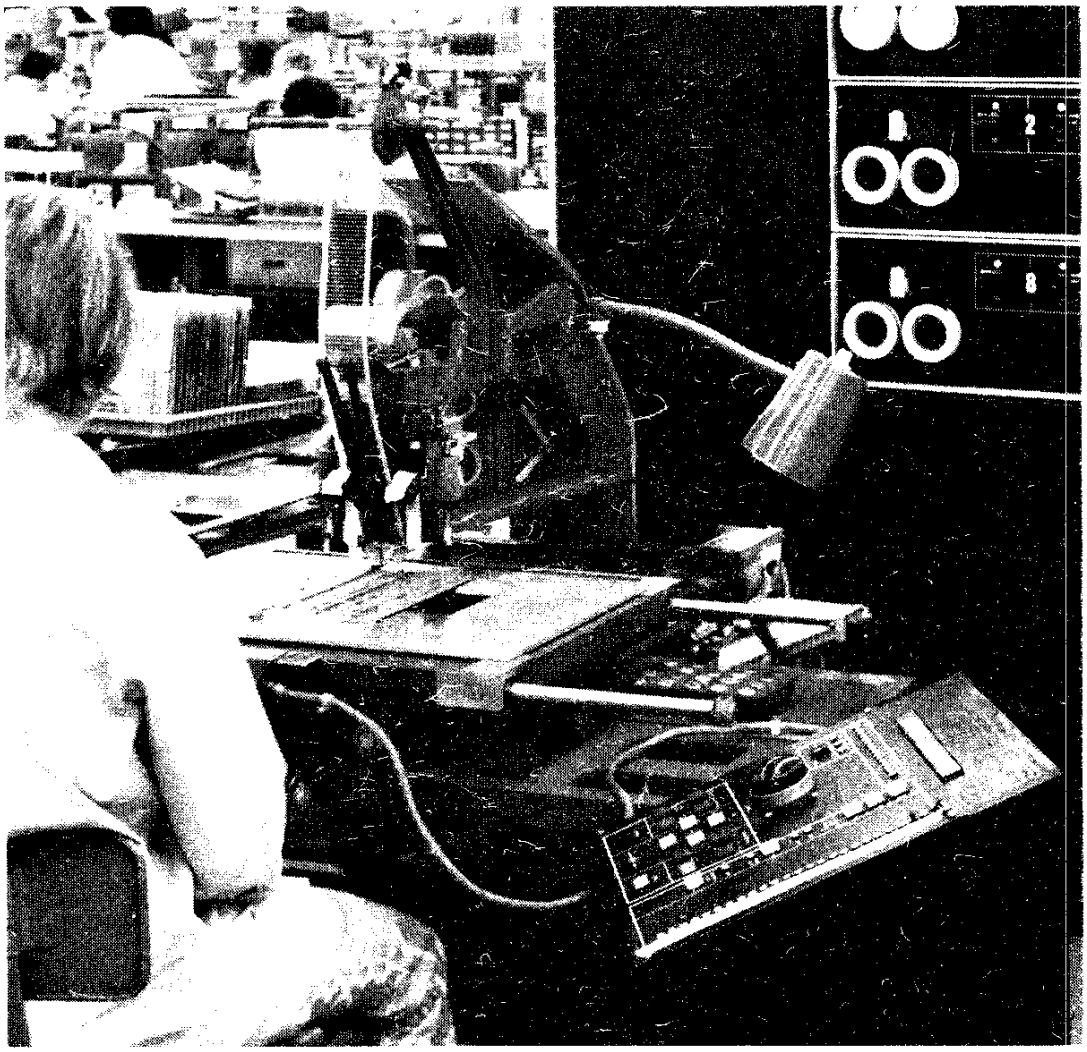


(6) Peripheral Manufacturing

The blossoming of more peripheral assembly lines is a very real indication of DEC's continual expansion of products. At the DEC manufacturing plants shown above, just such an assembly line is producing the famous DECtape. Each component is given the usual controlled inspection procedure. Modules, which are used to control the operation of each DECtape, are produced in DEC's automated module assembly area. Quality control is the highest prority item. A series of severe tests and checks are run on all products.

(7) PC Board Assembly

The PC Board Assembly includes inserting components, soldering component leads and gold plating all printed circuit connectors.

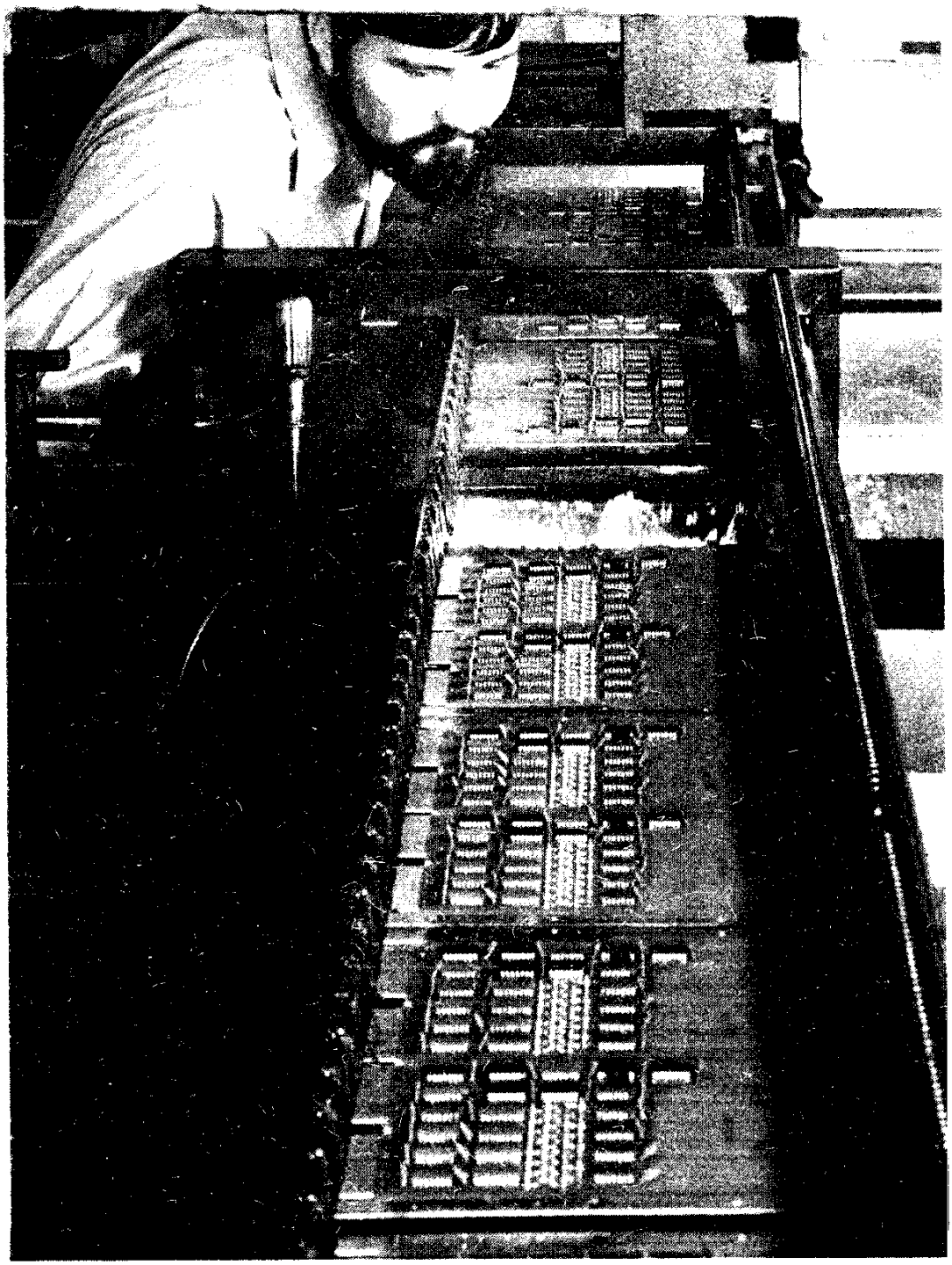


Component Insertion Machine

DEC designed and built a multistation component-insertion system to insert diodes, resistors, and capacitors into PC boards. Eight stations are controlled by one PDP-8 Computer. Each station contains a component-insertion machine with table driven stepping motors directly coupled to a rotary incremental-optical encoder.

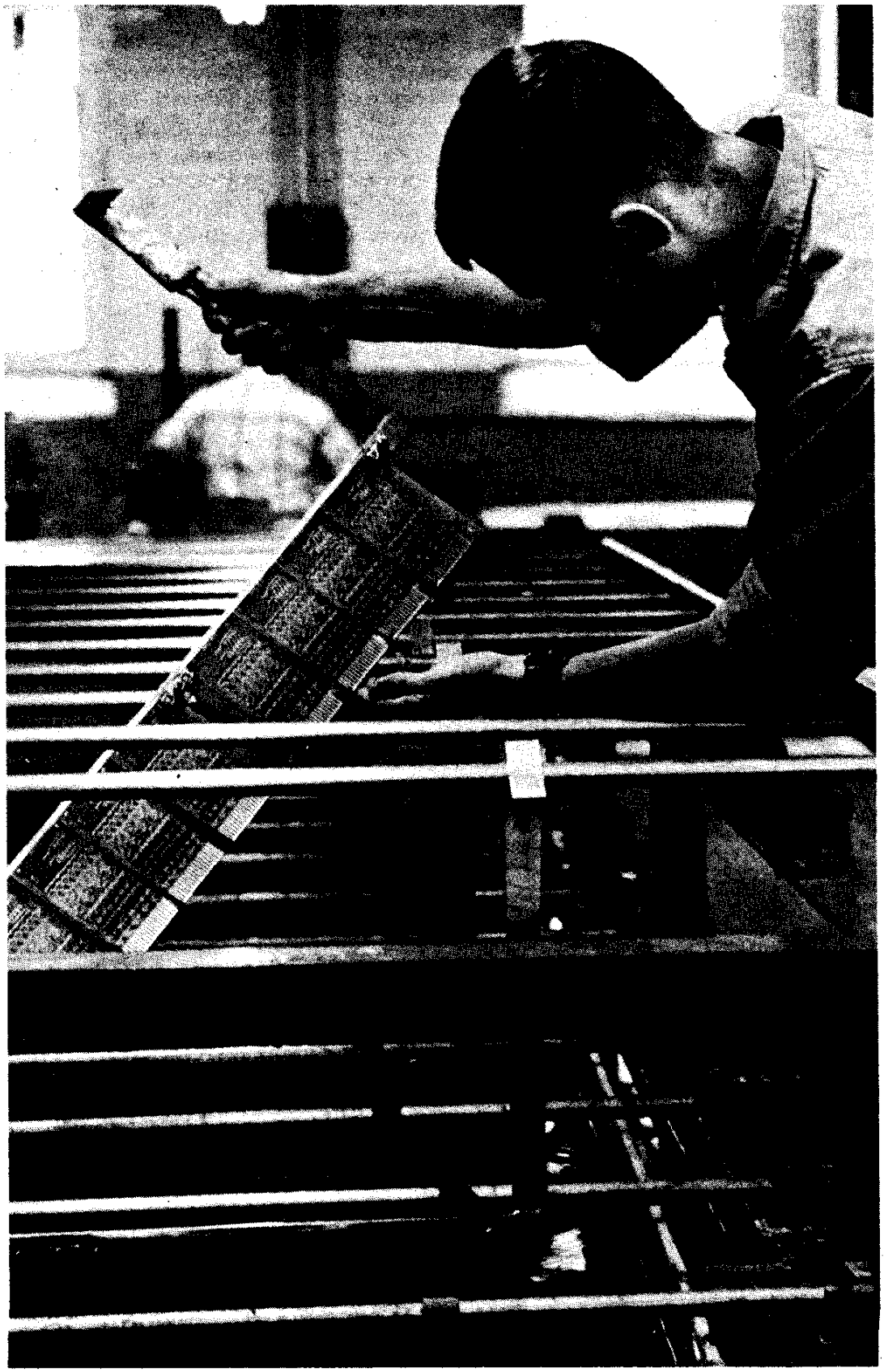
An X-Y table holding a batch of printed circuit boards is stepped back and forth under a stapling mechanism that inserts electronic components into predrilled holes on the boards at high rates.

The PDP-8 System uses a magnetic tape deck containing a library for PC-board parts lists. Each station has a custom-built control panel that permits the operator to start, stop, back up, go forward, jog-in offsets, and select parts from lists. The electronic parts are loaded into the insertion machines in paper-taped belts on large reels.



This flow-soldering machine solders all component leads to the board and makes all solder runs in one fast, exceedingly reliable, operation. More than 1000 modules are soldered on this assembly line each day.

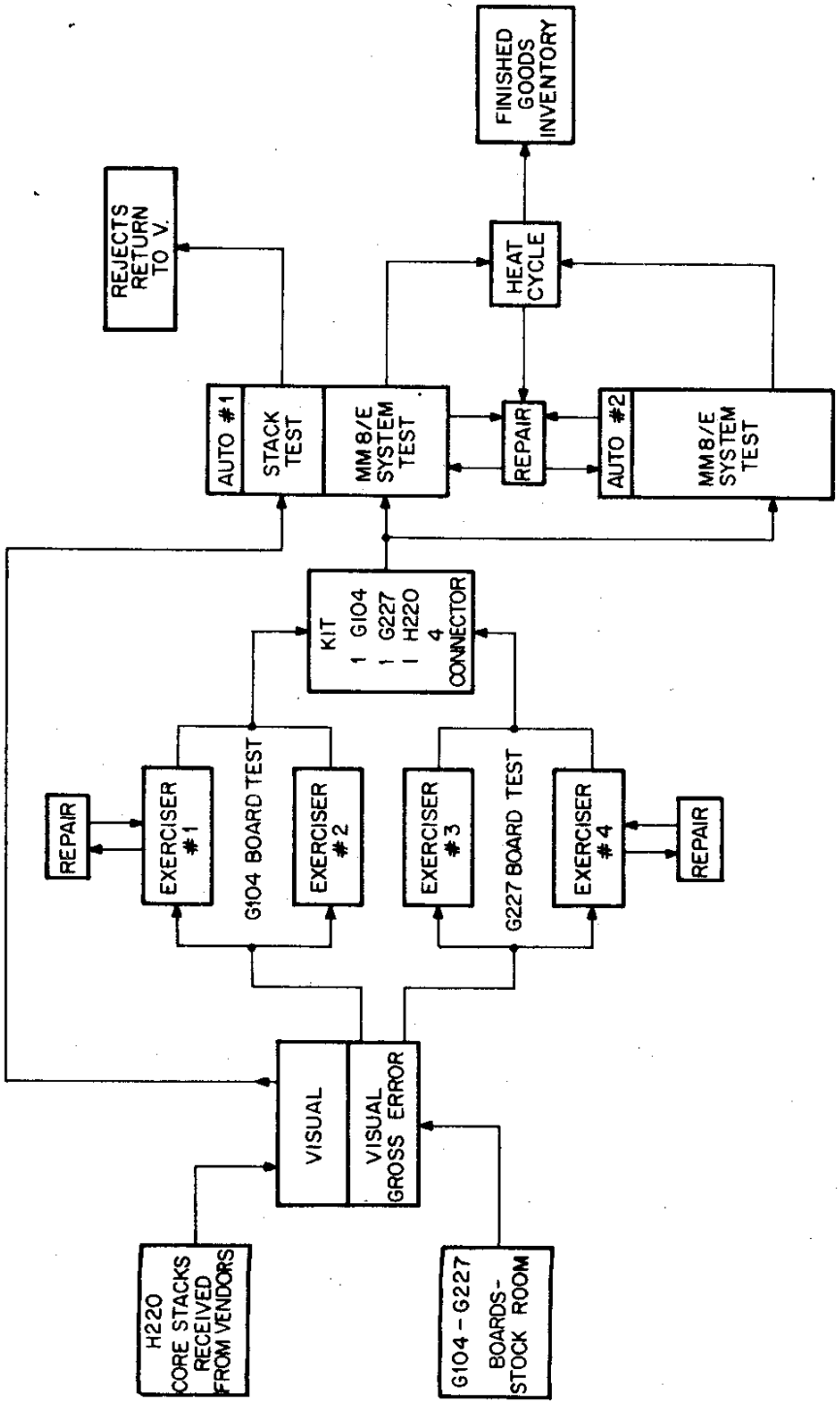
DEC has more than 2 million square feet of manufacturing space. This view shows a portion of a module assembly area.

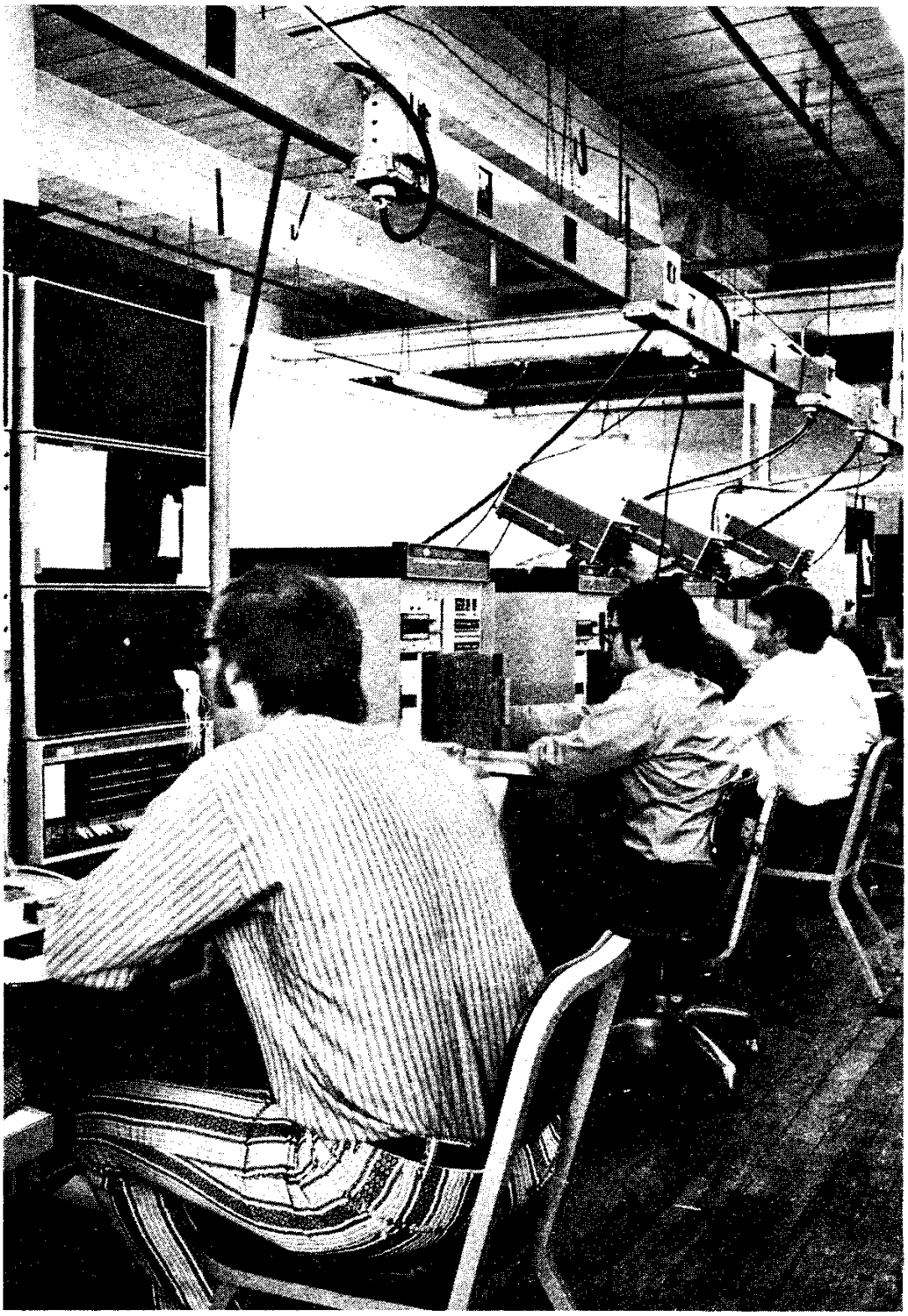


Checking the appearance of board contacts being gold-plated. Our 100 micro-inch plating is verified by periodic checking on a radiation gauge.

(8) Computers Test Memory Systems

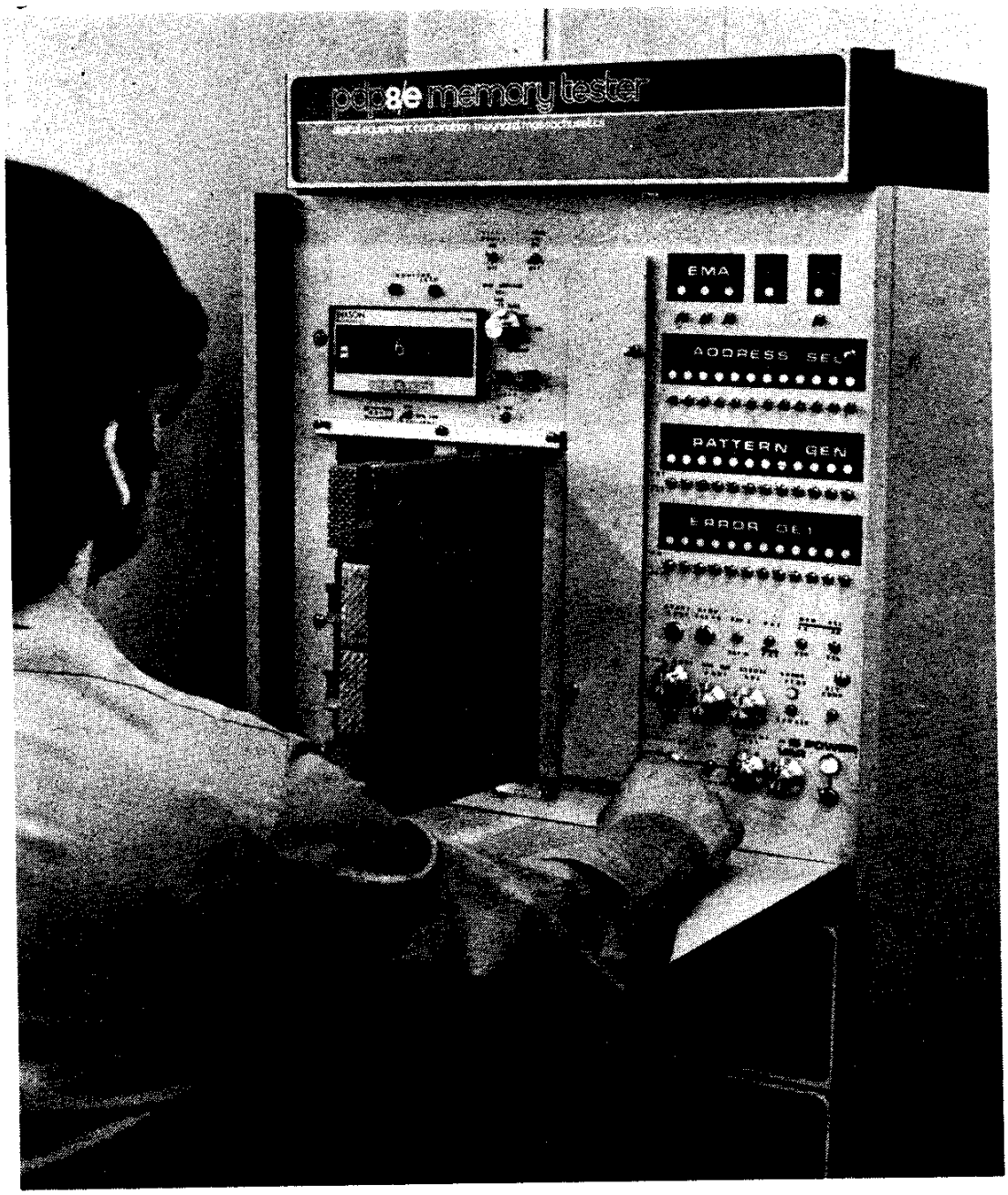
Computers perform three complex testing operations on each memory system before it is approved by quality control. Each module is visually inspected and taken to a manual memory exerciser, qualified, and placed with other modules where a memory system kit is assembled. A complete memory system test is performed, and the assembled memory system is qualified. A final test is performed with diagnostic programs, exercising the memory system at its highest specified temperature limits in a heat chamber. Refer to MM8-E flow of inspection and testing.





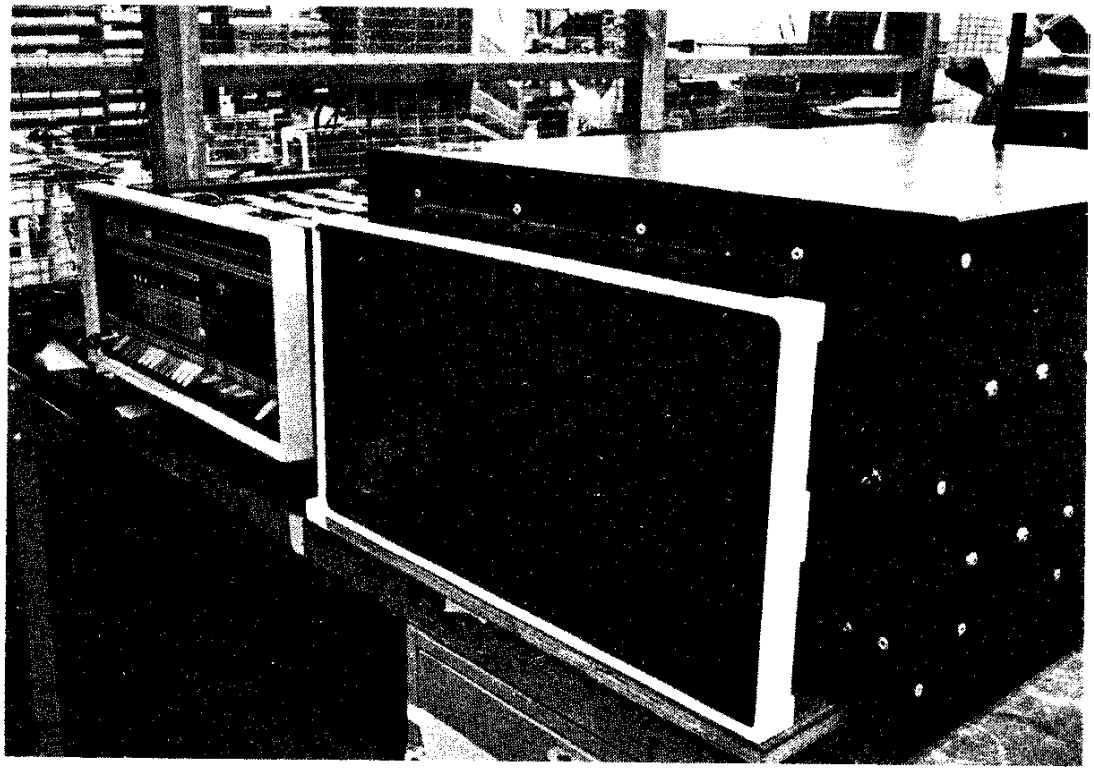
Memory Test Line

Qualifying PDP-8/E memory modules is accomplished by this test line. Every component in the memory modules are subject to thorough testing under a variety of conditions.



The PDP-8 Computer performs the dynamic testing of the memory units (MM8-E's—3-card ensemble). After each memory system kit has been assembled, the kit is tested at DEC's fully automatic station (AUTO #1 or AUTO #2) where typical operations of system characteristics are run to reflect normal operating frequency used by the computer. The tester varies the voltages and currents within the memory system upper and lower limits to ensure that the memory system meets the requirements of the specification. For each parameter tested, corresponding Schmoo-type curves are obtained. The total test time requires only 5 minutes for each memory system tested.

Again, a PDP-8 Computer is working to qualify new PDP-8 Computers. This automated testing technique allows no variation in quality; *no marginal units survive these tests.*



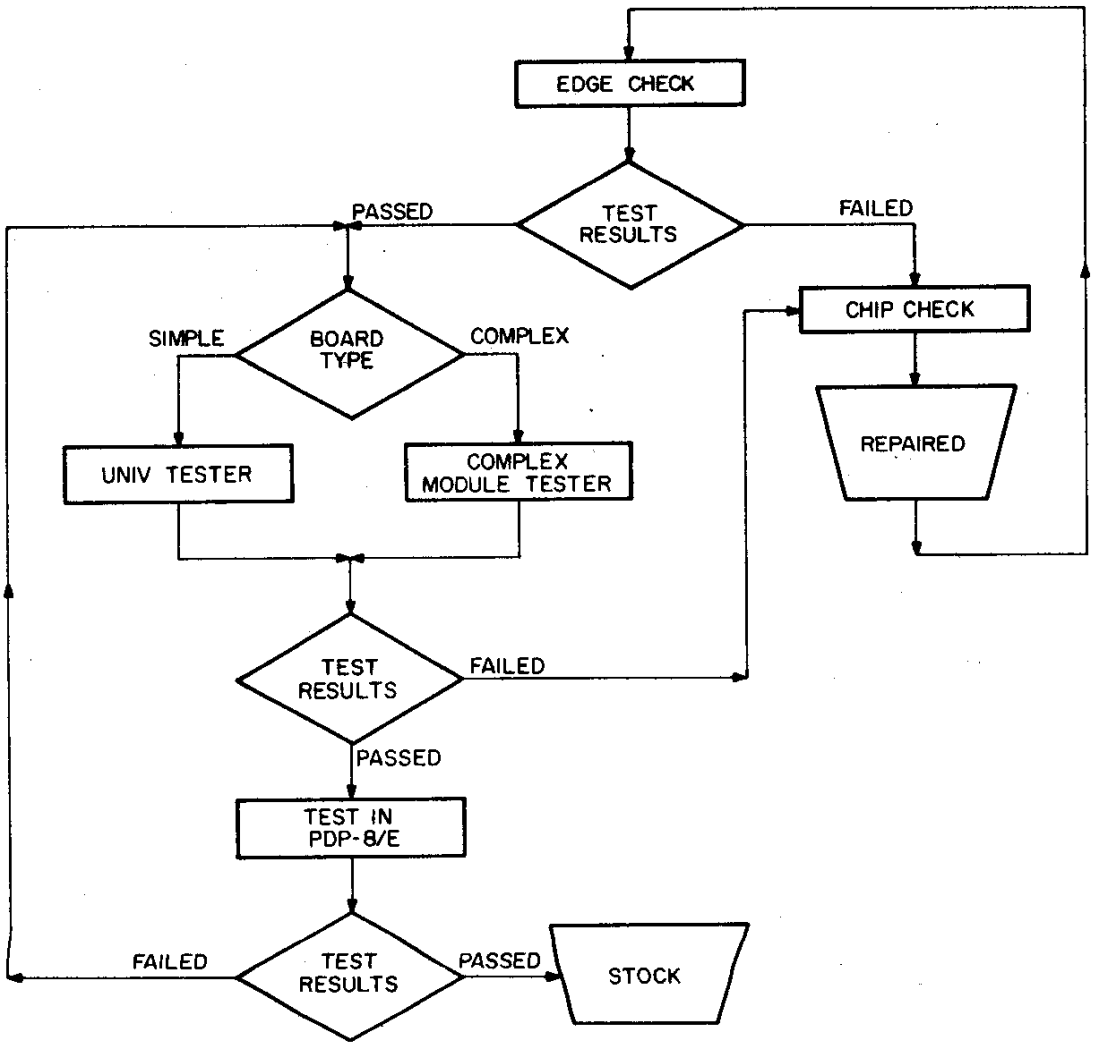
Memory System Heat Test

A final system test is performed by running memory diagnostic programs while the system is operating under maximum allowable temperature. The memory modules are installed in a heating chamber and connected to a PDP-8 Computer. If a fault occurs, a teleprinter connected to the computer prints out the type of fault; if the memory system performs flawlessly, the teleprinter prints a verification.

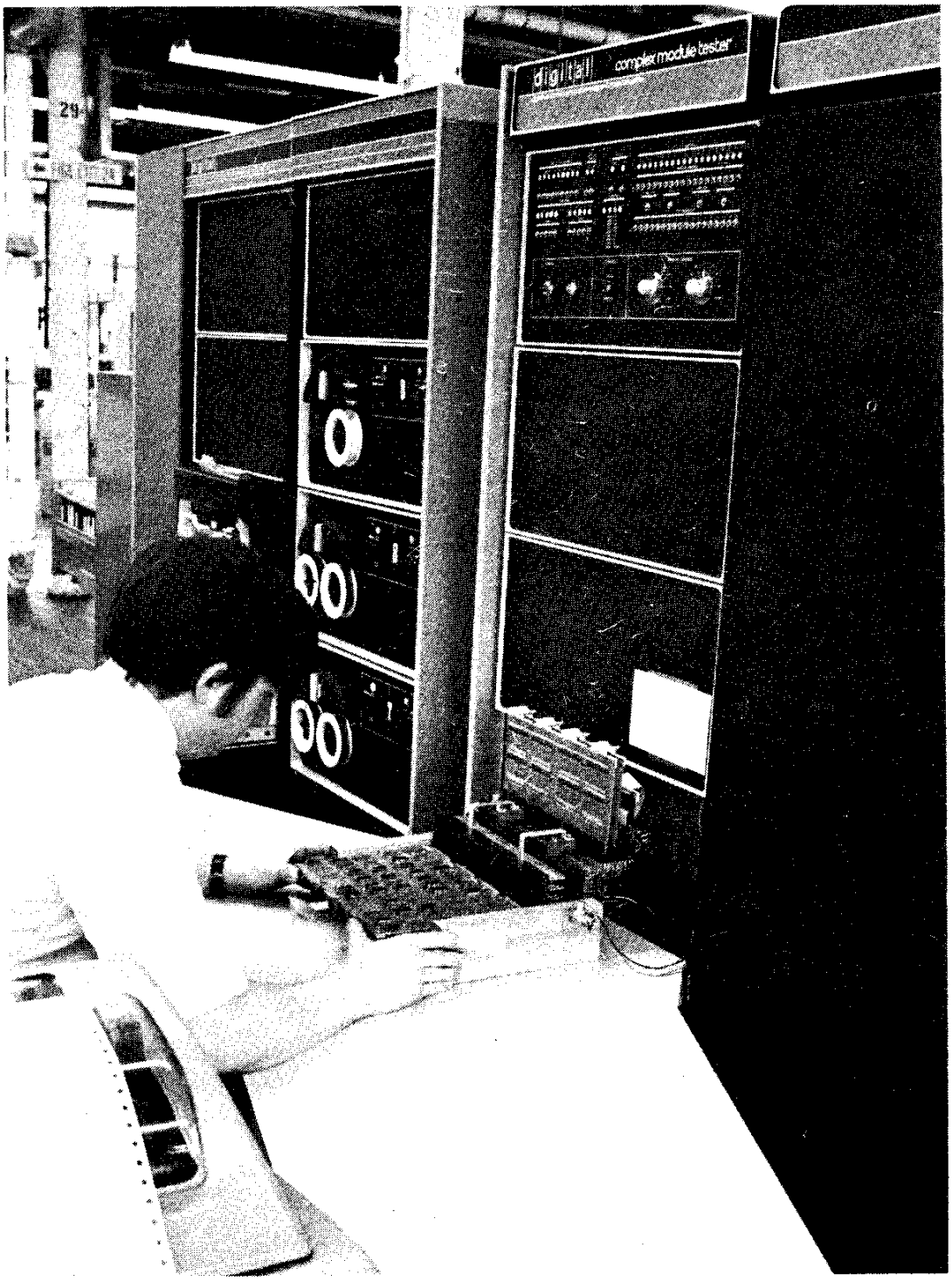
(9) Computers Test Logic Modules

A series of computerized tests are performed on all logic modules to maintain DEC's highest standards and ensure long life. Computers using diagnostic programs exercise and test every component on every module before a module is qualified for customer use. Hundreds of repetitive tests are performed in seconds as the computer evaluates every parameter, including maximum and minimum allowable current, frequency, and other important values. If a fault occurs, a teleprinter signals the operator; otherwise, the teleprinter verifies that the unit "passed the tests."

A detailed diagram of module testing is provided below, from the least complicated test to the most complicated test. Computerized testing begins with the edge check, which qualifies all of the circuit paths. If the module is simple, it is routed to the universal tester; otherwise, the module is qualified by the complex module tester. Any time a component failure is detected, a "chip" test is run to locate and replace the failed component. Each accepted module is then tested in a PDP-8/E System and qualified by a series of diagnostic programs that thoroughly exercise every component on the module.

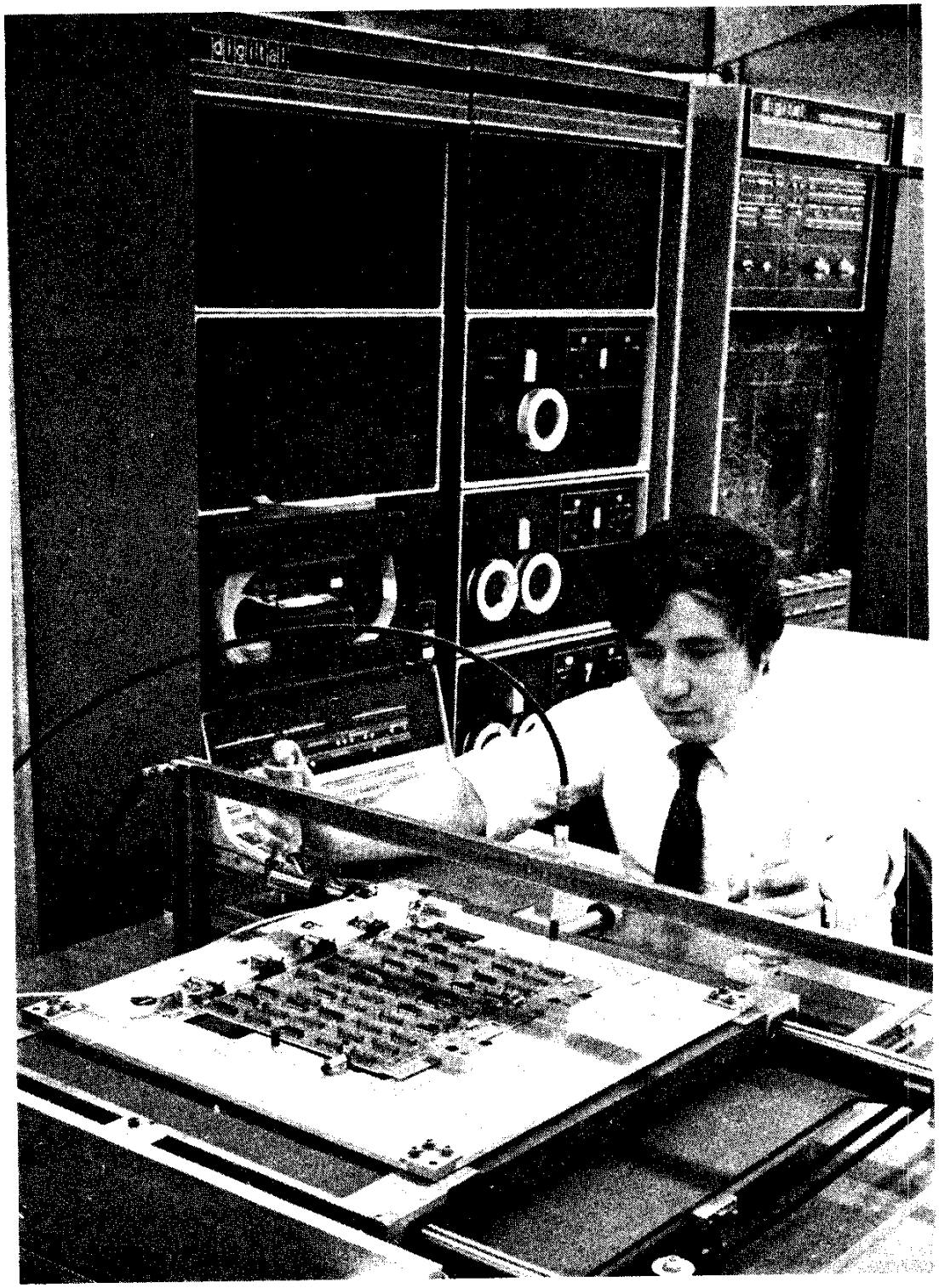


Module Computerized Tests Flow Diagram

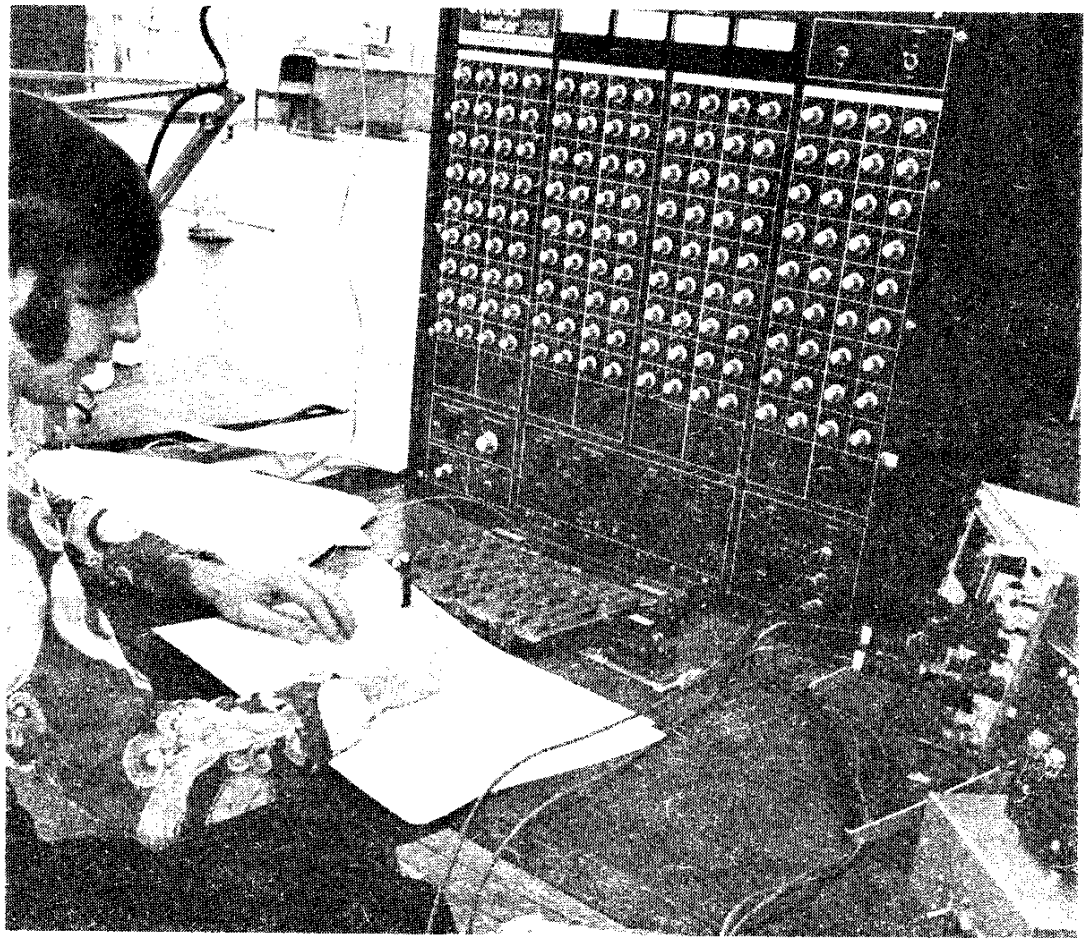


Module Computerized Tests

Complex Module Tester—The Complex Module Tester uses a Computer. The operator inserts the circuit board into a connector block and the computer applies the correct inputs and checks the correct outputs from that circuit board to verify its operation. If the circuit board is rejected here, it is passed to another test center where a technician uses the Universal Tester to further diagnose the fault.



Chip-Checker—The chip-checker tests individual IC's while mounted on a module board. This unit indexes in X and Y around a circuit board with a special probe that connects to and and checks out each integrated circuit on the circuit board. The computer in the background stores the programs for both testing and indexing the tester. DEC tests the integrated circuits (IC's) before being assembled on a circuit board by the incoming inspection method and tests once again after the IC's are assembled on a board.



Universal Tester or Logic Analyzer—This unit is the tester especially developed for PDP-8/E modules. Using this sensitive tester, a technician can isolate faults on any circuit card used in the PDP-8/E Computer. Through the various controls on the tester, the technician sets up all the various inputs that a circuit board uses. Then, with an oscilloscope he can monitor the output at various pins to verify the operation of circuit paths.

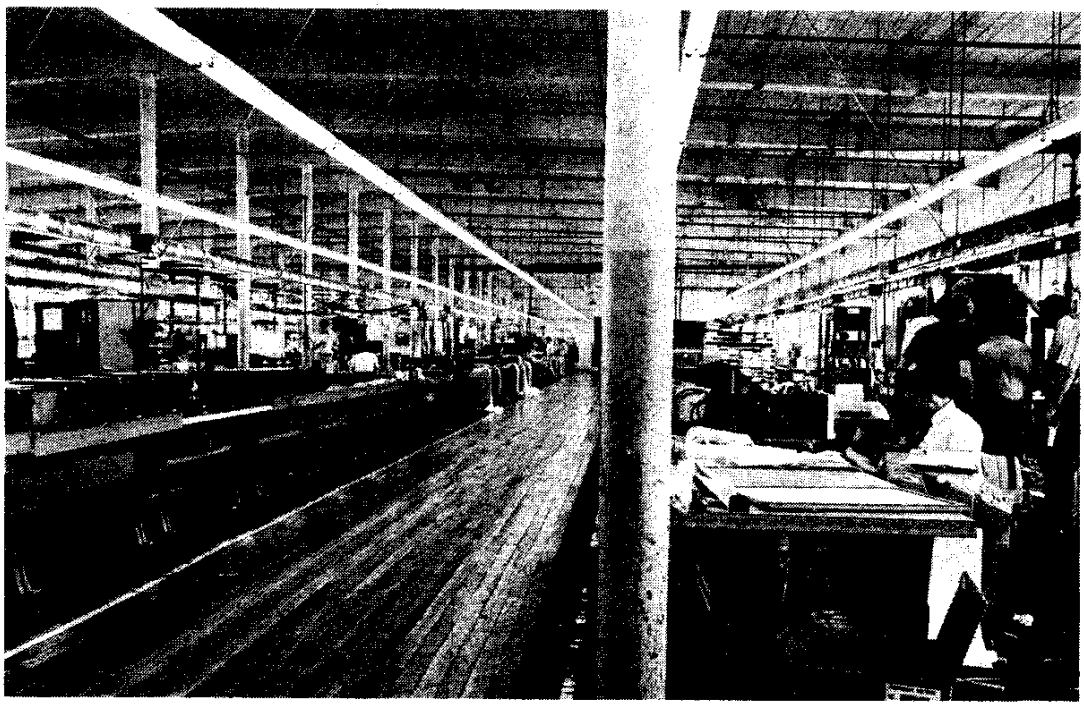
(11) Software Development

We develop new PDP-8/E software every day. Each new program is exhaustively tested on a PDP-8/E Computer before it is released for customer use. In addition to programs developed for customer use, DEC has developed a special series of diagnostic tests that are used by the various test stations.

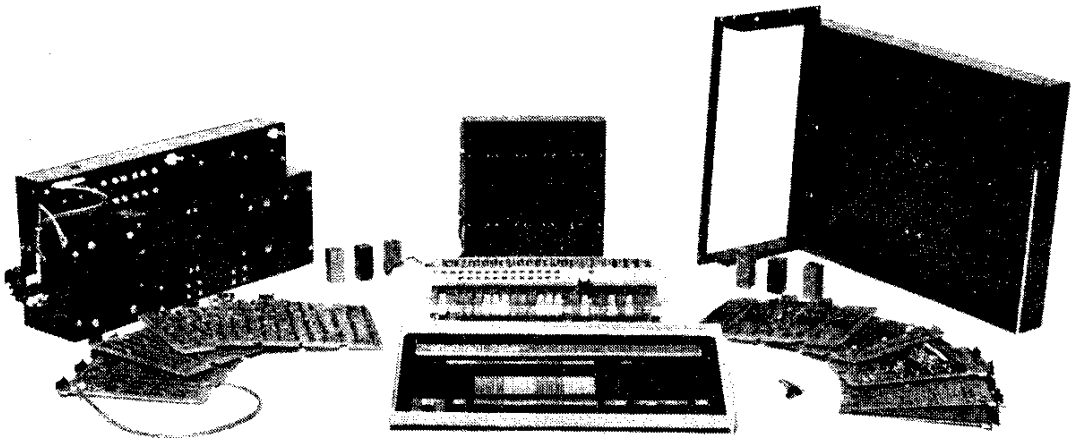
(12) Documentation Development

The explosion in computer technology demands the continual development of new computers and peripheral devices. In turn, continuing education for the people who use computers is absolutely necessary. DEC responds to this need for easily assimilated, accurate information by verifying PDP-8/E documentation with both engineering and programming. Our customers are equipped with up-to-date drawings, operating procedures, theory of operation, maintenance procedures, and programming instruction manuals.

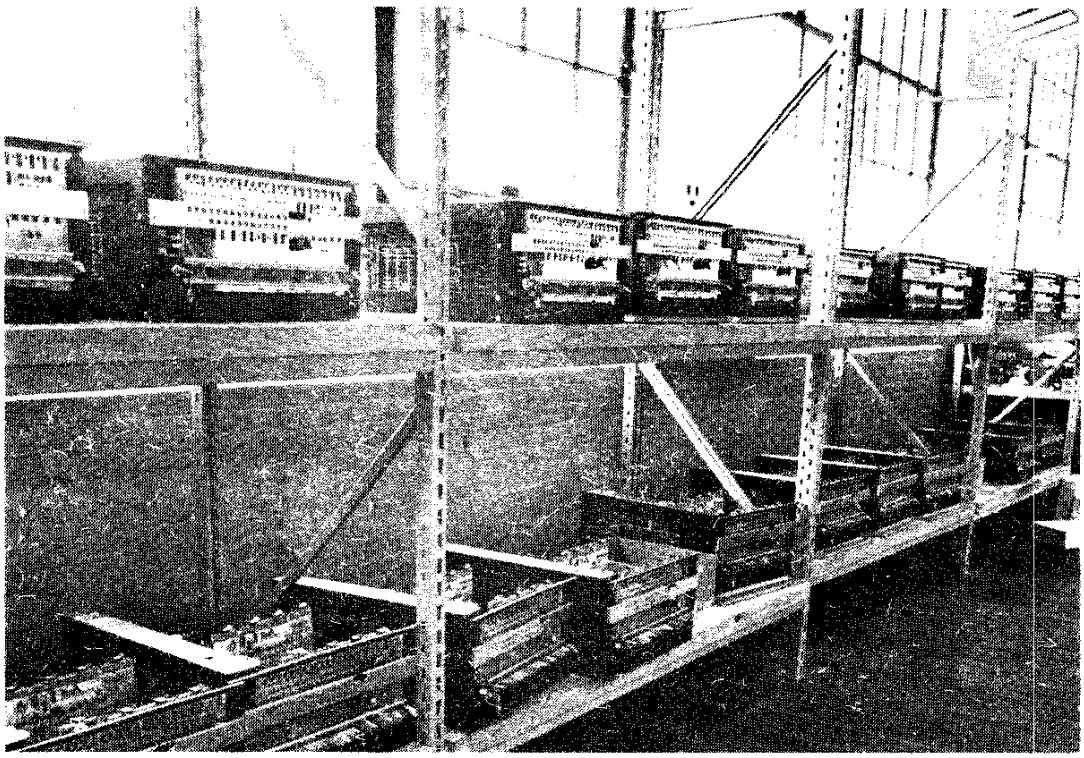
(13) The PDP-8/E Production Line

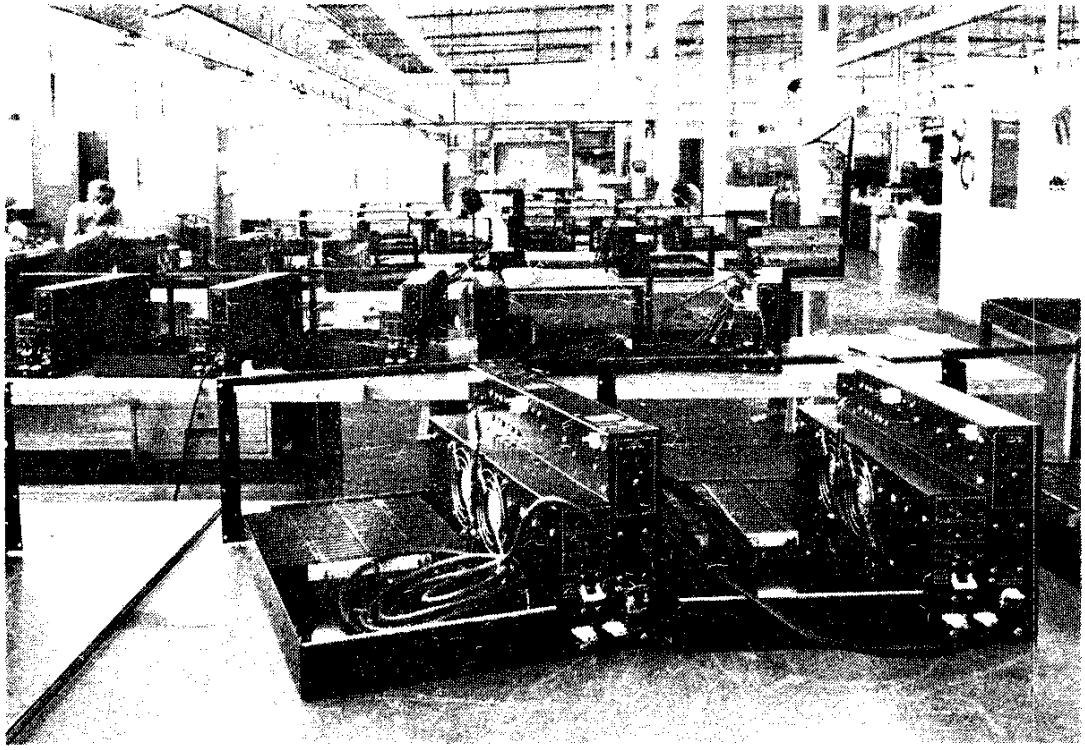


The PDP-8/E production line has the capability of manufacturing 1,000 PDP-8/E Computers per month.



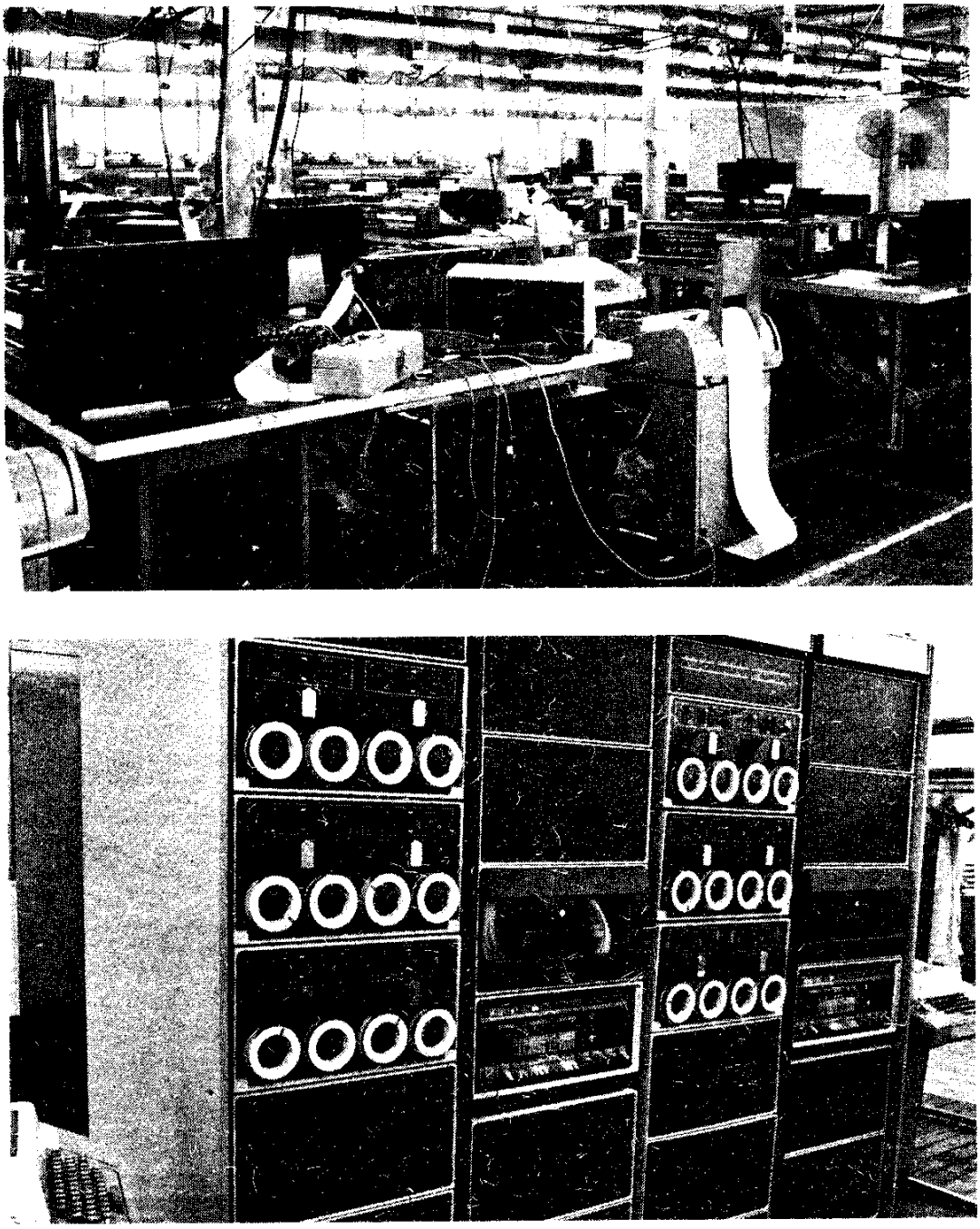
PDP-8/E System Assembly—After testing all the various components of a PDP-8/E Computer, the components are carefully assembled. This photo shows all of the components for a basic 4K Computer arranged to illustrate how modular the 8/E is and how spare parts can easily be the key to zero downtime.





Final Assembly Area

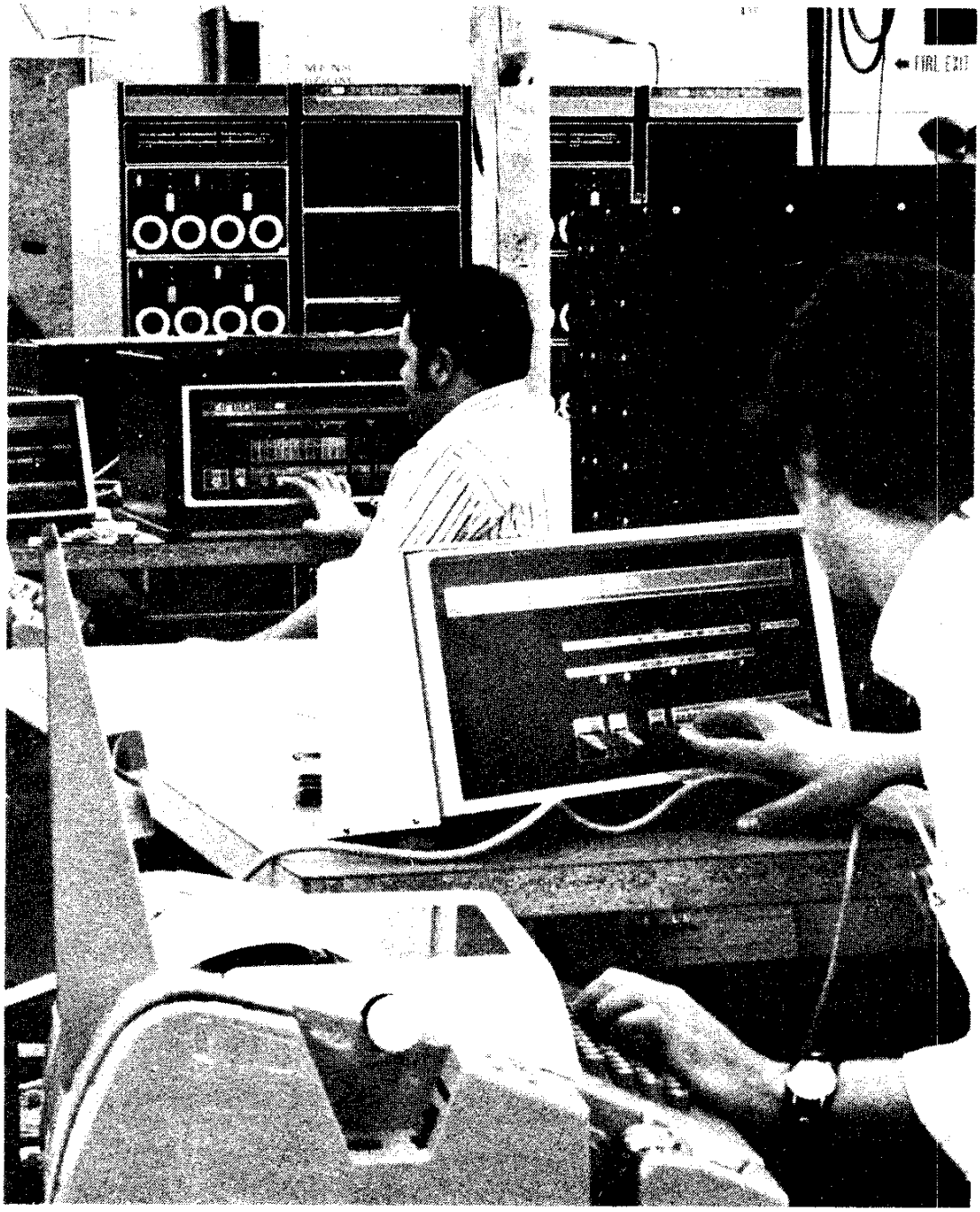
Here, 40 PDP-8/E Computers are shown in various stages of assembly. After assembly is complete, each unit is moved to another area where power is applied and the assembled unit is tested.



Acceptance Test Line

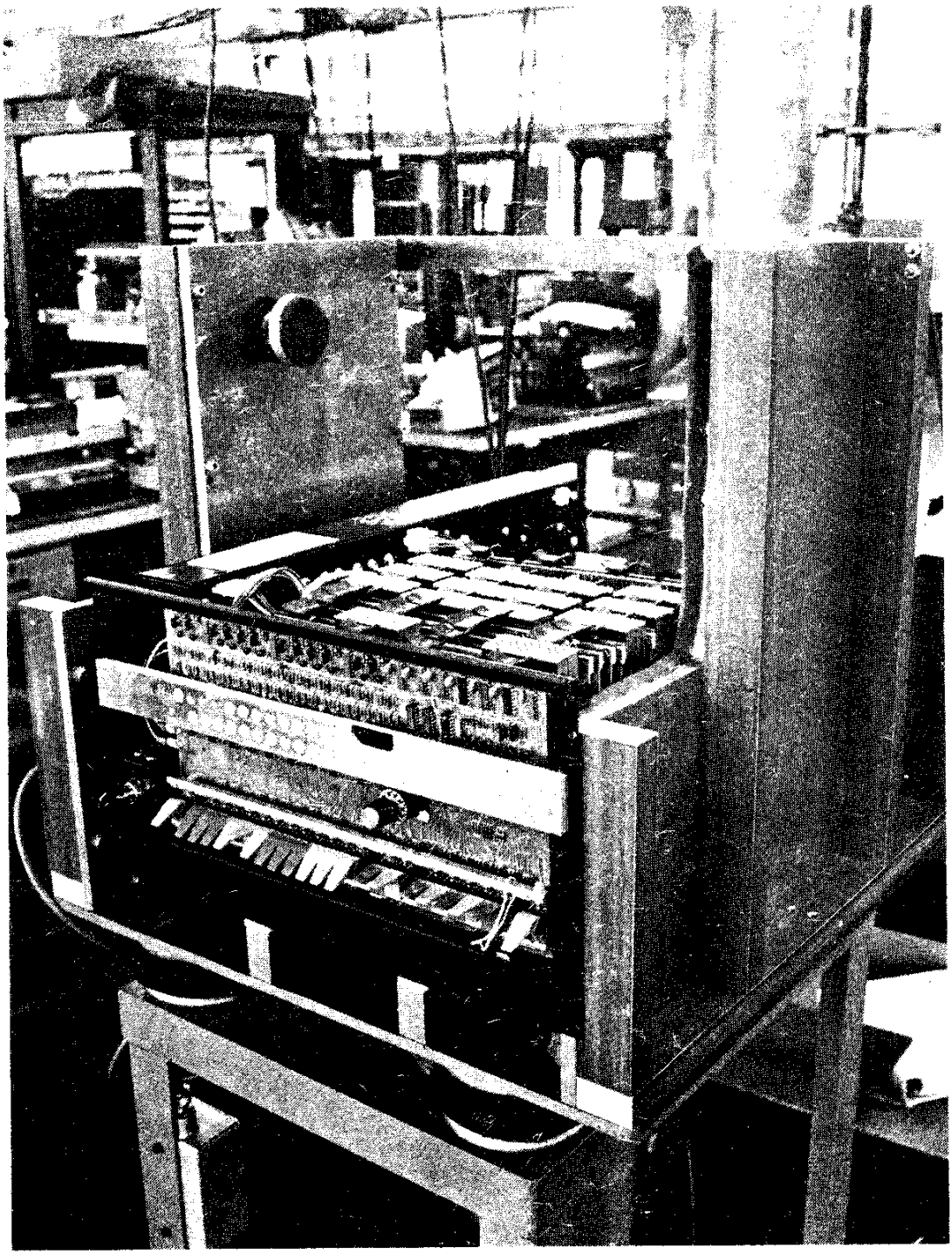
Assembled computer testing is done in DEC's acceptance test line. Up to 64 test stations can be controlled by one of two PDP-8/E Computer Systems; there are 6 DECtapes on each system, containing an assortment of test programs and exercises for each test station. Thus, 64 computers can be tested simultaneously by a master controller. The PDP-8/E master controller loads diagnostic programs directly into memory of the new PDP-8/E computers under test, thereby checking out the new computers thoroughly and efficiently. The DECtapes contain all of the programs required to check out the various PDP-8/E's, as well as the operating programs to control the entire test line.

(14) Computer Checkout



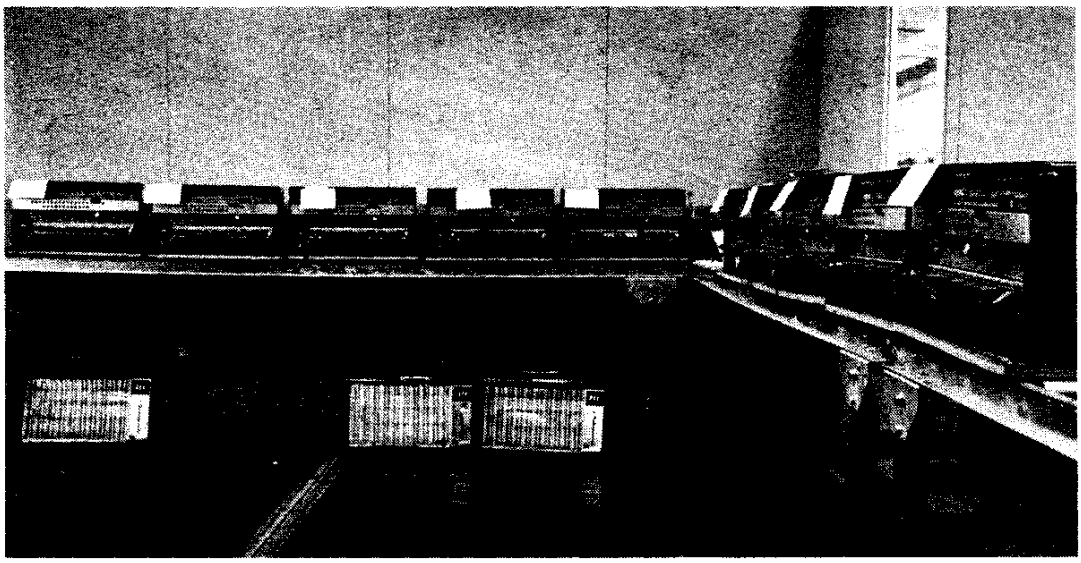
All assembled computers are tested at the 8/E Acceptance Test Station. By coding of the switches on the front of the computer, a technician can request certain diagnostic programs to be loaded into the PDP-8/E. Another switch enables Auto or Manual operation. The technician can either manually go through each test program while he is watching the results or place the switch in the Automatic Mode allowing the PDP-8/E Computer to continually cycle the various test programs through the unit without an operator. On the far left of the test panel is a switch labeled HEAT BOX. This switch activates the heater elements of another unit (not shown) and gives the computer a final heat test at this station.

(15) Vibration Test Station

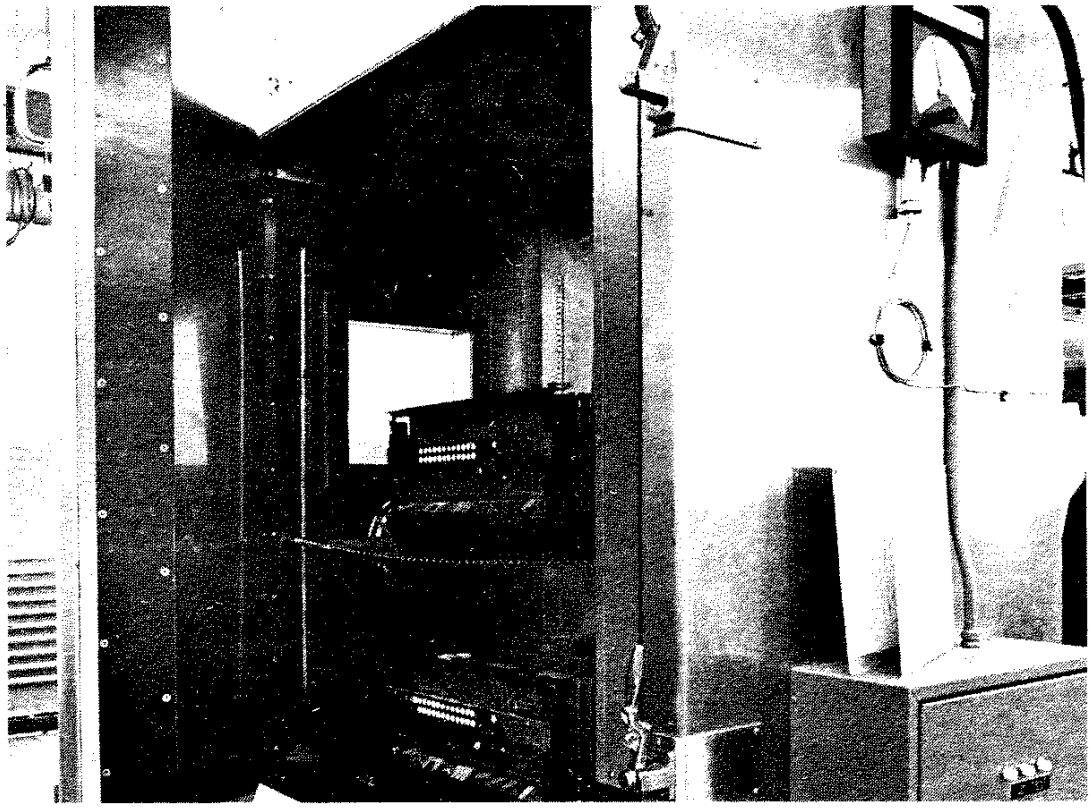


In this production phase of the computer testing, the 8/E is placed on a vibration table and vibrated for several minutes at 70 cycles per second. This test checks for any loose components, cold solder joints, and other malfunctions that can appear under severe vibration conditions. Following this test, the computer is rechecked with the various diagnostic programs. While the unit is undergoing the vibration tests, the memory checkerboard diagnostic is run.

(16) Sample 100-Hour Heat Tests



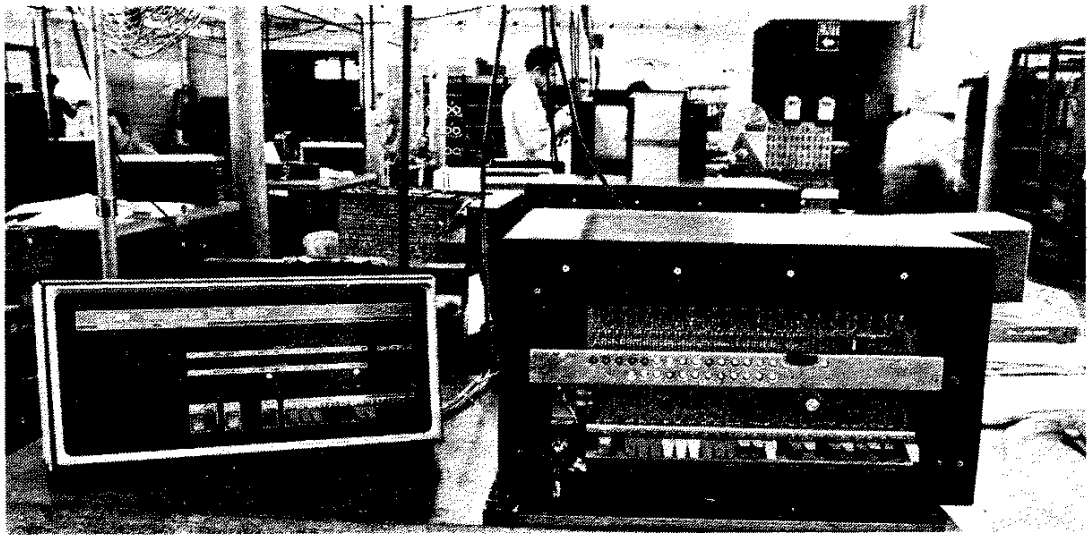
DEC takes a random sample of working PDP-8/Es and runs them for a period of 100 hours at 131° F. This workout allows us to check for "early failing components or sub-assemblies." The information gained helps us to improve the long-term reliability of all the units. In another test, all 8/Es are placed into a cold chamber at 32° F or 0° C. This forces a computer through another thermal shocking process with a very rapid change in its temperature. Following this cycle, the machine is returned to the heat room at 131 degrees F. This two-stage cycle not only verifies operation at the specified limits, but also subjects the machine to much more stress than the environmental change in the field.



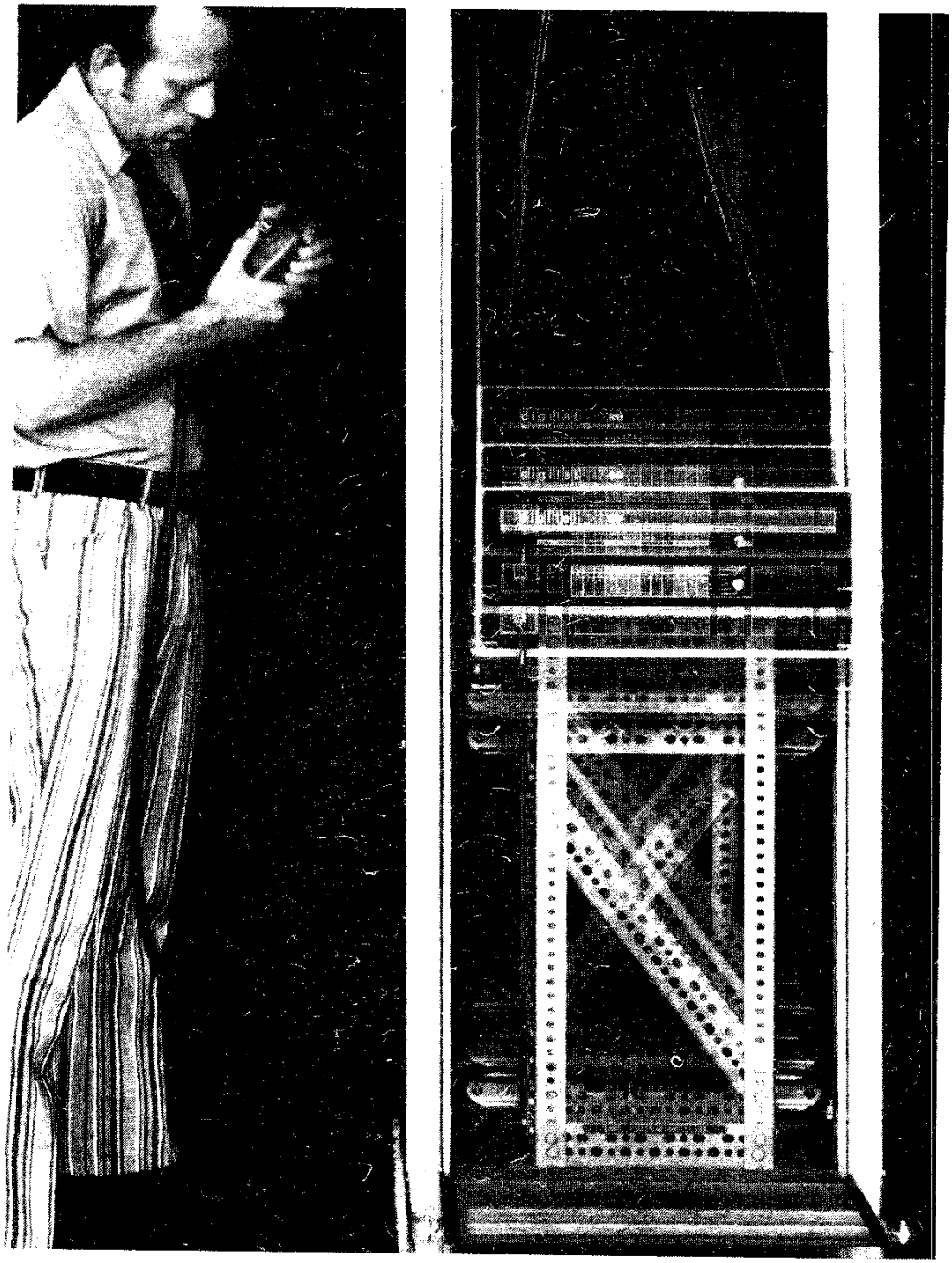
Cold Chamber

(17) Thermal Shocking

As a part of the testing and acceptance process, we place each computer in a cold chamber and a memory checkerboard program is run. The chamber temperature is reduced to the minimum specified temperature of the computer; then, the computer is placed in a heat chamber to operate at 131° F. The acceptance test station detects any faults while exercising the computer under test.



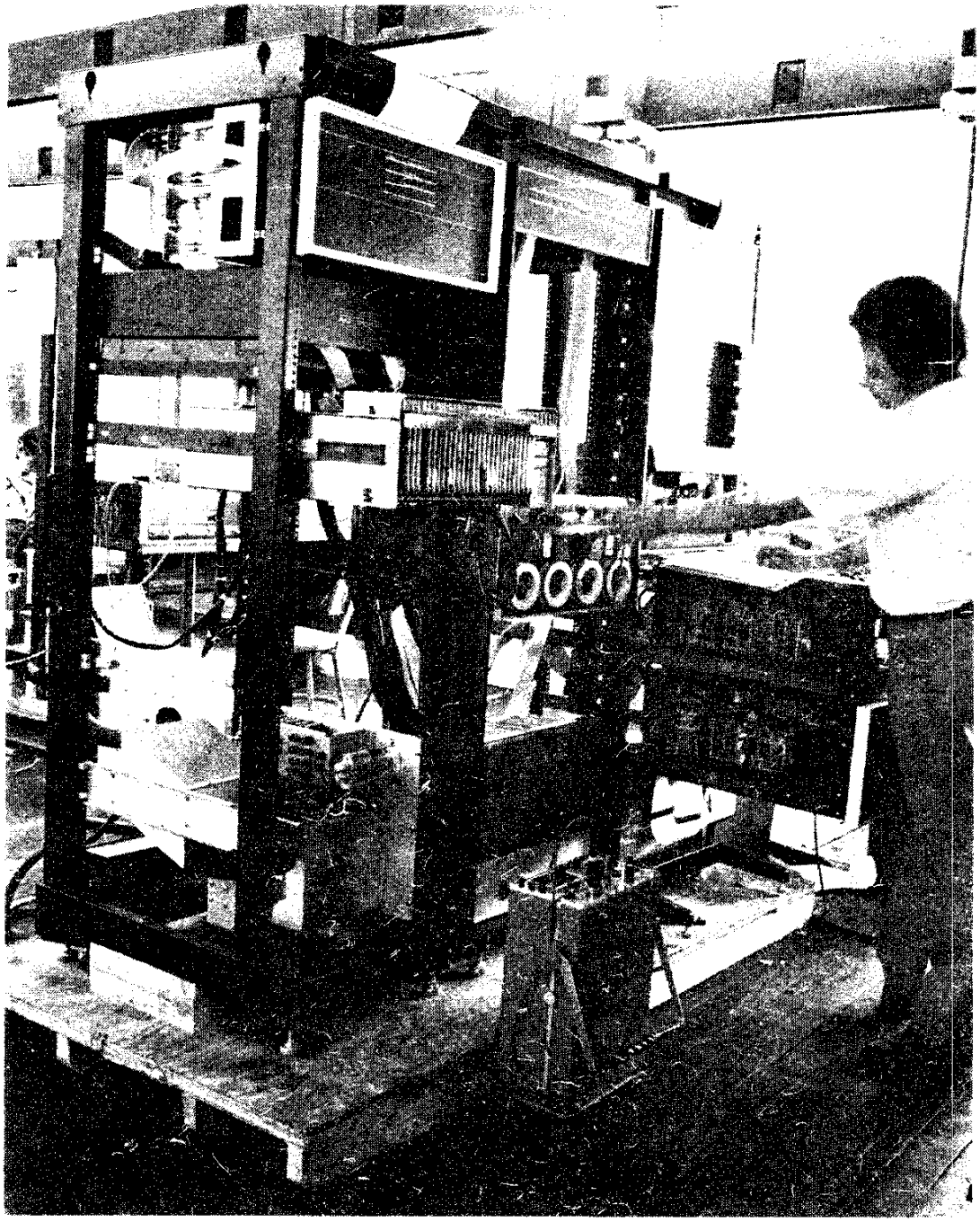
Heat Chamber



(18) The Drop Test

Some of the most frequent problems in initial installation of a computer are caused by the vibration and rough handling during shipment. To combat these problems, DEC has devised a test that is even rougher than your local transportation company. The 8/E is raised approximately 3 feet above the lower platform and then dropped *hard*. The test is calculated to place the various components in the 8/E under a 20G force. A second test is performed with the 8/E in a vertical position (panel up) with a 16G impact force.

(19) Quality Assurance and Field Service Acceptance
At the end of the acceptance test line, the Quality Assurance and Field Service Acceptance groups (independent of the production test groups) run their own tests to verify the quality and performance of the units being shipped.

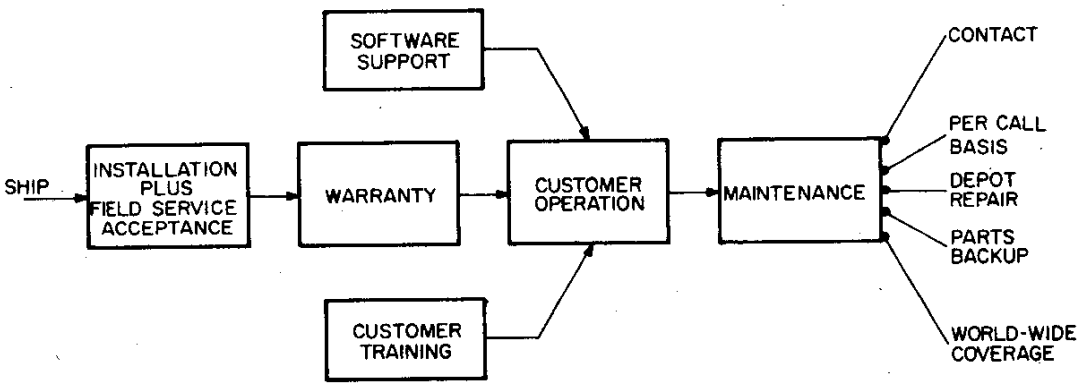


(20) PDP-8/E System Assembly and Test

After checkout of the basic computer and its internal options, the unit is moved to the system assembly area where it is installed in a cabinet containing peripheral equipment to form larger systems. In the system assembly and test area, all customer-ordered options are assembled and tested to make absolutely certain that the system is operating according to equipment and program specifications. This continuing testing process assures DEC's customers, all over the world, that each system delivered will go right to work for them and provide many years of reliable service thereafter.

CUSTOMER SERVICE

With the PDP-8/E computer fully checked out and shipped to a user facility, the scene shifts from the factory to the customer. Each PDP-8/E computer or system is installed by DEC's Field Service engineers. Each installation includes system performance checkout using a series of diagnostic programs and other programs to establish successful operation. Each system (depending upon the purchase agreement) is fully backed by a warranty which assures the customer of complete DEC support at no cost for a period of 90 days.



Customer Service

To further support the customer, DEC provides a software support service that assures a complete trouble-free operating software package.

For OEM* customers, DEC provides special documentation support on equipment produced by the OEM. DEC will provide a complete system package containing both theory of operation and maintenance.

How to use the PDP-8/E system and how to maintain it is another customer need that DEC satisfies by offering classroom and laboratory instruction designed to familiarize each customer with his system. Courses include programming, hardware familiarization and system familiarization that provides instruction on how to program a system, how to operate a system, how to maintain a system, and detailed knowledge of the system so that a customer may design and build interfaces to the system.

Each customer has the choice of maintaining his own system or employing DEC Field Service to support his system. His option does not stop there; he may elect to purchase a service contract or simply call his local DEC field service to obtain support on a per call basis. DEC support does not terminate; it continues throughout the life of the computer. The second PDP-1 computer system produced by DEC in 1959 has been supported by DEC Field Service for more than 12 years. This service will continue indefinitely.

* OEM — Original Equipment Manufacturer

CUSTOMER TRAINING PROGRAMS

Digital Equipment Corporation offers an extensive training program to every organization that purchases or presently owns a DEC computer. Our training objective is to familiarize the user with the hardware and software associated with his computer system, and with this in mind, we provide eleven courses for the PDP-8 Family Computers.

Software: (Programming)	Five courses ranging from a fundamental Introductory Programming Course to a sophisticated monitor system course. Designed to enable the user to: utilize the standard system software, write his own system programs, incorporate DEC programs as part of his system programs.
Hardware: (Maintenance) (Engineering)	Six courses ranging from hardware familiarization to system maintenance. Designed to enable the user to: isolate and evaluate problems if they occur, design interfaces for his system.

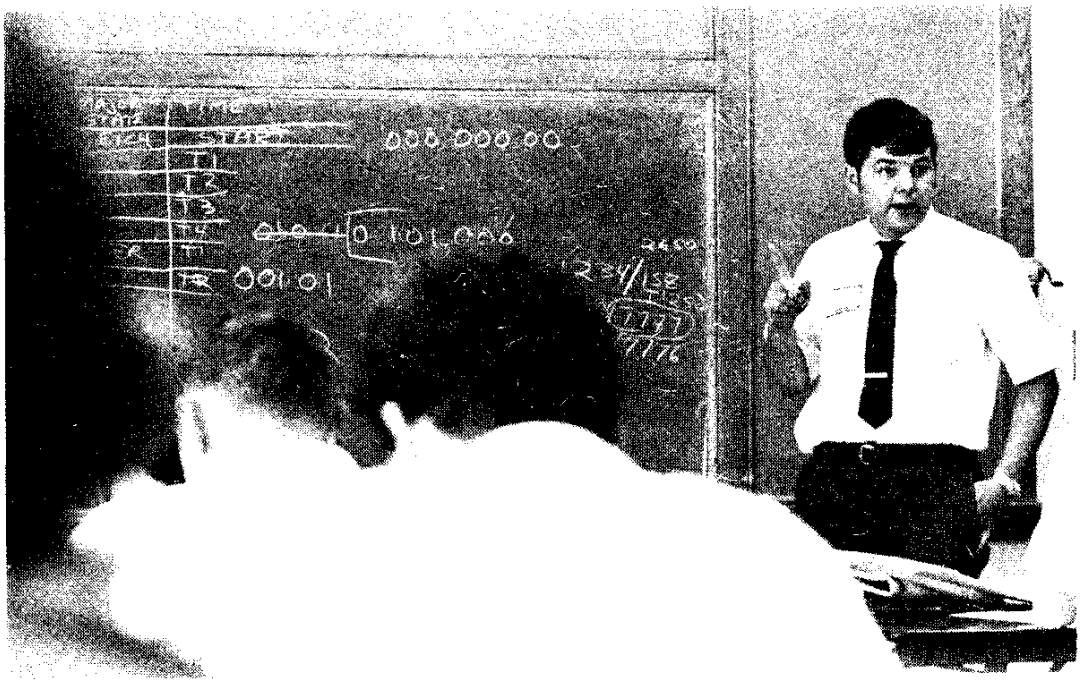
Digital offers training facilities in many countries in the world. We presently have training facilities in Maynard, Massachusetts; Palo Alto; California; Australia; England; France; Germany; and Scandinavia. Our training staff consists of full time professional instructors who continually re-evaluate our courses to ensure the content is current and that it meets the needs of our students. Special Arrangements can be made to conduct courses on-site.

The next few pages illustrate our training environment—from the formal classroom aspect to the lab sessions where the student reinforces his classroom learning with actual programming and debugging time on a computer system.

After completing their training, our students leave with a "can do" outlook. Come and find out for yourself.

For further information about our training program and the scheduling of our courses, check the appropriate block on the information request card in the back of the book.

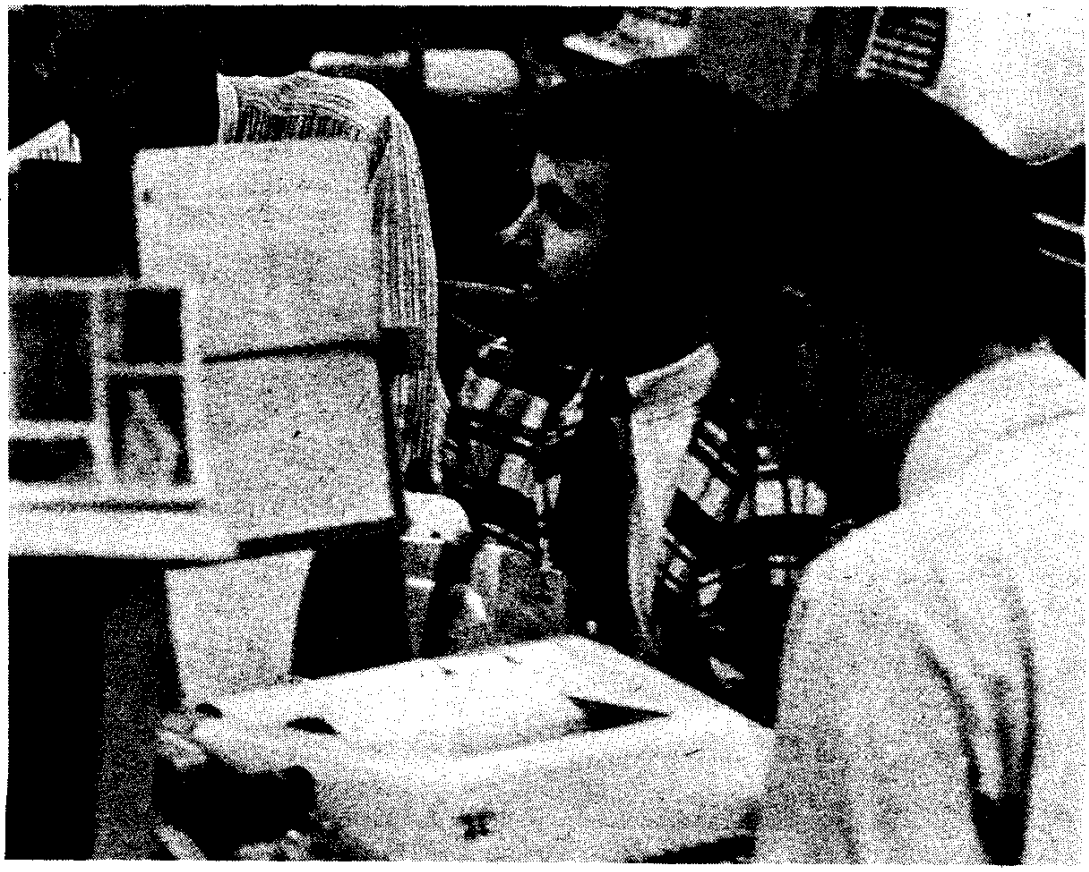
Each Digital customer is provided the opportunity to familiarize himself with all aspects of our computers and peripheral equipment. Professional class rooms employing the latest techniques are used to train customers to maintain and program the PDP-8/E and peripheral equipment. Well equipped laboratories with a complete array of equipment are employed to assure a high level of confidence of each graduating student. Courses are offered from the beginner level to the more advanced level of instruction.



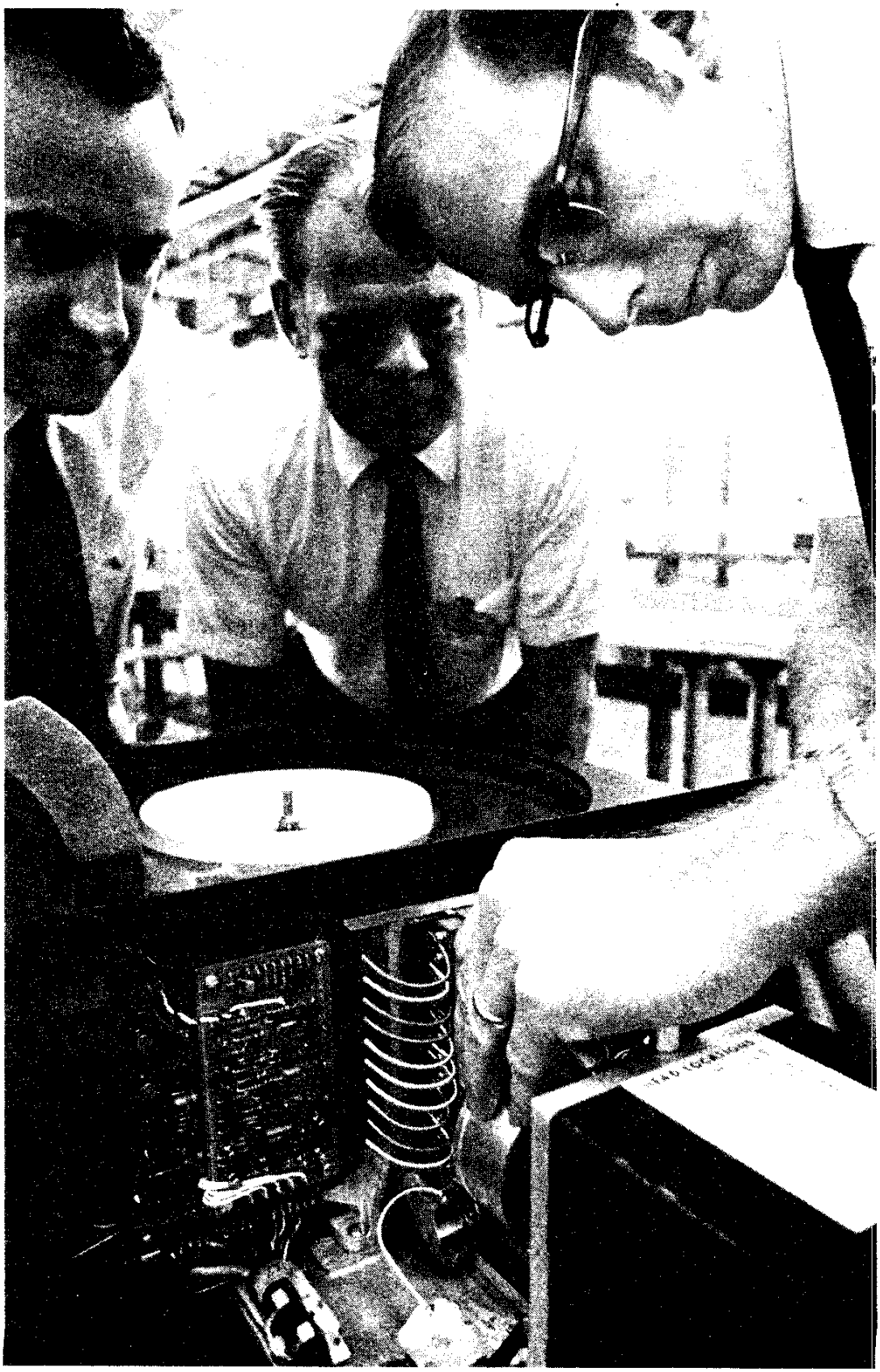
A hardware class goes through the logic with a timing breakdown.

Software students utilize lab periods with the computers to reenforce classroom learning.

One of the training laboratories—usually a very busy place.



Happiness is—an assembled, edited program that works.

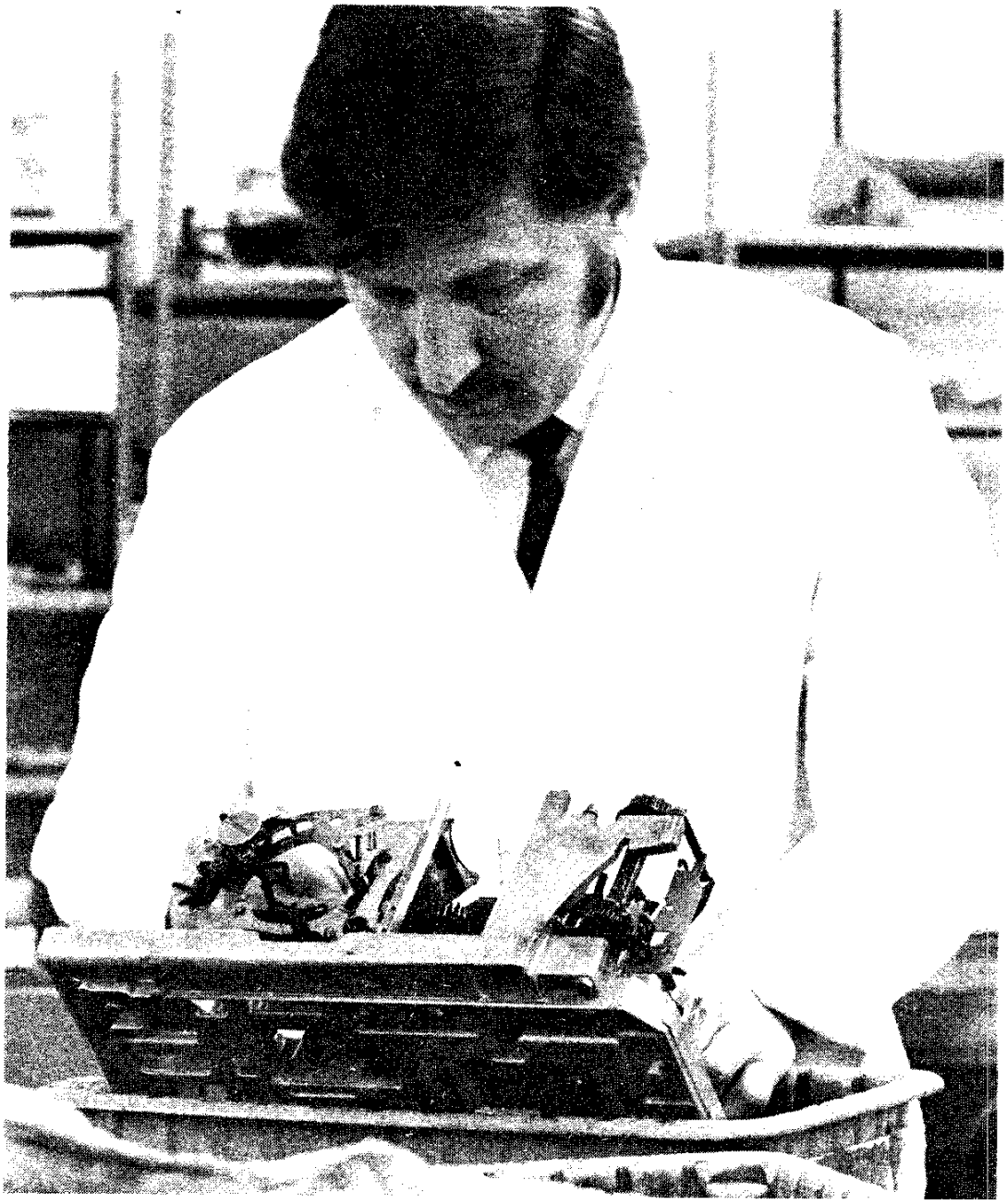


A peripheral class investigates the inner workings of one of our disk pack units.

REPAIR SERVICE

The key to maintaining your PDP-8/computer system is no further away than your telephone. Digital Equipment Corporation provides 113 service centers throughout the free world employing nearly 1000 trained engineers for repair and a complete range of technical assistance.

This field service engineer is not out to set the world's record on servicing a computer. However, like all field service engineers, he is fast, knowledgeable, professional, and courteous. It is men like him that give Digital Equipment Corporation "high marks" in field service.



For Depot Repair Service

Depot repair service saves the customer money and time. If you operate on a tight budget . . . . or if the DEC products you (or your customers) use are far from our service facilities—Digital's repair depots may be the most economical solution to your maintenance problems.

Depots provide cash-and-carry maintenance and repair service on Teletypes, computers, many standard options and peripherals. You save the cost of a service man's travel time and expense. DEC currently has depots in or near Boston, New York, Chicago, Houston, Los Angeles, San Francisco, Ottawa, Munich, and London. Other services provided at these depots include trading in your old equipment, converting your teletype or punch, etc.

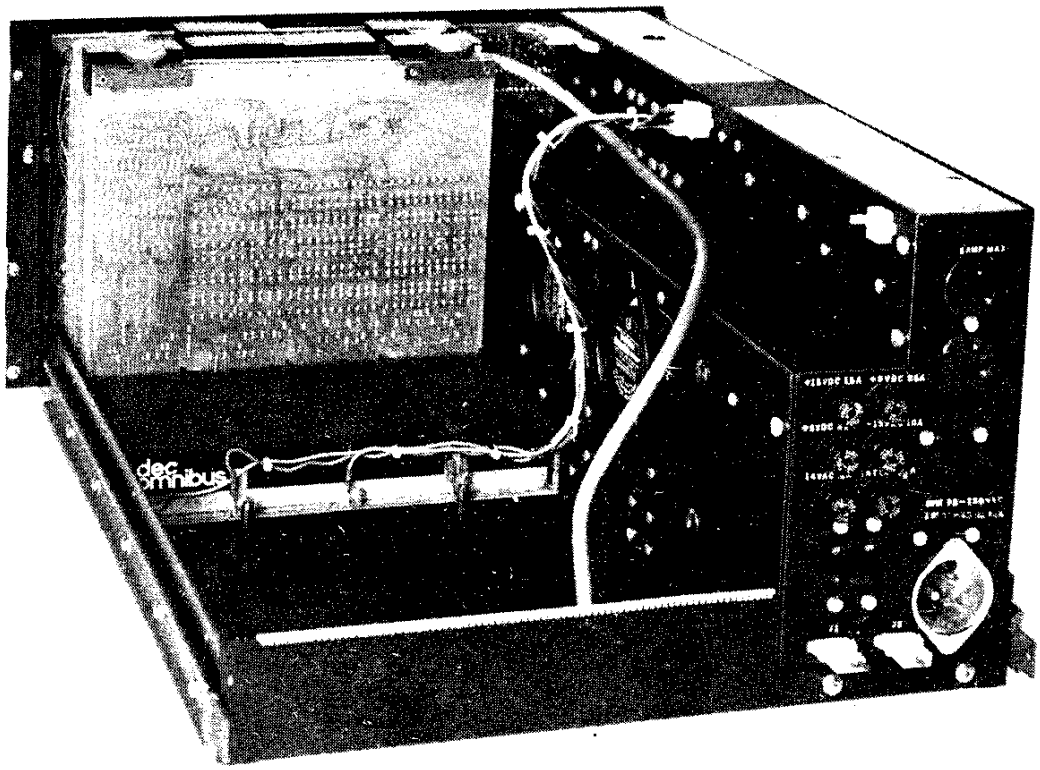
MAINTENANCE CONTRACTS

The best method of assuring that your operating system is performing in peak condition all of the time is with a field service contract. With a DEC Field Service Contract, a highly trained engineer or technician will come in at regular intervals and perform carefully planned preventive maintenance to keep your PDP-8/E in top condition. Should your computer go down, you're sure to get prompt, expert service to set it right again. Everything you need to keep up your computer is yours for a fixed monthly charge, whether you need little more than a quick dusting of the keys or a complete overhaul. All contract customers are preferred on a service priority basis.

Customer Service Contracts Guarantee Continuous Operation

part 1

BASIC SYSTEM



PDP-8/E Programmed Data Processor
(Table Model)

CHAPTER 1
SYSTEM INTRODUCTION

GENERAL

This chapter is divided into two sections: Section 1 describes the PDP-8/E basic processor and section 2 describes the PDP-8/M basic processor.

SECTION 1 THE PDP-8/E BASIC SYSTEM

The development of the PDP-8/E is the successful culmination of many years of computer design research—a process that has enabled Digital Equipment Corporation to provide better computers at the lowest possible price.

The PDP-8/E is specially designed as a general purpose computer. It is fast, compact, inexpensive, and easy to interface. The PDP-8/E is designed to meet the needs of the average user and is capable of modular expansion to accomodate most individual requirements for a user's specific applications.

The PDP-8/E basic processor is a single-address, fixed word length, parallel-transfer computer using 12-bit, 2's complement arithmetic. The cycle time of the 4096-word random address magnetic core memory is 1.2 microseconds for fetch and defer cycles without autoindex; and 1.4 microseconds for all other cycles. Standard features include indiret addressing and facilities for instruction skip and program interrupt as a function of the input/output device condition.

Five 12-bit registers are used to control computer operations, address memory, operate on data and store data. A Programmer's console provides switches to allow addressing and loading memory and indicators to observe the results. The PDP-8/E may also be programmed using the console Teletype with a reader/punch facility. Thus, programs can be loaded into memory using the switches on the Programmer's console, the Teletype keyboard, or the paper tape reader. Processor operation includes addressing memory, storing data, retrieving data, receiving and transmitting data and mathematical computations.

The 1.2/1.4 microsecond cycle time of the machine provides a computation rate of 385,000 additions per second. Each addition requires 2.6 microseconds (with one number in the accumulator) and subtraction requires 5.0 microseconds (with the subtrahend in the accumulator). Multiplication is performed in 256.5 microseconds or less 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 342.4 microseconds or less by a subroutine that produces a 12-bit quotient in the accumulator and a 12-bit remainder in core memory. Similar signed multiplication and division operations are performed in approximately 40 microseconds, utilizing the optional Extended Arithmetic Element.

The flexible, high-capacity input/ouput capabilities of the computer allow it to operate a large variety of peripheral machines. Besides the standard keyboard and paper-tape punch and reader equipment, these

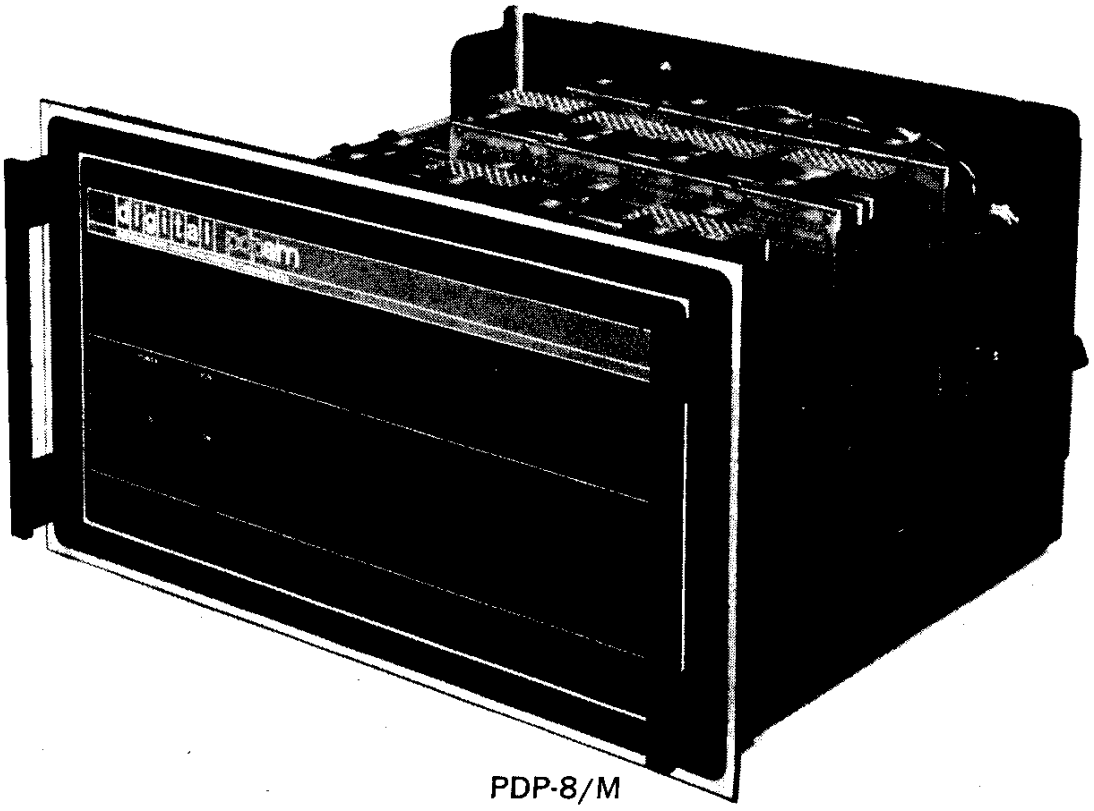
computers are capable of operating in conjunction with a number of optional devices (such as high-speed perforated-tape punch and reader equipment, card reader equipment, line printers, analog-to-digital converters, cathode ray tube (CRT) displays, magnetic tape equipment, a 32,764-word random-access disk file, a 262,112-word random-access disk file, etc.).

The PDP-8/E system is completely self-contained, and requires no special power sources or environmental conditions. A single source of 115V or 230V at 47 to 63 Hz, single-phase power is required. Internal power supplies produce the necessary operating voltages for the system.

The user has a choice of two basic configurations. Table Top or Rack Mountable. Standard DEC cabinets are available to accommodate those users desiring many peripherals. The Table Top version is a convenient approach for those desiring to use the processor in a small area, such as an office.

SECTION 2 THE PDP-8/M OEM PROCESSOR

DEC has recently introduced the OEM version of the PDP-8/E called the PDP-8/M. Because the OEM version is functionally the same processor as the PDP-8/E, all chapters contained in this handbook apply to the PDP-8/M as well as the PDP-8/E with the following exceptions:



PDP-8/M

General Description

The PDP-8/M is a 12-bit parallel computer with identical performance as the PDP-8/E. The PDP-8/M contains one OMNIBUS which defines the maximum basic system configuration. Expansion of the system to three (3) OMNIBUSes is possible.

The Basic PDP-8/M consists of the following components:

- KK8-E Central Processor—same as used in the PDP-8/E.
- MM8-E 4K Memory—same as the PDP-8/E.
- OMNIBUS—The PDP-8/M contains only one OMNIBUS (20 slots).
- The new H740 power supply is sufficient to drive one fully expanded OMNIBUS:

	+5V	−15V	+15V	# of Slots
BASIC 8/M-MC Options Permitted	6.6A	3.3A	0.6A	8
	10.4A	1.7A	0.4A	12
Total Available	17.0A	5.0A	1.0A	20

- The power switch can turn on the PDP-8/M directly, or can operate a remote power control, or both.
- Front Panel—The PDP-8/M offers two front panels corresponding to the two models offered.

The PDP-8/M-MC includes the KC8-M Operators Panel and 4K memory. This panel contains a Power On switch, a Power On indicator, a SW switch (for MI8-E Bootstrap Loader) and a RUN indicator. The indicators are solid state light emitting diodes.

The PDP-8/M-DC includes the KC8-ML Programmers Panel and 4K Memory. This panel is similar to, and has all the features (lights and switches) of the KC8-EA panel standard on the PDP-8/E. All lights, however, are solid state LED's. This panel is also offered as an option.

NOTE

There is no way to "initialize" the PDP-8/M with the KC8/M Operators Panel. An optional KL8-ML Programmers Panel, or a MI8-E Bootstrap Loader, or KP8-EA Power Fail and Auto Restart option, or customer defined loader is required.

All PDP-8/E options are applicable.

SECTION 3 PDP-8/E COMPUTER ORGANIZATION

The PDP-8/E system consists of a central processor, core memory, and input/output equipment facilities; all of which interrelate by means of a common bus called "OMNIBUS."

All arithmetic, logic, and system control operations are performed by the central processor. 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 of the core memory are performed by registers in the central processor, and the operation of the core memory is under control of timing signals produced by the central processor. Because of the close relationship of operations performed by the central processor and the core memory, both are described in this chapter.

Central processor interface circuits provide bussed connections to a variety of peripheral input/output equipment. Each input/output device is responsible for detecting its own select code and for providing all required input or output gating. Individually programmed data transfers between the central processor and peripheral equipment take place through the central processor accumulator. Data break transfers can be initiated by peripheral equipment, rather than under program control, through the data break facilities. Standard features of the computer allow peripheral equipment to perform certain control functions, i.e., instruction skipping and a transfer of program control initiated by a program interrupt.

Standard equipment provided with each system includes a console teleprinter control, which drives and controls a Teletype Model 33 Automatic Send-Receive (ASR). The ASR set is a standard machine operating from serial 11-unit code characters at a rate of 10 characters per second. The ASR provides a means of supplying data to the computer from keyboard or perforated tape; and supplies data as output from the computer in the form of typed copy, or typed copy and perforated tape. 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 other input/output equipment options are discussed in Chapter 7 of this handbook.

The basic system is illustrated in Figure 1-1. It contains ten PDP-8/E FLIP-CHIP modules called: Major register (M8300); Register Control (M8310); Bus-Loads (M8320); Timing Generator (M8330); Panel type KE8-EA; Teletype control (M8560); XY Driver and Current Source (G227); Memory Stack (H220); Sense/Inhibit (G104). (The last three modules are memory system modules.) and RFI Shield (M849).

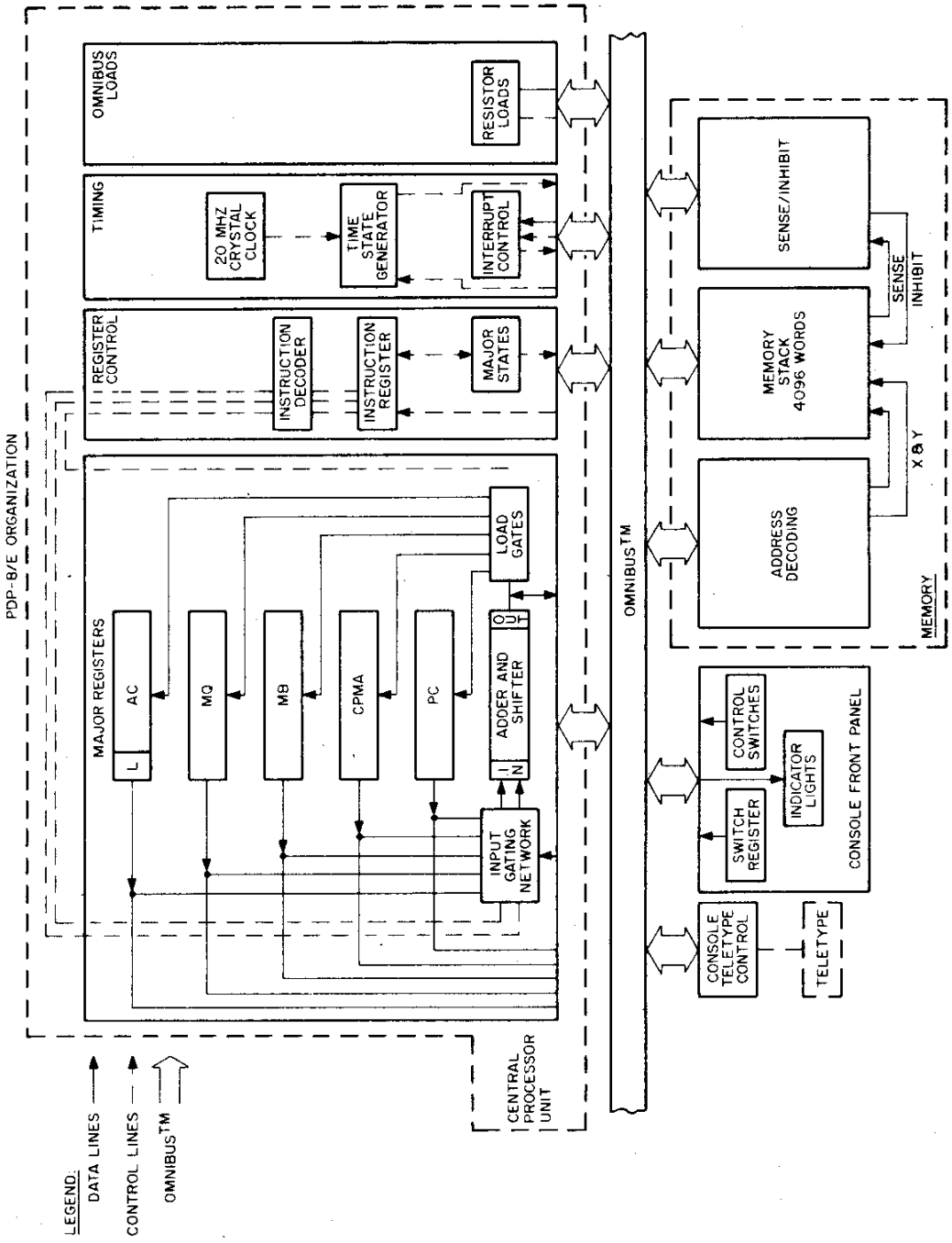


Figure 1-1 PDP-8/E Basic System Block Diagram

MAJOR REGISTERS (M8300)

In order to store, retrieve, control, and modify information and to perform the required logical, arithmetic, and data processing operations, the core memory and the central processor employ the logic complement and major registers shown in Figure 1-1 and described in the following paragraphs.

Accumulator (AC)

The AC is a 12-bit register in which arithmetic and logic operations are performed. Under program control the AC can be cleared or complemented, or its contents can be rotated right or left. The contents of the memory buffer register can be added to the contents of the AC (via the adder circuit), and the result stored in the AC. The contents of both of these registers can be combined by the logical AND operation with the result remaining in the AC. The inclusive OR may be performed between the AC and the switch register (on the programmer's console), and the result left in the AC. The AC also serves as an input/output register; all programmed information transfers between the core memory and an I/O device are passed through the AC to data lines located on the OMNIBUS.

Multiplier Quotient (MQ) Register

The MQ is a 12-bit bidirectional shift register that acts as an extension of the AC during EAE operations. The MQ contains the multiplier at the beginning of a multiplication and the least significant half of the product at the conclusion. The MQ contains the least significant half of the dividend at the start of a division and the quotient at the end. The MQ contains the least significant part of a number during a shift or a normalize operation. The MQ is also available as a temporary storage register adding additional capability and flexibility for the programmer.

Program Counter (PC)

The PC is a 12-bit register that is used to control the program sequence; that is, the order in which instructions are performed is determined by the PC. The PC contains the address of the core memory location from which the next instruction is taken. Information enters the PC from the core memory via the Memory Buffer and from the Memory Address Register. Information in the PC is transferred into the Memory Address register to determine the core memory address from which each instruction is taken. Incrementing the contents of the PC establishes the successive program core memory locations and provides skipping of an Instruction based upon a programmed test of information or conditions.

Central Processor Memory Address (CPMA) Register

The CPMA register is a 12-bit register that contains the address in core memory that is currently selected for reading or writing. Therefore, all 4096 words of core memory can be addressed directly by the CPMA. Data can be transferred into the CPMA from the Memory Buffer, from the Program Counter and from the switch register on the operator's console. Memory Addressing is also accomplished by each data break interface (refer to Chapter 6). This register is never cleared. New information is always jam transferred in and the original content is lost.

Memory Buffer (MB) Register

The MB register is a 12-bit register that is used for all information

transfers between the central processor registers and the core memory. Information can be transferred and temporarily held in the MB from the AC or PC. Also, the MB can simultaneously be loaded and incremented by one before being read back into memory. Information can be loaded into the MB from an I/O device during a data break or from core memory. Information is read from a memory location in 0.6 microseconds and re-written in the same location in another 0.6 microsecond of a single 1.2-microsecond duration memory cycle. Many machine cycles require modification of memory data. In such cycles, an extra 0.2 microsecond is inserted between read and write.

Data Gates and Adders

The Major Registers module also contains the gating necessary to move data from one register to another. At the heart of the data gating is a 12-bit parallel adder. Information from a register is gated to the adder inputs. The output of the adder is applied to a set of shift gates. The output of the shift gates serves as data input to all of the major registers.

REGISTER CONTROLS (M8310)

The Register Control module contains the Link, the Major Register Control circuits, the Major States register, the Instruction register, and the necessary control circuits for the Major Sates register and the Instruction register.

Link (L)

The Link is a 1-bit register that is used to extend the arithmetic facilities of the AC. It is used as the carry register for 2s 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 routine. Under program control, the L may be either cleared, complemented, or rotated as part of the AC.

Major Register Control Circuits

The Major Register Control Circuits enable the adder input and shift gates of the Major Register module. They also gate time pulses to cause loading of the appropriate major register.

Major State Register

The Major State register is the control for the three major states of the PDP-8/E. Conditional inputs, consisting of processor instructions combined with the output of the Major State register, determine which of the three major states (FETCH, DEFER, or EXECUTE) the processor is about to enter. Each of the major state signals, when asserted, is used to enable the corresponding register control circuitry.

Instruction Register (IR)

The IR is a 3-bit register that 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 during a fetch cycle. The contents of the IR are decoded to produce the eight basic instructions and affect the cycles and states entered during each step of the program.

BUS LOADS (M8320)

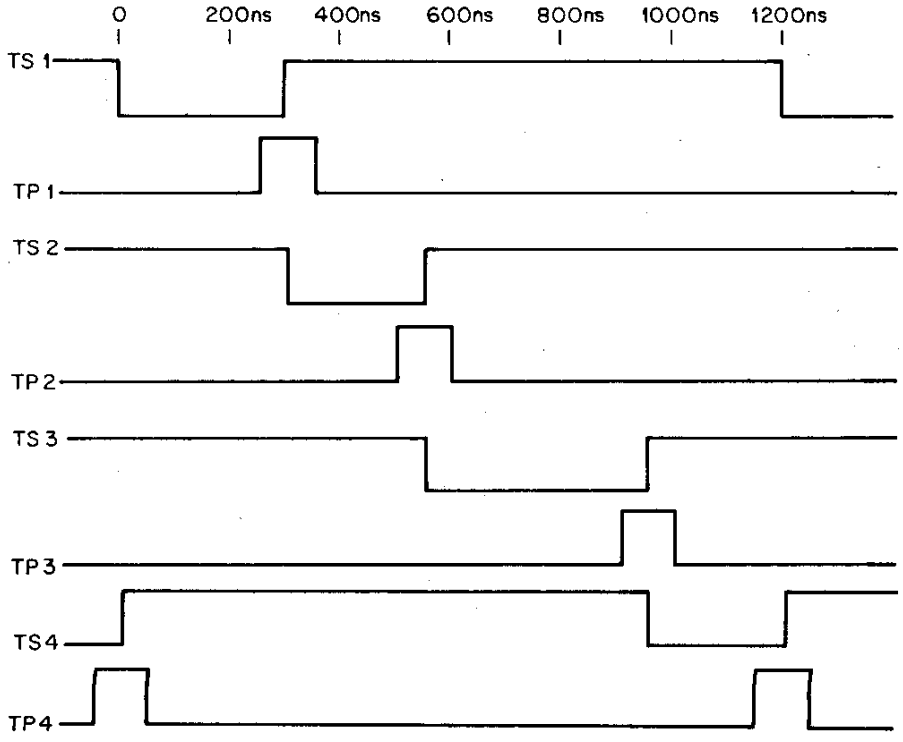
The Bus Loads module contains all of the necessary load resistors required to maintain a high inactive level for each of the busses in the system. There are basically seven groups of signals that the Bus Loads module services. These are: 1. Memory Address (MA); 2. Memory Data (MD); 3. Data; 4. I/O Control; 5. Break Control; 6. Timing; and 7. Miscellaneous Signals. Most lines are considered by the system to be inactive (voltage level high) until the line level is pulled to ground by some component connecting to the corresponding signal line.

TIMING GENERATOR (M8330)

The Timing Generator module contains the time pulse generator, Interrupt Control circuits, the Processor IOT Decoder, and miscellaneous control circuits.

The time pulse generator provides the timing pulses that determine the computer cycle time and are used to initiate sequential time-synchronized gating operations. 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 for peripheral device control to be utilized with devices using programmed or data break information transfers.

Four time states, TS1 through TS4, are provided by the time pulse generator. In addition, four time pulses are generated for use as gating pulses throughout the system. Each time pulse overlaps the end of one time state into the beginning of the next time state (refer to Figure 1-2). Memory timing is also provided by the time pulse generator.



MEMORY TIMING CYCLE (FAST CYCLE SHOWN)

The Interrupt Control circuits comprise the major portion of the Interrupt System. The circuitry responds whenever an INTERRUPT REQUEST signal is received from an interface controller module.

The Processor IOT Decoder decodes the last 9 memory data bits and determines the type of IOT instruction that is to be performed.

PROGRAMMER'S CONSOLE (KE8-EA)
The Programmer's Console contains a convenient array of controls and indicators that are used specifically for operation and maintenance. The Console has been configured to achieve convenient control of the system. Through switches and keys on the Console, the operator can STOP, START, EXAMINE, MODIFY, or CONTINUE a program. The indicators, when properly selected, display the machine status and contents of major registers.

A lighted indicator denotes the presence of a Binary 1 in a specific register bit position or control flip-flop.

The Programmer's Console contains a 12-bit switch register, ten control switches, and an indicator selector switch capable of selecting seven individual major states and status registers to be displayed on a 28-lamp indicator panel.

TELETYPE CONTROL (M8350)
The Teletype Control Module contains the necessary receive and transmit circuitry along with the control circuitry to interface the ASR 33 Teletype terminal with the processor.

PDP-8/E MEMORY SYSTEM (MM8-E)
The basic PDP-8/E memory system (MM8-E) is a 4096 word, 12-bit random access core memory that performs all normal functions of data storage and retrieval. The same basic 4K memory, consisting of 3 quad modules, can be used as an extended memory to increase the memory capacity up to the addressing capability (32K) of the PDP-8/E.

Memory location 0 is used to store the contents of the PC following a program interrupt, and location 1 is used to store the first instruction to be executed following a program interrupt. When a program interrupt occurs, the contents of the PC are stored in location 0 and program control is transferred to location 1 automatically. Core memory locations 10 (octal) through 17 (octal) are used for auto-indexing. All other locations can be used for the storage of either instructions or data.

The memory system contains circuits such as read/write switches, address decoders, inhibit drivers, and sense amplifiers. These circuits perform the electrical conversions necessary to transfer information to or from the core array. They perform no arithmetic or logic operations upon the data.

The XY Driver & Current Source (G227)
This PDP-8/E module contains the circuitry required to decode the address lines and drive the XY wires of a 4096 word core memory (i.e., address decoding, selection switches, XY current sources, stack discharge switch, and power on/off write protection). The XY currents are

controlled remotely by a control on the Sense/Inhibit board. The XY Driver and Current Source module requires no adjustments. The same module is used also in the memory parity option.

Memory Stack (H220)

The memory cores are mounted on a G619 Planar Stack Board. The whole module assembly is the H220 Stack. It contains 4096 words of 12-bit core memory and the X-axis and Y axis diode selection matrix. It also includes a resistor/thermistor combination that supplies temperature information to the XY current control. The core memory is a 3D/3-wire memory with center tapped sense/inhibit wire. This module has no connections from and to the OMNIBUS. The same stack is also used in the memory parity option.

Sense/Inhibit Module (G104)

The Sense/Inhibit module (G104) is a PDP-8/E module containing the sense amplifiers, memory register, and the inhibit drivers for a word length of 12 bits. It also includes the slice control and the —6V supply for the sense amplifiers, the current control for the XY current source, control logic for the strobe and clear, and the field select, which is used in the Sense/Inhibit as well as in the XY Driver. Three jumper connections determine the field. Slice level, strobe delay, and XY current can be selected within four discrete steps by appropriate jumper connections (two per axis). With a given stack the proper combination is known and the jumper connection can be selected. Adjustments in a system are, therefore, eliminated.

The memory parity option, consisting of another 3 modules, adds all the circuitry necessary to read, write, and store the parity for 32K of memory. Additional memory options are a 256-word Read Only Memory, a 1024-word Read Only Memory, a 256-word Read/Write Memory, and a Bootstrap Loader Read Only Memory. Refer to Chapter 7 for the description of each memory option.

MAJOR PROCESSOR STATES

The PDP-8/E utilizes three processor states to execute programmed instructions. To accomplish this, a major state generator is used to establish one state for each computer timing cycle. The major processor states are: FETCH, DEFER and EXECUTE. FETCH, DEFER and EXECUTE states determine and execute instructions.

Fetch (F) State

The computer enters the FETCH state to obtain a 12-bit instruction word. At the start of FETCH cycle, the contents of the PC are loaded into the CPMA, giving the first memory address and starting the memory cycle.

At the start of the FETCH cycle, the computer obtains the contents of an addressed memory location and places the 12 bits on the Memory Data lines of the OMNIBUS. The contents of these lines are decoded to determine the kind of instruction the processor must next perform. Once the processor decides the kind of instruction it must do, it then begins performing the instruction, entering a DEFER, or an EXECUTE state as required. When the instruction has been completely performed, the computer again enters the FETCH state to obtain the next instruction.

Assuming the computer is a fully automatic, running condition; that is, the computer is continuously functioning to FETCH and/or EXECUTE instructions, the PC register's contents are transferred to the MA register at TP4 time, initiating the FETCH state. When this is accomplished, the MA is simultaneously incremented by one and the result is transferred to the PC register. Sometime during TS2, the memory is strobed. The contents of the memory appear at the output of the sense amplifiers and are transferred to the MD lines. The contents of the first three bits of the memory data are transferred to the IR. This completes the activity during time TS2 of the FETCH state.

If the instruction is a multicycle (2 or 3) instruction, the memory address is computed, but no further action takes place until the Defer or Execute cycle. If, however, the instruction is a single cycle instruction (such as IOT, OPR, or JMP and bit 3 = 0) the instruction is carried out immediately. The Major State register gating causes the computer to enter the appropriate major state at the end of the cycle.

Defer (D) State
At the end of a FETCH cycle, the current instruction is directed to DEFER whenever indirect addressing is required.

The DEFER state is entered if a binary 1 is present in bit 3 of a memory reference instruction. This state causes the central processor to obtain the full 12-bit address of the operand from an address in either the current page or page zero, as specified by bits 4 through 11 of the instruction. This process of address deferring is called indirect addressing, because access to the operand is addressed indirectly, or deferred, to another memory location.

Execute (E) State
The EXECUTE state is entered for all memory reference instructions except JMP. During an AND, 2s complement add, or increment and skip if zero instruction, the contents of the core memory location specified by the address portion of the instruction are read into the MB and the operation specified by the Instruction Register is performed. During a deposit and clear accumulator instruction, the contents of the AC are transferred into the MB and stored in core memory at the address specified in the instruction. During a jump to a subroutine instruction, the EXECUTE state occurs to write the contents of the PC into the core memory location as designated by the instruction and to transfer this address +1 into the PC to bring about a change in program control.

In addition to the three major states described, the processor responds to other functions such as Data Break. During this time period, FETCH, DEFER and EXECUTE states are held inactive.

Direct Memory Access (DMA) State
A fourth state exists when any one of the other three major states is not enabled. This state, called DMA, is used to independently address memory and to store or read out information without the aid of processor instructions. DMA is used at the Programmer's Console when information is added to or taken from memory. Data Break devices also use the DMA state to perform block transfers.

INTERFACING

The PDP-8/E offers two approaches to interfacing with peripheral equipment. The OMNIBUS is an internal Input/Output Bus on which all I/O data and control signals are transferred. A variety of peripherals can interconnect through a peripheral control module from this bus. The OMNIBUS has eliminated wires by providing an etched circuit board on which connectors are mounted. For the convenience of the user, each pin assignment on one connector is identical to the next connector, allowing a module to be placed anywhere on the bus. The PDP-8/E options provide a complete line of peripherals used in most computer operations. However, for those requirements that are not covered by options, Chapter 9 provides the information needed so that a user can build his own interface. Special interface modules can also be constructed by DEC.

An External Bus is the second approach to interfacing to peripherals. The External Bus connects to the OMNIBUS and provides an extension to the bus system for users who already possess or desire to use PDP-8/I or PDP-8/L compatible peripherals. The interfacing details are provided in Chapter 10 of this handbook.

Three types of data transfer systems are available to the user. One system is the straight Input/Output transfer, which is the simplest and most direct type of transfer. The Program Interrupt system is a type of data transfer system useful for installations having more than one peripheral. For installations using extremely fast transfer rates, the data break system (sometimes called Direct Memory Access, or DMA) is available (refer to Chapter 6 for details on the Data Break system and Chapter 5 for details on programmed transfers).

DIFFERENCES BETWEEN PDP-8/E AND ITS PREDECESSORS

As a new computer is developed, differences between it and its predecessors inevitably result. In many cases these differences are either benign or beneficial. All differences are listed below in order to give users of earlier machines a concise summary.

Instruction Differences

NEW INSTRUCTION	OCTAL	PREVIOUS FUNCTIONS
Byte Swap (BSW)	7002	Rotate 2, no direction (no operation)
Swap MQ and AC (SWP)	7521	MQL MQA (worked as a SWP in KE8-I, but not documented)
Reserved for future expansion	7014	RAR RAL*
Reserved for future expansion	7016	RTR RTL*
MQ instructions	74X1	Only available with EAE on previous machines; otherwise produced a NOP.
Skip if interrupt on (SKON)	6000	No operation
Skip on interrupt request (SRQ)	6003	(ION)

NEW INSTRUCTION	OCTAL	PREVIOUS FUNCTIONS
Get flags (GTF)	6004	No operation, or ADC in PDP-8 with 189 A/D Converter
Restore flags (RTF)	6005	(ION) (ORed with ADC)
Skip if Greater Than (SGT)	6006	(IOF) (ORed with ADC)
Clear all flags (CAF)	6007	(ION) (ORed with ADC)

* These instructions produced predictable but undocumented results in PDP-8/I and PDP-8/L. They may have been used by some programmers to load the constants 3776 and 5775 into the AC. In the PDP-8/E, these codes are specifically reserved for future expansion, and should not be executed.

TTY Differences

The console TTY uses the same IOT's as in earlier machines, but also has added IOTs to enable or disable TTY flags onto the interrupt bus. Also included are additional IOTs to clear keyboard flags without advancing the reader and an IOT to set the printer flag. The skip IOT's can no longer be microprogrammed with the other IOTs of the same device code. This should impose no constraint on the user, since skips are generally not combined with other IOT's. Reader Run is no longer set by INITIALIZE. Hence any routines using the reader must begin with a KCC instruction.

External I/O Bus

In general, the signals and functions at the External I/O Bus Interface are the same as for previous machines. Users who constructed peripherals using previous editions of the Small Computer Handbook as a guide may expect their peripherals to work on the PDP-8/E. (Please note, however, that PDP-8/E is equipped only with a positive bus; and a DW08A bus converter is necessary to interface to older, negative bus equipment.)

The BAC lines at the External I/O Bus interface are merely the buffered DATA bits of the OMNIBUS. Since the DATA lines are used for bi-directional transfer, any input of data at the External I/O Bus interface will cause an immediate change at the BAC outputs. Simultaneous input and output transfers in the same IOP should be checked. In such situations, the register in the peripheral must be edge-triggered.

Also, at the conclusion of the IOP dialogue, the DATA bus is used for updating the PC, and then for determining break or API priority. Users can no longer rely on the BAC lines being available until the end of the major state. However, the IOP width and separation may be adjusted if desired, to accommodate slow I/O devices. External IOTs are faster in all cases except for IOTs ending in 7.

EAE

The Extended Arithmetic Element has been redesigned, and several powerful features have been added. Previous EAE users may use the EAE without modifying their programs, but it is a wise move to recode the EAE programs in order to make use of the new SAM, DCM, DAD, DST, DPIC and DPSZ instructions.

Data Break

The time required to access the Data Break system has been greatly reduced. Maximum benefit can be obtained on machines without EAE, and with only internal options (or options which are not activated while Data Break is in use). An added feature, ADM, allows the user to add an input word to the contents of a memory location. An internal multiplexing scheme allows the use of several (12 max.) external and/or internal break devices.

Control Panel

The control panel of PDP-8/E differs from panels of its predecessors as follows:

1. Only the MA (and EMA) and the RUN status are permanently displayed. All other registers are selected by a rotary switch.
2. Machine stops occur after TP4. Thus the MA lights indicate the next address to be accessed. Similarly, the Major State indicators show the next major state to be executed.
3. Extended field information is loaded via SR6-11 and the EXTD ADDR LOAD switch.
4. Operation of the ADDR LOAD switch places the CP in the FETCH state.
5. Programs are started by operating and releasing the CLEAR key, then operating and releasing the CONT key.
6. Turning the Power Switch to the PANEL LOCK position extinguishes all indicators except the RUN light.

ADDRESSING NONEXISTING CORE

An attempt to deposit data in a non-existant memory field will not stop the machine. An attempt to read data from a non-existant memory yields a zero operand. A jump to a non-existant memory will, of course, "hang up" the program, since there is then no way to jump back to existant memory.

CHAPTER 2

STANDARD SYSTEM OPERATION

GENERAL

The PDP-8/E Computer allows the operator to manually program the machine using the switch register located on the Programmer's Console or use the more automatic process with the ASR 33 Teletype Console which contains the Tape Reader/Punch combination as well as the standard Teletype keyboard. This chapter defines the operation of communicating with the processor in both the manual and program control modes.

The user should be thoroughly acquainted with the content of chapters 3 and 4 before operating the system.

CONTROLS AND INDICATORS

The controls and indicators on the Programmer's Console provide manual control and indicate the program conditions of the PDP-8/E. Controls on the Programmer's Console provide the operator with the hardware to start, stop, examine, modify, or continue a program. The indicators on the console provide a visual indication of the machine status and current program, the contents of the major registers, and the condition of the control flip-flops. A lighted indicator denotes the presence of a binary 1 in a specific register bit position or control flip-flop. Table 2-1 lists the functions of controls and indicators. The controls are divided into two groups; switches and keys. Keys are momentary, or spring-return, switches. Figure 2-1 illustrates the console. Controls and indicators of the standard Model 33 ASR Teletype unit are shown in Figure 2-2 and their functions are described in Table 2-2.

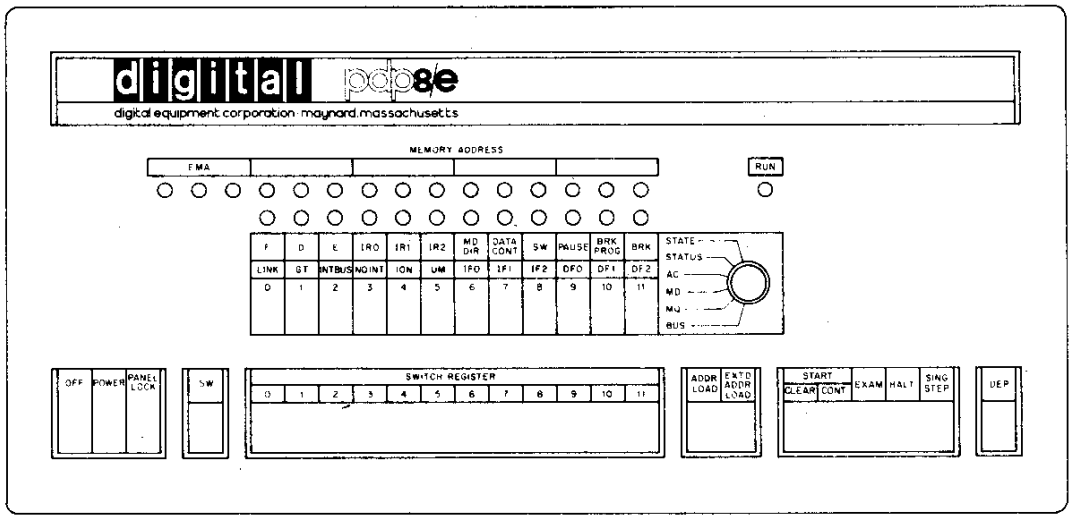


Figure 2-1 Programmer's Console

Table 2-1

Programmer's Console Control and Indicator Functions

CONTROL OR INDICATOR	FUNCTION
Off/Power/Panel Lock	This is a key operated switch. In the counter-clockwise, or OFF, position, the switch disconnects all primary power to the machine. In the POWER, or straight up position, it enables all manual controls and applies primary computer power. In the PANEL LOCK or clockwise position, it disables all keys and switches with the exception of the switch register and mode switch. In this position, a running program is protected from inadvertent switch operation and all panel indicators except the RUN light are turned off.
SW	When this switch is up, the line on the OMNIBUS line called SW is high; when the lever is down, the line is low. This switch is used by special peripheral controls, such as the Bootstrap Loader.
Switch Register Switches (SR)	These 12 switches provide a means of communication between operator and machine. They allow a 12-bit word to be input. When the switch is up it designates a binary 1 to the machine; switch down is a 0 (zero). These switches are used during manual functions or under program control.
Load Address Key (ADDR LOAD)	This key loads the contents of the Switch Register into the CPMA and forces Fetch to be set (no Major States while the Load Address Key is depressed).
Extended Address Load Key (EXTD ADDR LOAD)	This switch loads the contents of SR6-11 into the Data Field and Instruction Field registers of the Memory Extension Control. SR9-11 goes to Data Field 0-2. SR6-8 goes to Instruction Field 0-2.
Clear Key (CLEAR)	This key issues an Initialize Pulse, clearing the AC, LINK, Interrupt system, and I/O Flags.

Table 2-1 (Cont.)

CONTROL OR INDICATOR	FUNCTION
Continue Key (CONT)	This key resumes the computer program by issuing a Memory Start and setting the Run Flip-Flop. The word stored at the address currently held by the CPMA is taken as the first instruction.
Examine Key (EXAM)	Puts the contents of core memory at the address specified by the contents of the CPMA into the MB. Then the contents of the PC and CPMA are incremented by one to allow examination of the contents of sequential core memory addresses by repeating the operation of the examine switch.
Halt Switch (HALT)	This switch clears the Run flip-flop and causes the machine to stop at TS1 of the next Fetch cycle. This switch is also used for single instruction stepping.
Single Step Switch (SING STEP)	This switch clears the Run flip-flop and causes the machine to stop at TS1 of the next cycle. Thereafter, repeated depressing of the continue key steps the program one cycle at a time, so that the contents of registers can be observed in each state.
Deposit Key (DEP)	Loads the contents of the SR into the MB and core memory at the address given by the current contents of the CPMA. Then the contents of the PC and CPMA are incremented by one. This allows storing of information in sequential memory address by repeated operation of the deposit switch.
Indicator Selector Switch	This is a six-position rotary switch, used to select a register for display. The six positions are as follows: 1. STATE — Indicates an individual function for each bit; Bit 0—Fetch 1—Defer 2—Execute 3—IR 0 4—IR 1 5—IR 2

Table 2-1 (Cont.)

CONTROL OR INDICATOR	FUNCTION
	6—MD DIR 7—Data Control 8—SW 9—Pause 10—Break in Prog 11—Break
	2. STATUS — Indicates an individual function for each bit; Bit 0—Link 1—Greater Than Flag 2—Interrupt Bus 3—No Interrupt Allowed 4—Interrupt On 5—User Mode 6—Instruction Field 0 7—Instruction Field 1 8—Instruction Field 2 9—Data Field 0 10—Data Field 1 11—Data Field 2
Indicator Selector Switch	3. AC — Indicates bits 0-11 of the accumulator at TS1. 4. MD—Indicates Information just written or rewritten into memory. 5. MQ—Indicates contents of MQ register during TS1. 6. BUS—Indicates bits 0-11 of the DATA Lines.
Memory Address	Indicates the contents of the memory address which will be accessed next.
EMA	Indicates which Extended Memory field is being accessed.
Run Light	When lit means machine's timing is enabled and capable of executing instructions.

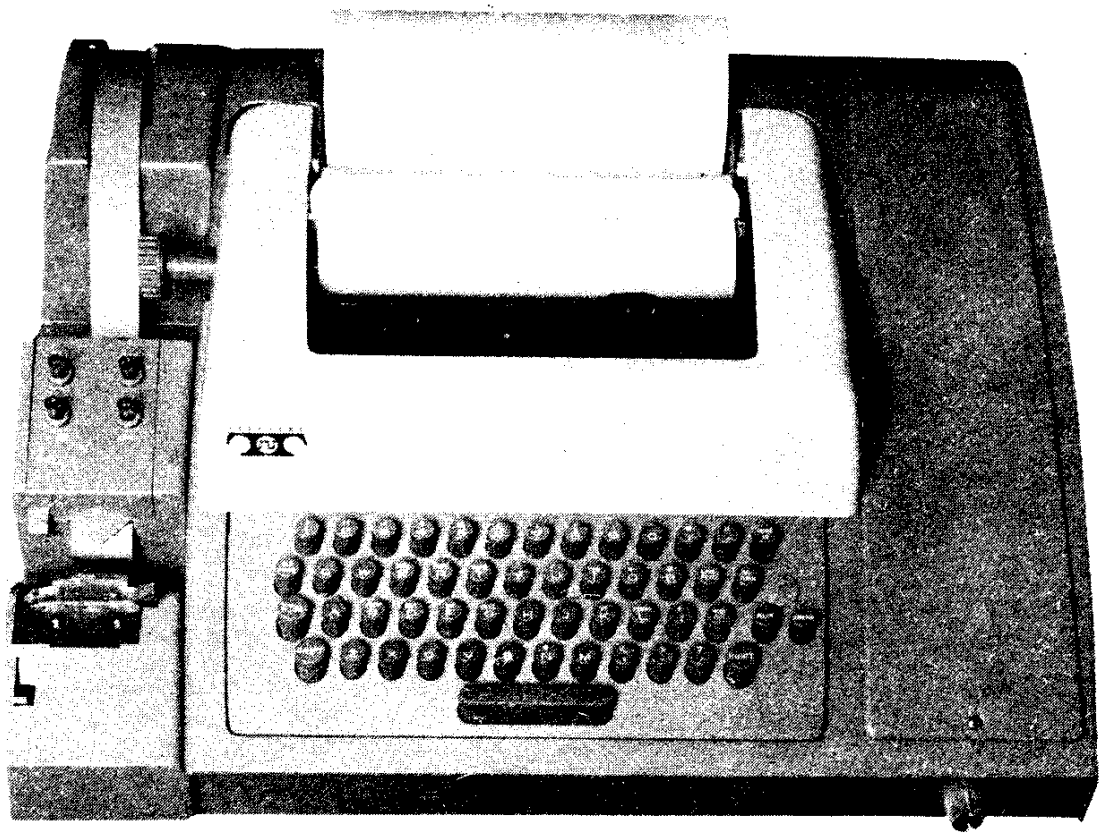


Figure 2-2 Teletype Model ASR33 Console

Table 2-2

ASR33 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 rubout of the character just punched.
OFF/ON Pushbuttons	Controls 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 FREE position the reader is disengaged, permitting the paper tape to be manually moved within the reader without necessarily reloading or unloading it. In the STOP position the reader mechanism is

Table 2-2 (Cont.)

CONTROL OR INDICATOR	FUNCTION
	engaged but de-energized. In the START position the reader is engaged and operated under program control. The tape may be loaded or unloaded in either the FREE or STOP positions.
Keyboard	Provides a means of printing on paper when used as a typewriter and punching tape when the punch ON pushbutton is pressed; also 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 to the Teletype and data connection to the processor. In the LINE position, the Teletype is energized and connected as a computer I/O device. 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 disconnected. Both LINE and LOCAL use of the Teletype require that the computer be energized through the POWER switch.

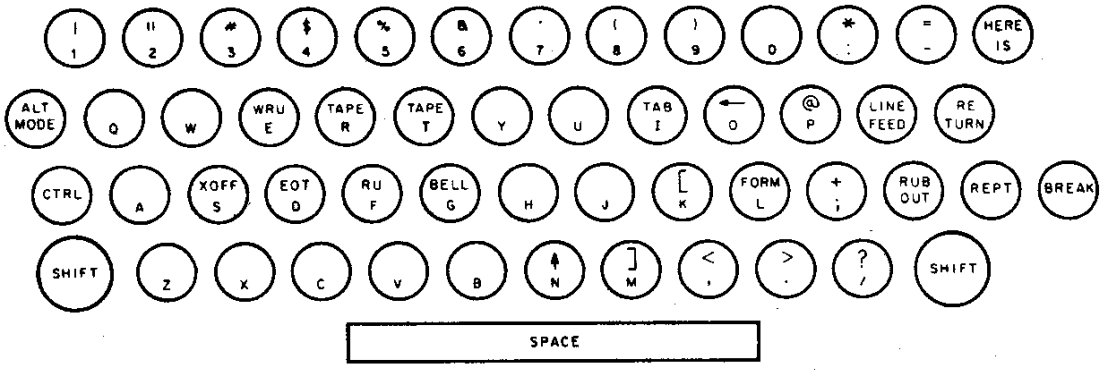


Figure 2-3 Teletype Keyboard

KEYBOARD OPERATION

The Teletype keyboard shown in Figure 2-3 is similar to a typewriter keyboard, except that some nonprinting characters are included as upper case elements. For typing characters or symbols, such as $, %, #, which appear on the upper portion of numeric keys and certain

alphabetic keys, the SHIFT key is held depressed while the desired key is operated.

Designations for certain nonprinting functions are shown on the upper part of some alphabetic keys. By holding the CTRL (control) key depressed and then depressing the desired key, these functions are activated. Table 2-3 lists several commonly used keys that have special functions in the symbolic language of PDP-8/E computers.

Table 2-3

Special Keyboard Functions

KEY	FUNCTION	USE
SPACE	space	used to combine and delimit symbols or numbers in a symbolic program
RETURN	carriage return	used to terminate line of symbolic program
HERE IS	blank tape	used for leader/trailer (effective only in LOCAL)
RUBOUT	rubout	used for deleting characters, punches all channels on paper tape
CTRL/REPT/P	code 200	used for leader/trailer of binary program paper tapes (keys must be released in reverse order: P, REPT, CTRL)
LINE FEED	line feed	follows carriage return to advance printer one line

PRINTER OPERATION

The printer provides a typed copy of input and output at ten characters per second maximum rate. When the Teletype unit is on line (LINE), the copy is generated by the computer; when the Teletype unit is off line (LOCAL), the copy is automatically generated whenever a key is struck.

PAPER TAPE READER OPERATION

The paper tape reader is used to input into memory data punched on eight-channel perforated paper tape at a maximum rate of ten characters per second. The reader control positions are shown in Figure 2-2 and are described below.

START — Activates the reader; reader sprocket wheel is engaged and operative.

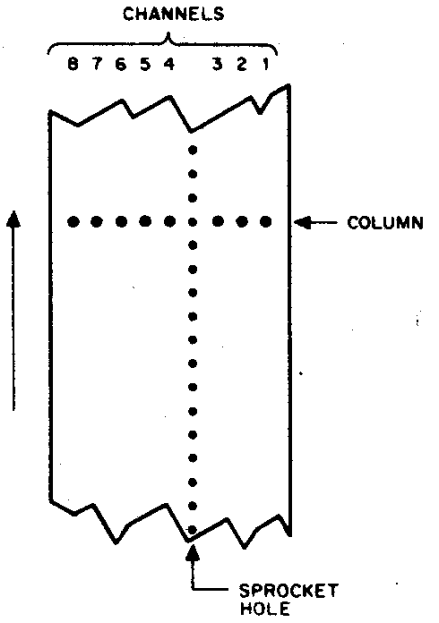
STOP — Deactivates the reader; reader sprocket wheel is engaged but not operative.

FREE — Deactivates the reader; reader sprocket wheel is disengaged.

PAPER TAPE PUNCH OPERATION

The paper tape punch is used to perforate eight-channel rolled oiled paper tape at a maximum rate of ten characters per second. The punch controls are shown in Figure 2-2 and described below.

REL.	Disengages the tape to allow tape removal or loading.
B. SP.	Backspaces the tape one space for each firm depression of the B. SP. button.
ON	Activates the paper tape punch.
OFF	Deactivates the paper tape punch.



Data is recorded (punched) on paper tape by groups of holes arranged in a definite format along the length of the tape. The tape is divided into channels, which run the length of the tape, and into columns, which extend across the width of the tape, as shown in the adjacent diagram. The paper tape readers and punches used with PDP-8 family computers accept eight-channel paper tape.

Generating a Symbolic Tape

The previously described components may be used to generate a symbolic program paper tape through the following procedure.

When switched to LOCAL, the Teletype unit is independent of the computer and functions like an electric typewriter. Any character struck on the keyboard is printed, and also punched on paper tape if the tape punch is ON. Each character struck on the keyboard is represented in code by one row of holes and spaces according to the ASCII code described in the following section and given in Appendix C.

A section of leader-trailer code several inches long is punched at the beginning of the symbolic tape, by pressing the HERE IS key on the

Teletype keyboard. The symbolic program is then carefully typed, following the conventions used in PDP-8 symbolic programs.

A typing error can be corrected using the B. SP. button of the paper tape punch and the RUBOUT key on the Teletype keyboard. The B. SP. button backspaces the paper tape one column for each depression of the button, and the RUBOUT key perforates all eight channels of a column (this perforation is ignored by the computer). Therefore, errors are removed by backspacing the tape to the error and typing rubouts over the error and all following characters. After typing rubouts, the correct information must be typed beginning where the error occurred.

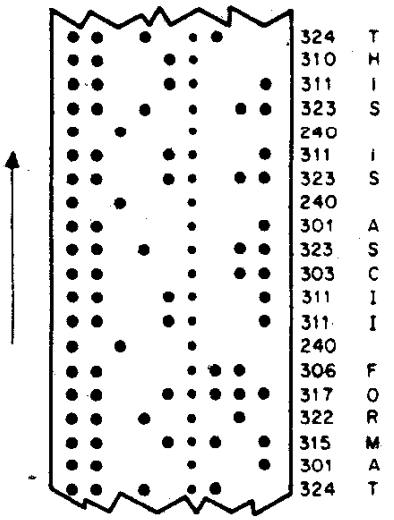
Once the symbolic tape is punched, a form feed is punched, and then more leader-trailer tape is generated by striking the HERE IS key. The tape is removed from the punch unit by tearing against the plastic cover of the punch. The symbolic program thus generated is the input to the assembler described in Chapter 4.

The program may be listed (typed out) by placing the paper tape in the paper tape reader. This is done by releasing the plastic cover of the reader unit and placing the eight-channel tape over the reader head with the smaller sprocket holes over the sprocket wheel, and replacing the cover. If the Teletype control is switched to LOCAL and the reader is switched to START, the tape will advance over the reader head and a printed copy of the program will be typed on the Teletype printer. If the tape punch is also ON, a duplicate of the tape will be generated at the same time.

Paper Tape Formats

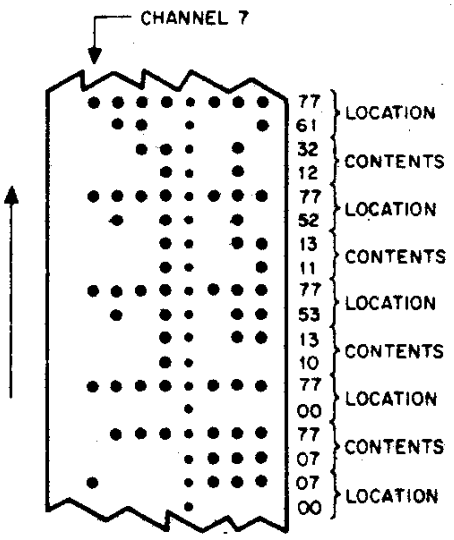
Manual use of the toggle switches on the operator console is a tedious and inefficient means of loading a program. This procedure is necessary in some instances, however, because the PDP-8/E computer must be programmed before any form of input to the memory unit is possible. For example, before any paper tape can be used to input information into the computer, the memory unit must have a stored program which will interpret the paper tape format for the computer. This loader program must be stored in memory with the console switches. A loader program consists of input instructions to accept information from the Teletype paper tape reader and instructions to store the incoming data in the proper memory locations.

Before the loader program can be written to accept information, the format in which the data is represented on the paper tape must be established. There are three basic paper tape formats commonly used in conjunction with PDP-8/E computer. The following paragraphs describe and illustrate these formats.



ASCII FORMAT

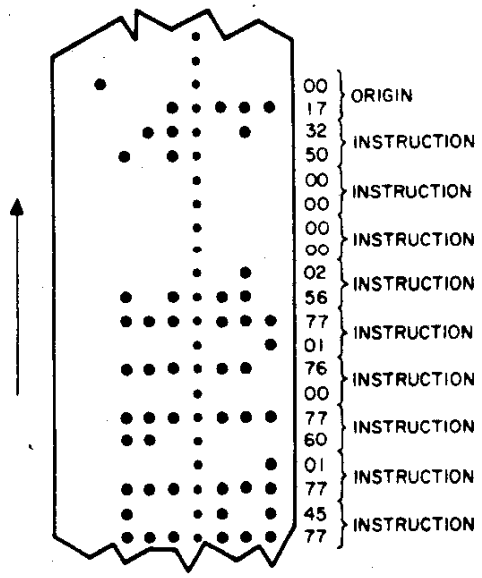
The USA Standard Code for Information Interchange (ASCII) format uses all eight channels* of the paper tape to represent a single character (letter, number, or symbol) as shown in the diagram at left. The complete code is given in Appendix C.



RIM (READ IN MODE) FORMAT

RIM format tape uses adjacent columns to represent 12-bit binary information directly. Channels 1 through 6 are used to represent either address or information to be stored. A channel 7 punch indicates that the adjacent column and the following column are to be interpreted as an address specifying the location in which the information of the following two columns is to be stored. The tape leader and trailer for RIM format tape must be punched in channel 8 only (octal 200).

* Channel 8 is normally designated for parity check. The Teletype units used with the PDP-8/E computer do not generate parity, and Channel 8 is always punched.



BIN (BINARY FORMAT)
BIN format tape is similar to RIM format except that only the first address of consecutive locations is specified. An address is designated by a channel 7 punch and information following an address is stored in sequential locations after the designated address, until another location is specified as an origin. The tape leader/trailer for BIN format tape must be punched in channel 8 (octal 200) only.

Paper Tape Loader Programs

The three previously described paper tape formats are each used for a separate purpose in conjunction with PDP-8/E computer. The ASCII format is used to represent symbolic programs on paper tape, which are then used as input to the assembler. The assembler translates the mnemonic instructions and symbolic addresses into binary instructions and absolute addresses. Once this translation has been performed by the assembler, a binary format tape is generated. ASCII tapes can be quickly recognized by noting that all channels of the tape are used.

The binary format tape is the common means of loading an assembled program into the core memory of a PDP-8/E computer. The BIN (Binary) loader is the program used to load these binary format paper tapes. Program instructions are stored in successive locations beginning with an origin which is signaled by a channel 7 punch on the paper tape. The BIN loader is a lengthy program requiring 83 memory locations. (As an alternative to manually entering the contents of all 83 locations, the RIM (Read In Mode) format is used.) Tapes in binary (BIN) format can be quickly recognized because channel 8 is not used in the middle of the tape, and channel 7 is punched infrequently.

The RIM loader is simpler than the BIN loader because the memory unit is supplied with a location for each incoming instruction. It consists of 17 instructions which must be toggled into memory. The BIN loader is punched in RIM format, and is loaded by the RIM loader; but it is used to load tapes punched in the BIN format, which is the output of the assembly program. Tapes in RIM format are similar in appearance to BIN tapes, but channel 7 is punched every fourth character.

OPERATING PROCEDURES

Many means are available for loading and unloading PDP-8/E information. The means used are dependent upon the form of the information, time limitations, and the peripheral equipment connected to the system. The following procedures are basic to any use of these systems,

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 Programmer's Console. Chief use of manual data storage is made to load the read-in mode (RIM) loader program into the computer core memory. The RIM loader is a program used to automatically load programs into the computer from perforated tape in RIM format. This program and the RIM tape format are described below. Use the following procedure to store data manually in the computer core memory.

Power For Manual Operation

a. Turn the OFF/POWER/PANEL Lock switch clockwise to the POWER position.

Memory Addressing for Manual Operation

a. Set the SWITCH REGISTER switches to correspond to the address bits of the word to be stored.
b. Press the LOAD ADDRESS key.
c. Observe that the address in the switch register is held in the computer 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.

Manual Data Input to Addressed Memory Location

a. Set the SWITCH REGISTER switches to correspond to the data or instruction word to be stored at the address just set into the CPMA.
b. Rotate the INDICATOR SELECTOR SWITCH to MD.
c. Lift and release the DEP key.
d. Observe that the data in the SWITCH REGISTER is the same as the data shown on the MD indicators. Data is now stored in the addressed location.
e. Check to see that the MA has been incremented by one so that additional data can be stored at sequential addresses by repeated SWITCH REGISTER settling and deposit key operation.

Checking the Contents of Any Address in Core Memory

a. Perform the Memory Addressing Procedure.
b. Depress the EXAMINE switch.
c. Rotate the INDICATOR SELECTOR switch to the MD position.
d. Observe the data shown on the MD indicators.
e. To observe the next location in core, the contents of the PC and the CPMA are automatically incremented by one. The operator simply depresses the EXAMINE switch and observes the content of the new location.

LOADING DATA UNDER PROGRAM CONTROL

Information can be stored or modified automatically in the computer only by using programs previously stored in core memory. For example,

having the RIM loader stored in core memory allows RIM format tape to be loaded as follows:

INITIALIZING THE SYSTEM

a. Rotate the OFF/POWER/PANEL LOCK switch clockwise to the POWER position.
b. Set the Teletype LINE/OFF/LOCAL switch to the LINE position.
c. Load the tape in the Teletype reader by settling 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, moving the tape either forward or backward until the punched leader section is over the read station, 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 reader results in 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. The directional arrow printed on the tape should point toward the operator.
d. Load the starting address of the RIM loader program (not the address of the program to be loaded) into the PC by means of the Switch Register and the LOAD ADDRESS switches.
e. In sequence, press the computer keys CLEAR and CONTINUE and set the 3-position Teletype reader switch to the START position. The tape is then read automatically.
f. Stop the computer program by means of the Halt switch when the reader reaches the trailer section of tape.

PROGRAM LOADING OPERATION

Automatic storing of the binary loader (BIN) program is performed by means of the RIM loader program as described below. 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 d. When the BIN program is loaded the computer stops; at this point the AC should contain all zeros if the program is stored properly. If the computer stops with a number other than zero in the AC, a checksum error has been detected. When the program has been stored, it can be initiated by loading the program starting address (usually designated on the leader of the tape) into the PC by means of the Switch Register and LOAD ADDRESS switches, then pressing and releasing the CLEAR key and then pressing the CONTinue key.

The steps involved in the process of loading programs and bringing the system up to the point where the user can communicate with the processor is illustrated in Figure 2-4. The loading flow diagram for each type of program to be loaded is referenced, and each flow diagram is accompanied with a corresponding procedure.

This loading procedure is greatly simplified when the user employs a mass storage device such as the Disk Monitor System. Using the Monitor, stored programs are called in from Disk files. This is illustrated in Figure 2-5.

Loaders

When a PDP-8/E computer is first received, it should be assumed that no useful information is in memory. The machine is not capable of performing any arithmetic operations or receiving data.

All tapes in the Program Library are written in binary format. The user, therefore, must load the machine so that it is capable of accepting binary tapes. Initially, sixteen RIM instructions are manually toggled into memory. The binary loader tape is then used to eliminate the necessity of toggling in 86 additional instructions.

RIM allows the binary loader tape to be read into memory and the binary loader tape allows the use of any tape in the Program Library such as symbolic editor.

READ-IN-MODE (RIM) LOADER

The RIM Loader is the very first program loaded into the computer, and it is loaded by the programmer using the console switches. The RIM Loader instructs the computer to receive and store, in core, data punched on paper tape in RIM coded format. (RIM Loader is used to load the BIN Loader described below.)

There are two RIM loader programs: one is used when the input is to be from the low-speed paper tape reader, and the other is used when input is to be from the high-speed paper tape reader. The locations and corresponding instructions for both loaders are listed in Table 2-4.

The procedure for loading (toggling) the RIM Loader into core is illustrated in Figure 2-6.

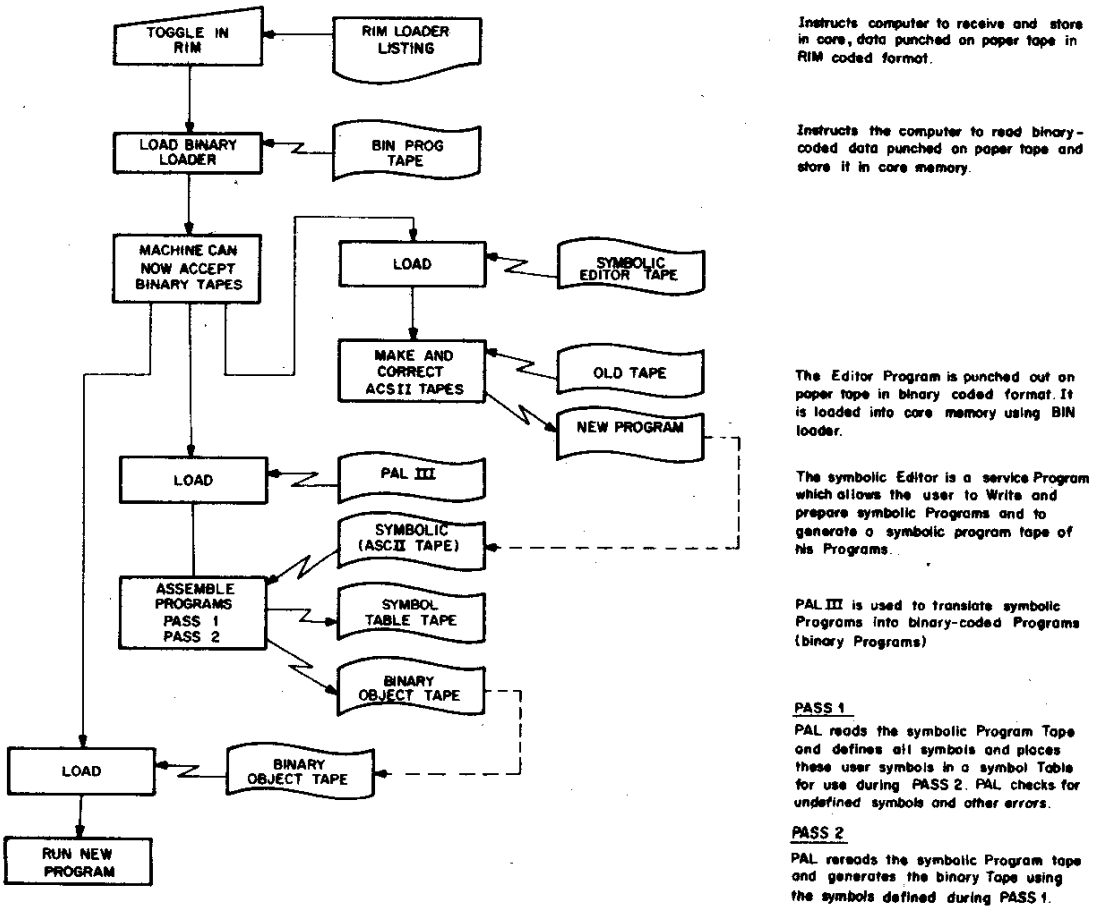


Figure 2-4 Loading Data Under Program Control

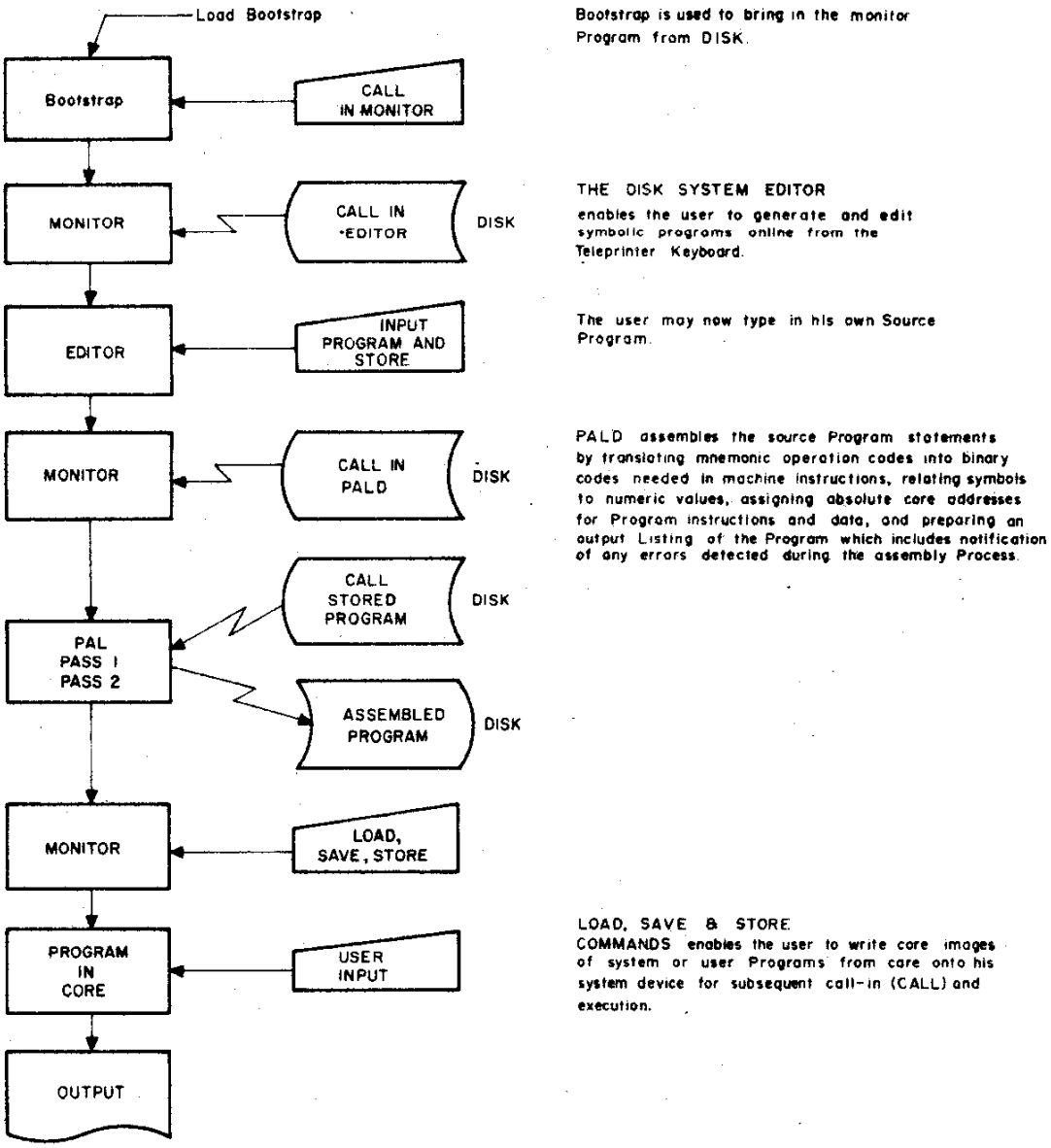


Figure 2-5 Loading Programs with Mass Storage Devices

Table 2-4
RIM Loader Programs

LOCATION	INSTRUCTION	
	Low-Speed Reader	High-Speed Reader
7756	6032	6014
7757	6031	6011
7760	5357	5357
7761	6036	6016
7762	7106	7106
7763	7006	7006
7764	7510	7510
7765	5357	5374
7766	7006	7006

LOCATION	INSTRUCTION	
	Low-Speed Reader	High-Speed Reader
7767	6031	6011
7770	5367	5367
7771	6034	6016
7772	7420	7420
7773	3776	3776
7774	3376	3376
7775	5356	5357
7776	0000	0000

After RIM has been loaded, it is good programming practice to verify that all instructions were stored properly. This can be done by performing the steps illustrated in Figure 2-7, which also shows how to correct an incorrectly stored instruction.
When loaded, the RIM Loader occupies absolute locations 7765 through 7776.

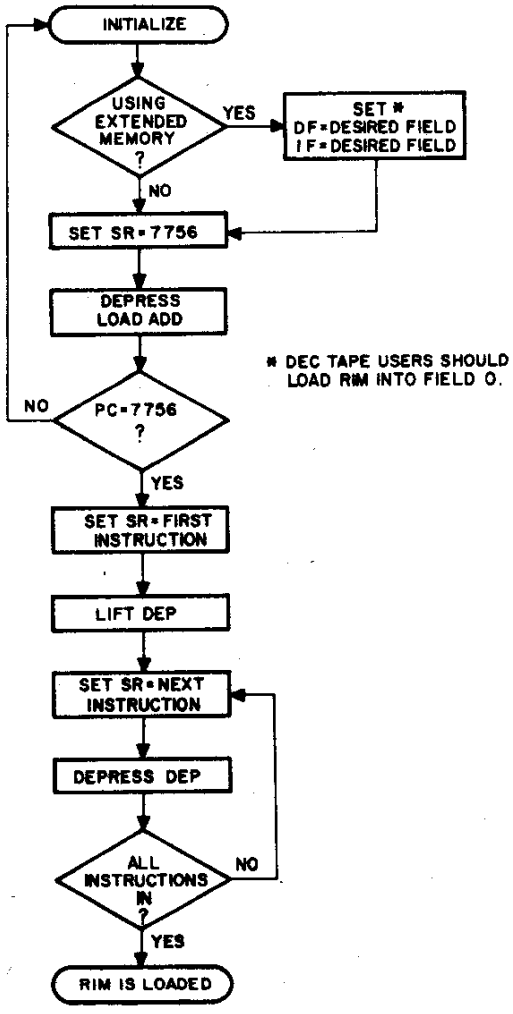


Figure 2-6 Loading the RIM Loader

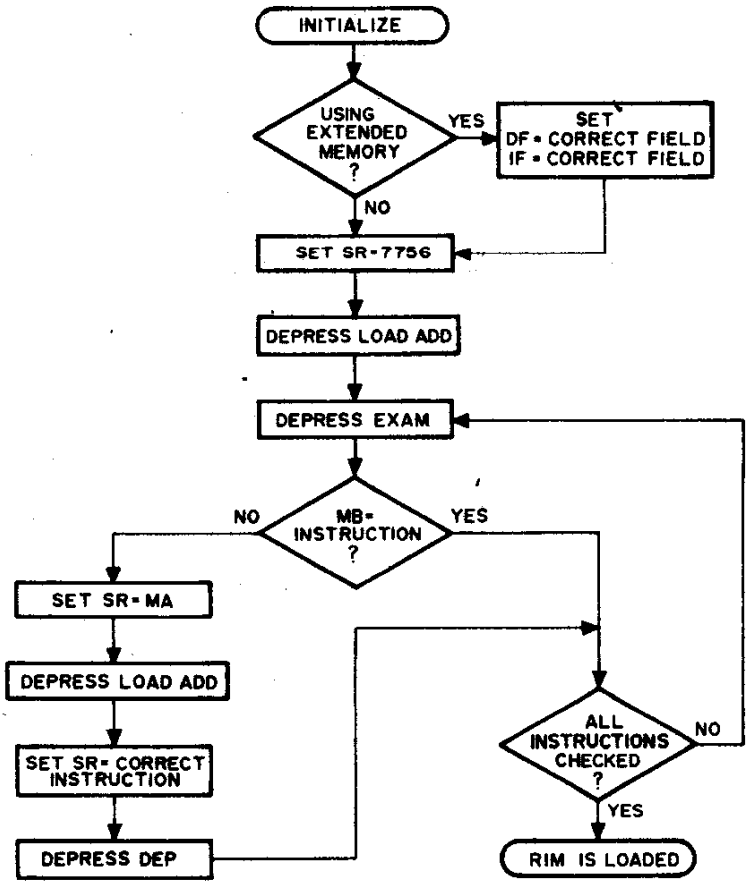


Figure 2-7 Checking the RIM Loader

BINARY (BIN) LOADER

The BIN Loader is a short utility program which, when in core, instructs the computer to read binary-coded data punched on paper tape and store it in core memory. BIN is used primarily to load the programs furnished in the software package (excluding the loaders and certain subroutines) and the programmer's binary tapes.

BIN is furnished to the programmer on punched paper tape in RIM-coded format. Therefore, RIM must be in core before BIN can be loaded. Figure 2-8 illustrates the steps necessary to properly load BIN. When loading, the input device (low- or high-speed reader) must correspond to the version of RIM loaded in the machine.

When stored in core. BIN resides on the last page of core, occupying absolute locations 7625 through 7752 and 7777.

BIN was purposely placed on the last page of core so that it would always be available for use—the programs in DEC's software package do not use the last page of core (excluding the Disk Monitor). The programmer must be aware that if he writes a program which uses the last page of core, BIN will be wiped out when that program runs on the computer. When this happens, the programmer must load RIM and then BIN before he can load another binary tape.

Figure 2-9 illustrates the procedure for loading binary tapes into core.

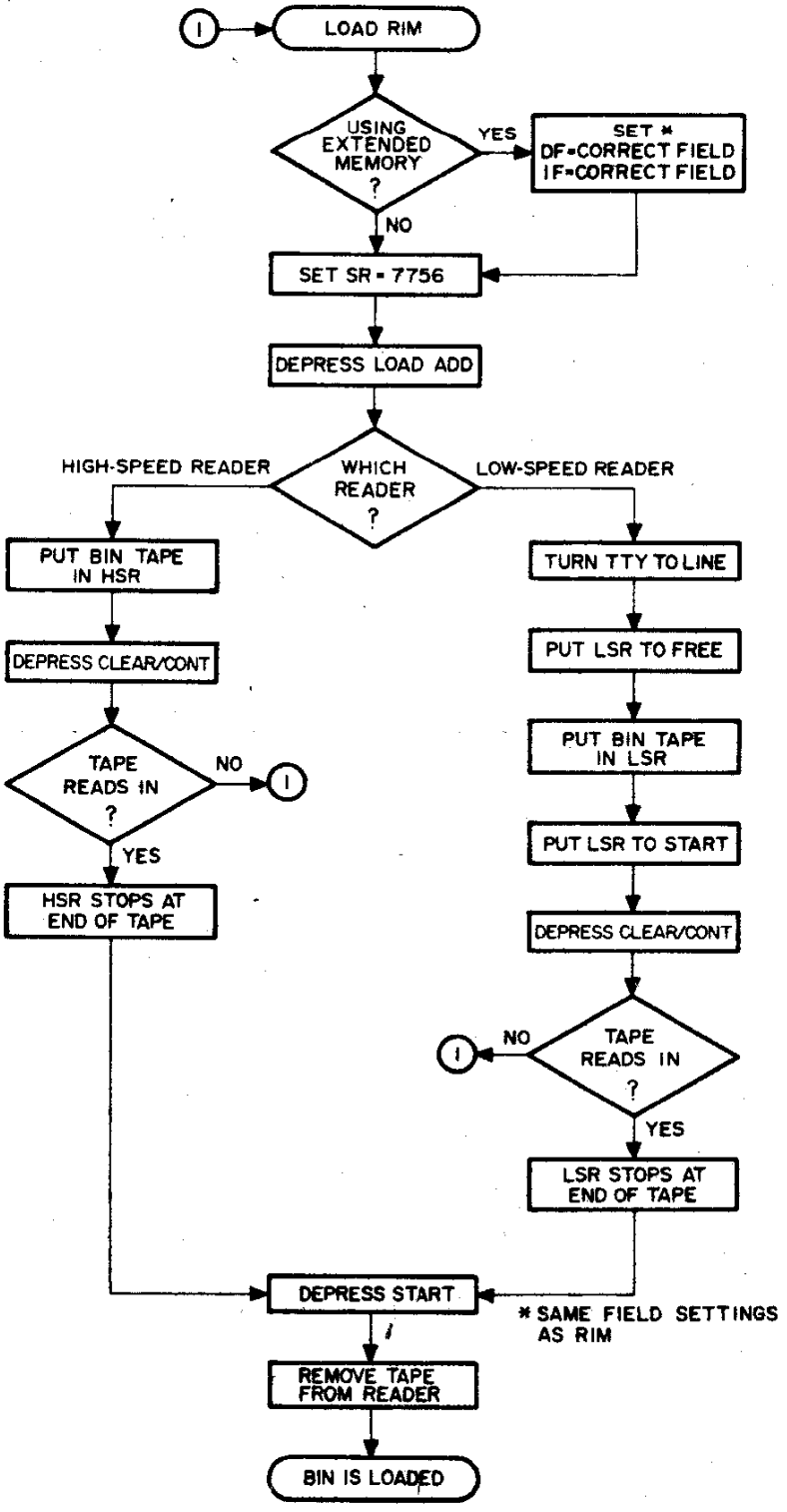


Figure 2-8 Loading the BIN Loader

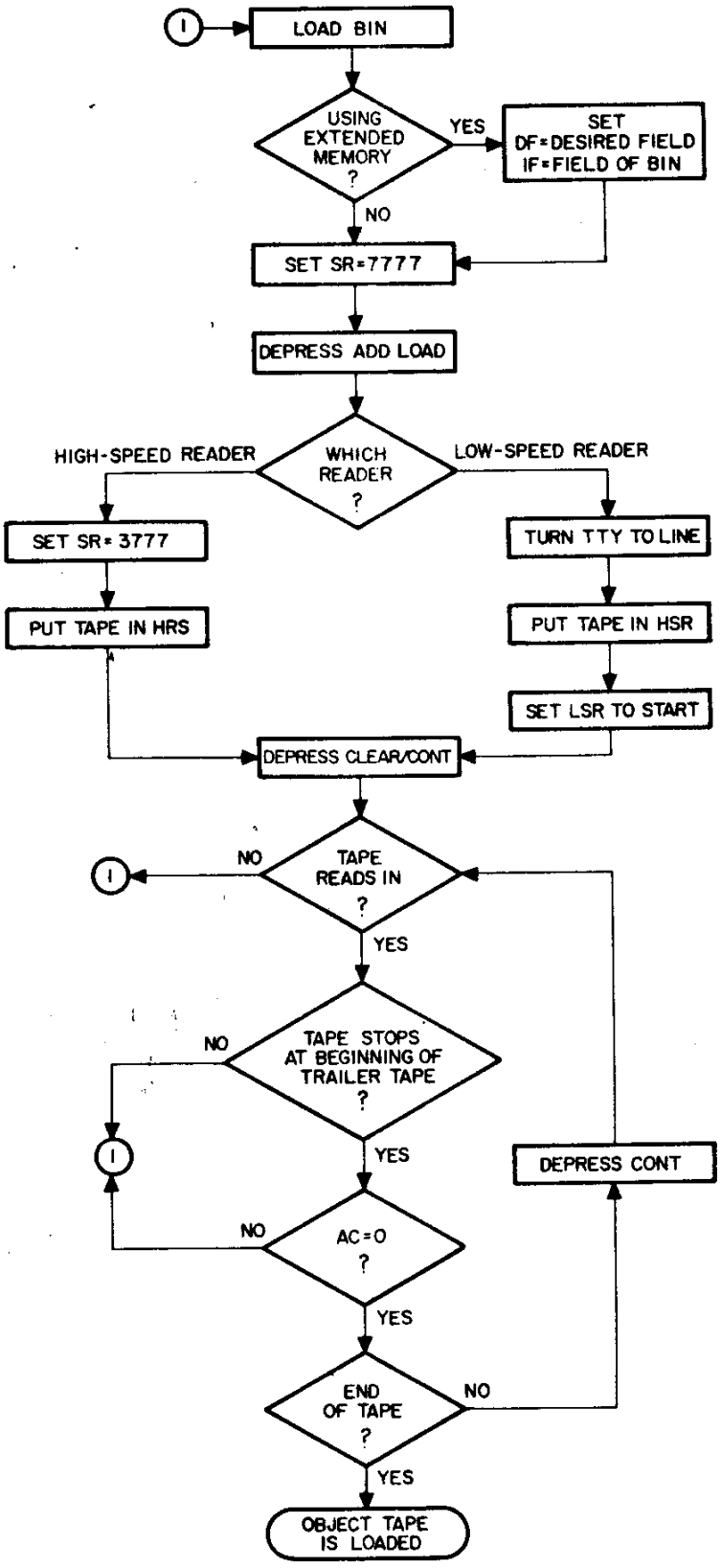


Figure 2-9 Loading A Binary Tape Using BIN

Symbolic Editor

The Symbolic Editor is a service program which allows the programmer to write and prepare symbolic programs and to generate a symbolic program tape of his programs. Editor is very flexible in that the programmer can type his symbolic program online from the Teletype keyboard, thus storing it directly into core memory. Then, using certain Editor commands, the programmer can have his program listed (printed) on the teleprinter for visual inspection.

Editor also allows the programmer to add, correct, or delete any portion of his symbolic program. When the programmer is satisfied that his program is correct and ready to be assembled or compiled, Editor can be commanded to generate a symbolic program tape of the stored program.

The Symbolic Editor program is issued on punched paper tape in binary-coded format. Therefore, it is loaded into core memory using the BIN Loader. When in core, Editor is activated for use by setting the switch register (SR) to 0200 (the starting address) and depressing the LOAD ADD (load address) and then START switches. Editor responds with a carriage return/line feed sequence on the Teletype.

Initially, Editor is in command mode, that is, it is ready to accept commands from the programmer; anything typed by the programmer is interpreted as a command to Editor. Editor accepts only legal commands, and if the programmer types something else, Editor ignores the command and types a question mark (?).

When not in command mode, Editor is in text mode, that is, all characters typed from the keyboard or tapes read in on the tape reader are interpreted as text to be put into the text buffer in the manner specified by a preceding Editor command. Figure 2-10 illustrates how the programmer can transfer Editor from one mode to another.

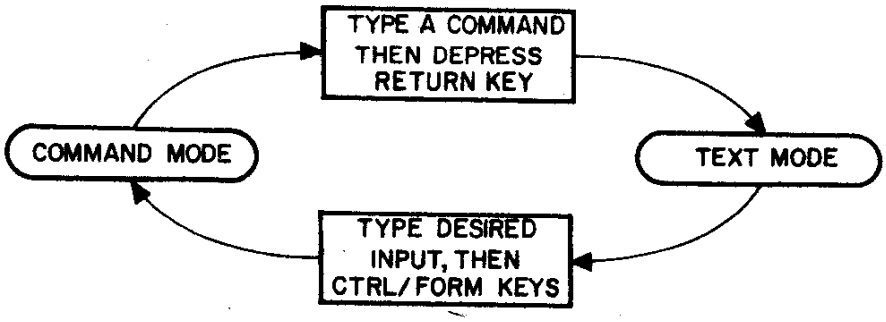


Figure 2-10 Transition Between Editor Modes

Seven of Editor's basic commands are briefly described below.

COMMAND	MEANING
A	Append incoming text from the keyboard into the text buffer immediately following the text currently stored in the buffer.
R	Read incoming text from the tape reader and append it to the text currently stored in the buffer.
L	List entire text buffer; the programmer can specify one line or a group of lines.
C	Change a line; the programmer precedes the command with the decimal line number or line numbers of the lines to be changed.
I	Insert into text buffer; the programmer specifies the decimal line number in his program where the inserted text is to begin.
D	Delete from text buffer; the programmer specifies the line or group of lines to be deleted.
P	Punch text buffer; the programmer can specify one line, a group of lines, or the entire text buffer.

All commands are executed when the RETURN key is depressed except the P command. To execute the P command, press the RETURN key on the Teletype, turn on the punch, and press the CONT (continue) switch on the computer console.

The above commands are only the seven basic commands. A summary of all commands is provided in Table 2-7 at the end of this section.

WRITING A PROGRAM

Now that you have some idea of what you can do with Editor and what Editor can do for you, we will write and edit a short program, explaining each step in the comments to the right of the printout.

The example program finds the larger of two numbers and halts with the number displayed in the accumulator (AC). The program is written in PAL III, to be assembled using the PAL III Assembler described later in this chapter.

The programmer loads Editor using the BIN loader (see Figure 2-8). Editor is then activated by loading the starting address (0200(octal)) and depressing the LOAD ADD, CLEAR and CONT switches. After Editor responds with a carriage return-line feed, the programmer types A and *RETURN key, Editor is now in text mode, that is, subsequent characters* typed are appended to the text buffer. The programmer now types the symbolic program. (Block indenting is facilitated using the CTRL/TAB key, which Editor has programmed to indent in ten-character increments.)

```
A
                *200
                CLA             /CLEAR AC
                TAD NUMB        /GET B
                CMA
                CMA             /1'S COMP B
                IAC             /-B
                TAD NUMA        /ADD -B + A
                SMA             /IF -B LARGER
                JMP .+4         /JUMP 4 LOCATIONS
                CLA             /CLEAR AC
                TAD NUMB        /GET B
                HLT             /B IS LARGER
                CLA             /CLEAR AC
                TAD NUMA        /GET A
                HLT             /A IS LARGER
NUMB,           0000
NUMA,           0000
                $
```

Visual inspection reveals that we have errors in lines 4, 16, and 17. (Editor maintains a line number count in decimal, with the first line typed being 1 and our last line being 18.) Line 4 can be removed using the D (Delete) command, and lines 16 and 17 can be corrected using the C (Change) command. However, Editor is presently in text mode, and in order to issue another command Editor must be transferred to command mode. This is done when the programmer types CTRL/FORM (depress and hold down the CTRL key while typing the FORM key).

CTRL/FORM (nonprinting)	The programmer types CTRL/FORM; Editor responds with CR/LF and rings the teleprinter bell, indicating that it is in command mode.
4D	The programmer types 4D and the RETURN key; Editor responds with a CR/LF and the line is deleted.
15, 16C	The programmer types 15, 16C and the RETURN key, informing Editor that lines 15 and 16 (formerly 16 and 17) are to be changed. Editor responds with a CR/LF, transfers *to text mode, and waits for the programmer to change the lines.*
NUMA, 1111 NUMB, 0011	The programmer types NUMA, 1111 and NUMB, 0011.

The symbolic program should now be correct. However, it is good programming practice to check the program after editing; this can be done using the L (List) command, but since only original lines 4, 16, and 17 were changed it is not necessary to have the whole program listed. The programmer can command Editor to list lines 4 through 17.

CTRL/FORM (nonprinting)	The programmer types CTRL/FORM to return Editor to command mode; Editor responds with CR/LF and rings the bell, and waits for the next command.
4, 17L	The programmer types 4, 17L and the RETURN key; Editor types lines 4 through 17.

```
            CMA         /1'S COMP B
            IAC         /-B
            TAD NUMA    /ADD -B + A
            SMA         /IF -B LARGER
            JMP .+4     /JUMP 4 LOCATIONS
            CLA         /CLEAR AC
            TAD NUMB    /GET B
            HLT         /B IS LARGER
            CLA         /CLEAR AC
            TAD NUMA    /GET A
            HLT         /A IS LARGER
NUMA,       1111
NUMB,       0011
            $
```

The changes were accepted properly. The symbolic program is correct and ready to be punched on paper tape.

GENERATING A PROGRAM TAPE

Before issuing the P (Punch) command, Editor must be in command mode. Figure 2-11 illustrates the procedures required to generate a symbolic program tape using Editor.

CTRL/FORM (nonprinting)	The programmer types CTRL/FORM; Editor responds with a question mark, indicating that Editor was already in command mode.
?	
P	The programmer commands Editor to punch the entire text buffer by typing P and the RETURN key.

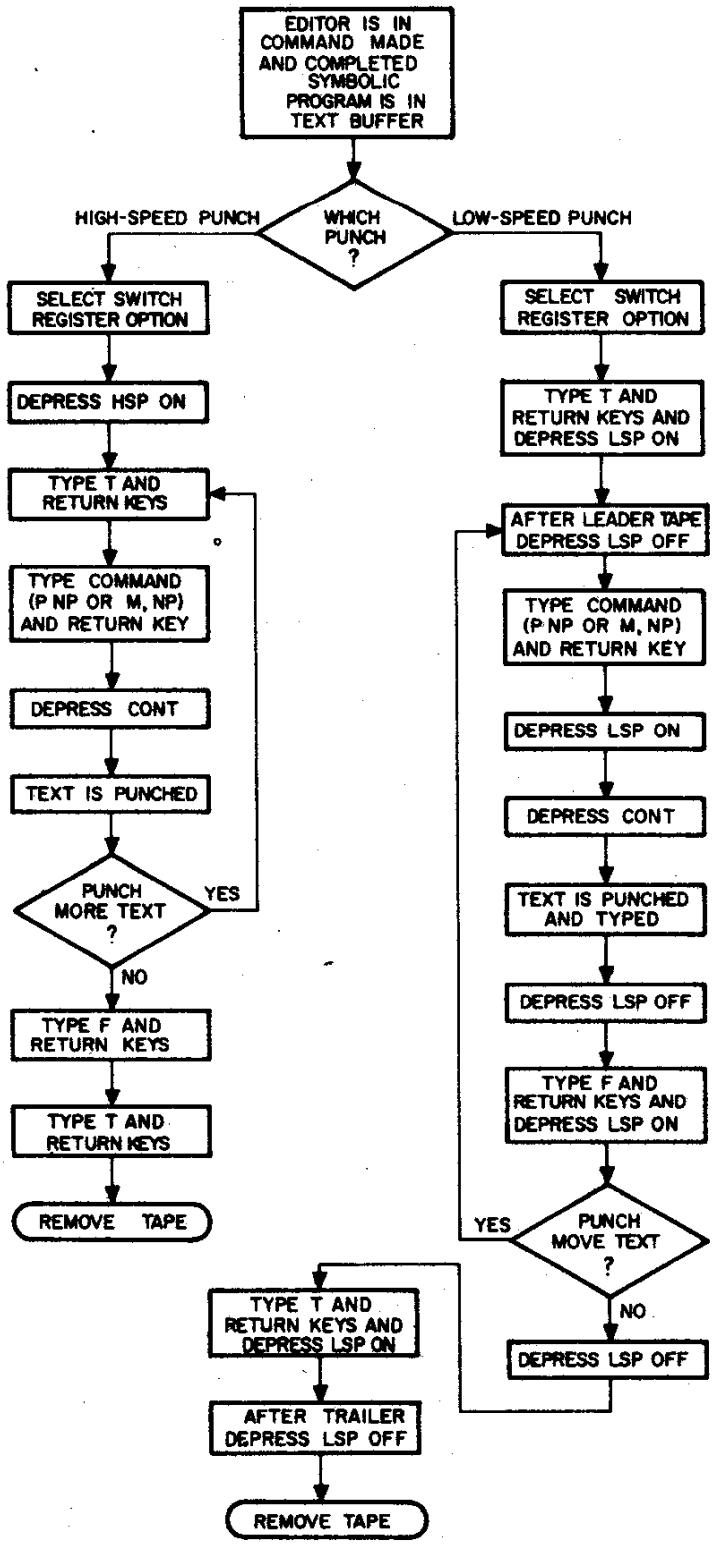


Figure 2-11 Generating a Symbolic Tape Using Editor

When Editor recognizes a P command it waits for the programmer to specify the low- or high-speed punch. If the programmer wants the pro-

gram punched and typed, he sets SR bit 10 to 0 and the program will be punched on the low-speed punch and simultaneously typed on the teleprinter. If the programmer want only a program tape and if he has a high-speed punch available, he sets SR bit 10 to 1 and the program will be punched on the high-speed punch. For the purposes of this discussion, a printed program listing is desired, so the low-speed punch is specified. The programmer turns on the low-speed punch and depresses the CONT switch on the computer console, and Editor begins punching and typing the contents of the entire text buffer.

An image of the stored symbolic program has been punched and typed by Editor

If the programmer stops the computer, e.g., purposely or accidentally turning the computer off, he may restart Editor at location 0200 or 0177 without disturbing the text in the buffer. Editor can also be restarted at location 0176; however, all text currently in the buffer is wiped out. Therefore, the programmer can restart at location 0176 to re-initialize for a new program.

SEARCH FEATURE

A very convenient feature available with Editor is the search feature which allows the programmer to search a line of text for a specified character. When the programmer types a line number followed by S, Editor waits for the user to type in the character for which it is to search. The search character is not echoed (printed on the teleprinter). When Editor locates and types the search character typing stops, and Editor waits for the programmer to either type new text and terminate the line with a RETURN key or to use one of the following special keys.

1.	←	to delete the entire line to the left,
2.	RETURN	to delete the entire line to the right,
3.	RUBOUT	to delete from right to left one character for each RUBOUT typed (a / is echoed for each RUBOUT typed).
4.	LINE FEED	to insert a carriage return/line feed (CR/LF) thus dividing the line into two,
5.	CTRL/FORM	to search for the next occurrence of the search character, and/or
6.	CTRL/BELL	to change the search character to the next character typed by the programmer.

INPUT/OUTPUT CONTROL

Switch register options are used with input and output commands to control the reading and punching of paper tape. The options available to the programmer are shown in Table 2-5. These options are used in conjunction with the "Select Switch Register Option" operation in Figure 2-11.

Table 2-5

Input/Output Control

SR BIT	POSITION	FUNCTION
0	0	Input text as is
	1	Convert all occurrences of 2 or more spaces to a tab
1	0	Output each tab as 8 spaces
	1	Tab is punched as tab/rubout
2	0	Output as specified
	1	Suppress output*
10	0	Low-speed punch and Teleprinter
	1	High-speed punch
11	0	Low-speed reader
	1	High-speed reader

* Bit 2 allows the user to interrupt any output command and return immediately to command mode; when desired, merely set bit 2 to 1.

ERROR DETECTION

Editor checks all commands for nonexistent information and incorrect formatting. When an error is detected, Editor types a question mark (?), and ignores the command. However, if an argument is provided for a command that doesn't require one, the argument is ignored and the command is executed properly.

Editor does not recognize extraneous and illegal control characters; therefore, a tape containing these characters can be cleared up or corrected by merely reading the tape into Editor and punching out a new tape.

SUMMARY OF SPECIAL KEYS AND COMMANDS

Using special keyboard keys and commands, the programmer controls Editor's operation. Certain keys have special meaning to Editor, of which some can be used in either command or text mode. The mode of operation determines the function of each key. The special keys and their functions are shown in Table 2-6.

Table 2-6

Special Keys

KEY	COMMAND MODE	TEXT MODE
RETURN	Execute preceding command	Enter line in text buffer
←	Cancel preceding command (Editor responds with a ? followed by a carriage return and line feed)	Cancel line to the left margin

Table 2-6 (Cont.)

Special Keys

KEY	COMMAND MODE	TEXT MODE
RUBOUT	same as ←	Delete to the left one character for each depression; a \ (backslash) is echoed (not used in Read (R) command)
CTRL/FORM	Respond with question mark and remain in command mode	Return to command mode and ring teleprinter bell
(period)	Value equal to decimal value of current line (may be used alone or with + or − and a number, e.g., .+8)	Legal text character
/	Value equal to number of last line in buffer; used as an argument	Legal text character
LINE FEED	List next line	Used in Search (S) command to insert CR/LF into line
ALTMODE	List next line	
>	List next line	
<	List previous line	
=	Used with . or / to obtain their value	
	Same as = (gives value of legitimate argument)	
CTRL/TAB		Produces a tab which on output is interpreted as 10 spaces or a tab/rubout, depending on SR option

Editor commands are given when in command mode. There are three basic types of commands: Input, Editing, and Output. Table 2-7 contains a summary of Editor commands and their function.

Table 2-7

Summary of Commands

TYPE	COMMAND	FUNCTION
Input	A	Append incoming text from keyboard into text buffer
	R	Append incoming text from tape reader into text buffer
Editing	L	List entire text buffer
	nL	List line n
	m,nL	List lines m through n inclusively
	nC	Change line n
	m,nC	Change lines m through n inclusively
	I	Insert before first line
	nI	Insert before line n
	K	Delete entire text buffer
	nD	Delete line n
	m,nD	Delete lines m through n inclusively
	m,n$jM	Move lines m through n to before line j
	G	Print next tagged line (if none, Editor types ?)
	nG	Print next tagged line after line n (if none, ?)
	S	Search buffer for character specified after RETURN key and allow modification (search character is not echoed on printer)
	nS	Search line n, as above
	m,nS	Search lines m through n inclusively, as above
Output	P	Punch entire text buffer
	nP	Punch line n
	m,nP	Punch lines m through n inclusively
	T	Punch about 6 inches of leader/trailer tape
	F	Punch a FORM FEED onto tape
	N	Do P, F, K, and R commands

m and n are decimal numbers, and m is smaller than n; j is a decimal number.

The P and N commands halt the Editor to allow the programmer to select I/O control; press CONT to execute these commands.

Commands are executed when the RETURN key is depressed, excluding the P and N commands.

PAL III SYMBOLIC ASSEMBLER

The PAL III Symbolic Assembler (PAL stands for Program Assembly Language) is an indispensable service program used to translate symbolic programs, which are written in the PAL III language, into binary-coded programs (binary programs).

PAL III is a two-pass assembler with an optional third pass, i.e., the symbolic program tape must be passed through the assembler two times to produce the binary-coded tape (binary tape), and the optional third pass produces a complete octal/symbolic program listing which can be

typed and/or punched if desired. A brief explanation of the three passes is given below.

Pass 1. The assembler reads the symbolic program tape and defines all symbols used and places the user symbols in a symbol table for use during Pass 2. The assembler checks for undefined symbols and certain other errors and types an error message on the teleprinter when an error is detected.

Pass 2. The assembler rereads the symbolic program tape and generates the binary tape using the symbols defined during Pass 1. When the low-speed punch is used, meaningless characters will be typed on the teleprinter, and these should be ignored by the programmer. The assembler checks illegal referencing during this pass and types an error message on the teleprinter when any is detected.

Pass 3. The assembler reads the symbolic program tape and types and/or punches the octal/symbolic program assembly listing. This listing thoroughly documents the assembled program and is useful when debugging and modifying the program.

The meaningless characters, error messages, and octal/symbolic program listing will be shown later in this section.

PAL III accepts symbolic program tapes from either the low-speed or high-speed reader and produces the binary tapes on either the low-speed or high-speed punch.

During assembly, the programmer communicates with PAL III via the switches on the computer console. Switch options are used to specify which pass the assembler is to perform and which reader and punch the assembler should accept input from and punch out on.

ASSEMBLING A SYMBOLIC PROGRAM

Earlier in this chapter, the programmer wrote a PAL III symbolic program and generated the symbolic program tape using Editor. That symbolic program can now be assembled to produce a binary program using PAL III. A listing of the symbolic program follows.

```
        *200
        CLA             /CLEAR AC
        TAD NUMB        /GET B
        CMA             /1'S COMP B
        IAC             /-B
        TAD NUMA        /ADD -B + A
        SMA             /IF -B LARGER
        JMP .+4         /JUMP 4 LOCATIONS
        CLA             /CLEAR AC
        TAD NUMB        /GET B
        HLT             /B IS LARGER
        CLA             /CLEAR AC
        TAD NUMA        /GET A
        HLT             /A IS LARGER
NUMA,   1111
NUMB,   0011
        S
```

First, PAL III must be loaded into core memory, and since PAL III is on punched paper tape in binary-coded format, it is loaded into core memory using the BIN Loader (see Figure 2-8 for loading procedures).

With PAL III in core, we are ready to assemble the symbolic program. Figures 2-12 and 2-13 illustrate the procedures for assembling with PAL III using the low-speed reader/punch and high-speed reader/punch, respectively. In these flowcharts, the switch register options are set for the appropriate reader/punch.

The low-speed reader and punch (LSR and LSP) are used in the following assembly (see Figure 2-13).

Initializing and Starting	Load PAL III into core memory using BIN. Set SR = 0200 and depress LOAD ADD. Turn TTY to LINE and put symbolic program tape in LSR .
Entering Pass 1	Set SR = 2200 and set LSR to START. Turn LSP ON and depress START.
NUMA 0215 NUMB 0216	Error messages would be typed now. Symbol table concludes Pass 1. Examine Symbol table. Make sure no "UA" diagnostics appear.
Entering Pass 2 BB: 8 8 = * < : <)	Put symbolic program tape in LSR. Set SR = 4200 and set LSR to START. Depress LSP to ON and depress CONT. Disregard meaningless characters while object tape is being punched. Error message would be typed now.
Entering Pass 3	Put symbolic program tape in LSR. Set SR = 6200 and set LSR to START. Depress LSP to ON and depress CONT. The octal/symbolic program listing is being typed and punched.

```
                    *200
0200    7200        CLA         /CLEAR AC
0201    1216        TAD NUMB    /GET B
0202    7040        CMA         /1'S COMP B
0203    7001        IAC         /-B
0204    1215        TAD NUMA    /ADD -B + A
0205    7500        SMA         /IF -B LARGER
0206    5212        JMP .+4     /JUMP 4 LOCATIONS
0207    7200        CLA         /CLEAR AC
0210    1216        TAD NUMB    /GET B
0211    7402        HLT         /B IS LARGER
0212    7200        CLA         /CLEAR AC
0213    1215        TAD NUMA    /GET A
0214    7402        HLT         /A IS LARGER
0215    1111  NUMA, 1111
0216    0011  NUMB, 0011
NUMA    0215
NUMB    0216
```

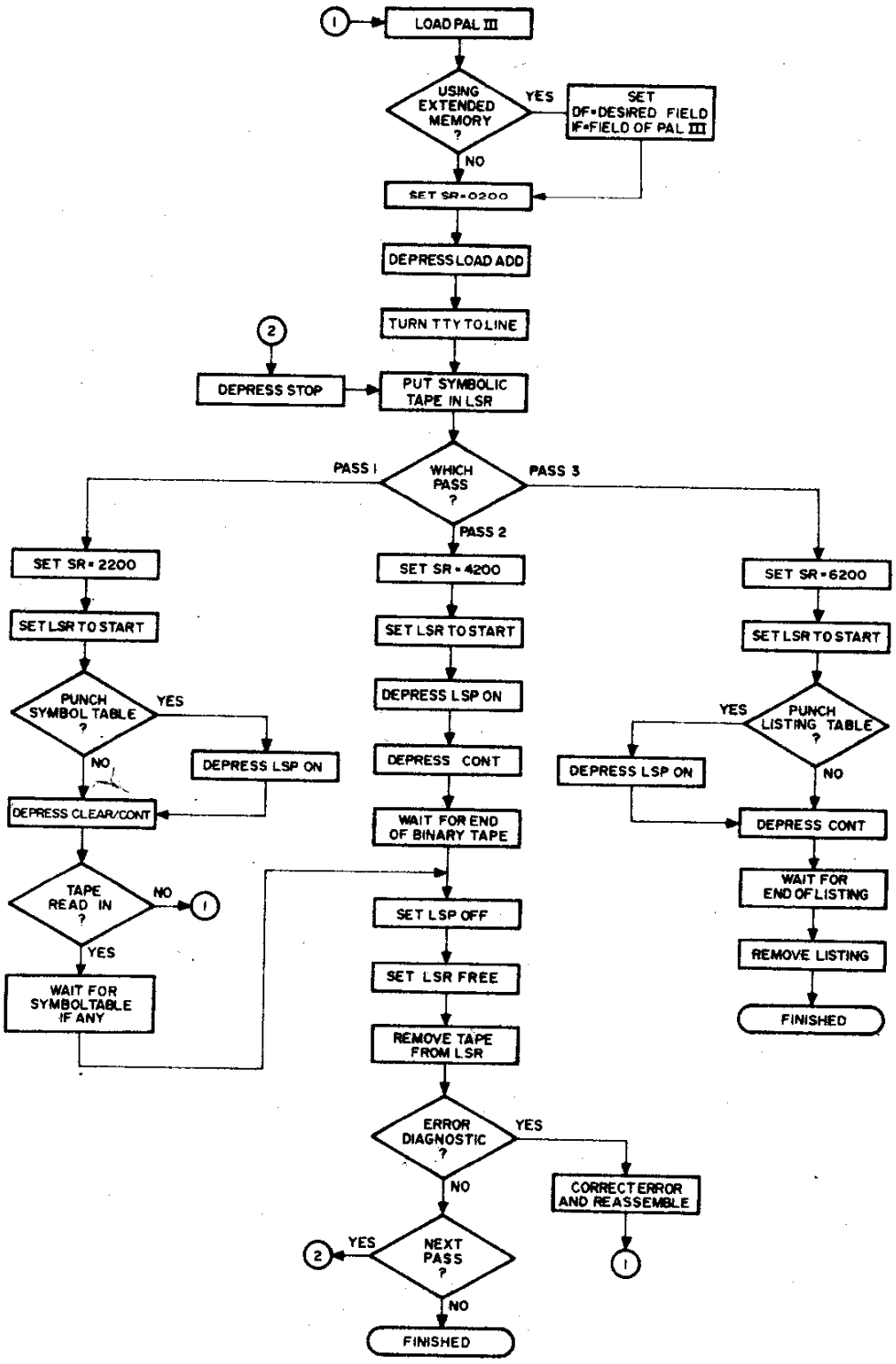


Figure 2-12 Assembling with PAL III Using Low-Speed Reader/Punch

The tape produced during Pass 2 is the binary tape, which is loaded into core memory using the BIN Loader. The symbol table tape produced during Pass 1, the binary tape produced during Pass 2, and the octal/symbolic program listing produced during Pass 3 are used when debugging the program.

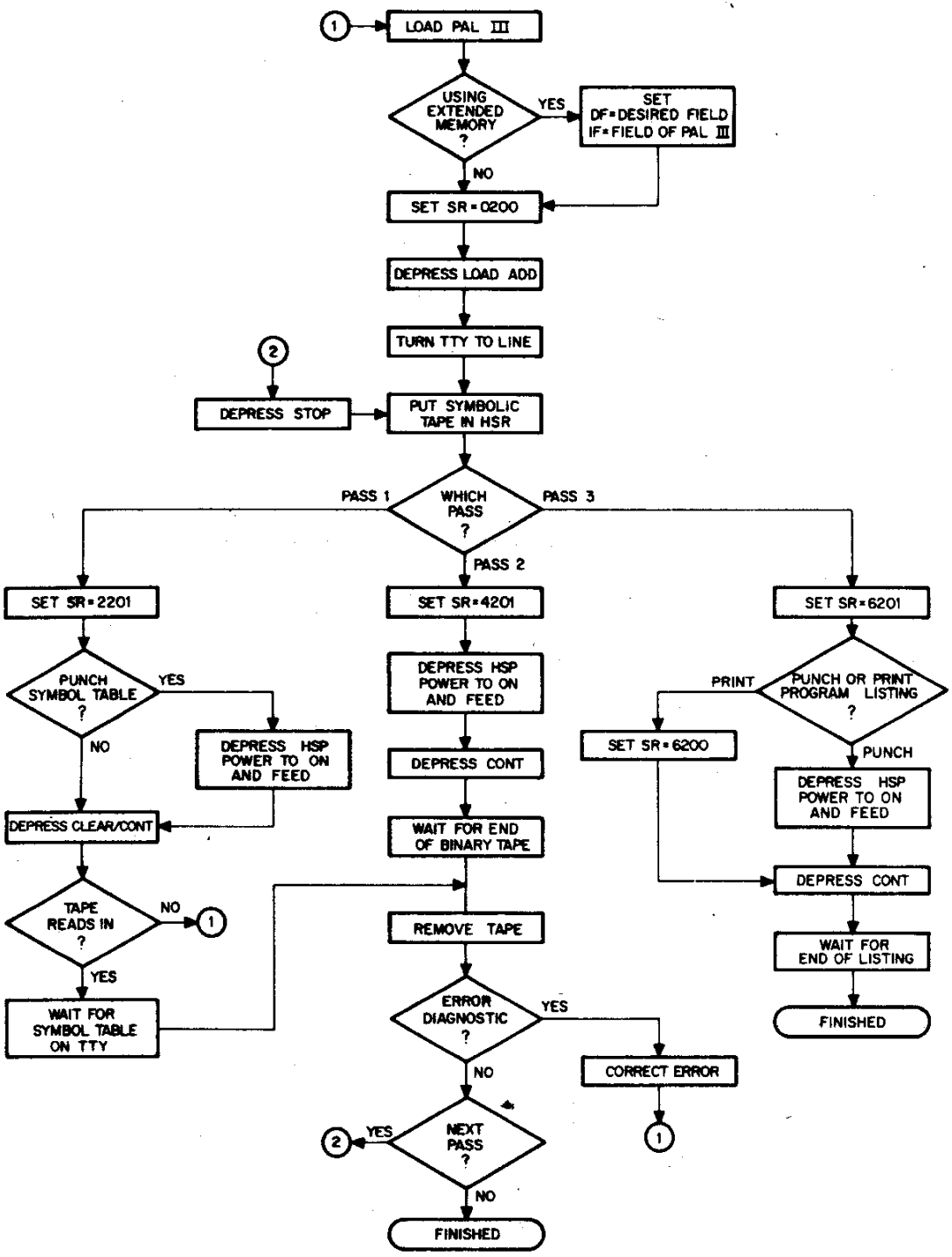


Figure 2-13 Assembling with PAL III Using High-Speed Reader/Punch

Off-Line Teletype Operation

The Teletype can be used separately from the computer for typing, punching tape, or duplicating tapes. To use the Teletye in this manner:

a. Ensure that the computer PANEL LOCK switch is positioned to the PANEL LOCK position.

b. Set the Teletype LINE/OFF/LOCAL switch to the LOCAL position.
c. 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, and press the @ (P) key. When the required amount of leader has been punched, release the @ key, and then 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.

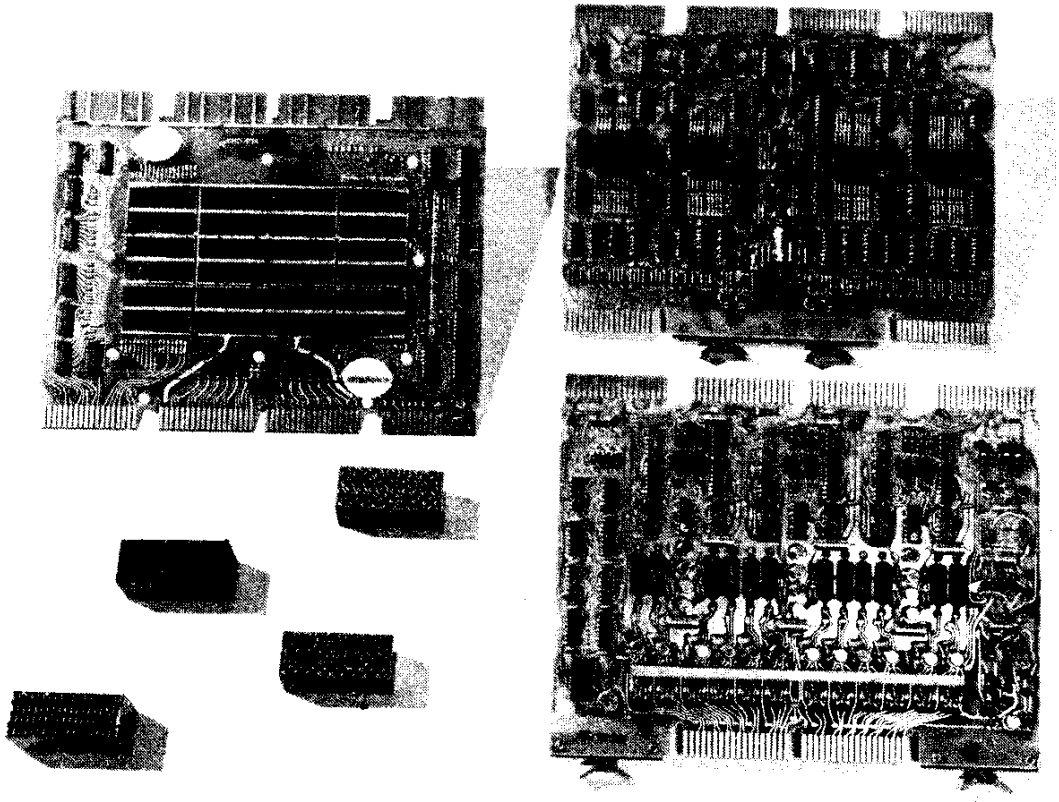
If an incorrect key is struck while punching a tape, the tape can be corrected as follows: If the error is noted after typing and punching any number (n) of characters, press the punch B.SP. (backspace) pushbutton n + 1 times and strike the keyboard RUB OUT key n + 1 times. Then continue typing and punching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

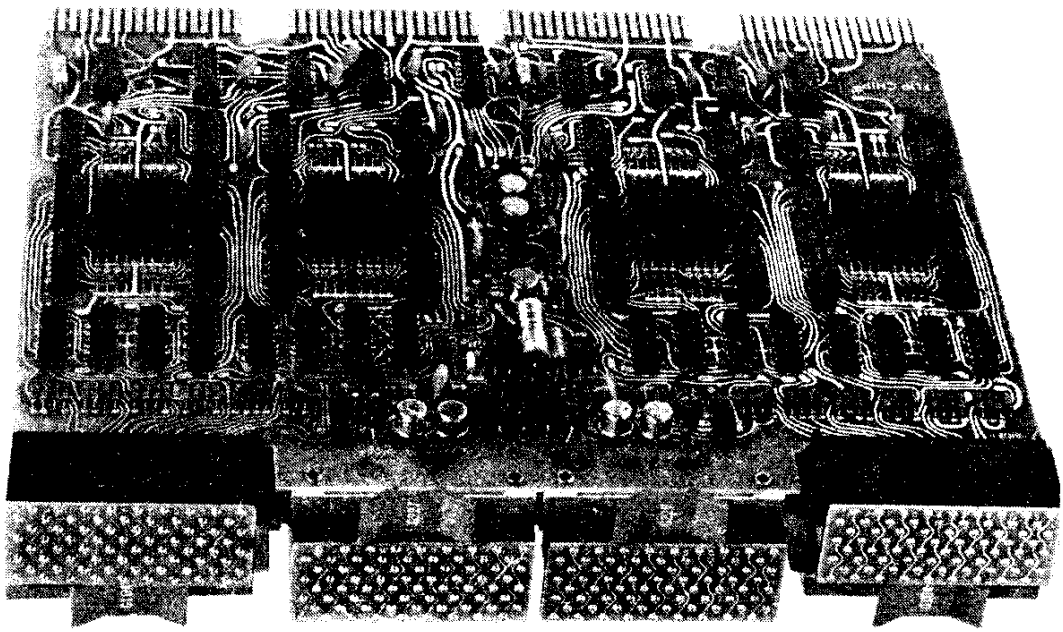
To duplicate and obtain a listing of an existing tape: Perform the procedure under steps a through c above. Then load the tape to be duplicated as described in step b of the procedure listed under Loading Data Under Program Control. Initiate tape duplication by setting the reader START/STOP/FREE switch to the START position. The punch and teleprinter stops 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 reached. Information to be inserted can then be punched manually by means of the keyboard. Information can be deleted by pressing the punch OFF pushbutton and operating the reader until the portion of the tape to be deleted has been typed. It may be necessary to backspace and rub out one or two characters on the new tape if the reader is not stopped precisely on time. The number of characters to be rubbed out can be determined exactly by the typed copy. Be sure to count spaces when counting typed characters. Continue duplicating the tape in the normal manner after making the corrections.

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

Program Control

When the program is stopped at the end of an instruction with 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 to the address wanted, and then operating the CONTinue key.





4K Memory

CHAPTER 3

MEMORY AND PROCESSOR INSTRUCTIONS

GENERAL

An instruction is a coded program step that tells the computer what to do for a single operation in a program. In the PDP-8 family of computers, there are two major types of instruction words: memory reference and augmented. Memory reference instructions store or retrieve data from core memory, while augmented instructions do not. A third type—a housekeeping instruction is defined later. All instructions are determined by bits 0 through 2 to specify the operation (op) code. Operation codes of 0(octal) through 4(octal) specify memory reference instructions, and code of 6(octal) and 7(octal) specify augmented instructions. Memory reference instruction times are 1.2 μs for instruction fetches and non-auto-indexed defer cycles, and 1.4 μs for all other cycles. IOT (Input/Output Transfer) instructions are 1.2 μs for options which communicate directly with the OMNIBUS; and 2.6, 3.6, or 4.6 μs for options which communicate via the KA8-E Positive I/O Bus Interface. The latter times are + or −5%, all other times are + or −1%.

The instruction repertoire is shown in Figure 3-1. This illustrates the eight basic instructions and the division into the three categories—the Memory Reference Instructions, Augmented Instructions, and the Housekeeping Instruction.

The Memory Reference Instructions are concerned with at least two memory addresses, the address of the initial memory location and the address of the data. Figure 3-2 illustrates an example of the address flow of a Memory Reference Instruction. In order to illustrate the basic flow, a simplified diagram is shown leaving out some of the more complex branching such as JUMP and AUTO INDEX instructions. Two cycles are required (at a minimum) to complete an instruction. The indirect addressing capability requires an additional cycle (DEFER) which must go back to memory for another address. On the next cycle (EXECUTE), the data is brought into the Central Processor and the instruction is completed.

The Augmented Instruction simplified flow program is illustrated in Figure 3-3. Notice that the instruction is completely performed in the FETCH cycle. Once memory is addressed and the instruction brought from memory to the central processor, memory is not referenced (i.e., no additional instruction is brought from memory). Instead, the instruction is completely carried out once the initial decoding has been accomplished. The last 9 bits, instead of being used to specify a memory address, are used as an extension of the basic instruction. The Housekeeping Instruction, JMP (sometimes called Unconditional Branch) is used only to force any desired address into the Program Counter. (See Program Control in chapter 4)

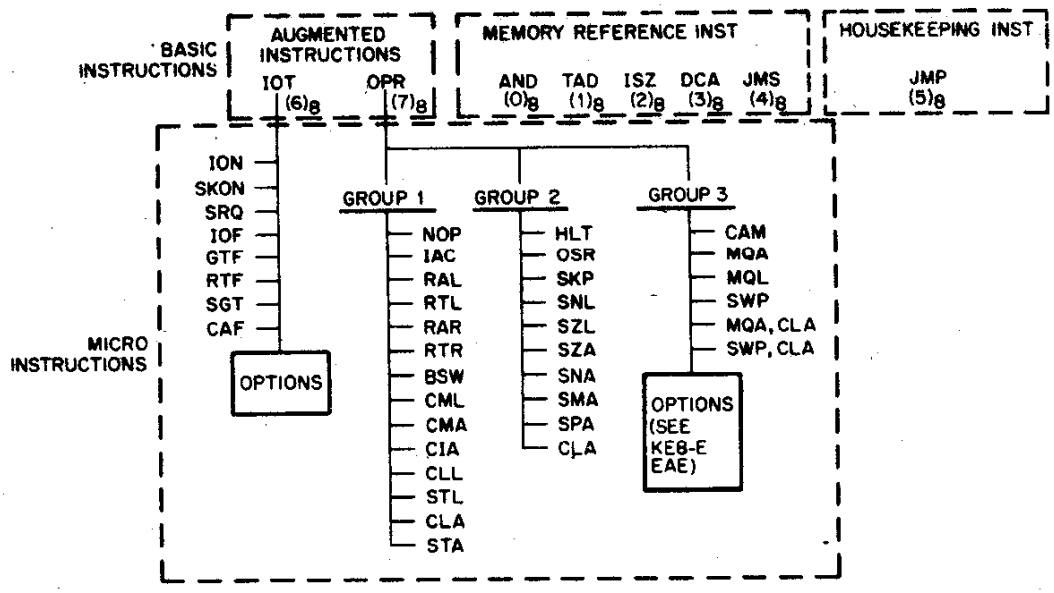


Figure 3-1 PDP-8/E Instruction Repertoire

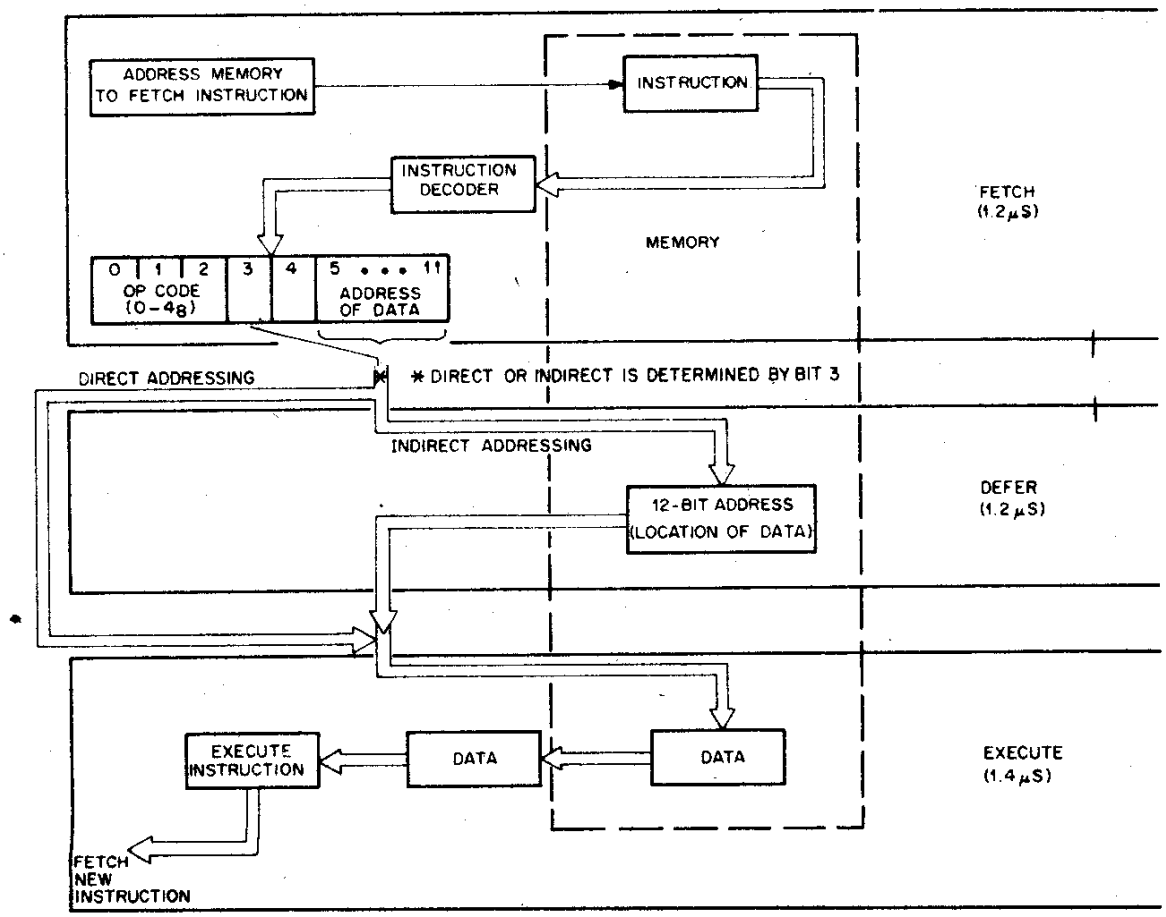


Figure 3-2 Memory Reference Instruction Simplified Flow

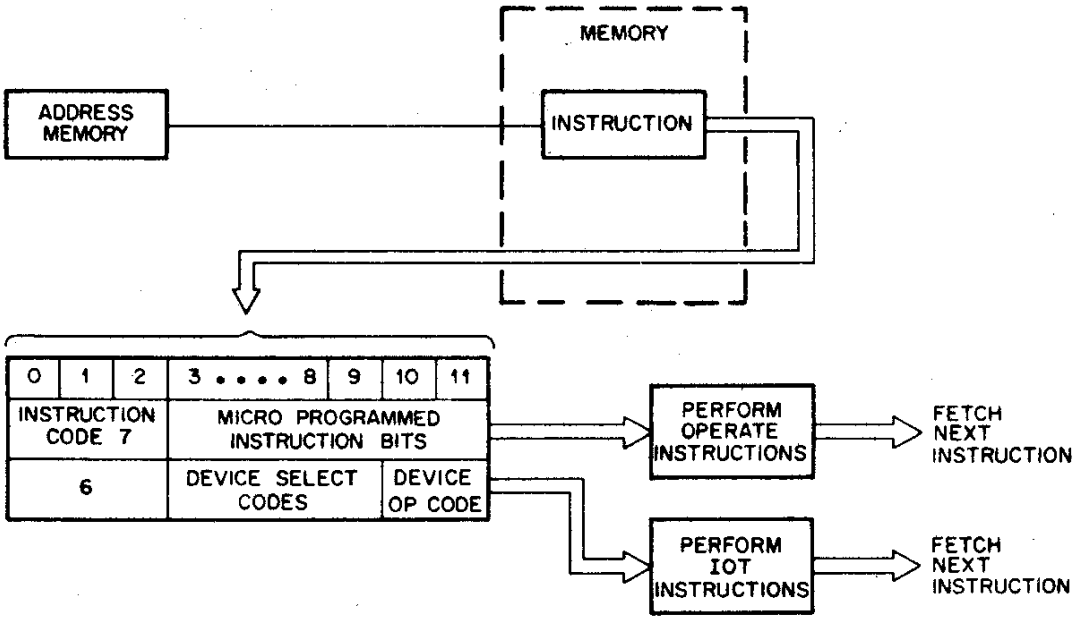


Figure 3-3 Augmented Instruction Simplified Flow Diagram

MEMORY REFERENCE INSTRUCTIONS

Logical AND (AND Y)

Octal Code: 0
Major States: F, (D), E
Execution Time: 2.6 μs with direct addressing, 3.8 μs with indirect addressing, 4.0 μs with auto-indexed indirect addressing.
Operation: The AND operation is performed between the contents of memory location Y and the contents of AC. The result is left in the AC, the original contents of the AC are lost, and the contents of Y are 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

Two's Complement Add (TAD Y)

Octal Code: 1
Major States: F, (D), E
Execution Time: 2.6 μs with direct addressing, 3.8 μs with indirect addressing, 4.0 μs with auto-indexed indirect addressing.

Operation: The contents of memory location Y are added to the contents of the AC in two's complement arithmetic. The result of this addition is held in the AC, the original contents of the AC are lost and the contents of Y are restored. If there is a carry from AC0, the link is complemented. This feature is useful in multiple precision arithmetic.

Increment and Skip if Zero (ISZ Y)

Octal Code: 2
Major States: F, (D), E
Execution Time: 2.6 μs with direct addressing, 3.8 μs with indirect addressing, 4.0 μs with auto-indexed indirect addressing.
Operation: The contents of memory location Y are incremented by one in two's complement arithmetic. If the resultant contents of Y equal zero, the contents of the PC are incremented by one and the next instruction is skipped. If the resultant contents of Y do not equal zero, the program proceeds to the next instruction. The incremented contents of Y are restored to memory. The contents of the AC are not affected by this instruction.

Deposit and Clear AC (DCA Y)

Octal Code: 3
Major States: F, (D), E
Execution time: 2.6 μs with direct addressing, 3.8 μs with indirect addressing, 4.0 μs with auto-indexed indirect addressing.
Operation: The contents of the AC are deposited in core memory at address Y and the AC is cleared. The previous contents of memory location Y are lost.

Jump to Subroutine (JMS Y)

Octal Code: 4
Major States: F, (D), E
Execution Time: 2.6 μs with direct addressing, 3.8 μs with indirect addressing, 4.0 μs with auto-indexed indirect addressing.
Operation: The contents of the PC are deposited in core memory location Y and the next instruction is taken from core memory location Y + 1. The contents of the AC are not affected by this instruction.

HOUSEKEEPING INSTRUCTION
Jump to Y (JMP Y)

Octal Code: 5
Major States: F, (D)
Execution Time: 1.2 μs with direct addressing, 2.4 μs with indirect addressing, 2.6 μs with auto-indexed indirect addressing.

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

AUGMENTED INSTRUCTIONS

Augmented instructions are one-cycle (FETCH) instructions that initiate various operations as a function of bit microprogramming. Augmented instructions are divided into two categories, neither of which are memory reference instructions. These are the INPUT/OUTPUT TRANSFER (IOT), which has an operation code of six; and the OPERATE, which has an operation code of seven. Bits 3 through 11 within each instruction function as an extension of the operations to be performed.

OPERATE INSTRUCTION

The operate instruction consists of three 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 contents of the accumulator and link and continuing to, or skipping, the next instruction based on the check. A 1 in bit 3 and a 0 in bit 11 designates an OPR 2 microinstruction. Group 3 is used to manipulate data between the MQ and AC Registers. A 1 in bits 3 and 11 designate an OPR 3 microinstruction. Each instruction is completed during a FETCH cycle. All operate instructions take place in 1.2 microseconds.

Group 1

The Group 1 microinstructions manipulate the contents of the accumulator and link. These instructions are microprogrammable; that is, they can be combined to perform specialized operations with other Group 1 instructions.

The Group 1 operate microinstruction format is shown in Figure 3-4, and the microinstructions are explained in the succeeding paragraphs. Bits within this group can be combined into one microinstruction. For example, it is possible to assign 1's to bits 5, 6, and 11, thereby creating a single instruction which clears the link, complements the accumulator and increments the accumulator. (The most frequently used combinations are listed in Appendix B.)

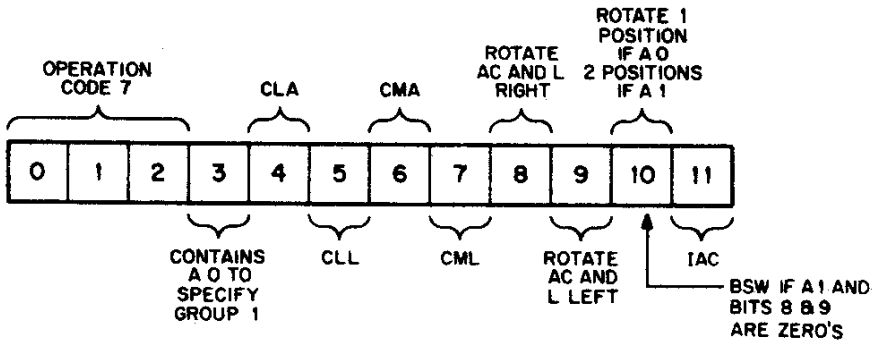


Figure 3-4 Group 1 Operate Instruction Bit Assignments

No Operation (NOP)

Octal Code: 7000
Sequence: None
Operation: This command causes a 1-cycle delay in the program before 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. The NOP also provides the programmer with a convenient means of removing an instruction.

Increment Accumulator (IAC)

Octal Code: 7001
Sequence: 3
Operation: The contents of the AC are incremented by one in two's complement arithmetic.

Rotate Accumulator Left (RAL)

Octal Code: 7004
Sequence: 4
Operation: The contents of the AC and link are rotated one binary position to the left. The contents of bits AC1-11 are shifted to the next greater significant bit, the content of AC0 is shifted into the L, and the content of the L is shifted into AC11.

Rotate Two Left (RTL)

Octal Code: 7006
Sequence: 4
Operation: The contents of the AC and link are rotated two binary positions to the left. This instruction is logically equal to two successive RAL operations.

Rotate Accumulator Right (RAR)

Octal Code: 7010
Sequence: 4
Operation: The contents of the AC and link are rotated one binary position to the right. The contents of bits AC0-10 are shifted to the next less significant bit, the content of AC11 is shifted into the L, and the content of the L is shifted into AC0.

Rotate Two Right (RTR)

Octal Code: 7012
Sequence: 4
Operation: The contents of the AC and link are rotated two binary positions to the right. This instrucion is logically equal to two successive RAR operations.

Byte Swap (BSW)

Octal Code: 7002
Sequence: 4
Operation: The right six bits of the accumulator are exchanged with the left six bits. AC0 is exchanged with AC6; AC1 with AC7, etc. The contents of the link are not affected.

Complement Link (CML)

Octal Code: 7020
Sequence: 2
Operation: The content of the L is complemented.

Complement Accumulator (CMA)

Octal Code: 7040
Sequence: 2
Operation: The contents of the AC are changed to the one's complement of the current contents of the AC. The content of each bit of the AC is complemented individually.

Complement and Increment Accumulator (CIA)

Octal Code: 7041
Sequence: 2, 3
Operation: The contents of the AC are converted from a binary value to their equivalent two's complement number. This conversion is accomplished by combining the CMA and IAC commands, thus the contents of the AC are complemented during sequence 2 and are incremented by one during sequence 3.

Clear Link (CLL)

Octal Code: 7100
Sequence: 1
Operation: The L is cleared (made equal to 0).

Set Link (STL)

Octal Code: 7120
Sequence: 1, 2
Operation: The L is set. This instruction is logically equal to combining the CLL and CML commands.

Clear Accumulator (CLA)

Octal Code: 7200
Sequence: 1
Operation: The content of each bit of the AC is cleared (made equal to 0).

Set Accumulator (STA)

Octal Code: 7240
Sequence: 1, 2
Operation: Each bit of the AC is set. This operation is logically equal to combining the CLA and CMA commands.

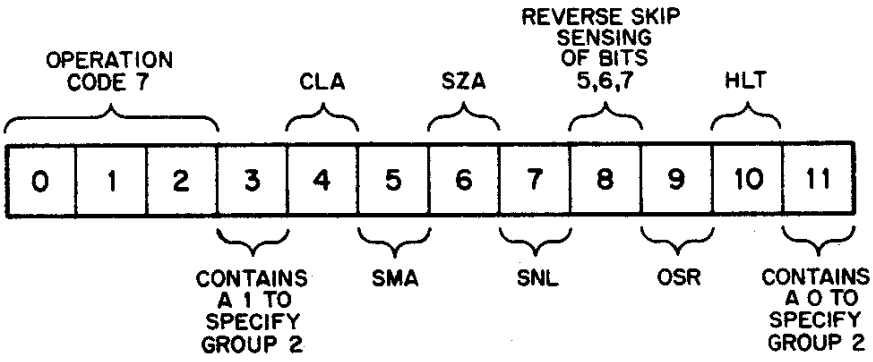
NOTE: The following codes are illegal and specifically reserved for future expansion: RAL RAR (octal code 7014), RTL RTR (octal code 7016) and any microprogramming of these bit combinations.

GROUP 2

Group 2 Operate microinstructions are often referred to as the "skip microinstructions" because they enable the programmer to perform tests on the accumulator and link and to skip the next instruction depending upon the results of the test. A skip instruction causes the computer to check for a specific condition, and if it is present, to skip the next instruction. If the condition is not present, the next instruction is executed.

The Group 2 operate microinstruction format is shown in Figure 3-5 and the primary microinstructions are explained in the following paragraphs. Any logical combination of bits within this group can be combined into one instruction. (The instructions constructed by most logical bit combinations are listed in Appendix B.)

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 1s are designed in bits 6 and 7 (SZA and SNL), the next instruction is skipped if either the contents of the AC = 0, or the content of L= 1. If 1s 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 0) – EITHER SMA OR SZA OR SNL
(BIT 8 IS A 1) – BOTH SPA AND SNA AND SZL
2 – CLA
3 – OSR
4 – HLT

Figure 3-5 Group 2 Operate Instruction Bit Assignments

Halt (HLT)

Octal Code: 7402
Sequence: 4
Operation: Clears the RUN flip-flop at Sequence 4, 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. All other OPR Group 2 Instructions are performed before the program stops.

OR with Switch Register (OSR)

Octal Code: 7404
Sequence: 3
Operation: The inclusive OR operation is performed between the contents of the AC and the contents of the SR. The result is left in the AC, the original contents of the AC are lost. The contents of the SR are unaffected by this command. When combined with the CLA command, the OSR performs a transfer of the contents of the SR into the AC.

Skip, Unconditional (SKP)

Octal Code: 7410
Sequence: 1
Operation: The contents of the PC are incremented by one so that the next sequential instruction is skipped.

Skip on Non-Zero Link (SNL)

Octal Code: 7420
Sequence: 1
Operation: The content of the L is sampled, and if it contains a 1, the contents of the PC are 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.

Skip on Zero Link (SZL)

Octal Code: 7430
Sequence: 1
Operation: The content of the L is sampled, and if it contains a 0 the contents of the PC are incremented by one so that the next sequential instruction is skipped. If the L contains a 1, no operation occurs and the next sequential instruction is initiated.

Skip on Zero Accumulator (SZA)

Octal Code: 7440
Sequence: 1
Operation: The content of each bit of the AC is sampled, and if all bits contain a 0 the contents of the PC are incremented by one so that the next sequential instruction is skipped. If any bit of the AC contains a 1, no operation occurs and the next sequential instruction is initiated.

Skip on Non-Zero Accumulator (SNA)

Octal Code: 7450
Sequence: 1
Operation: The content of each bit of the AC is sampled, and if any bit contains a 1 the contents of the PC are 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.

Skip on Minus Accumulator (SMA)

Octal Code: 7500
Sequence: 1
Operation: The content of the most significant bit of the AC is sampled, and if it contains a 1, indicating that the AC contains a negative two's complement number, the contents of the PC are incremented by one so that the next sequential instruction is skipped. If the AC contains a positive number no operation occurs and the next sequential instruction is initiated.

Skip on Positive Accumulator (SPA)

Octal Code: 7510
Sequence: 1
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 contents of the PC are incremented by one so that the next sequential instruction is skipped. If the AC contains a negative number, no operation occurs and the next sequential instruction is initiated.

Clear Accumulator (CLA)

Octal Code: 7600
Sequence: 2
Operation: Each bit of the AC is cleared to contain a binary 0.

Group 3

The Group 3 Operate microinstructions are concerned with the manipulation of data between the AC and the MQ registers. The MQ register is an auxiliary register for the temporary storage of data. It is sometimes convenient to temporarily store data in an auxiliary register rather than in a memory location. Group 3 instructions enable the loading of AC contents into the MQ; the loading of the MQ contents into the AC, the swapping of AC and MQ contents; the loading of the inclusive OR of the contents of the AC and MQ into the AC; and the clearing of both the AC and MQ. Although the register and instructions are primarily intended to be used with the Extended Arithmetic KE8-E option, they are available for use as a standard feature.

The format of the Group 3 instructions is shown in Figure 3-6. Having an operation code of 7, this instruction class is identified as group 3 only when both bits 3 and 11 contain a 1.

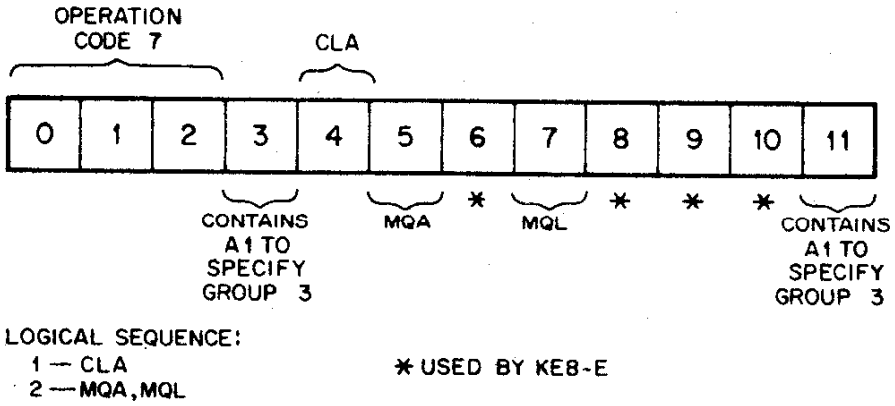


Figure 3-6 Bit Assignments for Group 3 Operate Instructions

Bits 6, 8, 9 and 10 should be zero when the KE8-E option is not employed. The description of the four Group 3 instructions is given in the following:

Instructions

Clear Accumulator and Multiplier Quotient (CAM)

Octal Code: 7621
Execution Time: 1.2 μs
Operation: Clears the AC during logical sequence 1, as in CLA; during logical sequence 2, the MQ is cleared.

Multiplier Quotient Load into Accumulator (MQA)

Octal Code: 7501
Execution Time: 1.2 μs
Operation: The contents of the MQ are inclusively ORed with the AC, and the result loaded into the AC. The previous contents of the AC are lost, but the contents of the MQ are not affected. This instruction provides the programmer with a direct inclusive OR instruction. This instruction may also be combined with a CLA instruction to effect a direct transfer of information from MQ to AC.

Load Multiplier Quotient (MQL)

Octal Code: 7421
Execution Time: 1.2 μs
Operation: Loads the content of the AC into the MQ, and then clears the AC.

Swap MQ and AC (SWP)

Octal Code: 7521
Execution Time: 1.2 μs
Operation: The contents of AC and MQ are exchanged. This instruction can be combined with the CLA bit

to move the contents of MQ to AC and then clear the MQ.

INPUT/OUTPUT TRANSFER (IOT)

Octal Code: 6
Major State: F
Execution Time: If the selected device is internal, the IOT takes place in 1.2 microseconds.
If the selected device is external, the computer enters an expanded cycle of 2.6 microseconds (if the IOP ends in 1, 2 or 4); 3.6 microseconds (if the IOP ends in 3, 5, or 6); or 4.6 microseconds (if the IOP ends in 7). An IOT ending in 0 always takes place in 1.2 microseconds.

Operation: Input/output transfer (IOT) instructions initiate the operation of peripheral equipment and effect information transfers between the processor and an I/O device. Upon recognition of the operation code 6 as an IOT instruction, the computer determines whether the selected device is internal (plugged directly into the OMNIBUS) or external (connected via the KA8-E Positive I/O Bus Interface module). The nature of the OMNIBUS is such that IOT's such as 6000 can be used for control codes for devices directly connected to the OMNIBUS, since the last 3 bits are decoded to determine the device operation.

The last 3 bits of the instruction cause the generation of "IOP PULSES" at the External Bus Interface as follows:

Instruction Bit	IOP	Sequence Time
11	IOP 1	1
10	IOP 2	2
9	IOP 4	3

IOP pulses 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 3-7. Operations performed by IOT microinstructions are explained in Chapters 5, 9, and 10.

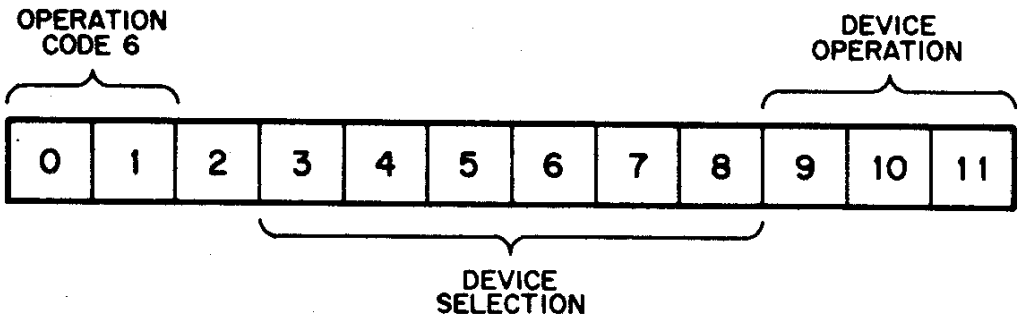


Figure 3-7 IOT Instruction Bit Assignment

PROGRAM INTERRUPT

The program interrupt features allow certain external conditions to interrupt the computer program. Program interrupts are used to either speed the information processing of input/output devices or 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 contents of the program counter are stored in location 0, and the program resumes operation in location 1 with the interrupt disabled. 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 with the interrupt re-enabled. Exit from the interrupt program, back to the original program, can be accomplished by a JMP I 0 instruction.

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 locations 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, just before control is transferred back to the original program. In this application, the ION instruction immediately precedes the last instruction in the routine. A delay of one instruction (regardless of the execution time of the following instruction), is inherent in the ION instruction to allow transfer of program control back to the original program before enabling the interrupt. Exit from the subroutine usually is accomplished by a JMP I 0 instruction.

The ION command can also be given during the subroutine as soon as the I/O device causing the interrupt has been identified. This latter method allows the subroutine which is handling a low priority interrupt to be interrupted, possible 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 contents of the program counter stored in location 0; so that the return address to the original program is not lost if another interrupt occurs.

Instructions

Interrupt Turn ON (ION)

Octal Code: 6001

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.

Skip If Interrupt ON (SKON)

Octal Code: 6000

Operation: The state of the interrupt enable flip-flop is tested. If this flip-flop is set, the next sequential instruction is skipped. Simultaneously with this test (and before a device flag can cause an interrupt) the interrupt system is turned off as described under the following IOF instruction.

Interrupt Turn Off (IOF)

Octal Code: 6002

Operation: This command causes the program interrupt feature to be disabled.

Skip If Interrupt Request (SRQ)

Octal Code: 6003

Operation: The state of the internal interrupt request bus is tested. If it is low, indicating one or more devices are requesting an interrupt, the next instruction is skipped.

Flag Processing IOTs

Get Flags (GTF)

Operation: 6004

Octal Code: The following machine states are read into the indicated bits of the accumulator:

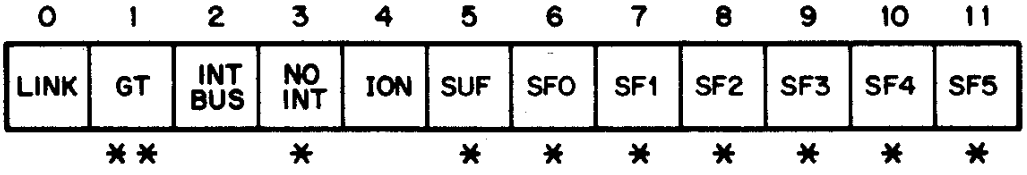


Figure 3-8 Flag Processing States

* Only if Extended Memory Control, type KM8-E, installed.

** Only if Extended Arithmetic Element (EAE), type KE8-E, installed.

Restore Flags (RTF)

Octal Code 6005

Operation This instruction is the converse of GTF. The bits in the AC (see figure 3-8) are used to set the corresponding flip-flops in the processor, the KM8-E Extended Memory Control or the KE8-E Extended Arithmetic Element. RTF enables the interrupt in the same manner as an ION instruction. Refer to the KM8-E Memory Extension and Time-Share Option in Chapter 7 for more details on the RTF instruction.

Skip if Greater Than (SGT)

Octal Code: 6006
Operation: If the GT flag is set, the next instruction is skipped. This instruction is implemented only if the KE8-E is installed.

Clear all Flags (CAF)

Octal Code: 6007
Operation: This instruction is logically equivalent to operating the CLEAR key on the panel. It generates INITIALIZE on the OMNIBUS and at the external I/O interface. The LINK and AC are cleared. The action of INITIALIZE is a function of the design of each peripheral, but generally INITIALIZE clears flags and motion control flip-flops, and sets the interrupt enable flip-flop of all peripherals.

NOTE

A CAF instruction should not be given when a device is active. For example: A CAF instruction should not be given until at least 100ms after a TLS instruction.

INSTRUCTION SUMMARY

A summary of the common usage of the eight basic instructions is provided in Table 3-1. As the reader continues on with the remaining chapters in this handbook, he should keep in mind the capability of all eight instructions since the processor's operation is determined by the use of each instruction. A program, for instance, is simply a collection of instructions strung out in a particular order for a particular purpose such as solving a problem. Each instruction performs a series of steps to satisfy the requirements of the program.

Table 3-1
Basic Instruction Usage Summary

BASIC INSTRUCTION	MINIMUM NO. OF CYCLES	COMMON USAGE	EXAMPLE OF USAGE
AND (0)(octal)	2	Data Manipulation	Strips unwanted bits from the AC.
TAD (1)(octal)	2	Data Manipulation	Provides arithmetic addition. Also serves to load the AC with the contents of some memory location.
ISZ (2)(octal)	2	Program Loops	Used for counting.
DCA (3)(octal)	2	Data Manipulation	Used when placing data into some memory location.
JMS (4)(octal)	2	Subroutine Entry	Provides entry to subroutines.
JMP (5)(octal)	1	Manipulation of Program Counter	Allows the programmer to go to a different portion of his program. Also provides subroutine exit.
IOT (6)(octal)	1	External Communication	Allows the program to converse with peripherals.
OPR (7)(octal)	1	Testing of AC and Link	Operates on and/or tests the contents of the AC and Link. Operates on the contents of the MQ Register.

CHAPTER 4
PDP-8/E PROGRAMMING SYSTEMS

GENERAL

This chapter deals with the concepts required to program the PDP-8/E and identifies the system programs available to the user. Two handbooks, INTRODUCTION TO PROGRAMMING and PROGRAMMING LANGUAGES, provide a more detailed treatment and description of the commonly used programming languages and programming systems.

The chapter is divided into 2 sections. Section 1 provides basic programming guidelines and section 2 identifies the various programming systems and commonly used languages available to the user.

SECTION 1

PDP-8/E PROGRAMMING FUNDAMENTALS

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

Total locations (decimal)	0-4095 or 4096
Total addresses (octal)	0-7777 or 10,000
Number of pages (decimal)	0-31 or 32
Page designations (octal)	0-37 or 40
Number of locations per page (decimal)	0-127 or 128
Addresses within a page (octal)	0-177 or 200

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

Address	Purpose
0 (octal)	Stores the contents of the program counter following a program interrupt.
1 (octal)	Stores the first instruction to be executed following a program interrupt.

10 (octal)—17 (octal) Auto-indexing.

MEMORY ADDRESSING

The programmer has 4096 (decimal) locations which he may address. However, as illustrated in Figure 3-2 of Chapter 3, when an instruction is fetched from memory, only bits 5 through 11 contain the address of the data. Addressing is accomplished using octal notation. Therefore the 4096 possible locations require addresses in octal from 0000 to 7777. This means that a total of 12 bits is required to specify an absolute address. So that all locations may be addressed as efficiently as possible, memory is addressed in terms of pages with a coding scheme that allows easy access to any one of the 10,000 octal locations. The page addressing scheme is illustrated in Figure 4-1 which shows the relationship of the 40 octal pages with the 10,000 octal locations. The programmer is interested in only three pages in memory at any one time:

a. The current page
b. Page 0
c. A location on other than the current page or page 0.

Page 0 is used to store commonly used operands and off-page pointers. For instance, the location of an indirect address used by instructions is usually on Page 0.

ABSOLUTE ADDR. (OCTAL)	CORE MEMORY PAGE (OCTAL)	PAGE ADDR. (OCTAL)
7777 7600	37	177 000
7577 7400	36	177 000
7377 7200	35	177 000
7177 7000	34	177 000
6777 6600	33	177 000
6577 6400	32	177 000
6377 6200	31	177 000
6177 6000	30	177 000
5777 5600	27	177 000
5577 5400	26	177 000
5377 5200	25	177 000
5177 5000	24	177 000
4777 4600	23	177 000
4577 4400	22	177 000
4377 4200	21	177 000
4177 4000	20	177 000
3777 3600	17	177 000
3577 3400	16	177 000
3377 3200	15	177 000
3177 3000	14	177 000
2777 2600	13	177 000
2577 2400	12	177 000
2377 2200	11	177 000
2177 2000	10	177 000
1777 1600	7	177 000
1577 1400	6	177 000
1377 1200	5	177 000
1177 1000	4	177 000
0777 0600	3	177 000
0577 0400	2	177 000
0377 0200	1	177 000
0177 0000	0	177 000

Figure 4-1 Memory Addressing Scheme

The format of the Central Processor (CPMA) Memory Address Register establishing the memory page must first be considered. The MA register is an unequally divided 12-bit register in which the least significant bits of the MA (bits 5-11) are called the page address bits and the most significant bits (bits 0-4) are called the page bits. The 12 bits of the Memory Address are established by the program counter (PC) and are re-established each time the PC loads an absolute address for the next instruction. Bits 0-4 are used to establish the memory page as shown in Figure 4-2. Because the pages to be addressed include pages 0-37 (octal), only five bits are required. The first two bits represent numbers from 0 to 3 and the next three represent numbers from 0 to 7. Because the locations to be addressed on any given page include locations 0-177 (octal), only seven bits are required to specify any one location. Bit 5 represents an octal 1 or 0; bits 6, 7, and 8 represent the second octal digit from 0 to 7; bits 9, 10, and 11 represent the last octal digit from 0 to 7. Thus, on the example shown in Figure 4-2, the MA register is addressing page 5, location 73 (absolute address 1273).

When the user first receives his PDP-8/E, he should assume that it has no information content in its memory. Before he can load instructions into memory, he must first perform the initializing and loading procedures described in Chapter 2. The following discussion assumes that the preliminary procedures have been completed and that the programmer now wants to load into core those instructions which will be called upon after a program has been written. His main concern is to decide in which memory locations he desires to place his instructions and in which locations he wishes to place the corresponding data.

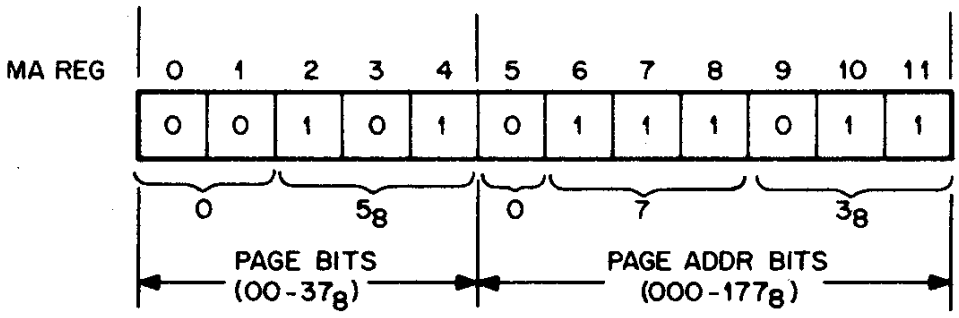


Figure 4-2 Format Establishing Memory Page

Initially, the programmer must load the Central Processor Memory Address Register with an address and then deposit a 12-bit instruction word in the format shown in Figure 4-3.

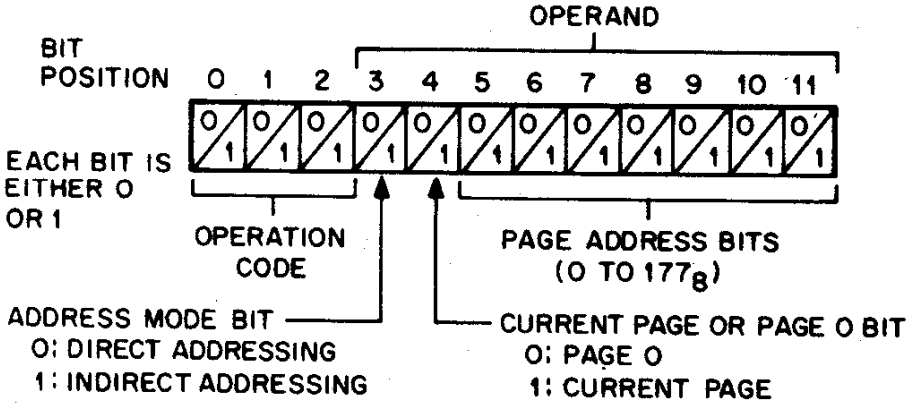


Figure 4-3 Format of a Memory Reference Instruction

The first three bits contain the instruction operation code and have nothing to do with addressing. The last seven bits address the location (from 0 to 177 octal), which will contain either data or a 12-bit address. Those address bits are located in the Memory Buffer Register, and are ineffective until bits 3 and 4 are decoded, at which time the address bits are transferred from the Memory Buffer to the Central Processor Memory Address Register. The page address (the first five bits of the MA register) is determined by whether bit 4 is a 0 or 1 (see Figure 4-4).

Where to place the instruction word or Data word is a very important consideration. At this point, the programmer has three choices in the location of the data:

a. the current page (that page containing the Instruction)
b. page 0
c. a page other than page 0 or the current page.

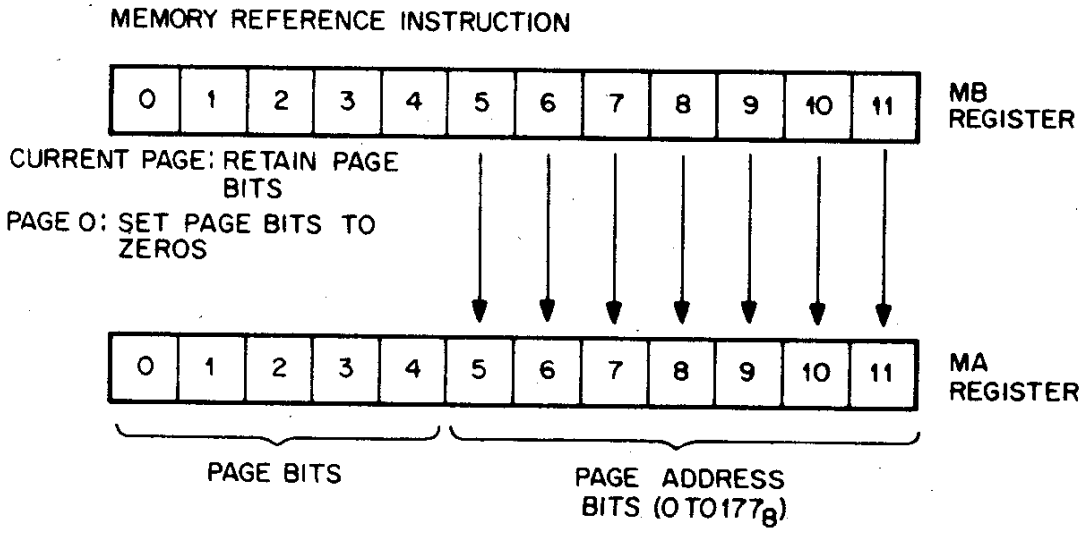


Figure 4-4 Transfer of Address Portion of MRI into MA Register

Current Page—If the programmer desires the data to be located in the current page, he must make bit 4 a 1 in the original instruction word. The logic within the processor causes the first five bits of the MA to remain, and transfers the last seven bits of the MB (the new address within a page) to the last seven bits of the MA Register. This method of updating the MA Register is illustrated in Figure 4-4.

Page 0—Page 0 is commonly used to store operands or address of operands or routines. The programmer must set bit 4 word to 0. The logic within the processor then places all zeros in MA bits 0 through 4 and transfers the content of the last 7 bits of the MB register to the last 7 bits of the MA Register. Thus, the page address is now page 0 and the address within page 0 is some address between 0 and 177 octal. This is illustrated in Figure 4-4.

Addressing A Page Other Than the Current Page or Page 0—The programmer may address a page other than the current page or page 0 by placing a 1 in bit 3 of the original instruction word. As before, the computer then goes to an address on the current page or on page 0, depending on the state of bit 4. The logic within the processor responds to bit 3 being a 1 by going into a defer state for a new address. This procedure is called "Indirect Addressing."

INDIRECT ADDRESSING

In the preceding section, the method of directly addressing 400(octal) memory locations by an MRI was described—namely those on page 0 and those on the current page. This section describes the method for addressing the other 7400(octal) memory locations. Bit 3 of an MRI designates the address mode. When bit 3 is a 0, the operand is a direct address. When bit 3 is a 1, the operand is an indirect address. An indirect address (pointer address) identifies the location that contains the desired address (effective address). To address a location that is not directly addressable, the absolute address of the desired location is stored in one of the 400(octal) directly addressable locations (pointer address); the pointer address is written as the operand of the MRI; and the letter I is written between the mnemonic and the operand. (During assembly, the presence of the I results in bit 3 of the MRI being set to 1.) Upon execution, the MRI will operate on the contents of the location identified by the address contained in the pointer location.

The two examples in Figure 4-5 illustrate the difference between direct addressing and indirect addressing. The first example shows a TAD instruction that uses direct addressing to get data stored on page 0 in location 50; the second is a TAD instruction that uses indirect addressing, with a pointer on page 0 in location 50, to obtain data stored in location 1275. (When references are made to them from various pages, constants and pointer addresses can be stored on page 0 to avoid the necessity of storing them on each applicable page.) The octal value 1050, in the first example, represents direct addressing (bit 3 = 0); the octal value 1450, in the second example, represents indirect addressing (bit 3 = 1). Both examples assume that the accumulator has previously been cleared.

Location	Content	
200	TAD 50	(TAD 50 = 1050_8)
.	.	Instruction
.	.	
50	1275	Data (Number) To Be Acted Upon By
.	.	Instruction Address
.	.	
1275	20	(Content of location 1275 is not used in the execution of the instruction in location 200.)

NOTE: AC = 1275 after executing the instruction in location 200

Location	Content	
200	TAD I 50	(TAD I 50 = 1450_8)
.	.	
.	.	Designates Indirect Addressing
.	.	Instruction
50	1275	Pointer Address
.	.	
.	.	
1275	20	Data (Number) To Be Acted Upon By Instruction Effective Address

NOTE: AC = 20 after executing the instruction in location 200.

Figure 4-5 Comparison of Direct and Indirect Addressing

The following three examples illustrate some additional ways in which indirect addressing can be used. As shown in example 1, indirect addressing makes it possible to transfer program control from off page 0 (or any other page) to any desired memory location. (Similarly, indirect addressing makes it possible for other memory reference instructions to address any of the 4,096(10) memory locations.) Example 2 shows a DCA instruction that uses indirect addressing with a pointer on the current page. The pointer in this case designates a location off the current page (location 227) in which the data is to be stored. (A pointer address is normally stored on the current page when all references to the designated location are from the current page.) Indirect addressing provides the means for returning to a main program from a subroutine, as shown in example 3. Indirect addressing is also effectively used in manipulating tables of data.

EXAMPLE 1

Location	Content	
75	JMP I 100	(JMP I 100 = 5500(octal))
.	.	
.	.	Designates Indirect Addressing Instruction

100	6000	Pointer Address
.	.	
.	.	
.	.	
6000	DCA 6100	Next Instruction To Be Executed
.	.	
.	.	

NOTE: Execution of the instruction in location 75 causes program control to be transferred to location 6000, and the next instruction to be executed is the DCA 6100 instruction.

EXAMPLE 2

Location	Content	
450	DCA I 577	(DCA I 577 = 3777(octal)) Designates Indirect Addressing Instruction
577	277	Pointer Address
.	.	
.	.	
227	nnnn	Data (Number) Stored By Instruction (Effective Address)

NOTES: 1. Memory Location 577 is location 177 of current page. Execution of the instruction in location 450 causes the contents of the accumulator to be stored in location 227.

EXAMPLE 3

Location	Content	
207	JMS I 70	(JMS I 70 = 4470(octal))
210	TAD 250	(The next instruction to be executed upon return from the subroutine.)
.	.	
.	.	
70	2000	(Starting address of the subroutine stored here.)
.	.	
.	.	
2000	aaaa	(Return address stored here by JMS instruction.)
2001	i i i	(First instruction of subroutine.)
.	.	
.	.	
2077	JMP I 2000	(Last instruction of subroutine.)

NOTES: 1. Execution of the instruction in location 207 causes the address 210 to be stored in location 2000 and the instruction in location 2001 to be executed next. Execution of the subroutine proceeds until the last instruction (JMP I 2000) causes control to be transferred back to the main program, continuing with the execution of the instruction stored in location 210.

2. A JMS instruction that uses indirect addressing is useful when the subroutine is too large to store on the current page.
3. Storing the pointer address on page 0 enables instructions on various pages to have access to the subroutine.

PROGRAMMING OPERATIONS

The programmer can make use of any combination of instructions. The following sections describe the more common programming operations.

STORING AND LOADING

Data is stored in any core memory location by use of the DCA (Deposit & Clear AC) instruction. This instruction clears the AC to simplify loading of the next data. If the data deposited is required in the AC for the next program operation, the DCA must be followed by a TAD for the same address. All loading of core memory information into the AC is accomplished by means of the TAD instruction.

The DCA instruction stores the contents of the AC in the referenced location, destroying the original contents of the location. The AC is then set to all zeroes. The following example shows the contents of the accumulator, link, and location 225 before and after executing the instruction DCA 225.

DCA 225

	AC	Link	Loc. 225
Before Execution	1234	1	7654
After Execution	0000	1	1234

The following facts should be kept in mind when using the DCA instruction:

a. The state of the link bit is not altered.
b. The AC is cleared.
c. The original contents of the addressed location are replaced by the contents of the AC.

PROGRAM CONTROL

The Program Counter is used to direct the processor to the next address of the next instruction to be fetched. Therefore, the content of the PC Register is always one more than the content of the Central Processor Memory Address (CPMA) Register. When an instruction has been completed and the processor is ready to go into a new fetch, the content of the Program Counter is transferred into the CPMA Register and the Program Counter with its original address is incremented by +1, thereby pointing to the next sequential address. This procedure is called Program Control because it directs the processor to the next instruction. Because this rigid sequence of instructions is not always desirable for practical programming, the PDP-8/E provides a means of jumping out of this sequence to transfer Program Control from one sequence of instructions to another or to allow the processor to enter a subroutine which is itself a sequence of instructions and re-enter the main program when the subroutine has been completed.

Transfer of program control to any core memory location uses the JMP or JMS instructions. The JMP I and JMS I (indirect address, bit 3 = 1) are 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 contents of the PC are stored in the specified address Y, and address Y + 1 is transferred into the PC. Subroutines or other pages may be entered via an indirect JMS. To exit a subroutine, the last instruction is a JMP I Y, which returns program control to the location stored in Y.

The JMP instruction loads the effective address of the instruction into the program counter, thereby changing the program sequence since the PC specifies the next instruction to be performed. In the following example, execution of the instruction in location 250 (JMP 300) causes the program to jump over the instructions in locations 251 through 277 and immediately transfer control to the instruction in location 300.

Location	Content	
250	JMP 300	(This instruction transfers program control to location 300.)
.	.	
.	.	
300	DCA 300	
.	.	

NOTE: The JMP instruction does not affect the contents of the AC or link.

A program written to perform a specific operation often includes sets of instructions which perform intermediate tasks. These intermediate tasks may be finding a square root, or typing a character on a keyboard. Such operations are often performed many times in the running of one program and may be coded as subroutines. To eliminate the need of writing the complete set of instructions each time the operation must be performed, the JMS (jump to subroutine) instruction is used. The JMS instruction stores a pointer address in the first location of the subroutine and transfers control to the second location of the subroutine. After the subroutine is executed, the pointer address identifies the next instruction to be executed. Thus, the programmer has at his disposal a simple means of exiting from the normal flow of his program to perform an intermediate task and a means of returning to the correct location upon completion of the task. (This return is accomplished using indirect addressing, which is discussed elsewhere in this chapter.)

The following example illustrates the action of the JMS instruction:

Location	Content	
PROGRAM		
200	JMS 350	(This instruction stores 0201 in location 350 and transfers program control to location 351.)

```
201             DCA 270         (This instruction stores the contents
                                of the AC in location 270 upon return
                                from the subroutine.)
 .               .
 .               .
 .               .
SUBROUTINE
350             0000            (This location is assumed to have an
                                initial value of 0000; after JMS 350
                                is executed, it is 0201.)
351             i i i           (First instruction of subroutine)
 .               .
 .               .
 .               .
375             JMP I 350       (Last instruction of subroutine)
```

The following should be kept in mind when using the JMS:

1. The value of the PC (the address of the JMS instruction +1) is always stored in the first location of the subroutine, replacing the original contents.
2. Program control is always transferred to the location designated by the operand + 1 (second location of the subroutine).
3. The normal return from a subroutine is made by using an indirect JMP to the first location of the subroutine (JMP I 350 in the above example); (Indirect addressing, as discussed in this chapter effectively transfers control to location 201).
4. When the results of the subroutine processing are contained in the AC and are to be used in the main program, they must be stored upon return from the subroutine before further calculations are performed. (In the above example, the results of the subroutine processing are stored in location 270.)

ARITHMETIC OPERATIONS

One arithmetic instruction is included in the order code, the two's complement add (TAD). 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 are performed in accordance with the common rules of binary arithmetic. In the PDP-8/E, as in other machines utilizing complementation techniques, negative numbers are represented as the complements 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 1s are replaced by 0s and all 0s are replaced by 1s. The two's complement of a number is equal to one plus the one's complement of the number.

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.

The TAD instruction (see Figure 4-6) performs a binary addition between the specified data word and the contents of the accumulator, leaving the result of the addition in the accumulator. If a carry out of the most significant bit of the accumulator should occur, the state of the link bit is complemented. The add instruction is called a Two's Complement Add to remind the programmer that negative numbers must be expressed as the two's complement of the positive value.

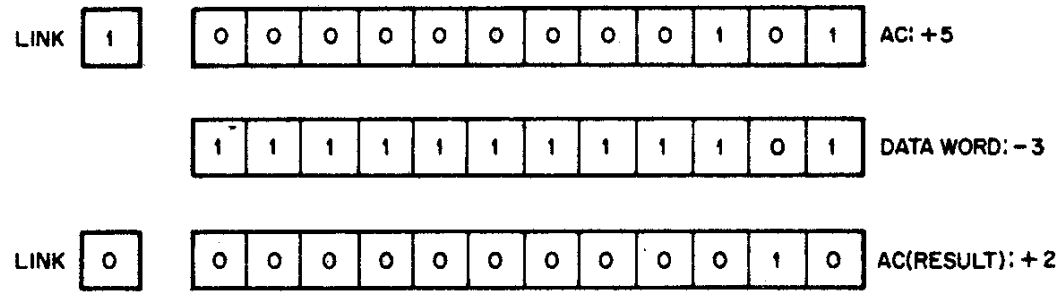


Figure 4-6 Operation of the TAD Instruction

The following points should be remembered when using the TAD instruction:

a. Negative numbers must be expressed as a two's complement of the positive value of the number.
b. A carry out of the accumulator will complement the link.
c. The data word in the referenced location is not affected.

LOGIC OPERATIONS

The PDP-8/E instruction list includes the logic instruction AND. A short routine can be written from this instruction to perform the exclusive OR operation.

Logical AND

The logical AND operation between the contents of the accumulator and the contents of a core memory location Y is performed directly by means of the AND Y instruction. The logical AND performs an AND operation with AC0 and MB0, AC1 and MB1, etc. The result remains in the AC, the original contents of the AC are lost, and the contents of location Y are unaffected.

The AND instruction causes a bit-by-bit Boolean AND operation between the contents of the accumulator and the data word specified by the instruction. The result is left in the accumulator as shown in Figure 4-7.

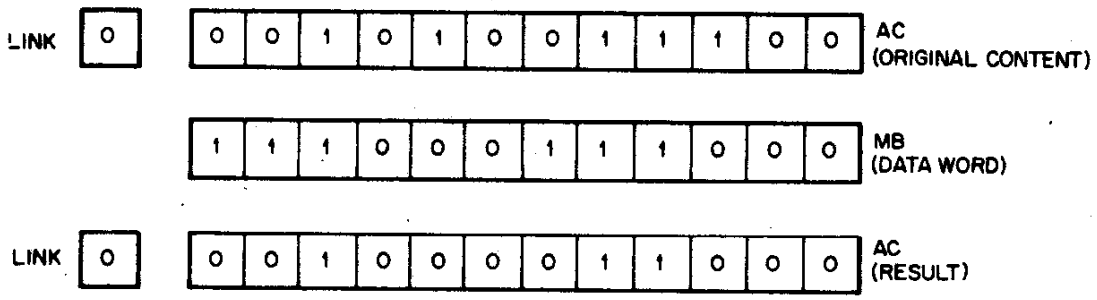


Figure 4-7 Operation of the AND Instruction

The following points should be noted with respect to the AND instruction:

a. A 1 appears in the AC only when a 1 is present in both the AC and the data word (The data word is often referred to as a mask).
b. The state of the link bit is not affected by the AND instruction.
c. The data word in the referenced location is not altered.

Inclusive OR

The Inclusive OR instruction makes use of the MQ register, which is a permanent part of the PDP-8/E. Assuming that 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)
/ENTER WITH ARGUMENT IN AC; EXIT WITH
/LOGICAL RESULT IN AC

/ADDRESS LABEL INSTRUCTION          REMARKS
    IOR,       0
               MQL                 /AC TO MQ, CLEAR AC
               TAD I IOR           /GET ADDRESS OF SECOND
               DCA TEM1            ARGUMENT
               TAD I TEM1
               MQA                 /IOR MQ TO AC
               ISZ IOR
               JMP I IOR
   TEM1,       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 instead of the address of B, in the second instruction of the calling sequence; however, the resulting subroutine is not as useful as the subroutine given here.

INDEXING OPERATIONS

External events can be counted by the program, and the count 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 contents of the accumulator. Counting a specified number of operations is performed by storing a two's complement negative number equivalent to the number of operations to be counted. Each time the operation is performed, the ISZ instruction is used to increment the contents of this stored number and to 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 contents of a memory location are incremented without disturbing the contents 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.

The ISZ instruction adds a 1 to the referenced data word and then examines the result of the addition. If a zero result occurs, the instruction following the ISZ is skipped. If the result is not zero, the instruction following the ISZ is performed. In either case, the result of the addition replaces the original data word in memory. The example below illustrates one method of adding the contents of a given location to the AC a specified number of times (multiplying) by using an ISZ instruction to increment a tally. The effect of this example is to multiply the contents of location 275 by 2. (To add the contents of a given location to the AC

twice, using the ISZ loop, as shown below, requires more instructions than merely repeating the TAD instruction or using a rotate instruction. However, when adding the contents four or more times, use of the ISZ loop requires fewer instructions.) In the first pass of the example, execution of ISZ 250 increments the contents of location 250 from 7776 to 7777 and then transfers control to the following instruction (JMP 200). In the second pass, execution of ISZ 250 increments the contents of location 250 from 7777 to 0000 and transfers control to the instruction in location 203, skipping over location 202.

CODING FOR ISZ LOOP

Location	Content
200	TAD 275
201	ISZ 250
202	JMP 200
203	DCA 276
.	.
.	.
.	.
250	7776
.	.
.	.
.	.
275	0100
276	0000

SEQUENCE OF EXECUTION FOR ISZ LOOP

Location	Content	Content After Instruction Execution AC	250	275	276
FIRST PASS					
200	TAD 275	0100	7776	0100	0000
201	ISZ 250	0100	7777	0100	0000
202	JMP 200	0100	7777	0100	0000
SECOND PASS					
200	TAD 275	0200	7777	0100	0000
201	ISZ 250	0200	0000	0100	0000
202	JMP 200	(Skipped	during	second	pass)
203	DCA 276	0000	0000	0100	0200

ISZ Instruction Incrementing a Tally

The following points should be kept in mind when using the ISZ instruction:

a. The contents of the AC and link are not disturbed.
b. The original word is replaced in main memory by the incremented value.
c. When using the ISZ for looping a specified number of times, the tally must be set to the negative of the desired number.
d. The ISZ performs the incrementation first and then checks for a zero result.

CODING A PROGRAM

The introduction of an assembler in Chapter 2 enabled the programmer to write a symbolic program using meaningful mnemonic codes rather than the octal representation of the instructions. The programmer could now write mnemonic programs such as the following example, which multiplies 18(10) by 36(10) using successive addition.

```
200     CLA CLL     Initialize
201     TAD 210     Set up a Tally
202     CIA         equal to —18(10) to
203     DCA 212     count the additions of 36
204     TAD 211     Add 36
205     ISZ 212     Skip if Tally is 0
206     JMP 204     Add another 36 if not done
207     HLT         Stop after 18 times
210     0022        Equal to 18(10)
211     0044        Equal to 36(10)
212     0000        Holds the tally
```

Writing the above program was greatly simplified because mnemonic codes were used for the octal instructions. However, writing down the absolute address of each instruction is clearly an inconvenience. If the programmer later adds or deletes instructions, thus altering the location assignments of his program, he has to rewrite those instructions whose operands refer to the altered assignments. If the programmer wishes to move the program to a different section of memory, he must rewrite the program. Since such changes must be made often, especially in large programs, a better means of assigning locations is needed. The assembler provides this better means.

Location Assignment

As in the previous program example, most programs are written in successive memory locations. If the programmer assigned an absolute location to the first instruction, the assembler could be told to assign the next instructions to the following locations in order. In programming the PDP-8/E the initial location is denoted by a precedent asterisk (*). The assembler maintains a current location counter by which it assigns successive locations to instructions. The asterisk causes the current location counter to be set to the value following the asterisk. With this improvement incorporated, and with the use of symbolic addresses, the previous example appears as shown in the following example.

```
*200
START,  CLA   CLL
        TAD   A
        CIA
        DCA   TALLY
        TAD   B
        ISZ   TALLY
        JMP   START +4
        HLT
    A,  0022
    B,  0044
TALLY,  0000
```

NOTE: In this example, CLA CLL is stored in location 200 and the successive instructions are stored in 201, 202, etc.

WRITING SUBROUTINES

Included in the memory reference instructions, given in Chapter 3, was the instruction JMS (jump to subroutine). This instruction is a modified JMP command which makes possible a later return to the point of departure from the main program. The JMS instruction automatically stores the location of the next instruction after the JMS in the location to which the program is instructed to jump, thereby enabling a return.

The programmer need only terminate the subroutine with an indirect JMP to the first location of the subroutine in order to return to the next instruction following the JMS instruction. The following simple program illustrates the use of a subroutine to double a number contained in the accumulator.

(Main Program)

```
START,    CLA CLL
          TAD N              Get the number in the AC
          JMS DOUBLE         Jump to subroutine to double N
          DCA TWON           First instruction after the subroutine
            .
            .
            .
N,        nnnn               Any number, N
TWON,     nnnn               2N will be stored here
```

Subroutine

```
DOUBLE,   0000
          CLL RAL            Rotate left, multiplying by 2
          SNL                Did overflow occur?
          JMP I DOUBLE
          RAR                If overflow occurs, display the num-
                             ber to be doubled in the AC and
          HLT                then stop the computer.
```

Notice that the first instruction of the subroutine is located in the second location of the subroutine. Any instruction stored in location DOUBLE would be lost when the return address is stored. Also note that the subroutine as it is written must be located on page 0 or current page, *because it is directly addressed. (A subroutine is often located on another* page and addressed indirectly as the next example demonstrates.)

The following program multiplies a number in the accumulator by a number stored in the location immediately following the JMS instruction.

Main Program

```
*200
START,          TAD A
                DCA .+3
                TAD B
                JMS I 30
                0000
                DCA PRDUCT
                .
                .
                .
PRDUCT,         0000
A,              0051
B,              0027
*30

                MULT
```

Subroutine

```
*6000
MULT,           0000
                CIA
                DCA MTALLY
                TAD I MULT
                ISZ MTALLY
                JMP .-2
                ISZ MULT
                JMP I MULT
MTALLY,         0000
```

The preceding example illustrates the following important points.

a. The JMS I 30 instruction could be used anywhere in core memory to jump to this subroutine because the pointer word (stored in location 30) is located on page 0, and all pages of memory can reference page 0.
b. The period was used to denote the current location in the instructions DCA .+3 and JMP .—2.
c. Since the result of the subroutine is left in the AC when jumping back to the main program, the next instruction should store the result for future use.
d. The first instruction of the subroutine is in location MULT + 1 since the next address in the main program is stored in MULT by the JMS instruction.
e. The first two instructions of the subroutine set the tally with the negative of the number in the AC.
f. The second number to be multiplied is brought into the subroutine by the TAD I MULT instruction, as it is stored in the location specified by the address that the JMS instruction automatically stores in the first location of the subroutine. This is a common technique for transferring information into a subroutine.

g. The ISZ MTALLY instruction is used in the subroutine to count the number of additions. The ISZ MULT instruction is used to increment the contents of MULT by one, thereby making the return jump (JMP I MULT) proceed to the next instruction after the location that held the number to be multiplied.

h. An interesting modification of the previous program is achieved through defining a "new operation" MLTPLY by including in the coding the statement MLTPLY = JMS I 30. The assembler would make a replacement in such a way that any time the programmer writes MLTPLY the computer would perform a jump to the subroutine and return to the program with the product in the AC.

ADDRESS MODIFICATION

A very powerful tool often used by the programmer is address modification, meaning the inclusion of instructions in a program to modify the operand portion of a memory reference instruction. It is a particularly useful technique when working with large blocks of stored data as illustrated by the two programs that follow (see Figure 4-8).

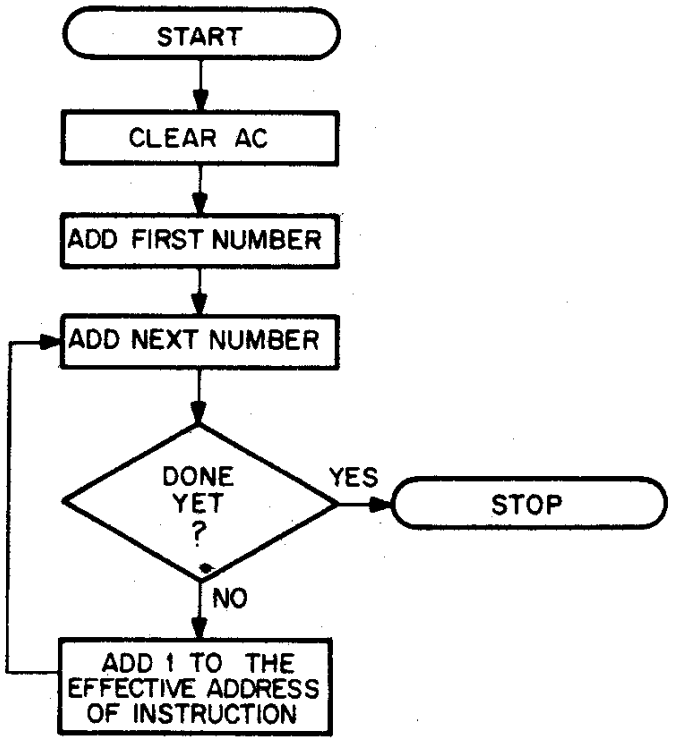


Figure 4-8 Address Modification

```
*200
START,                  CLA CLL
                        TAD K200
                        CIA
                        DCA TALLY
                        TAD K4000
                        DCA NUM
                        TAD K4200
                        DCA RESULT
```

```
AGAIN,          TAD I NUM
                JMS SQUARE
                DCA I RESULT
                ISZ RESULT
                ISZ NUM
                ISZ TALLY
                JMP AGAIN
                HLT
K200,           0200
TALLY,          0000
K4000,          4000
NUM,            0000
K4200, 4200
RESULT,         0000
*300
SQUARE,         0000
                DCA STORE
                TAD STORE
                CIA
                DCA COUNT
                TAD STORE
                ISZ COUNT
                JMP .-2
                JMP I SQUARE
STORE,          0000
COUNT,          0000
$
```

It will be noted that the first eight instructions are concerned with initializing the program. This initializing enables the stored program to be restarted several times and still operate on the correct locations. If the program had merely incremented locations K4000 and K4200 and utilized those locations for indirect addressing, it would only operate on the correct locations on the first running. On successive runnings, the program would be operating on successively higher locations in memory. With the program written as shown, however, the pointer words are automatically reset. This procedure is often referred to as "housekeeping."

LOOPING A PROGRAM

As many examples have already shown, the use of a program loop, in which a set of instructions is performed repeatedly, is common programming practice. Looping a program is one of the most powerful tools at the programmer's disposal. It enables him to perform similar operations many times using the same instructions, thus saving memory location because he need not store the same instructions many times. Looping also makes a program more flexible because it is relatively easy to change the number of loops required for varying conditions by resetting a counter. It should be remembered that looping is little more than a jump to an earlier part of the program; however, the jump is usually controlled by changing program conditions.

There are two basic methods of creating a program loop. The first me-

thod is using an ISZ (2nnn(octal)) instruction to count the number of passes made through the loop. The ISZ is usually followed by a JMP instruction to the beginning of the loop. This technique is very efficient when the required number of passes through the loop can be readily determined.

The second technique is to use the Group 2 Operate Microinstructions to test conditions other than the number of passes which have been made. Using this second technique, the program is required to loop until a specific condition is present in the accumulator or link bit, rather than until a predetermined number of passes are made.

To illustrate the use of an ISZ instruction in a program loop situation, consider the following program which simply sets the contents of all addresses from 2000 to 2777 to zero.

```
*200
CLEAR,      CLA
            TAD CONST
            DCA COUNT           /SET COUNT TO —1000.
            TAD TTABLE
            DCA STABLE          /SET STABLE TO 2000.
            DCA I STABLE        /CLEAR ONE LOCATION.
            ISZ STABLE          /SELECT NEXT LOCATION.
            ISZ COUNT           /IS OPERATION COMPLETE?
            JMP .—3             /NO: REPEAT.
            HLT                 /YES: HALT.
CONST,      7000                /2'S COMP OF 1000.
COUNT,      0
TTABLE,     TABLE
STABLE,     0                   /POINTER TO TABLE.
*2000
TABLE,      0
$
```

Several points should be carefully noted.

a. The first five instructions initialize the loop, but are not in it. The location COUNT is set to —1000 at the beginning, and 1 is added to it during each passage of the loop. After the 1000th (octal) passage, COUNT goes to zero, and the program skips the JMP instruction, and executes the HLT instruction. On each previous occasion, it executed the JMP instruction.

b. In the list of constants following the HLT instruction, TTABLE contains TABLE, which is defined below as having the value 2000, and containing 0. Therefore, STABLE contains 2000 initially. In order to understand this point, it must be remembered that an asterisk character causes the first location after the asterisk to be set to the value after the asterisk. Therefore, *in the previous example CLEAR equals 200 and TABLE equals* 2000.

c. ISZ STABLE adds 1 to the contents of location STABLE, forming 2001 on the first pass, 2002 on the second pass, and so on. Since it never reaches zero, it will never skip. This is a very

common use. It is said to be indexing the addresses from 2000 to 2777. (When using an ISZ instruction in this way, the programmer must be certain that it does not reach 0. Follow the ISZ instruction with a NOP if it does reach 0 so that the resulting skip will not modify the program sequence.)

d. For every ISZ instruction used in a program, there must be two initializing instructions before the loop, and there must be a constant and a counting location in a table of constants. This procedure allows the program to be rerun with the counting locations reset to the correct values.

The following program utilizes a Group 2 skip instruction to create a loop. The program will search all of core memory to find the first occurrence of the octal number 1234.

```
*0
NUMBER,     1234
*200
BEGIN,      CLA CLL
            TAD NUMBER
            CIA
            DCA COMPARE       /STORES MINUS NUMBER.
            DCA ENTRY         /SETS ENTRY TO 0.
REPEAT,     ISZ ENTRY         /INCREASES ENTRY.
            NOP
            TAD I ENTRY       /COMPARISON IS
            TAD COMPARE       /DONE HERE.
            SZA CLA
            JMP REPEAT
            TAD ENTRY
            HLT               /ENTRY IS IN AC.
COMPARE,    0
ENTRY,      0
$
```

This example shows that the program searches itself as well as all other core memory locations, and points up the following points:

a. The ISZ entry instruction is used to index the locations to be tested. The next instruction (NOP) is unnecessary; thus, if ENTRY becomes zero during the course of the program, the program will not be affected. It is important to protect against an ISZ instruction going to zero and skipping a necessary part of a program, if the ISZ is being used simply to index.

b. The number to be searched for is stored in location 0, and the search starts in location 1. Therefore, the program will find at least one occurrence of the number, and will halt after one complete pass through memory, if not before.

c. The program could be modified to bound the area of the search. If the contents of ENTRY are set equal to one less than the desired start location and the number being searched for is put in the location following the last location to be searched, the program will search only the designated area of memory.

d. The program could be restarted at location REPEAT in order to find a second occurrence of 1234 after being halted by the first occurrence.

AUTO-INDEXING

The PDP-8/E computer has eight special registers in page 0; locations 0010 through 0017. Whenever these locations are addressed indirectly by a memory reference instruction, the content of the register is incremented before it is used as the operand of the instruction. These locations can, therefore, be used in place of an ISZ instruction in an indexing application. Because of this, these eight locations are called autoindex registers. Autoindex registers act as any other location when addressed directly. The autoindexing feature is performed only when the location is addressed indirectly.

The following examples below are a modification of the first program example in the preceding section with an autoindex register used in place of the ISZ instruction. (The purpose of the program is to clear memory locations 2000 through 2777.)

Carefully notice the difference between the two examples, especially that TABLE now has to be set to TABLE-1 since this is incremented by the autoindexing register before being used for the first time. This point must be remembered when using an autoindex register. The register increments before the operation takes place; therefore, it must always be set to one less than the first value of the addresses to be indexed.

```
*10
INDEX,      0
*200
CLEAR,      CLA
            TAD CONST
            DCA COUNT
            TAD TTABLE
            DCA INDEX
            DCA I INDEX
            ISZ COUNT
            JMP .-2
            HLT
CONST,      7000
COUNT,      0
TTABLE,     TABLE-1
*2000
TABLE,      0
$
```

The memory search example of the preceding section could also be simplified using an autoindex register as shown below.

```
*0
NUMBER,     1234
*10
ENTRY,      0
*200
BEGIN,      CLA CLL
            TAD NUMBER
```

Notice that in this case ENTRY originally equals 0 because its content is incremented before being used to obtain data for the comparison.

```
            CIA
            DCA COMPARE
            DCA ENTRY
REPEAT,     TAD I ENTRY
            TAD COMPARE
            SZA CLA
            JMP REPEAT
            TAD ENTRY
            HLT
COMPARE,    0
$
```

PROGRAM DELAYS

Because computer development has been primarily sparked by a desire for speed in performing calculations, it seems inconsistent and self-defeating to slow the computer down with program delays. However, there are many occasions when a computer must be told to slow down or to wait for further information. This is because most peripheral equipment, and certainly the human operator, is very much slower than the computer program. A temporary delay may be introduced into the execution of a program when needed by causing the computer to enter one or more futile loops, which it must traverse a fixed number of times before jumping out. It is often necessary to have a computer perform a temporary delay while a peripheral device is processing data to be submitted to the computer. The delays can be accurately timed so as not to waste any more computer time than necessary.

The following is a simple delay routine using the ISZ instruction for an inner loop and an outer loop. When analyzing the example it should be remembered that the PDP-8/E represents only positive numbers up to 3777(octal) or 2047(10). Therefore, the computer counts up to 2047 (10) and then continues to count starting at the next octal number 4000(octal), which the computer interprets as —2048(10). Successive increments of this number will finally bring the count to zero. Thus, a location could be used to count from 1 up to 0 by using an ISZ instruction.

```
                (main program)
                .
                .
                .
            TAD CONST       /START OF DELAY ROUTINE
            DCA COUNT
            ISZ COUNT 1     /INNER
            JMP .—1         /LOOP
            ISZ COUNT
            JMP .—3
CONST,      6030            /SETS DELAY
COUNT,      0
COUNT 1     0
```

PROGRAM BRANCHING

Very few meaningful programs are written which do not take advantage

of the computer's ability to determine the future course the program should follow, based upon intermediate results. The procedure of testing a condition and providing alternative paths for the program to travel for each of the different results possible is called branching a program. The Group 2 microinstructions presented previously are most often used for this purpose. The ISZ instruction often referred to as a conditional skip instruction, also provides a branch in a program. This instruction operates upon the contents of a memory location, while the Group 2 microinstructions test the contents of the AC and L.

A typical example of a conditional skip would be a program to compare A and B and to reverse their order if B is larger than A (see Figure 4-9).

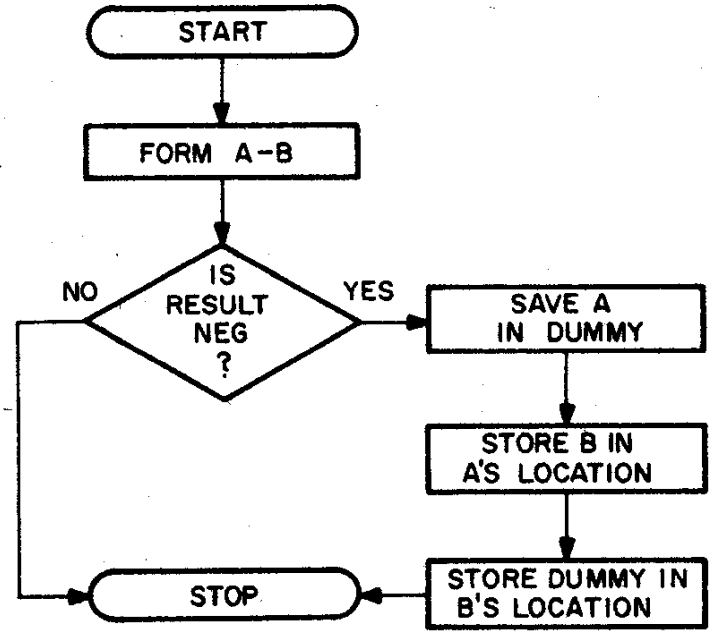


Figure 4-9 Conditional Skip

```
*200
TEST,       CLA CLL
            TAD B          /SUBTRACT B
            CIA            /FROM A
            TAD A          /HERE.
            SMA CLA
            HLT            /STOP HERE IF A IS GREATER
                           /OR EQUAL
            TAD A          /THE REMAINDER OF
            DCA DUMMY      /THE PROGRAM
            TAD B          /DOES THE SWITCH.
            DCA A
            TAD DUMMY
            DCA B
            HLT
A,          1234           /SUBSTITUTE ANY POSITIVE
B,          2460           /VALUES FOR A AND B.
DUMMY,      0
$
```

If A is less than B, their difference will be negative and the HALT will be skipped. The program will proceed to reverse the order of A and B. If A is greater than or equal to B, the program will halt.

MICROPROGRAMMING

Because PDP-8/E instructions of Group 1, Group 2, and Group 3 are determined by bit assignment, these instructions may be combined, or microprogrammed, to form new instructions enabling the computer to do more operations in less time.

Combining Microinstructions

The programmer should make certain that the program clears the accumulator and link before any arithmetic operations are performed. To perform this task, the program might include the following instructions (given in both octal and mnemonic form).

```
CLA        7200 (octal)
CLL        7100 (octal)
```

However, when the Group 1 instruction format is analyzed, Figure 4-10 illustrates the result.

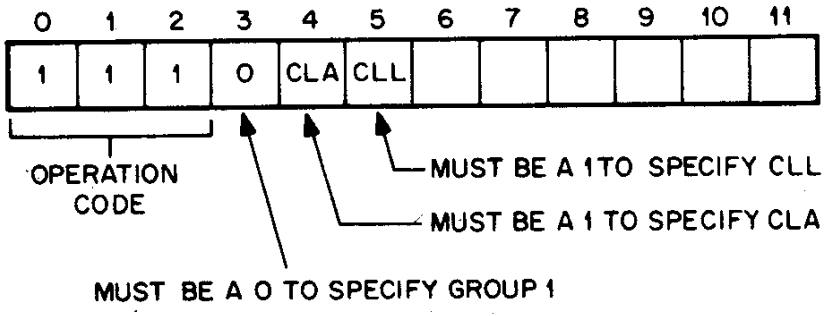


Figure 4-10 Group 1 Bit Assignments

Since the CLA and the CLL instructions occupy separate bit positions, they may be expressed in the same instruction, thus combining the two operations into one instruction. This instruction would be written as follows.

```
CLA CLL    7300 (octal)
```

In this manner, many operate microinstructions can be combined, making the execution of the program much more efficient. The assembler for the PDP-8/E will combine the instructions properly when they are written as above; that is, on the same coding line, separated by a space.

Illegal Combinations

Microprogramming, although very efficient, can also be troublesome for the new programmer. There are many violations of coding which the assembler will not accept.

One rule to remember is: "If you can't code it, the computer can't do it." In other words, the programmer could write a string of mnemonic microinstructions, but unless these microinstructions can be coded

correctly in octal representation, they cannot be performed. To illustrate this fact, suppose the programmer would like to complement the accumulator (CMA), complement the link (CML), and then skip on a nonzero link (SNL). He could write the following.

```
CMA     CML     SNL
```

These instructions require the bit assignments shown in Figure 4-11.

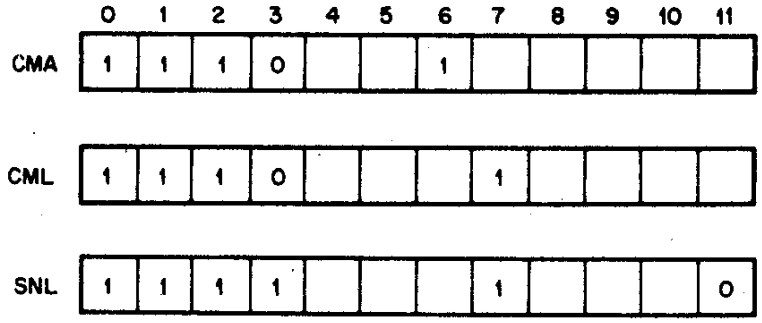


Figure 4-11 Example of Illegal Combinations

The three microinstructions cannot be combined in one instruction because bit 3 is required to be a 0 and a 1 simultaneously. Therefore, no instructions may be used which combine Group 1 and Group 2 microinstructions because bit 3 usage is not compatible. The CMA and CML can, however, be combined because their bit assignments are compatible. The combination would be as follows.

```
CMA  CML     7060 (octal)
```

To perform the original set of three operations, two instructions are needed.

```
CMA CML     7060 (octal)
SNL         7420 (octal)
```

Because Group 1 and Group 2 microinstructions cannot be combined, the commonly used microinstruction CLA is a member of both groups. Clearing the AC is often required in a program and it is very convenient to be able to microprogram the CLA with the members of both groups.

```
RAR         7010 (octal)
RTR         7012 (octal)
```

Although he can write the instruction "RAR RTR," it cannot be correctly converted to octal by the assembler because of the conflict in bit 10; therefore, it is illegal.

Combining Skip Microinstructions

Group 2 operate microinstructions use bit 8 to determine the instruction specified by bits 5, 6, and 7 as previously described. If bit 8 is a 0, the instructions SMA, SZA, and SNL are specified. If bit 8 is a 1, the instructions SPA, SNA, and SZL are specified. Thus, SMA cannot be combined with SZL because of the opposite values of bit 8.

OR GROUP—SMA OR SZA OR SNL

If bit 8 is a 0, the instruction skips on the logical OR of the conditions specified by the separate microinstructions. The next instruction is skipped if any of the stated conditions exist. For example, the combined microinstruction SMA SNL will skip under the following conditions:

a. The accumulator is negative, the link is zero.
b. The link is nonzero, the accumulator is not negative.
c. The accumulator is negative and the link is nonzero.

(It will not skip if all conditions fail.) This manner of combining the test conditions is described as the logical OR of the conditions.

AND GROUP—SPA AND SNA AND SZL

A value of bit 8 = 1 specifies the group of microinstructions SPA, SNA, and SZL, which combine to form instructions that act according to the logical AND of the conditions. In other words, the next instruction is skipped only if all conditions are satisfied. For example, the instruction SPA SZL will cause a skip of the next instruction only if the accumulator is positive and the link is zero. (It will not skip if either of the conditions fail.)

NOTES:
1. The programmer is not able to specify the manner of combination. The SMA, SZA, SNL conditions are always combined by the logical OR, and the SPA, SNA, SZL conditions are always joined by a logical AND.
2. Since the SPA microinstruction will skip on either a positive or a zero accumulator, to skip on a strictly positive (positive, nonzero) accumulator the combined microinstruction SPA SNA is used.

Order of Execution of Combined Microinstructions

The combined microinstructions are performed by the computer in a very definite sequence. When written separately, the order of execution of the instructions is the order in which they are encountered in the program. In writing a combined instruction of Group 1 or Group 2 microinstructions, the order written has no bearing upon the order of execution. This should be clear, because the combined instruction is a 12-bit binary number with certain bits set to a value of 1. The order in which the bits are set to 1 has no bearing on the final execution of the whole binary word.

GROUP 1

1. CLA, CLL—Clear the accumulator and/or clear the link are the first actions performed. They are effectively performed simultaneously and yet independently.
2. CMA, CML—Complement the accumulator and/or complement the link. These operations are also effectively performed simultaneously and independently.
3. IAC—Increment the accumulator. This operation is performed third, allowing a number in the AC to be complemented and then incremented by 1, thereby forming the two's complement, or negative, of the number.
4. RAR, RAL, RTR, RTL, BSW—The rotate instructions are performed last in sequence. Because of the bit assignment previously discussed, only one of the five operations may be performed in each combined instruction.

GROUP 2

1. Either SMA or SZA or SNL when bit 8 is a 0. Both SPA and SNA and SZL when bit 8 is a 1. Combined microinstructions specifying a skip are performed first. The microinstructions are combined to form one specific test; therefore, skip instructions are effectively performed simultaneously.
   Because of bit 8, only members of one skip group may be combined in an instruction.
2. CLA—Clear the accumulator. This instruction is performed second in sequence, allowing different arithmetic operations to be performed after testing (see Event 1) without the necessity of clearing the accumulator with a separate instruction before some subsequent arithmetic operation.
3. OSR—Inclusive OR between the switch register and the AC. This instruction is performed third in sequence, allowing the AC to be cleared first, and then loaded from the switch register.
4. HLT—The HLT is performed last to allow any other operations to be concluded before the program stops.

This is the order in which all combined instructions are performed. In order to perform operations in a different order, the instructions must be written separately, as shown in the following example. The following combined microinstruction looks as if it might clear the accumulator, perform an inclusive OR between the SR and the AC, and then skip on a nonzero accumulator.

```
CLA OSR SNA
```

However, the instruction would not perform in that manner, because the SNA would be executed first. In order to perform the skip last, the instructions must be separated, as follows:

```
CLA OSR
SNA
```

Microprogramming requires that the programmer carefully code mnemonics legally so that the instruction actually does what he desires it

to do. The sequence in which the operations are performed and the legality of combinations are crucial to PDP-8/E programming.

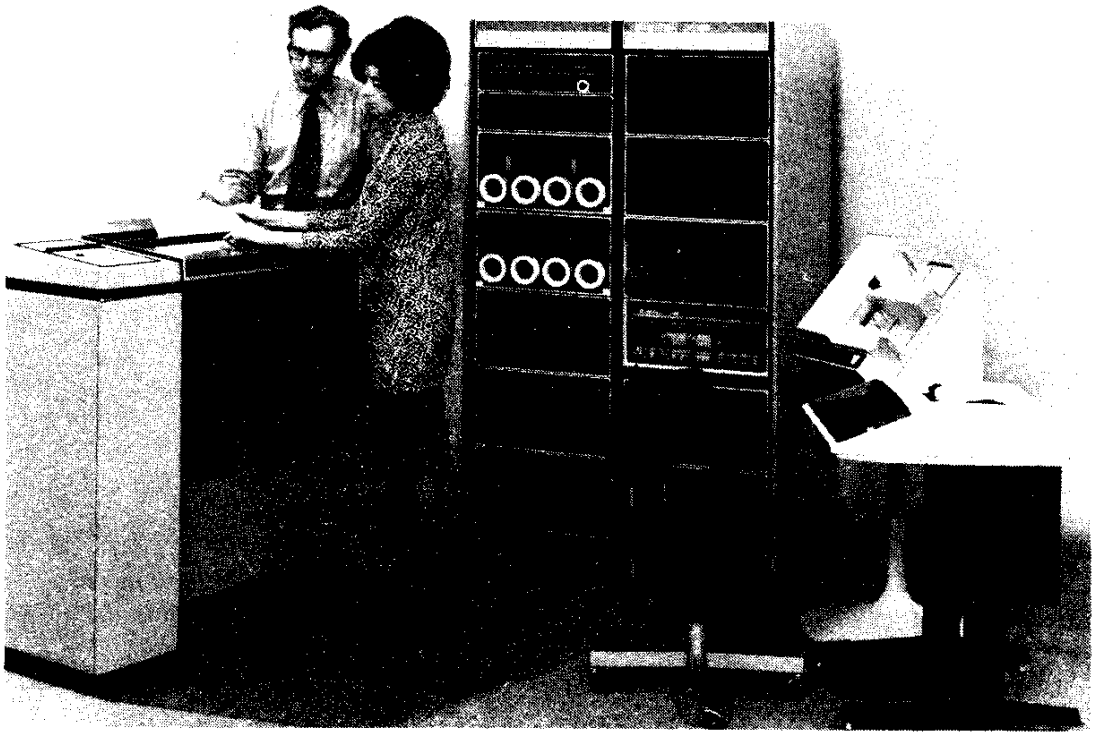
The following is a list of commonly used combined microinstructions, some of which have been assigned a separate mnemonic.

INSTRUCTION			EXPLANATION
—	CLA	CLL	Clear the accumulator and link.
CIA	CMA	IAC	Compliment and increment the accumulator. (Sets the accumulator equal to its own negative.)
LAS	CLA	OSR	Load accumulator from switches. (Loads the accumulator with the value of the switch register.)
STL	CLL	CML	Set the link (to a 1).
—	CLA	IAC	Sets the accumulator to a 1.
STA	CLA	CMA	Sets the accumulator to a −1.

In summary, the basic rules for combining operate microinstructions are as follows:

a. Group 1 and Group 2 microinstructions cannot be combined.
b. Rotate microinstructions (Group 1) cannot be combined with each other.
c. OR Group (SMA, SZA, or SNL) microinstructions cannot be combined with AND Group (SPA, SNA, or SZL) microinstructions.
d. OR Group microinstructions are combined as the logical OR of their respective skip conditions. AND Group microinstructions are combined as the logical AND of their respective skip conditions.
e. Order of execution for combined instructions is listed below.

Group 1	Group 2
1. CLA, CLL	1. SMA/SZA/SNL (OR group) or SPA/SNA/SZL (AND group)
2. CMA, CML	2. CLA
3. IAC	3. OSR
4. RAR, RAL, RTR, RTL, BSW	4. HLT



For the modern business manager, DEC offers complete data processing capability with on-line input/output devices such as a line printer, DECwriter, card reader, high-speed reader/punch; for mass storage, the DECtape and/or Disk systems are provided with complete file handling programs.

SECTION 2

PDP-8/E SYSTEM PROGRAMS

The Programming System for the PDP-8/E consists of SYSTEM PROGRAMS, UTILITY PROGRAMS, and APPLICATION PROGRAMS, and is complemented with the DECUS LIBRARY (see Figure 4-12).

More than 1000 PDP-8 programs are available to the user. Digital Equipment Corporation's Program Library, for instance, offers more than 700 programs from which to choose. In addition, a library containing programs developed by PDP-8 users, called the "DECUS LIBRARY," is available to all PDP-8 users. These programs cover a wide variety of applications in addition to the application programs developed by DEC. 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 available to all installations.

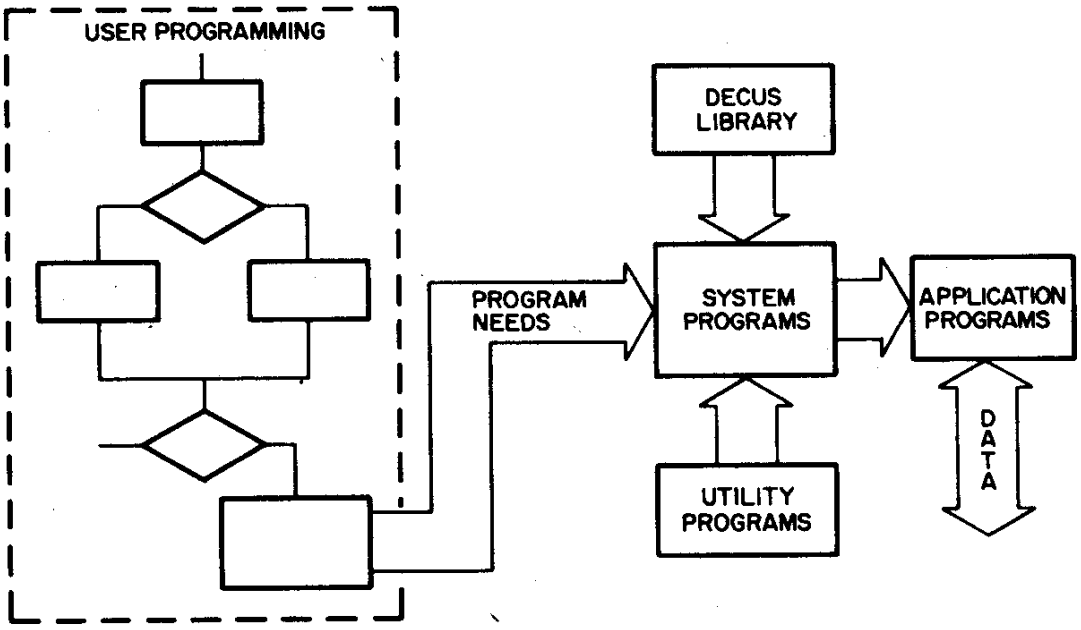


Figure 4-12 PDP-8/E Programming System

PDP-8/E Software Kit

A basic Family-of-8 software kit for the PDP-8/E computer is provided with each basic PDP-8/E system. The user receives one or more paper tapes and the necessary documentation for each program. Those programs provided in the basic software kit are identified by an asterisk (*) in the following program descriptions.

The following material is included in the basic PDP-8/E software kit:

PDP-8/E Small Computer Handbook
Introduction to Programming Handbook, 1970
Programming Languages Manual, 1970
Logic Handbook
PDP-8/E Instruction Cards (2 Cards)
FOCAL-69 Binary Tape and Utility Overlay Tape
PAL-III Symbolic Assembler
DDT Symbolic Debugging Program
ODT Octal Debugging Program (2 Tapes)
Symbolic Editor Program
RIM Punch (ASR-33 version)
Binary Punch (ASR-33 version)
Octal Memory Dump Program
Floating Point Subroutines (4 Tapes)
Software Performance Summary
A Complete Set of Maintenance (Diagnostic) Software

System Programs
The system programs are used, combined with the utility programs and/or the DECUS Library Program, to translate the user's ideas into desired application programs. System Programs include: Monitors, Editors, Assemblers, Compilers, Interpretive Languages, Debuggers, and Loaders.

Monitor Programs
Monitor programs used with the PDP-8/E include PS/8, Disk Monitor, and TSE.

PS/8 PROGRAMMING SYSTEM—PS/8, an 8K programming system, represents a significant advance in software development for small computers with capabilities which were formerly available only on such powerful machines as the PDP-10.

The PS/8 is a program development system for the PDP-8/E with minimum 8K of core and one or more of the following mass storage devices:

a. TC08/TU56 DECtape
b. RF08 Disk
c. RK8 Disk Pack
d. DF32 Disk (64K minimum)
e. TD8-E DECtape (with 12K Core)

The 8K programming system has the following features:

a. It allows the user device-independent access to up to 15 I/O devices, including up to eight DECtapes, up to four Disk units, Teletype, high speed paper tape reader and punch, card reader, line printer, and any other device for which it is possible to write a device handler in one, or in some cases two, pages.
b. The user program may call upon the monitor for several services, including loading device handlers for devices to which the user may assign a name, looking up input files on these devices, creating and closing variable and fixed-length output files on these devices, and getting and decoding a line of input from the console Teletype that identifies input and output files and options.

PS/8 gives the user, in addition to the language processor (8K FOCAL, 8K FORTRAN, PAL-8, and SABR), absolute and relocatable loaders, a symbolic editor, CONVERT (a program to provide file compatibility with the present Disk Monitor System), PIP (Peripheral Interchange Program), and an invisible ODT (Octal Debugging Technique), which allows the programmer to debug programs without giving up valuable core space.

Advantages offered by the PS/8 System include:

a. Device Independence—
    1. Programs can be run using the most effective I/O devices available at a given installation.
    2. As an installation grows, programs need not be rewritten for increased I/O capability.
    3. A user device handler may easily be added.
b. User Interfacing—
    1. User may use any standard I/O device without having to directly program the device.
    2. User may use the system command decoder to vary the I/O used in the programs.
c. Performance—Increased, due to efficient use of storage devices (especially noticeable on DECtape).
d. Expandability—As the system grows, so does the capability of the PS/8 system.
e. The PS/8 Library of Programs.
f. Increased power, derived from the addition of 4K of core to an existing 4K disk monitor system.
g. A program to convert all 4K monitor files to PS/8 files.
h. The capability of accommodating any amount of core storage from 8K to 32K.

Disk Monitor System

A keyboard monitor is available to Disk System users that allows them to save core images on the Disk and restore them to memory. The extensive software package available consists of: FORTRAN Compiler, Program Assembly Language (PAL-D), Editor Program (Editor), Peripheral Interchange Program (PIP), and Dynamic Debugging Technique (DDT-D) Program.

In addition, the user may save and restore his own core images and use the remainder of the available device storage for temporary storage of source or binary data. The monitor system may also be used with DECtape.

Time-Sharing Monitor

TSE (Time-Sharing PDP-8/E) is a general purpose, stand-alone monitor time-sharing system. The TSE can accommodate up to 16 users simultaneously. A minimum of 12K of core memory and an RF08-type disk is required for a comprehensive library of system programs, which provide facilities for compiling, assembling, editing, loading, saving, calling, debugging, and running user programs on line.

The center of a TSE system is a complex of programs called Monitor. Monitor coordinates the operations of the various units, allocates the time and services of the computer to users, and controls their access to the system. The computer works on user programs simultaneously by segregating the central processor operations from the time-consuming interactions of the human users. Execution of various programs are interspersed without interfering with one another and without detectable delays in the responses to the individual user.

Consult the DECUS Library for additional monitor programs.

EDITOR PROGRAM

Symbolic Paper Tape Editor*

The Symbolic Paper Tape Editor program is used to edit, correct, and update symbolic program tapes using the PDP-8/E, the Teletype unit, and/or the high-speed reader. With Editor in core memory, the user reads in portions of his symbolic tape, removes, changes, or adds instructions, and gets back a complete new symbolic tape with errors removed. He can work through the program 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. The program requires 4K core and a Teletype. Consult the DECUS Library for additional editor programs.

ASSEMBLER PROGRAMS

Assembler Programs used with the PDP-8/E include PAL-III, PAL-D, MACRO-8, and 8K-SABR. The use of an assembly program has become a standard practice in programming digital computers. This process allows the programmer to code instructions in a symbolic language, one he can work with more conveniently then the 12-bit binary numbers that actually operate the computer. The assembly program 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 more efficiently and debugging is considerably simplified.

PAL-III*—PAL-III is a basic assembler allowing symbolic references, origins, and expressions. The output is in a form suitable for input to the binary loader.

PAL-III is a two-pass assembler with an optional third pass; i.e., the symbolic program tape must be passed through the assembler two times to produce the binary-coded tape, and the optional third pass produces a complete octal symbolic program listing, which can be typed and/or punched.

*Part of the basic software package.

PAL-III accepts symbolic program tapes from either the low-speed or high-speed reader and produces the binary tapes on either the low-speed or high-speed punch.

During assembly, the programmer communicates with PAL-III via the switches on the computer console. Switch options are used to specify which pass the assembler is to perform and which reader and punch the assembler should accept input from and punch out on.
PAL-III requires 4K of core memory and a Teletype.

PAL-D—PAL-D incorporates most of the features of both PAL-III and MACRO-8, and is used only in the Disk Monitor System. PAL-D is designed primarily for 4K PDP-8/E computers with disk.

PAL-8—PAL-8 is an extended assembler which runs under the PS/8 Programming System. It includes the best features of PAL-III and MACRO-8 plus a number of additional features:

- Conditional assembly.
- Large symbol table (up to 1800 symbols) (12K core).
- High speed Binary symbol table search.
- Paginated listings with page headings and page numbers.

MACRO-8*—MACRO-8 is an advanced assembler which has the same features as PAL-III, plus the following additional features: user-defined macros, double precision integers, floating-point constants, arithmetic and Boolean operators, literals, text facilities, and automatic off-page linkage generation. To incorporate such features, the size of the user's symbol table was decreased. However, the programmer can increase or decrease the size of the permanent symbol table at the expense of some of the more space-consuming features.

MACRO-8 requires 4K of core memory and a Teletype.

8K SABR—8K SABR (Symbolic Assembler for Binary Relocatable Programs) is an advanced one-pass symbolic assembler. It translates symbolic programs written in the SABR language into binary relocatable code acceptable to the computer. SABR programs are core page independent. Therefore, programs may be written without regard to the 128-word core page of the computer. SABR automatically generates off-page and off-field references for direct or indirect statements. It also automatically connects instructions on one page to those that overflow onto the next. The list of available pseudo-ops is extensive, including external subroutine calling, argument passing, and conditional assembly. SABR offers an optional second pass to produce a side-by-side octal/symbolic listing of the assembled program.

The relocatable binary tapes produced by SABR are loaded into any field of core memory using the 8K linking Loader, as are the comprehensive library of subprograms. These subprograms may be called by any SABR program.

*Part of the basic software package.

The high speed reader and punch is recommended for the 8K SABR program.

Consult the DECUS Library for additional assembler programs.

COMPILER PROGRAMS

Compiler programs used with the PDP-8/E include DIBOL, FORTRAN 4K and 8K, and ALGOL-8. Although the DIBOL program is listed with compiler programs, DIBOL includes a package which contains a Monitor Program, an Editor Program, and utility programs.

DIBOL Software System—DIBOL (**DI**gital Equipment Corporation **B**usiness **O**riented **L**anguage) is the first business language that was designed specifically for a minicomputer. The DIBOL Software System is a complete business oriented software system for implementing business applications such as billing, Accounts Receivable, Inventory Control, Cost Accounting, payroll, Accounts Payable, Sales Analysis, and many other accounting and business management functions. This software system comprises a simple business oriented language, a data management system to provide input, sorting, editing and filing facilities, and a monitor program to organize the DIBOL facilities into a unified system. These components of the software system enable you to develop business applications "your way".

The DIBOL configuration consists of:

a. One PDP-8/E,
b. One high speed paper tape reader and punch,
c. Four DECtapes,
d. One ASR-33,
e. 8K of core,
f. LE8 Line Printer, and
g. The DIBOL Software System Package.

Refer to appendix A for more details on the DIBOL Software System.

4K FORTRAN—The 4K 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.

4K FORTRAN consists of a compiler, a debugging aid, and an operating system. The one-pass compiler translates FORTRAN coded symbolic language statements into binary code and produces a binary tape. The debugging aid (Symbolprint) lists the variables used and their locations in core and indicates the section of core used by the compiled program. The program requires 4K of core memory and a Teletype.

Document: 4K FORTRAN Programmer's Reference Manual

8K FORTRAN Compiler—

The PDP-8 Paper Tape System version of 8K FORTRAN has the following features: subroutines, two levels of subscripting, function subprograms, relocatable output, COMMON statements, library subroutines, six types of format specifications, and I/O supervisors. 8K FORTRAN requires the use of the 8K SABR assembler and Linking Loader.

This compiler utilizes all available core from 8K to 32K, and correctly loads programs over page boundaries.

The program requires 8K of core memory, a Teletype and high speed Reader/Punch.

INTERPRETIVE PROGRAMS
Interpretive Programs used with the PDP-8/E include FOCAL and BASIC.

FOCAL-8—FOCAL-8 (FOrmula CALculator) is an on-line conversational language designed for solving complicated calculations. The language consists of short statements and mathematical expressions in standard notation. FOCAL puts the full calculating power and speed of the computer at the user's fingertips without the user having to master the intricacies of machine-language programming. FOCAL is an easy way of simulating mathematical models, plotting curves, handling set of simultaneous equations, and much more.

FOCAL is available in several configurations.

*a) Single-user FOCAL—requires 4K and Teletype.
b) Four-user FOCAL—requires 8K core and four Teletypes (console plus three).
c) Seven-user FOCAL requires 8K core, seven Teletypes (console plus six), and a RF08 or DF32D disk.

BASIC 8
BASIC 8 is a modified version of the algebraic language developed at Dartmouth College. The BASIC language is composed of easy-to-learn English statements and mathematical expressions.

BASIC 8 is available in five versions:

a.	EDUSYST-10	One user—requires basic processor and Teletype,
b.	EDUSYST-20	Two to five users—requires basic processor with 8K memory and two to five Teletypes.
c.	EDUSYST-30	One user Batch—requires basic processor, teletype, and either DECtape or Disk (RF08 or DF32-D),
d.	EDUSYST-40	Combination of b and c.
e.	EDUSYST-50	Eight to sixteen users—multiple language capability—requires basic processor with 8K memory, DECtape or Disk (RF08 or DF32-D), and eight to sixteen Teletypes.

DEBUGGER PROGRAMS
Debugger Programs used with the PDP-8/E include ODT-8 and DDT-8.

DDT-8 (Dynamic Debugging Techniques)*—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.

*Part of the basic software package.

Using the standard Teletype, the user can communicate conveniently with the PDP-8/E 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.

DDT-8 requires 4K of core memory and a Teletype.

ODT-8 (Octal Debugging Technique)*—ODT-8 allows the programmer to do all the things mentioned in DDT-8 by communicating with his object program using the octal representation of his binary program. ODT-8 occupies less core storage than DDT-8 and can be loaded in upper memory or lower memory, depending on where the binary program resides.

ODT-8 requires 4K of core memory and a Teletype.
Consult the DECUS Library for additional debugging programs.

LOADERS

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 Teletype or high speed paper tape reader.

Binary Loader*
The Binary Loader is a short routine for reading and storing information contained in binary-coded tapes, using the Teletype or high-speed paper tape reader.

The Binary Loader accepts tapes prepared by the use of PAL or MACRO-8. Diagnostic messages may be included on tapes produced when using either PAL or MACRO. The Binary Loader ignores all diagnostic messages. See Appendix A and the DECUS Library listing for additional loader programs.

Linking Loader
The Linking Loader is capable of loading and linking a user's program and subprograms in any field(s) of memory. The Linking Loader has options which can obtain storage map listings of core availability for the user.

The Linking Loader has the capability to search program libraries for subroutines which are referenced by the program in core and load those subroutines needed. A library is a collection of relocatable subroutines (FORTRAN or SABR output) with a directory at the beginning to facilitate searching.

The Linking Loader is capable of loading any number of user and library programs into any field of memory. Several programs are usually loaded into each field. Because of the space reserved for the Linkage Routines, the available space in field 0 is three pages smaller than in all other fields.

*Part of the basic software package.

UTILITY PROGRAMS

PDP-8/E 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.

Most of these programs require only 4K of core memory and a Teletype. Some of the primary utility programs include data conversion programs and maintenance and diagnostic programs (see Appendix A for a listing of other utility programs).

Data Conversion Programs—Data conversion programs used with the PDP-8/E include the Floating Point Package and Math Function Routines.

Octal Memory Dump—This program enables the user to dump in octal mode any or all data in any memory field to either the Teletype or high-speed paper tape punch. During dumping, the absolute address of each location being dumped is held in the accumulator. When dumping is completed, output devices and memory fields can be changed to dump another section of memory. The program requires one core page.

Mathematical Function Routines

The programming system includes a set of mathematical function routines to perform the following operations: single precision multiplication, division, square root; double precision sine, cosine, multiply, divide; and arithmetic and logical shifts.

Floating Point Package*

The Floating Point Package permits the PDP-8/E to perform arithmetic operations that many other computers can perform only after the addition of costly optional hardware. Floating point operands retain 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, a particularly advantageous feature to the inexperienced programmer. The floating point subroutines and interpreter permit the programmer to encode arithmetic operations to either six or ten decimal digits of precision as easily as though the machine had floating point hardware. Also included in the package are input and output conversion routines.

Any common storage reserved by the programs being loaded is allocated in field 1 from location 200 upwards. The space reserved for common storage is subtracted from the available loading area in field 1. The program reserving the largest amount of common storage must be loaded first.

The Run-Time Linkage Routines necessary to execute SABR programs are automatically loaded into the required areas of every field by the Linking Loader as part of its initialization. The user needs to know nothing more about these routines than the particular areas of core they occupy.

*Part of the basic software package.

MAINTENANCE AND DIAGNOSTIC PROGRAMS

A complete set of standard diagnostic programs is provided (see Appendix A) 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.

Consult the DECUS Library for additional utility programs.

THE DECUS LIBRARY

Although each PDP-8/E is delivered to the user complete with an extensive program kit, the DECUS Library provides the user with a continually growing assortment of various programs which are available for general use. DECUS includes a wide variety of system programs, utility programs, and application programs. A listing of the programs available from the DECUS Library can be obtained from the Program Library, Digital Equipment Corporation, Maynard, Mass. 01754.

APPLICATION PROGRAMS

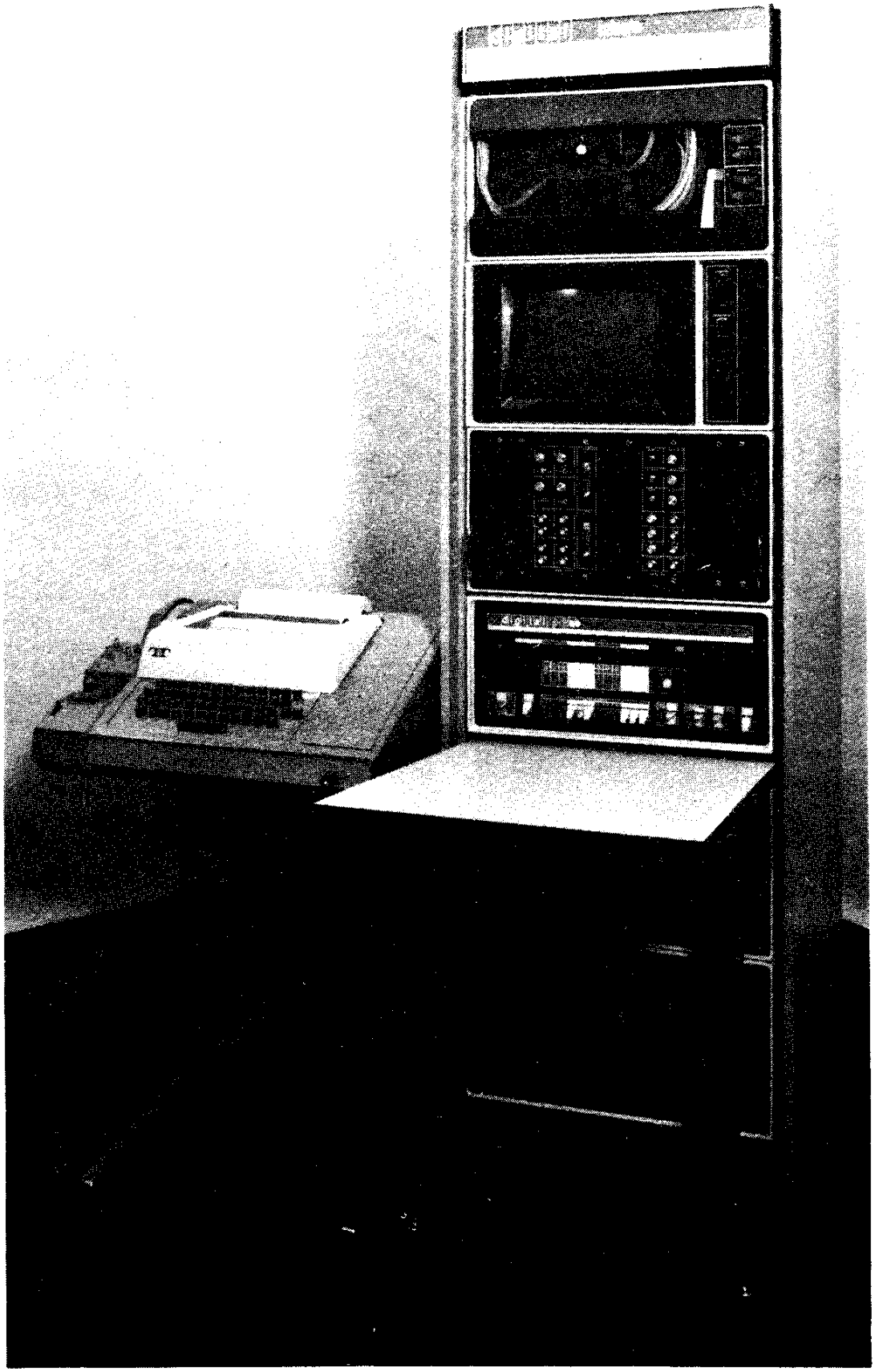
Integrated Application Packs are special-purpose software packages which provide a specific solution to one aspect of a general problem. By using the proper application pack and additional hardware and/or software, the user can have a computer system customized to fit his particular requirements. Although numerous programs are offered by the DECUS Library, some of the commonly used application programs include:

Display 8
IDAC 8
LAB 8/e
EDUSYSTEMS
DIBOL
Quick Point 8
Typeset 8
PHA 8
Chromatographic Data Processing (CDP)

LAB-8/E System Software

The LAB-8/E is a major step forward in low-cost laboratory computing. At a cost lower than most special purpose instruments, LAB-8/E includes an analog-to-digital converter, real time clock with three Schmitt triggers, point plot display control, and Digital's newest small general purpose computer, the PDP-8/E.

*Part of the basic software package.



LAB-8/E

The LAB-8/E is designed to be used as a total laboratory system, not a computer with laboratory-type peripherals. The peripherals have been designed to plug into the laboratory option cabinet H945. The Schmitt triggers may start the clock, the clock may start the analog-to-digital converter, the analog-to-digital converter may increment the multiplexer, and so on. This kind of flexibility lets you configure an interactive LAB-8/E system from a PDP-8/E and lets you expand the system to suit your particular needs, in most cases without the expense of extra cabinets, space, and cabling.

A significant feature of the LAB-8/E is its software. Programs developed for over 11,000 "family of 8" computers are compatible with the PDP-8/E. Special groups of applications software developed for the LAB-8/E allow many users to put their system to work on the day it arrives. Users may share their programs through DECUS, one of the world's largest computer users groups.

The LAB-8/E is being used in biomedical research for EEG, ECG, EMG, Behavior Studies, diagnostic assistance, patient monitoring, and similar applications. In analytical instrumentation, LAB-8/E is used for NMR work, electrochemistry, kenetic studies, reporting, automation of instruments, etc. In engineering and science, the LAB-8/E is used in simulation techniques, and laboratory applications in physics, biology and psychology. In industrial testing, LAB-8/E is used for material testing, sound and vibration analysis and real-time data acquisition and data analysis.

The LAB-8/E features some unique software packages such as BASIC and Advanced Averages, Auto and Cross Correlation, NMR Averager, Simulator, Fast Fourier Transforms, Histrogram programs, DAQUAN (data acquisition and analysis), and real-time BASIC, the easy to learn, conversational language. Refer to the LAB-8/E Users Handbook, DEC-LB-HRZA-D for more information.

INDAC SOFTWARE FOR IDACS-8 SYSTEMS

INDAC software package is designed for real time data acquisition and control applications. This field proven software presents the total system capability to a design or process engineer by integrating various process interfaces and allowing English like statements for data input/output in a real time environment.

INDAC system software fully supports INDAC language. It is a BASIC like language with special commands for program scheduling as a function of time, sequence or external event. Communication between process variables and computer is achieved by much simplified input/output commands.

The elements of INDAC software are:

- SYSTEM BUILDER—GENDAC—A conversational program allowing system configuration for a specific installation. It allows adding *new I/O handlers, library routines, etc. to an INDAC system.*
- INDAC COMPILER—The compiler converts a source program into object code, which is executable at runtime. It provides extensive diagnostics to help debug source program.

- INDAC EXECUTIVE—The executive schedules various tasks at run-time, supervises allocation of disk and core, handles clock interrupts, controls the flow of data from input to output devices, and allows on-line communications between the user and INDAC system.

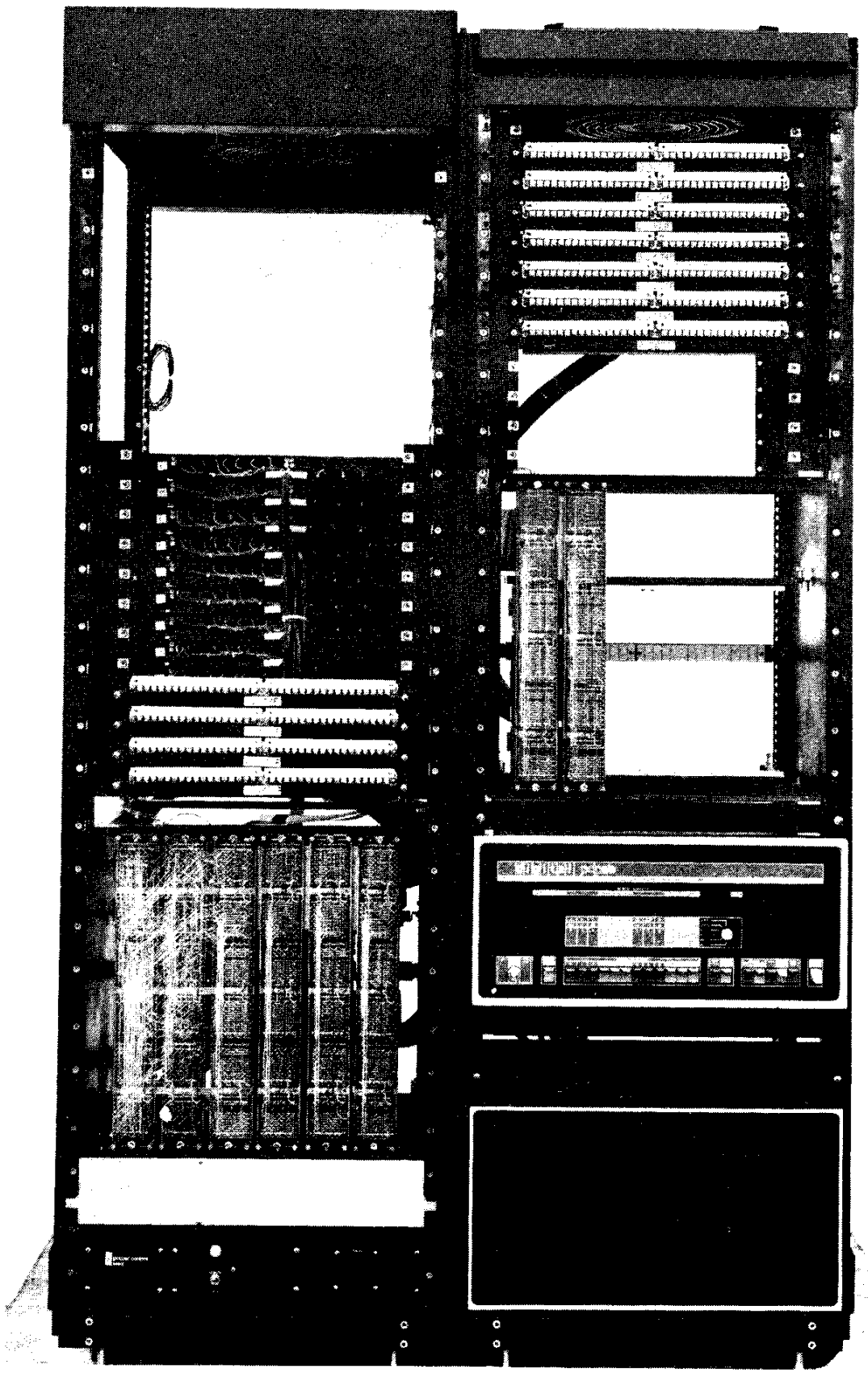
- I/O HANDLERS LIBRARY—The input/output handlers library services various devices in INDAC system, which can be addressed with simple GET or SEND statements. The devices supported include the complete line of analog input, digital input/output and analog output sub-systems besides standard peripherals.

- LIBRARY—The library contains the arithmetic and transcendental functions such as sine, cosine, arctangent, or log, as well as conversion routines for commonly used thermocouples.

- SUPPORTING SOFTWARE—INDAC support programs are unique in the field of small computer systems. INDAC program preparation is achieved by powerful PDP-8 disc monitor system. User can establish and maintain files for source programs, edit and compile them, in a file-to-file operation. Other support programs help core examination for system configuration and other related debugging tasks.

INDAC software has been proven in various field installations for applications such as:

Quality control and testing of air-conditioner valves
Performance testing of internal combustion engines
Control of semi-conductor diffusion furnaces



INDAC-8 System

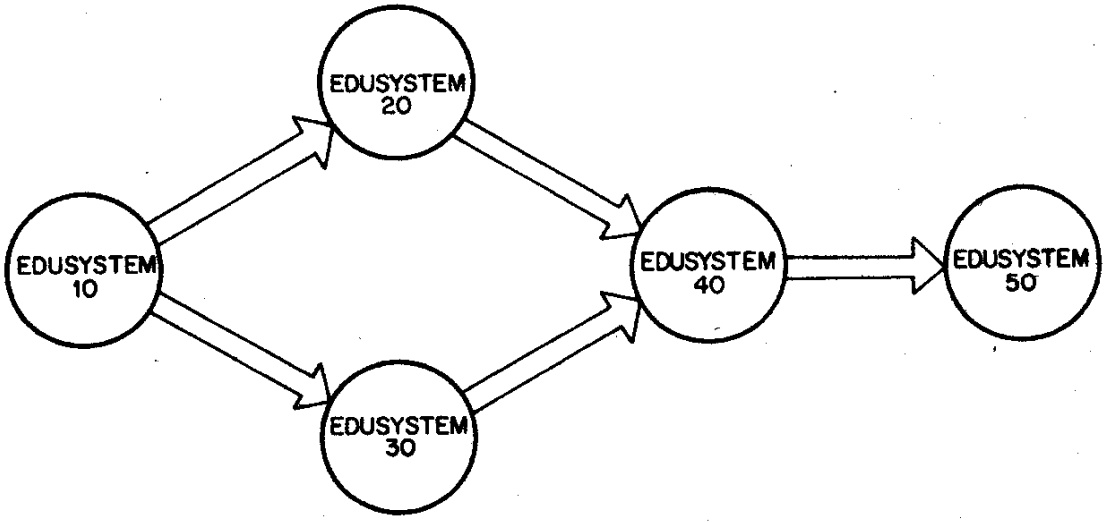
EDUSYSTEMS 10 THROUGH 50

DEC's Edusystem series is ready to serve the needs of schools—now and for the future.

The Edusystem series is a reliable, proven, expandable, and instructionally complete classroom computer system. Designed as a complete instructional package, Edusystems 10 through 50 offer a range of classroom computer systems that can update the calculator user to computers or fill the most ambitious computer science and administrative needs of the modern school.

The PDP-8 family of computers is the nucleus of the system. Time-proven curriculum materials and a truly comprehensive library of application programs in all subject areas are brought together in the classrooms by Edusystems as a total instructional package for the beginner or sophisticated teacher-user.

Each Edusystem is completely expandable to a higher system; no non-functioning leftovers or obsolescent units. The expansion route for Edusystems is diagrammed below:



The Edusystem-10 graduate, who requires a system capable of involving a much greater number of students, has two choices: (1) Edusystem-20 or (2) Edusystem-30. Both are excellent systems and the choice comes in tailoring the system to the individual school.

Edusystem-20 can accommodate up to five on-line terminals for time-sharing BASIC on one PDP-8/E Computer. The benefit here is a very great degree of interaction between student and the computer, accompanied by increased motivation.

Edusystem-30 offers a capability of "batching" mark-sense cards that students mark with an ordinary pencil. This relieves the bottleneck of students waiting to sit down and type their programs. The benefits here are stored or "saved" programs and throughput, more students actually getting involved with "hands on" computer experience. With an optional DECwriter and accompanying paper-tape reader, Edusystem-30 can realistically process up to 100 programs each hour.

The multiple terminals of Edusystem-20 allow students to interact with the computer in exercising simulation programs. Population growth, genetic change, environmental pollution, and Civil War battles are only a few examples of simulation programs that encourage the student to learn by doing.

Edusystem-30 is a true miniature of the gigantic systems in commercial and scientific computer centers.

For those who need more:

- Edusystem-20 offers even greater throughput;
- Edusystem-30 is for users who want multiple terminals for greater student involvement;
- Edusystem-40—a combination of Edusystem-20 and Edusystem-30.
- Edusystem-50 is the top of the PDP-8 based Edusystem line. Edusystem-50 offers the school well versed in computer sciences a true computer center installation. Sixteen simultaneous users may be handled by Edusystem-50 either at the computer site or over telephone lines.

Edusystem-50 is versatile; each of the many users has his choice of computer language. BASIC, FOCAL, ALGOL, FORTRAN, and PAL (an assembly language) are available to any user at any time.

In all of these Edusystems, the design criteria are reliability, ease of operation and the production of a complete instructional program. Each Edusystem includes a teacher's library of curriculum material that includes:

Edusystem User's Guide; DEC
Teach Yourself BASIC, I and II; Tecnica
Basic BASIC; Hayden
A FOCAL Primer; Cornell University
Computer Methods in Mathematics; Addison-Wesley
Computer Assisted Math Program; Scott-Foresman
CAMP, First Course
CAMP, Second Course
CAMP, Algebra
CAMP, Geometry
CAMP, Intermediate Mathematics
Teacher's Guides

Problem Solving with the Computer; Entelek
Computers in the Classroom; DEC
Problems for Computer Mathematics; DEC
An Introduction to Computer Science; Scott-Foresman
with a Teacher's Commentary

Fundamentals of Digital's Computers; Howard W. Sams
Introduction to Programming; DEC
Programming Languages; DEC
"Introducing BASIC with the Overhead Projector"; DEC
23 viewgraph transparencies and teacher's guide
BASIC Application Programs, Sets I, II, III, IV and V; DEC
Program listings and descriptions in all subject areas
BASIC Simulation Programs, Volumes I through VI; Huntington Project and DEC

EDUSYSTEM-10

Edusystem-10 is a low-cost, general-purpose instructional computer system—ideal for the school just getting started with computers.

Edusystem-10 is a logical follow-on to the use of simple electronic calculators in the Math and Science Labs, with the added ease and power of the English-like control language BASIC.

Edusystem-10's BASIC allows students with no experience to perform calculations the minute they sit at the terminal and to develop their first simple program during their first class period.

Edusystem-10 is composed of:

- Hardware
  - PDP-8/e Computer with 4K (4096) words of core storage
  - Power Fail Detect and Auto Restart
  - Hardware bootstrap loader
  - ASR-33 Teleprinter with paper-tape reader/punch
- Software
  - BASIC compiler and Teacher's Curriculum Materials
  - FOCAL
  - FORTRAN*
  - PAL III (Assembly)*
- Input/Output
  - 1 interactive terminal

*** Available with optional high-speed, paper-tape reader/punch**

EDUSYSTEM-20

Edusystem-20 is a low-cost time-sharing system that simultaneously supports two to five terminals. Edusystem-20 is intended for school systems intending to use the computer with large classes of students or in several classes at the same time.

A typical distribution of five terminals is:
a. Three terminals in the Math Lab
b. One terminal in a Science Lab
c. The fifth terminal operating over telephone lines at a second school.

Edusystem-20, as with Edusystem-10, generally is operated in a problem-solving role. The multiple terminals of the Edusystem-20 also make the system ideal for simulation programs.

Significantly larger and more complex programs can be run on Edusystem-20 and the practical limit to the size of the student programs is a function of the number of terminals being used and the amount of core storage available. Edusystem-20 BASIC offers extended features such as a complete EDIT command allowing simple debugging of student programs.

Edusystem-20 is composed of:

- Hardware
  - PDP-8/e Computer with 8K or more of core storage
  - Power Fail Detect and Auto Restart
  - Hardware Bootstrap Loader
  - 1 to 5 ASR-33 Teleprinters
- Software
  - Time-shared BASIC and Teacher's Curriculum Materials
  - Time-shared FOCAL
  - FORTRAN*
  - PAL III (assembly)*

EDUSYSTEM-30

Edusystem-30 is a true miniature of the massive commercial and university computer centers. Edusystem-30 offers stored programs on a mass storage device, "batch" operation with pencil marked cards and sophisticated BASIC software.

Edusystem-30 has **throughput.** As many as 50 student programs can be run in one hour when the output device is a standard ASR-33 teleprinter, and over 100 programs can be run in one hour with an optional DECwriter. Edusystem-30 gives the lower dollar-per-student figure possible.

Commonly used utility programs can be stored on the mass storage device, and students need only enter data cards.

Edusystem-30 is a natural base for expansion to a PS/8 programming system for computer science and administrative work.

Edusystem-30 consists of:

- Harware
  - PDP-8/e Computer with 4K of core storage
  - A mass storage device (disk or DECtape*)
  - Power Fail & Auto Restart
  - Optical Mark Card Reader
  - ASR-33 Teleprinter
  - Optional DECwriter or line printer
- Software
  - Batch BASIC, cards, templates and Teacher's Materials
  - FOCAL
  - FORTRAN**
  - PAL III (assembly)**
- Input/Output
  - Card reader/teleprinter
- *Environmental Requirements*
  - Control of excessive dust, temperature and humidity (formal computer room not required).

* TC08

** Available with optional high-speed, paper-tape reader/punch

EDUSYSTEM-40

Edusystem-40 is the ideal school-wide instructional computer system operating under one language. Edusystem-40 offers all the advantages of both Edusystems-20 and -30.

Edusystem-40 can "batch" mark-sense cards for the large volume of instructional use, and then provide interactive terminal use for intensive work with a smaller number of either advanced or slower students. The interactive terminal allows the slower student the motivation and infinite patience of the computer, while the advanced student uses it as a means of investigation and expression.

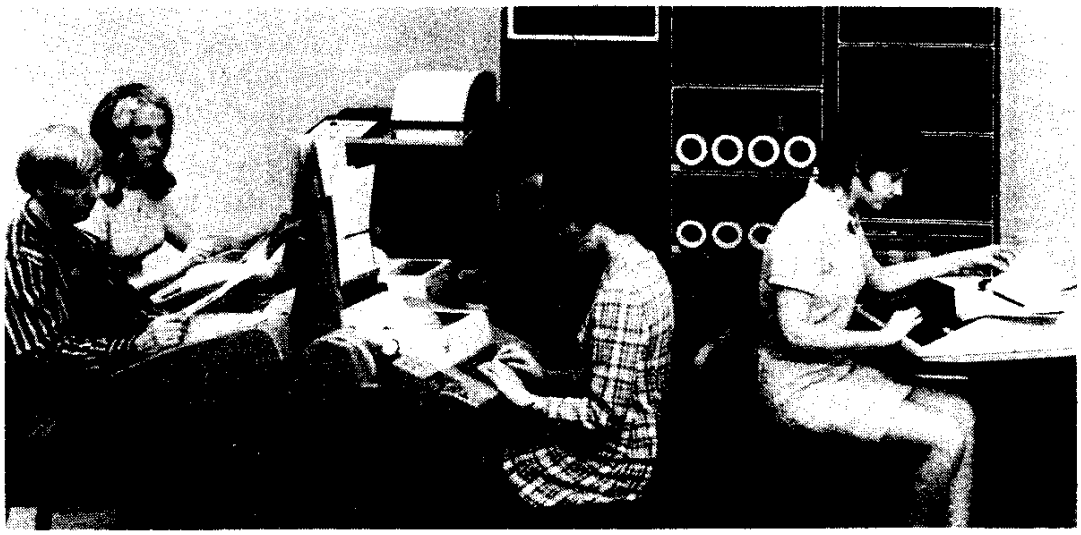
Edusystem-40 software is provided in card format to make use of the high-speed input of the card reader.

Edusystem-40 consists of:

- Hardware
  - PDP-8/e Computer with 8K or more of core storage
  - Mass-storage device (disk or DECtape*)
  - Power Fail and Auto Restart
  - Hardware Bootstrap
  - Optical Mark Card Reader
  - 1 to 5 ASR-33 Teleprinters
  - Optional DECwriter or Lineprinter
- Software
  - Batch BASIC with cards, templates and Teacher's Materials
  - Time-Sharing BASIC
  - Time-Sharing FOCAL
  - FORTRAN**
  - PAL III (assembly)
- Input/Output
  - 1 to 5 interactive terminals, or card reader/teleprinter
- *Environmental Requirements*
  - Control of excessive dust, temperature and humidity (formal computer room not required)

* TC08

** Full standard FORTRAN II available with 64K of disk or DECtape configuration (PS/8)



EDUSYSTEM-50

Edusystem-50 is a true time-sharing system that offers multiple languages to 16 simultaneous users. Edusystem-50 supports from 4 to 16 terminals in BASIC, FOCAL, ALGOL, FORTRAN or PAL (assembly) at the same time.

The computer science class might be using 6 terminals and PAL, while the Math Lab is using 8 terminals operating with BASIC, and the Physics class is using 2 terminals with the simplified FORTRAN.

Edusystem-50's BASIC allows users to maintain files on DECtape and output through the optional line printer. Edusystem-50 can do administrative and instructional work at the same time.

Edusystem-50 consists of:

- Hardware
  - PDP-8/E Computer with 12K or more core storage
  - 262K disk storage
  - DECtapes*
  - Time-Sharing hardware/software
  - Clock
  - 1 to 16 ASR-33 Teleprinters/remote lines
  - Optional Line Printer
  - High-Speed Paper-Tape Reader/Punch
  - Power Fail Detect and Auto Restart
  - Hardware Bootstrap Loader
- Software
  - Time-Shared 8 (TSS/8) Monitor System
  - BASIC, FOCAL, ALGOL, FORTRAN-D, PAL-D
  - System Manager's Guide and documentation
  - User's Guides
  - Teacher's Curriculum Materials
- *Input/Output*
  - 1 to 16 interactive terminals and optional line printer
- Environmental Requirements
  - Control of excessive dust, temperature and humidity (formal computer room not required)

TYPESET-8 COMPUTERIZED SYSTEM

Typeset-8 is a computerized system that produces punched paper tape containing all the hyphenation, justification and format commands needed to drive just about any Typesetting machine on the market.

A perforator operator sits down at the same perforating machine he has always used. When he is through typing, he feeds the tape into a Photoelectric reader, which transfers the combined instructions and marked up copy to paper tape for processing. Letter spacing, word spacing, and end-of-line decisions are unnecessary; therefore, manual justification is completely eliminated. As the perforated tape is being read, the computer simultaneously processes the instructions and text, as the Typeset-8 punch perforates the output tape.

Hot Metal

When the output tape is placed in the tape reader of the linecasting machine, it is processed in the same manner as a manually prepared tape. As the perforated tape is being read, the linecasting machine drops the proper mats to produce lines of justified tape.

Photo Composition (cold type)

Photo composition and associated new processes for composing a complete type form ready for plate making and the press begins and ends in the same way as the older and more conventional methods of printing from hot metal. The main difference is that the input tape contains slightly more complex format codes to produce varying typefaces, styles, kerning, leading, and column widths.

Business

In addition to typesetting, DEC offers a business package designed to computerize many areas of administrative activities with considerable time-saving and cost-saving features.

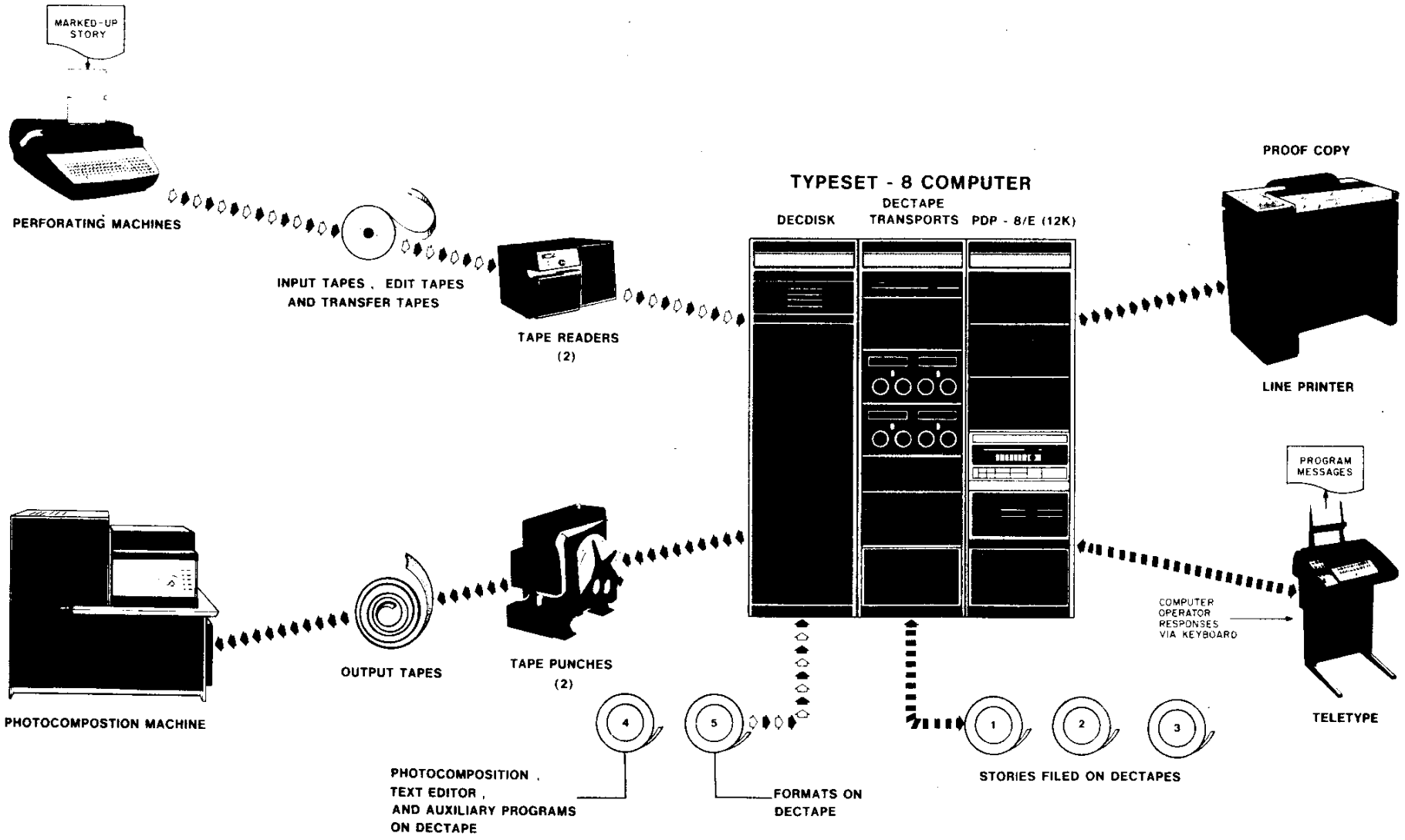
DEC's Business Package include 5 systems: Payroll, Circulation, Advertising, Accounts Payable, and General Ledger. These systems are made to order for your business. They are intended to do the detail work and give the manager time to manage. For example, with DEC's Business Package:

- You can print a complete payroll for 500 employees in less than three hours.
- You can have detailed sales analysis reports on circulation and advertising.
- Your accountant can indicate the due bills he doesn't want to pay; and your computer automatically pays the rest.
- *You can use Business Package reports to determine if, why, and* where business is increasing or decreasing.
- You can pinpoint your troublespots.
- You will have time to manage.

STORAGE AND EDIT SYSTEM

The PDP8/E Storage and Edit System is designed for customers who need the capability to retain text on a storage media for an indefinite period of time. The stored information can be corrected, reformatted, updated, and reworked until a final proof is accepted by the editorial staff. Proofing is done through copy output on a line printer, thus eliminating the expense, time, and manual intervention involved in using a photocomp machine as a proofing device.

During initialization of the system, the "master system tape" is placed on a DECtape transport and then transferred to the disks by the executive. The master tape is then removed, and three storage tapes mounted. With the system in this configuration (i.e., 3 storage tapes), immediate access can be gained to 100 stories, consisting of approximately 1 million TTS characters or 21,000 11-pica lines. Additional transports can be added to increase the immediate access by 7000 lines/transport. A directory of stories currently on the system is retained on the storage tapes and can be requested for printout through the monitor. Files that are retained for an extended length of time and updated less frequently can be stored on separate tapes and incorporated into the system when needed.


MARKED-UP STORY
PERFORATING MACHINES
INPUT TAPES , EDIT TAPES AND TRANSFER TAPES
TAPE READERS (2)
TYPESET - 8 COMPUTER
DECDISK
DECTAPE TRANSPORTS
PDP - 8/E (12K)
PROOF COPY
LINE PRINTER
PROGRAM MESSAGES
COMPUTER OPERATOR RESPONSES VIA KEYBOARD
TELETYPE
PHOTOCOMPOSTION MACHINE
OUTPUT TAPES
TAPE PUNCHES (2)
4
5
1
2
3
PHOTOCOMPOSITION , TEXT EDITOR , AND AUXILIARY PROGRAMS ON DECTAPE
FORMATS ON DECTAPE
STORIES FILED ON DECTAPES

CHAPTER 5

PROGRAMMED DATA TRANSFERS

GENERAL

The nature of the IOT instruction was explained in Chapter 3, programming the Teletype was defined in Chapter 4, and the discussion of data transfers as viewed from the peripheral will be discussed in Chapters 9 and 10. This chapter deals with programmed data transfers as viewed from the processor. This discussion is directed to the major considerations of setting up the mechanism for transferring data to and from the processor.

Three types of data transfers are offered by the PDP-8/E processor to receive, store, and transmit data between one or more peripherals and the processor. These transfers are called:

a) Programmed I/O Transfers,
b) Programmed Transfers using the Program Interrupt Facility,
c) Data Transfers using the Data Break Facility.

The first two types of transfers, described in this chapter, are controlled by the program, while the data break transfers are controlled by each peripheral. Data break transfers are discussed in Chapter 6.

PROGRAMMED DATA TRANSFER vs. DATA BREAK

Most input/output (I/O) transfers are controlled by the computer program. Such an information transfer requires perhaps five times as much computer time as does a data break transfer. However, in terms of real time, the duration of a programmed transfer is rather small, due to the high speed of the computer, and is usually well within the limits required for laboratory or process control instrumentation. To get maximum benefit from the control features of the PDP-8/E, the user should utilize programmed data transfers in most cases. Moreover, the peripheral control circuits are usually simpler and less expensive than are those of data break transfer peripherals.

PERIPHERAL REQUIREMENTS

Programmed data transfers use the processor accumulator (AC) register as an intermediate storage point between core memory and a buffer register within the peripheral. If an output data transfer is to be performed, the computer program causes data to be loaded from core memory into the AC. The program then uses an input/output transfer (IOT) instruction to, first, select the desired peripheral and, second, direct the peripheral to generate certain control signals, which cause the data in the AC to be placed onto the OMNIBUS DATA lines. The peripheral, whose control module monitors these DATA lines, then strobes the data into its input buffer register and prepares to process the information. Each transfer from computer to peripheral is handled in this manner.

If the transfer is from peripheral to computer, the process is reversed.

The IOT instruction selects the peripheral and directs it to generate control signals. The peripheral then places the data in its output buffer register on the OMNIBUS DATA lines. A processor timing signal strobes the information into the AC. The program may then either deposit the data into a memory location or use it in some other way.

Each peripheral connected to the OMNIBUS transfers data in the manner described. The bus system of I/O transfers, by which many peripherals monitor the OMNIBUS signal lines, imposes the following requirements on the peripheral equipment:

a. Each peripheral must contain a device selector circuit which monitors Memory Data (MD) lines 3 through 8 of the OMNIBUS. During an IOT instruction, these MD lines carry a selection code which is unique for each peripheral and which must be decoded by the peripheral's device selector.
b. Each peripheral must contain gating circuits which monitor the MD9-11 lines of the OMNIBUS. During an IOT instruction, these MD lines carry command signals that the peripheral must translate into transfer control signals.
c. Each peripheral must contain gating circuits at the input of a receiving register and at the output of a transmitting register (the functions of these registers may be realized with a single buffer register if desired). These gating circuits must be capable of strobing data into or out of the registers when triggered by a command from the device selector.
d. Each peripheral must contain a "Busy/Done" flag (a flip-flop) and gating circuit, which together can assert the OMNIBUS SKIP line when commanded by an IOT instruction. When the flag is set, the peripheral is ready for a word transfer.

PRINCIPLES OF PROGRAMMED I/O TRANSFERS

The simplest and most straightforward type of transfer is the programmed transfer. Rather than responding to an interrupt request and checking each flag to find out which device made the request, the program remains in a continuous wait loop. This method is convenient when the purpose of the processor is to service one or more peripherals. If the user desires to do processing in addition to servicing peripherals, he should employ the interrupt facility.

The IOT Instruction

The nature of the IOT instruction is summarized as follows:

a. PROCESSOR OPERATION CODE—IOTs use operation code 6 and are one-cycle augmented instructions.
b. DEVICE SELECTION—The middle six bits of the instruction word are used for device selection; each peripheral decodes these bits and responds to the IOT only if it sees its particular number or device code. (There is an obvious corollary to this statement. Peripherals generally do not share the same device code.)
c. DEVICE OPERATION CODE—The last three bits of the IOT are used to tell the device what to do. The method varies, depending on whether the device plugs into the OMNIBUS or is at-

tached to the External I/O Bus interface. The end result is the same, however, in that the last three bits define the operation to be performed. The usage is as follows:

Binary Value			Octal Value	Conventional Usage
Bit 9	Bit 10	Bit 11 (LSB)		
0	0	1	1	Sampling flags, skipping
0	1	0	2	Clearing flags, clearing AC
1	0	0	4	Reading, loading, and clearing buffers

Flags

In the control section of every I/O device is a flag, or status flip-flop. The flag is cleared when the computer is first turned on. It can be cleared by one of the device's IOTs, and is set whenever the device finishes its operation. The state of this flag can also be tested, usually by an IOT that causes the next instruction to be skipped if the flag is set.

Input devices (from peripheral to processor) set flags when they have data to be serviced by the processor.

Output devices (from processor to peripheral) receive an IOT instruction that clears the flag. When the device has processed the data received from the processor, it then sets the flag (to let the processor know that it is ready for a new instruction).

In order to understand the purpose of a flag, consider the following situation:

A program requires that a number be entered into the program via the Teletype keyboard. Obviously, the operator might elect to take a lot of time deciding what number to enter. Equally obviously, the program must wait until that number has been entered; otherwise, wrong computation will take place. Thus, a synchronization problem exists. The processor must wait until the operator has made up his mind.

This problem is easily overcome by making use of the flag. The flag sets when the keyboard electronics has a character available in its buffer. Furthermore, the program can find out when the flag sets by means of the skip IOT. All that has to be done is to use the following instructions:

```
Sometimes Called a "Wait Loop"
        KSF       /SKIP IF THE KEYBOARD FLAG IS SET.
        JMP .-1   /IF YOU DIDN'T SKIP THE LAST TIME, TRY AGAIN.
                  /EVENTUALLY THE PROGRAM WILL GET PAST THE JUMP INSTRUCTION.
                  /WHEN THE OPERATOR STRIKES A KEY.
```

Note that "JMP .—1" means "jump back to previous instruction." Assuming that the flag becomes set, the program then clears the flag and brings the character into the AC. If it does not clear the flag at this time, the device will not get a second character. Also, the program must

move the character into the accumulator so that it can process the character (store or operate upon it). This is illustrated by:

```
.
KCC          /CLEAR THE KEYBOARD FLAG.
KRS          /BRING THE CHARACTER INTO THE AC.
.
.
```

A more convenient method of accomplishing this in one instruction as illustrated by the following:

```
KSF          /WAIT FOR THE FLAG TO SET.
JMP .—1
KRB          /CLEAR THE FLAG AND GET THE CHARACTER.
.
.
```

Data Transfers

As seen from the example above, data is moved from a peripheral into the AC under program control. However, the input transfer may be an OR with the previous contents of the AC, or it may be a jam transfer, depending on the design of the peripheral: therefore, the user should consult the detailed IOT listing for the specific peripheral (given in Chapter 7).

Output transfers work in a similar manner. Data is loaded into the AC using the TAD instruction. Then an output IOT is given to move the data to the peripheral and (usually) to initiate some sort of operation. The flag is used slightly differently, but it serves much the same purpose. Consider the following example involving the Teletype printer:

```
.
TAD DATA     /GET A CHARACTER TO BE PRINTED
TLS          /SEND IT TO THE PRINTER ELECTRONICS, CLEAR
             /THE FLAG, AND SAY "GO."
TSF          /HANG AROUND UNTIL THE FLAG SETS, SO YOU
JMP .—1      /DON'T ACCIDENTALLY TRY TO PRINT TWO
.            /DIFFERENT CHARACTERS AT THE SAME TIME.
.
```

If the printer flag is set ahead of time, the flag may be tested before doing the "TLS." This technique speeds up the program, since the computer can be doing instructions instead of just waiting for the flag.

```
TFL                          /SET PRINTER FLAG
.
.                            /GET CHARACTER
TSF
JMP .—1
TLS                          /PRINT IT
.
```

The user should not try to be exotic and microcode skip and clear IOTs into one instruction. It won't always work. Trouble could develop if the flag happened to set between the skip and clear functions, because the program might have cleared the flag before it found out the flag had been set.

PRINCIPLES OF PROGRAM INTERRUPTS

GENERAL

There may come a time when waiting for flags consumes too much valuable time. What is needed is some way of ignoring peripherals until they need attention.

For instance, the device flag sets when a new character is available if the device is an input device. If the device is an output device, the flag sets when the device is ready to receive a new character. Wouldn't it be desirable to service peripherals only if one or more flags happen to set?

To accomplish this, all peripherals OR their flags onto a special line called the Interrupt Request line. If this line becomes ground, it means that there's a flag (and hence a device) somewhere that needs attention. However, the flag must **demand** attention; not just pull a line which can be tested with an IOT. The "demand attention" circuitry is known as the interrupt system, and works as follows:

1. The interrupt system is automatically turned off when the computer is first turned on.
2. The interrupt system is turned on by an IOT (ION), but the actual enabling is delayed by one instruction. (The reason for this will be shown later.)
3. If the interrupt is enabled and some device's flag gets set, the processor **automatically** responds by executing a JMS to location zero and simultaneously turning off the interrupt system. Note that this JMS is hardware-generated.
4. In case it is needed, there's an IOT (IOF) which turns off the interrupt system.

The interrupt system is a simple piece of hardware, but has far-reaching program implications. Let's examine some of those implications.

Coding a Program Interrupt. Assume, for example, that the operation is reading and punching information using the PC8/E Reader/Punch combination. Between reading and punching, the processor is doing a simple, unrelated program. Once things get started, the procedure that follows is indicated in figure 5-1.

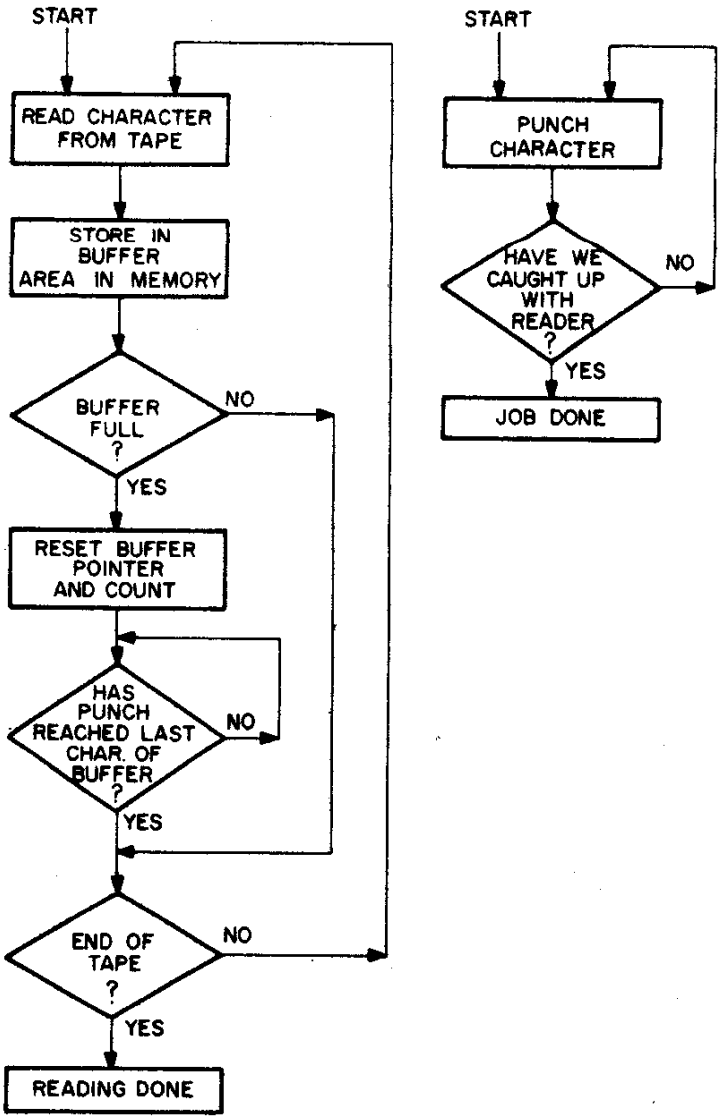


Figure 5-1 Program Interrupt Example

```
/The Interrupt Service Routine

0,      xxxx        /WHERE THE RETURN ADDRESS IS
                    /STORED WHEN THE INTERRUPT OC-
                    /CURS.

1,      JMP I.+1    /THIS INSTRUCTION GETS TO THE
2,      FLAGS       /FLAG TEST ROUTINE WHICH IS
                    /USUALLY ON OTHER THAN PAGE 0.

/The flag-testing and restoration routine

FLAGS,  DCA AC      /SAVE AC
        RAL
        DCA LNK     /AND LINK
        RSF         /CHECK READER FLAG
```

```
            SKP
            JMP RDR
            PSF             /AND PUNCH FLAG
            SKP
            JMP PUNCH
            KCC             /IF WE GET HERE, SOMEONE PROB-
                            /ABLY STRUCK A TELETYPE KEY
            TCF             /OR SOME OTHER STRANGE INTER-
                            /RUPT EXISTS.
DISMIS,     CLA CLL         /HERE'S HOW WE EXIT
            TAD LNK         /RESTORE LINK
            RAR
            TAD AC          /AND AC,
            ION             /TURN ON THE INTERRUPT
            JMP I 0         /AND USE THE ONE-CYCLE DELAY
                            /TO GET THE PC LOADED.
/THE READER SERVICE ROUTINE. WE KNOW THE FLAG IS SET AL-
/READY, OR WE WOULDN'T BE HERE.
RDR,        RRB             /GET CHARACTER FROM READER,
                            /CLEAR FLAG
            DCA I 10        /SAVE IN BUFFER
            ISZ COUNT1
            JMP RDGO        /BUFFER NOT FULL
            TAD BUFF        /BUFFER FULL
            DCA 10
            TAD KCOUNT
            DCA COUNT1
            JMP DISMIS      /NOTICE-WE DID NOT MOVE
                            /READER.
RDGO,       RFC             /MOVE READER
            JMP DISMIS      /AND EXIT
/THE PUNCH SERVICE ROUTINE
PUNCH,      TAD I 11        /GET WORD FROM BUFFER.
            PLS             /PUNCH IT AND CLEAR FLAG
```

```
                CLA
                TAD 11          /COMPARE ADDRESS PUNCHED
                CIA
                TAD 10          /WITH READER ADDRESS
                SNA CLA
                JMP FINISH      /ADDRESSES EQUAL, SO JOB DONE.

                ISZ COUNT2
                JMP DISMIS
                TAD KCOUNT      /TOP OF BUFFER, SO RESET
                DCA COUNT2      /COUNTER
                TAD BUFF
                DCA 11          /AND POINTER.
                JMP RDGO        /AND RESTART THE READER.
/OK—THAT TAKES CARE OF THE INTERRUPT HANDLER. NOW COMES
/THE MAIN PROGRAM.
START,          RFC             /START READER
                PLS             /AND PUNCH
                TAD KCOUNT      /INITIALIZE ALL VARIABLES
                DCA COUNT1
                TAD KCOUNT
                DCA COUNT2
                TAD BUFF
                DCA 10
                TAD BUFF
                DCA 11
                ION             /AND TURN ON INTERRUPT
                ISZ PHONEY      /THIS IS A TRIVIAL NULL
                JMP .—1         /JOB WHICH SLOWLY INCREMENTS
                IAC             /THE AC.
                JMP .—3
FINISH,         HLT
/CONSTANTS
```

```
KCOUNT,     -100
BUFF,       377
/VARIABLES
COUNT1,     0
COUNT2,     0
AC,         0
LNK,        0
PHONEY,     0
```

The reader stops automatically when tape runs out of the reader. When the punch pointer becomes equal to the reader pointer, the job is finished. Notice how the one-instruction delay built into the ION instruction allows the computer to obtain the return address from location 0 before new interrupts can occur.

In the above example, the background null job (the job being done when interrupts were not being serviced) was unrelated to the interrupt routine. In the more general case, the background job is some sort of processing job operating on the buffer after reading and before punching. Clearly, the programming problem is more severe. The reader must be ahead of the processing, which must be ahead of the punching. Such problems are beyond the scope of this chapter, and the reader should, therefore, consult Chapter 5 of "Introduction To Programming."

THE PDP-8/E System fits just about anywhere. The user who begins with an inexpensive system can later convert his table-top system to a larger cabinet configuration and still retain most of his hardware.

CHAPTER 6

DATA BREAK TRANSFERS

GENERAL

Data break (sometimes called Direct Memory Access or DMA) is another form of data transfer used with high speed devices. Generally, a mass-storage device utilizes this form of data transfer. In the case of a high-speed device such as a disk, it becomes desirable to transfer a block of information at the fastest possible transfer rate. A block of data containing up to 4K words would require one data break for each word transferred.

Data break is the process of stealing memory cycles for the purpose of adding data to memory or removing data from memory without disturbing the major registers within the processor. Data breaks, therefore, stall the processor only during that period of time when some action is required by the processor, such as transferring data in or out of memory.

Data break is desirable when a large quantity of words is to be transferred. Transferring one 12-bit word is a relatively easy task to accomplish. If a word is to be transferred out to some peripheral, the only requirement would be to stop the program and jump to some subroutine that would address memory and transfer data out to that peripheral. Therefore, all that would be needed is a program subroutine, a memory address, and a 12-bit register to receive the 12-bit word at the peripheral end. It would obviously be more practical to use a programmed type of data transfer rather than a data break transfer.

To use data break transfers effectively, a series of words should be transferred and the peripheral should be considered a high-speed peripheral. To accomplish this, however, the process becomes more complicated. For example, if 50 words are to be transferred, 50 addresses are necessary, along with a means of telling the processor when the 50 words have been transferred. Thus, an address incrementing device is required to add a number to the current address and a word count mechanism is required to count the number of transfers.

THE BASIC DATA BREAK SYSTEM

The purchaser of a PDP-8/E and a data break peripheral receives, in addition to his basic programming package, subroutines corresponding to the data break device and a Programmer's Reference Manual to initially set up and program the device. Such subroutines might include the Interrupt Service Routine, the Search Subroutine, and the Read and Write Subroutines, depending upon the type of data break peripheral. These are first loaded into memory to provide the necessary IOTs required for addressing and initializing transfer control. The user then calls these subroutines.

The basic data break system is illustrated in Figure 6-1. The user calls for the data break, and the software routine initializes the data break transfer. The peripheral control contains a break priority circuit, a current address register, a word count register, and a data register.

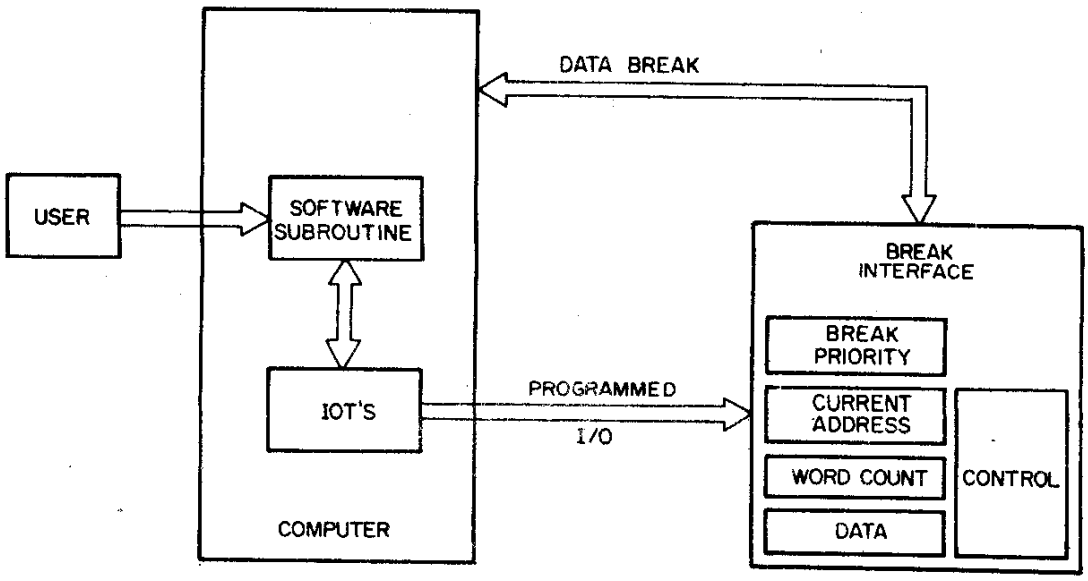


Figure 6-1 General Block Diagram of Basic Data Break Interfaces

Its function is to address memory, count the number of word transfers and receive or transmit data.

Current Address (CA) Register
The Current Address Register provides the ability to sequentially address a series of memory locations by incrementing before each transfer. Initially, the CA register is loaded with an address that is one location before the desired address. The register is then incremented (a one is added to the register) and the address is placed onto the memory data lines.

Word Count (WC) Register
The Word Count Register serves to count the number of data words in a block of data that is transferred from or to a peripheral. The two's complement of the number of words to be transferred is placed in the register initially. The register increments at each word transfer until the WC register contains zeros. When the WC register overflows, a signal is generated which clears the enable circuits.

Data Break Priority
Up to 12 data break devices can be simultaneously attached to the PDP-8/E. Each device is assigned a line on the OMNIBUS for use in determining priority.

Bit 0 represents the highest priority, and bit 11 represents the lowest. When a device makes a break request, the priority of the other devices also making a break request is tested. This guarantees that a faster and higher priority device will make the first data transfer.

Data Register
Data transfer occurs between the device Data Register and the Memory Buffer Register. The data register only serves either to receive or to transmit data.

Data Break Configurations

Two types of data break configurations (single-cycle and three-cycle) may be used on the PDP-8/E.

Single-cycle devices contain the WC and CA registers in the peripheral. Once data transfer begins, the processor stalls only once (one cycle is stolen) while the data transfer takes place.

Three-cycle devices use memory to contain the WC and CA registers. The processor then must stall one cycle for the word count, one cycle for the current address, and one cycle for the word transfer.

ONE-CYCLE DATA BREAK TRANSFERS

A simplified block diagram of the one-cycle data break operation is illustrated in Figure 6-2, and a flow chart showing the interaction between the processor and the device is illustrated in Figure 6-3. Because this is a one-cycle data break device, Figure 6-2 shows the word count and current address registers in the peripheral.

Figure 6-3 divides the chart in terms of actions required by the processor and actions required by the device. The "flow of events" time periods are divided to reflect the INITIAL SET-UP, DATA TRANSFER, and EXIT.

Initial Set-Up

The Program Subroutine initializes the word count (WC) and current address (CA) hardware registers located in the peripheral and generates IOTs to enable the device control logic and specify the direction of transfer.

Data Transfer

The 12-bit word count register becomes incremented just prior to a word transfer and provides an overflow signal to the control logic after the register is incremented to all zeros.

The 12-bit current address register is incremented at the same time as the WC register. If an overflow in the WC register occurs, the device clears the enable circuits after the current word has been transferred. Before the current address can be placed on the memory address lines, a priority test must be made. If the device is the highest priority, the contents of the CA register are placed on the memory address lines.

After the priority test, the break device generates CPU disabling signals which tell the processor to stall while a word is transferred into or out of memory. While the contents of the CA registers are placed on the MA lines, a memory direction signal line (transfer into or out of memory) is enabled. The transfer of data occurs between the memory buffer register and the device data register.

At the end of the memory cycle, the CPU disabling signals are removed and the processor resumes operation. If the device is still enabled (WC overflow has not occurred), it then repeats the data transfer cycle. If WC overflow has occurred, the device begins an EXIT.

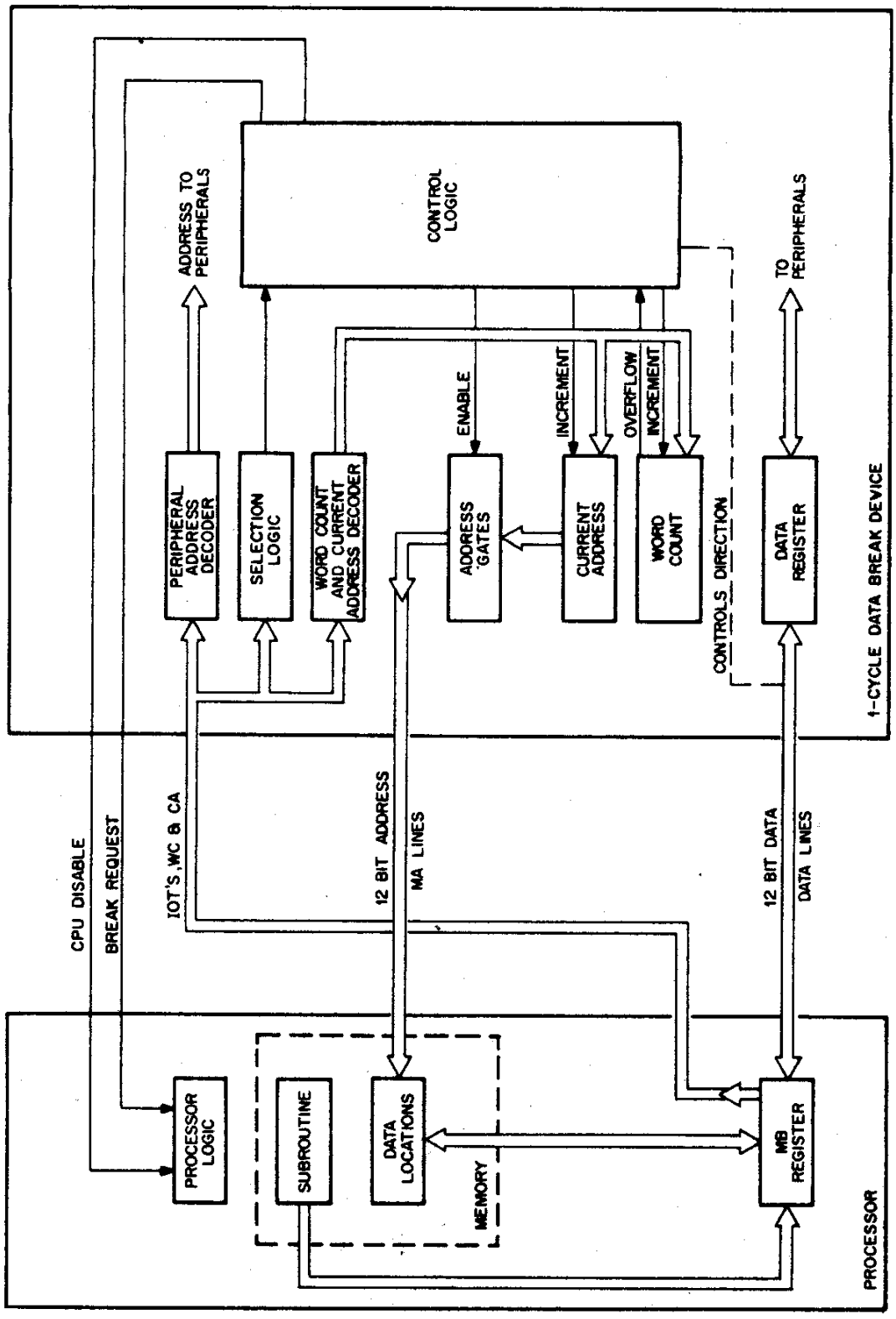


Figure 6-2 1-Cycle Data Break Simplified Block Diagram

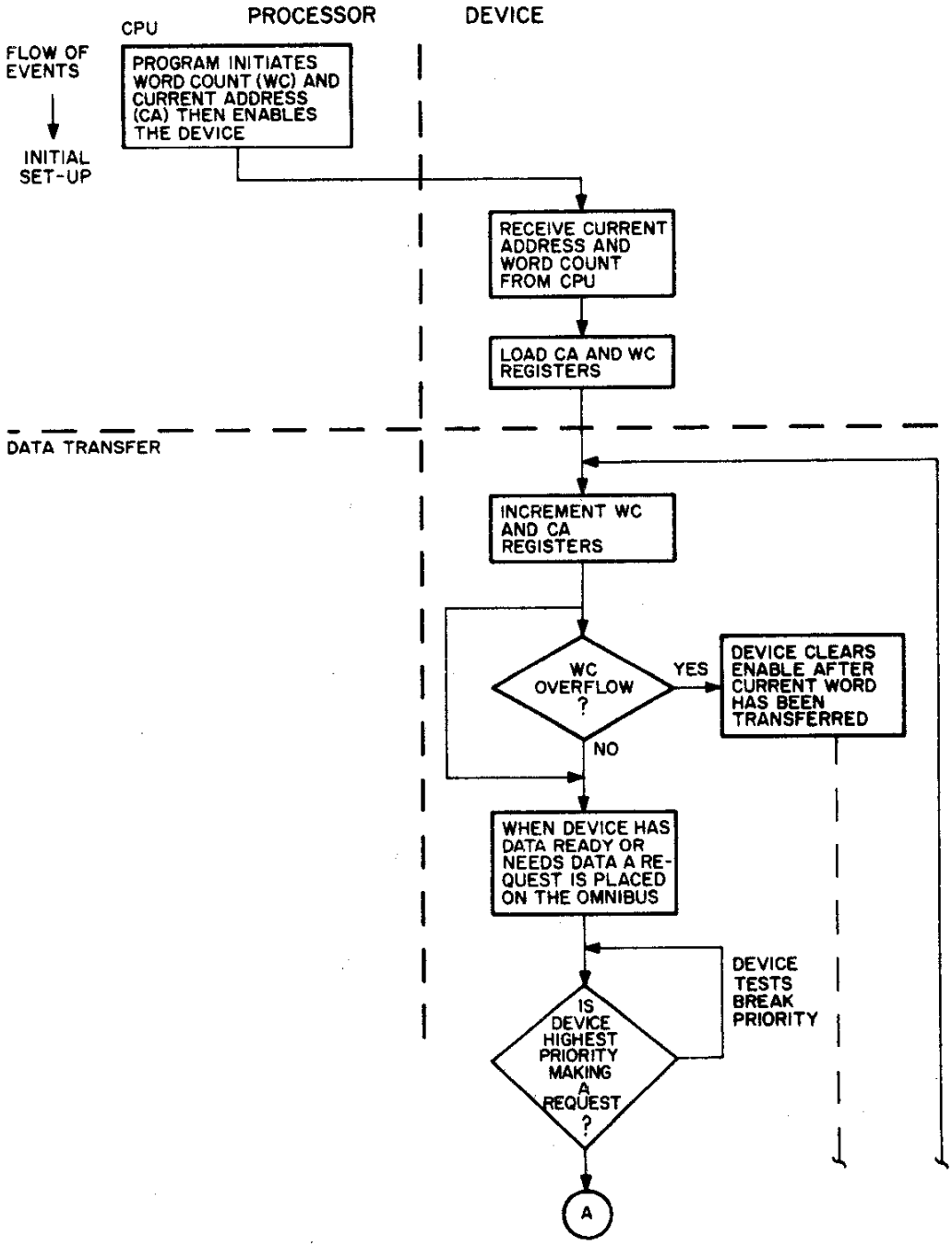


Figure 6-3 1-Cycle Data Break Flow Chart

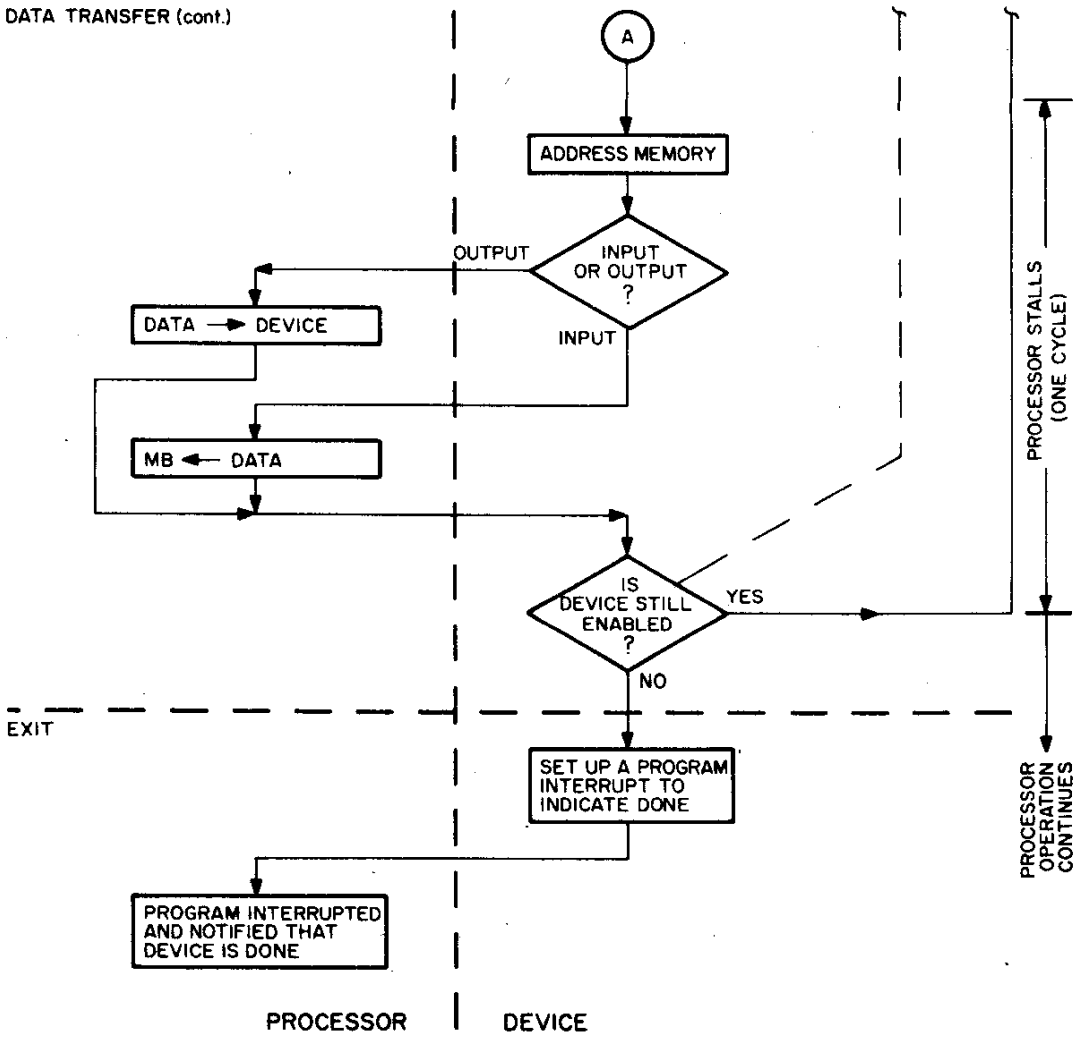


Figure 6-3 1-Cycle Data Break Flow Chart (Continued)

EXIT
To exit, the device sets a flag indicating that the block transfer has been completed. The processor then responds to this flag in the manner described in Chapter 5.

THREE-CYCLE DATA BREAK TRANSFERS

A simplified block diagram of the three-cycle data break operation is shown in Figure 6-4, and a flow chart illustrating the interaction between the processor and the device is shown in Figure 6-5. The WC and CA registers are located in memory. Therefore, the program loads the WC register with the required word count and the CA register with the address where the data will be either stored or retrieved.

The flow chart in Figure 6-4 is divided into time periods to reflect the INITIAL SET-UP, WORD COUNT, CURRENT ADDRESS, DATA TRANSFER, and EXIT. Because the word count and current address registers are located in memory, two additional data break cycles are required. Incrementing of both the CA and WC registers is accomplished in the processor.

Initial Set-Up

For a three-cycle operation, the data break subroutine must load the word count and current address memory locations. In addition, IOTs to enable the data break control logic and peripheral addresses are generated by the subroutine and transferred to the device.

Once the device decodes the IOTs, a break request is initiated and a priority test is performed. If the device has the highest priority, it generates a group of CPU disable signals and the system begins a word count cycle.

Word Count

The memory address of the word count register is hard-wired in the peripheral. This address is gated into the break address register and placed onto the memory address lines. An overriding line to the address register forces a zero into bit 11 (for an even address).

The processor then fetches the contents of the word count register to the memory buffer register. A one generated by the control logic is forced into bit 11 of the data bus, and transferred to the memory buffer register via the adder circuits. The resulting addition is tested for overflow. The contents of the memory buffer register are immediately placed into the word count register. If word count overflow occurred, the device clears its enable after the current word has been transferred. Just prior to entering the current address cycle, the priority is tested again.

Current Address

Updating the Current Address Register is accompilshed in the same manner as Word Count. However, instead of a zero being forced into bit 11 of the break address register, a one is used.

At the end of the current memory cycle, the contents of the MB register are written back into the CA register and also transferred to the device break address register. On the next memory cycle, the device break

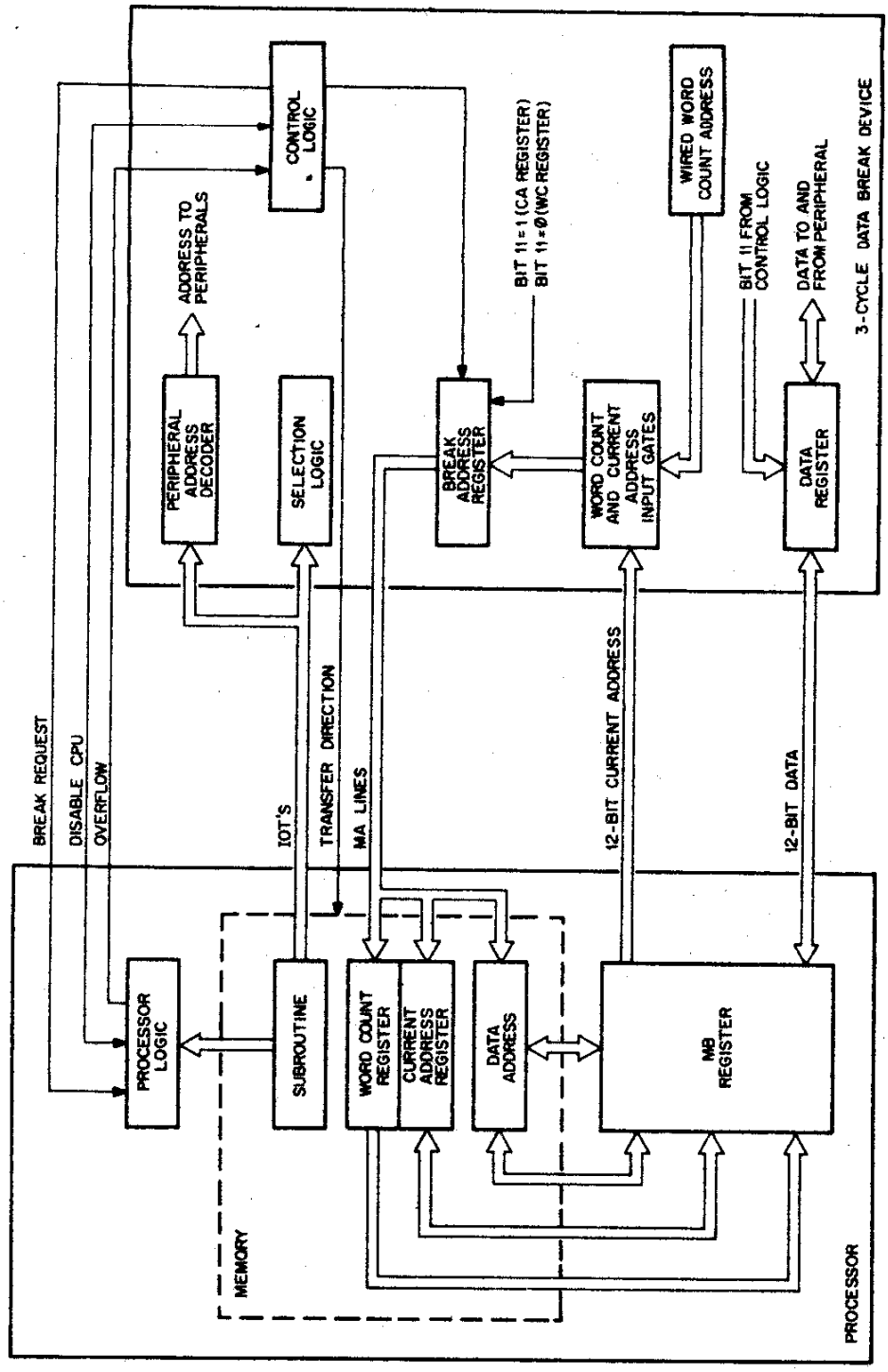


Figure 6-4 3-Cycle Data Break Simplified Block Diagram

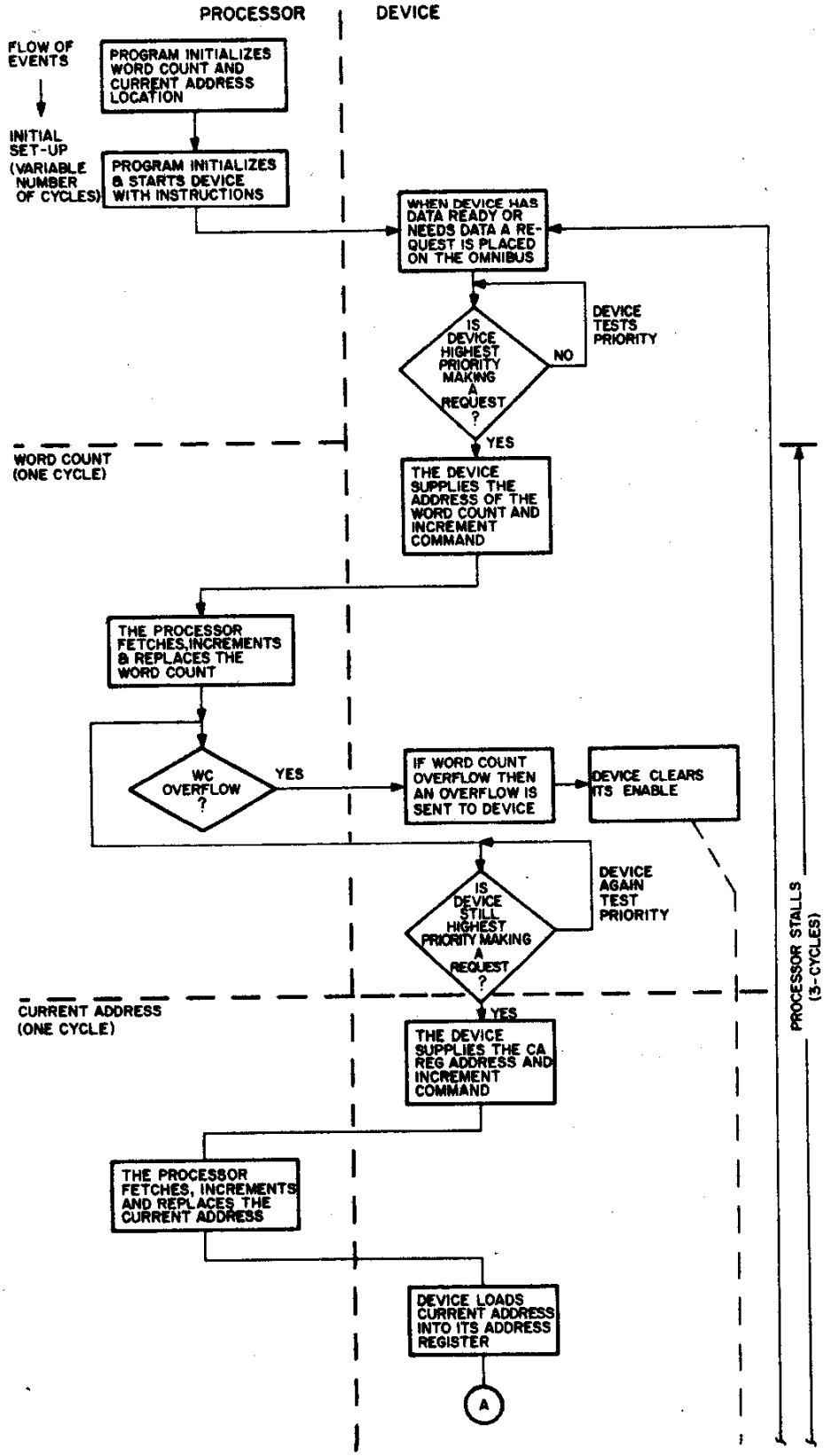


Figure 6-5 3-Cycle Data Break Flow Chart

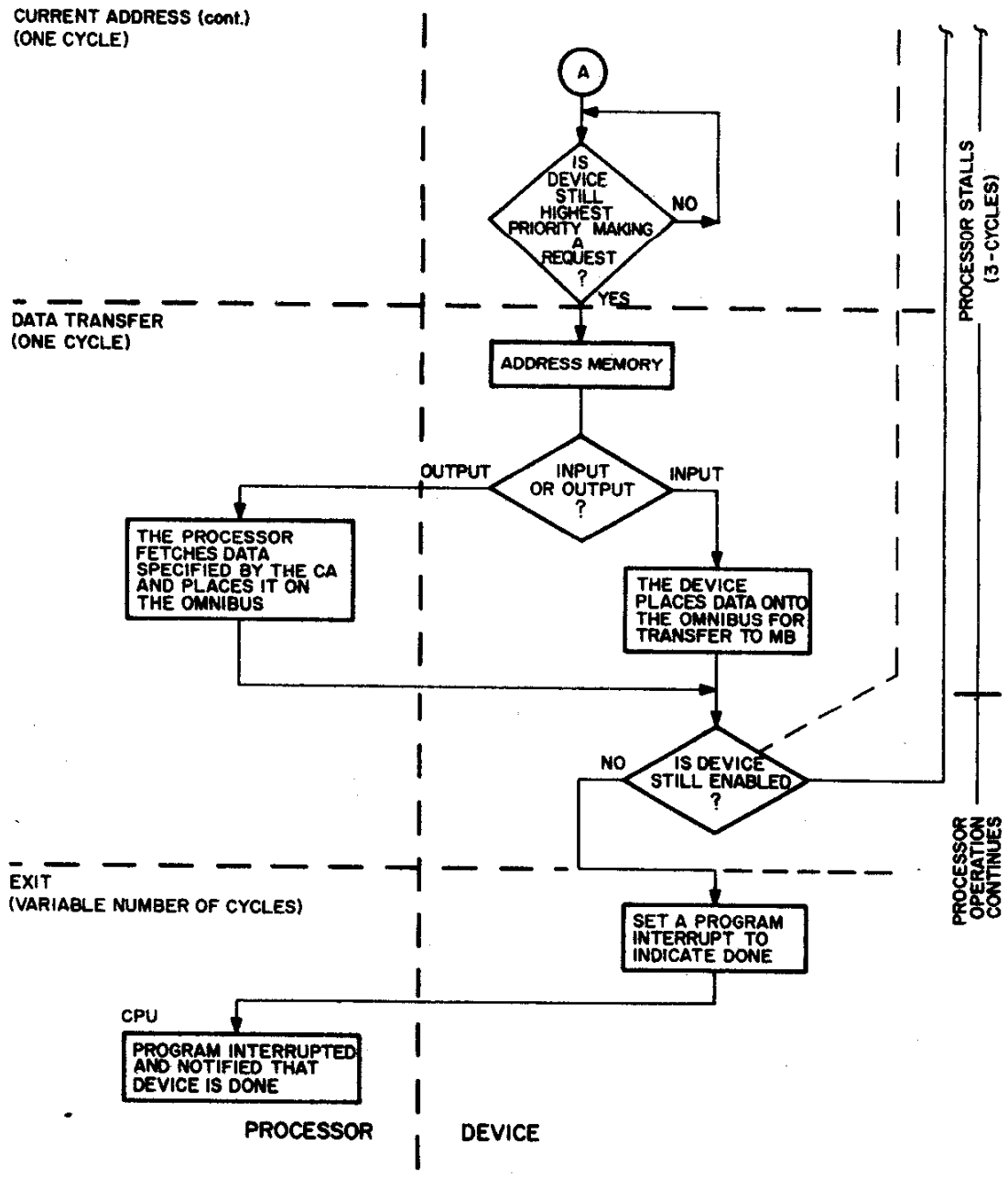


Figure 6-5 3-Cycle Data Break Flow Chart (Continued)

address register points to the data address and the system is now ready to transfer data. Just prior to addressing the data location, another priority test is made.

Data Transfer

When the current address is placed on the address lines, a transfer direction signal is generated, and data is placed onto the OMNIBUS data lines. The data transfer is between the device data register and the *memory buffer register. The CPU disabling* signals are removed at the end of the cycle. If the WC overflow signal has not been received, the device is still enabled. A break request is again initiated and the word count—current address—data transfer cycle is repeated. A word count overflow causes the device to set its device flag and to begin an exit.

Exit
A flag checking subroutine terminates the data break.

PROGRAMMING EXAMPLE

An example of programming for the data break operation is given below. This example deals with programming a DF32 Disk, and provides a subroutine only to transfer a block of words between Disk and Memory. It does not include the subroutines which are necessary to determine the word count, starting address in memory, and disk address. This example serves to illustrate the software required with data break.

BLOCK TRANSFER SUBROUTINE

Subroutine Call

```
•
•
•                       /PRECEDING INSTRUCTION OF THE MAIN
                        /PROGRAM
JMS I DRW               /DRW CONTAINS ADDRESS OF DISK ROUTINE
DEA & MEA               /CONTENTS TO BE PLACED IN THE DISK
                        /AND MEMORY EXTENDED ADDRESSES
CA                      /CONTENTS TO BE PLACED IN THE CURRENT
                        /ADDRESS REGISTER
WC                      /CONTENTS TO BE PLACED IN THE WORD
                         /COUNT REGISTER
MEMDIR                  /CONTENTS CONTAINS A 2 FOR READ AND 4
                        /FOR WRITE
DA                      /CONTENTS TO BE PLACED IN THE DISK
                        /ADDRESS
•                       /NEXT INSTRUCTION IN MAIN PROGRAM
•
•
•
DRW, DISK
DISK,   0               /RETURN POINTER
        TAD I DISK      /TRANSFER CONTENTS TO BE PLACED IN
                        /DEA and MEA TO AC REGISTER
        ISZ DISK        /MOVE POINTER TO NEXT LOCATION OF CALL
                        /(CA)
        DEAL            /LOAD DISK EXTENDED ADDRESS
        CLA             /CLEAR AC
        TAD I DISK      /TRANSFER CONTENTS TO BE PLACED IN CA
                        /REGISTER TO THE AC REGISTER
        ISZ DISK        /MOVE POINTER TO NEXT LOCATION (WC)
        DCA I DISKCA    /STORE UPDATED CURRENT ADDRESS
        TAD I DISK      /TRANSFER THE CONTENTS TO BE PLACED
                        /IN THE WC REGISTER TO AC
        ISZ DISK        /MOVE POINTER TO NEXT LOCATION (MEM
                        /DIR)
        DCA I DISKWC    /STORE UPDATED WORD COUNT
        TAD I DISK      /TRANSFER THE CONTENT OF MEM DIR (2
                        /OR 4) TO AC
        ISZ DISK        /MOVE POINTER TO NEXT LOCATION (DA)
```

```
        TAD IOT         /ADD 6601 TO THE AC TO BUILD READ OR
                        /WRITE IOT
        DCA GO          /DEPOSIT IOT TO BE EXECUTED
        TAD I DISK      /TRANSFER DISK ADDRESS TO AC
        ISZ DISK        /MOVE POINTER TO THE NEXT MEMORY
                        /LOCATION (MAIN PROGRAM)
GO,     0               /EXECUTE IOT TO LOAD DISK ADDRESS
                        /REGISTER AND BEGIN TRANSFER
        DFSC            /SKIP ON COMPLETION FLAG
        JMP .-1         /CHECK FLAG AGAIN
        DFSE            /SKIP ON NO ERROR FLAG
        JMP ERR         /JUMP ERROR
        JMP I DISK      /DISK CONTAINS ADDRESS OF NEXT
                        /INSTRUCTION OF MAIN PROGRAM
DISKCA, 7751
DISKWC, 7750
IOT,    6601
```

part 2

PDP-8/E OPTIONS

CHAPTER 7

PDP-8/E OPTIONS

GENERAL

Chapter 7 contains descriptions of all the standard peripheral devices that are optionally available for the PDP-8/E computer. Section 1 deals with the mechanical expansion options. Section 2 covers the computer internal options. Section 3 describes the OMNIBUS input/output equipment options. Section 4 is concerned with the external bus input/output equipment options.

SECTION 1 MECHANICAL EXPANSION OPTIONS

Included in this section are those options which affect the external physical properties of the PDP-8/E computer, such as cabinets and panels. Further details regarding installation of these options appear in Chapter 11 of this handbook.

SYSTEM EXPANDER BOXES

Type BA8-AA System Expander Box

The BA8-AA includes a power chassis assembly and OMNIBUS assembly, capable of accommodating up to 20 PDP-8/E modules, and a BC08H-3F Cable Set (three and a half feet in length) with rack-mountable slides included, as well as a Type KC8-EB blank front panel.

Type BA8-AB System Expander Box

The BA8-AB includes a power chassis assembly and OMNIBUS assembly, capable of accommodating up to 20 PDP-8/E modules, and a BC08H-3F Cable set (three and a half feet in length) with table-top cover included, as well as a Type KC8-EB blank front panel.

Type BE8-A OMNIBUS Expander

The BE8-A includes an additional OMNIBUS assembly, capable of accommodating up to 20 PDP-8/E modules, together with M935 Bus Connectors for expanding either the PDP-8/E, the BA8-AA, or the BA8-AB to 38 slots.

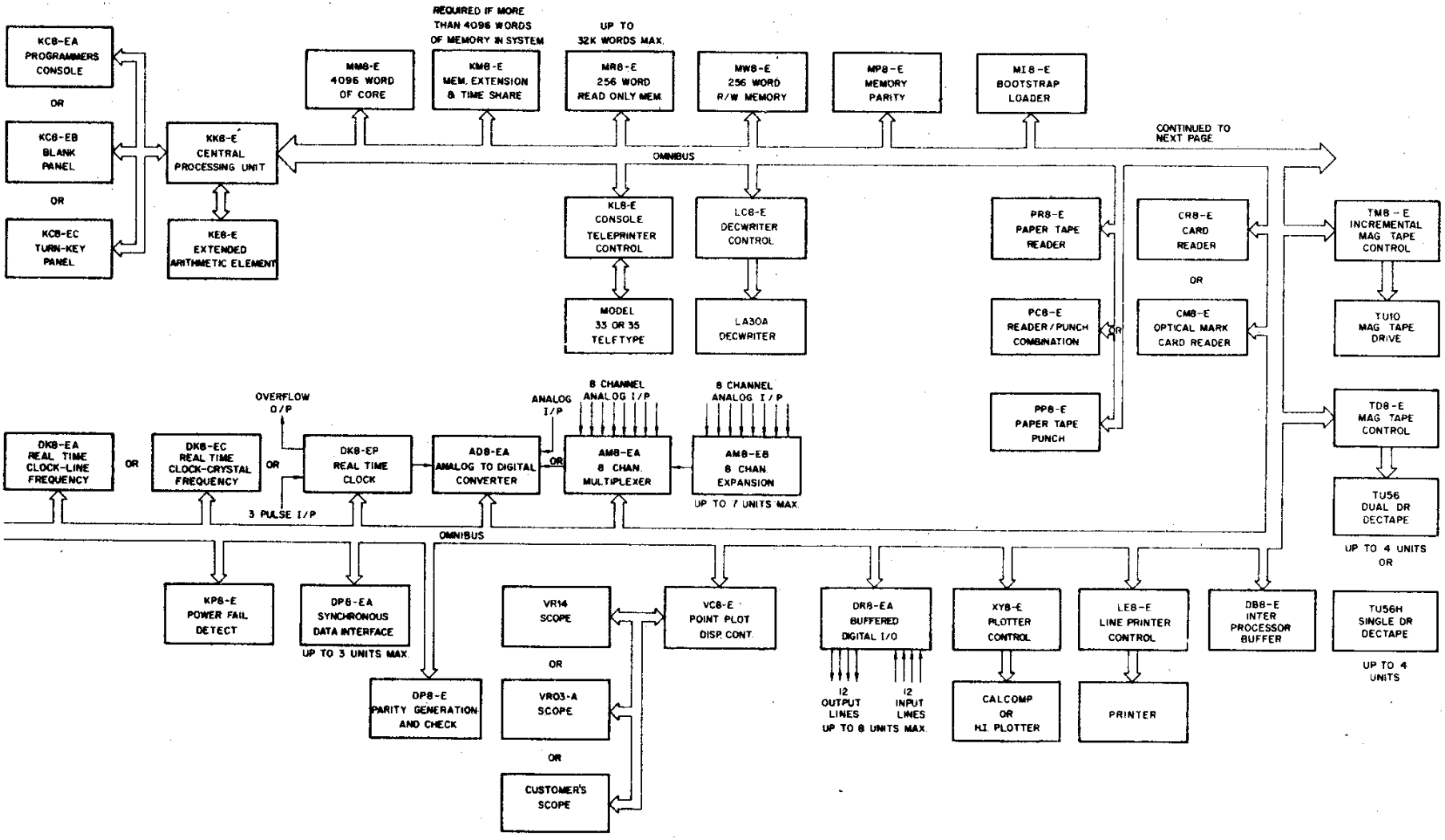
PANEL OPTIONS

Type KC8-EC Turn-Key Front Panel

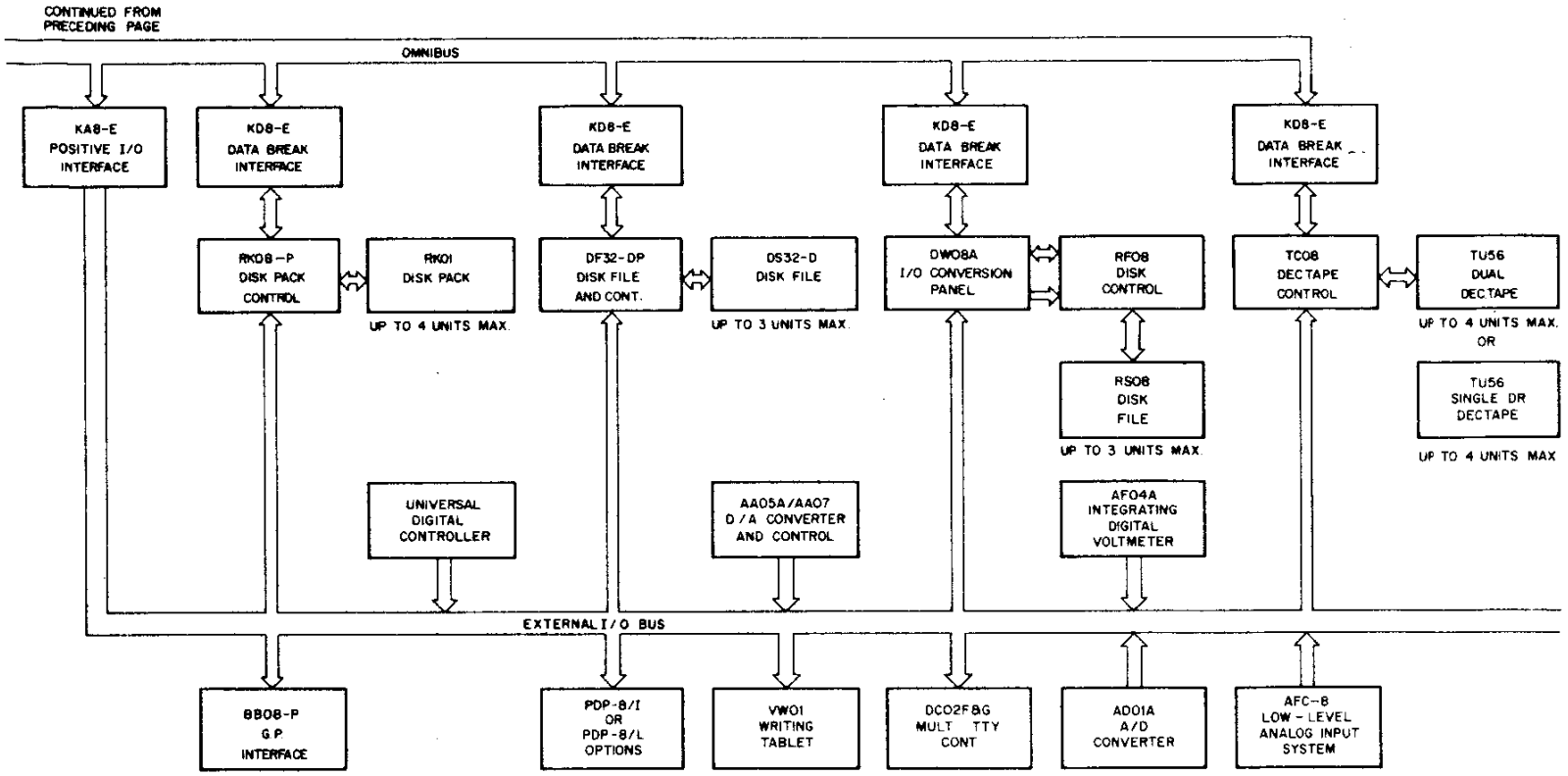
The KC8-EC panel contains a key-lock ON/OFF switch for controlling the application of power to the PDP-8/E system. The KP8-E Power Fail option is a prerequisite for using this panel. It is used as an alternate panel to the PDP-8/E.

Type KC8-EB Blank Front Panel

This panel can be used in place of the KC8-EA panel or the KC8-EC panel on the PDP-8/E, and is supplied with the BA8-AA or BA8-BA Expander Box option. It is used on the PDP-8/E when the system is equipped with a KP8-E, and the user wishes to switch power externally.



PDP-8/E System Block Diagram



PDP-8/E System Block Diagram (Continued)

SECTION 2 COMPUTER INTERNAL OPTIONS

This section deals with internal options for the PDP-8/E computer, including those concerned with the extended arithmetic facility, memory equipment, real-time measurement, and power fail detection and restart.

Many of these are covered in greater detail in Chapters 3 and 4 of this handbook.

The execution time for IOT instructions in this section is 1.2 μs, except where otherwise specified in the instructions for extended arithmetic.

TYPE KE8-E EXTENDED ARITHMETIC ELEMENT (EAE)

The KE8-E option plugs into the PDP8-E OMNIBUS to enable the central processor to perform arithmetic operations at high speeds by incorporating the EAE components with the existing central processor logic so that they operate asynchronously. This two-module option consists of circuits that perform parallel arithmetic operations on positive binary numbers, and includes:

a. A 5-bit Step Counter (SC) Register. This register is used to record the number of steps performed, and stops many EAE instructions after the correct number of operations. For these instructions, the SC is automatically loaded, and the instruction is terminated when the SC becomes a fixed number. There is one instance, the normalize (NMI) instruction, when the SC is of interest to the programmer. For this reason, instructions allow the programmer to load the SC from memory or the AC, depending upon the mode of operation; and to transfer the contents of the SC to the AC for storage upon interrupt or for program analysis.

b. A 4-bit instruction register (EAE IR)—This register consists of flip-flops set to MB (6,8-10) during the Fetch cycle of an EAE instruction.

c. The EAE timing and control logic—all EAE logic is contained in two modules which plug into the OMNIBUS. The KE8-E EAE logic circuits are used in conjunction with the accumulator (AC), link (L), multiplier quotient (MQ), and memory buffer (MB), though asynchronously with them to perform arithmetic operations. When this option is added to a PDP-8/E system, a class of instructions is added to the Group 3 Operate instruction list.

d. A mode flip-flop which controls the instruction set of the EAE. The mode flip-flop is set to mode A when power is applied to the machine, when the CLEAR key on the panel is operated, and when the CAF instruction is issued.

PROGRAMMING

The Extended Arithmetic Element (EAE) microinstructions are specified by an operate instruction (operation code 7) in which MB(3) and MB(11) always contain binary 1's. The instruction set is arranged so that programs written for the PDP-8/I EAE can be run on the PDP-8/E without modification. A greatly expanded instruction set is also available for new programming.

Two modes of operation, hereafter designated Mode A and Mode B, are available. Mode A, which is the mode in which the computer starts, is the PDP-8/I compatible mode.

COMMON OPERATIONS

Several EAE operations may be executed in either mode of operation. The common features of these operations are describd below.

Two-word instructions

Many EAE instructions require more than 12 bits. For these instructions, a second 12-bit word is obtained from the next location in memory. The second word is interpreted by the EAE hardware, and used either as an argument or the address of an argument. Program resumes at the location following the second word.

Multiplication

The Multiply instruction is a two-word instruction. The multiplier is either the second word or is located in the address specified by the second word, depending upon the mode. The contents of the MQ are multiplied by the multiplier and the 24-bit result is left in the AC (most significant bits), and MQ. The multiplication is an unsigned integer multiply, i.e. the multiplier and multiplicand are treated as 12-bit positive numbers with binary point at the right-hand end of the word. The binary point of the product is at the right-hand end of the MQ. If the AC is non-zero at the start of the multiply, its contents are added to the product. The Link is cleared. The SC is used in the execution of this instruction.

Division

The Divide Instruction is a two-word instruction. The division is either the second word or is located in the address specified by the second word. The contents of the AC (most significant bits) and MQ are divided by the divisor, and the quotient and remainder are left in the MQ and AC respectively. The division is an unsigned integer divide. The Link is cleared if the first subtraction produces a negative result, indicating that divide overflow has not taken place. If the first subtraction produces a positive result, the Link is set (indicating overflow) and the division is terminated. The contents of the AC and MQ are modified if divide overflow occurs. Ordinarily, the divide instruction is followed by a test of the Link to check for overflow before more computation occurs. The SC is used in the execution of this instruction.

Left Shift

The Link, AC and MQ are treated as one long register. The previous content of the Link is lost, AC0 is shifted into the Link, MQ0 is shifted into AC11, and zero enters the vacated MQ11 position. The second word of the two-word shift left instruction is loaded into the SC and thus defines the number of shifts to be performed.

Logical Right Shift

The Link is first cleared, then the AC and MQ, but not the Link are treated as one long register. MQ11 is either lost or shifted into the GT flag (depending on the mode). AC11 is shifted into MQ0, and the state of the Link is loaded into AC0. This instruction effectively divides the number of the AC and MQ by two for each place shifted. As in Left Shift, the number of positions shifted is defined by the last five bits of the second word of the two-word instruction.

Arithmetic Right Shift

This operation is identical to Logical Right Shift, except that the Link is initially loaded with the content of AC0, maintaining the sign of the number in the vacated bits. Because a right shift means shifting the contents of the AC and MQ one place to the right for each place shifted, the value of the 24 bits is effectively divided by two in signed arithmetic.

Normalization

The Normalize instruction is typically used to cast out and to account for leading zeroes when performing floating-point arithmetic. The Step Counter is initially cleared; then the contents of the L, AC and MQ are shifted left, as described above under Left Shift, until AC0 and AC1 are different or until the 24 bits contained in AC and MQ contains the number $(6000\ 0000)_8$. The Step Counter is incremented once for each shift. Normalize instruction must not be "ORed" with other EAE operations. At the conclusion of the Normalize instruction, the Step Counter contains a number equal to the number of shifts that were required to perform the normalization and is the EXPONENT (the binary power of 2) the 24-bit number. Thus the normalize instruction converts the number in the AC and MQ into the format. $M*2^n$; where M is the new result in AC and MQ and n is the contents of the step Counter. (The asterisk is a commonly-used symbol for multiplication)

MODE CHANGING INSTRUCTIONS (All instructions take place in 1.2 μsec.)

Switch from A to B (SWAB)

Octal Code: 7431

Operation: If the mode flip-flop is "A", it is changed to "B". If the mode flip-flop is already B, no operation occurs.

Switch from B to A (SWBA)

Octal Code: 7447

Operation: If the mode flip-flop is "B", it is changed to "A". If the mode is already A, no operation occurs.

Mode A Instructions

EAE instructions are augmented instructions, and can be combined to perform non-conflicting logical operations, as indicated in Figure 7-1.

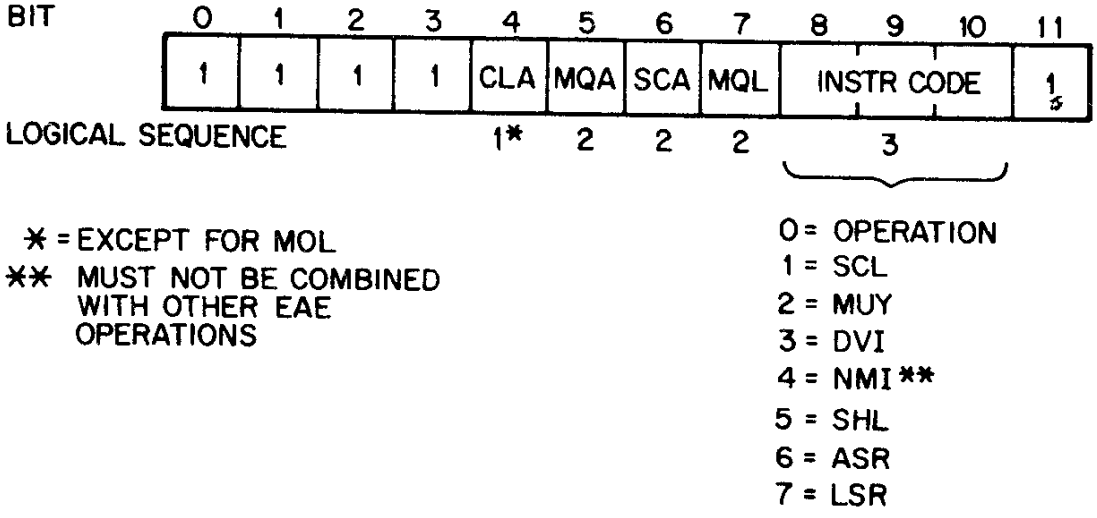


Figure 7-1 EAE Mode "A" Bit Assignments

The GT flag, explained in more detail under Mode B, is always zero for Mode A instructions. The instructions involving only bits 4, 5 and 7 have already been defined under Group 3 operate instructions in Chapter 3. For convenience, a summary of these instructions is given below. All execution times are 1.2 μs.

Mnemonic	Octal	Description
CAM	7621	0 → AC, O→MQ
MQA	7501	MQ "OR" ed with AC→AC
MQA CLA	7701	MQ→AC
MQL	7421	AC→MQ, O → AC
SWP	7521	AC→MQ, MQ→AC

The following Mode A instructions are added by the KE8-E hardware:

Step Counter "OR" with AC (SCA)

Octal Code: 7441
Execution time: 1.2 μs.
Operation: The contents of the Step Counter are "OR" ed with the five least-significant bits of the AC, and the result loaded into the AC.

Step Counter to AC (SCA CLA)

Octal Code: 7641
Execution time: 1.2 μs.
Operation: The contents of the Step Counter are loaded into AC 7-11. AC 0-6 are cleared.

Step Counter Load from Memory (SCL)

Octal Code: 7403
Execution time: 2.6 μs.
Operation: The next word in memory is treated as an operand. The one's complement of the last five bits of this operand are loaded into the Step Counter, and program resumes at the instruction word following the operand. The SCL instruction is most commonly used in interrupt servicing for restoration of the Step Counter.

Multiply (MUY)

Octal Code: 7405
Execution time: 7.4 μs.
Operation: The second word of this two-word instruction is the multiplier. Multiplication takes place as described above under "Common Operations"

Divide (DVI)

Octal Code: 7407
Execution time: 7.4 μs. if no divide overflow, 2.6 μs. if divide overflow.
Operation: The second word of this two-word instruction. Division takes place as described above under "Common Operations." Program resumes at the location following the divisor. If the Link = 1, at the conclusion of the division, divide overflow occurred; otherwise, the divide was legal.

Normalize (NMI)

Octal Code: 7411
Execution time: 1.5 + 0.3*N μs., where N is the number of shifts necessary to normalize.
Operation: The contents of AC and MQ are normalized, as described above under "Common Operations". This

command must not be combined with any other EAE commands. NMI "OR"ed with MQL is the SWAB instruction described under Mode Changing.

Shift Left (SHL)

Octal Code: 7413

Execution time: 2.6 + 0.3*N μs., where N is the number of shifts.

Operation: The number of shifts performed is equal to one more than the number in the last five bits of the second word. See "Common Operations" above for a description of Left Shift.

Arithmetic Shift Right (ASR)

Octal Code: 7415

Execution time: 2.6 + 0.3*N μs., where N is the number of shifts.

Operation: The number of shifts performed is equal to one more than the number in the last five bits of the second word. The old content of MQ11 is lost. See "Common Operations" above for a description of Arithmetic Right Shift.

Logical Shift Right (LSR)

Octal Code: 7417

Execution time: 2.6 + 0.3*N μs., where N is the number of shifts.

Operation: The number of shifts performed is equal to one more than the number in the last five bits of the second word. The old content of MQ11 is lost. See "Common Operations" above for a description of Logical Right Shift.

Mode B Instructions

Mode B differs from Mode A in the use of bit 6 of the instruction word, in the location of operands and in greatly increased double-precision arithmetic capability. As in Mode A instructions, these EAE instructions are able to be combined to form non-conflicting logical operations. See Figure 7-2 for Mode B bit assignments.

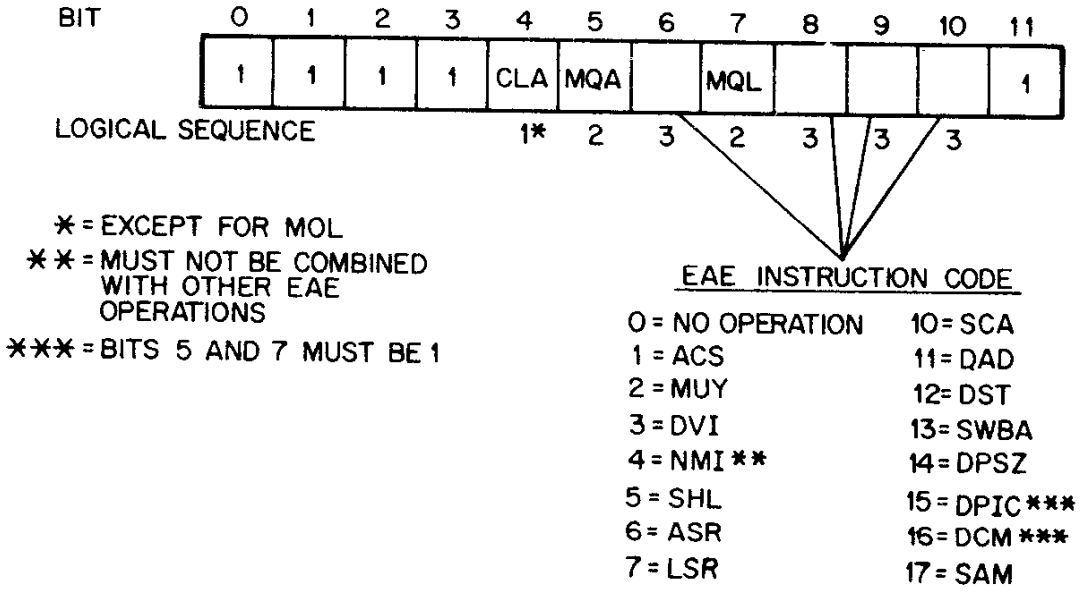


Figure 7-2 EAE Mode "B" Bit Assignments

In Mode B the GT flag, a flip-flop in the KE8-E, is activated. The GT flag may be manipulated by IOT instructions described in Chapter 3. It is loaded by the SAM instructions, and receives the contents of MQ11 on right shifts.
The instructions involving only bits 4, 5 and 7 have already been defined under Group 3 operate instructions in Chapter 3. For convenience, a summary of these instructions is given below. All execution times are 1.2 μs.

Mnemonic	Octal	Description
CAM	7621	0 → AC, 0 → MQ
MQA	7501	MQ "OR"ed with AC → AC
MQA CLA	7701	MQ → AC
MQL	7421	AC → MQ, 0 → AC
SWP	7521	AC → MQ, MQ → AC

The following Mode B instructions are added by the KE8-E hardware:

Accumulator to Step Count (ACS)

Octal Code: 7403
Execution time: 1.2 μs.
Operation: Bits 7-11 of the AC are loaded into the Step Counter, and the AC is then cleared.

Multiply (MUY)

Octal Code: 7405
Execution time: 8.6 μs.
Operation: The second word is the address of the multiplier. If extended memory is being used, the multiplier is obtained from the Data field. Multiplication takes place as described above under "Common Operations".

Divide (DVI)

Octal Code: 7407
Execution time: 8.6 μs.
Operation: The second word is the address of the divisor. If extended memory is being used, the divisor is obtained from the Data field. Division takes place as described above under "Common Operations." Program resumes at the location following the address of the divisor. If the Link = 1 at the conclusion of the division, divide overflow occurred: otherwise, the divide was legal.

Normalize (NMI)

Octal Code: 7411
Execution time: 1.5 + 0.3*N μs., where N is the number of shifts necessary to normalize.
Operation: The contents of AC and MQ are normalized as described above under "Common Operations". If the contents of AC and MQ equals 40000000, the AC is cleared. This command must not be combined with any other EAE commands. NMI "OR"ed with MQL is the SWAB instruction and has no effect if the mode is already B.

Shift Left (SHL)

Octal Code: 7413

Execution time: 2.9 + 0.3*N μs., where N is the number of shifts.

Operation: The number of shifts performed is equal to the number in the last five bits of the second word. A shift of zero is a legal command, and does not modify the L, AC or MQ. See "Common Operations" above for a description of Left Shift.

Arithmetic Shift Right (ASR)

Octal Code: 7415

Execution time: 2.9 + 0.3*N μs., where N is the number of shifts.

Operation: The Link is made equal to ACO, and remains in this state for the remainder of the instruction. The number of shifts performed is equal to the number in the last five bits of the second word. A shift of zero is a legal command which makes the Link equal to ACO, but does not modify the AC or MQ. Bits shifted out of MQ11 are shifted into the GT flag, to facilitate round-off operations. See "Common Operations" above for a description of Arithmetic Right Shift.

Logical Shift Right (LSR)

Octal Code: 7417

Execution time: 2.9 + 0.3*N μs., where N is the number of shifts.

Operation: The Link is cleared and remains cleared for the remainder of the instruction. The number of shifts performed is equal to the number in the last five bits of the second word. A shift of zero is a legal command which clears the Link, but does not modify the AC or MQ. Bits shifted out of MQ11 are shifted into the GT flag to facilitate round-off operations. See "Common Operations" above for a description of Logical Right Shift.

Step Counter "OR" with AC (SCA)

Octal Code: 7441

Execution time: 1.2 μs.

Operation: The contents of the Step Counter are "OR"ed with the five least-significant bits of the AC, and the result loaded into the AC.

Step Counter to AC (SCA CLA)

Octal Code: 7641

Execution time: 1.2 μs.

Operation: The contents of the Step Counter are loaded into AC 7-11. AC 0-6 are cleared.

Subtract AC from MQ (SAM)

Octal Code: 7457

Execution time: 1.2 μs.

Operation: The contents of the AC are subtracted from the MQ in 2's complement arithmetic. The result is loaded into the AC. The MQ remains unchanged. If a borrow is propagated from the most significant bit, the Link is set. Otherwise, the Link is cleared. Hence, the Link is set if the original AC was greater than the MQ, and is cleared if the original AC was less than or

equal to the MQ. If one wishes to compare signed numbers, the GT flag is helpful. This flag is set if the signed number in the MQ is greater than or equal to the original signed number in the AC; and is cleared otherwise. The SC is not modified. If MQ0 = original AC0, complement of MQ0 → GT
If MQ0 = original AC0, new AC0 → GT
THUS:

1 → L if AC > MQ
0 → L if AC ≤ MQ
} When AC and MQ are considered to be positive 12-bit numbers

1 → GT if AC ≤ MQ
0 → GT if AC > MQ
} When the AC and MQ are considered to be signed 2's complement 12-bit numbers.

Double-Precision Operations

These instructions are available only in Mode B. The AC and MQ are treated as a single 24-bit register, with the most significant half of the word in the AC. For operations involving addition or incrementation, the Link is set if a carry occurs from the most significant bit; if no carry occurs, the Link is cleared. The Link is not modified by the DST and DPSZ instructions. The SC is not modified by any of the double-precision instructions.

Two of the double-precision instructions (DAD and DST) are two-word instructions. For these instructions, the contents of the second word (augmented by the Data field bits, if extended memory is used) define the address of the least-significant half of a double-precision word. The most significant half of the word is located in the memory location following the least significant half-word. This format must be adhered to in order for the instruction to work as defined below.

Double-Precision Add (DAD)

Octal Code: 7443
Execution time: 5.2 μs.
Operation: The double-precision word specified by the second word is added to the previous contents of the AC and MQ; The result is left in the AC and MQ. If there is a carry from the most significant bit, the Link is set; if there is no carry, the Link is cleared. This instruction can be micro-programmed with the CAM instruction to produce a Double-Precision Load (DLD, octal 7763).

Double-Precision Store (DST)

Octal Code: 7445
Execution time: 5.2 μs.
Operation: The contents of the MQ and AC are stored at the double-precision location (two consecutive memory locations) defined by the (address) second word. The AC, MQ and Link are not changed by this instruction. This instruction can be micro-programmed with the CAM instruction to produce a Double-Precision Deposit Zero instruction (DDZ, octal 7765).

Double-Precision Increment (DPIC)

Octal Code: 7573
Execution time: 1.8 μs.
Operation: The constant "one" is added to the double-precision number in the AC, MQ in 2's complement arithmetic; the result is left in the AC and MQ. The carry (or lack thereof) is propagated to the Link. This instruction requires the MQL and MQA bits be "1" to work as defined. It may be microprogrammed with the CLA bit to load the AC, MQ with the constant "1".

Double-Precision Complement (DCM)

Octal Code: 7575
Execution time: 1.8 μs.
Operation: The number in the AC and MQ is complemented and incremented to form the 2's complement of the original number; the result is left in the AC and MQ. The carry (or lack thereof) from the most significant bit is propagated to the Link. This instruction requires the MQL and MQA bits be "1" to work as defined.

Double-Precision Skip if Zero (DPSZ)

Octal: 7451
Execution time: 1.2 μs.
Operation: The 24-bit number in the AC, MQ is tested. If all bits are zero, the next instruction is skipped. If any bit is a one, the next instruction is executed. This instruction, when combined with the CAM instruction, is used to test mode by clearing the AC and MQ and then attempting the DPSZ instruction.

SUMMARY

The chart below (Figure 7-3) indicates the difference in operation of instructions found in both modes.

INSTRUCTION	MODE A	MODE B
MUY	The next location holds the multiplier.	The next location holds the address of the multiplier.
DVI	The next location holds the divisor.	The next location holds the address of the divisor.
SHL, LSR, ASR	The next location holds one less than the number of shifts. On Right Shifts, MQ11 is lost.	The next location holds the number of shifts. (A shift of zero places is legal). On Right Shifts, MQ11 is shifted into the GT flag.

Figure 7-3 Instruction Differences

Figure 7-4 summarizes cycle times and indicates the longest practical machine cycle. Note that the longest cycle time plus 0.3 μs. is the maximum time to enter a DMA cycle, provided the Break Device synchronizes at Int. Strobe time as recommended in Chapter 9. It is possible, by a small amount of programming, to reduce the longest cycle to 6.2 μs. This programming consists of pretesting the AC on a normalize, and limiting long shifts to 15 places. Note, for example, that

```
MQL          /AC MQ, O AC
LSR          /Mode B Shift, 6 places
6
```

is equivalent to an 18-bit logical right shift and has a longest cycle of 3.5 μs., rather than 7.1 μs. Also, the total execution time for a straight 18-bit shift is 8.3 μs., as opposed to 5.9 μs. for the above sequence.

	MODE A			MODE B			
	MEM CYCLES	INSTR TIME	LONGEST CYCLE	MEM CYCLES	INSTR TIME	LONGEST CYCLE	NOTES
SWAB	1	1.2μs	1.2μs	1	1.2	1.2	
SWBA	1	1.2	1.2	1	1.2	1.2	
SCL	2	2.6	1.4	Not Available			
ACS	Not Available			1	1.2	1.2	
MUY	2	7.4	6.2	3	8.6	6.2	
DVI	2	7.4	6.2	3	8.6	6.2	No overflow
NMI	1	1.5+.3N	8.1	1	1.5+.3N	8.1	
SHL	2	2.6+.3N	8.9*	2	2.9+.3N	9.2**	25-place shift
ASR	2	2.6+.3N	8.9*	2	2.9+.3N	9.2**	25-place shift
LSR	2	2.6+.3N	8.9*	2	2.9+.3N	9.2**	25-place shift
SCA	1	1.2	1.2	1	1.2	1.2	
DAD	Not Available			4	5.2	1.4	
DST	Not Available			4	5.2	1.4	
DPSZ	Not Available			1	1.2	1.2	
DPIC	Not Available			1	1.6	1.6	
DCM	Not Available			1	1.6	1.6	
SAM	Not Available			1	1.2	1.2	

*Computed from 1.4+.3N
**Computed from 1.7+.3N

Figure 7-4 EAE Mode A/Mode B Instruction Times

MEMORY EQUIPMENT OPTIONS

The basic memory (MM8-E) is a 4096-word, random-access core memory that performs all the functions of data storage and retrieval. The MM8-E is packaged on three PDP-8/E modules that plug into the OMNIBUS. These modules, when used with the KM8-E Memory Extension and Time Share option, can extend memory capacity up to 32,768 words in increments of 4096 words. In addition, this option enables the PDP-8/E to operate in a time-sharing environment. With the addition of the MP8-E Memory Parity option, all transfers to and from memory can be checked for parity. Other memory options include the MR8-EA 256-Word Read-Only Memory, the MR8-EB 1024-word Read-Only Memory, the MW8-E 256-word Read/Write Memory, and the MI8-E Bootstrap Loader.

KM8-E Memory Extension and Time-Share Option

This option provides the user with two primary capabilities. The memory extension portion extends the addressing capabilities of the machine from 4069 words up to 32,768 words. The time-share portion enables the computer to operate in either the normal manner (Executive Mode) or the User Mode. User Mode enables the machine to function in a time-sharing environment in which a user program is prevented from disturbing or interfering with another user program. The KM8-E option is packaged on one PDP-8/E module that plugs into the OMNIBUS. This option is required whenever memory capacity is extended beyond 4096 words.

Memory Extension Description

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

Instruction Field Register (IF)—The IF is a three-bit register that serves as an extension of the PC. The contents of the IF determine 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. Depressing the console EXTD ADDR LOAD switch transfers the instruction field in SWITCH REGISTER bits 6 through 8 into the IF register. During a JMP or JMS instruction, the IF is set by transfer of information from the instruction buffer register. When a program interrupt occurs, the contents of the IF are automatically stored in bits 0 through 2 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 three-bit register determines the memory field from which operands are taken in indirectly-addressed AND, TAD, ISZ, or DCA instructions. Depressing the console EXTD ADDR LOAD switch transfers the SWITCH REGISTER bits 9 through 11 into the DF register. During a CDF instruction, the DF register is loaded from MD6-8 to establish a new data field. When a program interrupt occurs, the contents of the DF are automatically stored in bits 3-5 of the save field register. The DF is set by a transfer of information from save field register bits 3 through 5 by the RMF instruction. This action is required to restore the data field at the conclusion of the program interrupt subroutine.

Instruction Buffer Register (IB)—The IB serves as a three-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's console switches. The IB is set by depressing of the console EXTD ADDR LOAD switch in the same manner as the instruction field register. A CIF microinstruction loads the IB with the programmed field on MD6-8. An RMF microinstruction transfers save field register bits 0 through 2 into the IB to restore the instruction field that existed before a program interrupt.

Save Field Register (SF)—When a program interrupt occurs, this seven-bit register is loaded from the user build flip-flop, and the IR and UF registers. The SF register is loaded 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 registers are cleared. An RMF instruction can be given immediately before exit from the program interrupt subroutine to restore the instruction field and data field by transferring the SF into the IB and the DF registers. (Also, see GTF and RTF instructions.)

Extended Address Gating—This logic consists of an output gating structure and control logic for gating the extended memory field address to core memory. The contents of the IF register are placed on the EMA0-2 lines unless an AND I, TAD I, ISZ I, or DCA I instruction is encountered. If such an instruction is encountered, the contents of the IF are placed on EMA 0-2 for the Fetch and Defer cycles, and the contents of the DF are placed on EMA0-2 for the Execute cycle. The extended memory field address is changed only at TP4 and remains available for the entire memory cycle.

Data 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 via DATA lines 6-11. During an RIB or GTF instruction, bits 6 through 11 of the AC receive contents of the save field register. During an RIF instruction, bits 6 through 8 of the AC receive the contents of the instruction field register. During an RDF instruction, bits 6 through 8 of the AC receive the contents of the data field register.

Device Selector and Instruction Decoding—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 2N. Bits 9 through 11 are also decoded to implement specific commands. The instruction decoding logic is common to the time-share portion of the KM8-E option.

Programming

Instructions associated with the extended memory portion KM8-E option are defined below:

Get Flags (GTF)

Octal Code: 6004
Operation: Reads the contents of the interrupt inhibit flip-flop, and the SF register to AC3, AC5-11 respectively. The other AC bits are loaded with information from the CPU and the EAE; i.e., link, greater-than-flag, interrupt bus, interrupt on.

Restore Flags (RTF)

Octal Code: 6005
Operation: Loads the user buffer flip-flop, the instruction buffer register, and the data field register with the contents of AC bits 5, 6-8, and 9-11 and inhibits processor interrupts until the next JMP or JMS instruction. At the conclusion of the JMP or JMS instruction, the contents of the user buffer flip-flop and the instruction buffer register are transferred into the user field flip-flop and the instruction field register, respectively. The contents of the other AC bits are loaded into the CPU and EAE to cause the converse of the GTF instruction. The Interrupt On flip-flop in the CPU is unconditionally set by this instruction.

Change to Data Field N (CDF)

Octal Code: 62N1
Operation: Loads the data field register with the program-selected field number (N = 0 to 7). All subsequent memory requests for operands are automatically switched to that data field, except for directly-addressed AND, TAD, ISZ, or DCA instructions.

Change to Instruction Field N (CIF)

Octal Code: 62N2
Operation: Loads the instruction buffer register with the program-selected field number (N = 0 to 7) and inhibits processor interrupts until the next JMP or JMS instruction. At the conclusion of either of these instructions, the contents of the instruction buffer register are transferred into the instruction field register.

Change Data Field, Change Instruction Field (CDF, CIF)

Octal Code: 62N3
Operation: Performs the combination of CDF and CIF operations.

Read Data Field (RDF)

Octal Code: 6214
Operation: ORs the contents of the data field register into bits 6-8 of the AC. All other bits of the AC are unaffected.

Read Instruction Field (RIF)

Octal Code: 6224
Operation: ORs the contents of the instruction field register into bits 6-8 of the AC. All other bits of the AC are unaffected.

Read Interrupt Buffer (RIB)

Octal Code: 6234
Operation: ORs the contents of the save field register (which is loaded from the instruction and data field during a program interrupt) into bits 6-8 and 9-11 of the AC, respectively. Thus, AC 6-11 contains the instruction and data fields that were in use before the last program interrupt. AC 5 is loaded by the time-share bit of the save field register. All other bits of the AC are unaffected.

Restore Memory Field (RMF)

Octal Code: 6244
Operation: Restores the contents of the save field register (which is loaded from the instruction and data field during a program interrupt) into the instruction buffer, the data field register, and the user buffer (if time share option is enabled). This command is used upon exit from the program interrupt subroutine in another field.

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 contents of the instruction field register, except for indirectly addressed operands, which are specified by the contents 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 instruction, as follows:

Indirect (Bit 3)	Page or Z Bit (Bit 4)	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 contents of field m located in page 0 designated by bits 5 through 11.

1	1	m	n	The absolute address of the operand in field n is taken from the contents of field m located in the current page, designated by bits 5 through 11.

Each field of extended memory contains eight auto-index registers in addresses 10 through 17. For example, assume that a program in field 2 is running (IF = 2) and using operands in field 1 (DF = 1) when the instruction TAD I 10 is fetched. The defer cycle is entered (bit 3 = 1), and the contents of location 10 in field 2 are read, incremented, and rewritten. If address 10 in field 2 originally contained 4321, it now contains 4322. In the execute cycle, the operand is fetched from location 4322 of field 1. Program control is transferred between memory fields by the CIF instruction. The instruction does not change the instruction field directly, as this would make it impossible to execute the next sequential instruction; instead, it loads the new instruction field in 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, because the IF is an extension on the most significant end of the PC, the 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.

NOTE

The IF is not incremented if the PC goes from 7777 to 0000. This feature protects the user from accidentally entering a nonexistent field.

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 2
/INSTRUCTION FIELD = 2; DATA FIELD = 2
/CALL A SUBROUTINE IN MEMORY FIELD 1
/INDICATE CALLING FIELD LOCATION BY THE CONTENTS OF THE DATA
 FIELD

          CIF 10          /CHANGE TO INSTRUCTION
                          /FIELD 1 = 6212
          JMS I SUBRP     /SUBRP = ENTRY ADDRESS
          CDF 20          /RESTORE DATA FIELD
            .
            .
            .
SUBRP,    SUBR            /POINTER
                          /CALLED SUBROUTINE, LOCATED IN
                          /FIELD 1
```

```
SUBR,   0               /RETURN ADDRESS STORED HERE
        CLA
        RDF             /READ DATA FIELD INTO AC
        TAD RETURN      /CONTENTS OF THE AC = 6202 +
                        /DATA FIELD BITS
        DCA EXIT        /STORE INSTRUCTION SUBROUTINE
          .             /NOW CHANGE DATA FIELD IF DESIRED
          .
          .
          .
EXIT,                   /A CIF INSTRUCTION
        JMP I SUBR      /RETURN TO CALLING PROGRAM
RETURN, CIF             /USED TO CALCULATE EXIT
                        /INSTRUCTION
```

When a program interrupt occurs, the current instruction and data field numbers are automatically stored in the 6-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 contents of the SF. Alternatively, the GTF and RTF instructions may be used to handle the Save Field and Link information. 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 CONTENTS OF
                        /LOCATION 0 AND LOAD IF FROM IB
OR
0,      0               /PC STORAGE
        DCA ACSV        /SAVE AC,
        MQA CLA
        DCA MQSV        /MQ,
        GTF
        DCA FLAGS
          .
          .
          .
        CLA
        TAD MQSV
```

```
MQL             /RESTORE MQ
TAD FLAGS
RTF             /REPLACE FLAGS, ION
CLA
TAD ACSV        /AC
JMP I 0         /AND EXIT
```

Time-Share Description

The Time-Share portion of the KM8-E operates in two modes as denoted by the user flag (UF) flip-flop. When the UF flip-flop is in the logic 1 state, operation is in the user mode and a user program is running in the central processor. When the UF flip-flop is in the logic 0 state, operation is in the executive mode and the time-sharing system's monitor is in control of the central processor. The four instructions (CINT, SINT, CUF, and SUF) are used by the time-sharing system's monitor in the executive mode and are never used by a user program. If a user program attempted to use one of these instructions, execution of the instruction would be blocked (see next paragraph). The KM8-E option adds the necessary hardware to the PDP-8/E to implement these instructions.

In executive mode, the computer operates normally. When the computer is operated in user mode, operation is normal except for IOT, HLT, LAS, and OSR instructions. When one of these instructions is encountered, the hardware inhibits the normal instruction sequence (other than rewriting the instruction in memory), and generates an interrupt at the end of the current memory cycle by setting the UINT flip-flop. The time-sharing system's monitor program then analyzes the source of interrupt, and takes appropriate action.

The time-share option requires at least 8K of core memory; thus, it is packaged with the memory extension option. A jumper on the KM8-E module is used to select the time-share function. The module is shipped with this jumper in place (time-share function disabled).

Programming

Instructions associated with the time-share portion of the KM8-E are defined as follows:

Clear User Interrupt (CINT)

Octal Code: 6204
Operation: Clears the user interrupt flip-flop.

Skip on User Interrupt (SINT)

Octal Code: 6254
Operation: Increments the PC when the user interrupt flip-flop is set *so the next sequential instruction is skipped.*

Clear User Flag (CUF)

Octal Code: 6264
Operation: Clears the user buffer flip-flop.

NOTE

If the machine is stopped while in user mode, the user flag (UF) is cleared by operating the extended address load key (EXT ADDR LOAD).

Octal Code: 6274

Operation: Sets user buffer flip-flop and inhibits processor interrupts until the next JMP or JMS instruction. At the conclusion of either of these instructions, the content of the user buffer flip-flop is transferred into the user field flip-flop.

MP8-E Memory Parity

The memory parity option adds the circuits required to generate, store, and check the parity of memory words. This option replaces the 12-bit memory system with, effectively, a 13-bit system by adding the generating and storage capabilities for the parity bit. Odd parity (odd number of binary ones in the 13-bit word) is generated and stored for each word entered into memory. Parity is formed for each word retrieved from memory and this result is checked against its stored parity bit. If the two differ, a parity error flag is set to indicate that an error occurred. This flag is normally connected to the program interrupt system to cause the computer to enter a program interrupt subroutine for locating the interrupt source. Once the interrupting source is located, the computer enters an appropriate service routine to service the error condition. This routine can repeat the program step in which the error occurred to verify the error condition, can perform a simple read/write check for the error's address, or can determine machine status for the error detected and re-establish or print out these conditions, and then halt. The routine can also return the machine to the main program.

The MP8-E option consists of three PDP-8/E modules that plug into the OMNIBUS. Two of these modules (X-Y Driver and Current Source, and Core Stack) are identical to those of the MM8-E basic core memory and use the same addressing methods. However, only eight bits of the possible 12 bits are used. These eight-bit locations correspond to the eight possible memory fields and store up to 32,768 (8 $\times$ 4096) parity bits. The third module (Sense-Inhibit) contains device and operation decoding circuits, field decoding circuits, eight sense amplifiers, an eight-bit register, eight inhibit drivers and circuits for controlling the operations. This module also contains three control and status flip-flops that are controlled by IOT instructions. These flip-flops select odd or even parity generation and checking, enable or disable interrupts for parity errors, and store a parity error condition.

The following routine initializes the parity bits for a read-only or write-protected memory:

```
                        /INITIALIZE LOC 10 WITH STARTING ADD.
                        /TURN OFF PARITY INTERRUPT
                        /SET COUNTER
LOOP,   TAD I 10        /READ DATA, REWRITE PARITY
        ISZ COUNT
        JMP LOOP        /CONTINUE UNTIL DONE
                        /CLEAR PARITY ERROR FLAG
                        /TURN ON PARITY INTERRUPT
```

Programming
Instructions associated with the MP8-E option are:

Disable Memory Parity Error Interrupt (DPI)

Octal Code: 6100
Operation: Disables the generation of interrupts for parity errors by clearing the interrupt enable flip-flop of the memory parity option.

Skip On No Memory Parity Error (SMP)

Octal Code: 6101
Operation: Senses the memory parity error flag; if it contains a 0 (signifying no error has been detected), the PC is incremented so that the next instruction is skipped.

Enable Memory Parity Error Interrupt (EPI)

Octal Code: 6103
Operation: Enables interrupts from the memory parity option. The memory parity interrupt is automatically enabled when power is turned on, by the CLEAR key on the front panel and by the CAF IOT instruction.

Clear Memory Parity Error Flag (CMP)

Octal Code: 6104
Operation: Clears the memory parity error flag. The parity error flag is also cleared when power is turned on, by the CLEAR key on the front panel, and by the CAF IOT instruction.

Skip on No Memory Parity Error and Clear Memory Parity Error Flag (SMP, CMP)

Octal Code: 6105
Operation: Senses the memory parity error flag; if it contains a 0, the next instruction is skipped. The memory parity error flag is then cleared.

Check For Even Parity (CEP)

Octal Code: 6106
Operation: Causes parity to be checked for an even number of binary 1's in the entire word. This operation is effective only during the execute cycle immediately following this instructon.

Skip on Memory Parity Option (SPO)

Octal Code: 6107
Operation: Increments the PC when the system includes a memory parity option so that the next sequential instruction is skipped.

Use of these instructions is discussed below:

a. The DPI instruction is useful in certain diagnostic maintenance programs where it is desired to disable interrupts resulting from parity errors. This instruction also gives the user more flexibility for multiple program interrupt usage.
b. The SMP instruction is used as a programmed check for memory parity errors. When used in a program interrupt subroutine, this instruction can be followed by a jump to a portion of the routine that services the memory parity option.
c. The EPI instruction is used to return the memory parity option to normal operation after a DPI command.
d. The CMP instruction initializes the memory parity option in preparation for normal programmed operation of the computer.
e. The CEP instruction is useful in diagnostic maintenance programs. By altering the parity check from odd to even, parity errors can be forced, to permit checking for proper functions of the parity option.
f. The SPO instruction permits the user to automatically check whether or not the system is equipped with a memory parity option.
g. The SMP, CMP instruction is a combination of SMP and CMP instructions, and permits the operations performed by these instructions to be implemented by one instruction.

MW8-E 256-Word Read/Write Memory

This option is a 256-word read/write memory with a write-protect feature. The MW8-E option can be configured with the basic core and is required whenever a MR8-E ROM is used. When used with the basic 4096 core memory, a KM8-E Memory Extension and Time-Share option is also required.

The MW8-E option has four primary uses:

a. It is used with ROM for variable storage and interrupt handling.
b. It is used to simulate a ROM (through write-protect feature) for program debugging and short-term storage.
c. It is used as a general-purpose ROM such as might be needed for program constants.
d. It can be used with the time-sharing feature of KM8-E to protect a user monitor program.

The MW8-E option is packaged on two PDP-8/E modules that plug into the OMNIBUS. One module contains X-Y drive circuits and the core memory. The second module contains the sense-inhibit and input/output gating circuits.

The write-protect feature is selected using a switch mounted on the sense-inhibit module. When placed to ROM, this switch prevents writing of input data into the storage. When placed to NORM, both read and write operations can be performed.

An MW8-E can be assigned any memory field address; however, it must be assigned a block of 256 addresses beginning with an even-number

page. Field and page addresses are selected using jumpers in its address decoding circuits. When used in other than field 0, the KM8-E option is also required.

MR8-EA 256-Word Read-Only-Memory
The MR8-EA option provides the user with read-only-memory (ROM) capabilities such as might be used for hardwired controller, communications or process-control functions. This option is provided in 256-word increments package on one module. However, the module, because of its thickness, requires two module slots.

Information stored by the ROM is established by wiring of the unit at the factory. Therefore, the information content must be specified by the user at the time of purchase. A recommended approach to defining the ROM content is to use the MW8-E 256-Word Read/Write option to simulate the ROM (through use of the MW8-E write-protect feature) for program definition and debugging. A copy of the resulting program can then be supplied to define the content of the ROM.

The number of ROM modules used is limited only by the amount of basic core or read/write capability required and the maximum address capabilities of the machine. A ROM can be assigned any memory field address, however, it must be assigned a block of 256 addresses beginning with an even-number memory page. Field and page addresses are selected by jumpers on its address decoding circuits. When used in other than field 0, the KM8-E option is required.

In situations where a small amount of ROM is desired, an MR8-E can be installed which uses locations already allotted to the 4K memory. The MR8-E automatically disables core memory using the same address. The core addresses can be re-enabled by removal of the MR8-E.

MR8-EB 1024-word Read-Only Memory
(same as MR8-EA, except larger)

MI8-E Bootstrap Loader
This option uses a 32-word read-only-memory (ROM) with diodes that can be arranged to accommodate any program up to 32 words in length. This option is normally used as a hardware Read-In-Mode (RIM) paper tape loader for loading of programs from the PDP-8/E paper tape reader of the console teleprinter. However, it can be used for any user-designated programs of 32 words or less. The MI8-E option is contained on one PDP-8/E module that plugs into the OMNIBUS.

The MI8-E operates in a shadow address mode with core memory. That is, the addresses used for this device can overlap core memory addresses and can be used by core memory whenever the MI8-E option is not operating. The MI8-E can be used in any memory field; the field is selected by jumpers on the module. For a 32-word program, the MI8-E occupies the last 32 locations in the field (7740 (octal) through 7777 (octal)). The starting and ending addresses within this 32-address group are selected by jumpers on the module. Thus, programs requiring less than 32 locations can also be readily implemented.

The MI8-E option is selected, using the console SW control. However, this control has no effect unless the machine is stopped (RUN flop is reset). When this control is depressed, addresses 7740 (octal) through 7777 (octal) access the MI8-E hardware only. Core memory is prevented from responding to these addresses by outputs of the MI8-E control logic.

To operate the M18-E option, the SW key is depressed, loading the starting address and starting the computer. The MI8-E then assumes control and provides instructions from its ROM to the MD lines during each FETCH major state. These instructions can load paper tape programs from the PDP-8/E paper tape reader or the console teleprinter, or perform user-designated functions. When the ending address is reached (as determined by module jumpers and MA inputs), the last instruction is executed and the Bootstrap Loader resets itself.

REAL TIME CLOCK OPTIONS

Type DK8-EA Real Time Clock (Line Frequency)

The DK8-EA is a fixed-interval line frequency clock option to the PDP-8/E that causes an interrupt 100 or 120 times per second, depending on line frequency. The clock and control are contained on one PDP-8/E module, which plugs into the OMNIBUS.

Programming

The following instructions control the DK8-EA line frequency clock:

Enable Interrupt (CLEI)

Octal Code: 6131
Operation: Enables the clock interrupt so that each clock pulse will cause a program interrupt request.

Disable Clock Interrupt (CLDI)

Octal Code: 6131
Operation: Disables the clock interrupt so that the clock cannot cause program interrupts.

Skip on Clock Flag and Clear Flag (CLSK)

Octal Code: 6133
Operation: Senses the clock flag, which is set with each clock pulse; if it is set, the next sequential instruction is skipped, and the clock flag is cleared.

Type DK8-EC Real Time Clock (Crystal)

The DK8-EC is a fixed-interval crystal-controlled clock option to the PDP-8/E that is used to cause an interrupt every 50, 500, or 5,000 times per second (jumper selectable). The clock frequency is derived from a 20-MHz crystal. The clock and control are contained on one PDP-8/E module, which plugs into the Omnibus.

Programming

The instructions which control the DK8-EC crystal clock are the same as those shown above for the DK-EC line frequency clock.

Type DK8-EP Programmable Real Time Clock

The DK8-EP real time clock option offers the PDP-8/E user a method for accurately measuring and counting intervals or events in a number of ways.

The DK8-EP system consists of:

a. A 12-bit binary counter using MSI integrated circuits with an overflow bit.
b. A 12-bit buffer register.
c. A 20-MHz crystal clock with frequency dividers.
d. A PDP-8/E module (M860) containing all control functions, IOT decoding, and registers.

Logically, the DK8-EP contains the following features:

a. Clock Enable Register
   This register controls the rate of the time base and the mode of counting, and selectively enables each of the three input channels and the interrupt line.
b. Clock Buffer
   The Clock Buffer stores data being transferred from the AC to the clock counter, or from the clock counter to the AC. It also permits presetting of the clock counter.
c. Clock Counter
   This register is a 12-bit binary counter that may load the clock buffer or to be loaded from it. When an overflow occurs and the clock enable mode is 01, the clock buffer is automatically loaded into the clock counter. The overflow is set by the most significant bit of the clock enable register going from 1 to 0.
d. Programmable Time Base
   The Programmable Time Base provides count pulses to the clock counter according to the rate set by the clock enable register.
e. Crystal Clock
   The clock is a simple crystal-controlled clock, which operates at 20 MHz + or − 0.1%. MSI integrated circuit decade counters divide the base clock frequency down to any of the following rates: 1 MHz, 100 kHz, 10 kHz, 1 kHz, or 100 Hz.

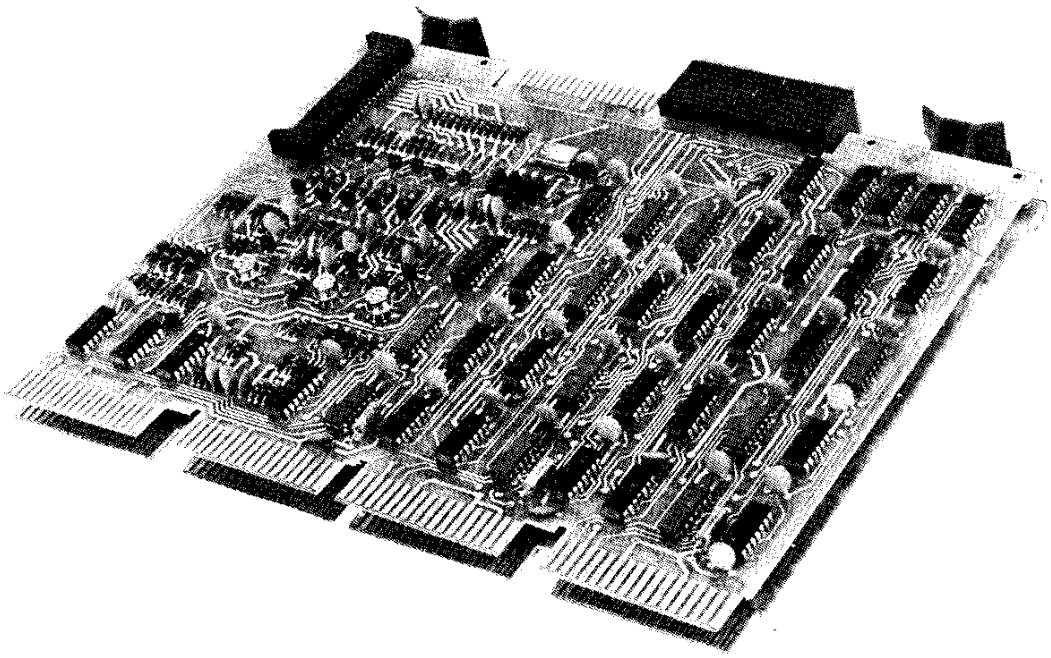
Programming

The following IOT instructions control the DK8-EP real time clock:

Skip on Clock Interrupt (CLSK)

Octal Code: 6131
Operation: Causes the content of the PC to be incremented by one if an interrupt condition exists, so that the next instruction is skipped. The interrupt conditions are as follows:

*a. Enable Event Interrupt 1 and Input 1
*b. Enable Event Interrupt 2 and Input 2
*c. Enable Event Interrupt 3 and Input 3
*d. Enable Overflow Interrupt and Overflow



DK8-EP Real Time Clock

AC to Clock Buffer (CLAB)

Octal Code: 6133
Operation: Causes the content of the AC to be transferred into the Clock Buffer; then causes the content of the Clock Buffer to be transferred into the Clock Counter. The AC is not changed.

Clear Clock Enable Register per AC (CLZE)

Octal Code: 6130
Operation: Clears the bits in the Clock Enable Register corresponding to those bits set in the AC. The AC is not changed.

Set Clock Enable Register per AC (CLDE)

Octal Code: 6132
Operation: Sets the bits in the Clock Enable Register corresponding to those bits set in the AC. The AC is not changed.

Load Clock Enable Register (CLEN)

Octal Code: 6134
Operation: Causes the content of the Clock Enable Register to be transferred into the AC.

Clock Enable Registers Functions

AC BIT	FUNCTION
0	Enables clock overflow to cause an interrupt.
1 & 2	Mode
	00 Counter runs at selected rate. Overflow occurs every 4096 counts. Flag remains set.
	01 Counter runs at selected rate. Overflow causes Clock Buffer to be transferred to the Clock Counter, which continues to run. Overflow remains set until cleared with IOT 6135.
	*10 Counter runs at selected rate. When an enabled event occurs, the Clock Counter is transferred to the Clock Buffer, and the Counter continues.
	*11 Counter runs at selected rate. When an enabled input occurs on any channel three, the Clock Counter is transferred to the Clock Buffer, and the Clock Counter continues to run from zero.

Rate Selection

Contents of Bits 3-5	Octal Value	Interval Between Pulses	Frequency
000	0	Stop	0
*001	1	-	External Input
010	2	10^{-2} sec	100 Hz
011	3	10^{-3} sec	1 KHz
100	4	10^{-4} sec	10 KHz
101	5	10^{-5} sec	100 KHz
110	6	10^{-6} sec	1 MHz
111	7	Stop	0

* Available only as a LAB-8/E Option

6	Overflow starts ADC. (When the Clock Counter overflows, the analog-to-digital converter, type AD8-EA, is started.)
7	When set to 1, inhibits clock.
*8	Events in Channels 1, 2, or 3 cause an interrupt request and overflow.
*9, 10, & 11	Enable Events 1, 2, and 3 9 —Event 3 10—Event 2 11—Event 1

Clock Status to AC (CLSA)

Octal Code: 6135

Operation: Interrogates the Clock Input and Overflow Status flip-flops. The clock stauts information is inclusively ORed into the AC, then the status bits corresponding to set AC bits are cleared. This ensures that only one occurrence of an Event will be transferred to the program. The status condition is established as follows:

AC Bit	Status Condition
0	Overflow
* 9	Event 3
*10	Event 2
*11	Event 1

Clock Buffer to AC (CLBA)

Octal Code: 6136

Operation: Clears the AC, then transfers the content of the Clock Buffer into the AC.

Clock Counter to AC (CLCA)

Octal Code: 6137

Operation: Clears the AC, transfers the content of the Clock Counter to the Clock Buffer, then transfers the content of the Clock Buffer into the AC.

NOTE

The clock counter may be read while it is counting. Gating in the clock control section prevents data from being strobed out of the counter before a specified time following a clock pulse. This time, approximately 300 ns, allows the data to settle in the counter.

This feature allows the counter to be read any number of times without introducing timing errors in counting the amount of time between intervals, and also eliminates false counts that are the result of reading the counter as one or more bits are in transition from one state to another.

* Available only on LAB-8/E Option

Example Subroutine #1

This example illustrates how the DK8-EP can be used as a double-precision (24-bit) free-running clock, using the clock counter as the low order 12 bits and a memory location as the high order 12 bits. Because all of the clock's registers have been set to zero initially by the clear key, the program needs only to zero the high order words, set the enable register, and turn on the interrupt. After 4096 counts, the clock counter overflows, signalling an interupt. The service routine simply increments the high order word, then returns to the main program.

```
        CLA
        DCA HIGH        /ZERO HIGH ORDER WORD
        TAD ENABLE      /OVER + MODE 00 + RATE
        CLOE            /SET ENABLE REGISTER
        ION             /INTERRUPT ON

ENABLE = OVERFL + MODE 00 + RATE

                        /SERVICE ROUTINE
        CLSK            /CLOCK SKIP?
        JMP OTHERS      /NOT A CLOCK FLAG
        CLSA            /READ STATUS, CLEAR FLAGS
        SPA CLA         /IGNORE OTHER CLOCK INTERRUPTS
        ISZ HIGH        /INCREMENT HIGH
        JMP RETURN      /RETURN TO MAIN PROGRAM
```

With this simple program, time can be kept during program execution. With the clock set to its fastest rate (1 μs per tick), this double-precision counter could mark time for only just over 16 seconds; with the clock set to its slowest rate, it could mark time for over 100 days.

A simple routine could be written to interrogate elapsed time by using the CLCA (clock counter to AC) command.

Example Subroutine #2

The DK8-EP can also easily be programmed to function as an alarm clock, counting off a period of time, giving an alarm, automatically resetting itself, and continuing. The alarm could be used to ring a bell, as indicated in the example; however, a more practical use would be to start an analog-to-digital conversion to take a number of samples from the outside world.

This example will ring the bell every second:

```
START,  CLA
        TAD COUNTER         /SET COUNTER TO -1000
        CLAB
        CLA
        TAD ENABLE          /SET ENABLE REGISTER
        CLOE
AGAIN,  CLSK                /CLOCK SKIP?
        JMP .-1
        CLSA                /YES, READ STATUS
        CLA
        TAD BELL            /RING BELL
```

```
            TLS
            TSF
            JMP .-1
            JMP AGAIN
COUNTER,    -1750
ENABLE,     MODE 01 + 1 MS
BELL,       207
```

This program could easily be modified to work in the interrupt mode by setting bit 0 of the enable register to a 1. An interrupt would then occur every second, and this could be used to ring the bell.

Type KP8-E Power Fail Detect

The KP8-E and its related shut-down and restart subroutines are designed to restore computer operation automatically following a failure of the computer's primary power source. This OMNIBUS option protects an operating program in the event of such a failure by causing a program interrupt, enabling continued operation for 1 ms; this allows the interrupt routine to detect the low power as initiator of the interrupt and to store both the contents of active registers (AC, L, MQ, etc.) and the program count in known core memory locations.

Variations of the AC line below the predetermined threshold level at a rate of one per second or less will also cause the shut-down circuits to be activated. When power is restored the power low flag clears, and a routine beginning in address 0000 starts automatically. This routine restores the contents of the active registers and program counter to the conditions that existed when the interrupt occurred, then continues the interrupted program.

The power failure option consists of three circuits, contained on a single PDP-8/E module.

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

b. A shutdown sequence circuit, which ensures that, when a power interrupt occurs, the computer logic circuits will continue operation for 1 ms to allow a program subroutine to store the contents of the active registers. If, at the end of the 1 ms interval, computer operation still continues, it is halted. When power conditions are suitable for computer operation, a restart circuit clears the power low flag and restarts the program. A manual RESTART switch located on the right side of the power fall module enables or disables the automatic restart operation. With this switch in the ON (up) position, the option clears the MA and produces a MEMORY START pulse 500 ms after power conditions are satisfactory. The MA is cleared so that operation restarts by executing the instruction in address 0000. That instruction must be a JMP to the starting address of the subroutine that restores the contents of the active registers and the program counter to the conditions existing prior to the power low interrupt. The 500-ms delay ensures that slow mechanical devices, such as Teletype equipment, have completed any previous operation 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 (down) 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. The shut-down circuitry is unaffected by the switch.

c. A skip circuit provides programmed sensing of the condition of the power low flag by adding the IOT SPL (6102) instruction to the computer repertoire.

Programming
Skip On Power Low (SPL)

Octal Code: *6102*
Operation: Senses the content of the power low flag. If the power low flag contains a 1 (indicating that a power failure has been detected), the contents of the PC are incremented by one, so that the next sequential instruction is skipped.

Because the time that computer operation can be extended after a power failure is limited to 1 ms, the condition of the power low flag should be the first status check made by the program interrupt subroutine. The interrupt subroutine, starting with the SPL microinstruction (and including the power fail program sequence), can be executed in less than 30 μs. The power fail program sequence stores the contents of the active registers and program counter in designated core memory locations, then relocates the calling instruction of the power restore subroutine to address 0000, as follows:

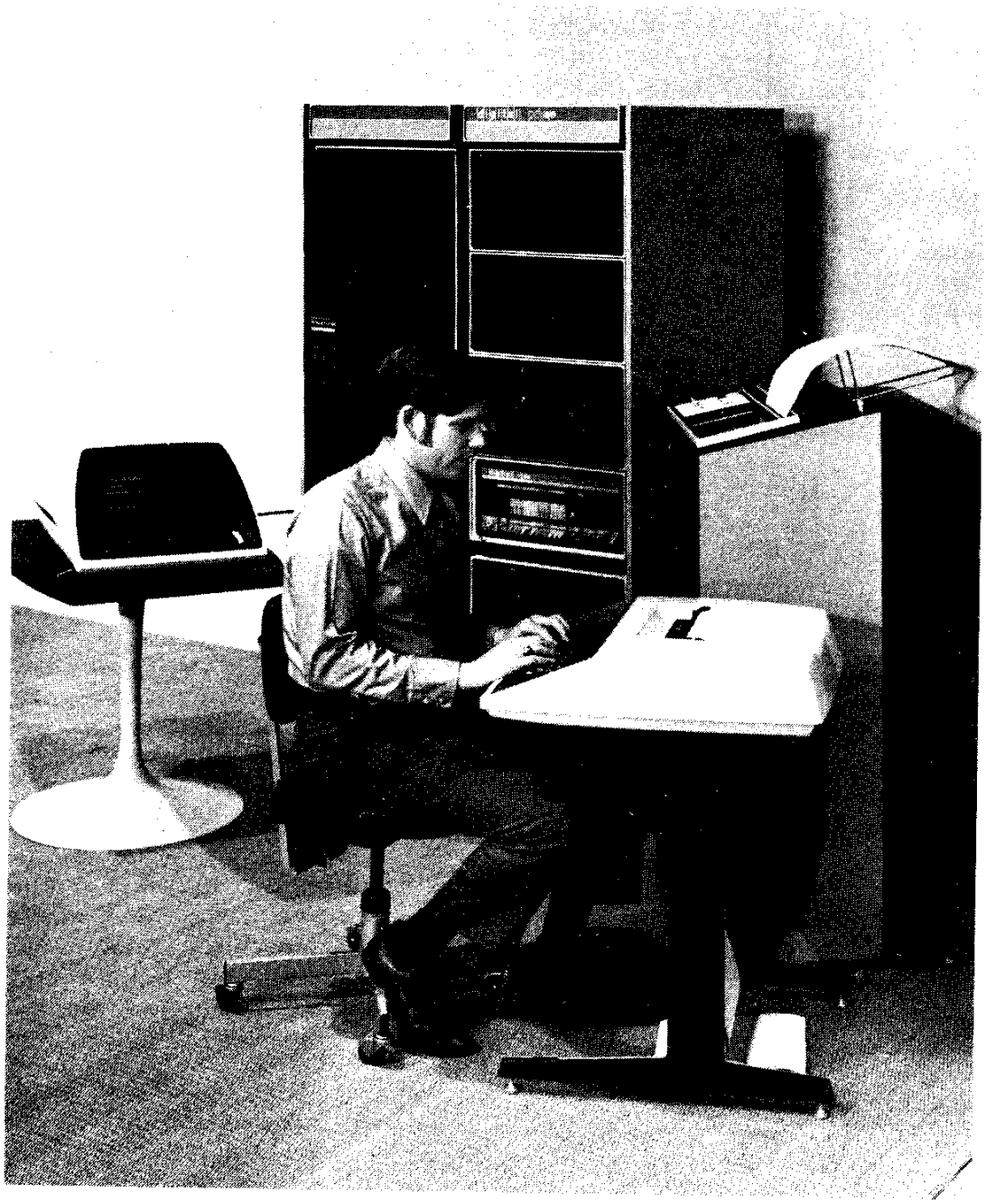
```
Address     Instruction     Remarks
0000        —               /STORAGE FOR PC AFTER PROGRAM INTERRUPT
0001        JMP FLAGS       /INSTRUCTION EXECUTED AFTER PROGRAM INTERRUPT

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

            MQA             /GET MQ
            DCA MQ          /SAVE MQ
            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 LOCAION OF RESTART ROUTINE
```

Automatic program restart begins by executing the instruction stored in address 0000 by the power fail routine. The power restore subroutine restores the contents 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 MQ       /GET MQ
         MQL          /RESTORE MQ
         TAD LINK     /GET LINK
         CLL RAL      /RESTORE LINK
         TAD AC       /RESTORE AC
         ION          /TURN ON INTERRUPT
         JMP I PC     /RETURN TO INTERRUPTED PROGRAM
```



Computers are considered a basic tool for the scientist and mathematician. DEC's PDP-8/E system is especially appealing because it offers a low cost processing and problem solving capability that might be expected on larger and more expensive computers.

SECTION 3 OMNIBUS INPUT/OUTPUT EQUIPMENT OPTIONS

This section describes those options to the PDP-8/E computer that perform transfers of information to and from the computer by means of the OMNIBUS. In all cases, execution time for IOT instructions in this section is 1.2 μs.

Many of these options and their manner of transfer are discussed in greater detail in Chapter 5 and 9 of this handbook.

CONSOLE TELEPRINTERS

DECwriter

The PDP-8/E DECwriter option comprises the LA30 DECwriter and LC8-E Control.

LC8-E DECwriter Control

The LC8-E DECwriter Control is an interface between the PDP-8/E processor and the parallel version of the LA30 DECwriter. The LC8-E Control is one PDP-8/E module which plugs into the OMNIBUS.

The device codes for the keyboard and printer are selectable by means of wired jumpers on the module, allowing several LC8-E controls to be used by the same processor.

Connection to the LA30 is made by a standard 25 foot cable which plugs directly into the LC8-E module.

In operation, the LA30 is considered as two devices, a keyboard and a printer. Therefore two device codes are assigned. If the LC8-E is used to replace the KL8-E console Teletype control, these device codes would be codes 03 for the keyboard and 04 for the printer. Other pairs of device codes can be assigned according to the normal sequence for additional Teletype controllers. The instruction list given assumes that device codes 03 and 04 have been selected. The control unit contains a programmable interrupt enable flip-flop which controls the generation of program interrupt requests from both the keyboard and printer. This flip-flop is set when power is turned on or when INITIALIZE is generated. It can also be set or cleared under program control (as specified by AC11) by the KIE instruction.

Specifications

Type of transmission	Parallel TTL levels
Type of reception	Parallel TTL levels
Number of data elements per character	Seven
Maximum input/output rate	30 characters per second*

* See LA30 Specification

Keyboard

When a key is depressed on the LA30 keyboard, the seven bit ASCII representation of the character is established on the seven data input lines to the LC8-E control. Also generated is the signal Transmitter Stroke to transfer this character into the LC8-E input buffer and to set the keyboard (receiver) flag. This causes a program interrupt request if the interrupt enable flip-flop is set and can be tested by a skip IOT whether

DEC's new LA30 DECwriter is a dot matrix impact printer that operates at a speed of 30 characters per second, three times the speed of commonly used teleprinters. Its quiet operation and high reliability are the result of the systematic elimination of mechanical parts, substituting, where possible, solid state logic modules.

the interrupt is enabled or not enabled. A READ IOT transfers the buffer contents to AC5-11, sets AC4 and clears the keyboard (receiver) flag. Setting AC4 is to make the input character compatible with the Teletype, where the most significant bit is always set on keyboard input.

PROGRAMMING

The following instructions assume device code 03 and that the LC8-E replaces the KL8-E. For any other device codes and in use as an additional keyboard, other mnemonics should be assigned.

Clear Keyboard Flag (KCF)

Octal Code: 6030
Operation: Clears the keyboard Flag.

Skip on Keyboard Flag (KSF)

Octal Code: 6031
Operation: Senses the keyboard flag and increments the PC if it is set, thereby skipping the next sequential instruction.

Read Keyboard Buffer Static (KRS)

Octal Code: 6034
Operation: Inclusively OR's the contents of the LC8-E input buffer with the AC and leaves the result in the AC Register.

Set/Clear Interrupt Enable (KIE)

Octal Code: 6035
Operation: Sets or clears the interrupt enable flip-flop as defined by AC11. Set if AC11(1); clear if AC11(0).

Read Keyboard Buffer Dynamic (KRB)

Octal Code: 6036
Operation: Performs the combined operations of KCC & KRS instructions. Clears the AC and the Keyboard Flag; loads AC5-11 from the LC8-E input buffer; sets AC4.

Printer

An IOT instruction is used to load the LC8-E printer buffer from AC5-11 and clear the printer flag. The information in the buffer is transferred to the LA30 DECwriter on the seven data output lines from the LC8-E; when they have settled, the control line receive strobe is asserted to initiate a print operation. When the LA30 DECwriter has completed the print operation, it indicates that it is again ready to print by setting the printer flag in the LC8-E. This causes a Program Interrupt Request if Interrupt Enable is set; the flag can be tested by a SKIP IOT whether Interrupt is enabled or not enabled.

PROGRAMMING

As with the Keyboard, it is assumed that the LC8-E replaces the KL8-E Console Teletype Control. The following instructions apply to the printer operation:

Set Printer Flag (TFL)

Octal Code: 6040
Operation: Sets the Printer Flag.

Skip on Printer Flag (TSF)

Octal Code: 6041
Operation: Senses the printer flag and increments the PC if it is set thereby skipping the next sequential instruction.

Clear Printer Flag (TCF)
Octal Code: 6042
Operation: Clears the Printer Flag.

Load Printer Buffer and Print (TPC)
Octal Code: 6044
Operation: Transfers AC5-11 to the LC8-E Printer Buffer and at TS1 of the next instruction asserts Receive Strobe to cause the character held in the buffer to be printed.

Skip on Printer or Keyboard Interrupt (TSK)
Octal Code: 6045
Operation: If either the printer flag or keyboard flag is set and the interrupt Enable flip-flop is set, increments the PC thereby skipping the next sequential instruction.

Load Printer Sequence (TLS)
Octal Code: 6046
Operation: This instruction combines TCF and TPC. It clears the printer flag and transfers the contents of AC5-11 to the LC8-E printer buffer. At TS1 of the following instruction, it asserts Receive Strobe to cause a character held in the buffer to be printed.

LA30 Differences from Teletype
From the above instruction lists it can be seen that the LC8-E is very similar to the KL8-E Console Teleprinter Control. There are differences mostly caused by the different characteristics of the LA30. These differences are summarized in the following:

1. There is no paper tape reader; hense no reader control,
2. There is no paper tape punch,
3. The maximum input/output rate is 30 characters/second,
4. Output to the printer section of the LA30 is only 7 bits (AC5-11), 8 bits can be sent but AC4 is ignored,
5. If a non-printing character is sent to the LA30, it does not go through a normal print cycle but indicates that it is ready to print again in approximately 1 to 2μ seconds.
6. The LA30 has no hardware TAB, FORM FEED or VERTICAL TAB feature,
7. Carriage return takes several character times but the Printer Flag is set approx. 2μs after a CAR RET is sent. The Printer Flag is not set again until the CAR RET has finished and the next character has been printed,
8. It is possible, by changing the internal switch on the LA30, for the keyboard to generate lower case characters. Normally this is set so that upper case is generated whether the keyboard is in SHIFT or not. The Printer cannot print lower case; it interprets lower case codes as upper case,
9. There is no BELL, CNTRL G is treated as non-printing,
10. End of line ($>$ 80 characters) is trapped and any subsequent characters sent before a CAR RET are not printed.

LA30 DECwriter

DEC's new LA30 DECwriter is a dot matrix impact printer that operates at a speed of 30 characters per second, three times the speed of commonly used Teletypewriters. Its quiet operation and high reliability are the result of the systematic elimination of mechanical parts, substituting, where possible, solid state logic modules. This reduction in moving parts means, for instance, that when the DECwriter is idle no parts are moving; conventional Teletypewriters with their extensive mechanical linkages can wear out even while not being used.

In order to print a character on the DECwriter, a- 7-dot matrix is moved along the 9 7/8" wide page by a stepping motor. Seven spring-loaded wires, driven by solenoids, are arranged vertically in the printing head. Characters are created while a solid state logic controlled motor advances the head along the line.

The DECwriter is a full-scale hard copy I/O terminal that uses a dot matrix to generate a character on ordinary paper; most others require special thermal or electrostatic paper. The DECwriter also uses the same widely available fan-folded paper as 80-column line printers, which allows a user to reduce costs by standardizing size and opening second sources for his paper supply. The terminal uses a standard, ½ in. wide, 40-yard nylon ribbon.

DECwriter

Specifications

Printer	
Printing Speed:	30 characters per second asynchronous; 250 ms carriage return (max.)
Line Length:	80 character positions
Character Spacing:	10 characters per inch
Line Spacing:	6 lines per inch
Paper:	9-7/8"-wide tractor-driven continuous form original plus one copy
Typeface:	5 x 7 dot matrix
Ribbon:	½-inch x 120 feet, nylon
Data Entry	
Code:	USASCII-1968 96 characters (128 optional)
Interface:	LC8-E
Environmental/Physical	
Temperature:	50° F-100° F
Humidity:	5-90% (noncondensing)
Power:	Type LA30-PA: 115VAC, 60 Hz Type LA30-PD: 230VAC, 50 Hz
Dimensions:	20-½ inches wide x 31 inches high x 24 inches deep

LT33-CC **(KSR33)** Keyboard Send and Receive only Friction Feed. Interface control is provided using KL8-E.

LT33-DC **(ASR 33)** Synchronous Read and Punch Friction Feed. Interface control is provided using KL8-E.

Model ASR 33 Teletype

The standard Teletype Model ASR 33 (automatic send-receive) is 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 tape at ten characters per second. Signals transferred between the Model ASR 33 and the control logic are standard, serial, 11-unit code, Teletype signals. The signals consist of spaces and marks which correspond to open circuit and bias current in the Teletype, and to Os and 1s in the Teletype control and computer. The start bit (space, 0, open circuit) and subsequent eight-character bits are one-unit-of-time duration and are followed by the stop bit, which occupies two units.

The eight-bit code used by the Model ASR 33 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 (octal) = Teletype). This code is read in the normal octal form used in the computer. Bits are numbered from right to left, from 1 through 8, with bits 1 through 3 containing the least significant octal number. The first information bit transmitted is bit 1, which is read into AC11. Figure 7-5 illustrates the context and description of the ASCII Teletype code and its associated bit content in the AC.

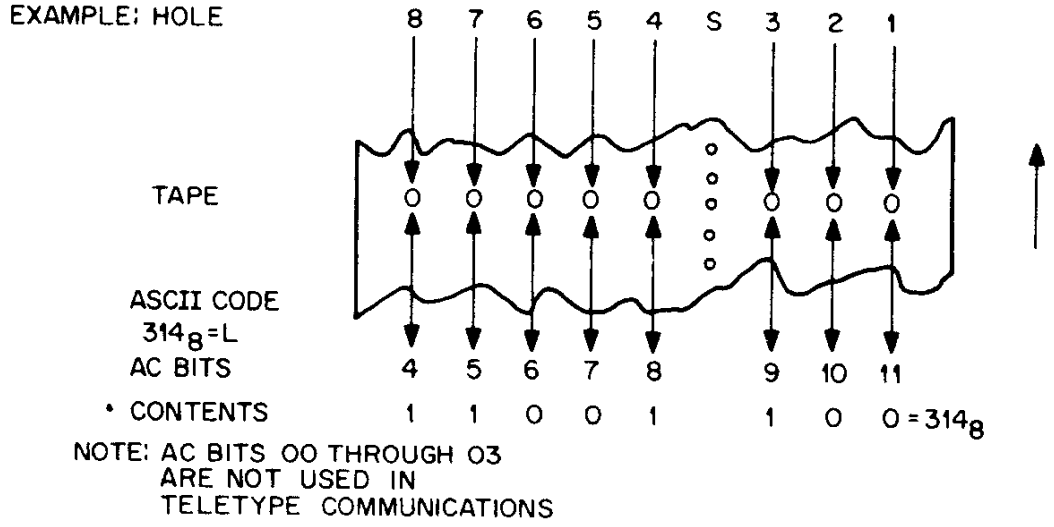


Figure 7-5 Relationship Between Teletype Tape AC Contents and Binary and Octal Number Being Transferred

The ASR 33 generates all assigned codes except 340 through 374 and 376, which are not assigned. Generally, codes 207, 212, 215, 240 through 337, and 377 are sufficient for Teletype operation. The ASR 33 detects all characters, but does not interpret all of the codes that it generates as commands.

The standard number of characters printed per line by the ASR 33 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 C lists the Teletype character code. Punched tape format (for 264 (octal)) is as follows:

Binary Code (Punch = 1)	8 7	6 5 4	Tape Channel S (Sprocket)	3 2 1
Octal Code	1 0	1 1 0		1 0 0

Teleprinter Control

Refer to Data Communications Equipment Options—KL8-E Asynchronous Data Control.

PAPER TAPE READER AND PUNCH OPTIONS

The options available for paper tape facilities are listed below.

PR8-E	Reader (with Control Unit)
PP8-E	Punch (with Control Unit)
PC8-E	Reader/Punch (with Control Unit)

Type PR8-E Paper Tape Reader

The PR8-E is available in two versions: the rack mounted version (PR8-EA) and the Table Top version (PR8-EB).

The PR8-E reader senses eight-hole unoiled grey perforated paper tape photoelectrically at a maximum rate of 300 characters per second. The control unit of the PR8-E plugs into the OMNIBUS and controls the action of the reader from program instructions. All connections between the control unit and the reader are made using a BC08-K cable.

Reader/Punch

A read operation is initiated by an RFC instruction from the computer. The control unit, in turn, initiates tape movement and sensing of a character, transfers the character to its reader buffer (RB), and sets its device flag to indicate that a character is available for transfer to the computer. The computer senses the reader flag by issuing an RSF instruction, and transfers the character from the RB to AC04 through 11 by issuing an RRB instruction. The RRB instruction also clears the reader flag to ready the unit for another read operation.

The control unit also contains an interrupt enable flip-flop. This flip-flop, controlled by program instructions, determines whether the reader can generate an interrupt request to the program interrupt facility. When set by an RPE instruction or initialize input, this flip-flop enables generation of an interrupt request from the reader flag being set. When cleared by a PCE instruction, this flip-flop inhibits interrupt requests.

Programming

Instructions for operating the reader are as follows:

Set Reader/Punch Interrupt Enable (RPE)

Octal Code: 6010

Operation: Sets the reader/punch interrupt enable flip-flop so that an interrupt request can be generated when reader or punch flag is set.

Skip on Reader Flag (RSF)

Octal Code: 6011

Operation: Senses the reader flag; if it contains a binary one, increments the PC by one so that the next sequential instruction is skipped.

Read Reader Buffer (RRB)

Octal Code: 6012

Operation: ORs the content of the reader buffer into AC4-11 and clears the reader flag. This command does not clear the AC.

Reader Fetch Character (RFC)

Octal Code: 6014

Operation: Clears the reader flag, loads one character into the RB from the tape, and sets the reader flag when the RB is full.

Read Buffer and Fetch New Character (RRB, RFC)

Octal Code: 6016

Operation: Combines RRB and RFC. The contents of the reader buffer is ORed into the AC. The flag is immediately cleared, and a new character is read from tape into the reader buffer. The flag is then set.

Clear Reader/Punch Interrupt Enable (PCE)

Octal Code: 6020

Operation: Clears the reader/punch interrupt enable flip-flop so that interrupt requests cannot be generated.

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

```
        RFC         /FETCH CHARACTER FROM TAPE
LOOK,   RSF         /SKIP IF READER FLAG = 1
        JMP LOOK    /JUMP BACK & TEST FLAG AGAIN
        CLA         /CLEAR AC
        RRB         /LOAD AC FROM RB, CLEAR READER FLAG
```

PP8-E Paper Tape Punch

The PP8-E is available is two versions: The rack mountable version (PP8-EA) and the table top version (PP8-EB).

The PP8-E paper tape punch consists of a control unit and a punch assembly that perforates eight-hole unoiled grey perforated paper tape at a rate of 50 characters per second. The control unit plugs into the OMNIBUS of the PDP-8/E, and controls the operation of the punch from program instructions. All connections between the control unit and the punch are made using a BC08-K cable.

A punch operation can be performed using a PCF instruction, followed by a PPC instruction or by a PLS instruction. In either event, the punch flag and punch buffer (PB) are cleared. An eight-bit character is then loaded into the PB from AC04 through 11 and the character is perforated on tape. After the character is punched, the punch flag is set to denote that the punching operation is complete. The punch flag is sensed by a PSF instruction to begin another punch operation.

The control unit also contains an interrupt enable flip-flop that is set or cleared by program instruction. When set by an RPE instruction or INITIALIZE, this flip-flop enables gating of an interrupt request to the program interrupt facility when the reader or punch flag is set. When cleared by a PCE instruction, this flip-flop prevents interrupt requests.

Programming

Instructions for the punch are as follows:

Set Reader/Punch Interrupt Enable (RPE)

Octal Code: 6010

Operation: Sets the reader/punch interrupt enable flip-flop so that an interrupt request can be generated when punch or reader flag is set.

Clear Reader/Punch Interrupt Enable (PCE)

Octal Code: 6020

Operation: Clears the reader/punch enable flip-flop so that interrupt requests cannot be generated.

Skip on Punch Flag (PSF)

Octal Code: 6021

Operation: Senses the punch flag; if it contains a binary one, increments the PC by one so that the next sequential instruction is skipped.

Clear Punch Flag (PCF)

Octal Code: 6022

Operation: Clears the punch flag in preparation for receiving a new character from the computer.

Load Punch Buffer and Punch Character (PPC)

Octal Code: 6024

Operation: Transfers the eight-bit character in AC4-11 into the PB, then punches that character. The instruction does not clear the punch flag or the PB.

Load Punch Buffer Sequence (PLS)

Octal Code: 6026

Operation: Clears the punch flag, transfers the contents of AC4-11 into the punch buffer, punches the character in the PB on tape, and sets the punch flag when the operation is completed.

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

```
FREE,PSF        /SKIP IF PUNCH FLAG = 1
     JMP FREE   /JUMP BACK & TEST FLAG AGAIN
     PLS        /CLEAR PUNCH FLAG & PB, LOAD PB
                /FROM AC, PUNCH CHARACTER, SET
                /PUNCH FLAG WHEN DONE
```

PC8-E Reader/Punch

The PC8-E is available in two versions: the rack mountable version (PC8-EA) and the table top version (PC8-EB).

The PC8-E consists of a reader and punch mounted on the same chassis and a control unit which plugs into the OMNIBUS and controls the action of the reader/punch from program instructions. All connections between the control unit and reader/punch are made using two BC08-K cables.

The reader portion of the PC8-E operates in the same manner as the PR8-E. Similarly, the punch portion of the PC8-E operates in the same manner as the PP8-E.

Specifications

Tape Type	1-inch fan-folded unoiled grey paper
Channels	8 data channels plus feedhole
Read Character Rate (Continuous)	300 characters/second
Read Character Rate (Start-Stop Mode)	25 characters/second
Punch Character Rate	50 characters/second

Programming

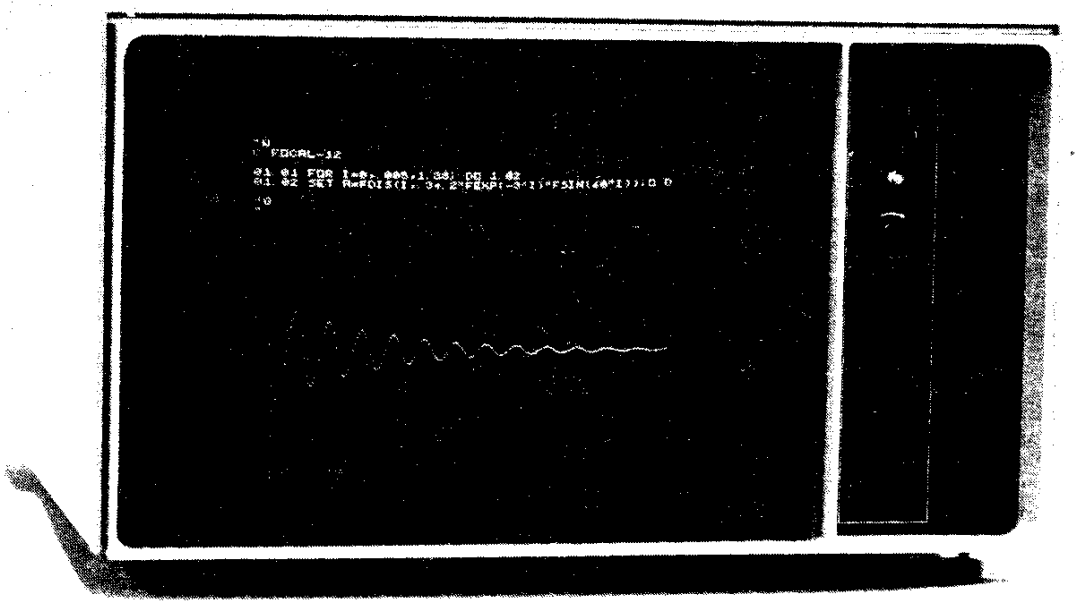
The PC8-E uses the same instructions (and programming sequence) as the PR8-E and PP8-E.

CRT DISPLAYS

Point Plot Display System

The VC8-E, when combined with a VR14 Oscilloscope, or a customer's scope, is capable of displaying data in the form of 1024_{10} by 1024_{10} dot array. Under programmed control, a bright spot may be momentarily produced at any selected point in this array. Thus a series of these intensified dots may be programmed to produce graphical output on a CRT.

Interfacing to the PDP-8/E Processor is accomplished with the VC8-E Control which plugs directly into the OMNIBUS. Information is applied from the processor's AC Register to the display by means of programmed IOT instructions. The displayed information can therefore be on line sampling or memory data or data from a mass storage device. The graphical presentation is limited only by the extent of programming the user desires to implement.



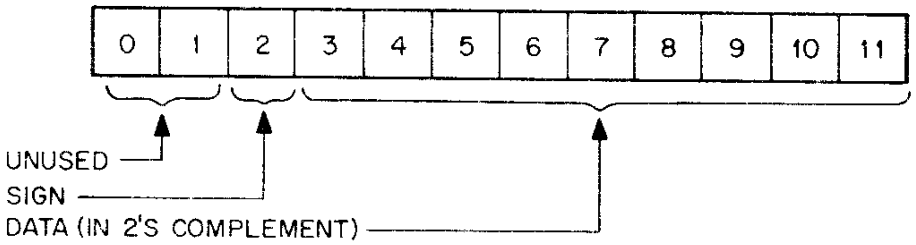
VR14 Display

Type VR14 Oscilloscope Display

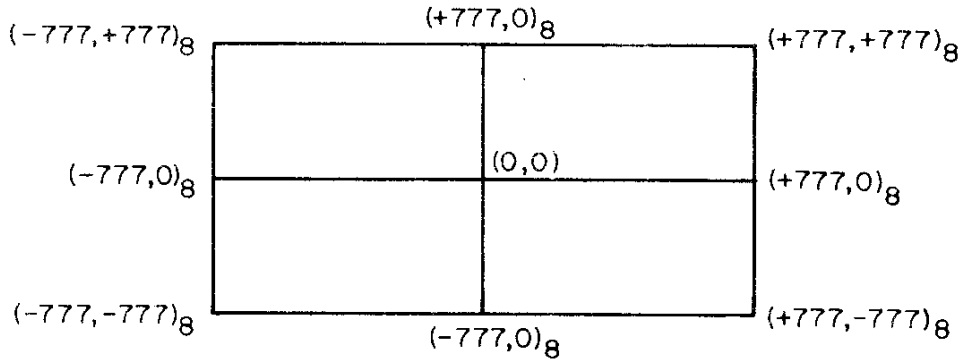
The VR14 is a compact solid-state CRT display with self-contained power supplies and a viewing area of 6¾ in. x 9 in. The VR14 can plot 1500 random points and up to 75 in. of vector with no flicker. X/Y deflection speed is 900 ns intensified, 700 ns non-intensified, and less than 20 μs is required for a maximum deflection step in any direction. Interface with the VC8-E is by means of connector assembly BC01K-10 (10 feet), BC01K-25 (25 feet), or BC01K-50 (50 feet), with ten feet the standard length.

VC8-E Point Plot Display Control

The VC8-E is a two-axis (X and Y) digital-to-analog converter plus intensifying circuitry (Z axis) that provides deflection and intensity information to the display oscilloscope. Coordinate data is transferred to the X and Y axis from bits 2-11 of the PDP-8/E accumulator. This data must be in the range of $\pm 777_8$ and transferred from the rightmost 10 bits of the PDP-8/E accumulator.



The position of the oscilloscope beam will be determined by the contents of the X and Y buffer registers. Coordinate (0,0) is located in the center of the screen.



The user is reminded of the relationship between the signed octal numbers used above and their corresponding 2's complement form.

Signed Values (used in example)	2's Complement (10 bit)	Position
+777	0777	Top or right
•	•	
•	•	
•	•	
+1	0001	
0	0000	Center
−1	1777	
•	•	
•	•	
•	•	
−777	1001	Bottom or left

Specifications

The VC8-E consists of a two-axis digital-to-analog converter and intensifying circuit that provides deflection and intensity signals, which are then applied to the input amplifier circuitry of such display units as the Type VR14 oscilloscopes. The control circuit for the VC8-E is located on a PDP-8/E module (M869), which plugs into the OMNIBUS.

The basic system of the VC8-E consists of the following circuitry:

a. OMNIBUS interface, IOT decoding, skip, clear AC, and interrupt control.

b. X-axis buffer, D/A converter, filter and summing amplifier, and bipolar line driver.

c. Y-axis buffer, D/A converter, filter and summing amplifier, and bipolar line driver.

d. Z-axis control, which consists of provision for intensity signal necessary for the VR03A oscilloscope and intensity and channel select signals necessary for the VR14 oscilloscope.

NOTE on Display Times

The display times of those instructions that include intensification depend upon the type of oscilloscope used.

VR14	21 μs
Tektronix 602	6 μs

A switch is provided to select the proper setting interval.

Programming

The instructions for outputting data to the oscilloscope display are defined as follows:

Clear All Logic (DILC)

Octal Code: 6050
Operation: Clears enables, flags, and delays.

Clear Done Flag (DICD)

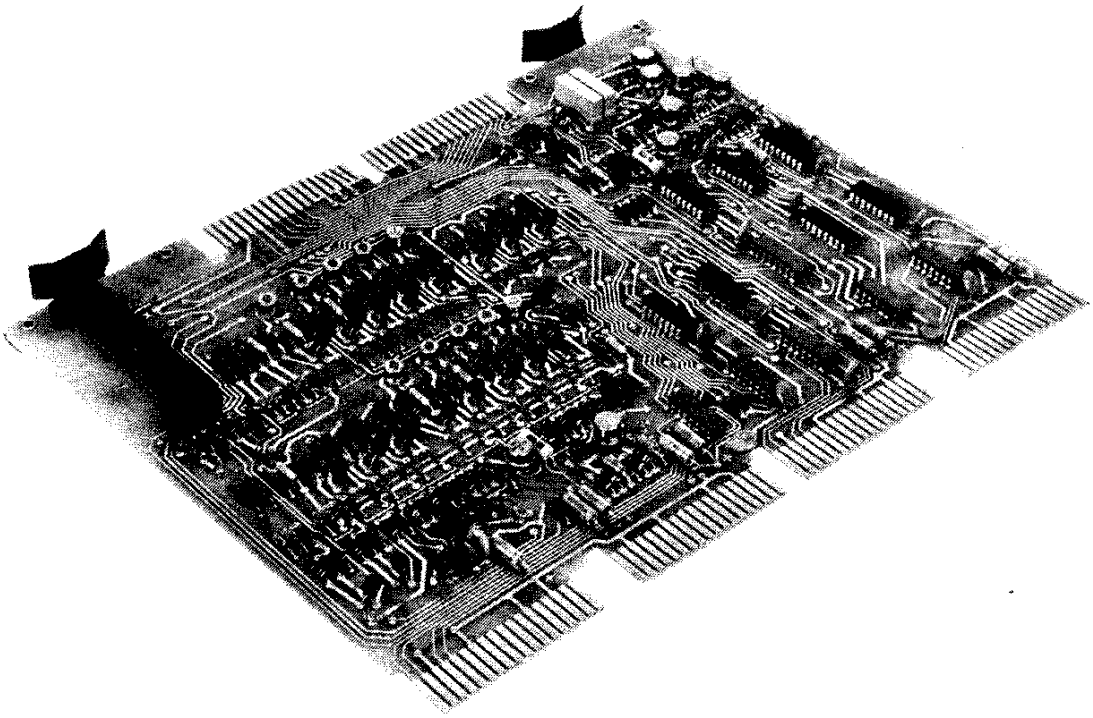
Octal Code: 6051
Operation: Clears Done Flag.

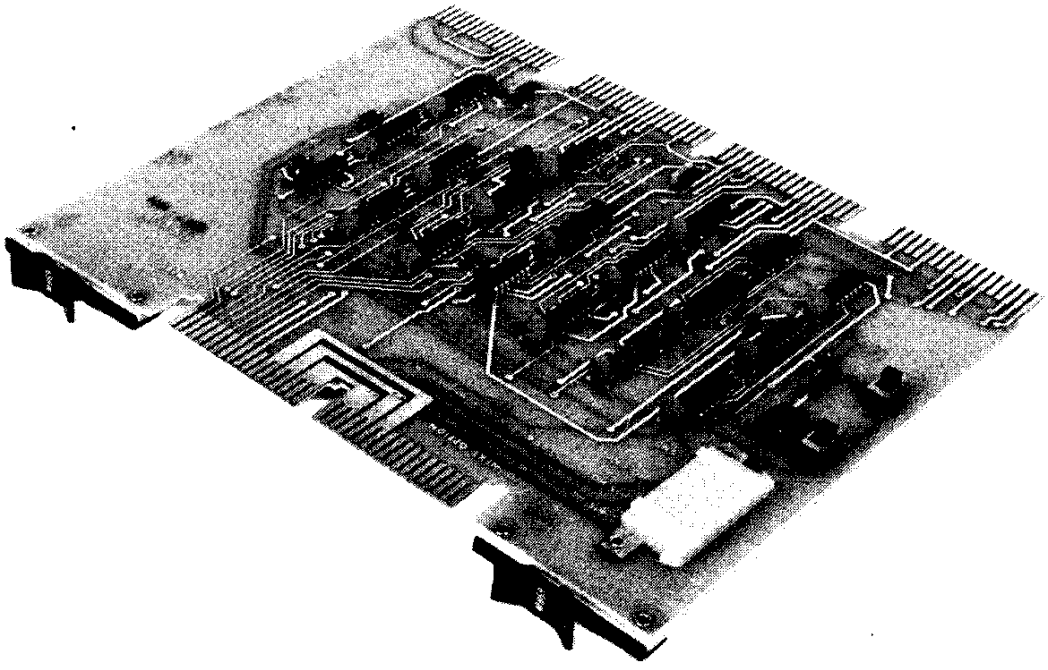
Skip On Done Flag (DISD)

Octal Code: 6052
Operation: Skip if Done Flag (1). Do not Clear Done Flag.

Load X Register (DILX)

Octal Code: 6053
Operation: Clear Done Flag; load X register, wait for settle.* Set Done Flag. Do not clear AC.





VC8-E

Load Y Register (DILY)

Octal Code: 6054

Operation: Clear Done Flag; load Y register, wait for settle.* Set Done Flag. Do not Clear AC.

Intensify (DIXY)

Octal Code: 6055

Operation: Clear Done Flag; intensify; Set Done Flag.

Load Enable (DILE)

Octal Code: 6056

Operation: Transfer contents of AC to Enable Register as defined below. Clears AC.

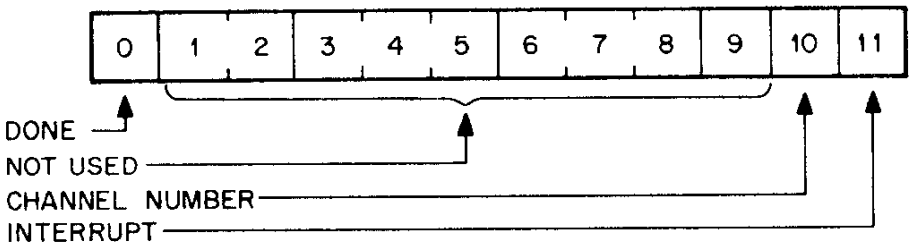
Read Enable/Status Register (DIRE)

Octal Code: 6057

Operation: Transfer the contents of the Display Enable/Status Register to the AC as defined below:

* SEE NOTE ON DISPLAY TIMES

Display Enable/Status Register



The Done Flag (bit 0) may be read using a DIRE (transfer enable to AC) command. It may not be set under program control using the DILE (load enable, clear AC) command.

Channel number selects the VR14 display channel. Bit 10 = 0, channel 0; Bit 10 = 1, channel 1.

Interrupt set to a one will cause the processor to interrupt (JMS 0) on done = 1.

Both channel number and interrupt may be loaded from the read into the AC using the DILE and DIRE commands respectively.

Programming Examples

The VC8-E is a very fast display control. So fast, in fact, that most display oscilloscopes cannot position their beam before an intensify command is performed. For this reason a "DONE" Flag has been incorporated into the control and should be used whenever random points are plotted sequentially.

```
        •
        •
        •
        CLA
        TAD   X                 /GET X-COORDINATE
        DILX                    /LOAD X REGISTER
        CLA
        TAD   Y                 /GET Y-COORDINATE
        DILY                    /LOAD Y REGISTER
        DISD                    /SKIP ON DISPLAY DONE FLAG
        JMP .—1
        DIXY                    /INTENSIFY POINT
```

The following example displays a dot on the screen whose coordinates are set by the position of the ADC's parameter knobs 0 and 1:

```
START,  CLA
        JMS     SAMPLE  /POSITION OF KNOB 0
        DILX            /LOAD
        CLA     IAC
        JMS     SAMPLE  /POSITION OF KNOB 1
        DILY            /LOAD
        DISD
        JMP .—1

        DIXY            /INTENSIFY
        JMP     START
SAMPLE, 0
        ADLM
        ADST
```

```
        JMP .—1
        ADRB
        JMP I SAMPLE
```

A fun program for the VC8-E is Kaleidoscope. Pictures on the screen are varied by manipulating the switch register bits 9, 10, and 11.

```
START,  TAD Y
        JMS SCALE
        CMA
        TAD X
        DCA X
        TAD X
        DILX
        JMS SCALE
        TAD Y
        DILY
        DISD
        JMP .—1
        DIXY
        DCA Y
        JMP START

SCALE,  0
        DCA TEM
        OSR
        CIA
        DCA C
        TAD TEM
        CLL
        SPA
        CML
        RAR
        ISZ C
        JMP .—5
        JMP I SCALE
```

VT05 ALPHANUMERIC DISPLAY TERMINAL

The VT05 is a flexible, high-performance alphanumeric display terminal with a video cathode ray tube display and communications equipment. It is capable of transmitting data over standard phone lines and data sets in half or full duplex modes at rates up to 300 Baud. For remote users, the VT05 serves as a non-mechanical terminal that handles data speeds many times faster than that of conventional teletypewriters. If desired, the alphanumerics can be superimposed on a background video image derived from a closed circuit TV camera or video tape player.

For user convenience, the VT05 display includes the following outstanding features:

- Completely interchangeable with Teletype (20 mil current loop)
- EIA RS-232C compatible communications interface
- Totally self-contained
- Direct cursor addressing
- Concurrent video-alphanumeric imaging
- Easy-to-read characters
- Solid-state circuitry
- Comprehensive 64/128 character set keyboard

The VT05 Alphanumeric Display Terminal can be controlled by the KL8-E, EA, EB, EC or the DC02-FB, DC02-G and BC01A-25. The same program used with the Teletype units is used with the VT05 display.

Specifications

DISPLAY

Screen Size—10⅛" x 7⅝"
Character Display Area—8¾" x 6⅝"
Characters/Line—72
Number of Lines—20
Number of Characters Displayable—1440
Contrast Ratio—12:1
Type of Phosphor—P4 (white)
Deflection Type—Magnetic
Deflection Method—Raster Scan
Character Generation Method—5 x 7 dot matrix
Character Generator—Read Only Memory (ROM)
Refresh Buffer—MOS Memory
Memory Size:
ROM—2240 bits
Refresh Buffer—9816 bits
Display Refresh Rate—60 times/sec or 50 times/sec synchronized to power line frequency
Character Set—Upper case ASCII
Character Size—.23" x .11"
Cursor—Non-destructive, blinking (underline)

VIDEO

Standard EIA-compatible signal

KEYBOARD/CONTROL

Type—Electronic (wafer switch)
Standard model Teletype layout
Character Set—Selectable (upper case, standard ASCII; upper/lower case, full ASCII)
Controls:

Cursor —Up, down, left, right, home up
—Direct addressing, Tab
Erase —To end of line, to end of frame
Erase Lock —Prevents inadvertent erasure
Power —On, off
Mode —Remote, local
Transmission —Full, half duplex

MECHANICAL/ENVIRONMENTAL

Dimensions:
Width—19"
Height—12"
Depth—30"
Weight—55 lbs.

Heat Dissipation—800 BTU/hr. maximum
Operating Temperature—40°—100°F, 4.4°—37.8°C
Humidity—10 to 95%

POWER INPUT

VT05-A: 95-130 VAC, 60 Hz ± 2 Hz, single phase
VT05-B: 190-260 VAC, 60 Hz ± 2 Hz
VT05-C: 95-130 VAC, 50 Hz ± 2 Hz
VT05-D 190-260 VAC, 50 Hz ± 2 Hz
Power Consumption—130 watts

DATA TRANSMISSION

Type—Crystal-controlled, selectable speed; send/receive 110, 150, 300 Baud

APPLICATIONS

General-Purpose Timesharing

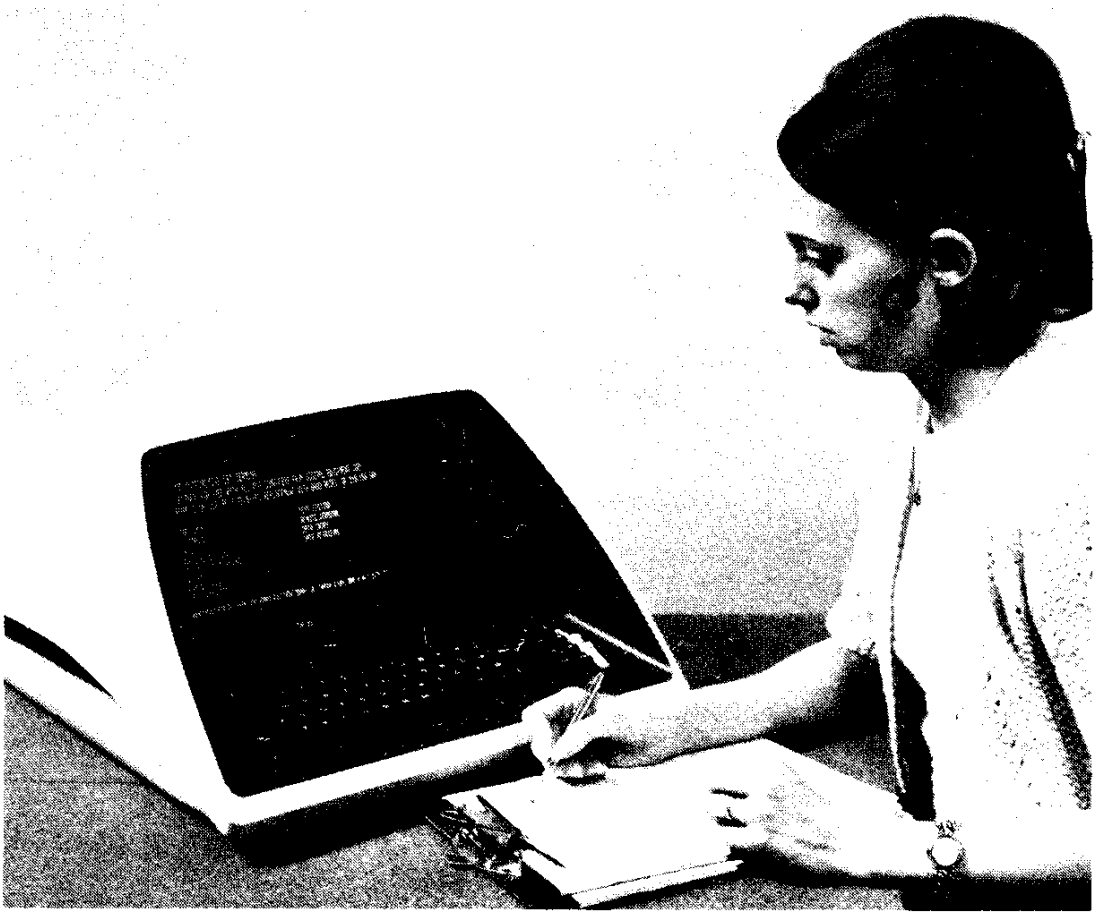
Timesharing systems are pioneering a new way of life in many scientific and technical disciplines. The time spent by professional workers at the terminal in dialog with a computer is critical productivity time. The obviously strong need for terminal equipment that increases this productivity is satisfied by the VT05 Alphanumeric Display Terminal. It is designed to make the professional's "on-line" time totally useful. Also, its selectable transmission speeds allow terminal users to utilize any available data communication system, including simple acoustical couplers and digital modems.

Computer-Aided Instruction

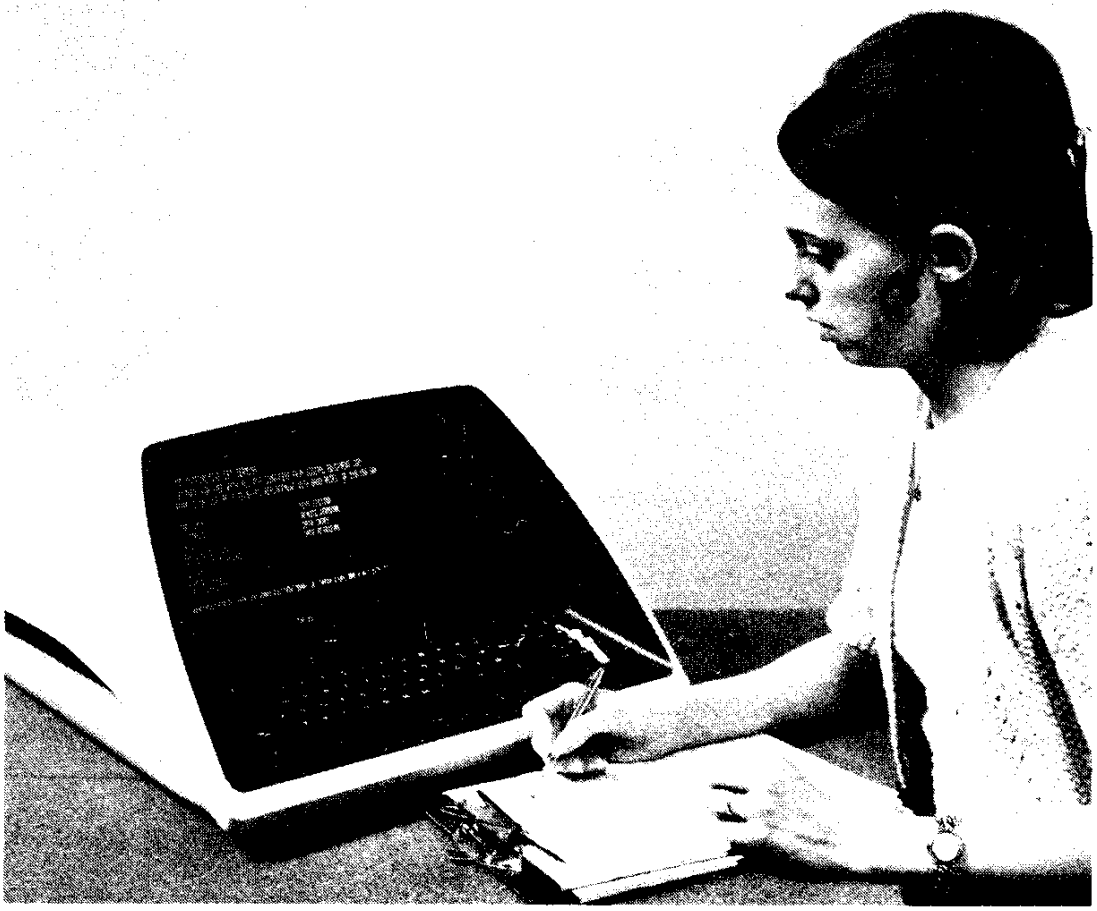
In the learning process, the VT05 terminal enables the simultaneous display of background video images and foreground alphanumeric information. At the elementary instruction level, foreground displays of words and numbers can be reinforced by static or dynamic pictures of the things themselves. The same technique is also appropriate for advanced levels of instruction such as medical school anatomy classes, repair mechanic training, and even photo intelligence evaluations. The background video image can be obtained directly from a TV camera or indirectly from a video tape player.

Hospital Systems

The VT05 fulfills all the necessary requirements for use in the hospital environment in multi-station paging, clinical and research applications. It is noiseless (no bothersome hum or clatter) and consequently eliminates intrusion upon the user, patients or subjects in the immediate vicinity. Also, it is extraordinarily simple to operate; no instruction manual is required, so anyone who can type can run it.



VT05 Display



VT05 Display

The VT05 utilizes solid-state elements, thereby guaranteeing high reliability with correspondingly fewer maintenance problems. It is completely portable, weighing only 55 pounds, and is easily connected to a standard acoustical coupler or a data set even by an unskilled operator.

The CRT screen displays a total of 1440 characters. A keyboard-controlled cursor is operated under program control to help revise, correct or delete any character, any line or any combination. This control via the computer allows simple question-and-answer type data logging to be accomplished at remote stations by non-computer operators.

Industrial and Commercial

The VT05 is completely self-contained on one rugged, compact package. It includes the keyboard, CRT, refresh memory, communications interface, and power supply.

The characters displayed on the CRT are refreshed 60 (50) times per second which obviates any flicker. A tinted glass shield is provided to reduce glare and make the VT05 visually comfortable to use. The simple keyboard allows for rapid entry of data.

All of these features, plus its handsome modern design, make the VT05 an ideal clerical tool for office or laboratory. With its video capability, moreover, it can also serve as a remote monitor for hazardous experiments or production processes; e.g., working with radioactive materials, noxious fumes, or toxic substances.

ASCII CODE ASSIGNMENTS

STANDARD TRANSMIT CODE ASSIGNMENTS

Bit No. 7 6 5 / 4 3 2 1	0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
0 0 0 0			SPACE	0	@	P	@	P
0 0 0 1			!	1	A	Q	A	Q
0 0 1 0			"	2	B	R	B	R
0 0 1 1			#	3	C	S	C	S
0 1 0 0			$	4	D	T	D	T
0 1 0 1			%	5	E	U	E	U
0 1 1 0			&	6	F	V	F	V
0 1 1 1			'	7	G	W	G	W
1 0 0 0	C← (BS)	C→	(	8	H	X	H	X
1 0 0 1	HT		)	9	I	Y	I	Y
1 0 1 0	LF	C↑	*	:	J	Z	J	Z
1 0 1 1	C↓		+	;	K	[	K	[
1 1 0 0			,	<	L	\	L	\
1 1 0 1	CR	HOME	-	=	M	]	M	ALT.
1 1 1 0		ERASE LINE	.	>	N	^	N	^
1 1 1 1		ERASE SCREEN	/	?	O	_	O	DEL (RUB OUT)

FULL ASCII TRANSMIT CODE ASSIGNMENTS

Bit No. 7 6 5 / 4 3 2 1	0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
0 0 0 0			SPACE	0	@	P	`	p
0 0 0 1			!	1	A	Q	a	q
0 0 1 0			"	2	B	R	b	r
0 0 1 1			#	3	C	S	c	s
0 1 0 0			$	4	D	T	d	t
0 1 0 1			%	5	E	U	e	u
0 1 1 0			&	6	F	V	f	v
0 1 1 1			'	7	G	W	g	w
1 0 0 0	C← (BS)	C→	(	8	H	X	h	x
1 0 0 1	HT		)	9	I	Y	i	y
1 0 1 0	LF	C↑	*	:	J	Z	j	z
1 0 1 1	C↓	ALT	+	;	K	[	k	{
1 1 0 0			,	<	L	\	l	|
1 1 0 1	CR	HOME	-	=	M	]	m	}
1 1 1 0		ERASE LINE	.	>	N	^	n	~
1 1 1 1		ERASE SCREEN	/	?	O	_	o	DEL (RUB OUT)

RECEIVE CODE ASSIGNMENTS

Bit No. 7 6 5 / 4 3 2 1	0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
0 0 0 0			SPACE	0	@	P	@	P
0 0 0 1			!	1	A	Q	A	Q
0 0 1 0			"	2	B	R	B	R
0 0 1 1			#	3	C	S	C	S
0 1 0 0			$	4	D	T	D	T
0 1 0 1			%	5	E	U	E	U
0 1 1 0			&	6	F	V	F	V
0 1 1 1	BELL		'	7	G	W	G	W
1 0 0 0	C← (BS)	C→	(	8	H	X	H	X
1 0 0 1	HT		)	9	I	Y	I	Y
1 0 1 0	LF	C↑	*	:	J	Z	J	Z
1 0 1 1	C↓		+	;	K	[	K	[
1 1 0 0			,	<	L	/	L	/
1 1 0 1	CR	HOME	-	=	M	]	M	
1 1 1 0	CAD	ERASE LINE	.	>	N	^	N	^
1 1 1 1		ERASE SCREEN	/	?	O	_	O	

CURSOR ADDRESS CODE ASSIGNMENTS

Bit No. 7 6 5 / 4 3 2 1	0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
0 0 0 0			1	17	33	49	65	
0 0 0 1			2	18	34	50	66	
0 0 1 0			3	19	35	51	67	
0 0 1 1			4	20	36	52	68	
0 1 0 0			5	21	37	53	69	
0 1 0 1			6	22	38	54	70	
0 1 1 0			7	23	39	55	71	
0 1 1 1			8	24	40	56	72	
1 0 0 0			9	25	41	57		
1 0 0 1			10	26	42	58		
1 0 1 0			11	27	43	59		
1 0 1 1			12	28	44	60		
1 1 0 0			13	29	45	61		
1 1 0 1			14	30	46	62		
1 1 1 0			15	31	47	63		
1 1 1 1			16	32	48	64		

C = Cursor Function

X/Y PLOTTER OPTIONS

Type XY8/E Incremental Plotter Control

The XY8/E Incremental plotter control provides the control logic and interface necessary to operate an encoded or unencoded digital incremental plotter. It operates with a variety of plotters to display data graphically on paper or film.

Except for setting the coordinates at which plotting begins, all plotter operations are controlled by the plotter control logic and the processor. A series of functions controlled by IOT instructions initializes the plotter control logic, generates program interrupt requests to indicate change in status, and initiates a plotting operation.

The principles of operation are basically the same for all plotters. Drum plotter operations are described below:

Bidirectional rotary stepping motors are employed for both the X and Y axes. Recording is produced by movement of a pen in relation to the surface of graph paper, with each instruction resulting in an incremental step. X-axis deflection is derived from the motion of the drum; Y-axis deflection is derived from the motion of the pen carriage. Further instructions lower and raise the pen to and from the surface of the paper. Inputs to the plotter from the digital signal source consist of drum up, drum down, carriage right, carriage left, pen up, and pen down pulses. All recording (discrete points, continuous curves, or symbols) is accomplished by the incremental stepping action of the paper drum and pen carriage.

Controls on the plotters 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.

The entire interface is contained on a PDP-8/E module which plugs into the OMNIBUS.

NOTE

Unencoded Plotters can be:
CAL-COMP 500 Series
CAL-COMP 600 Series
HOUSTON INSTRUMENTS DP-10

Encoded Plotters can be:
CAL-COMP 600 Series
CAL-COMP 700 Series
CAL-COMP 800 SERIES (MICROFILM RECORDERS)

Programming

The following IOT instructions are used to operate the digital incremental plotters:

Clear Interrupt Enable (PLCE)

Octal Code: 6500
Operation: Clears the interrupt enable flip-flop.

Skip Plotter Flag (PLSF)

Octal Code: 6501
Operation: Senses the Plotter Flag, and, if it is set, increments the contents of the PC by one so that the next sequential instruction is skipped.

Clear Plotter Flag (PLCF)

Octal Code: 6502
Operation: Clears the Plotter Flag preparatory to issuing a plotter operation command.

Pen Up (PLPU)

Octal Code: 6503
Operation: Raises the plotter pen from the surface of the graph paper (unencoded plotters only).

Load Direction Register, Set Flag (PLLR)

Octal Code: 6504
Operation: Loads the direction register from AC6-11, which performs the following function:

Unencoded Plotter

AC BIT	FUNCTION
6	Pen Right
7	Pen Left
8	Drum Down
9	Drum Up

Encoded Plotter Function

AC06-11	PAPER PLOTTERS (700 SERIES)	FILM PLOTTERS (800 SERIES)
10	+y	+y
11	+x +y	+x +y
12	+x	+x
13	+x −y	+x −y

14	−y	−y
15	−x −y	−x −y
16	−x	−x
17	−x +y	−x +y
30	not used	CRT Shift
31	Pen Up	Beam Off
32	Pen Down	Beam On
33	Start Zip	not used
34	Block Code	−z (+ Aux 1)
35	Plot Code	+z (+ Aux 2)
36	Start Incr.	not used
37	Sync	Sync
50	+y/2	
51	+x/2 +y/2	
52	+x/2	
53	+x/2 −y/2	REMAINING
54	−y/2	CODES
55	−x/2 −y/2	NOT
56	−x/2	USED
57	−x/2 +y/2	
70	+x +y/2	
71	−x +y/2	
72	+x/2 +y	
73	−x/2 +y	
74	+x −y/2	
75	−x −y/2	
76	+x/2 −y	
77	−x/2 −y	

Pen Down (PLPD)

Octal Code: 6505
Operation: Lowers the pen to the surface of the graph paper (unencoded plotters only).

Clear Flag, Load Direction Register, Set Flag (PLCF, PLLR)

Octal code: 6506
Operation: This microinstruction combines octal instructions 6502 and 6504. It clears the Plotter Flag, loads the direction register from AC6-11, then sets the flag.

Set Interrupt Enable (PLSE)

Octal code: 6507
Operation: Sets the interrupt enable flip-flop.
Any sequence of programmed IOTs must assume that the pen location is known at the start of the routine, as there is no way to specify an absolute location in an incremental plotter except by the manual controls on the recorder. During a subroutine, the PDP-8/E can track the location of the pen on the paper by counting the instructions that increment the position of the pen and the drum.

Type XY8-EA Digital Incremental Plotter
The XY8-EA consists of the XY8-E interface described above and the Calcomp (California Computer Products) Model 565 Digital Incremental Plotter. The Model 565 is a high-speed, drum-type plotter, capable of performing up to 18,000 steps per minute. Each of these steps causes the drum or pen carriage to move a fixed increment in either a positive or negative direction. The size of this increment can be 0.01 inch, 0.003 inch, or 0.1 mm, depending on the gear ratios used for the drum and carriage drives.

A bidirectional roll paper feed and takeup mechanism accepts chart paper rolls 12 inches wide by 120 feet long. The paper is driven by pins on the drum which engage holes on both edges of the paper to maintain registration between the recording pen and the paper. If desired, single sheets of chart paper may be used for plotting instead of the roll paper.

Type XY8-EB Digital Incremental Plotter
The XY8-EB consists of the XY8-E interface together with the Calcomp Model 563 Digital Incremental Plotter. This is very similar to the Type XY8-EA, except that the Model 563 accepts a paper width of 30 inches.

Type XY8-EH, EJ, EK Digital Incremental Flatbed Plotter

The XY8-EH, EJ, EK plotters consists of the XY8-E interface plotter control described above and the Houston Instruments Model DP-10 Plotter.

The Digital Incremental plotter combines the low price and physical attributes of a high quality flatbed X-Y recorder, with the precision, accuracy, and stability obtainable only with incremental positioning. The X and Y pen positioning beams are driven by precision, bi-directional stepping motors which are geared to produce an increment of .005" for each input pulse. Plots up to 11" x 17" may be generated online, off-line, or remotely depending on the system configuration. Input to the plotter is standard 8 vector format, and is plug to plug compatible with existing incremental plotter controllers designed to drive continuous chart plotters such as the XY8-E.

SPECIFICATIONS

Input Requirements:

- Positive or Negative going pulse greater than 10 volt amplitude, less than 10μ seconds rise time with greater than 4μ seconds duration. SK—19—32—SL connector.
- Maximum Pulse rate—200 increment commands/second (1/3.33 ms).
- Pen Up/Down stabilizing time—60 ms, Down; 10 ms, Up.

- Input Functions—(+X), (−X), (+Y), (−Y), PEN UP, PEN DOWN. Normally, eight plotting vectors are obtained by appropriate combination of the basic directions.
- 3 volt inputs available for compatibility with DTL or TTL logic. Specify on original order!

Step Size and Speed

- .005" Increment
- 1.5 IPS (0°, 90°, 180°, 270°)
- 2.12 IPS (45°, 135°, 225°, 315°)

Physical Dimensions and Mounting (Vertical Mount)

- Depth—6½ inches
  Width—17-⅝ inches (19" with rack mounting)
  Height—15-¾ inches

Pens and Chart Hold Down

- Supplied with ball point and fibre tip disposable pens.
- Chart held down by vacuum system—capable of handling either 8½" x 11" or 11" x 17" charts.

CONTROLS

- "POWER" "ON/OFF"—Toggle Switch
- Manual Pen Position—Three-position Toggle switches with center "off" position. One switch positions pen in the (+) or (−) X direction; the other positions the pen in (+) or (−) Y. A single step in the respective direction will result if the switch is momentarily actuated. If the switch remains actuated for more than approximately 1 second, a continuous movement will occur at a nominal speed of 1 inch/second, until the switch is released. These positioning switches are not functional when the "Pen" is in the "Remote" position.
- "LOAD/PLOT" Toggle Switch—spring loaded to the "PLOT" mode. Whenever the "LOAD" position is momentarily selected, the pen will automatically be positioned to the lower left corner of the plotter for the dual purpose of (1) establishing an X and Y reference zero point, and (2) locating the pen beam so that a new chart may be loaded without interference.
- "PEN" "REMOTE"/"UP"/"DOWN"—3-position toggle switch—In "REMOTE" position, the pen will be under program control and will remain latched in whatever state that was last selected

by the program. "UP" and "DOWN" will allow the operator to raise or lower the pen manually, irrespective of the program selected state of the pen.

POWER REQUIREMENTS:

Can be connected for either 115 or 230 VAC, 50/60 Hz. Connected for 115 VAC unless suffix "J."

Maximum Apparent Power is 120 Volt/Amperes with Pen Down and No Pen Motion. Line Voltage tolerance is plus or minus 10% from nominal input requirements.

CONFIGURATIONS

Type	Type Number
115VAC input power	XY8-EH
230VAC input power	XY8-EJ
Table Top version, 115VAC	XY8-EK

LINE PRINTER OPTION

LE8 Line Printer

The LE-8 line printer offers the user a low-cost, high-speed, flexible method of printing computer output at a rate dependent upon the option selected. It accepts ASCII characters from the AC.

Each character is selected from the set of 64 (or 96) available by means of six-bit or seven-bit binary code. (Appendix E lists the ASCII code for each character.) Each code is loaded separately from the computer into a 20-character (or, in the case of the 132-column model, 22-character) core storage Line Printer Buffer from AC 6-11 (or AC 5-11), with the least significant bit appearing in AC11. After each code is transferred into the Line Printer Buffer, the Line Printer Done Flag appears, indicating that the printer is ready to receive the next character. When the Line Printer Buffer is filled, or a control character has been received, the print cycle is initiated. Character codes not in the character set in Appendix E are printed as spaces. The line feed command and carriage return command are similar to the corresponding commands in the Teletype. The form feed command advances the paper to the top of the page. The Printer Done Flag is set after each of these operations.

During the print cycle, the paper and inked ribbon pass between a row of 80 hammers (132 in the 132-column model) and the continuously rotating drum that contains all of the available characters. Variable reluctance pickoffs scan the stored characters in synchronism with the rotating characters, and the control system actuates the appropriate hammer as the desired character approaches the print position. The full line is printed in 20-column segments, with one drum revolution required for each segment. After the last character of a line is printed, the Line Printer Buffer is cleared automatically.

There are no operator controls in the control module. The following controls are on the printer:

The PDP-8/E system provides a low cost data retrieval system for such areas as car rentals, hotel/motel reservation, inventory control, data management, commodity market information, warehouse automated storage and retrieval and scores of other applications. Terminals such as the DECwriter, Display, line printer and card reader are ideal for data retrieval systems. Should the user desire remote terminals, both the display and the DECwriter can be placed in a remote facility and still be a part of the PDP-8/E system. Also, the line printer and card reader with another PDP-8/E operate well as a remote batch facility.

The PDP-8/E system provides a low cost data retrieval system for such areas as car rentals, hotel/motel reservation, inventory control, data management, commodity market information, warehouse automated storage and retrieval and scores of other applications. Terminals such as the DECwriter, Display, line printer and card reader are ideal for data retrieval systems. Should the user desire remote terminals, both the display and the DECwriter can be placed in a remote facility and still be a part of the PDP-8/E system. Also, the line printer and card reader with another PDP-8/E operate well as a remote batch facility.

TOP OF FORM—Advances paper to top-of-form position; disabled in on-line mode

PAPER STEP—Advances paper one line; disabled in on-line mode

ON LINE/OFF LINE—Selects mode of operation for the printer

MASTER CLEAR—Initializes printer to ensure proper state of electronic elements

PRINT INHIBIT—Inhibits print hammers.

The line printer is available in any of the following combinations:

LE8-FA	80 columns	64 characters	60 Hz
LE8-FB	80 columns	64 characters	50 Hz
LE8-HA	80 columns	96 characters	60 Hz
LE8-HB	80 columns	96 characters	50 Hz
LE8-JA	132 columns	64 characters	60 Hz
LE8-JB	132 columns	64 characters	50 Hz
LE8-KA	132 columns	96 characters	60 Hz
LE8-KB	132 columns	96 characters	50 Hz

The interface is contained on one PDP-8/E module, which plugs into the OMNIBUS.

The specifications for the LE8 line printer are as follows:

Printable characters	
character set	64 or 96
type	Open Gothic print
size	Typically 0.095 inches high and 0.065 inches wide
code format	ASCII
characters per line	80 or 132
drum speed	1760 rpm (64 character drum)
Print rate	
80 column model	
64 character	356 Lines/minute, columns 1-80 460 Lines/minute, columns 1-60 650 Lines/minute, columns 1-40 1110 Lines/minute, columns 1-20
96 character	253 Lines/minute, columns 1-80 330 Lines/minute, columns 1-60 478 Lines/minute, columns 1-40 843 Lines/minute, columns 1-20
132 column model	
64 character	245 Lines/minute, columns 1-132 290 Lines/minute, columns 1-110 356 Lines/minute, columns 1-88 460 Lines/minute, columns 1-66 650 Lines/minute, columns 1-44 1110 Lines/minute, columns 1-22

96 character	173 Lines/minute, columns 1-132 205 Lines/minute, columns 1-110 253 Lines/minute, columns 1-88 330 Lines/minute, columns 1-66 478 Lines/minute, columns 1-44 843 Lines/minute, columns 1-22
Format	Top-of-form control, single line advance and perforation step over.
Paper feed	One pair of pin-feed tractors for ½-inch hole center, edge-punched paper. Adjustable for any paper width from 4 inches to 9-7/8 inches on the 80-column model; or a maximum width of 14-7/8 inches for the 132 column model.
Paper slew speed	13 inches per second
Print area	8 or 13.2 inches wide, left justified
Character Spacing	10 characters per inch
Line spacing	6 lines per inch for 80-column, 6 or 8 lines for 132-column printer
Line advance time	20 milliseconds
Character synchronization	Variable reluctance pick-offs sense drum position

Printer Dimensions

	80 column	132 column
Height	46 inches	46 inches
Width	24 inches	48 inches
Depth	22 inches	25-inches
Weight	275 pounds	420 pounds

Printer Power Requirements

115 vac + or — 10%, 60 Hz + or — 3 Hz, single phase, 300 watts
or
240 vac + or — 10% 50 Hz + or — 3 Hz, single phase, 300 watts

Signal cable	25 foot interconnecting signal cable is supplied with system
Paper	
Type	standard fanfold, edge punched
Dimensions	4 inches to 9-7/8 inches wide (80 column) 4 inches to 14-7/8 inches wide (132 column) with 11 inches between folds

weight		
(single copy)	15 pound bond (minimum)	
multi copy)	12 pound bond with single-shot carbon for up to six parts	
Ribbon		
type	inked roll	
width	9 inches (80 column); 14 inches (132 column)	

Programming
The IOT instructions which command the line printer are:

Skip on Character Flag (PSKF)

Octal Code: 6661
Operation: Senses the content of the line printer done flag; if it contains a binary 1, the contents of the PC are incremented by one so that the next sequential instruction is skipped.

Clear the Character Flag (PCLF)

Octal Code: 6662
Operation: Clears the Line Printer Done Flag.

Skip on Error (PSKE)

Octal Code: 6663
Operation: Senses the content of the Line Printer Error Flag; if it contains a binary 1, indicating that an error (drum gate open, out of paper, excessive temperature) has been detected, the contents of the PC are incremented by one so that the next sequential instruction is skipped.

Load Printer Buffer, Print on Full Buffer or Control Character (PSTB)

Octal Code: 6664
Operation: Loads the character into the print buffer, and prints if the buffer is full, or if the character was a control instruction. This instruction does not clear the AC.

Set Program Interrupt Enable Flag (PSIE)

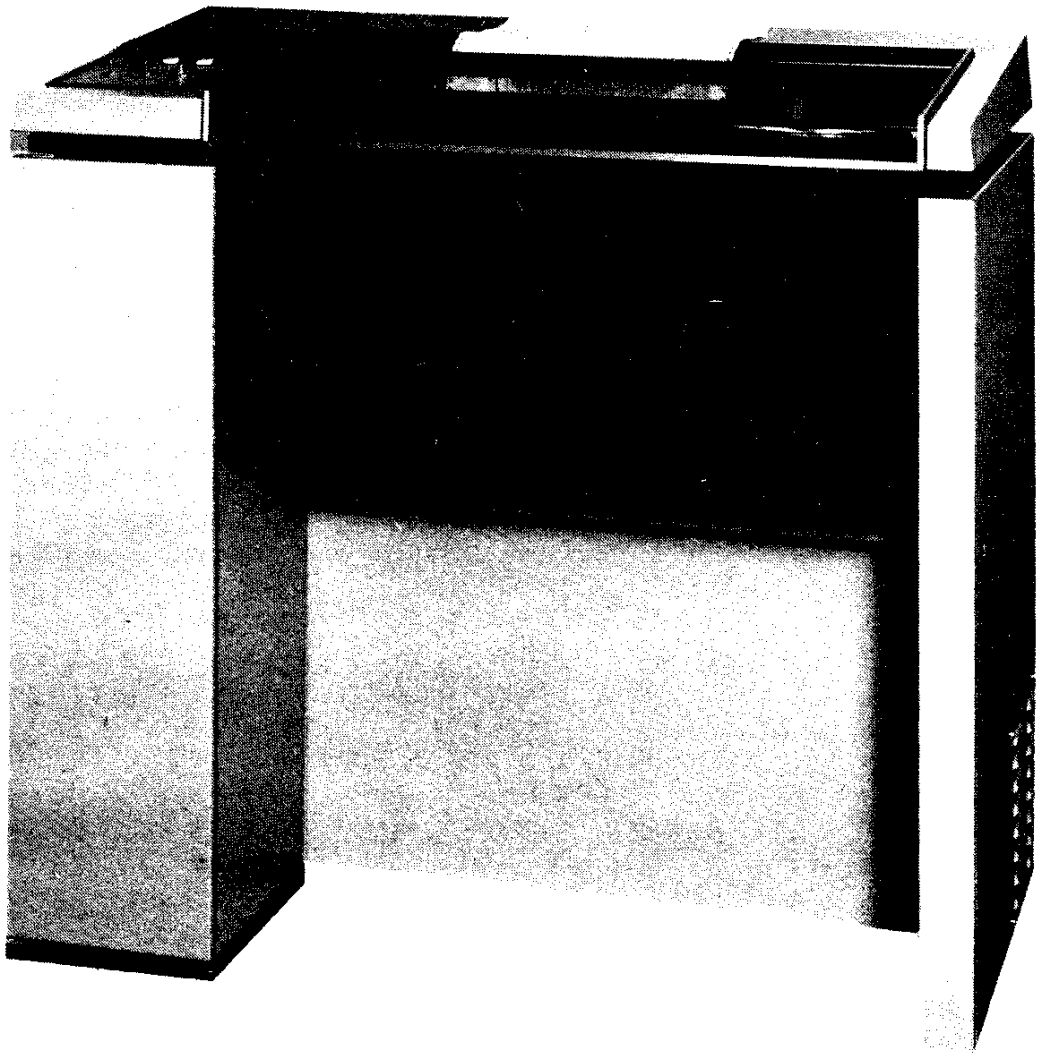
Octal Code: 6665
Operation: Sets the interrupt enable (IE) flip-flop to a one, permiting the Printer Done Flag to request a program interrupt.

Clear Line Printer Flag, Load Character, and Print (PCLF, PSTB)

Octal Code: 6666
Operation: This is a microprogram combination of PCLF and PSTB.

Clear Program Interrupt Flag (PCIE)

Octal Code 6667
Operation: Clears the interrupt enable flip-flop.



LE-8 Line Printer 132—Column Model

DATA COMMUNICATIONS EQUIPMENT OPTIONS

DP8-EA and DP8-EB Synchronous Modem Interface

The DP8-EA and DP8-EB interface the PDP-8/E with a full-dulplex or half-duplex synchronous modem for computer-to-terminal or intercomputer transfer of data. The PDP-8/E is capable of interfacing up to four communication links (channels), each having its own unique program instructions, modem interface, baud rate, priority assignment and access addresses. Data is exchanged between the PDP-8/E and the DP8-EA/EB in parallel form, using the data break facility of the computer. Data is exchanged between the DP8-EA/EB and a modem in serial form. Thus, the DP8-EA/EB performs parallel-to-serial conversion for transmit functions and serial-to-parallel conversion for receive functions. The interface also provides level conversion, character detection, modem control, and program-controlled interface with the computer.

Data exchange between the PDP-8/E and DP8-EA/EB interface is accomplished via the data break facility to or from any location within 32K of memory. Word count (WC), Current Address (CA) and character detection are performed using additional data break cycles to a specified set of locations in field zero. The DP8-E interface for one communication link consists primarily of MSI logic packaged on two PDP-8/E modules, which plug into the OMNIBUS. A cable provided with the modules mates with a connector on the modem. Two types of interface units (designated DP8-EA and DP8-EB) are available. A DP8-EA interface operates with a Bell System 200-Series Modem or equivalent, and DP8-EB operates with a Bell System 300-Series Modem or equivalent.
connector.

Specifications

Data Transfers	Transfers are maintained via three single cycle data breaks (1.4 micro seconds each cycle) for both transmit and receive. An additional cycle is required for each special character to be tested for (receive circuits only).
Transfer Mode	Modem—Full—or half-duplex (serial data) Computer—Multi-cycle data break (parallel data)
Modem Interface	Jumper-selectable for: 1) Bipolar EIA/CCITT (RS232-C) 2) Current Mode; where MARK = 5 ma or less and SPACE = 23 ma or greater 3) TTL compatible
Baud Rate	71,000 bits/second (max)
Character Length	Jumper selectable for 6, 7, or 8 bits
Response Time	Break cycles: 1/Baud rate Program Interrupts: 1/Baud rate * bits/character (one character time)
Character Recognition	Detects four program selected characters. Flag bit (Bit 0) stored with character determines whether program is flagged or character is stripped.
Cycle Time	Single Cycle Data Break—1.4 Micro Seconds All Instructions—1.2 micro seconds.

Carrier Detect	Jumper selection detects carrier/AGC ON and/or OFF transitions.
Control Transfers	Control transfers are maintained via the Data Bus. The types of control available are: Idle, Terminal Ready, Enable, Transmit Request and Transfer Field.
Synchronization Character	Transmit: Non-hardware function; part of data for transmission. Receive: Receive sync code is jumper-selectable. Two or more consecutive sync characters must be detected before hardware is activated.
Clock	Modem timing or tabs for customer-supplied clock

Modem Compatibility (Typical)

Type	Speed (Baud)
Bell 201A	2000
" 203A	2400
" 205B	600,1200 or 1800
" 301B	40,800
" 303B,C	19,000 to 50,000
Rixon FM-12	1200
" Sebit 48	4800
G.E. TDM Series	2400
Lenkurt 26C	120-2400

Additional Features	
Break Priority	Jumper selectable for priorities 0 through 6.
Device Codes	Jumper selectable for using up to four DP8-E modules.
Access Address	Jumper selectable for up to six groups corresponding to DP8-E assignment (up to four active and two spares). Each group or interface module can have up to 16 access or file addresses.

Current Mode Electrical Specifications (Applicable to the Bell 300 Series Modem or equivalent

Receiver Input Current/Voltage levels with 100 ohms Termination
Mark—5 ma ($-0.7 < Eo < 1$)

Driver Output Impedance with Power Off: Not Specified

Driver Output Short Circuit Current: Not Specified

Driver Slew Rate between the 7 ma and the 21 ma levels

Typical 14 ma/100 ns
Max. 14 ma/ 50 ns
Min. 14 ma/200 ns

Receiver Input Impedance

$120 > Zin > 90$

Receiver Output with Open Circuit Input

Logic one—Mark—off

Receiver Output with Input > 23 ma

Logic Zero—Space—On

Receiver Output with Input < 5 ma

Logic ONE—Mark-off

Driver Distortion Limits

Mark to Space or Space to Mark must be achieved within 25% of bit interval.

Receiver Open Circuit Voltage

$-0.8V$ to $-1.3V$

RS-232-C Electrical Specifications

Driver output logic levels with 3K to 7K load

$15 \text{ volts} > V_{oh} > 5V$
$-5 \text{ volts} > V_{ol} > -15V$

Driver output voltage with open circuit

$V_o < 25$ volts

Driver output impedance with power off

$Z_o > 300$ ohms

Output short circuit current

$1_o < 5$ amps

Driver slew rate

$\frac{dv}{dt} < 30$ volts/usec.

Receiver input impedance

$7K \text{ ohms} > R_{in} > 3K \text{ ohms}$

Receiver input voltage

$\pm 15V$ compatible w/driver

Receiver output with open circuit input

Mark

Receiver output with 300 ohms to ground on input

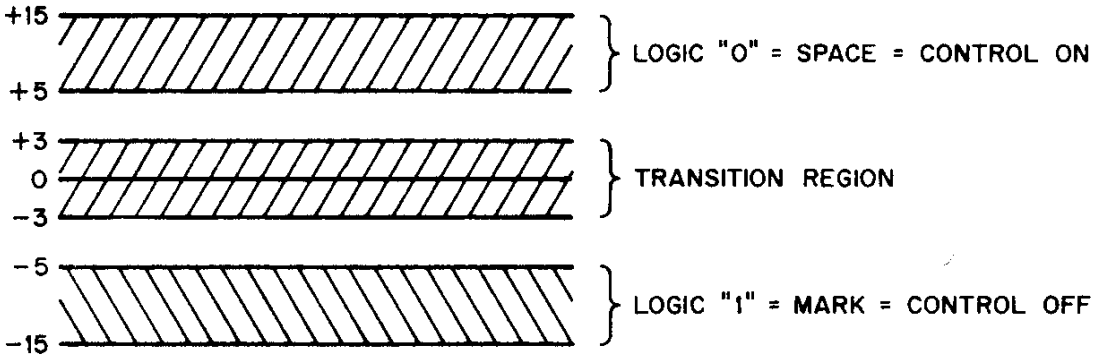
Mark

Receiver output with +3 volt input

Space

Receiver output with −3 volt input

Mark



Programming

The IOT instructions which follow control the DP8-EA/EB. For multiple-channel interfacing, the octal codes listed are used for Channel 1; IOTs containing device codes 42 and 43 are as used for Channel 2, etc., as follows:

Channel	Access Addresses (9 per channel)	IOT Device Codes
1	7720-7730	640x/641x
2	7700-7710	642x/643x
3	7660-7670	644x/645x
4	7640-7650	646x/647x
	*7620-7630	
	*7600-7610	

* These spare access addresses may be used in case of conflict with existing programs.

Access address assignments are determined by low order bits (8-11) as follows:

0000, 0001, 0010, 0011 } Test Characters

0100 Receive Word Count (WC) [2's Complement of Number of Words to be received]

0101 Receive Current Address (CA)

0111 Transmit Word Count (WC)

1000 Transmit Current Address (CA) [2's Complement of Number of Words to be transmitted]

1001, 0110 } *

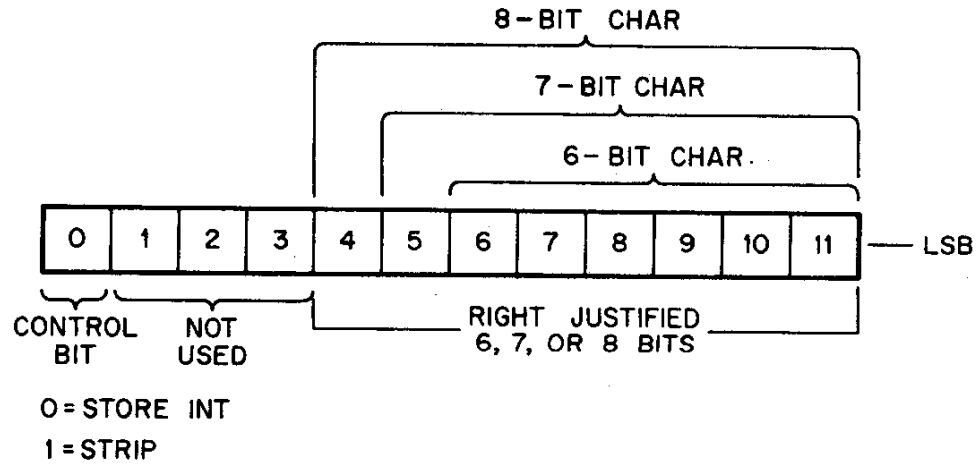
* Access address counter increments to these locations prior to character transfer with the PDP. When the counter is set at 0110, a Received Character has been transferred to a location specified by the Receive Current Address. When the counter is set at 1001, a character for transmit has been transferred to the DP8 from the location specified by the Transmit Current Address.

Test Characters

A Test Character is a vehicle by which the programmer is provided greater control and flexibility over the input/output of data. Four test character locations are available as indicated in the access address assignments.

Test Characters allow the user to identify interesting characters by causing the DP8-E character detected flag to set and thereby cause an interrupt. This can be used with the SRCD instruction which asserts the SKIP line when the Character Detected Flag is set.

The format of the test character is given below. Bit 0 is the control bit that determines if the test character is to be stored or stripped. The least significant bit is bit 11 and the most significant bit is bit 4.



TEST CHARACTER WORD FORMAT

Word Count and Current Address Bits—Word Count and current address are 12 bits wide.
Transmit and Receive Data—the 6, 7, or 8 bit character is right justified. When 6 or 7 bit characters are used, the remaining bits up to 8 should be negated.
Control Word—The AC bits vs. Control are as follows:

AC00 Terminal Rdy
AC01 Idle (1)
AC02 Enable (2)
AC03 Send Rqst
AC04 For customer use (Write only TTL output)
AC05 For customer use (Write only TTL output)

1. If word count goes to zero while in IDLE mode, the Transmit Current address and Word Count will no longer be incremented and access to the last address will continue until the instruction SSTO (Skip on transmit word count o'flow) is assigned or the Idle Bit is negated.

2. If Enable is negated, Interrupt Request, Break in progress and Break Priority Gates are inhibited and the Break Request Flip Flop is latched in the ZERO state.

Character Recognition—Character recognition (detection) is accomplished for 6, 7, or 8 bit characters. The characters must be stored right justified. When 6 or 7 bit characters are used, the remaining bits, up to 8, should be negated.
The stripping or flag generation upon character detection is dependent upon MD00. If MD00 is set to a ONE and the stored character is found to compare with the received character, further memory cycles will be terminated (i.e. the word count and current address will not be incremented and there will be no stored character. If MD00 is a ZERO and there is a character comparison, the character detected flag will be raised, the number of the recognized character will be stored for one character time in a two bit register, and the received character will be transferred to the current address.

Field Selection—The selected field (increments of 4K of core up to 32K) combined with the current address forms a 15 bit address for transfers to and from core.
The field for character transfer is specified by program instruction (SLFL) and the contents of the AC. The field vs. AC assignments are as follows:

AC00, AC01, AC02 — Transmit field (octal 0-7)

AC03, AC04, AC05 — Receive field (octal 0-7)

Character Detected (Reading of)—When the instruction "Read Character Detected (SRCD)" is used to determine what character was detected, two bits, corresponding to the two low-order bits of the Access Address are transferred to AC10 and AC11 as follows:

AC10	AC11	Access Address (Base 2)
0	0	0000
0	1	0001
1	0	0010
1	1	0011

Instructions
All instructions are fully decoded and two device codes are required for an instruction set. Up to four sets of instructions are available and are paired as follows:
640X/641X, 642X/643X, 644X/645X, 646X/647X.

Transmit Go (SGTT)
Octal Code: 6405/6425
6445/6465

Operation:

SGTT sets the Transmit Go Flip Flop. This instruction implies that the program is ready to transmit data (i.e., the Current Address (CA) and Word Count (WC)), have been updated. Upon receipt of this instruction, the hardware will assert the modem Request to Send (RS) lead. When the modem responds with Clear to Send (CS), memory references will begin. Memory references will cease only when WC decrements to Zero (WC → 0). In this event if SGTT is not issued in less than one character time, the transmit line will be maintained at Mark Hold. Transmit Request should be asserted SGTT instruction and should not be cleared until two bit times after the last bit has been transmitted.

Receive Go (SGRR)

Octal Code: 6404/6424
6444/6465

Operation: SGRR sets the Receive Go Flip Flop. This instruction implies that the program is ready to receive data from the communications line, (i.e. the Current Address (CA) and Word Count (WC),) have been updated. The hardware, upon receipt of this instruction, will begin memory references if two consecutive synchronizing characters have been recognized by the hardware on the incoming serial data line. Memory references will cease only when WC decrements to Zero (WC → 0) and SGRR is not issued in less than one character time.

Skip if Character Detected (SSCD)

Octal Code: 6400/6420
6440/6460

Operation: The SSCD Instruction causes the program to skip the next instruction if the character detect flag is a ONE. The character detect flag is a ONE if an assembled character is found to compare one of the stored characters in one of the first four locations of the Access Address. Additionally, the SSCD Instruction clears the character detected flag. If the program is required to identify which of the four stored characters compared to the contents of the Receive Buffer, then a Read Character detected (SRCD Instruction should be utilized. See the SRCD instruction for details).

Clear Sync Detect (SCSD)

Octal Code: 6406/6426
6446/6466

Operation: Clears the "Sync Character Detection" Flip Flops. This instruction enables the programmer to initialize the sync detection circuits and clear the receive registers without initializing the modem interface.

Skip if Receive Word Count Overflow (SSRO)

Octal Code: 6402/6424
6444/6464

Operation: Skips the next instruction and clears the flag if the Receive O'Flow Flag is a ONE. The receive O'Flow Flag is a ONE if during the Receive Data break sequence the Word Count (in core) overflowed.

Skip if Transmit Word Count Overflow (SSTO)

Octal Code: 6403/6423
6443/6463

Operation: Skips the next instruction and clears the flag if the Transmit O'Flow Flag is a ONE. The Transmit O'Flow Flag is a ONE if during the Transmit Data Break sequence the Word Count (in core) overflowed.

Clear Synchronous Interface (SCSI)

Octal Code: 6401/6421
6441/6461

Operation: Initializes all active functions in the synchronous interface.

Read Transfer Address Register (SRTA)

Octal Code: 6407/6427
6447/6467

Operation: Transfers the contents of the transfer address register to AC00-AC11. In use, the Transfer latches the Current Address (CA) prior to incrementing and returning it to core. During Data transfers (transmit or receive) this register then becomes the 8/E's memory address (MA). This instruction is primarily for diagnostic and/or program debug.

Load Control (SLCC)

Octal Code: 6412/6432
6452/6472

Operation: Transfers the contents of AC00-AC05 for selecting Terminal Ready, Idle Mode and Synchronous Interface Enable respectively.
(AC00) *Terminal Ready* permits the modem to enter into the data mode.
(AC01 *Idle Mode* allows a continuous transmission from the same location in core without program intervention. The hardware will enter the Idle Mode when the Word Count goes to ZERO. Further, the transmit current address and Word Count will no longer be incremented and access to the last address will continue until the SGTT Instruction is issued or the Idle Bit is negated.
(AC02) *Interface Enable* allows program interrupts and data break cycles.
(AC03) *Transmit Request* activates the Request to Send line. See SGTT instruction.
(AC04, AC05) are for customer use. When modem timing signals are used one EIA (or current mode) transmitter is available to be used with AC04 or AC05.

Skip if Ring Flag (SSRG)

Octal Code: 6410/6430
6450/6470

Operation: Skips the next instruction and clears the Ring Flag if the Ring Flag is a ONE.

Skip if Carrier/AGC Flag (SSCA)

Octal Code: 6411/6431
6451/6471

Operation: Skips the next instruction and clears the Carrier/AGC Flag if the Flag is in the ONE state. The Carrier/AGC Flag is in the ONE state if the Carrier/AGC line has made an ON and/or OFF transition. The detected transitions are jumper selectable.

Read Status 2 (SRS2)

Octal Code: 6414/6434
6454/6474

Operation: Transfers status to AC00-AC07. This instruction is primarily for diagnostic and/or program debug. The AC vs. Status is as follows:

AC00 Carrier/AGC
AC01 Request to Send
AC02 Terminal Ready
AC03 Clear to Send
AC04 TEMA 0 } Field Select Register
AC05 TEMA 1 } Field Select Register
AC06 TEMA 2 } Field Select Register
AC07 Receive Data (inv.)

Read Status 1 (SRS1)

Octal Code: 6415/6435
6455/6475

Operation: Transfers status to AC00-AC07. This instruction is primarily for diagnostic and/or program debug. The AC vs. Status is as follows:

AC00 R-RQST Receive and Transmit
AC01 T-RQST Break Requests
AC02 Sync 2 Received "Sync"
AC03 Sync 1 Characters
AC04 REMA 0 } Field Select Register
AC05 REMA 1 } Field Select Register
AC06 REMA 2 } Field Select Register
AC07 Modem Ready

Load Field (SLFL)

Octal Code: 6413/6433
6453/6473

Operation: Transfers the contents of AC00-AC05 to the field select registers. AC00-AC02 selects the transmit field while AC03-AC05 selects the Receive Field. The selected field (increments of 4K of core—up to 32K) combined with the current address forms a 15 bit address for data transfers to and from core.

Skip on Bus Error (SSBE)

Octal Code: 6416/6436
6456/6476

Operation: Skips the next instruction and clears the Bus Error Flag if the flag was in the ONE state. The Bus Error Flag will be in the ONE state if a Transmit or Receive Break Request has not been serviced in less than 1/BAUD time. This flag implies that the Break bus is either overloaded or is inoperative.

Read Character Detected (SRCD)

Octal Code: 6417/6437
6457/6477

Operation: The contents of a two bit register which contains the address of the detected character is transferred to AC10 and AC11. The two bits correspond to the two low order bits of the access address where the characters for detection are stored.

Maintenance Instruction

The SRCD instruction issued when AC00 is set to a ONE causes a single clock pulse on the *External Clock* or secondary Transmit data line (circuit SBA) Jumper selectable line to the modem.

Summary of Instructions

CODE	MNEMONIC	INSTRUCTION
6400/6420/6440/6460	SSCD	Skip if character detected
6401/6421/6441/6461	SCSI	Clear Synchronous Interface
6402/6422/6442/6462	SSRO	Skip if Receive Word Count O'Flow
6403/6423/6443/6463	SSTO	Skip if Transmit Word Count O'Flow
6404/6424/6444/6464	SGRR	Receive Go
6405/6425/6445/6465	SGTT	Transmit Go
6406/6426/6446/6466	SCSD	Clear Sync Detect
6407/6427/6447/6467	SRTA	Read Transfer Address Register
6410/6430/6450/6470	SSRG	Skip if Ring Flag
6411/6431/6451/6471	SSCA	Skip if Carrier/AGC Flag
6412/6432/6452/6472	SLCC	Load Control
6413/6433/6453/6473	SLFL	Load Field
6414/6434/6454/6474	SRS2	Read Status 2
		AC00 Carrier/AGC
		AC01 Request to Send
		AC02 Terminal Ready
		AC03 Clear to Send
		AC04 TEMA 0
		AC05 TEMA 1
		AC06 TEMA 2
		AC07 Receive Data (Inv.)
6415/6435/6455/6475	SRS1	Read Status 1
		AC00 R-RQST
		AC01 T-RQST
		AC02 SYNC 2
		AC03 SYNC 1
		AC04 REMA 0
		AC05 REMA 1
		AC06 REMA 2
		AC07 Modem Ready
6416/6436/1656/1676	SSBE	Skip on Bus Error
6417/6437/6457/6477	SRCD	Read Character Detected Low Order Bits (Access Address) AC10 and AC11
6417/6437/6457/6477 with AC00—ONE	——	Test Clock

Interfacing

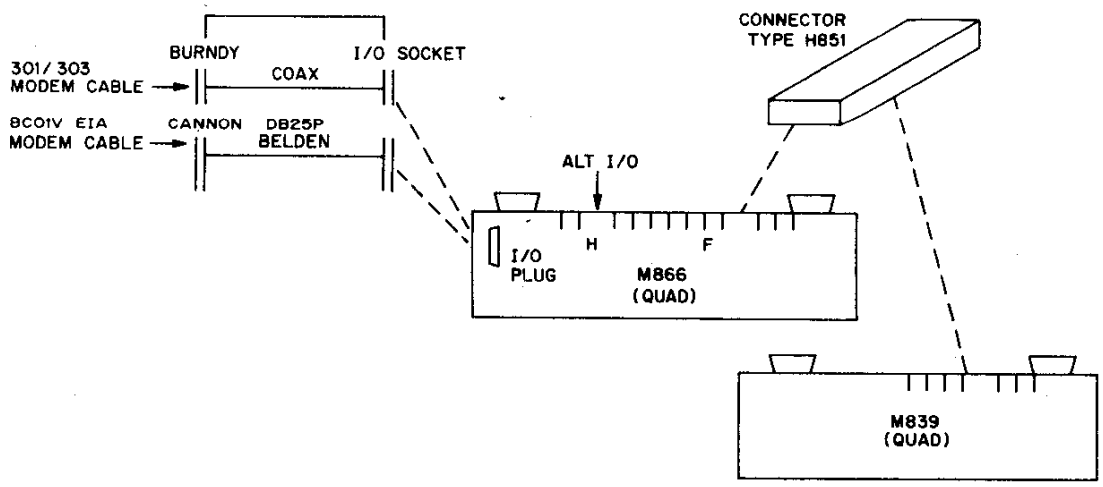
DP8-E Terminated Modem Leads—The following chart shows the modem control leads for models 201, 301 and 303 as used in the DP8-E. Unless otherwise specified the 201 levels are Bi-polar levels while the 301 and 303 are current mode.

Logic Print	Model 301 (EB)	Model 303 (EB)	Model 201 (EA)
Send Data	Send Data	Send Data	Send Data
Received Data	Received Data	Receive Data	Receive Data
Clear to Send	Clear to Send	Clear to Send	Clear to Send
Interlock/Data Set Ready	Interlock	Data Set Ready	Interlock
Carrier/AGC	Carrier On-Off	AGC Lock	Carrier on-off
Serial Clock Transmit	Serial Clock Transmit	Serial Clock Transmit	Serial Clock Transmit
Serial Clock Receive	Serial Clock Receive	Serial Clock Receive	Serial Clock Receive
Terminal Ready		Data Terminal Rdy*	Remote Control
Ring		Ring Indicator*	Ring Indicator 1
External Timing	Serial Clock Transmit (External)	Serial Clock Transmit (External)	External Timing

* Bi-polar

Interface Connections—The DP8-E is interfaced to a modem according to the following:

DP8-EA → EIA (Assembly → M839 + M866 + BC01V-25
DP8-EB → 301/303 (Assembly → M839 + M866 + BC01W-25



Top Connector and I/O Pin Assignments

M839 M866 "F" Connector

NAME	SOURCE		Pin
	M839	M866	
SRCD (Low)		*	FA1
This Code (Inverted)	*		FB1
INT RQST	*		FC1
REC DATA (Mark +3)		*	FD1
REMA (Inverted)	*		FE1
TEMA (Inverted)	*		FF1
SCR-P (Inverted)		*	FH1
BREAK GRANT		*	FJ1
INITIALIZE (Inverted)	*		FK1
GTP4		*	FL1
BREAK RQST		*	FM1
SSBE (Inverted)		*	FN1
SCRT (Inverted)		*	FP1
CS		*	FR1
SD	*		FS1
IDLE		*	FT1
RQST	*		FU1
SRS1	*		FV1
Ground			FA2-V2
Spare			None

The BC01 V (Edge to EIA) and BC01 w(Edge to 301/303) adapter cables perform the following:

I/O CONNECTORS

NAME	Edge	EIA	301	303
Signal Ground	VV	7	All	(note 1)
Frame Ground	B	1	Shields	
Clear to Send	T	5	C	C
Receive Data	J	3	K	K
Interlock/Data Set Ready	Z	6	F	F
Serial Clock XMIT	N	15	J	J
Serial Clock Receive	R	17	L	L
Carrier/AGC	BB	8	M	M
Ring	X	22	—	F(outer)
Send Data	F	2	E	E
Terminal Ready	DD	20	—	M(outer)
Send Request	V	4	D	D
External Timing	L	24	H	H
-6 Volts	—	—	—	—
+6.4 Volts	—	—	—	—
External Clock	L	—	—	—
SEC Transmit Data	FF	—	—	—
SEC Receive Data	JJ	—	—	—

NOTE: 303 Modem connectors F + M shields (outer connector) carry EIA signals as indicated.

Jumper Selection

Unless otherwise specified, the following selections will be provided:

a. DP8-E A, B	ACCESS ADDRESS	IOT CODES	BREAK PRIORITY
1st	7720-7730	640X/641X	5
2nd	7700-7710	642X/643X	4
3rd	7660-7670	644X/645X	3
4th	7640-7650	646X/647X	2

b. 8 Bits/Character
c. Normal Clock Phase
d. Level conversion for DP8-EA will be EIA
e. Level conversion for DP-8EB will be current mode.
f. Sync code will be 226 (Octal)
g. Carrier on/off transister
h. Full Duplex

To alter selection:

M839 Jumpers	Select	Remove Jumper
Bits/Character	6	B8, B7, B78, C7, C8
	7	B8, B6, C8
	8	B7, B6, C7
	1	P2, P3, P4, P5, P6, P7
	2	P1, P3, P4, P5, P6, P7
	3	P1, P2, P4, P5, P6, P7
	4	P1, P2, P3, P5, P6, P7
	5	P1, P2, P3, P4, P6, P7
	6	P1, P2, P3, P4, P5, P7
	7	P1, P2, P3, P4, P5, P6
Break Priority (Generate)	7720(1)	A6
	7700(2)	A6, A7
	7660(3)	A5
	7640(4)	A5, A7
	7620(5)	A5, A6
	7600(6)	A5, A6, A7
Access Address	640X/641X(1)	6, 7
	642X/643X(2)	6, N7
	644X/645X(3)	N6, 7
	646X/647X(4)	N6, N7
Device Code	226	S5, S6, S8, S11
Sync Code (Remove jumper S4-S7 to Select Zero)		

NOTES: (1) 1st DP8 (2) 2nd DP8 (3) 3rd DP8
(4) 4th DP8 (5) Spare

M866 Jumpers*

*Jumpers are production inserted with the exception of: C and △

	Select	Remove Jumper	Add Jumper
CO/AGC Transition	ON	OFF	
	OFF	ON	
	ON & OFF	OFF & ON	
Clock Phase	Inverted	N(TWO)	△(TWO)
Level Conversion	EIA	T, CT	
	Current	T, E	C(Six)
	TTL	E, CE	
Break Priority (Detect)	1	P1—P6	
	2	P2—P6	
	3	P3—P6	
	4	P4—P6	
	5	P5—P6	
	6	P6	
	7		
Full Duplex		HD	

EIA RS-232-C Interface Pin Assignments

Pin Number	Circuit	Description
1	AA	Protective Ground
2	BA	Transmitted Data
3	BB	Received Data
4	CA	Request to Send
5	CB	Clear to Send
6	CC	Data Set Ready
7	AB	Signal Ground (Common Return)
8	CF	Received Line Signal Detector
9	—	(Reserved for Data Set Testing)
10	—	(Reserved for Data Set Testing)
11	—	Unassigned
12	SCF	Sec. Rec'd Line Sig. Detector
13	SCB	Sec. Clear to Send
14	SBA	Secondary Transmitted Data
15	DB	Transm. Signal Element Timing (DCE Source)
16	SBB	Secondary Received Data
17	DD	Received Signal Element Timing (DCE Source)
18		Unassigned
19	SCA	Secondary Request to Send
20	CD	Data Terminal Ready
21	CG	Signal Quality Detector
22	CE	Ring Indicator
23	CH/CI	Data Signal Rate Selector (DTE/DCE Source)
24	DA	Transmit Signal Element Timing (DTE/DCE Source)
25		Unassigned

EIA (RS-232-C) to Equivalent CCITT

Inter-change Circuit	CCITT Equivalent	Description
AA	101	Protective Ground
AB	102	Signal Ground/Common Return
BA	103	Transmitted Data
BB	104	Received Data
CA	105	Request to Send
CB	106	Clear to Send
CC	107	Data Set Ready
CD	108.2	Data Terminal Ready
CE	125	Ring Indicator
CF	109	Received Line Signal Detector
CG	110	Signal Quality Detector
CH	111	Data Signal Rate Selector (DTE)
CI	112	Data Signal Rate Selector (DCE)
DA	113	Transmitter Signal Element Timing (TDE)
DB	114	Transmitter Signal Element Timing (DCE)
DD	115	Receiver Signal Element Timing (DCE)
SBA	118	Secondary Transmitted Data
SBB	119	Secondary Received Data
SCA	120	Secondary Request to Send
SCB	121	Secondary Clear to Send
SCF	122	Sec. Rec'd Line Signal Detector

DP8-EP Redundancy Check Option

The DP8-EP redundancy check option is designed to complement the DP8-EA and DP8-EB synchronous interface by providing parity generation and checking facilities. Vertical redundancy checks (VRC), logitudinal redundancy checks (LRC), and cycilic redundancy checks (CRC) can be performed by this option. The cycilic redundancy check is industry compatible CRC-12 and CRC-16.

The DP8-EP operates directly with the PDP-8/E from program-controlled instructions. Thus, when not used with the communications equipment, it can be used with other devices. The DP8-EP consists primarily of MSI logic packaged on a single PDP-8/E module, which plugs into the OMNIBUS. All control functions and character options are programmable. The primary purpose of the DP8-EP parity option is to reduce processor overhead for data communications applications where character parity (VRC) and/or Block Check Character (BCC) Accumulation (LRC or CRC) are required for error detection. The types of parity generation or checks that the DP8-EP can perform are defined below:

a. Vertical Redundancy—Parity is on a character basis where one bit slot of each character is reserved for the parity bit. Odd parity (odd number of binary ones) is generated by this option; however, capabilities are provided for checking odd or even parity.

b. Longitudinal Redundancy—This type is a BCC accumulation over a block of message characters; that is, the LRC is an accumulated EXCLUSIVE OR of all character bits (including parity

bits) in a message. This method is more reliable than the VRC in detecting errors. A system may use both the LRC and VRC to increase the probability of detecting errors. Both the transmitting and receiving station must compute the BCC; at the end of the message block, the BCC Accumulations are compared at the receiving stations. If they are equal, the message is assumed to be without error.

c. Cyclic Redundancy—As implemented in this option, is industry compatible for CRC BCC accumulation (CRC16/12).

The CRC check sum is the remainder derived from dividing the numerical value of the message by a constant. The division is performed serially, the quotient is discarded, and the remainder is stored. Both the transmitting and receiving stations must compute the BCC accumulation. At the end of each message block, the BCC accumulation is sent to the receiving station for comparison with the receive station check sum. If the two are equal, the message is assumed to be without error. CRC and VRC operations can be combined to increase the probability of error detection.

Specifications

Vertical Redundancy Check (VRC)	Tests or computes odd parity for up to eight-bit characters. Parity bit is either right-justified (AC11) or left-justified (AC04).
Longitudinal Redundancy	Computes or compares BCC accumulation for 6, 7, 8, 12 or 16-bit characters. Two bytes required for LRC 16.
Cyclic Redundancy Check (CRC)	Industry compatible for CRC-12 and CRC-16. Division constants used are: $X^{12} + X^{11} + X^3 + X^2 + X^1$ for CRC 12, and $X^{16} + X^{15} + X^2 + 1$ for CRC-16 where X is modulo 2.
Cycle Times	VRC (compute): 1.4 μs (test) : 1.2 μs CRC or LRC : 1.2 μs CRC and VRC (compute): 1.4 μs CRC and VRC (test) : 1.2 μs LRC and VRC (compute): 1.4 μs LRC and VRC (test) : 1.2 μs

Programming

The instructions associated with the DP8-EP option are as follows:

Compute VRC (RCCV)

Octal Code: *6113*

Operation: Transfers character in AC4-11 to DP8-EP parity register, Clears AC and generates odd vertical parity. Result is then jam-transferred to AC with parity bit in AC04 or AC11 as defined by the program.

Test VRC and Skip (RCTV)
Octal Code: 6110
Operation: Checks parity of character in AC4-11. For odd parity, the next instruction is skipped if parity of character is odd.

Generate BCC (RCGB)
Octal Code: 6114
Operation: Generates an LRC or CRC Block Check Character (BCC). The LRC can be generated from 6, 7, 8, 12, or 16 (two six-bit bytes) bit characters, while CRC 12/16 can be computed from 6 to 8-bit characters, respectively. BCC verification: The transmitted BCC is compared to the Receive BCC by treating the BCC as part of the overall accumulation. In doing so the receive BCC generator will go to zero if there were no errors in transmission.
This instruction also provides the functions defined for RCCV and RCTV if the appropriate control bits are included. (see RCLC instruction).

READ BCC LOW (RLRL)
Octal Code: 6112
Operation: Jam-transfers the 6, 7, 8, or 12 LSBs of BCC accumulation to the AC (right-justified). The quantity of bits transferred to the AC is dependent on the BCC length selected with the RCLC instruction. The LSB of each byte is also right-justified.

Read BCC High (RCRH)
Octal Code: 6111
Operation: Jam-transfers the 8 MSBs of BCC accumulation to AC (right-justified). The instruction is used for the 16-bit BCC. The LSB of each byte is also right-justified.

Clear BCC Accumulation (RCCB)
Octal Code: 6116
Operation: Clears the 16-bit BCC register.

Load Control (RCLC)
Octal Code: 6115
Operation: Jam-transfers content of AC to redundancy control register to define the operation as follows:

AC05 = 1: CRC BCC
0: LRC BCC

AC 6 7 8

0 0 0 = 16-bit BCC
0 0 1 = 12-bit BCC
0 1 0 = 8-bit BCC
0 1 1 = 7-bit BCC
1 0 0 = 6-bit BCC

AC 9 = 0: Generated parity to AC4
= 1: Generated parity to AC11

AC10 = 1: An RCGB instruction also causes a RCCV instruction sequence to accrue. The BCC accumulation will be computed with the corrected character parity.

AC11 = 1: An RCGB instruction also causes a RCTV instruction sequence to accrue.

Maintenance Test Clock (RCTC)

Octal Code: 6117

Operation: This instruction can only be implemented by grounding test point DA1 on the module. RCTC causes a single clock pulse to the registers, permitting single step testing of LRC and CRC operations.

Interface to Bell 201 Modems

The DP8-EA Synchronous Data Interface module is connected to a Bell Model 201 modem (or equivalent) by a 25-ft cable terminated at the modem end with a 25-pin male connector. Standard interface signals are bipolar (EIA/CCITT); however, current mode or TTL compatible signals can be selected using jumper options on the DP8-EA. Interface signals versus connector pin assignments are provided in Table 7-1. In addition, signal or protective ground is provided on pin 1. Signal ground is provided on pin 7.

Table 7-1
Connector Pin Assignments for Bell Series 201

Pin	Signal
2	Send Data
3	Receive Data
4	Send Request
5	Clear to send
6	Interlock
8	Carrier on-off
15	Serial Clock transmit
17	Serial clock Receive
20	Remote control
22	Ring indicator 1
24	External timing

KL8-E Asynchronous Data Control

The KL8-E control unit is a PDP-8/E module which plugs into the OMNIBUS and controls the operation of a Teletype or other similar asynchronous devices from the programmed instructions. This module contains the shift clock, the control logic for IOT decoding, parallel-to-serial and serial-to-parallel converters, and program control of interrupt and flag facilities. Serial information read or written by the Teletype unit is assembled or disassembled by the KL8-E control for transfer between the Teletype and the AC.

For program operation, the Teletype unit and control are considered as a Teletype in (TTI) for input intelligence from the keyboard or the perforated-tape reader, and as a Teletype Out (TTO) for computer output information to be printed and/or punched on tape. Therefore, two device select codes are used; select code 03 initiates operations associated with the keyboard/reader (TTI), and select code 04 performs operations associated with the teleprinter/punch (TTO). The control unit contains a programmable interrupt enable flip-flop that is common to both the keyboard/reader and teleprinter/punch. This flip-flop is set when power is turned on or INITIALIZE is generated, and can be set or cleared (as specified by AC11) using the KIE instruction. If AC11 is a 1 when the KIE instruction is issued, the interrupt enable flip-flop is set to permit the generation of interrupt requests whenever the keyboard/reader flag or teleprinter/punch flag is set. In contrast, if AC11 is a 0 when KIE is issued, no interrupt can be generated by this control unit. Functions performed by the keyboard/reader and teleprinter/punch are described in subsquent paragraphs.

Specifications

(Also see KL8-EA—KL8-EG)

Interface	20 ma current loop operation. Reader controlled by reader enable leads
Character Parameters	8 data bits and 1 or 2 (standard) stop units
Baud Rate (Standard)	110 (Other rates available upon special order)
Binary Input/Output	8-bit parallel
Other features	1) Programmable interrupt enable 2) Program control for clearing keyboard flag without setting reader run flip-flop. 3) Program control for setting teleprinter flag. 4) Jumper-selectable device codes 5) Input to 17 KL8-E's per PDP-8/E

Keyboard/Reader

The keyboard and tape reader control contains an eight-bit shift register (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 TTI register. The Teletype character code is loaded into the TTI so that spaces (the absence of holes) correspond with binary 0s and holes (marks) correspond to binary 1s. Upon program command, the contents of the TTI are transferred in parallel to the AC.

When a Teletype character is to be read from the paper tape reader, 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.

When an eight-bit character has been assembled in the TTI, the keyboard flag is set to cause a program interrupt if the interrupt enable flip-flop has been set. When the program services the interrupt, it senses the flag with a KSF instruction and, with the flag set, issues a KRB instruction which clears the AC, clears the keyboard flag, transfers the contents of the TTI into the AC, and enables advance of the tape feed mechanism.

Programming

The following instructions are used for supplying data to the computer from the keyboard/reader:

Clear Keyboard Flag (KCF)

Octal Code: 6030
Operation: Clears the keyboard flag without setting the reader run flip-flop. The AC is not cleared by this instruction.

Skip on Keyboard Flag (KSF)

Octal Code: 6031
Operation: Increments the contents of the PC if the keyboard flag is set, so that the next instruction is skipped.

Clear Keyboard Flag (KCC)

Octal Code: 6032
Operation: Clears the keyboard flag and AC and sets the reader run flip-flop. This action allows the hardware to begin assembling the next input character in the TTI register. If the reader is activated and there is tape in the reader, a serial character is read from tape and is assembled in the TTI register. The keyboard can also load characters into the TTI register provided the reader is deactivated. In either case, the keyboard flag is set when the character is assembled on the TTI.

Read Keyboard Buffer Static (KRS)

Octal Code: 6034
Operation: ORs the contents of TTI register with AC4 through 11, and leaves the result in AC4-11. This is termed a static command because neither the AC nor keyboard flag is cleared.

Set/Clear Interrupt Enable (KIE)

Octal Code: 6035
Operation: Sets or clears the interrupt enable flip-flop as defined by AC11. If AC11 is asserted, generates an interrupt request for a keyboard or teleprinter flag. If AC11 is negated, interrupt requests cannot be generated.

Read Keyboard Buffer Dynamic (KRB)

Octal Code: 6036

Operation: Performs the combined operations of the KCC and KRS instructions. Clears the AC and keyboard flag and transfers the contents of the TTI register to AC4 through AC11. This instruction also sets the reader run flip-flop to begin assembly of another character in the TTI register. When this operation is complete, the keyboard flag is set to indicate another character is available.

A typical TTI instruction sequence for keyboard (manual) input is:

```
LOOK, KSF          /SKIP IF KEYBOARD FLAGS
      JMP LOOK     /JUMP 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 transferred from the TTI to the AC.

The computer clears all flags which are on the clear flag bus (including both the keyboard flag and the reader run enable) when the console CLEAR pushbutton is depressed. This means that the user program must set the reader run enable to obtain data from the reader. The instruction sequence given below is a typical TTI instruction sequence for both keyboard and reader input.

If this sequence of instructions is made a subroutine of the main program, it can be accessed each time an input character is desired. Consequently,

```
      KCC            /CLEAR TTI FLAG, SET READER RUN CLEAR/AC
      .
      .
      .
READ, 0              /STORE PC HERE FOR RETURN ADDRESS
      KSF            /SKIP IF FLAG = 1
      JMP .—1        /TEST FLAG AGAIN
      KRB            /READ CHAR INTO AC
      JMP 1 READ     /EXIT TO MAIN PROGRAM
      .
      .
      .
```

Teleprinter/Punch

On program command a character is transferred from the AC to the output shift register (TTO) for transmission to the teleprinter/punch unit. The Teletype control generates the start space, shifts the eight character bits serially into the printer selector magnet of the Teletype unit, and then generates the stop marks. Bit transfer from the TTO to the teleprinter punch unit is at the normal Teletype rate. A character transfer requires 100 ms for completion at 110 baud. The teleprinter flag is set when the last bit of the character code is sent to the teleprinter/punch, indicating that the TTO is ready to receive a new character from the AC.

The flag activates the program interrupt synchronization element and the instruction skip element. When using instruction skip, the program checks the flag by means of the TSF instruction. If the flag is set, the program issues the TLS instruction, which clears the flag and sends a new character from the AC to the TTO. AC-to-TTO transfer time is short compared to the print/punch time, so the program must wait for the flag to set before issuing another TLS.

Programming

Instructions for use in outputting teletype data are as follows:

Set Teleprinter Flag (TFL)

Octal Code: 6040
Operation: Sets the teleprinter flag to ready the logic for another character.

Skip on Teleprinter Flag (TSF)

Octal Code: 6041
Operation: If the teleprinter flag is set, increments the contents of the PC by one so that the next sequential instruction is skipped.

Clear Teleprinter Flag (TCF)

Octal Code: 6042
Operation: Clears the teleprinter flag. This instruction can be micro-programmed with TPC.

Load Teleprinter and Print (TCP)

Octal Code: 6044
Operation: Transfers AC bits 4-11 to the TTO and starts shifting the character out to the printer/punch unit. This instruction does not clear the teleprinter flag. This instruction can be micro-programmed with TCF to produce TLS.

Skip On Printer or Keyboard Flag (TSK)

Octal Code: 6045
Operation: Skips the next instruction if the printer or keyboard flag is set and the interrupt enable flip-flop is set.

Load Teleprinter Sequence (TLS)

Octal Code: 6046
Operation: This instruction combines TCF and TPC. The teleprinter flag is cleared and the contents of AC bits 4-11 are transferred to the TTO, where the hardware shifts the character out to the printer/punch unit. When the TTO has finished outputting the character and is ready for another character, the teleprinter flag is set. The whole operation, from the time at which the TLS has cleared the flag and TTO starts character transfer, until the time the hardware finishes with the character and again sets the flag, requires 100 ms at 110 baud.

A typical TTO instruction sequence is:

```
                .
                .
                .
                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
                .
                .
                .
```

This sequence sends one character to the TTO and waits for printing/punching before sending another character. It does not require that the flag be set to output the character. By making the instruction sequence 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
        .
        .
```

By rearranging this subroutine, the 100 ms (at 110 baud) spent waiting for the character to be output and the flag to be set is used to continue the main program, making more efficient use of program time.

```
TYPE,   0           /TEST FLAG TO SEE IF TELEPRINTER FREE,
        TSF         /SKIP IF YES OR . . .
        JMP .—1     /WAIT TILL IT IS BY TESTING AGAIN AND
                    /AGAIN
        TLS         /OUTPUT CHARACTER
        CLA         /CLEAR CHARACTER FROM AC
        JMP I TYPE  /EXIT TO CONTINUE PROGRAM
```

This subroutine tests the flag first, and waits only if a previous character is still being outputted. It clears the AC, exits immediately after sending the character to the TTO, and continues to run the user's program, instead of waiting while the Teletype (a much slower I/O device) is typing/punching the preceding character.

The computer clears all flags which are on the clear flag bus (including teleprinter flags) when the console CLEAR pushbutton is depressed. This means that the user program must account for setting the teleprinter flag initially and after each TCF (if any), or the program hangs up in the wait loop of the print routine. The only way to set the flag

is by initializing it. 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 as the first instruction of the program and makes optimum use of the punch/print time.

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

Asynchronous Data Controls KL8-EA through KL8-EG

In addition to the KL8/E Asynchronous Data Control described above, the following opotions are available:

KL8-EA 110 baud, EIA data lead interface
KL8-EB 150 baud, EIA data lead interface
KL8-EC 300 baud, EIA data lead interface
KL8-ED 600 baud, EIA data lead interface
KL8-EE 1200 baud, EIA data lead interface
KL8-EF 1200 baud transmit, 150 baud receive, EIA data lead interface
KL8-EG 2400 baud transmit, 150 baud receive, EIA data lead interface

These options are programmed identically to the KL8-E previously described, and like the KL8-E have jumper selectable device codes so that a number of asynchronous data terminals (not to exceed 17) may be connected to a PDP-8/E merely by adding the necessary KL8-units. Each unit requires two device codes.

The EIA data lead interface conditions the transmitted and received data leads to the requirements of EIA specification RS-232C and CCITT Recommendation V24. These leads, along with Data Terminal Ready and Request to Send (both of which are held in the asserted state), are brought out in a standard 25 pin male connector suitable for direct connection to a modem. The modem used should be a full duplex private (non-switched) line modem such as the Bell System 103F or a switched network modem used in manual mode such as the Bell System 103A without automatic answering. Since the KL8 Asynchronous Data Controls do not provide program control of the modem interface leads, use of these controls in automatic originating or automatic answering applications is not recommended.

DEC offers the CR8-E Card Reader and Control Option and the CM8-E Optical Mark Card Reader and Control Option.

CARD READER OPTIONS

Type CR8-E Card Reader and Control

The CR8-E Card Reader option equips the PDP-8/E computer to accept input from EIA standard data cards. It reads 12-row, 80-column punched cards at a nominal rate of 200 cards per minute photoelectrically. The control circuit for this device is located on a single PDP-8/E module, which plugs into the OMNIBUS. The card reader has an internal power supply and can be tested off-line. For table space requirements, please refer to the specification section which follows.

A select instruction starts a card moving through the read station, where all 80 columns are read on a column-by-column basis, beginning with column one. Card data may be read in any one of three modes. In the binary reading mode, the data is transferred directly from the rows of the card to bits in the AC. The top row of the card (row 12) goes into AC0 and the bottom row (row 9) goes into AC11. In the alphanumeric reading mode, the data is automatically decoded into a six-bit BCD representation and transferred into the least significant six bits of the accumulator. Use of the six-bit decoding minimizes the size of translation tables and is fully compatible with the Hollerith code as used at this time. A proposed expansion of the Hollerith code would require use of the compressed reading mode. In this mode, rows 9, 12, 11, 0, and 8 are transferred directly to AC4, AC5, AC6, AC7, and AC8, respectively, while rows 1 through 7 are decoded into three-bit BCD representation in AC9, AC10, and AC11. This decoding is based on the lack of double punches in rows 1 through 7, both in the present Hollerith and the proposed extension of Hollerith. If such a double punch is read in the compressed reading mode, the CR8-E validity checking circuitry will assert a one in AC0 (the sign bit). Regardless of the reading mode being used, a punched hole is interpreted as a binary one and the absence of a hole is binary zero.

Four program flags indicate card reader conditions to the computer. (The status of these flags may be examined by means of the RCNI instruction.) The Data Ready Flag sets, requesting a program interrupt, when a column of information is ready to be transferred into the AC. A read instruction (alphanumeric, binary, or compressed) must be issued within 1.0 ms after the Data Ready Flag sets in order to avoid data loss. The Card Done Flag sets, requesting a program interrupt, when the card leaves the read station. A new select instruction must be issued immediately after the Card Done Flag sets to keep the reader operating at rated speed.

The Ready True Transition Flag sets, requesting a program interrupt, whenever the ready lead from the card reader to the control goes true, indicating that the card reader is ready. This feature permits the computer program to perform other tasks while awaiting manual intervention to clear a card reader problem such as lack of cards. The interrupt will notify the computer when the card reader is ready to resume reading cards. The fourth flag, the Trouble Transition Flag, sets, requesting a program interrupt, whenever the ready lead from the card reader to the control goes false, indicating an error condition in the card reader. (Error condition when used here refers to a transport error, not a data error

such as an improper validity check.) This flag is cleared by initialize or by means of the Clear Transition Flags instruction.

Specifications

Size:	18 in. high; 14 in. wide; and 18 in. deep.
Weight:	52 lb.
Card Rate:	200 per minute
Input Power:	115 Vac + or − 10 Vac, 60 Hz + or − 5 Hz, single phase, 300W maximum (50 Hz unit available)
Card Specification:	The card reader is designed to read 7⅜ in. x 3¼ in. cards conforming to the material and size requirements of EIA Standard RS-292 Media 1.
Card Capacity:	Both input hopper and output stacker hold 400 cards. Cards may be added or removed during reader operation.

Programming

The following instructions are used with the CR8-E option:

Skip on Data Ready (RCSF)

Octal Code: 6631
Operation: Senses the status of the data ready flag; if it is set (indicating that information for one card column is ready to be read), the contents of the PC are incremented by one, so that the next sequential instruction is skipped.

Read Alphanumeric (RCRA)

Octal Code: 6632
Operation: Transfers the six-bit Hollerith code for the 12 bits of a card column into AC6-11, and clears the Data Ready Flag. This instruction does not detect illegal characters.

Read Binary (RCRB)

Octal Code: 6634
Operation: Transfers the 12-bit binary code for a card column directly into the AC, and clears the Data Ready Flag. Information from the card column is transferred into the AC so that card rows 12, 11, and 0 enter AC0-2 and card rows 1 through 9 enter AC3-11, respectively.

Read Conditions Out to Card Reader (RCNO)

Octal Code: 6635
Operation: Reads AC10 into a Ready True Transition/Trouble Transition interrupt enable flip-flop. If AC10 is a 1, this flip-flop is set, enabling the generation of an interrupt whenever the reader goes from not ready to ready or from ready to not ready. This flip-flop is cleared when the PDP-8/E is initialized. The RCNO instruction also reads AC11 into a flip-flop which, when set by AC11 being a 1, enables the generation of an interrupt whenever the card done or data

ready flags are raised. For program compatibility with other family-of-eight computers, initializing the PDP-8/E sets the card done/data ready interrupt enable.

Read Compressed (RCRC)

Octal Code: 6636
Operation: Transfers an eight-bit compressed code for the 12 bits of a card column into AC4-11, and clears the data ready flag. Data from row 9 goes to AC4, zones 12, 11, and 10 to AC5, 6, and 7 respectively, and data from row 8 goes to AC8. Data from rows 1 through 7 is compressed into a BCD representation in AC9, 10, and 11. Should there be more than one bit of data in rows 1 through 7 (an invalid condition), hardware validity check circuitry will read a 1 into AC0 (sign bit).

Read Conditions in from Card Reader (RCNI)

Octal Code: 6637
Operation: Status of Ready True Transition Flag, Trouble Transition Flag, Card Done Flag, and Data Ready Flag is read into AC3, AC2, AC1, and AC0 respectively.

Skip on Card Done Flag (RCSD)

Octal Code: 6671
Operation: Senses the status of the card done flag; if it is set (indicating that the card has passed the read station), the contents of the PC are incremented, skipping the next instruction.

Select Card Reader and Skip if Ready (RCSE)

Octal Code: 6672
Operation: Senses the status of the card reader; if it is ready, the contents of the PC are incremented, skipping the next sequential instruction, a card is started toward the read station from the input hopper, and the Card Done Flag is cleared.

Clear Card Done Flag (RCRD)

Octal Code: 6674
Operation: Clears the Card Done Flag. This instruction allows a program to stop reading at any point in the card deck.

Skip if Interrupt Being Generated (RCSI)

Octal Code: 6675
Operation: Senses the status of all flags. If a flag is raised and the generation of interrupts by that flag is enabled, the next sequential instruction is skipped.

Clear Transition Flags (RCTF)

Octal Code: 6677
Operation: Clears the Trouble Transition Flag and the Ready True Transition Flag.

Example Subroutine
A logical instruction sequence to read cards is the following:

```
START,  RCSE            /START CARD MOTION AND SKIP IF
                        /READY
        JMP NOTRDY      /JUMP TO SUBROUTINE THAT TYPES
                        /OUT "CARD READER MANUAL INTER-
                        /VENTION 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 FLAG
        DCA I 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
```

The CR8-E performs validity checking only when using the RCRC instruction. A programmed validity check can also be performed by reading each card column in both the alphanumeric and binary modes (within the 1.0 ms time limitation), and then making a comparison check.

Controls and Indicators
Before commencing a card reading program, load the input hopper with cards and press MOTOR START and READ START pushbuttons. The functions of the manual controls and indicators, as they appear from left to right, are as follows:

CONTROL OR INDICATOR	FUNCTION
ON/OFF	Toggle power switch. Applies power to all circuits except the drive motor.
MOTOR START	Momentary action pushbutton with separate indicator. Applies power to the main drive motor. This is also used as a reset to clear error indicators, and, therefore, will not operate if there is an unremedied condition such as: a. Empty input hopper b. Full output hopper c. All photo cells not lit d. Internal power supply not operational.
READ START	Momentary action pushbutton with separate indicator. Causes ready line to go

CONTROL OR INDICATOR	FUNCTION
	high, enabling card reading under control of the external read instructions. If the read instruction is being given, or the control cable is disconnected (off-line testing), card reading begins immediately at full reading speed.
READ STOP	Momentary action pushbutton with separate indicator. Inhibits further card reading until READ START is pressed again. Ready line goes low, and read stop condition is indicated. This signal does not stop the drive motor. However, a read stop condition is indicated anytime the drive motor is stopped.

CONTROL OR INDICATOR	FUNCTION
INDICATORS	
1. PICK FAIL	Lights when a card fails to enter the read station after two successive pick attempts.
2. DARK CHECK	After the card enters the read station, a check is made at hypothetical positions 0 and 81 to be sure all photocells are dark. If not, this indicator lights, and data outputs are immediately inhibited.
3. STACKER FAIL	When three cards have passed the read station and none have been stacked, this indicator lights, preventing more than three cards from being in the track at once.
4. HOPPER EMPTY	Indicates that the input hopper is empty.
5. STACKER FULL	When approximately 400 cards are in the stacker hopper, this indicator lights.
6. SYNC FAIL	This indicator lights if the sync signal is lost. Internal timing signals are derived from an oscillator that is synced to the track speed.
7. LIGHT CHECK	The photocells must always be illuminated except during the time a card is being read. The Light Check Detector is inhibited each time a card enters the read station until position (count of) 84 is reached. If it fails to leave the read station by that time, this indicator lights.

NOTE

The CR8-E and the CM8-E cannot be used on the same system, because they share IOT codes.

Type CM8-E Optical Mark Card Reader and Control

The CM8-E Optical Mark Card Reader option permits the PDP-8/E computer to accept information from marked or punched data cards with timing marks at a nominal rate of 200 cards per minute. It reads 12-row, 40-column mark sense cards and 12-row, 40-column punched cards. The control circuit is located on a single PDP-8/E module that plugs into the OMNIBUS.

A select instruction starts a card moving through the read station, where all 40 columns are read on a column-by-column basis, beginning with column one. Card data may be read in any one of three modes. In the binary mode, the data is transferred directly from the rows of the card to bits in the AC. The top row of the card (row 12) goes into AC0 and the bottom row (row 9) goes into AC11. In the alphanumeric mode, the data is automatically decoded into a six-bit BCD representation and transferred into the least significant six bits of the accumulator. Use of the six-bit decoding minimizes the size of translation tables and is fully compatible with the Hollerith code as used at this time. A proposed expansion of the Hollerith code would require use of the compressed reading mode. In this mode, rows 9, 12, 11, 0, and 8 are transferred directly to AC4, AC5, AC6, AC7, and AC8, respectively, while rows 1 through 7 are decoded into three-bit BCD representation in AC9, AC10, and AC11. This decoding is based on the lack of double punches in rows 1 through 7, both in the present Hollerith and the proposed extension of Hollerith. If such a double punch is read in the compressed reading mode, the CR8-E validity checking circuitry will assert a one in AC0 (the sign bit). Regardless of the reading mode being used, a punched hole or a non-reflective mark is interpreted as a binary one and the absence of such a hole or mark is interpreted as a binary zero.

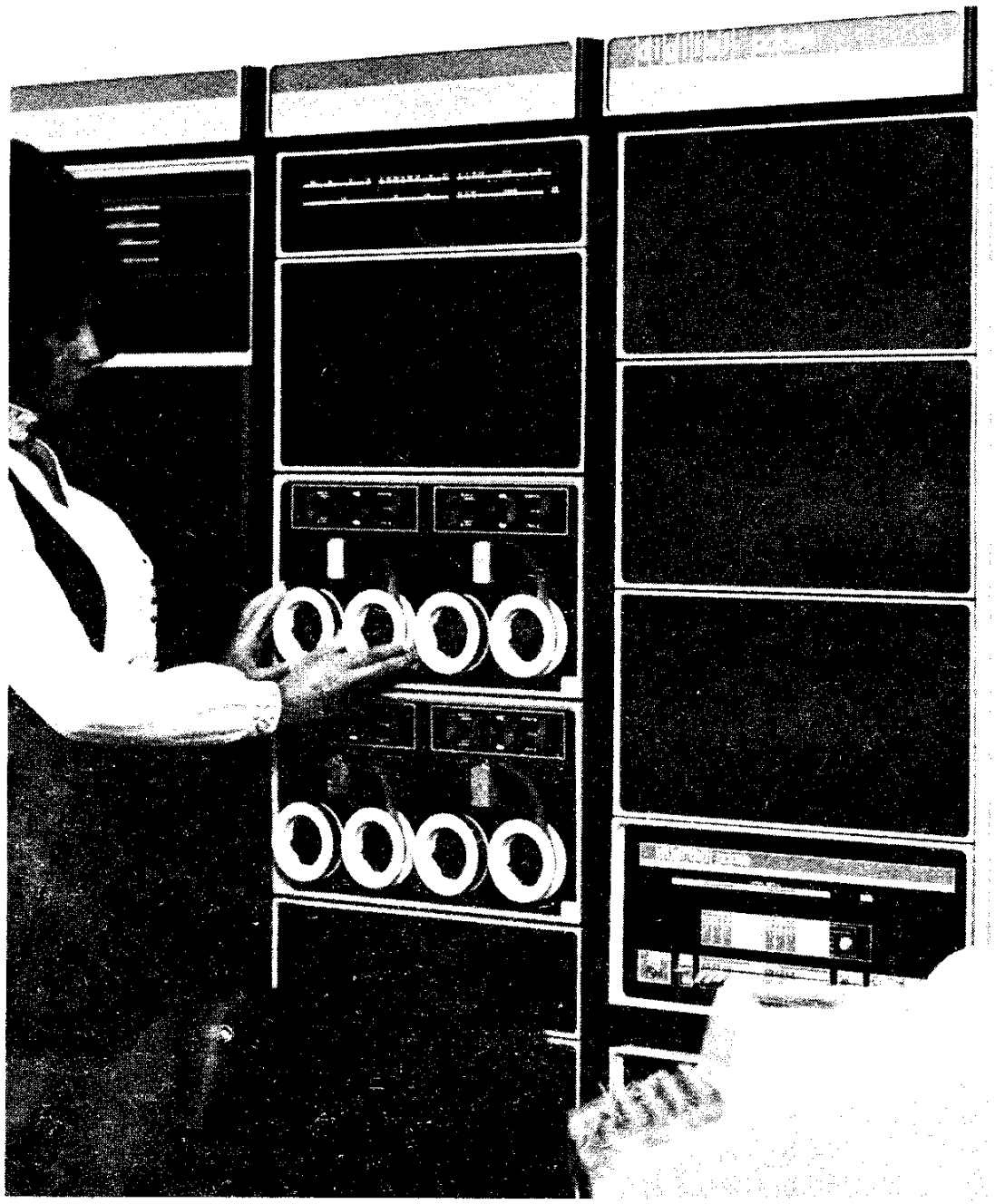
The flag and interrupt facilities are identical to those of the CR8-E.

Specifications

Card Rate:	200 per minute
Input Power:	115 VAC + or − 10 VAC, 60 Hz + or − 5 Hz, a single phase, 300W maximum (50 Hz unit available)
Card Specification:	The card reader is designed to read 7⅜ in. x 3¼ in. optical mark cards conforming to the material and size requirements of EIA Standard RS-292 Media 1. Format and printing requirements are specified in the DEC Mark Sense Card Specification.
Card Capacity:	Both input hopper and output stacker hold 400 cards. Cards may be added or removed during reader operation.

Programming

The instruction set and example subroutine set forth for the CR8-E also applies to the CM8-E.



The TU56 DECtape unit is a low price form of magnetic tape storage equipment with pocket sized reels. The DECtape unit is very reliable and not sensitive to line voltage or frequency variation, and is block addressable like a linear disk. Each TU56 provides more than 262,000 words of storage—131,072 words on each reel and a redundant recording format that records each bit of data on two separate tracks to assure high reliability. It has a transfer rate of 33,300 3-bit characters per second and operates at a speed of 97 inches per second ± 14 ips.

OMNIBUS MAGNETIC TAPE OPTIONS

The OMNIBUS Magnetic Tape Options include:

a. The TD8-EM Dual DECtape Transport Control and TU56 Dual DECtape Transport
b. The TM8-E DECmagtape Transport Control and TU10 DECmagtape transport

DECtapes

The DECtape unit can interface directly with the OMNIBUS via the TD8-E or to the External Bus via the TC08. The configurations are defined in the following table. For information on the TC08 Controller, refer to section 4 of this chapter.

Four basic DECtape configurations are identified in the following table.

SYSTEM DESIGNATION	DECtape	CONTROL	PREREQUISITE	REMARKS
None	TU56 (Dual Drive)	TC08	KA8-E* KD8-E PDP-8/E	Up to 4 Dual TU56's per control. (8 drive units)
None	TU56H (Single Drive)	TC08	KA8-E* KD8-E PDP-8/E	Up to 4 single DECtape drive units.
TD8-EM	TU56-M (Dual Drive)	TD8-E	PDP-8/E	Control plugs into OMNIBUS.
TD8-EH	TU56-MH (Single Drive)	TD8-E	PDP-8/E	Control plugs into OMNIBUS.

TD8-E DECtape Option

The DECtape system is a standard option for the PDP-8/E that serves as an auxiliary magnetic tape data storage facility. The DECtape system stores information at fixed positions on magnetic tape, as in magnetic disk or drum storage devices, rather than at unknown or variable positions, as in conventional magnetic tape systems. This feature allows replacement of blocks of data on tape in a random fashion without disturbing other previously recorded information. In particular, during the writing of information on tape, the system reads format (mark) and timing information from the tape and uses this information to determine the exact position at which to record the information to be written. Similarly, in reading, the same mark and timing information has a number of features to improve its reliability and make it exceptionally useful for program updating and program editing applications. These features are: phase or polarity sensed recording on redundant tracks, bidirectional reading and writing, and a simple mechanical mechanism util-

* Magnetic tape options operated on the external bus of the PDP-8/E require the use of the KA8-E Positive I/O Bus Interface module and the KD8-E Data Break Interface module as prerequisites.

izing hydrodynamically lubricated tape guiding (the tape floats on air over the tape guides while in motion).

Specifications

Tape Characteristics	Capacity—260 feet of ¾ inch, 1 mil Mylar sandwich tape, coated both sides. Reel diameter—3.9 inches Tape Handling—direct drive hubs and specially designed guides float the tape over the head. No capstans or pinch rollers are used. Speed—97 ± 14 ips Density—350 ± 55 bpi Information capacity—2702_8 Blocks with 201_8 12-bit words per block (188,672 12-bit words) Tape Motion—bi-directional
Word Transfer Rate	33,300 3-bit characters per second
Addressing	Mark and timing tracks allow searching for a particular block by number in a forward or backward direction.
Tape Motion Timing	Start Time—150 msec ± 15 msec Stop time—100 msec ± 10 msec Turn around time—200 msec ± 20 msec
Mounting	TU56 Drive mounts in a standard 19 inch equipment rack
Size	10 ½ inches high 19 inches wide 9 ¾ inches deep } TU56 Drive 1 Quad Module } TD8-E Control plugs into OMNIBUS
Cooling	Internally mounted fan provided for TU56
Environmental Conditions	Temperature—40°F to 90°F Note: The magnetic tape manufacturer recommends 40-60% relative humidity and 60° to 80° as an acceptable operating environment for DECtape.

Tape Compatibility

Tapes may be certified, programmed, read, modified, and rewritten interchangeably on either the larger automatic DECtape units (TC08/TC01) or on the TD8-E. DEC provides all the necessary subroutines and MAINDECs for the TD8-E; for example:

- Read/Write Subroutines
- Tape Certification Routine
- MAINDEC Maintenance Programs

- PS/8 Programming System (12K Minimum Configuration)
- A new 4K Keyboard Operating System with Program Directory, Line Editor, and PAL III* Assembler.
- A DECtape Copy Program

* (A Paper Tape Device is required; either ASR-33 or PC8-E, for input and output with PAL-III.)

TD8-E DECTAPE CONTROL

The TD8-E is a low cost interface for the TU56 DECtape units. A TD8-EM consists of a TD8-E and one TU56-M Dual DECtape drive. The TD8-EH consists of a TD8-E and one TU56-MH Single DECtape drive.

The TD8-E is contained on a single quad Flip-Chip module which plugs directly into the OMNIBUS of the PDP-8/E. It is connected to the TU56 by a special interface cable (P.N. 7008447). It uses a standard TU56 with no modifications. The Read/Write Amplifiers (G888) must be plugged into the TU56 drives.

When reading, writing, or searching, the PDP-8/E acts as a controller for the DECtape. That is, all data transfers to and from the 8/E are through the AC in non-interrupt, non-data break mode. The PDP-8/E is completely committed to the tape operation and cannot perform any other functions until the tape operations have been completed.

Up to four TD8-E interfaces can be used with a PDP-8/E. Each TD8-E can drive either a single or dual transport. It is therefore possible to have eight DECtape drives connected to the PDP-8/E through four TD8-E's. When a dual transport is used on the TD8-E's, the first TD8-E will control units 0 and 1; the second TD8-E will control units 2 and 3; the third, units 4 and 5; and the fourth, units 6 and 7.

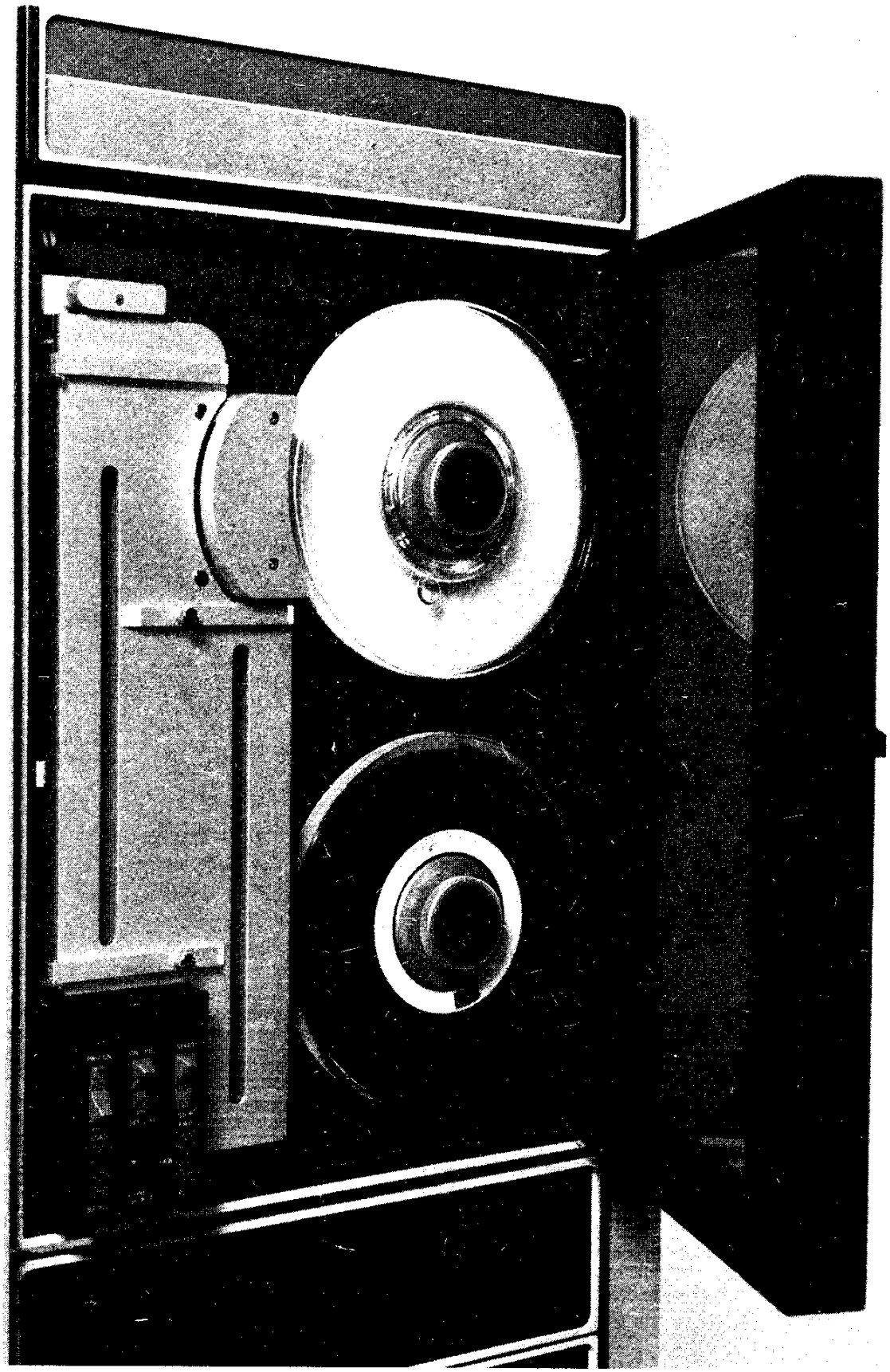
A comprehensive set of diagnostic routines is included with the TD8-E which checks all of its functions. The TD8-E is also supplied with subroutine software which search, read, and write PDP-8 compatible DECtapes. DECtapes written with the TC01 or TC08 control can be read with the TD8-E using this software and vice versa. Because of the close dependency of the hardware with the software, Digital Equipment Corporation will not guarantee operation of the TD8-E with any software other than that which is supplied by Digital Equipment Corporation.

The TD8-E was designed as a low-cost DECtape interface with limited functions. It is not a replacement for the TC08 which makes transfers of data direct to memory concurrent with CP operations. Its primary use is for library storage of programs and blocks of data. The TD8-E will, however, like the TC08, certify DECtapes by writing and verifying the mark and time tracks and block numbers.

Refer to Section 4 of this chapter for a detailed discussion of TU56.

TU10 DECmagnetic Tapes

The DECmagtape can interface directly with the OMNIBUS via the TM8-E, or to the EXTERNAL BUS via the TC58. The configurations of both categories are defined in the following table. For information on the TC58 controller, refer to section 4.



TU10 DECmagtape

DECmagtape Configurations

SYSTEM OPTION	EQUIPMENT	NO. OF CHANNELS	DENSITIES (BPI)	TAPE SPEED (IPS)	OTHER INFORMATION
TM8-EA	TM8-E Control & TU10-EA(master)	9	800	45	Control plugs into OMNIBUS. TU10-EA contains a master and one slave. Up to 7 additional TU10 slaves may be added. 7 and 9 track TU10's can be mixed on the same system. For example, a 7 track master can be operated with a 9 track slave etc. The master consists of logic modules which plug into the TU-10 electronics.
TM8-FA	TM8-E Control & TU10-FA(master)	7	800/556/200	45	Same as above.
TC58 *	TC58 Control(master) & TU10-EE(slave)	9	800	45	DW08A I/O conversion panel, KA8-E Positive I/O Bus and KD8-E Data Break Interface are prerequisites. The master is contained with the TC58 controller. Up to 7 additional TU10 slaves may be added. 7 and 9 track TU10's can be mixed on the same system.
TC58 *	TC58 Control(master) & TU10-FE(slave)	7	800/556/200	45	Same as above.

* Refer to Section 4 for TC58 Description

OMNIBUS DECmagtape Unit and Control Type TM8-E/F

NOTE

The following information is preliminary and is subject to change without notice. The reader should consult with the local DEC sales office.

The TM8E control provides the interface between the PDP-8/E and the TU10 master-slave magnetic tape transport system. The TU10 master can control 7 slaves; therefore the TM8-E is capable of controlling 8 transports.

The data transfer is via single cycle data break with a transfer rate of 36 KHZ. The transport operates at 45 ips and uses 7 channel formats at 200, 556 or 800 bpi or 9 channel format at 800 bpi.

The TM8-E contains six registers which are used to control the transports and report the status of the transports to the computer. The registers are loaded and read using IOT instructions which require no data break.

PROGRAMMING

The following Instructions are used to program the TM8-E:

Load Word Count Register (LWCR)

Octal Code: 6701
Operation: Load Word Count Register and Clear the AC
AC → WC, 0 → AC

Clear Word Count Register (CWCR)

Octal Code: 6702
Operation: Clear Word Count Register

Load Current Address Register (LCAR)

Octal Code: 6703
Operation: Load Current Address Register and Clear the AC
AC → CA, 0 → AC

Clear Current Address (CCAR)

Octal Code: 6704
Operation: Clear Current Address

Load Command Register (LCMR)

Octal Code: 6705
Operation: Load Command Register and Clear the AC
AC → CM, 0 → AC

Load Function Register (LFGR)

Octal Code: 6706
Operation: Load Function Register (GO bit) and Clear AC
AC → Function Register 0 → AC

Load Data Buffer Register (LDBR)

Octal Code: 6707
Operation: Load Data Buffer Register and Clear AC
AC → DB, 0 → AC

Read Word Count Register (RWCR)

Octal Code: 6711
Operation: Clear AC and Read Word Count Register
0 → AC, WC → AC

Clear Transport (CLT)

Octal Code: 6712
Operation: Clear Transport

Read Current Address Register (RCAR)

Octal Code: 6713
Operation: Clear AC and Read Current Address Register
0 → AC, then CA → AC

Read Main Status Register (RMSR)

Octal Code: 6714
Operation: Clear AC and Read Main Status Register
0 → AC, then MS → AC

Read Command Register (RCMR)

Octal Code: 6715
Operation: Clear AC and Read Command Register
0 → AC, then CM → AC

Read Function Register & Status (RFSR)

Octal Code: 6716
Operation: Clear AC Read Function Register and Status 1
0 → AC, then Function and Status 1 → AC

Read Data Buffer (RDBR)

Octal Code: 6717
Operation: Clear AC and Read Data Buffer
0 → AC, then DB → AC

Skip if Error Flag (SKEF)

Octal Code: 6721
Operation: Skip if error flag is set.

Skip if Control not Busy (SKCB)

Octal Code: 6722
Operation: Skip if the control is not busy. The TM8-E becomes busy when a go is given to the transport and becomes not busy at MTTF.

Skip When Job Done (SKJD)

Octal Code: 6723
Operation: Skip if the job is done (MTTF is set). The job done flag (MTTF) sets at the end (LRCS) of a Read, Read/Compare, Write File Mark, or Write operation, and at the end of a record, if an EOT, EOF or BOT was encountered or the WC overflowed during a space operation.
MTTF sets when a transport begins to do a rewind and a new transport may be selected, when a transport goes off-line following an off-line operation, and when a rewinding transport has reached BOT and is ready.

Skip When Tape Ready (SKTR)

Octal Code: 6724
Operation: Skip if tape unit is ready (TUR is true).

Clear Controller and Master (CLF)

Octal Code: 6725
Operation: Clear the Controller and Transport Master if TUR, if not clear MTTF, EF and Status Registers.
0 → Control Registers

Octal Code: 6726
Reserved for Maintenance

Octal Code: 6727
Reserved for Maintenance

Description of Registers

6701 LWCR The 12 bit Word Count Register may be loaded from AC 0—11 any time the control is not busy. If the register is loaded during Control Busy, data reliability and tape compatibility cannot be assured. The Word Count must be loaded to the 2's complement of the number of words to be transferred or blocks to be spaced. The Word Count is incremented at TPI of the break cycle during Data Transfers and at LPCS during a space forward, and at the first word of a block during a Space Reverse.

Recommended block length is per USA Standards, Document USAS X3.22-1967. Recorded Magnetic Tape for information interchange (800 cpi, NRZ1).

6702 CWCR This IOT clears the Word Count Register and is essentially for maintenance use. It should not be used during Control Busy.

6703 LCAR The 12 bit Current Address Register may be *loaded from AC 0–11 any time the control is not* busy. It must be loaded to one less the Memory Address where the first data is taken or placed. If the Register is loaded during Control Busy, the following occurs:

1). In wrap around mode, function bit 6 = 0, location of the data transfer can not be assured within the selected memory field.

2). In EMA INC Enable mode, function bit 6 = 1, location of the data transfer can not be assured within the memory.

The Current Address Register is incremented at Break Request prior to the break cycle.

6704 CCAR — This IOT clears the Current Address Register and is essentially for maintenance use. It should not be used during Control Busy.

6705 LCMR — The Command Register can only be loaded from AC 0—11 during Control Not Busy. If the IOT is issued during Control Busy, an illegal function will be indicated and the current operation aborted. The transport may have to be rewound.

Bits

0, 1, 2	Unit selection: These determine which of the eight transports will be used.
0 0 0	Transport 0
0 0 1	Transport 1
0 1 0	Transport 2
0 1 1	Transport 3
1 0 0	Transport 4
1 0 1	Transport 5
1 1 0	Transport 6
1 1 1	Transport 7
Bit 3	Parity: 0 = Even 1 = Odd
Bit 4	Enable Interrupt on Error Flag
Bit 5	Enable Interrupt on MTTF (job done flag)
Bits 6, 7, 8	Extended Memory Address: These three bits determine which memory field the controller uses. The manner in which these bits are loaded depends upon the setting of the EMA Enable bit, Function Register bit 6.

Bits 6 7 8	
0 0 0	Field 0
0 0 1	Field 1
0 1 0	Field 2
0 1 1	Field 3
1 0 0	Field 4
1 0 1	Field 5
1 1 0	Field 6
1 1 1	Field 7

Bit 9 — Reserved for Future Use

Bits 10, 11 — Density: These bits select the density for the transports operation.

10	11		
0	0	200 bpi	7 channel
0	1	556 bpi	7 channel
1	0	800 bpi	7 channel

This also serves as a core dump mode. When issued to a 9 channel transport, data is written in 7 channel format and zero's are written in channels 0 and 1 on the tape.

1	1	800 bpi	9 channel

6706 LFGR — The function register must be the last register to be loaded, since this register contains the GO bit.

Bit 0	Bit 1	Bit 2	
0	0	0	Off Line: The selected transport is taken off-line and rewound to BOT. The MTTF is set when the transport responds to the function, the controller may then select and use another transport. The transport must be manually reset to the on-line state. The Word Count and Current Address Registers need not be loaded.
0	0	1	Rewind: The transport rewinds at high speed (150 ips) to BOT and stops. The MTTF is set when the transport responds to the function. The controller may then select and use another transport. Should the rewinding

transport be reselected, another MTTF will occur when the tape has stopped at BOT. The word count and Current Address Registers need not be loaded.

0	1	0	Read: Data may be transferred from the tape to memory in the forward direction only. All registers must be loaded.
0	1	1	Read/Compare: Tape data is compared to data in core memory. All registers must be loaded. If there is a comparison error, CA incrementation ceases, and the R/C error bit is set. Tape motion continues to the end of the record. The CA register contains the address of the word which failed.
1	0	0	Write: Data may be written on the tape in the forward direction only. All registers must be loaded. When the proper number of words have been written the transport writes the appropriate check characters to end the block.
1	0	1	Write End of File (File Mark): The transport writes the file mark which consists of a one word block. The CA and WC registers need not be loaded.
1	1	0	Space Forward: The transport moves forward at 45 ips the number of records specified by the WC register, or until a File Mark is read. If End of Tape is read space forward will stop at the first inter-record gap. The CA register need not be loaded.
1	1	1	Space Reverse: The transport moves in the reverse direction at 45 ips the number of blocks specified by the WC or until a file mark or BOT marker is read. The CA register need not be read.

Bit 3 Extend Gap: This bit causes the transport to write with a minimum 3 inch gap between blocks.

Bit 4 Enable Check Characters: When this bit is set, it will allow the check characters to be read into the computer during a read function. When the word count overflows, this bit will allow at least one break during 7-track operation for the LPC or two breaks during 9-track operation for the CRC and LPCC. If a record length incorrect error occurs, the check character is considered bad and can not be used. This feature will be used primarily for 9-track error correction.

Bit 5 — GO: This bit causes the controller to issue a GO command to the transport when the transport is capable of accepting it. The GO will not be issued if the specified function is illegal.

Bit 6 — EMA INC Enable: If this bit is not set, the TM8-E will treat the extended memory the same way any other PDP-8 Family data break option would, i.e., each 4K block is used in a wrap around mode.

If this bit is set, the extended memory will be treated as a continuous memory rather than as 4K blocks. When the last location in a field is reached, the EMA bits are incremented and the transfer continues in the next field. I.e.: If a word is placed in field 2, location 7777, the following word will be placed in field 3, location 0000 if the EMA increment bit is set. If it is not, the word will be placed in field 2, location 0000.

In both modes of operation, the Current Address must be set to one less than the first location to be accessed. The 12 bit CA register and the 3 EMA bits are treated as one 15 bit register with the EMA bits most significant. I.e.: to access field 2, location 20, load EMA = 2, CA = 0017; to access field 2, location 0, load EMA = 1, CA = 7777 if in EMA increment mode; to access field 2, location 0, load EMA = 2, CA = 7777 if not in EMA increment mode.

If memory field 7 is selected, the EMA cannot increment, but will wrap around in field 7 and an EMA 7 increment error will occur.

6707	LDBR	Load Data Buffer Register and Clear the AC: This is primarily used for maintenance.
6711	RWCR	Clear the AC and Read The Word Count Register into the AC: This is primarily used for maintenance but also may be useful during Error Check routines.
6712	CLT	Clear Transport: This will clear the transport's master registers.
6713	RCAR	Clear the AC and Read Current Address Register: This is primarily used for maintenance but may also help during error check routines.
6714	RMSR	The 12 bit main status register is used to report the most important status of the transport and control to the computer. It may be read into AC 0–11 at any time.

Bit	
0	Error Flag: The Error Flag will interrupt the processor if the interrupt enable bit (CM04) is set. An illegal function or select error will set the Error flag immediately, halting data breaks and ending a Write operation. BOT, EOT, Read/Compare Error, Bad Tape, Lateral or Longitudinal parity errors, Record length incorrect, data late, or EMA 7 increment error will set the Error Flag after MTTF is set.
1	Rewind Status: The selected transport is rewinding.
2	Beginning of Tape (BOT): The BOT reflective strip is being sensed by the selected transport.
3	Select Remote: The selected transport is not on-line.
4	Parity Error: A longitudinal or lateral parity error has been detected.
5	File Mark (EOF): The selected transport detected a file mark during a space, read, or Read/Compare operation.
6	Record length incorrect: During a read or READ/Compare operation, the record length was different from the contents of the WC. The Word Count may be read to determine whether the record was long or short.
7	Data Request Late: The computer failed to service the break request before the next data transfer to or from the transport.
8	End of Tape (EOT): The EOT reflective strip has been sensed by the selected transport.
9	File Protect: The selected transport has a write lockout ring. No write functions will be accepted.
10	Read/Compare Error: A comparison failure occurred during the Read/Compare function. The CA contains the address of the bad word.
11	Illegal Function: 1. Issuance of LCMR, LFGR, or LDBR while the control is busy. 2. Specifying any density but 800 bpi for a 9-channel transport. 3. A space reverse function when the transport is at BOT.

4. Read, Read/Compare or Space Forward after a Write or WEOF command.
5. Changing to transports which is not ready. (TUR is false)

6716	RFSR	Clears the AC and Read the Function and 2nd status register.
Bits 0-5		Function Register.
6		Transport channel: The selected transport is 7-channel if the bit is 0, and 9-channel if it is 1.
7		Bad Tape: Bad tape error indicates two or more consecutive characters missing, followed by data within the time of settling down. The CRC and LPCC will not cause bad tape errors.
8		EMA 7 INC Error: This occurs if an attempt is made to increment the EMA from field 7 to field 0. The data will wrap around in field 7.
9		Lateral Parity Error: A lateral parity error was detected.
10		Reserved for Future Use.
11		Longitudinal Parity Error: A longitudinal parity error was detected.
6717	RDBR	Clear the AC and Read data buffer into the AC. This is primarily used for maintenance.

TU10 MASTER

The TU10 Master controls the function timing, write pulses, generation of all the check characters and checking of parity, and is capable of controlling 8 slaves on a common bus. The TU10 Master unit includes 1 TU10 slave.

TU10 SLAVE

The TU10 DECmagtape Transport is a solid-state, magnetic tape handling device that controls tape motion and reads or records digital information on magnetic tape in industry-compatible formats.

The TU10 uses vacuum columns and a servo-controlled single capstan to control tape motion. The only contact with the oxide surface is the magnetic head and a rolling contact on one low-friction, low-inertia bearing. Dancer arms and pinch rollers, which shorten tape life and can cause errors, are not used in the TU10.

Tape transport commands can be issued manually from the TU10 control panel or remotely from the processor by means of the Controller. Indicators on both the transport and the controller indicate transport status.

Each tape transport consists of the TU10 cabinet, reel and reel motor control, capstan drive, and read/write components. The circuitry which controls the motion of the transport, generates the write pulses, timing gaps, parity, and check characters, and checks the parity is located in the Controller. These logic circuits may be shared by up to 8 TU10's.

SPECIFICATION

Power and Cabling

TU10 Power:	tape transport power (reel motors and fans) provided by internal power supply in each transport

Cabling:

a) 2 BC08P-15 to connect TC58
   1 BC08N-15 to TU10
b) 3 BC08N-15 to Bus TU10's together
c) 2 BC08L-15 to connect TM8-E to TU10

TU10 DECmagtape Transport

Mounting:	mounts in standard H960-CA cabinet
Size:	26 inches high, 19 inches wide, 25 inches deep
Cooling:	internally mounted fans
Controls:	front panel mounted

Environmental Conditions

Temperature:	40°F to 100°F for system 60°F to 80°F for magnetic tape
Humidity:	20% to 95% (non-condensation) for system 40% to 60% (non-condensation) for tape

Power Input Requirements

TU10-EE, FE	115 Vac, 60 Hz at 14A
TU10-EH, FH	115 Vac, 50 Hz at 14A
TU10-EF, FF	230 Vac, 60 Hz at 7A
TU10-EJ, FJ	230 Vac, 50 Hz at 7A

Local Transport Controls

PWR ON/PWR OFF	power control switch
ON-LINE/OFF-LINE	local or programmed operation
START/STOP	tape motion control
LOAD/BR REL	releases brake for loading
UNIT SELECT	selects unit for program control
FWD/REW/REV	tape direction control

Tape Characteristics

Capacity:	2400 feet of ½-inch, industry standard, 1-mil Mylar tape.
Reel Diameter:	10-½ inch standard reels
Tape Handling:	direct-drive reel motors; servo-controlled single capstan; vacuum tape buffer chambers with constant tape winding tension
Tape Speed:	45 inches per second, reading and writing
Rewind Speed:	150 inches per second (approximately 3-minute rewind time for 2400-foot reel)
Packing Density:	**7-channel**—200, 556, and 800 BPI, selectable under program control
	9-channel—800 BPI

Data Recording and Transfer

Recording Mode:	NRZI, industry compatible
Magnetic Head:	Dual gap, read-after-write
Data Transfers:	Direct memory access (non-processor request)
Transfer Rate:	36,000 characters per second, maximum
BOT, EOT Detection:	photoelectric sensing of reflective strip, industry compatible
Write Protection:	write protect ring sensing
Data Checking:	read-after-write parity checking; longitudinal redundancy check; cyclic redundancy check (9-channel only)
Interrecord Gap:	reads tape with gap of 0.48 inches or more; writes tape with gap of 0.52 inches or more (compatible with industry standard)

TU10 Models

No. of Channels	Type of Unit	115 VAC 60 Hz	115 VAC 50 Hz	230 VAC 60 Hz	230 VAC 50 Hz
9-channel	Master	TU10-EA	TU10-EC	TU10-EB	TU10-ED
	Slave	TU10-EE	TU10-EH	TU10-EF	TU10-EJ
7-channel	Master	TU10-FA	TU10-FC	TU10-FB	TU10-FD
	Slave	TU10-FE	TU10-FH	TU10-FF	TU10-FJ

NOTE: DECmagtape units TU20 and TU30 are also compatible with the TM8-E and TC58 controllers.

LABORATORY PERIPHERALS

AD8-EA Analog-to-Digital Converter

The AD8-EA converter is a 10-bit successive-approximation type with sample and hold circuits, conversion circuits, an input buffer, and control logic contained on two PDP-8/E modules. The converter can be used singularly with one channel input having an input range from −5 to +5 volts or can be used with AM8-EA and AM8-EB Multiplexers to perform conversion for up to 16 channels having full-scale inputs from + 1 to − 1 volts. Analog inputs are connected to the module by H855 connectors from the multiplexer or by a shielded twisted pair from an external device.

Operation of the AD8-EA converter is controlled by IOT instructions. A conversion is initiated by an ADST instruction, or from the Real Time Clock DK8-EP. An input starts the conversion and clears the A/D Done Flag. When the conversion is complete, the converter sets its A/D Done Flag. This flag is sensed by an ADSK instruction. If it set, the next instruction is skipped so that the 10-bit digital word can be transferred to AC2-11 by an ADRB instruction. Since the 10-bit word is in two's complement form, AC00 and AC01 copy AC02 (sign-extended format). The converter contains an interrupt enable flip-flop that is controlled by program instructions. When enabled, this flip-flop permits the converter to generate interrupt requests to the program interrupt facility upon completion of conversion.

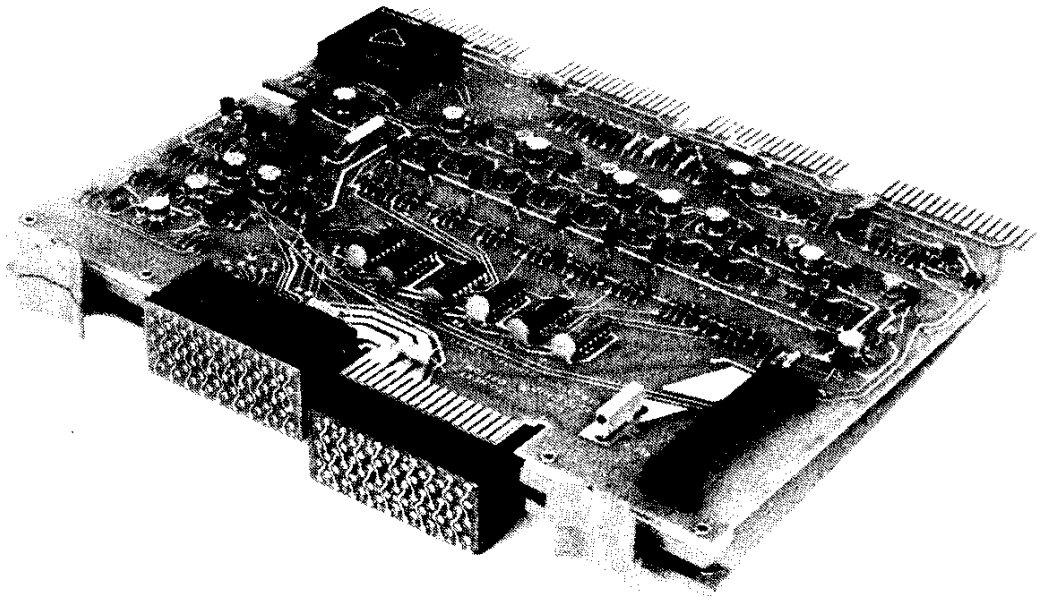
The converter also contains circuits for detection and sensing of a timing error. A timing error is defined as the receipt of a conversion request while a conversion is in progress. If this condition occurs, a Timing Error Flag is set. The Timing Error Flag is sensed by an ADSE instruction. An ADST or ADCL instruction clears this flag. This instruction also clears the A/D Done Flag so that another conversion can be implemented.

Specifications

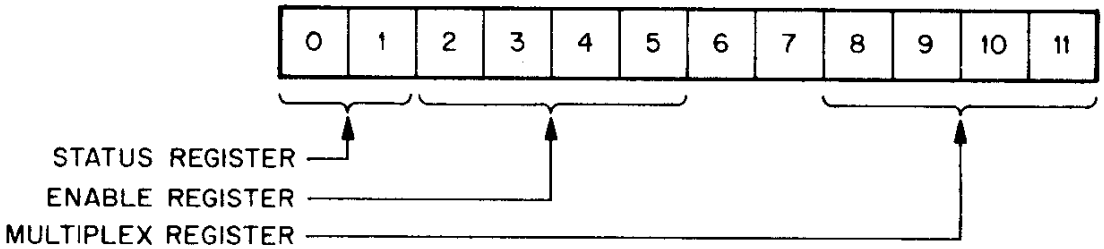
Input Voltage Range:	−5.0 to +5.0 Volts
Input Impedance:	Signal Return Input $\geqslant$ 10KΩ Signal Input $>$ IMΩ
Output Format:	Parallel: 10 bits right-justified and sign extended, two's complement
Resolution:	$\pm$ ½ LSB at 20 degrees C greater
Conversion Rate:	than 50 kHz
Sample Acquisition Time:	Approx 3 μs
Aperture Time:	200 ns

PROGRAMMING

The following instructions are used to program the operations involving the AD8-EA and AM8-E. Each instruction is completed in 1.2 μsec.



AD8-EA A/D Converter



Status Register	0—A/D Done Flag Status (Done = 1) 1—Timing Error Flag Status (Error = 1)
Enable Register	2—Enable Interrupt on A/D Done = 1 3—Enable Interrupt on Timing Error = 1 (note 1) 4—Enable External (e.g. clock) A/D start 5—Auto-Increment Mode (note 2)
Multiplex Register	8-11—Indicates current channel (0 — 17_8) to be sampled by A/D

Note 1: The Timing Error Flag indicates that either an ADRB, and ADLM, ADST or external A/D start was attempted while a conversion was in progress. ADLM will be honored while an external A/D start or ADST will be ignored under this condition. ADLM or ADRB will cause an erroneous result to appear in A/D Buffer.

Note 2: When this bit is set, the occurrence of A/D Done = 1 will increment the Multiplex Register by 1. Incrementing past channel 17_8 will cause the MUX register to reset to channel 0.

Eight instructions are used to program the A/D Converter and Multiplexer. Each instruction is completed in 1.2μs and is defined as follows:

Clear All (ADCL)

Octal Code: 6530
Operation: Clears the A/D Done Flag and Timing Error Flags to ready the converter for another conversion. This instruction also clears the MUX and Status Register.

Load Multiplexer (ADLM)

Octal Code: 6531
Operation: Load Multiplexer register from AC8-11 and Clear AC.

Start Conversion (ADST)

Octal Code: 6532
Operation: Clear A/D Done and Timing Error Flags and Start A/D Converter. Channel to be converted is determined by MUX register.

Read A/D Buffer (ADRB)

Octal Code: 6533
Operation: Clear A/D Done Flag and load the contents of the A/D Buffer into ACO-11.

Skip On A/D Done (ADSK)

Octal Code: 6534
Operation: Skip the next instruction if A/D Done = 1. Do not clear flag.

Skip On Timing Error (ADSE)

Octal Code: 6535
Operation: Skip the next instruction if Timing Error Flag = 1. Do not clear flag.

Load Enable Register (ADLE)

Octal Code: 6536
Operation: Load Enable Register from AC2-5 and clear AC Register.

Read Status Register (ADRS)

Octal Code: 6537
Operation: Read A/D Status, Enable Register, and MUX into ACO-11.

PROGRAMMING EXAMPLES

Normal Mode—

The simplest method of programming the analog-to-digital converter is to have the program issue a start command, loop on the done flag until the conversion process is complete and the done flag is set to a "one", then the value of the converter's buffer is read into the PDP-8/E Accumulator. The program looks like this:

```
ADST        /Clear the ADC done flag and start conversion
ADSK        /Skip the next instruction when done
JMP .—1     /Jump back one instruction
ADRB        /Read ADC buffer into AC
```

If the Analog-to-Digital Converter had been enabled to accept start pulses from an external device, such as a clock, then a timing error could occur. To check for this the following code could be added after the ADST command:

```
•
•
•
ADSE        /Skip the next instruction on error
SKP         /unconditional skip
JMS ERROR   /Go to error routine
•
•
•
```

When the ADC is equipped with the multiplexer option, the channel to be sampled is selected prior to starting the conversion process. This is done using the ADLM command. For example a simple program to continuously "read" the value of one of the parameter knobs and display the digital value in the PDP-8/e accumulator look like this:

```
START,  CLA             /clear the PDP-8/E accumulator
        TAD CHN         /get the channel # (0-3 for knobs)
AGAIN,  ADLM            /load multiplexer from AC
        ADST            /start
        ADSK            /skip when finished
        JMP .—1         /
        ADRB            /Read ADC value
        JMP AGAIN       /repeat process
```

Clock Mode—

In this special mode, an external event, usually the clock overflow starts the conversion process. This mode sample is taken at regular intervals as defined by the clock rate. The following example takes 1000_{10} samples at the specified clock rate and stores them in memory.

```
        •
        •
        •
INITIALIZE CLOCK AND ADC ENABLE REGISTER
        •
        •
        •
START,  CLA
        TAD NUMBER
        DCA COUNTER
        TAD ADDRESS
        DCA POINTER     /POINTER IS AN AUTO-INDEX REGISTER

ADLOOP  ADSK
        JMP .—1
        ADRB
        DCA I POINTER
        ISZ COUNTER
        JMP ADLOOP
        •
        •
        •

NUMBER ,—1750₈          /# of samples (1000₁₀ in this case)
COUNTER , 0
ADDRESS , n-1           /Beginning of table
POINTER , 0
```

Fast Sample Mode—

In fast sample mode the PDP-8/E processor is allowed to proceed while the called for conversion is still in process. The conversion still requires its full time to complete but since the order of events has changed, the sample may be taken at the full speed of the Analog-to-Digital Converter. The following example demonstrates this:-

```
                •
ADLOOP ,    ADSK
            JMP .—1
ADSTART ,   ADRB
            ADST
            DCA I POINTER
            ISZ COUNTER
            JMP ADLOOP
```

To make use of this program it is necessary to enter the program at ADSTART.

AM8-EA 8-Channel Analog Multiplexer

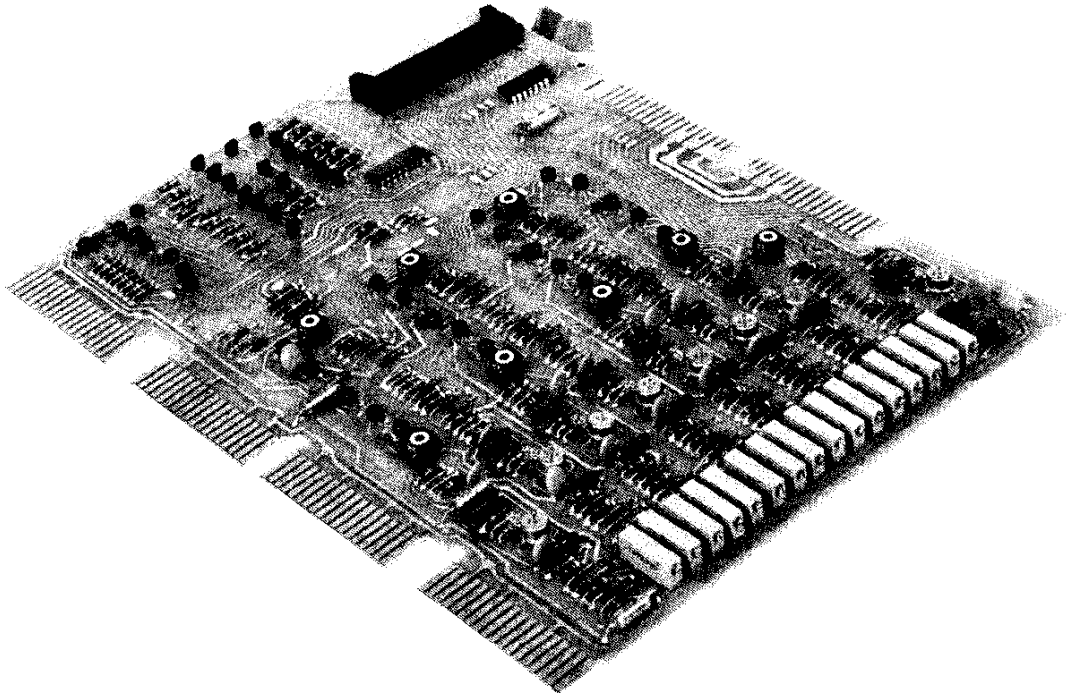
The AM8-EA is an 8-channel analog multiplexer designed for use with the AD8-EA A/D Converter. The multiplexer accepts bipolar analog input having a full-scale range of ±1v and converts these inputs into a full-scale ±5.0 volt output supplied to the AD8-EA Converter.

The AM8-EA consists of multiplexer switches and scaling amplifiers for 8 analog channels. The AD8-EA can be expanded to 16 channels in eight-channel groups by adding an AM8-EA 8-channel Multiplexer Module. Multiplexer operation is controlled by the AD8-EA IOT instructions. These instructions and the associated multiplexer control provide the capabilities for random or sequential selection of channels, combining the operation of the A/D converter with that of the multiplexer. Two programmable address modes are provided: autoincrement or non-autoincrement. The AM8-EA is set to nonautoincrement mode when INITIALIZE is generated.

In the autoincrement mode, channel addresses are incremented automatically at the completion of a conversion by an A/D Done Flag from the converter. The program specifies the first address of interest by issuing an ADLM instruction and then can issue an instruction to start an A/D conversion. Upon completion, the A/D Flag increments the multiplexer channel address for the next sample. This process can continue until the AUTO MODE flip-flop is reset.

Specifications

Input Voltage	Bipolar, ±1V
Input Impedance	70K ohms ± 2%, shunted by 300 pf
Output	Bipolar, ± 5V full scale
Common Mode Rejection	Greater than 25 dB, 35dB typical



AM8-E 8-Channel Analog Multiplexer

Overload Protection	±67V from fault line (indefinitely)
Overload Recovery Time	8 μs
Frequency Response	Flat from 0 to 30 KHz, —3dB 60 KHz
Leakage Current	Negligible at 70K ohms impedance
Long Term Stability (1 hour)	1% for ± 30°C

DR8-EA 12-Channel Buffered Digital I/O

The DR8-EA Digital I/O can be used to control 12 discrete digital switching circuits located externally, and can be used to accept 12 discrete digital inputs from external sources. The unit consists of IOT control logic, a 12-bit input buffer, a 12-bit output buffer, and 3 multiplexer ICs that control the flow of data for input and output operations. All circuits are TTL logic and are mounted on a single PDP-8/E module which plugs into the OMNIBUS. The standard TTL outputs are connected to the external load via two H854 connectors on the module. Inputs from external sources are also connected to the DR8-EA using H854 connectors.

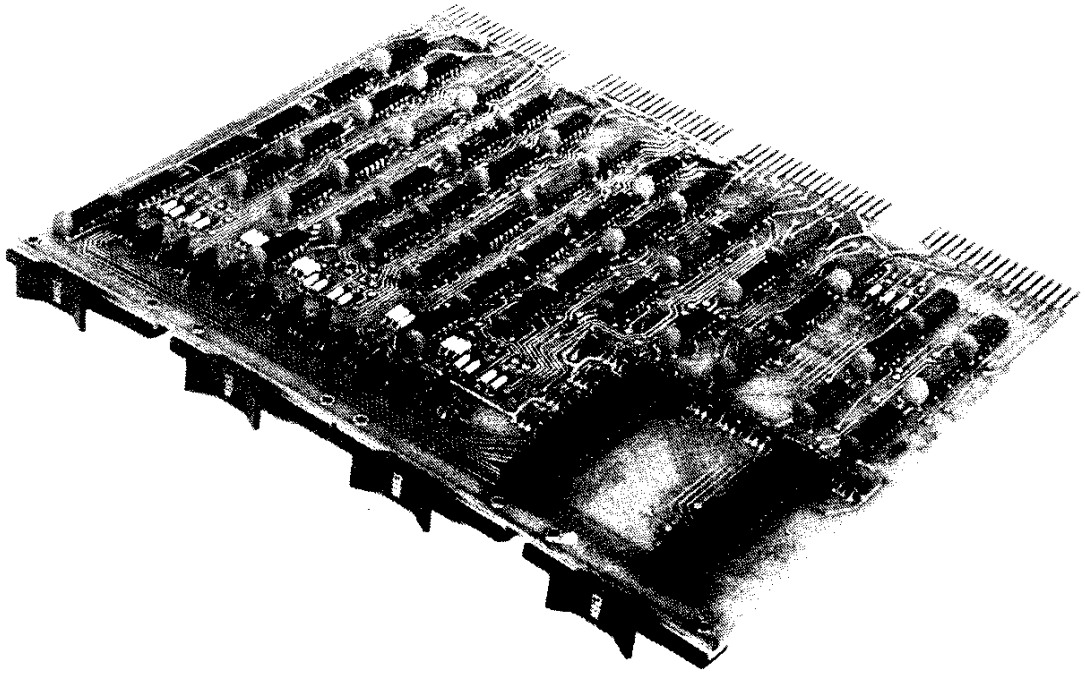
Data outputs are updated under program control. Standard output drivers have a TTL 30-unit load capability. For an output function the

computer issues a DBRO, DBSO, or DBCO instruction. For DBSO instructions, only logical ones in the AC are loaded into the output register; AC bits containing logical ZEROS do not affect output register bits. For DBCO instructions, logical ONE's in AC result in logical ZEROS in corresponding bits of the output register. For DBRO instructions, the contents of the output register is transferred into the AC register.

Data inputs must be TTL compatible, have negative transition to .8V or less for a logical ONE, and have a pulse duration of greater than 50ns. Pulse rise and fall time should be less than 150ns for maximum noise immunity. In one mode of operation, the input register bits, once set by the data inputs, remain set until read by a DBRI instruction. In the second mode of operation, the input can be placed directly through gating on the bus, and will remain as long as the input remains. The DBRI instruction is also used to read the input data. When this IOT is issued, the content of the input register is gated to the AC via the OMNIBUS. A DBCI instruction, used with DBRI instruction, enables inputs that occurred too late to be read by the next DBRI instruction. Correct usage of this feature results in "zero dead time" for events. Any of the input lines can cause an interrupt if the proper jumpers are selected. The interrupt facility can be enabled by instruction DBEI and disabled by instruction DBDI.

A maximum of 8 DR8-EA options can be used. Each device selector code is determined by the user by means of jumpers. Device codes 50 to 57 are legal; however, the DR8-EA normally comes with device code 50 installed.

The DR8-EA is contained entirely on one PDP-8/E module.



DR8-EA 12 Channel Buffered Digital I/O

Specifications

Input Format	Parallel, 12 bits.
Input Levels	Compatible TTL levels. Input circuitry switches at 0.8 to 2.4 Volts, and is protected to allow input swings as positive as +20 Volts and as negative as −15 Volts.
Input Connections and Pulse Width	Inputs to inverter buffers are normally held high by resistors. A negative transition of 0.8V or less will cause the input to become a logical ONE. Optional inputs bypass the flip-flop for direct interrogation of input line status.
Output Format	Parallel, 12 bits.
Output Levels	TTL-compatible levels capable of driving 30 unit loads. Output lines are protected from short circuits to ground.
Environmental:	0°C to 55°C 10% to 90% relative humidity (non-condensing)
Power Requirements:	+5.0 volts, 2.25 amps (worst case)

Programming

The following instructions are used for DR8-EA operation. The X refers to a jumper selectable code. However, the DR8-EA normally comes with code 50 installed.

Disable Interrupt (DBDI)

Octal Code: 65X0
Operation: Disable all interrupts that are caused by a logical ONE on the input.

Enable Interrupts (DBEI)

Octal Code: 65X1
Operation: Set Interrupt Enable Flip-Flop. This tests the IN FLAG and causes an Interrupt Request if IN FLAG equals ONE.

Skip on Flag (DBSK)

Octal Code: 65X2
Operation: Tests the IN FLAG. If the Flag is a ONE, the next sequential memory location is skipped.

Clear Selective Input Register (DBCI)

Octal Code: 65X3
Operation: ONE's in the AC clear respective bits in the Input Register.

Transfer Input to the AC(DBRI)

Octal Code: 65X4
Operation: Transfers the complete 12-bit Input Register to the AC.

Clear Selective Output Register (DBCO)

Octal Code: 65X5
Operation: ONE's in the AC clear the respective bits in the Output Register.

Set Selective Output Register (DBSO)

Octal Code: 65X6
Operation: ONE's in the AC set the respective bits in the Output Register.

Transfer Output to AC (DBRO)

Octal Code: 65X7
Operation: Transfer the complete 12-bit Output Register to the AC.

Programming examples

To clear all registers

```
        CLA CMA    /Set AC to 7777
        DBCI       /set all input bits to zero
        DBCO       /set all output bits to zero
        DBDI       /disable interrupts
```

To service the occurrence of events

```
START,DBSK         /has event happened
        JMP-1      /no, check again
        DBRI       /yes, read register
        DBCI       /clear way for reoccurrence
        SPA        /was it event 0
        JMS SUB0   /yes, go service 0
        RAL        /no, shift left
        SPA        /was it event 1
        JMS SUB1   /yes, go service 1
        RAL        /no, shift left
        . . .
        RAL        /no, shift left
        SPA        /was it event 11
        JMS SUB11  /yes, go service 11
        JMP START  /no, go wait for another event
        SUB0, 0    /return location
        DCA SAVE   /save for further checking
        . . .
        (Service event)
        . . .
        CLA
        TAD SAVE   /get for further checking
        JMP I S0   /go check further
S1,
        . . .
```

Interface

The DR8-EA interfaces to the PDP-8/E OMNIBUS by plugging directly into the bus.

Interface to the outside world is by two (2) edge connectors on the M863 module. Signals leaving the board (12 bits parallel) are high (+3 volts) for a logical false and ground (0 volts) for a logical true. Each output line has approximately 20 milliamperes of drive (high level) and 20 milliamperes of sink (low level). Output levels remain fixed except when changed by the processor.

Signals entering from the "outside world" must be TTL in nature. The input represents approximately two (2) unit loads. When jumpered for "edge detection" a negative going edge (3 volts to 0 volts) is sensed. The signal must remain low (0 volts) for at least 50 NS. When sensing for an external level (jumpered so as to bypass the "flop") ground (0 volts) represents a logical true and a high (+3 volts) represents a logical false. With all bits jumpered this way the option represents a 12-bit parallel input register rather than an event detector.

An optional means of interfacing to the DR8-EA is available by using two (2) BC08J-X cables. Each cable (ribbon type) is terminated by a Berg type connector on one end (for interfacing to the DR8-EA module) and a standard DEC flip-chip on the other. One cable is used for the input and the other for output.

Cable Descriptions

The 7008418 cable is used to jumper the input to the output for diagnostic purposes. It is part of the DR8-EA option. If the user desires interface cables, the following can be purchased:
The BC08J cable consisting of the 1210073-0 connector, cable and the M953 module, and is available in several standard lengths.

Jumper Descriptions

The chart defined below will enable the user to change the IOT device code by changing the jumper across the specified split lug.

device selector		jumper		
(normal conf)	50	6H	7H	8H
	51	6H	7H	8L
	52	6H	7L	8H
	53	6H	7L	8L
	54	6L	7H	8H
	55	6L	7H	8L
	56	6L	7L	8H
	57	6L	7L	8L

The normal configuration will be factory installed with device selector code 50.

The input jumpers will be factory installed with A jumper, (edge triggered flip-flop). To change to level enables, use jumper B. The A,B, lugs are on all 12 bits.

Jumpers will also be provided to insulate the inputs from the interrupt and skip circuitry.

J2 — Input		J1 — Output	
D	— Bit 0	D	— Bit 0
F	— Bit 1	F	— Bit 1
J	— Bit 2	J	— Bit 2
L	— Bit 3	L	— Bit 3
N	— Bit 4	N	— Bit 4
R	— Bit 5	R	— Bit 5
T	— Bit 6	T	— Bit 6
V	— Bit 7	V	— Bit 7
X	— Bit 8	X	— Bit 8
Z	— Bit 9	Z	— Bit 9
BB	— Bit 10	BB	— Bit 10
DD	— Bit 11	DP	— Bit 11

Pin Connections

The output and input pins corresponding the AC bit enabled on the DR8-EA are as follows:

Input and Output End Pins (BC08J)

Bit 0 — B1	Gnds A1, C1, F1, K1,
Bit 1 — D2	N1, R1, T1, C2,
Bit 2 — D1	F2, J2, L2, N2,
Bit 3 — E2	R2, U2
Bit 4 — E1	
Bit 5 — H2	
Bit 6 — H1	
Bit 7 — K2	
Bit 8 — J1	
Bit 9 — M2	
Bit 10 — L1	
Bit 11 — P2	

LABORATORY MOUNTING PANEL

The laboratory peripheral panel is designed for compact yet versatile packaging of modular accessory epuipment for laboratory environments. The panel is a 19-inch rack-mounted unit with H945 panel mounting frame and housing that accepts plug-in type modules or module panels. Modules can be single-width, double-width, or other multiples of single-width, and may contain a printed circuit card mounted on the vertical dimension. Controls and input/output connectors for peripheral equipment are mounted on the module front panel. Modules or module panels are attached to the panel frame using one fastener at the top and bottom of the module panel.

The following options are available:

H945 Housing (Rack Mountable Chassis) for mounting laboratory peripherals including space for mounting 11 panel units; 5 single panel units; 3 double panel units, and a single 1½ panel unit filler panels.

H945-BA	115V Table Top Version.
H945-BB	230V Table Top Version.
H945-CA	115V Rack Mount Version.
H945-CB	230V Rack Mount Version.
DK8-EF	Optional panel for Real Time Clock type DK8-ES. Contains three 3-conductor phone jacks for event inputs and outputs; three one-turn potentiometers for voltage adjustments and three 4-pole double throw switches for line, or plus or minus voltage references.
AM8-EC	Analog input panel—16-channel A/D panel used for AM8-EA multiplexer inputs. Panel contains four 3-conductor phone jacks and four 10-turn vernier controls and 2 connectors. Panel repuires 3 single-panel-unit widths.
AM8-ED	Simple analog input panel 16-channel A/D panel used for AM8-EA multiplexer inputs. Panel contains two connectors and requires a single-panel-unit width.
VR03-A	Model 602 Tektronix Oscilloscope and VM03 Mounting Hardware.
VM03	Model 602 Tektronix Oscilloscope Mounting Hardware.

DB8-E INTERPROCESSOR BUFFER

The DB8-E interprocessor buffer allows two PDP-8/E's to transfer data between themselves or it may be used single ended as a data path between a PDP-8/E and user designed logic.

Device codes are jumper selectable between 50 and 57 allowing up to 8 DB8-E's to be connected to one PDP-8/E. The PDP-8/E's may be interconnected at distances up to 100 feet apart by means of two (2) BC08-R type cables.

All logic is mounted on a single QUAD size board which plugs directly into the OMNIBUS. Two (2) 40 pin connectors type H854 mounted on the module receive cable type BC08-R or BC08-J. On the terminal end of the cable, connector type H856 is provided.

SPECIFICATIONS

Maximum Transfer Rate	One 12-bit word at a maximum rate of approximately 5K Hz.
Physical Characteristics	The entire option is contained on one 8½" PDP-8/E QUAD module.
Temperature Operating Range	32°F to 131°F (0°C to 55°C)
Power Requirements	+5 volts at 600ma.
Data Format	12 parallel bits in and 12 parallel bits out.

PROGRAMMING

The following instructions are used for the DB8-E operation:

Skip on Receive Flag (DBRF)
Octal Code: 65X1
Operation: Skip if the Receive Flag equals one.

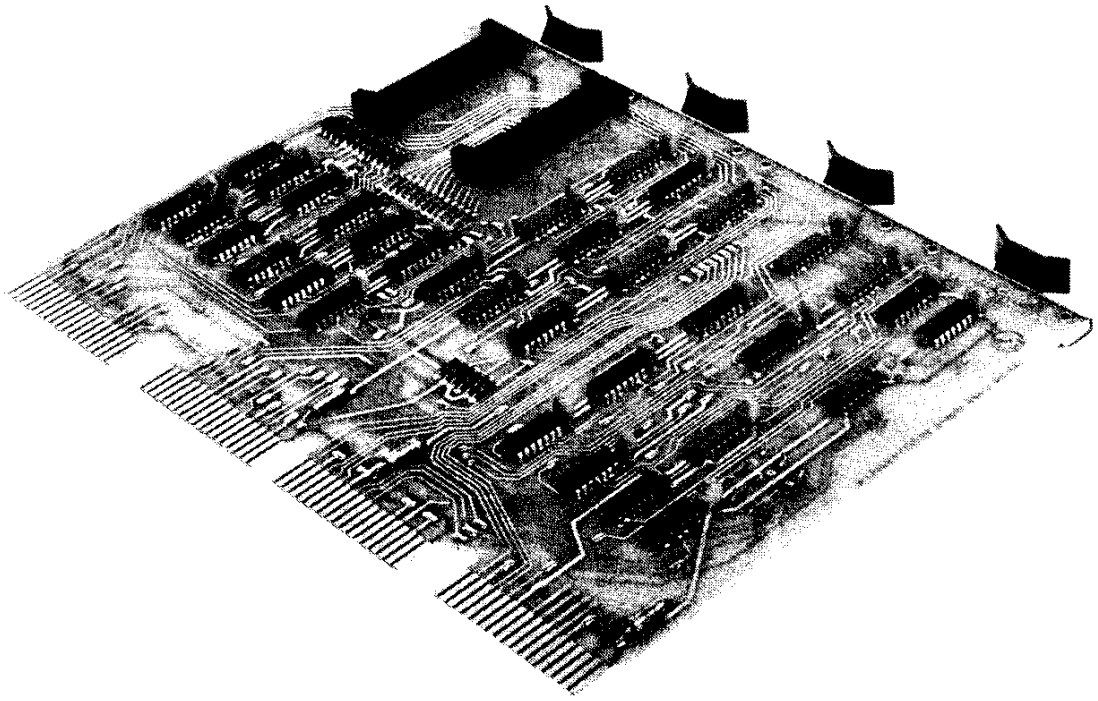
Read Incoming Data (DBRD)
Octal Code: 65X2
Operation: Read the Incoming Data into the AC and clear the Receive Flag.

Skip on Transmit Flag (DBTF)
Octal Code: 65X3
Operation: Skip if the Transmit Flag equals one.

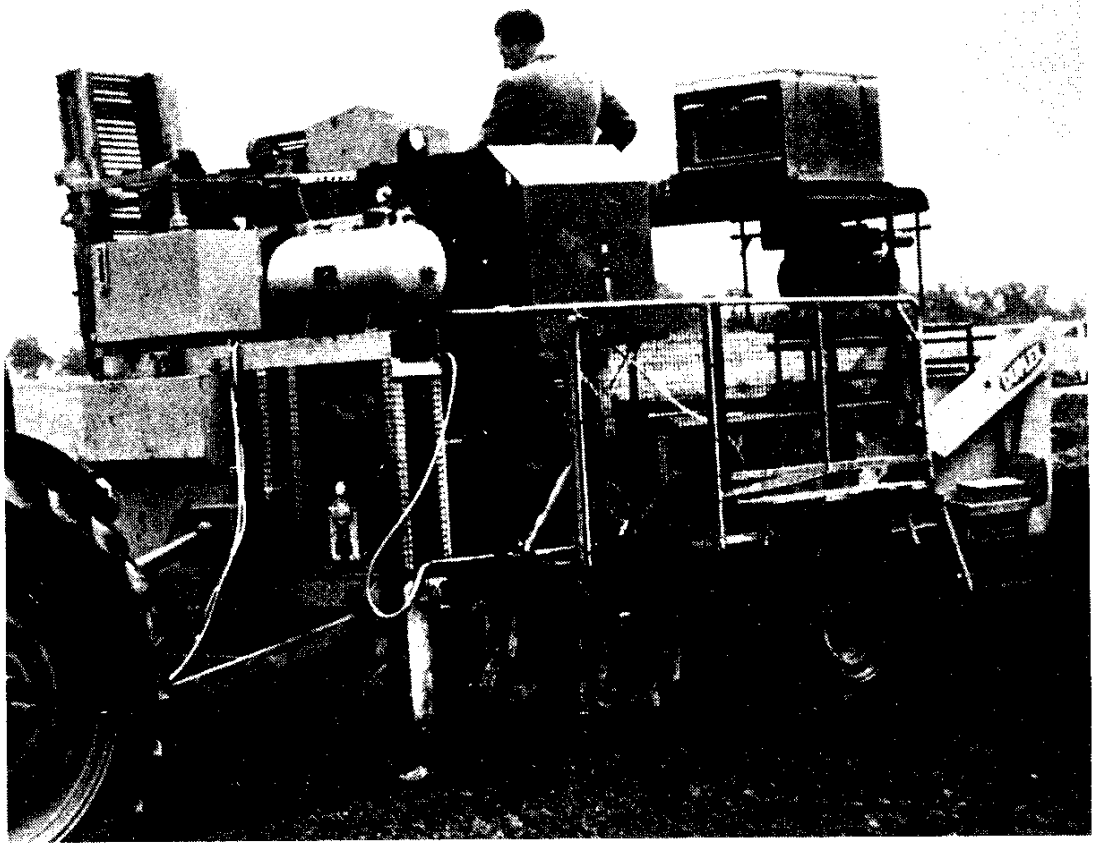
Transmit Data (DBTD)
Octal Code: 65X4
Operation: Transfer the contents of the AC Register to the Transmit Buffer. Transmit Data and set the Transmit Flag.

Enable Interrupt (DBEI)
Octal Code: 65X5
Operation: Enable the Interrupt Request Line.

Disable Interrupt (DBDI)
Octal Code: 65X6
Operation: Disable the Interrupt Request Line.



DB8-E interprocessor buffer



Instant evaluation of experimental farm machinery under test at the National Institute of Agricultural Engineering's Scottish Station near Edinburg, Scotland, is carried out by a PDP-8. The computer is mounted on a pre-production model of a potato picker (computer is in enclosure at technician's right). Picker is capable of rejecting all foreign matter from potato crop by X-ray techniques.

Besides the potato picker, the computer has been used to control various tractor-mounted farm equipment and a mechanical handling system. The PDP-8 gives the institute an instantly modifiable control and monitoring system, which may be connected without difficulty to *any piece of equipment. When not controlling and evaluating farm* equipment, it performs other functions.

SECTION 4 EXTERNAL BUS INPUT/OUTPUT EQUIPMENT OPTIONS

The following equipment options are provided for the External Bus. Positive I/O Bus Interface (KA8-E) is a prerequisite for external I/O transfer and the Data Break Interface (KD8-E) is a prerequisite for any external data break peripheral. Refer to Chapter 10 for the External Bus Interface Discussion.

EXTERNAL BUS INTERFACE CONTROL OPTIONS

The external bus interface options enable the PDP-8/E user to interface PDP-8/I and PDP-8/L type peripherals (such as mass storage devices, data acquisition, and control equipment) with the PDP-8/E. It also permits user-designed equipment to be interfaced with the 8/E external bus through the use of a general-purpose interface unit. A type KA8-E Positive I/O Bus Interface unit is required for any type of peripheral connected to the external bus. A type KD8-E Data Break Interface unit is required for each external peripheral that uses the data break facilities of the computer. These interface units and the BB08-P General Purpose Interface are described below. The detailed relationships of programmed I/O transfers and data break transfers are described in Chapter 10.

KA8-E Positive I/O Bus Interface

The KA8-E option enables the PDP-8/E user to interface PDP-8/I and PDP-8/L type peripherals with a PDP-8/E. This option converts OMNIBUS signals into positive programmed I/O bus signals used by PDP-8/I and PDP-8/L type peripherals. For example, 8/I and 8/L type peripherals require IOP pulses to perform their operations. The OMNIBUS does not generate internal IOP pulses, but does provide signals (MD bits 9-11) that can be converted to IOP pulses. Other signals normally required for programmed I/O transfers are also available on the OMNIBUS. The KA8-E merely buffers these signals and makes them available to the external bus at the correct time. Similarly, the KA8-E buffers peripheral inputs and makes them available to the OMNIBUS. A detailed description of the external bus interface, including signals, levels, timing relationships, and other interface data, is provided in Chapter 10.

Only one KA8-E can be used per machine. This module is required both for programmed I/O transfers and for external bus data break transfers. The KA8-E is also required whenever the BB08-P General Purpose Interface option is used, and when user-designed or user-installed logic is to be connected to the external bus. The KA8-E Positive I/O Bus Interface is contained on one PDP-8/E module that plugs into the OMNIBUS.

See Chapter 10 for details of interfacing with the external bus.

BB08-P General Purpose Interface Unit

The BB08-P General-Purpose Bus Interface provides the PDP-8/E user with the capability of interfacing user-designed or user-installed logic *with the PDP-8/E external bus. (The KA8-E Positive I/O Bus Interface* module is a prerequisite for using the BB08-P.)

The BB08-P can interface one receive (input) and one transmit (output) device, or two receive, or two transmit devices, and control related

transfers from program instructions. In addition, the unit can supply operating power for the user's device.

The BB08-P logic is housed in one prewired DEC type H943 Mounting Panel with a self-contained Type H716 Power Supply. There are 34 module sockets not used by option modules; thus, these sockets are available for user logic modules. The spare sockets, located in two adjacent rows, can accommodate 34 single-height modules, or 17 double height modules or combinations.

The basic data format for transfers is 12-bit parallel. The organization of fields within this format is at the user's discretion; however, user logic must operate according to the following rules:

a. Data user logic to computer, via BB08, must take inverted positive-bus form:
   0V (L) = logic true (1);
   +3V (H) = logic false (0).
b. Data from computer to user logic, via BB08, must be accepted in true positive-bus form:
   +3V (H) = logic true (1);
   0V (L) = logic false (0).
c. User logic must provide pulses to the BB08 to set the Transmit and Receive flags as required. These pulses must take the form of 0V to +3V transitions of not less than 100 ns duration. Rise and fall times of these pulses should be 150 ns or less.

The user may, at his discretion, use any or all of the following spare logic gates on modules of the BB08 option:

a. Four C/D flip-flops on the M216 or M206 Module at panel location A06.
b. Two TTL logic inverters on the M111 Module at panel location A07.
c. Eight open-collector bus drivers on the M623 Module at panel location 805.
d. Eight TLL two-input NAND gates on the M113 Module at panel location B06.

The BB08-P receive section consists of 12 level-converter gates, a device selector, and Receive Flag circuits. For transfers to the computer, the user device sets the Receive Flag to initiate a program interrupt for servicing the device. The computer then interrogates the skip chain by issuing Skip-on-External-Flag instructions. When a skip instruction with 37 (octal) is detected by the BB08, this device returns a skip pulse that causes a conditional jump in the computer. The program then clears the Receive Flag by issuing an IOT 6372 (octal) and transfers the input word to the computer with a 6374 (octal) instruction.

The BB08-P transmit section consists of 12 level-converter gates, a device selector, and Transmit Flag circuits. The most important difference between transmit and receive logic is that transfer from the buffered accumulator bus to the user's device is enabled whenever the BB08 device selector decodes 636X (octal).

Specifications

Data Format	12-bit parallel. Can be discrete bits or any organization of fields.
Receive/Input	TTL compatible of the inverted positive bus form: 0V (L) = logic 1 +3V (H) = logic 0
Transmit/Output	TTL compatible of the true positive bus form: +3V (H) = logic 1 0V (L) = logic 0
Power Supply Outputs (Available for User Logic)	3A at +5V 1.3A at + 15V
Power	115VAC, 60 Hz, 1A

Programming
The following instructions are used for BB08-P operation:

Skip On Transmit Flag (GTSF)

Octal Code:	6361
Execution Time:	2.6 μs
Operation:	Skips the next instruction if the Transmit Flag is set.

Clear Transmit Flag (GCTF)

Octal Code:	6362
Execution Time:	2.6 μs
Operation:	Resets the Transmit Flag.

(User Designated)

Octal Code:	6364
Execution Time:	2.6 μs
Operation:	This instruction is not used by BB08-P; however, the BB08-P decodes this IOT to make IOP4 available to user. User can use the IOP4 pulse to strobe data into his device.

Skip On Receive Flag (GRSF)

Octal Code:	6371
Execution Time:	2.6 μs
Operation:	Skips the next instruction if Receive Flag is set.

Clear Receive Flag (GCRF)

Octal Code:	6372
Execution Time:	2.6 μs
Operation:	Resets the Receive Flag.

Read Device Buffer (GRDB)

Octal Code: 6374
Execution Time: 2.6 μs
Operation: Transfers data from receive device to AC0-11.

KD8-E Data Break Interface

The KD8-E Data Break Interface option provides the PDP-8/E user with the one- and three-cycle data break facilities of the computer. Each KD8-E implements one of the 12 available data break channels of the PDP-8/E.

Each KD8-E contains the hardware to implement one standard data break channel and logic for establishing multiplexing priority between break devices. The KD8-E option is contained on one PDP-8/E module that plugs into the OMNIBUS.

Data break operations and the relationships of the KD8-E for these operations are described in detail in Chapter 10. Transfer time is 1.4 microseconds (715 kHz) for the single-cycle data break devices and 4.2 microseconds for 3-cycle data break devices.

RANDOM ACCESS DISK DEVICES

RK8 Disk System

The RK8 Disk System provides the PDP-8/E user with a modular, random-access, mass-storage device that utilizes removable disk cartridges as a storage medium. A basic system consists of one RK01 Disk Drive and Control and one RK08-P Disk Interface Control housed in a DEC H950 cabinet with self-contained power supplies, control, and indicator panel. The basic system provides the capability of storing 831,488 12-bit words and is readily expandable to over 3.3 million words in increments of 831,488 words. The RK08-P controls up to four RK01 units; thus, the expansion requires the addition of up to three RK01s.

Complete write protection is provided either on a sector basis, under manual/program control, or on whole disk basis, under manual control. This feature allows the programmer complete flexibility. Sectors of the disk containing system programs can be write-protected, while other sectors, containing data or new programs under development, can be write-permitted.

Sector numbers can be sequentially located on the disk, or scattered throughout a track. This feature allows for maximum throughput. Staggering of sector numbers throughout the track allows for computation time between data block transfers and enables the most efficient use of the system.

Complete transfer rate optimization is available. Consecutive data blocks can be transferred every 5 ms without specifying continuation. This feature eliminates the need to wait one disk revolution between blocks. A complete 4K of data can be transferred in just 80 ms.

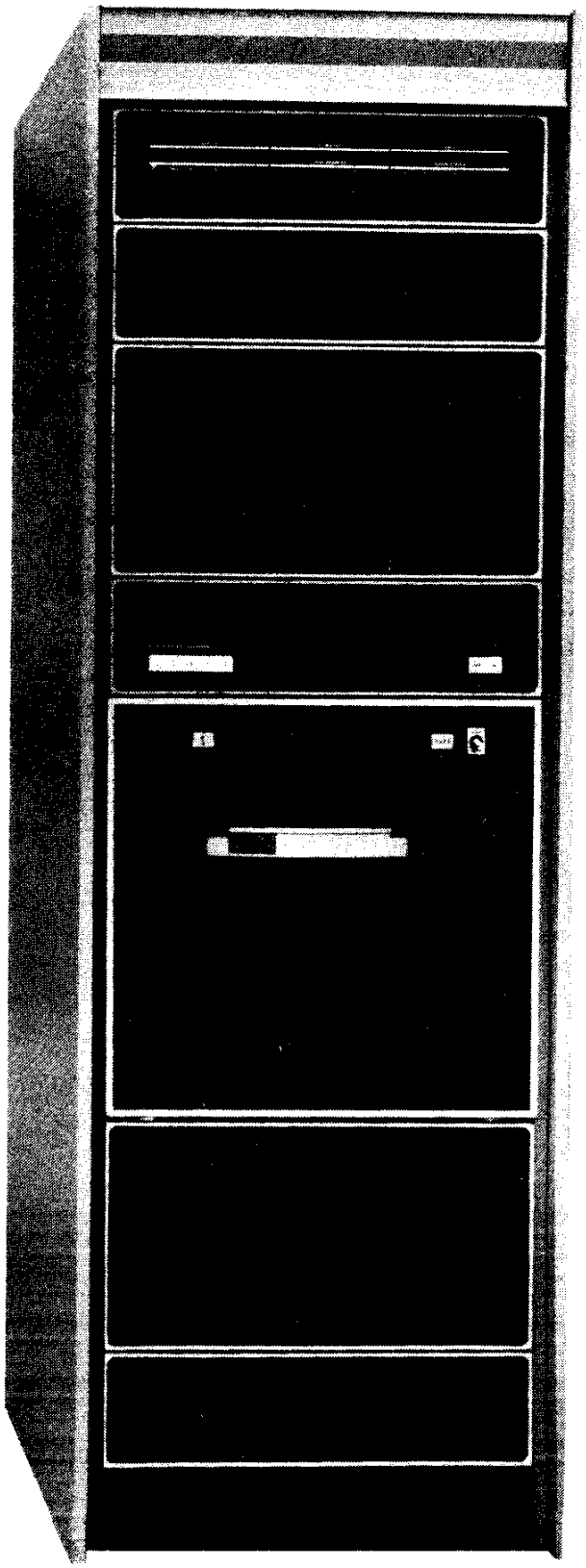
Complete track verification is performed with a preformatted cartridge. A separate leader containing track, surface, and sector address is written and verified by the format program, and checked on each transfer. This feature ensures positive track identification before a transfer is enabled.

Complete self-checking maintenance features are built in. These features permit fast and easy diagnosis of any errors.

Data transfers are performed using the single-cycle data break facilities of the PDP-8/E; thus, the KA8-E External Bus Interface and KD8-E Data Break Interface are prerequisites for using the RK8 on the PDP-8/E external bus.

Specifications

Disk Capacity:	
Storage Capacity:	Each RK01 Disk Cartridge stores 831,488 (includes three spare tracks) 12-bit words.
Expansion:	Four RK01 drives can be controlled by one RK08 control for a total of 3,325,952 words.
Data Tracks:	200 (plus three tracks)
Words per Track:	4096 (2048 words on 0 and 1 surfaces)
Sectors:	16



RK8 Disk System

Words per Sector:	256
Minimum Block Size:	256
Maximum Block Size:	4096

Read/Write Heads

Type = Tunnel erase
Number = Two (one per disk surface)

Recording Parameters

Recording Method:	Double Frequency—Time plus data
Density:	704 bits/inch (outer track) to 1026 bits/inch (inner track)
Speed:	1500 + or − 30 rpm
Transfer Path:	Single cycle Data Break
Transfer Rate:	16.7 μs per word
Minimum Access Time:	2 ms step plus 37 ms settle time (Adjacent Tracks)
Average Access:	134 ms (includes settle time)
Maximum Access:	441 ms (includes settle time)
Latency Maximum:	40 ms (one revolution)
Latency Average:	20 ms (½ revolution)
Program Interrupt:	Transfer Done Flag Track Found Flag (Enabled Separately) Error Flag (Enabled Separately)
Write Lock:	The two words at the beginning of each track contain the status information that is automatically checked for write lock of sectors.

RK01 Disk Drive and Control

The RK01 Disk Drive and Control contains the drive electronics and mechanism for accepting and releasing the disk, positioning the read/write heads, and controlling reading and writing of data from RK08-P commands.

The recording medium is a removable disk cartridge mounted in a protective case. The disk is an aluminum platter coated on both sides with magnetic oxide. When the cartridge is inserted into the drive mechanism, a read/write access door is automatically opened to permit positioning of the read/write heads over the disk surfaces. The disk is retained on its drive spindle by a magnet, and is driven counterclockwise at a uniform rate of 1500 rpm by a hysteresis synchronous drive motor. Information is recorded on or retrieved from both surfaces of the disk by the upper and lower read/write heads. These heads are cushioned from the disk surface by a film of forced air which keeps them between 125 and 160 microinches from the disk surface.

The upper surface of a disk mounted on the drive spindle is normally designated surface 0; the lower surface 1. Each disk surface is divided into 203 tracks (200 data tracks and 3 spares), with track 000 on the outer periphery and track 202 on the innermost portion of the recording area. Thus, for 203 combinations, eight bits must be allotted to address a track. The interval required for moving the read/write heads from one track to an adjacent track is approximately 2 ms; this means that, for a

move between the two possible extremes (track 000 and track 203), approximately 443 ms (including settling time) is required.

Each track is divided into 16 sectors with sectors 0 (octal) through 7 (octal) on the upper or 0 surface and sectors 10 (octal) through 17 (octal) on the lower or 1 surface. Sector assignments can be staggered as desired for the most efficient operation. Four bits (three for sector and one for surface) must be allotted for selection of a sector. A sector provides storage capabilities for two 12-bit header words, 256 12-bit data words and a 12-bit parity word. The first header word defines the track and sector address and the second word stores sector protect and status bits. The 12-bit parity word defines the longitudinal parity of the data words.

Information is recorded on the disk using the double-frequency or self-clocking method, in which a composite signal (generated from clock and data signals) is supplied to the read/write electronics. The clock frequency (720 kHz) determines the basic recording rate and cell width. For the recording of a logical zero, only the clock frequency is present in the composite signal and the results in a single change of direction in the magnetic flux pattern. However, for the recording of a logical one, a data pulse, occurring at twice the clock rate, is combined with the clock signal. This composite signal produces two changes in direction of the flux pattern. Thus, a component of the clock frequency is always present for timing purposes.

RK08-P Disk Interface Control

The RK08-P interfaces up to four RK01 Disk Drive and Control units with the external bus of the PDP-8/E. As part of this function, the RP08-P:

a. Decodes programmed IOT instructions.
b. Accepts and stores word count, current address, and command and address words.
c. Selects the program-designated RK01 and starts the program-designated operation.
d. Jointly (with the selected RK01) locates the disk address (track sector and surface).
e. Generates interrupt and break requests to initiate a single-cycle data break.
f. Buffers the input/output data and performs related conversion (serial-to-parallel and parallel-to-serial).
g. Performs housekeeping chores for single-cycle data break transfers (incrementing of word count and current address and providing address for transfer).
h. Generates and checks longitudinal parity for the sectors.
i. Provides flags for denoting current conditions and error status.
j. Provides logic for aiding in maintenance of an RK8 system.

Data transfers between the disk and core memory are implemented using the single-cycle data break facilities of the computer. (Refer to Chapter 10 for a detailed description of single-cycle data break.) For this operation, the computer specifies the number of words to be transferred (word count), and loads this information into the RK08-P word count

register by issuing a DLWC instruction. The first core memory address involved in the transfer is then loaded into the current address register by issuing a DLCA instruction. Next the computer loads a command word defining the selected disk, extended memory address (if this option is used), the interrupts it will accept, and the type of operation to be performed. This information is loaded into the command register of the RK08 by a DLDC instruction. Next the program loads the disk address (track, sector, and surface address) into corresponding registers in the RK08. This information defines disk starting location. If a read operation is to be performed, a DLDR instruction is issued to load the disk address and start the operation. Similarly, for a write operation, a DLDW instruction is issued. When the correct data track is located, a data break request is generated and the data break interface logic assumes control of the transfers. For this phase, the data break logic controls the loading and conversion of disk data words by asserting a B BREAK line to the RK08-P. Information read from the disk is converted from serial form for parallel transfer to the core memory.

This information is routed via external bus lines DATA00-11, gated by the Data Break Interface, and supplied to the MB via OMNIBUS lines DATA0-11. For write operations, information is accessed at the memory location specified by the current address. Data is provided to the RK08-P via the MD00-11 OMNIBUS lines and the Positive I/O Bus Interface (where it is buffered), and supplied to the RK08-P via the BMB00-11 external bus lines. The data is loaded into an RK08-P buffer under data break control and is shifted out to the disk read/write electronics. For each word transfer, the core memory address is specified by the current address register, and then the current address and word count registers are incremented. When the proper number of words have been transferred, as denoted by word count overflow, the RK08 generates an interrupt request. When the computer honors this request, it senses the transfer done flag of the RK08 with a DSKD instruction. If this flag is set, the computer senses the peripheral error flag with a DSKE instruction to determine if any errors were recorded during the transfers. If not, the transfer is assumed to be valid.

Programming

The following instructions are associated with the RK08-P control and the RK01 disk drive and control:

Load Disk Address (DLDA)—(Maintenance Only)

Octal Code:	6731
Execution Time:	2.6 μs
Operation:	Loads the Disk address for maintenance or diagnostic functions.

Load Command Register (DLDC)

Octal Code:	*6732*
Execution Time:	2.6 μs
Operation:	Loads the contents of AC0-11 into the Command Register and clears AC. The Command Register bit usage is shown in Figure 7-6.

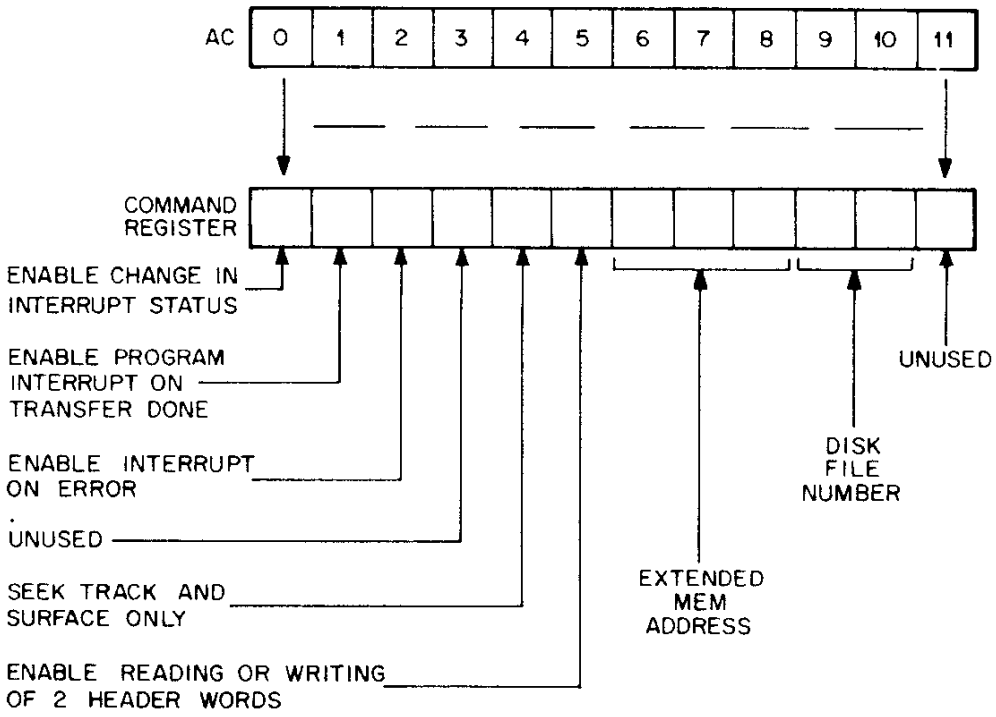


Figure 7-6 Command Word

Load Disk Address and Read (DLDR)

Octal Code: 6733
Execution Time: 3.6 μs
Operation: Loads track, surface, and sector address from AC, then clears AC. Starts read from disk if Command Register bit 4 is zero. If bit 4 is a one, instruction is executed only to seek track and surface. The relationship of bits in the AC transfer is shown in Figure 7-7.

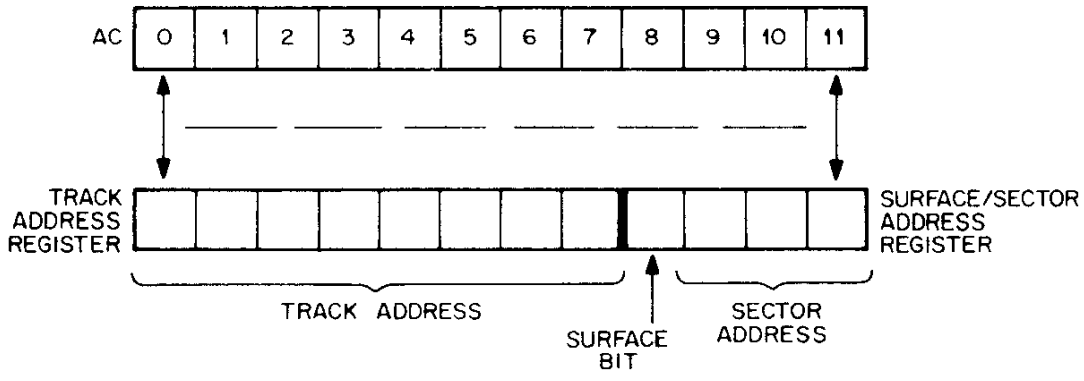


Figure 7-7 Address Word (Read)

Read Disk Address (DRDA)

Octal Code: 6734
Execution Time: 2.6 μs
Operation: Clears AC and reads content of Track Address Counter and Surface/Sector counters into AC0-11.

Load Disk Address and Write (DLDW)

Octal Code: 6735
Execution Time: 3.6 μs
Operation: Loads track, surface, and sector address from AC, then clears AC. Also starts a write operation if Command Register bit 4 is zero. If bit 4 is a one, instruction is executed only to seek track and surface.

Read Disk Command Register (DRDC)

Octal Code: 6736
Execution Time: 3.6 μs
Operation: Clears AC, then reads content of Command Register to AC0-11.

Load Disk Address and Check Parity (DCHP)

Octal Code: 6737
Execution Time: 4.6 μs
Operation: Loads track, surface, and sector address from AC, then clears AC. If bit 4 of Command Register is zero, reads data at specified address and checks parity. If bit 4 is one, instruction is executed to seek track and surface.

Read Disk Status Register (DRDS)

Octal Code: 6741
Execution Time: 2.6 μs
Operation: Clears AC, then reads content of Status Register into AC. Status bits are defined in Figure 7-8.

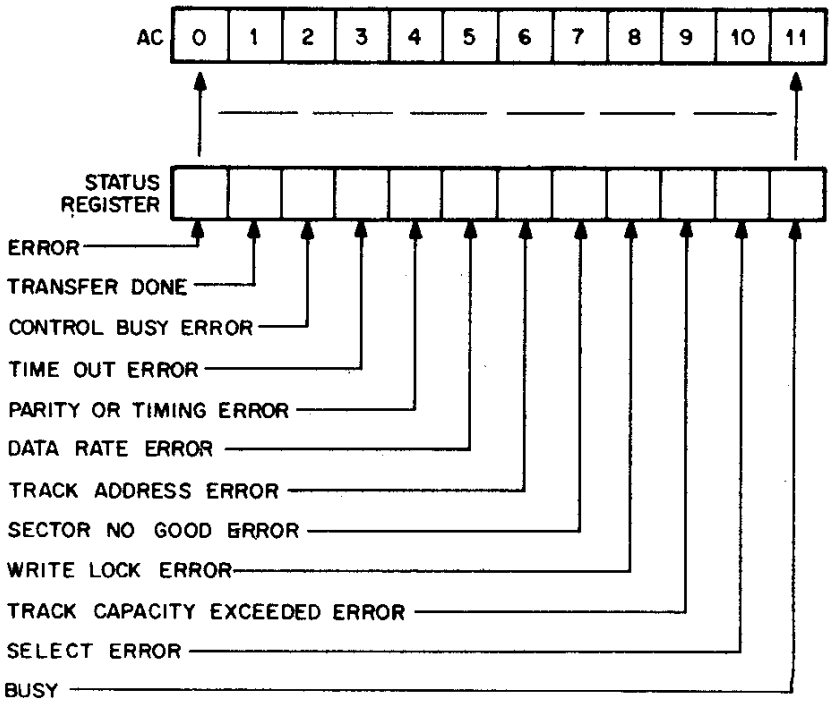


Figure 7-8 Status Word

Clear Status Register (DCLS)

Octal Code: 6742
Execution Time: 2.6 μs
Operation: Clears Status Register.

Load Maintenance Register (DMNT)

Octal Code: 6743
Execution Time: 3.6 μs
Operation: Loads the content of AC into Maintenance Register and performs specified operation. Bits remain in Maintenance Register until DMNT is issued with AC bits = 0, Maintenance functions are defined in Figure 7-9.

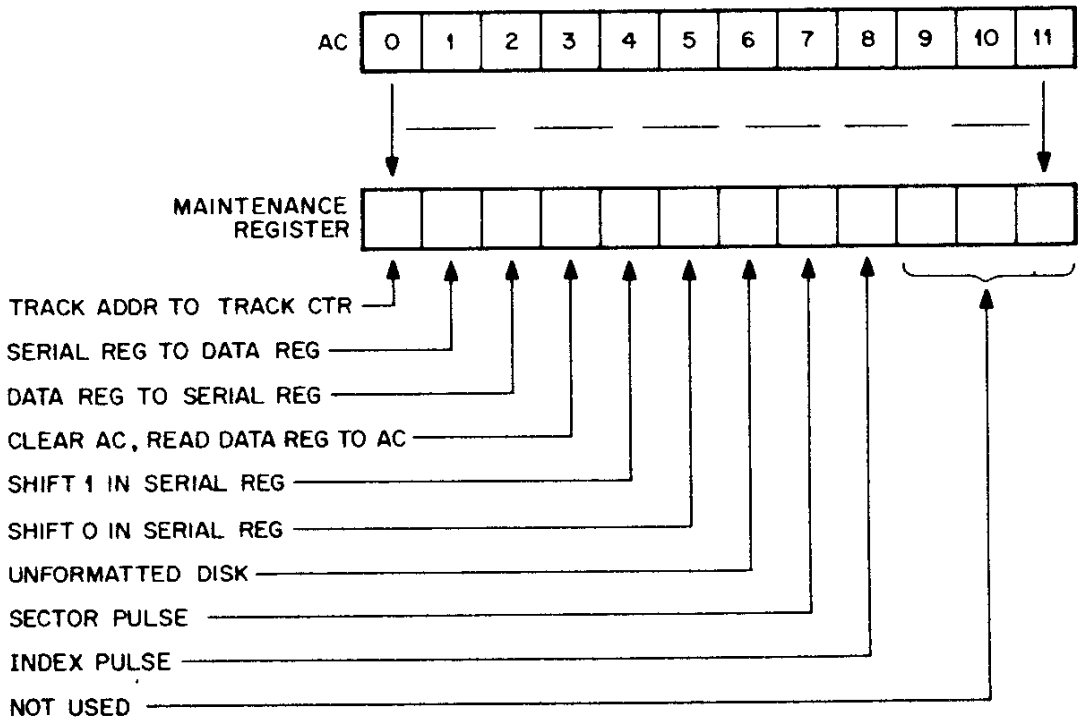


Figure 7-9 Maintenance Word

Skip on Disk Done (DSKD)

Octal Code: 6745
Execution Time: 3.6 μs
Operation: The Disk Transfer Done Flag is sensed; if it is set, the next instruction is skipped.

Skip on Disk Error (DSKE)

Octal Code: 6747
Execution Time: 4.6 μs
Operation: The Error Flag is sensed; if it is set, the next instruction is skipped.

Clear All (DCLA)

Octal Code: 6751
Execution Time: 2.6 μs
Operation: Clears selected disk to track 000, and clears all control registers and flags except Disk Selection. Transfer Done is set when disk is positioned to track 000.

Read Word Count Register (DRWC)

Octal Code: 6752
Execution Time: 2.6 μs
Operation: Clears the AC, then reads contents of WC register to AC0-11.

Load Word Count Register (DLWC)

Octal Code: 6753
Execution Time: 3.6 μs
Operation: Loads the contents of AC into WC register, then clears AC.

Load Current Address Register (DLCA)

Octal Code: 6755
Execution Time: 3.6 μs
Operation: Loads the contents of AC into CA register, then clears AC.

Read Current Address Register (DRCA)

Octal Code: 6757
Execution Time: 4.6 μs
Operation: Clears the AC, then reads contents of the CA register to AC0-11.

Example Subroutine

```
DISKIO,  0              /SUBROUTINE ENTRY POINT
         TAD CNT        /FETCH # OF WORDS TO BE READ
         DLWC           /LOAD WORD COUNT REGISTER
         TAD CUR        /FETCH ADD OF MEMORY BUFFER
         DLCA           /LOAD CUR ADDR REGISTER
         TAD 0000       /PUT COMMAND IN ACCUMULATOR
         DLDC           /LOAD COMMAND REGISTER
         TAD ADR        /FETCH TRACK, SURFACE, & SECTOR
         DLDR           /LOAD DISK ADR AND READ DATA
                        /DLDW WOULD HAVE WRITTEN
         DSKD           /DONE WITH TRANSFER?
         JMP.—1         /NO—GO BACK 1
         DSKE           /YES—IS THERE AN ERROR?
         JMP I DISKIO   /NO ERROR—RETURN
         JMP ERR        /GO TO ERROR SUBROUTINE
CNT,     XXX            /# WORDS TO BE READ
CUR,     YYY            /BEG ADDR OF MEMORY BUFFER
ADR,     ZZZ            /DISK ADDRESS
```

DF32-D DEC Disk File & Control & DS32-D DEC Disk File Expander

The DF32-D Disk File is a fast, low-cost, random-access, bulk-storage device and control for use with the PDP-8/E computer. [When the DF32-D is used with the PDP-8/E, the KD8-E Data Break interface and the KA8-E Positive I/O Bus interface are also required.] Operating through the three-cycle Data Break Facility, the DF32-D provides 32,768 13-bit words (12 bits plus parity) of storage, and is economically expandable to 131,072 words when using the DS32-D Expander Disk.

Transfer rate of the DF32-D is 32 or 64 μs per word (optional when timing track is written); average access time is 16.67 ms for 60 Hz power (20 ms with 50 Hz power).

Two basic assemblies make up the DF32-D; the storage unit with read/write electronics and computer interface logic. The storage unit contains a nickel-cobalt-plated disk, driven by a hysteresis synchronous motor. Data is recorded on a single disk surface by 16 fixed-position read/write heads.

Disk motor and shaft, read/write data heads, and timing and address heads are mounted on a 19-inch relay rack assembly, which permits easy access to the unit by sliding the unit in and out of a standard Digital Equipment Corporation cabinet.

The DS32-D Extender Disk File is also a slide-mounted assembly with a storage element and read/write electronics. Information transfers are made via the DF32-D logic, and are controlled by the DF32-D.

Specifications

Storage Capacity	32,768 13-bit words; expandble to 131,072 words in increments of 32,768 words, using DS32-D.	
Data transfer rate	60 Hz power	50 Hz power
	32 (64) μs per word	39 (78) μs per word
Average access time	16.67 ms	20.0 ms
Write lock switches	Inhibit writing on lower and/or upper 16K or any 32K disk surface; may be used to inhibit one or more 32K disks in an expanded configuration.	
Addressing Scheme	Random or absolute addressing from 0 to 32K words with variable block sizes from 1 word to 4096 words.	
Data assembly	Read/write on disk is serial, with external transfer parallel by word.	
Data Availability	16 μs (48 μs with alternate timing track) from the time word is assembled until new word starts to shift into assembly register. (A similar timing condition exists during the write operation.)	
Data tracks	16 per disk, 2048 words per track	
Recording method	NRZI	
Density (max)	1100 · BPI	
Timing tracks	2 plus 2 spare	

Size	10-½ in. high and 23-⅝ in. deep in a standard 10-in. rack.	
Heat Dissipation	1700 Btu/hr.	
Data Transfer Path	3-cycle Break	Address Locations 7750 Word Count 7751 Memory Address
Program Interrupt	Data Transfer-completion flag and/or non-existent disk.	
Write lock Switches	Inhibit write only on lower or upper 16K or both on one or more discs.	
Select Switches	Rotary Switches to select disk unit number.	

Programming

The following instructions operate the disk system:

Clear Disk Memory Address Register (DCMA)

Octal Code: 6601
Execution Time: 2.6 μs
Operation: Clears disk memory address register, parity error, and completion flags. This instruction also clears the disk memory request flag and interrupt flags.

Load Disk Memory Address Register and Read (DMAR)

Octal Code: 6603
Execution Time: 3.6 μs
Operation: Loads the content of the AC into the disk memory address register an clears the AC. This IOT initiates readings of information from the disk into the specified core location. Clears parity error and completion flags.

Load Disk Memory Address Register and Write (DMAW)

Octal Code: 6605
Execution Time: 3.6 μs
Operation: Loads the content of the AC into the disk memory address register and clears the AC. This disk then begins to write information into the disk from the specified core location. Clears parity error and completion flags. Data break must be allowed to occur within 33 μs (66 μs) after issuing this instruction

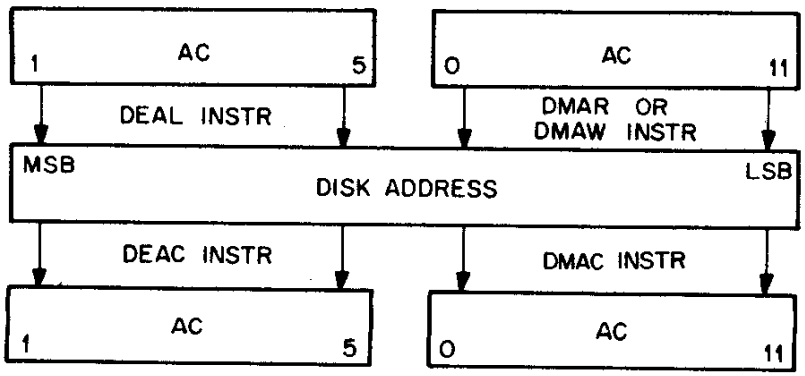


Figure 7-10 Address Words

Clear Disk Extended Address Register (DCEA)

Octal Code: 6611
Execution Time: 2.6 μs
Operation: Clears the disk extended address and memory address extension register.

Skip on Address Confirmed Flag (DSAC)

Octal Code: 6612
Execution Time: 2.6 μs
Operation: Skips next instruction if address confirmed flag is a one. Flag is set for 16 μs whenever the address on the disk equals the contents of the disk address registers. Clears the AC.

Load Disk Extended Address (DEAL)

Octal Code: 6615
Execution Time: 3.6 μs
Operation: Clears the disk extended address and memory address extension registers and loads them with the track address data held in the AC. ORs the contents of these registers, plus the photocell mark and three error flags, into the AC. (See DEAC instruction.)

Read Disk Extended Address Register (DEAC)

Octal Code: 6616
Execution Time: 3.6 μs
Operation: Clears the AC, then loads the contents of the disk extended address register into the AC to allow program evaluation. Skips the next instruction if address confirmed flag is a one.

NOTE

Write lock switch status is true only when disk unit contains a write command. The nonexistent disk condition will appear following the completion of a data transfer during read, where the address acknowledged was the last address of a disk and the next word to be addressed falls within a nonexistent disk. The completion flag for this data transfer is set by the nonexistent disk condition 16 μs after the data transfer.

Skip On Zero Error Flag (DFSE)

Octal Code: 6621
Execution Time: 2.6 μs
Operation: Skips the next instruction if parity error, data request late, and write lock switch flag are all zero. Indicates no errors.

Skip on Data Completion Flag (DFSC)

Octal Code: 6622
Execution Time: 2.6 μs
Operation: Skips the next instruction if the completion flag is a one, indicating data transfer is complete.

Read Disk Memory Address Register (DMAC)

Octal Code: 6626
Execution Time: 3.6 μs
Operation: Clears the AC, then loads the contents of the disk memory address dress register into the AC to allow program evaluation. During read, the final address will be the last one transferred.

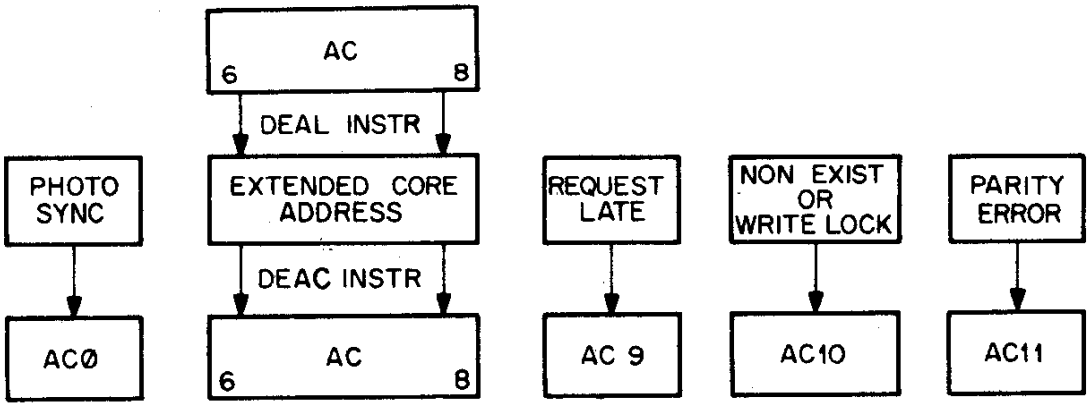


Figure 7-11

Three-cycle data break locations: Work Count address is 7750 (field 0). Current Address is 7751 (field 0).

Three maintenance IOTs are also used by the DF32-D. These IOTs are used to simulate certain pulses within the disk control for static logic tests. Since they all use device code 63, this code should not be used by other peripheral devices when a DF32-D is part of the system.

NOTE

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

Bits 1-5 (DEAL, DEAC Inst.)	+ Accumulator (Low Order 12 Bits) 0-11 of DMAW or DMAR	→	Disk Address (17 Bit)
Field Bits 6-8 (DEAL, DEAC 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 address are manipulated by the DEAL and DEAC instructions. Low order bits are manipulated in the AC.

TYPE RF08 DISK FILE AND CONTROL AND TYPE RS08 EXPANDER DISK FILE

The RF08 control and the RS08 disk combine to provide fast, low-cost, random access, bulk storage for the computer. One RF08/RS08 provides 262,144 13-bit words of storage. Up to four RS08 disks can be added to the RF08 control for a total of 1,048,576 words of storage. Data is recorded on a single disk surface by 128 fixed read/write heads.

Data transfer is accomplished through the three-cycle break system of the computer and its associated required options, which are the same as for the DF32/DS32 system. Fast track-switching time permits spiral read or write. Data may be read or written in blocks of from 1 to 4096 words. Transfers accross disks are handled automatically by the control unit.

RF08/RS08 Specifications

Disks	Four RS08s may be controlled by one RF08 for 1,048,576 words.	
Storage Capacity	Each RS08 stores 262,144 13-bit words (12 plus one even parity bit)	
Data Transfer Path	3-Cycle Break	Address Locations 7750 Word Count 7751 Current Address
Data Transfer Rate	60 Hz Power 16.0 μs per word	50 Hz Power 19.2 μs per word
Minimum Access Time	258 μs	320 μs
Average Access Time	16.9 ms	20.3 ms
Maximum Access Time	33.6 ms	40.3 ms
Program Interrupt	33 ms Clock Flag Data Transmission Complete Flag Error Flag	
Write Lock Switches	Eight switches per disk capable of locking out any combination of eight 16,384 word blocks in addresses 0 to 131,071.	
Data Tracks	128	
Words Per Track	2048	
Recording Method	NRZ1	
Density	1100 bpl Maximum	
Timing Tracks	3 plus 3 spare (spares can be used to recover data on disk)	

RF08/RS08 Specifications (Cont)

Operating Environment	Recommended temperature 65° to 90°F.
Vibration/Shock	Good isolation is provided. To prevent data errors, extreme vibrations should be avoided while the RS08 is transferring information.
Heat Dissipation	RF08: 150W RS08: 300W
AC Power Requirements	115/230 ± 10% Vac, single phase, 50 ± 2 or 60 ± 2 Hz, 5A (maximum) for logic power. (Logic power for one RF08 and up to four RS08s is provided by one DEC Type 705B Power Supply) Additional line current is required for RS08 disk motor as shown below.
RS08 Motor Power Requirements	Motor start, 5.5A for 20 ± 3s. Motor run, 4.0A continuous @ 115 Vac. (A stepdown autotransformer is provided for 230 Vac operation).
Line Frequency Stability	Maximum line frequency drift 0.1 Hz/s. A constant frequency motor-generator set or static ac/ac inverter should be provided for installation with unstable power sources.
Motor Bearing Life	Expected operating life of at least 20,000 hours, under standard computer operating environment.
Reliability	Six recoverable errors and one nonrecoverable error in 2 x 10^9 bits transferred. A recoverable error is defined as an error that occurs only once in four successive reads. All other errors are nonrecoverable. On-off cycling of the RS08 is not recommended. For this reason, the RS08 motor control operates independently of the computer power control.
Cabinet	A dedicated cabinet is designed to accommodate one RF08, up to two RS08s and power supply. Two additional RS08s can be mounted in a second cabinet. Other equipment should not be mounted in disk cabinets.
Shipping Information	Weight of RF08, one RS08, power supply and cabinet: 590 lb (crated) 500 lb (uncrated) Weight of RF08, two RS08, power supply and cabinet:

Programming Instructions

The programming instructions for the RFO8/RSO8 differ slightly from those provided in the DF32/DS32 description. The extended address capability and associated instructions (DCEA, DEAL, and DEAC) are replaced, in sequence, by interrupt enable and memory address extension register instructions (DCIM, DIML, and DIMA).

Clear Disk Interrupt Enable and Core Memory Address Extension Register (DCIM)

Octal Code: 6611
Event Time: 1
Indicators: Iot, Fetch, Pause [IR = 6, F]
Execution Time: 4.25 μs
Operation: Clear the disk interrupt enable (DIE) and core memory address extension (MAE) registers.
Symbol: 0 → DIE, 0 → MAE

Load Interrupt Enable and Memory Address Extension Register (DIML)

Octal Code: 6615
Event Time: 1, 3
Indicators: Iot, Fetch, Pause [IR = 6, F]
Execution Time: 4.25 μs
Operation: Clear the interrupt enable (IE) and MAE, then load the interrupt enable and 'memory address extension registers with data held in the AC. Then clear AC.

NOTE

Transfers cannot occur across memory fields. Attempts to do so will cause the transfer to "wrap around" within the specified memory field.

Symbol: 0 → IE, 0 → MAE
AC 3-15 → IE, AC 6-8 → MAE
0 → AC
AC TO DISK STATUS REGISTER

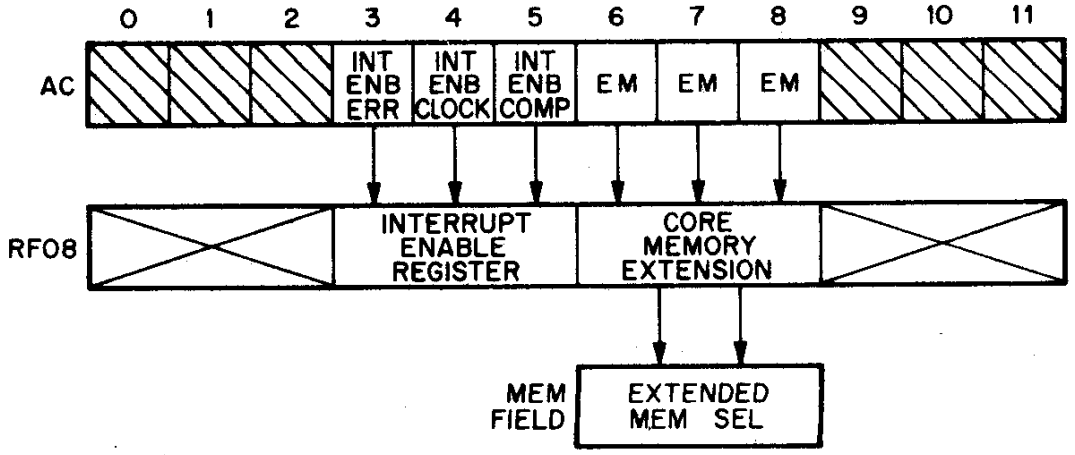


Figure 7-12 AC TO DISK STATUS REGISTER

Load Interrupt and Extended Memory Address (DIMA)
Octal Code: 6616
Event Time: 2, 3
Indicators: Iot, Fetch, Pause [IR = 6, F]
Execution Time: 4.25 μs
Operation: Clear the AC. Then load the contents of the status register (STR), into the AC to allow program evaluation.

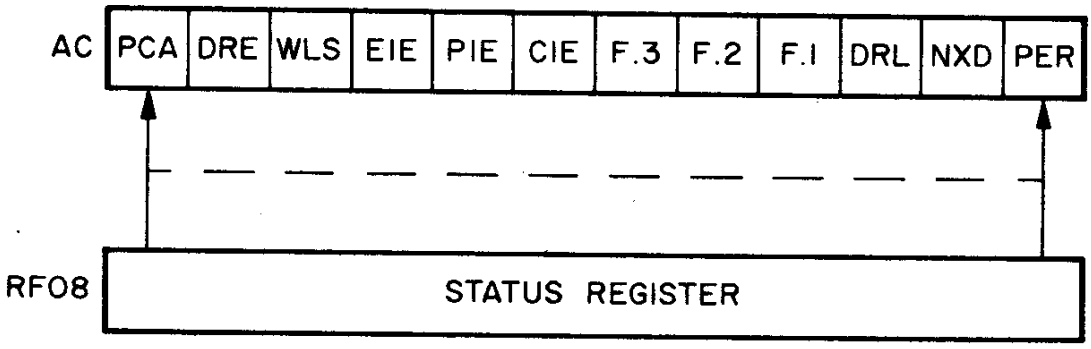


Figure 7-13 DISK TO AC STATUS REGISTER

AC Bit	Abbr.	Description
0	PCA	Photocell Sync Mark (available 100 μs status)
1	DRE	Data Request Enable (maintenance only status)
2	WLS	Write Lock Status
3	EIE	Error Interrupt Enable
4	PIE	Photocell Interrupt Enable
5	CIE	Completion Interrupt Enable
6-8	F	(FIELD) Core Memory Extension Fields
9	DRL	Data Request Late
10	NXD	Nonexistent Disk
11	PER	Parity Error
Symbol:	*0 → AC*	
	STR → AC	

In addition to these changes in instructions, the RF08/RS08 utilizes six additional instructions: DFSE, DISK, DCXA, DXAL, DXAC, and DMMT.

Skip on Disk Error (DFSE)
Octal Code: 6621
Event Time: 1
Indicators: Iot, Fetch, Pause [IR = 6, F]
Execution Time: 4.25 μs
Operation: Skip next instruction if there is parity error, data request late, write lock status, or nonexistent disk flag set.
Symbol: Parity error, data request late, write lock status, or nonexistent disk flags are set, PC + 1 → PC.

Skip Error or Completion Flag (DISK)
Octal Code: 6623
Event Time: 2
Indicators: Iot, Fetch, Pause [IR = 6, F]
Execution Time: 4.25 μs
Operation: If either the error or data completion flag (or both) is set, the next instruction is skipped.
Symbol: If PER or Data Complete, PC + 1 → PC.

Clear High Order Address Register (DCXA)
Octal Code: 6641
Event Time: 1
Indicators: Iot, Fetch, Pause [IR = 6, F]
Execution Time: 4.25 μs
Operation: Clear the high order 8-bit disk address register (DAR).
Symbol: 0 → DAR

Clear and Load High Order Address Register (DXAL)
Octal Code: 6643
Event Time: 1, 2
Indicators: Iot, Fetch, Pause [IR = 6, F]
Execution Time: 4.25 μs
Operation: Clear the high order 8 bits of the DAR. Then load the DAR from data stored in the AC. Then clear AC.
Symbol: 0 → DAR high order 8 bits,
AC → DAR,
0 → AC

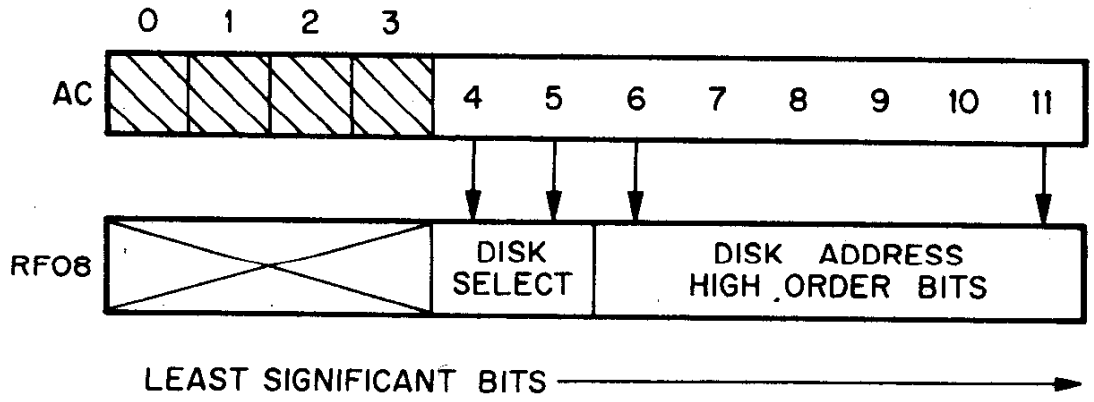


Figure 7-14 Higher Order Address Word Transfer

Clear Accumulator and Load DAR Into AC (DXAC)
Octal Code: 6645
Event Time: 1, 3
Indicators: Iot, Fetch, Pause [IR = 6, F]
Execution Time: 4.25 μs
Operation: Clear the AC; then load the contents of the high order 8-bit DAR into the AC.
Symbol: 0 → AC,
DAR high order 8 bits → AC

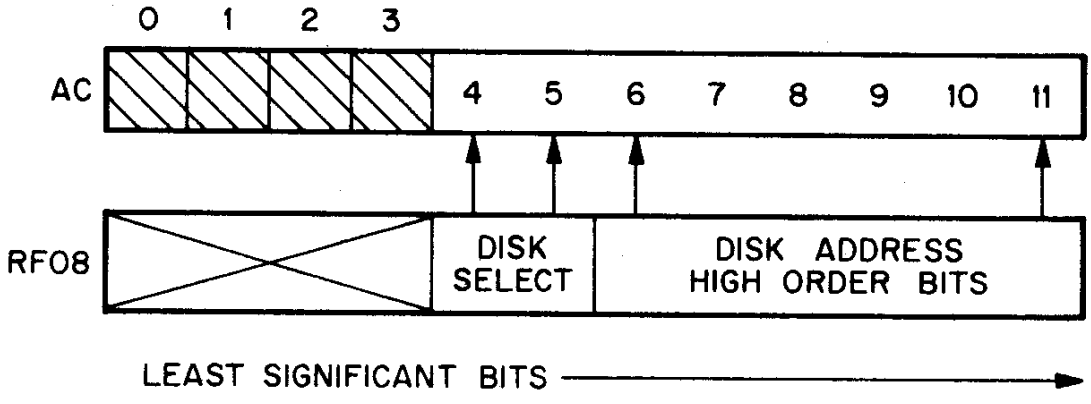


Figure 7-15 Disk Address Transfer to AC

Initiate Maintenance Register (maintenance purposes only) (DMMT)

Octal Code: 6646
Event Time: 2, 3
Indicators: Iot, Fetch, Pause [IR = 6, F]
Execution Time: 4.25 μs
Operation: For maintenance purposes only with the appropriate maintenance cable connections and the disk disconnected from the RS08 logic, the following standard signals may be generated by IOT 66-46 and associated AC bits. AC is cleared and the maintenance register (MAIR) is initiated by issuing an IOT 6601 command.

AC (1) Track A Pulse
AC (1) Track B Pulse
AC (1) Track C Pulse
AC (1) DATA PULSE (DATA HEAD #0)
AC (1) Photocell
AC (1) DBR
Setting DBR to a 1 causes data break request in computer.

Symbol: AC → MAIR

Three-cycle data break locations: word count address is 7550 (field 0), current address is 7751 (field 0).

DF32 Programming Compatibility

The IOT instructions 660X and 6622 are identical in every respect to the DF32 instruction; i.e., the same operations are performed. The 661X and 662X instructions differ only in the following:

a. OIT 6615 does not transmit the extended disk address bits for addressing over 32K; instead, AC 3-5 are assigned to enable or disable conditions on the program interrupt line. The AC is cleared upon execution of this instruction.

b. IOT 6616 no longer reads back the extended address bits by 1 through 5 into the AC. These bits are assigned to examine the status of interrupt enable. In addition, AC2 indicates the status of write lock and AC10 shows only nonexistent disk conditions. AC1 shows the condition of data request enable used for maintenance purposes.

c. IOT 6621 has been changed to skip on error rather than no-error. Non-existent disk has been included as an error skip condition.

d. IOT 6623 (DISK) is a new skip instruction that will skip on either error or completion flags or both.

The DF32 maintenance instruction IOT 663X is not assigned to the RF08 system.

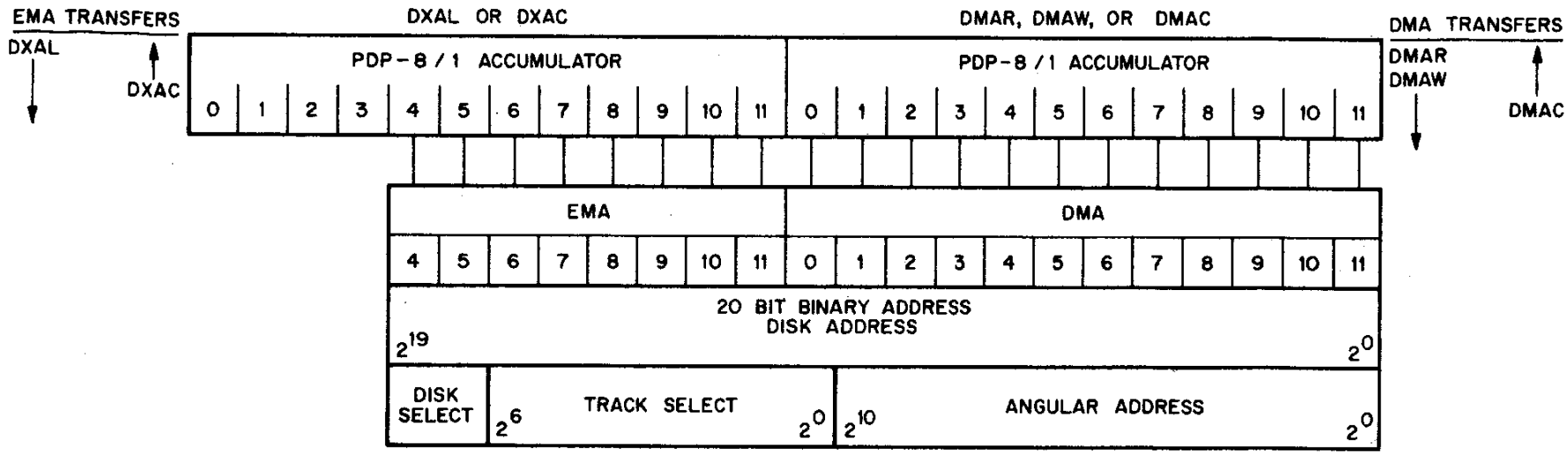


Figure 7-16 RF08 Addressing Format

Programming Example

Software

A sample of a typical I/O routine for the RF08/RS08 is as follows:

```
0200  4777           JMS I   (DISKIO)
0201  0000  FUNCT,    0               /X0=READ, X1=WRITE (X=0=7 MEMORY FIELD)
0202  0000  WDCT,     0               /+ WORD COUNT
0203  0000  CORE,     0               /CORE LOCATION
0204  0000  DSKHI,    0               /HIGH ORDER 8 BITS
0205  0000  DSKLOW,   0               /LOW ORDER 12 BITS
0206  5020           JMP ERROR               /ERROR RETURN (AC=ERROR CONDITION)
                                      /NORMAL RETURN (AC=0)

0207  0000  DISKIO,   0
0210  7300           CLL CLA
0211  1607           TAD I DISKIO
0212  6615           DIML            /LOAD EXTENDED MEMORY BITS
0213  1607           TAD I DISKIO
0214  0376           AND (7
0215  7640           SZA CLA

0216  7126           STL RTL         /+2
0217  1375           TAD (3
0220  1374           TAD (6600
0221  3236           DCA RORW                /6603=READ, 6605=WRITE
0222  2207           ISZ DISKIO
0223  1607           TAD I DISKIO
0224  7041           CIA
0225  3773           DCA I (7750     /STORE=WORD COUNT
0226  2207           ISZ DISKIO
0227  1607           TAD I DISKIO
0230  3772           DCA I (7751)    /LOAD CORE ADDRESS
0231  2207           ISZ DISKIO
0232  1607           TAD I DISKIO
0233  6643           DXAL            /LOAD HIGH ORDER 9
                                      /BITS OF DISK ADDRESS,

0234  1607           TAD I DISKIO
0235  2207           ISZ DISKIO

0236  0000  RORW, .   0               /READ OR WRITE

0237  6623           DISK            /DONE?
0240  5237           JMP .-1         /NO
0241  6621           DFSE            /YES, ERROR?
0242  2207           ISZ DISKIO      /SKIP TO NORMAL RETURN
0243  5607           JMP I DISKIO    /RETURN


      6615  DIML = 6615
      6623  DISK = 6623
      6643  DXAL = 6643
      6621  DFSE = 6621
      0020  ERROR = 20


0372  7751
0373  7750
0374  6600
0375  0003
0376  0007
0377  0207


CORE    0203
DFSE    6621
DIML    6615
DISK    6623
DISKIO  0207
DSKHI   0204
DSKLOW  0205
DXAL    6643
ERROR   0020
FUNCT   0201
RORW    0236
WDCT    0202
```

MAGNETIC TAPE OPTIONS

The External Bus Magnetic Tape Options include:

a. The TU56 Dual DECtape Transport and TC08 DECtape control,
b. The TU10 DECMAGtape Transport and TC58 Automatic Magnetic Tape Control.

DECtape

The DECtape system is a standard option for the PDP-8/E that serves as an auxiliary magnetic tape data storage facility. The DECtape system stores information at fixed positions on magnetic tape, as in magnetic disk or drum storage devices, rather than at unknown or variable positions, as in conventional magnetic tape systems. This feature allows replacement of blocks of data on tape in a random fashion without disturbing other previously recorded information. In particular, during the writing of information on tape, the system reads format (mark) and timing information from the tape and uses this information to determine the exact position at which to record the information to be written. Similarly, in reading, the same mark and timing information has a number of features to improve its reliability and make it exceptionally useful for program updating and program editing applications. These features are: phase or polarity sensed recording on redundant tracks, bidirectional reading and writing, and a simple mechanical mechanism utilizing hydrodynamically lubricated tape guiding (the tape floats on air over the tape guides while in motion).
Four basic DECtape configurations are identified in the following table.

SYSTEM DESIGNATION	DECtape	CONTROL	PREREQUISITE	REMARKS
None	TU56 (Dual Drive)	TC08	KA8-E KD8-E PDP-8/E	Up to 4 Dual TU56's per control. (8 drive units)
None	TU56 (Single Drive)	TC08	KA8-E KD8-E PDP-8/E	Up to 4 single DECtape drive units.
TD8-EM	TU56 (Dual Drive)	TD8-E	PDP-8/E	Up to 4 Dual Drive TU56's per control. (8 drive units) Control plugs into OMNIBUS.
TD8-EA	TU56H (Single Drive)	TD8-E	PDP-8/E	Up to 4 single drive units. Control plugs into OMNIBUS.

Magnetic tape options operated on the external bus of the PDP-8/E require the use of the KA8-E Positive I/O Bus Interface module and the KD8-E Data Break Interface module as prerequisites.

DECtape Format

DECtape utilizes a 10-track read/write head. Tracks are arranged in five nonadjacent redundant channels: a timing channel, a mark channel, and three information channels. Redundant recording of each character bit on nonadjacent tracks materially reduces bit dropouts and minimizes the effect of skew. The series-connection of corresponding track heads within a channel and the use of Manchester phase recording techniques, rather than amplitude sensing techniques, virtually eliminate dropouts.

The timing and mark channels control the timing of operations within the control unit and establish the format of data contained on the information channels. The timing and mark channels are recorded prior to all normal data reading and writing on the information channels. The timing of operations performed by the tape drive and some control functions are determined by the information on the timing channel. Therefore, wide variations in the speed of tape motion do not affect system performance. Information read from the mark channel is used during reading and writing data to indicate the beginning and end of data blocks and to determine the functions performed by the system in each control mode. During normal data reading, the control assembles 12-bit computer-length words from four successive lines read from the information channels of the tape. During normal data writing, the control disassembles 12-bit words and distributes the bits so they are recorded on four successive lines on the information channels. A mark-channel error-check circuit ensures that one of the permissible marks is read in every six lines on the tape. This 6-line mark-channel sensing requires that data be recorded in 12-line segments (12 being the lowest common multiple of 6-line marks and 4-line data words) which correspond to three 12-bit words.

A tape contains a series of data blocks that can be of any length which is a multiple of three 12-bit words. Block length is determined by information on the mark channel. A uniform block length is usually established over the entire length of a reel of tape by a program that writes mark and timing information at specific locations. The ability to write variable-length blocks is useful for certain data formats. For example, small blocks containing index or tag information can be alternated with large blocks of data. (Software supplied with DECtape allows writing for fixed block lengths only.)

Between the blocks of data are areas called interblock zones. The interblock zones consist of 30 lines on tape before and after a block of data. Each of these 30 lines is divided into five 6-line control words. These 6-line control words allow compatibility between DECtape written on any of DEC's 12-, 18-, or 36-bit computers. As used on the PDP-8/E, only the last four lines of each control word are used.

Block numbers normally occur in sequence from 1 to n. There is one block numbered 0 and one block n + 1. Programs are entered with a statement of the first block number to be used and the total number of blocks to be read or written. The total length of the tape is equivalent to 849,036 lines, which can be divided into any number of blocks up to 4096 by prerecording of the mark track. The maximum number of blocks is determined by the following equation in which n(b) equals number of

blocks and n(w) equals number of words per block (n(w) must be divisible by 3).

$$n(b) = \frac{212112}{n(w) + 15} - 2$$

DECtape format is illustrated in Figures 7-17 through 7-20.

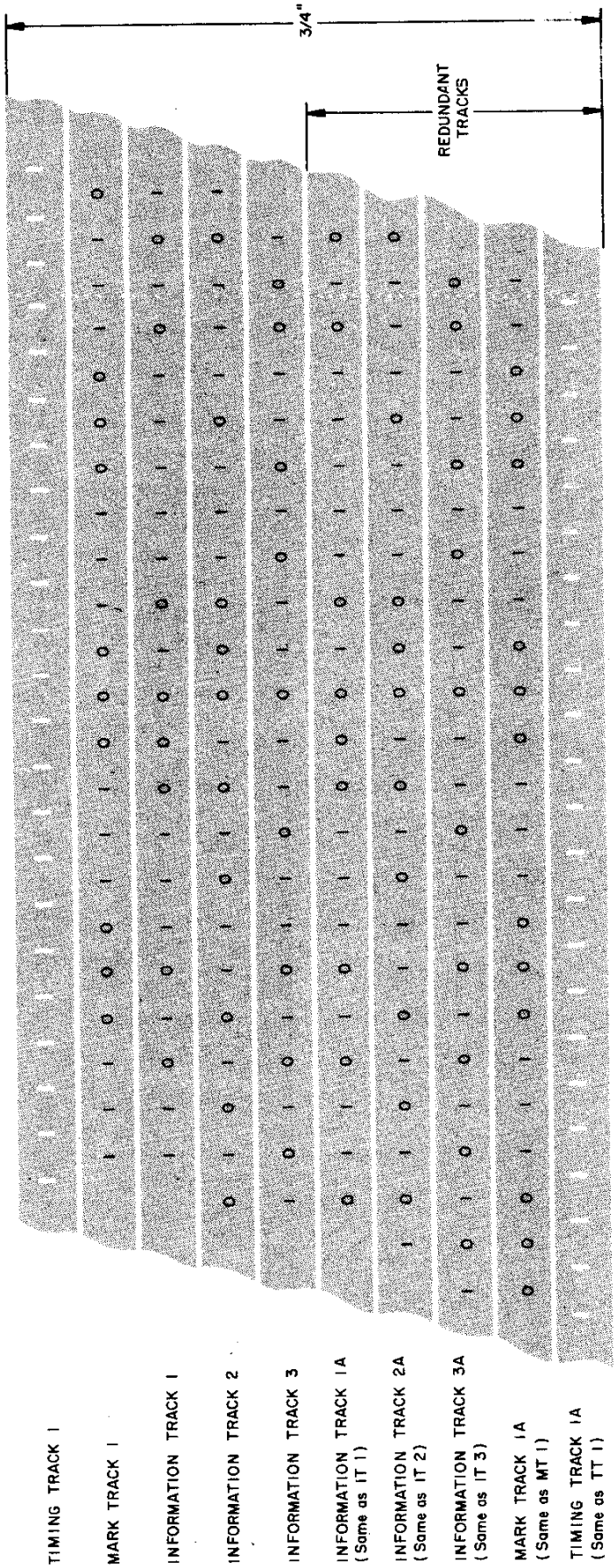


Figure 7-17 DECtape Track Allocations

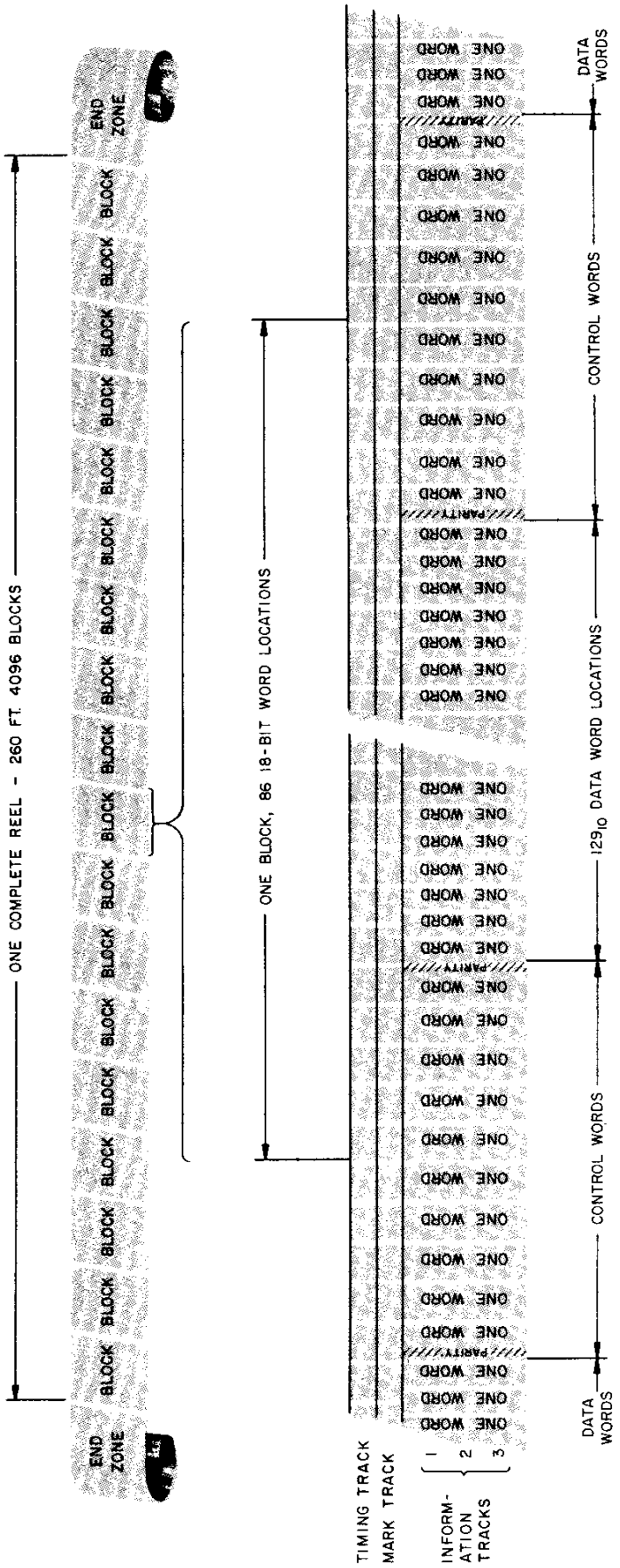


Figure 7-18 DECtape Mark Channel Format

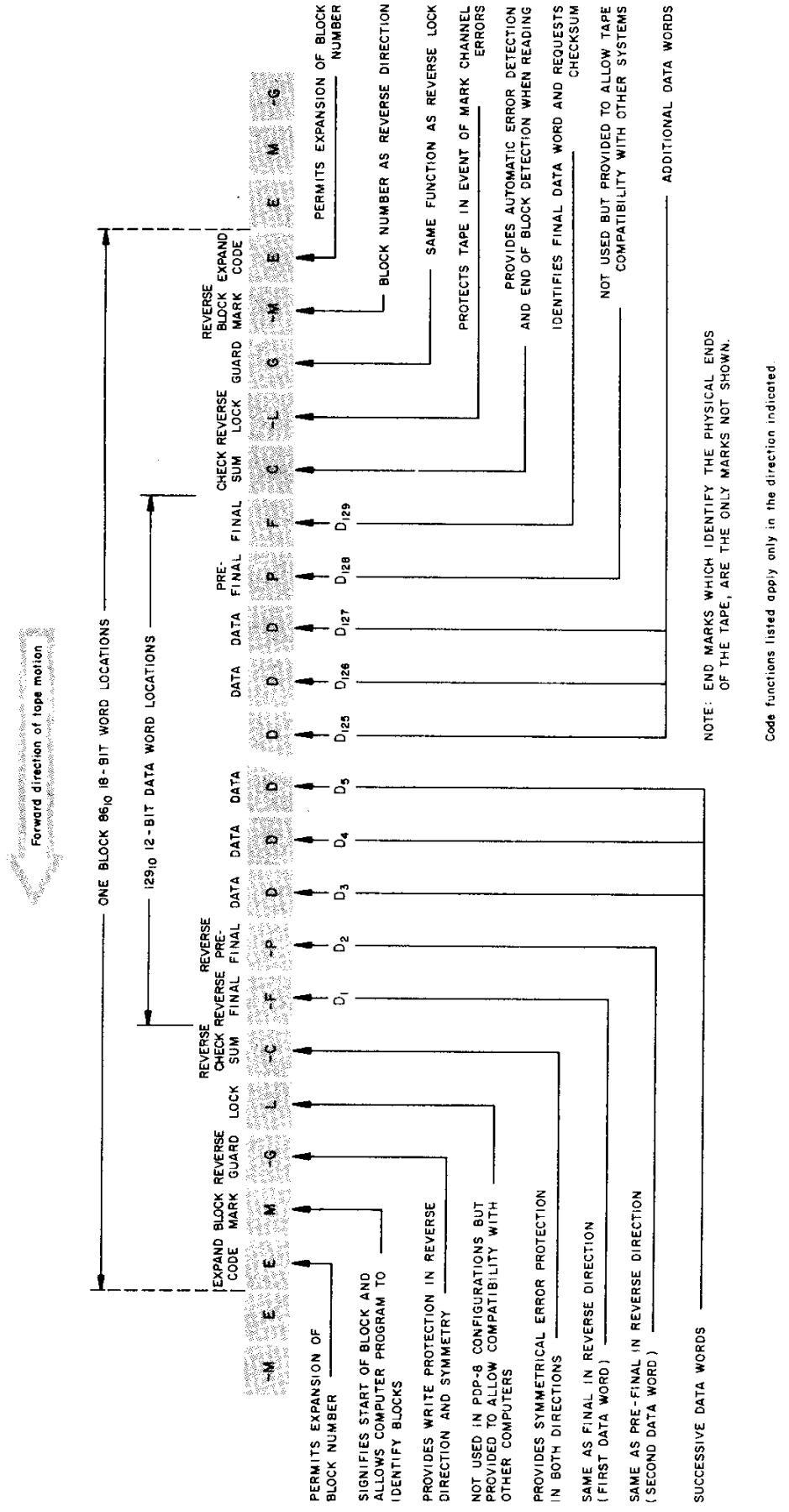


Figure 7-19 DECtape Control Word and Data Word Assignments

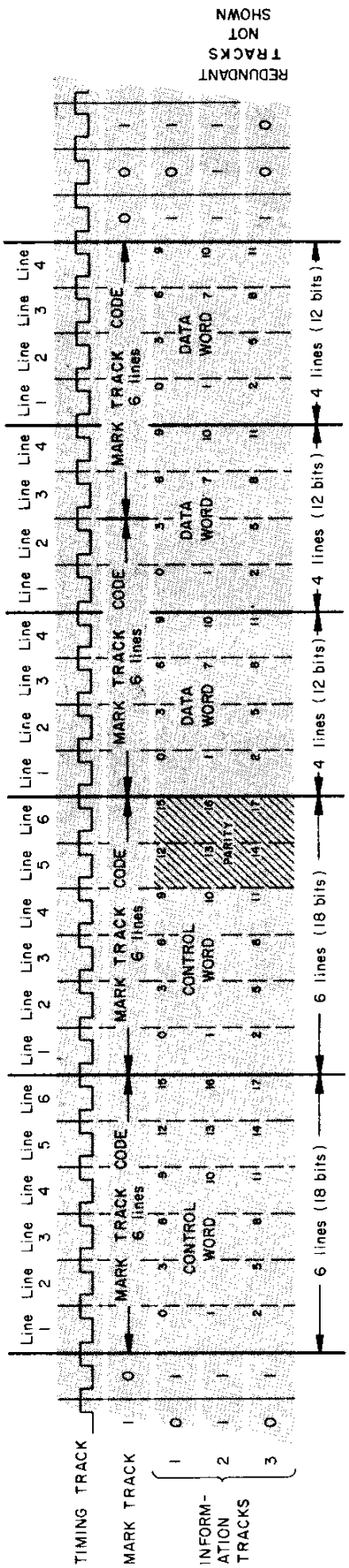


Figure 7-20 DECtape Format Details

TU56 Dual DECtape Transport and TC08-P DECtape Control
A DECtape system on the external bus can contain up to four TU56 Dual DECtape transports (the equivalent of up to eight single tape transports) controlled from one TC08-P unit. Data transfers between the computer and tape are implemented using the three-cycle data break facilities of the computer (refer to Chapter 10 for three-cycle data break description). Thus, the KA8-E Positive I/O Bus Interface and KD8-E Data Break Interface units are prerequisites.

Data is stored on tape in the form of three-bit words (refer to tape format) and is transferred between the tape and computer in the form of 12-bit words. A 12-bit read/write buffer in the TC08 assembles and disassembles the information for transfer. For transfers to the computer, data is read from four consecutive lines of tape and assembled into a 12-bit word. When transferred to the computer, the 12-bit word is supplied via external bus lines DATA00-11 to the KD8-E Data Break Interface. This unit, in turn, provides the word to OMNIBUS lines DATA0-11 under data break control. For transfers to tape, the KA8-E unit buffers the 12-bit words and provides them to the TC08 via external bus lines BMB00-11. The TC08 disassembles these words and supplies them to the tape transport for the writing of four tape lines. Transfer of command and control signals is effected by IOT instructions. These instructions are provided to the TC08 via the BMB00-11 external bus lines.

The TC08 contains registers and control flip-flops that form two status registers (designated A and B) for transfer of information to and from the computer accumulator.

TU56 Dual DECtape Transport
The TU56 provides the PDP-8/E user with a compact, high-reliability dual-reel tape transport in just 10½ inches of rack space. When used with the TC08-P control, the TU56 provides two fixed -address, magnetic tape facilities for high-speed loading, readout, and updating of programs and data. The TU56 transport contains the tape read/write heads, drive mechanisms, and switching circuits for tape drive and direction. All transport operations (except local) are controlled by the TC08 from program instructions. The TC08 selects the transport, controls tape motion and direction, selects a read or write operation and buffers data transferred. Information is stored in the form of three-bit words on a one-mil Mylar tape with ten tracks. This tape, ¾ inches in width and 260 feet in length, is contained on a reel that is less than four inches in diameter. Information can be recorded or read for either direction of tape motion.

Redundant recording (each bit of data and timing is recorded on two tracks) ensures high reliability and eliminates the need for parity checking. Data words are recorded on six of the ten tracks and four tracks are allotted for mark and timing channels. Other features include TTL logic, dynamic braking for shorter turnaround time, and DC motor drive to eliminate line frequency dependency. Connections from the read/write head are made directly to the external control, which contains the read and write amplifiers.

The logic circuits of the TU56 transport control tape movement in either direction over the read/write head. Tape drive motor control is com-

pletely through the use of solid-state switching circuits to provide fast, reliable operation. These circuits control the torque of the two motors that transport the tape across the head according to the established function of the device: i.e., go, stop, forward, or reverse. In normal tape movement, full torque is applied to the forward or leading motor and a reduced torque is applied to the reverse or trailing motor to keep proper tension on the tape. Since tape motion is bidirectional, each motor serves as either the leading or trailing drive for the tape, depending upon the forward or reverse control status of the TU56.

Tape movement can be controlled by commands originating in the computer or by manual operation of switches on the front panel of the transport. Manual control is used to mount new reels of tape on the transport, or as a quick maintenance check for proper operation of the control logic in moving the tape.

Since DECtape is a fixed address system, the programmer need not know accurately where the tape has stopped. To locate a specific point on tape he must only start the tape motion in the search mode. The address of the block currently passing over the head will be automatically transferred to core where it can be compared with the desired block address and tape motion continued or reversed accordingly. TU56 typical time characteristics are provided below, but are not accurately controlled.

Start Time	150 ms*
Stop Time	100 ms*
Turnaround Time	200 ms*

*Also, see control specifications. These times are frequently lengthened by the particular control.

Specifications

Transfer rate	33,300 three-bit characters per second
Information capacity	2.7 million bits per reel
Density	350 + or — 55 bits per inch
Tape speed	93 + or — 12 inches per second
Tape motion	Bidirectional
Start time	150 + or — 15 ms
Stop time	100 + or — 10 ms
Turn around time	200 + or — 50 ms
Reel capacity	250 ft. of ¾ inch, 1 mil Mylar tape
Reel size	3.9 inches in diameter
Mounting	Mounts in a standard 19-inch equipment rack
Size	10½ in. high, 19 in. wide, 9¾ in. deep
Cooling	Internally mounted fans provided
Power requirements	a. + 10V @ 0.53 amps or + 5V @ 0.55 amps b. — 15V @ 0.45 amps c. 115/220 VAC + or — 10% @ 2.85/1.43 amps 47-63 Hz
Environmental	Temperature: 40 degrees F to 90 degrees F Humidity: 15% to 80% Relative Humidity Internal Temp Rise: 10% F above ambient
Reliability	Recoverable Error Rate-less than 1 part in 2.5 x 10 ↑ 10 transfers

TC08 DECtape Control

The TC08 control buffers and controls information transfers between one to eight TU56 transports (one to four TU56 Dual DECtape transports) interfacing with the external bus of the PDP-8/E. Transfers are implemented using the three-cycle data break facilities of the computer; thus, the KA8-E Positive I/O Bus Interface and the KD8-E Data Break Interface modules are prerequisites.

During both input and output operations, the TC08 receives data and control information from the processor and generates the appropriate signals to the selected transport to execute the programmed commands. Binary information is transferred between the tape transport and the computer as one 12-bit computer word every 133⅓ μs. When writing, the TC08-P disassembles the 12-bit word into four successive three-bit words to be written on tape. During read operations, the TC08 assembles the four successive three-bit words into one 12-bit word for transfer to the computer. Transfers between the computer and the control always occur in parallel for a 12-bit word. Data transfers use the three-cycle data-break (high speed channel) facility of the computer. (Refer to Chapter 10 for details of 3-cycle data-break transfers.)

The TC08 contains the following primary control and data processing circuits:

a. Device selector and IOT decoding logic to command a transport from program instructions.
b. A 12-bit buffer register for assembling tape inputs and disassembling computer data.
c. A command and status register (designated Status Register A) for defining: (1) the active transport, (2) direction of tape, (3) tape motion, (4) operating mode, (5) function (read/write, search, etc.), (6) interrupt enable, and (7) clearing of flags.
d. A status register (designated Status Register B) for indicating error status and other status.
e. Flag circuits that provide the program with conditional indications and requests.
f. Tape motion and direction control circuits.
g. Mark track generation and detection circuits with error detectors.
h. Longitudinal parity generation and checking circuits.
i. Data break request circuits.

Programmed IOT instructions are generated to clear, read, or load Status Register A and to read or load Status Register B. An IOT skip instruction is also provided to test the status of flag circuits. These instructions are provided to the TC08-P control via the KA8-E Positive I/O Bus Interface and external bus lines BMB-00-11..

A control and indicator panel is also provided with the TC08. A single control, NORMAL/WRTM, places the TC08 in the write timing and mark track mode (WRTM), or else in the NORMAL mode. The indicators denote the current status of the control including the tape transport selected, motion, function, interrupt status, error flags, and other status indications.

Three program flags in the TC08-P control serve as condition indicators and request originators.

a. DECtape Flag (DT): This flag indicates the active/done status of the current function.
b. Data Flag (DF): This flag requests a data break to transfer a block number into the computer during a search function, or when a data word transfer is required during a read or write function.
c. Error Flag (EF): Detection of any nonoperative condition by the control sets this flag in status register B and stops (except for parity errors) the selected transport. The error conditions indicated by this flag are:
   (1) Mark Track Error: This error occurs any time the information read from the mark channel is erroneously decoded.
   (2) End of Tape: The end zone on either end of the tape is over the read head.
   (3) Select Error: This error occurs 5 μs after loading status register A to indicate any one of the following conditions:
   (a) Specifying a unit select code which does not correspond to any transport select number, or which is set to multiple transports.
   (b) Specifying a write function with the WRITE ENABLED/WRITE LOCK switch in the WRITE LOCK position on the selected transport.
   (c) Specifying an unused function code (i.e., AC6-8 = 111).
   (d) Specifying any function except write timing and mark track with the NORMAL/WRTM switch in the WRTM position.
   (e) Specifying the write timing and mark track function with the NORMAL/WRTM switch in the NORMAL position.
   (4) Parity Error: This error occurs during a read data function if the longitudinal parity or check sum over the entire data word, the reverse check character, and the check character is not equal to 1.
   (5) Timing Error: This error indicates a program fault caused by one of the following conditions:
   (a) A data break did not occur within 17 μs (+ or − 30%) of the data break request.
   (b) The DT flag was not cleared by the program before the control attempt to set it.
   (c) The read data or write data function was specified while a data block was passing the read head.

Three-cycle data break locations: The TC08-P uses location 7754 of field 0 for word count and 7755 of field 0 for current address.

Control Modes—The DECtape system operates in either the normal or continuous mode, as determined by bit 5 of status register A during a *DTXA command. Operation in each mode is as follows:*

a. Normal (NM): Data transfers and flag settings are controlled by the format of information on the tape.

b. Continuous (CM): Data transfers and flag settings are controlled by a word count read from core memory during the first cycle of each three-cycle data break, and by tape format.

Functions—The DECtape system performs one of seven functions, as determined by the octal digit loaded into status register A during a DTXA command. These functions are:

a. Move: Initiates movement of the selected transport tape in either direction. Mark channel decoding is inhibited in this mode except for end of tape.
b. Search: As the tape is moved in either direction, sensing of a block mark causes a data transfer of the block number. If the word count overflows in either NM or CM, the DT flag is set and causes a program interrupt. After finding the first block number, the CM can be used to avoid all intermediate interrupts between the current and the desired block number. This makes a virtually automatic search possible.
c. Read Data: This function is used to transfer blocks of data into core memory with the transfer controlled by the tape format. In NM, the DT flag is set at the end of a block and causes a program interrupt. In CM, transfers stop when the word count overflows, the remainder of the block is read for parity checking, and then the DT flag is set.
d. Read All: Read all is used to read tape in an unusual format, since it causes all lines to be read. In NM, the DT flag is set at each data transfer. In CM, the DT flag is set when WCO occurs. In either case, the DT flag causes a program interrupt.
e. Write Data: This function is used to write blocks of data with the transfer controlled by the standard tape format. After word count overflow occurs, zeros are written in all lines of the tape to the end of the current block. Then the parity checksum for the block is written. The DT flag rises as in the read function.
f. Write All: The write all function is used to write an unusual tape format (e.g., block numbers). The DT flag assertions are similar to the read all function.
g. Write Timing and Mark Track: This function is used to write on the timing and mark tracks, permitting blocks to be established or block lengths to be changed. The DT flag assertions are also similar to the read all function. This function is illegal unless a manual switch in the control is positioned to WRTM.

Programmed Operation—Prerecording of a reel of DECtape, prior to its use for data storage, is accomplished in two passes. During the first pass, the timing and mark channels are placed on the tape. During the second pass, forward and reverse block mark numbers, the standard data pattern, and the automatic parity checks are written. These functions are performed by the DECTOG program. Prerecording utilizes the write timing and mark channel function, and a manual switch on the control, which permits writing on the timing and mark channels, activates a clock, which produces the timing channel recording pattern and enables flags for program control. Unless this control function and switch are used simultaneously, it is physically impossible to write on the

mark or timing channels. An indicator lamp on the control panel lights when the manual NORMAL/WRTM switch is in the WRTM position. Under these conditions only, the write register and write amplifier, used to write on information channel 1 (bits 0, 3, 6, and 9), are used to write on the mark channel. This prerecording operation need only be performed once for each reel of DECtape.

There are two registers in the TC08 DECtape Control that govern tape operation and provide status information to the operating program. Status register A contains three unit selection bits, two motion bits, the continuous mode/normal mode bit, three function bits, and three bits that control the flags. Status register B contains the three memory field bits and the error status bits. PDP-8/E IOT microinstructions are used to clear, read, and load these registers. In addition, there is an IOT skip instruction to test control status.

Since all data transfers between DECtape and the computer memory are controlled by the data break facility, the program must set the WC and CA registers (locations 7754 and 7755, respectively) before a data break. After initiating a DECtape operation, the program should always check for error conditions (a program interrupt would be initiated if the error flag is enabled and if the program interrupt system is enabled). The DECtape system should be started in the search function to locate the block number selected for transfer; when the correct block is found, the transfer is accomplished by programmed setting of the WC, CA, and status register A.

When searching, the DECtape control reads block numbers only. These are used by the operating program to locate the correct block number. In NM, the DECtape flag is raised at each block number. In CM, the DECtape flag is raised only after the word count reaches zero. The current address is not incremented during searching and the block number is placed in core memory at the location specified by the content of the CA. Data is transferred to or from the computer core memory from locations specified by the CA register which is incremented by one before each transfer.

Each time the DECtape system is ready to transfer a 12-bit word, and when the start of the data position of the block is detected, the data flag is raised to initiate a data break request to the data break facility. Therefore, the main computer program continues running, but is interrupted approximately every 133⅓ μs for a data break to transfer a word. Transfers occur between DECtape and successive core memory locations specified by the CA. The initial transfer address minus one is stored in the CA by an initializing routine. The number of words transferred is determined by the tape format in NM, or by tape format and the word count in CM. At the conclusion of the data transfer, the DT flag is raised and a program interrupt occurs. The interrupt subroutine checks the DECtape error bits to determine the validity of the transfer, and either initiates a search for the next information to be transferred or returns to the main program.

During all normal writing transfers, a check character (the six-bit logical equivalent of the words in the data block) is computed automatically by

the control and is recorded automatically as one of the control words immediately following the data portion of the block. This same character is used during reading to determine that the data playback and recognition take place without error.

Any one of the eight tape transports may be selected for use by the program. After using a particular transport, the program can stop the transport currently being used and select another transport, or can select another transport while permitting the original selection to continue running. This is a particularly useful feature when rapid searching is desired, since several transports may be used simultaneously. Caution must be exercised, however; although the original transport continues to run, no tape-end detection or other sensing takes place. Automatic tape-end sensing that stops tape motion occurs in all functions, but only in the selected tape transport.

The following is a list of timing considerations for programmed operations. (These times are based on 129 12-bit data words per block.)

n(s) = the number of block numbers to be read in the search function and CM, counting through the one causing the word count overflow. Only the block number causing the word count overflow requests a program interrupt.

n(d) = number of words transferred divided by the number of words per block. If the remainder does not equal 0, use the next larger whole number.

n(A) = number of words transferred.

OPERATION	TIMING
Answer a data break request	Up to 17 μs + or − 30%
Word transfer rate	One 12-bit word every 133 μs + or − 30%
Block transfer rate	One 129-word block every 18.2 ms + or − 30%
Change funtcion from search to read data for the current block after DT flag from block number	400 μs + or − 30%
Change function from search to write data for current block after DT flag from block number	400 μs + or − 30%
Change function from read data to search for the next block after DT flag from transfer completion	1000 μs + or − 30%
Change function from write data to search for next block after DT flag from transfer completion	1000 μs + or − 30%
DECtape flag rises in continuous mode	
Move function	Never
Search function	(n(s)) x (18.2 ms + or − 30%)
Read data function	(n(D)) x (18.2 ms + or − 30%)
Read all function	(n(A)) x (133 μs + or − 30%)

OPERATION	TIMING
Write data function	(n(D)) x (18.2 ms + or − 30%)
Write all function	(n(A)) x (133 μs + or − 30%)
Write T & M function	(n(A)) x (133 μs + or − 30%)
In normal mode	
Move function	Never
Search function	Every 18.2 ms + or − 30%
Read data function	Every 18.2 ms + or − 30%
Read all function	Every 133 μs + or − 30%
Write data function	Every 18.2 ms + or − 30%
Write all function	Every 133 μs + or − 30%
Write T & M function	Every 133 μs + or − 30%

Programming

The following instructions are associated with TC08-P operation:

Read Status Register A (DTRA)

Octal Code: 6761
Execution Time: 2.6 μs
Operation: Transfers content of Status Register A to the AC. ORs AC0-9 with Status Register with the result appearing in AC. The AC is not cleared before the transfer. AC bit assignments are defined in Figure 7-21.

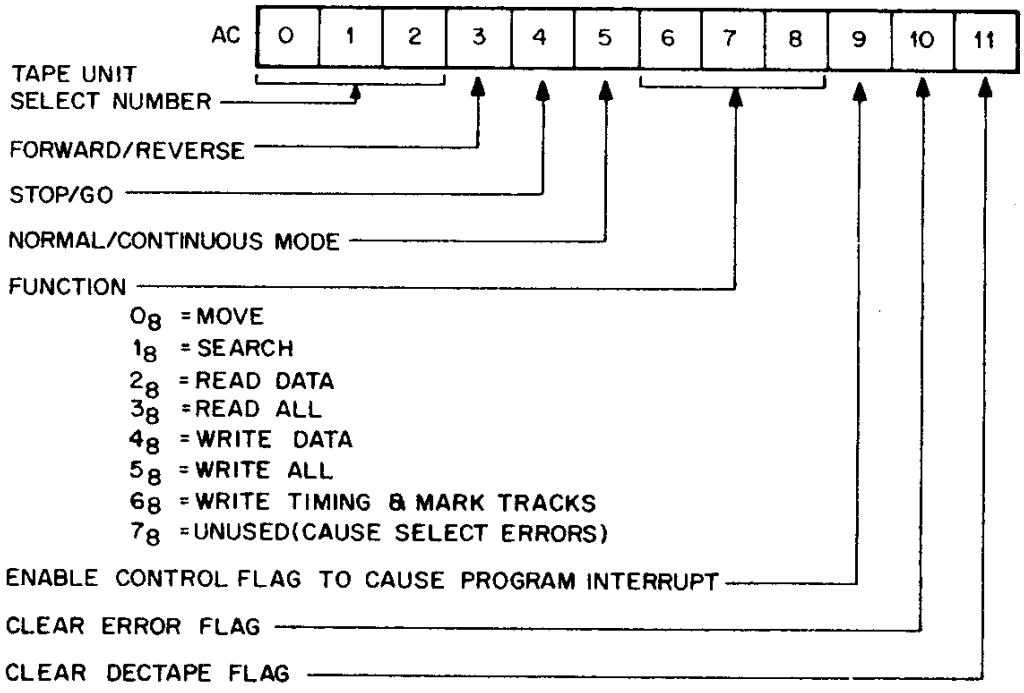


Figure 7-21 Status Register A Bit Assignments

Clear Status Register A (DTCA)

Octal Code: 6762
Execution Time: 2.6 μs
Operation: Clears Status Register A; DECtape and Error flag are undisturbed.

Clear and Load Status Register A (DTLA)

Octal Code: 6766
Execution Time: 3.6 μs
Operation: Clears Status Register A, then EXCLUSIVE ORs content of AC0-9 into Status Register A. Samples AC10 and 11 to control clearing of DECtape and error flags, then clears AC.

Load Status Register A (DTXA)

Octal Code: 6764
Execution Time: 2.6 μs
Operation: EXCLUSIVE ORs content of AC0-9 into Status Register A. Samples AC bits 10 and 11 to control clearing of Error and DECtape flags, then clears the AC.

Skip On Flag (DTSF)

Octal Code: 6771
Execution Time: 2.6 μs
Operation: If either DECtape or Error flags is set, skips the next instruction.

Read Status Register B

Octal Code: 6772
Execution Time: 2.6 μs
Operation: ORs content of Status Register B into AC. The AC is not cleared before transfer; AC bit assignments are defined in Figure 7-22.

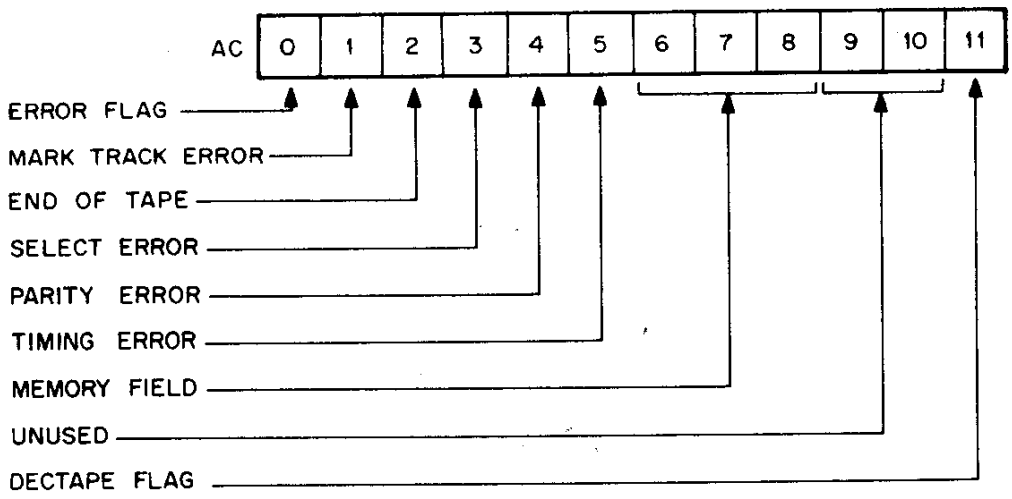


Figure 7-22 Status Register B Bit Assignments

Load Status Register B

Octal Code: 6774
Execution Time: 2.6 μs
Operation: Loads memory field portion of Status Register B with content of AC6-8, then clears the AC.

An elementary subroutine for reading or writing DECtape is given below. This routine does not use the interrupt, and exits with the DECtape drive halted.

The format for calling the subroutine is:

```
JMS (IDTAPE)    Effective JMS to IDTAPE, i.e., indirect JMS if IDTAPE
                  is not on same page as calling sequence.
WORD 1,         Bits 0-2, unit number
                Bit 3, start search (0=forward 1=reverse)
                Bits 6-8, memory field for transfer
                Bit 10, error return (0=JMP WORD 5)
                                     (1=JMP I WORD 5)
                Bit 11, function (0=READ 1=WRITE)
WORD 2,         Block number for start of transfer
WORD 3,         2's complement of the number of words to transfer
WORD 4,         Memory address of first transfer minus 1
WORD 5,         Error return or address for error return (to correspond
                  to Bit 10 of Word 1)
RETURN,         Transfer completed, return with AC cleared
```

```
ID7400,  7400                   /AND MASK (MUST BE FIRST CELL IN
                                /PAGE)
IDTAPE,  0                      /ENTRY TO SUBROUTINE
         CLA
         TAD    I IDTAPE       /SAVE WORD 1
         DCA    IDCODE
         ISZ    IDTAPE         /ADVANCE TO BLOCK NUMBER (WORD
                                /2)
         TAD    IDCODE
ID0200,  AND    ID7400         /UNIT NUMBER AND DIRECTION BIT
         TAD    ID0010         /PUT INTO SEARCH MODE
         DTCA   DTXA
         DTLB                  /CLEAR FIELD BITS
         TAD    IDWC
         DCA    I IDCA         /SET UP CURRENT ADDRESS (7755)

/ERROR WHILE SEARCHING . . . NORMALLY ENTERED WITH B
/STATUS REGISTER IN THE AC, PERFORMS TURN AROUND IF END
/ZONE ERROR, AND FORCES THE STOP-GO BIT TO GO

IDSERR,  RTL
         RAL                   /MOVE END ZONE FLAG TO LINK
         CLA    CML
         TAD    ID0200         /GET DECTAPE GO FLAG

/CHANGE DIRECTION IF AND ONLY IF THE LINK IS ZERO,

IDCONT,  SNL                   /CHECK DIRECTION AND SIGN
         TAD    ID0400         /REVERSE DIRECTION
         DTXA                  /ENTER AND GO IN SEARCH MODE
         DTSF   DTRB           /IDLE . . AND LOAD ERROR FLAG
         JMP    .-1            /WAIT UNTIL FLAG COMES UP
         SPA                   /TEST ERROR FLAG
         JMP    IDSERR
```

```
        DTRA            /GET DIRECTION BIT
        RTL
        RTL             /DIRECTION BIT GOES TO LINK
        SZL     CLA
        TAD     ID0002  /REVERSE . . . GET "BLOCK TO FIND"
                        /-2
        TAD     I IDWC  /ADD IN LAST BLOCK SEEN
        CMA             /COMPLEMENT
        TAD     I IDTAPE /ADD IN "BLOCK TO FIND"
        CMA
        SZA     CLA     /BLOCK NUMBERS MATCH?
        JMP     IDCONT  /REENTER SEARCH LOOP
        SZL             /CHECK DIRECTION BIT
        JMP     IDCONT+1 /TURN AROUND IF REVERSE

/END OF SEARCH LOOP, TAPE IS NOW AT DESIRED BLOCK
/TRAVELING IN A FORWARD DIRECTION,

        ISZ     IDTAPE
        TAD     I IDTAPE /GET WORD COUNT
        DCA     I IDWC
        ISZ     IDTAPE
        TAD     I IDTAPE /GET TRANSFER ADDRESS
        DCA     I IDCA
        TAD     IDCODE
        DTLB            /LOAD FIELD BITS
        IAC             /GET READ-WRITE FLAG
        AND     IDCODE
        RTL     CLL     /MULTIPLY BY 20 (OCTAL)
        RTL
        TAD     ID0130  /BUILD INSTRUCTION
        DTXA            /START UP READ OR WRITE
        DTSF    DTRB    /WAIT . . AND LOAD ERROR FLAG
        JMP     .-1
        ISZ     IDTAPE  /ADVANCE TO WORD 5
        SMA             /SKIP IF ERROR FLAG SET
        ISZ     IDTAPE  /ADVANCE TO WORD 6 . . . NORMAL
                        /EXIT
        SPA     CLA     /SKIP FOR NORMAL EXIT
        TAD     IDCODE  /GET INDIRECT RETURN BIT
        RTR             /MOVE TO LINK
        SNL     CLA     /SKIP IF JMP I <WORD 5>
        JMP     .+3
        TAD     I IDTAPE /MAKE DOUBLE INDIRECT RETURN
        DCA     IDTAPE
        DTRA
        AND     ID0200  /GET STOP-GO BIT
        TAD     ID0002  /PRESERVE DECTAPE ERROR FLAGS
        DTXA            /STOP TAPE
        JMP     I IDTAPE /EXIT
IDWC,   7754            /WORD COUNT FOR DATA BREAK
IDCA,   7755            /CURRENT ADDRESS FOR DATA BREAK
ID0010, 10              /SEARCH FUNCTION BIT
```

```
ID0400,  400          /FORWARD-REVERSE BIT
ID0130,  130          /USED TO BUILD READ AND WRITE
                      /CODE

ID0002,  2
IDCODE,  0
```

Software

Four types of programs have been developed as DECtape software for the PDP-8/E:

a. Subroutines which the programmer may easily incorporate into a program for data storage, logging, data acquisition, data buffering (queuing), etc.
b. A library calling system for storing named programs on DECtape and a means of calling them with a minimal size loader.
c. System software which provides for storing, assembling, and editing of programs on DECtape, thereby greatly increasing the versatility and flexibility of the PDP-8/E.
d. Programs for preformatting tapes controlled by the content of the switch register to write the timing and mark channels, to write block formats, to exercise the tape and check for errors, and to provide each of maintenance.

Program development has resulted in a series of subroutines which read or write any number of DECtape blocks, read any number of 129-word blocks as 128 words (one memory page), or search for any block (used by read and write, or to position the tape). These programs are assembled with the user's program and are called by a JMS instruction. The program interrupt is used to detect the setting of the DECtape flag, thus allowing the main program to proceed while the DECtape operation is being completed. A program flag is set when the operation has been completed. Thus, the program effectively allows concurrent operation of several input/output devices along with operation of the DECtape system. These programs occupy two memory pages (400 (octal) = 256 (decimal) words).

The library system has the following features: First, the computer state remains unchanged when it exits. Second, the library calls programs by name from the keyboard and allows for expansion of the program file stored on the tape. Finally, the library conforms to existing system conventions, namely, that all of memory except for the last memory page (7600 (octal)—7777 (octal)) is available to the programmer. The PDP-8/E DECtape library system is loaded by a 17 (decimal)—instruction bootstrap routine that starts at address 7600 (octal). This loader calls a larger program into the last memory page, whose function is to preserve on the tape the content of memory from 6000 (octal) through 7577 (octal), and then 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 basic system tape contains the following programs:

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

b. 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.

c. GETSYS: Generates a skeleton library tape on a specified DECtape unit.

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

Starting with the basic library tape, the user can build a complete file of his active programs and continuously update it. One of the uses of the library tape may be illustrated as follows:

The programmer may call the PDP-8/E FORTRAN compiler from the library tape and with it compile the program, obtaining the object program. The FORTRAN operating system may then be called from the library tape and used to load the object program. At this time the library program UPDATE is called, the operator defines a new program file (consisting of the FORTRAN operating system and the object program), adn adds it to the library tape. As a result, the entire operating program and the object program are now available on the DECtape library tape.

The DECtape system software is permanently stored on DECtape, from which it can be rapidly loaded. Any systems programs such as the assemblers (XPAL and XMACRO), the Symbolic Editor (XEDIT), or the Binary Loader (XLOAD), can be loaded in less than one minute.

The system software uses a standard DECtape format. There are 128 (200 (octal)) words per block and 1464 (2701(octal)) blocks, so the user has the remaining 1336 blocks for rapid access storage of his own programs.

The primary advantage for users are:

a. Efficient use of high-speed transfer rates between DECtape and core memory.

b. Symbolic programs may now be stored, edited, and assembled on DECtape, greatly increasing the versatility and flexibility of the PDP-8/E.

c. The computational workload can be more than doubled, compared to high-speed paper tape systems.

User's programs are written exactly as before for assembly by the PAL or MACRO-8 Assemblers. Using the Symbolic Editor, source programs are typed directly onto DECtape. After assembly, fast symbolic debugging can be done with DDT-8—after loading the program symbol table into DDT with the symbolic loader, XSYM.

The Binary Loader (XLOAD) can load the assembled binary program directly from the DECtape for program execution. Source files, symbol table, and program listings can be stored on DECtape and listed later, if desired. A duplicating program, XDUP, is available for copying programs.

This DECtape system also includes system calls to load any program from DECtape, to update or delete source files, and to restore the system for use by another programmer.

Although the system operates with one DECtape, a two DECtape configuration is strongly recommended as it will permit duplication of programs and saving of back-up master tapes. In a single DECtape system, if the system library is accidentally destroyed, the stored data cannot be replaced immediately because there is no means of recovery.

The last group of programs, called DECTOG, is a collection of short routines controlled by the content of the switch register. It provides for the recording of timing and mark channels and permits block formats to be recorded for any block length. Patterns may be written in these blocks and then read and checked. Writing and reading is done in both directions and checked. Specified areas of tape may be "rocked" for specified periods of time. A given reel of tape may thus be thoroughly checked before it is used for data storage. These programs may also be used for maintenance and checkout purposes.

EXTERNAL BUS DECmagnetic Tape Options

The DECmagtape unit can interface directly with the OMNIBUS via the TM8-E or to the External Bus via the TC58. The configurations of both categories are defined in the following table. For information on the TM8-E, refer to section 3.

DECmagtape Configurations

SYSTEM OPTION	EQUIPMENT	NO. OF CHANNELS	DENSITIES (BPI)	TAPE SPEED (IPS)	OTHER INFORMATION
TM8-EA*	TM8-E Control & TU10-EA(master)	9	800	45	Control plugs into OMNIBUS. TU10-EA contains a master and one slave. Up to 7 additional TU10 slaves may be added. 7 and 9 track TU10's can be mixed on the same system. For example, a 7 track master can be operated with a 9 track slave etc. The master consists of logic modules which plug into the TU-10 electronics.
TM8-FA*	TM8-E Control & TU10-FA(master)	7	800/556/200	45	Same as above.
TC58	TC58 Control(master) & TU10-EE(slave)	9	800	45	DW08A I/O conversion panel, KA8-E Positive I/O Bus and KD8-E Data Break Interface are prerequisites. The master is contained with the TC58 controller. Up to 7 additional TU10 slaves may be added. 7 and 9 track TU10's can be mixed on the same system.
TC58	TC58 Control(master) & TU10-FE(slave)	7	800/556/200	45	Same as above.

* Refer to Section 3 for TM8 options.

TC58 DECmagtape System

The TC58 will control the operation of a maximum of eight digital magnetic tape transports, Types TU10, TU20 and TU30. The TC58 interfaces to and uses the KA8-E Positive I/O Bus and the KD8-E Data Break Interface.

The TC58 performs similar functions as the TM8-E. However, no master unit is required. The DECmagtape system configurations are illustrated in the following table. Refer to section 3 for a description of the TU10 DECmagtape unit.

Operation

Transfers are governed by the in-memory WC and CA register associated with the assigned data channel (memory locations 7752_8 and 7753_8). Since the CA is incremented before each data transfer, its initial contents should be set to the desired initial address minus one. The WC is also incremented before each transfer and must be set to the two's complement of the desired number of data words to be transferred. In this way, the word transfer which causes the word count to overflow (WC becomes zero) is the last transfer to take place. The number of IOT instructions required for the TC58 is minimized by transferring all necessary control data (i.e., unit number, function, mode, direction, etc.) from the AC to the control using OIT instructions. Similarly, all status information (i.e., status bits, error flags, etc.) can be read into the AC from the control unit by IOT instructions.

During normal data reading, the control assembles 12-bit computer words from successive frames read from the information channels of the tape. During normal data writing, the control disassembles 12-bit words and distributes the bits so they are recorded on successive frames of the information channels.

Programming

The commands for the magnetic tape control system are as follows:

Skip on Error Flag or Magnetic Tape Flag (MTSF)

Octal Code:	6701
Operation:	The status of the error flag (EF) and the magnetic tape flag (MTF) are sampled. If either or both are set to 1, the content of the PC is incremented by one to skip the next sequential instruction.
Symbol:	If MTF or EF = 1, PC + 1 → PC

Skip on Tape Control Ready (MTCR)

Octal Code:	6711
Operation:	If the tape control is ready to receive a command, the PC is incremented by one to skip the next sequential instruction.
Symbol:	If Tape Control Ready, PC + 1 → PC

Skip on Tape Transport Ready (MTTR)

Octal Code:	6721
Event Time:	1
Indicators:	Iot, Fetch, Pause [IR = 6, F]

Execution Time: 4.25 μs
Operation: The next sequential instruction is skipped if the selected tape transport is ready.
Symbol: If tape unit ready, PC + 1 → PC

Clear Registers, Error Flag and Magnetic Tape Flag (MTAF)

Octal Code: 6712
Event Time: 2
Indicators: Iot, Fetch, Pause [IR = 6, F]
Execution Time: 4.25 μs
Operation: Clears the status and command registers, and the EF and MTF if tape control is ready. If tape control is not ready, clears MTF and EF flags only.
Symbol: If tape control is ready, 0 → MTF, 0 → EF, 0 → command register
If tape control is not ready, 0 → MTF, 0 → EF

Inclusive OR Contents of Command Register (MTRC)

Octal Code: 6724
Event Time: 3
Indicators: Iot, Fetch, Pause [IR = 6, F]
Execution Time: 4.25 μs
Operation: Inclusively OR the contents of the command register into AC0-11.
Symbol: AC V command register → AC

Inclusive OR Contents of Accumulator (MTCM)

Octal Code: 6714
Event Time: 3
Indicators: Iot, Fetch, Pause [IR = 6, F]
Execution Time: 4.25 μs
Operation: Inclusively OR the contents of AC0-5, AC9-11 into the command register; jam transfer AC6-8 (command function).
Symbol: AC05, AC09-11 V command register → command register
AC6-8 → command register bits 6-8

Load Command Register (MTLC)

Octal Code: 6716
Event Time: 2, 3
Indicators: Iot, Fetch, Pause [IR = 6, F]
Execution Time: 4.25 μs
Operation: Load the contents of AC0-11 into the command register
Symbol: AC0-11 → command register

Inclusive OR Contents of Status Register (None)

Octal Code: 6704
Event Time: 3
Indicators: Iot, Fetch, Pause [IR = 6, F]
Execution Time: 4.25 μs

Operation:	Inclusively OR the contents of the status register into AC0-11.
Symbol:	Status Register V AC → AC0-11

Read Status Register (MTRS)

Octal Code:	6706
Event Time:	2, 3
Indicators:	Iot, Fetch, Pause [IR = 6, F]
Execution Time:	4.25 μs
Operation:	Read the contents of the status register into AC0-11.
Symbol:	Status Register → AC0-11

Mag Tape "GO" (MTGO)

Octal Code:	6722
Event Time:	2
Indicators:	Iot, Fetch, Pause [IR = 6, F]
Execution Time:	4.25 μs
Operation:	Set "GO" bit to execute command in the command register if command is legal.
Symbol:	None

Clear the AC (None)

Octal Code:	6702
Event Time:	2
Indicators:	Iot, Fetch, Pause [IR = 6, F]
Execution Time:	4.25 μs
Operation:	Clear the accumulator.
Symbol:	0 → AC

Although any number of tapes may be simultaneously rewinding, data transfer may take place to or from only one transport at any given time. In this context, data transfer includes these functions: read or write data, write EOF (end-of-file), read/compare, and space. When any of these functions are in process, the tape control is in the not-ready condition. A transport is said to be not ready when the tape is in motion, when transport power is off, or when it is off-line.

Data transmission may take place in either parity mode; add for binary, even for BCD. When reading a record in which the number of characters is not a multiple of the number of characters per word, the final characters come into memory left-justified.

Ten bits in the magnetic tape status register retain error and tape status information. Some error types are combinations, such as lateral and longitudinal parity errors (parity checks occur after both reading and writing of data), or have a combined meaning, such as illegal command, to allow for the maximum use of the available bits.

The magnetic tape status register reflects the state of the currently selected tape unit. Interrupts may occur only for the selected unit. Therefore, other units which may be rewinding, for example, will not interrupt when done.

Magnetic Tape Functions

For all functions listed below, upon completion of the data operation

(after the end-of-record character passes the read head), the MTF (magnetic tape flag) is set, an interrupt occurs (if enabled), and errors are checked.

No Operation—NO OP command defines no function in the command register. A MTGO instruction with NO OP will cause an illegal command error (set EF).

Space—There are two commands for spacing records, SPACE FORWARD and SPACE REVERSE. The number of records to be spaced (two's complement) is loaded into the WC. The CA need not be set. The MTF (magnetic tape flag) is set, and an interrupt occurs at WC overflow, EOF (end of file), or EOT (end of tape), whichever occurs first. When issuing a space command, both the density and parity bits must be set to the density and parity in which the records where originally written.

Load Point or Beginning of Tape (BOT) detection during a backspace terminates the function with the BOT bit set. If a SPACE REVERSE command is given when a transport is set at BOT, the command is ignored, the illegal command error and BOT bits are set, and an interrupt occurs.

Read Data—Records may be read into memory only in the forward mode. Both CA and WC must be initialized CA, to the initial core address minus one; WC, to the two's complement of the number of words to be read. Both identify and parity bits must be set.

If WC is set to less than the actual record length, only the desired number of words are transferred into memory. If WC is greater than or equal to the actual record length, the entire record is read into memory. In either case, both parity checks are performed, the MTF is set, and an interrupt occurs when the end-of-record (EOR) mark passes the read head. If either lateral or longitudinal parity errors or bad tape have been detected, or an incorrect record length error occurs (WC not equal to the number of words in the record), the appropriate status bits are set. An interrupt occurs only when the MTF is set.

To continue reading without stopping tape motion, MTAF (clear MTF) and MTGO instructions must be executed. If the MTGO command is not given before the shut down delay terminates, the transport will stop.

Write Data—Data may be written on magnetic tape in the forward direction only. For the WRITE DATA function, the CA and WC and density and parity bits must be initialized. WRITE DATA is controlled by the WC, such that when the WC overflows, data transfer stops, and the EOR character and IRG (interrecord gap) are written. The MTF is set after the EOR has passed the read head. To continue writing, a MTGO command must be issued before the shut down delay terminates. If any errors occur, the EF will be set when the MTF is set.

A special feature of this control is the write extended interrecord gap capability. This occurs on a write operation when command register bit 5 is set. The effect is to cause a 3-inch interrecord gap to be produced before the record is written. The bit is automatically cleared when the wriitng begins. This is very useful for creating a 3-inch gap of blank tape over areas where tape performance is marginal.

Write EOF—The WRITE EOF command transfers a single character (17_8) record to magnetic tape and follows it with an EOR character. Ca and WC are ignored for WRITE EOF. The density bits must be set, and the command register parity bit should be set to even (BCD) parity. If it is set to odd parity, the control will automatically change it to even.

When the EOF marker is written, the MTF is set and an interrupt occurs. The tape transport stops, and the EOF status bit is set, confirming the writing of EOF. If odd parity is required after a WRITE EOF, it must be specifically requested through the MTLC command.

Read/Compare—The READ/COMPARE function compares tape data with core memory data. It can be useful for searching and positioning a magnetic tape to a specific record, such as a label or leader, whose content is known in core memory, or to check a record just written. READ/COMPARE occurs in the forward direction only; CA and WC must be set. If there is a comparison failure, incrementing of the CA ceases, and the READ/COMPARE error bit is set in the status register. Tape motion continues to the end of the record; the MTF is then set and an interrupt occurs. If there has been a READ/COMPARE error, examination of the CA reveals the word that failed to compare.

Rewind—The high-speed REWIND command does not require setting of the CA or WC. Density and parity settings are also ignored. The REWIND command rewinds the tape to loadpoint (BOT) and stops. Another unit may be selected after the command is issued and the rewind is in process. MTF is set, and an interrupt occurs (if the unit is selected) when the unit is ready to accept a new command. The selected unit's status can be read to determine or verify that REWIND is in progress.

Continued Operation

To continue operating in the same mode, the MTGO instruction is given before tape motion stops. The order of commands required for continued operation is as follows:

a. MTCM, if the command is to be changed.
b. MTAF, will only clear MTF and EF flags since tape control will be in a not ready state.
c. MTGO, if MTCM requested an illegal condition, the EF will be set at this time.

To change modes of operation, either in the same or opposite direction, the MTCM command is given to change the mode and an MTGO command is given to request the continued operation of the drive. If a change in direction is ordered, the transport will stop, pause, and automatically start up again.

If the WRITE function is being performed, the only forward change in command that can be given is WRITE EOF.

If no MTGO instruction is given, the transport will shut down in the interrecord gap.

NOTE

No flags will be set when the control becomes ready or the transport becomes ready, except if the REWIND command is present in the command register and the selected drive reaches BOT and is ready for a new command.

If a WRITE (odd parity) command is changed to WRITE EOF, the parity is automatically changed to even.

NOTE

Even parity will remain in the command register unless changed by a new command instruction, MTLC, which clears and loads the entire command register.

Status or Error Conditions

Twelve bits in the magnetic tape status register indicate status or error conditions. They are set by the control and cleared by the program.

The magnetic tape status register (STR) bits are:

BIT*	FUNCTION (WHEN SET)
0	Error Flag (EF)
1	Tape rewinding
2	Beginning of tape (BOT)
3	Illegal command
4	Parity error (Lateral or Longitudinal)
5	End of file (EOF)
6	End of tape (EOT)
7	Read/compare error
8	Record length incorrect WC = 0 (long) WC = (short)
9	Data request late
10	Bad tape
11	Magnetic tape flag (MTF) or job done

MTF (STR11)—The MTF flag is set under the following conditions:

a. Whenever the tape control has completed an operation (after the EOR mark passes the read head).

b. When the selected transport becomes ready following a normal REWIND function.

These functions will also set the EF if any errors are present.

EOF (STR5)—EOF is sensed and may be encountered for those functions which come under the heading of READ STATUS FUNCTION; i.e., SPACE, READ DATA, or READ/COMPARE and WRITE EOF. When EOF is

* The register bits are equivalent in position to the AC bits (i.e., STR 0 = AC 0, etc.).

encountered, the tape control sets EOF = 1. MTF is also set; hence, an interrupt* occurs and the EOF status bit may be checked.

EOT (STR6) and BOT (STR2)—EOT detection occurs during any forward command when the EOT reflective strip is sensed. When EOT is sensed, the EOT bit is set, but the function continues to completion. At this time the MTF is set (and EF is set), and an interrupt occurs.

BOT detection status bit occurs only when the BOT reflective strip is read on the transport that is selected.

When BOT detection occurs and the unit is in reverse, the function terminates. If a tape unit is at load point when a REVERSE command is given, an illegal command error bit is set causing an EF with BOT set. An interrupt then occurs.

Illegal Command Error (STR3)—The illegal command error bit is set under the following conditions:

a. A command is issued to the tape control with the control not ready.

b. A MTGO command is issued to a tape unit which is not ready, and the tape control is ready.

c. Any command (see figure 7-23) which the tape control, although ready, cannot perform; for example:

   (1) WRITE with WRITE LOCK condition

   (2) 9-channel tape and incorrect density

   (3) BOT and SPACE REVERSE

Parity (STR4)—Longitudinal and lateral parity checks will occur in both reading and writing. The parity bit is set for either lateral or longitudinal parity failure. A function is not interrupted, however, until MTF is set. Maintenance panel indicators are available to determine which type of parity error occurred.

Read/Compare Error (STR7)—When READ/COMMAND function is underway, STR7 is set to 1 for a READ/COMPARE error (see earlier portion of this section on READ/COMPARE for further details.

Bad Tape (STR10)—A BAD TAPE error indicates detection of a bad spot on the tape. Bad tape is defined as three or more consecutive missing characters followed by data, within the period defined by the shutdown delay. The error bit is set by the tape control when this occurs. MTF and interrupt do not occur until the end of the record in which the error was detected.

Tape Rewinding (STR1)—When a REWIND command has been issued to a tape unit and the function is underway, the tape rewinding bit is set in the control. This is a transport status bit, and any selected transport which is in a high-speed rewind will cause this bit to be set.

* All references to the interrupts assume the tape flags have been enables to the interrupt (command register bit 9 = 1) and that the unit is selected.

Record Length Incorrect (STR8)—During a READ or READ/COMPARE, this bit is set when the WC overflow differs from the number of words in the record. The EF flag is set.

Data Request Late (STR9)—This bit can be set whenever data transmission is in progress. When the data flag causes a data break cycle, the data must be transmitted before a write pulse or a read pulse occurs. If it does not, this error will occur, and data transmission will cease. The EF flag and bit 9 of the status register are set when the MTF is set.

Error Flag (STR0)—EF is set whenever any error status bit is present at the time that MTF is set. However, when an illegal command is given, the EF is set and the MTF is not set.

Command Register Contents

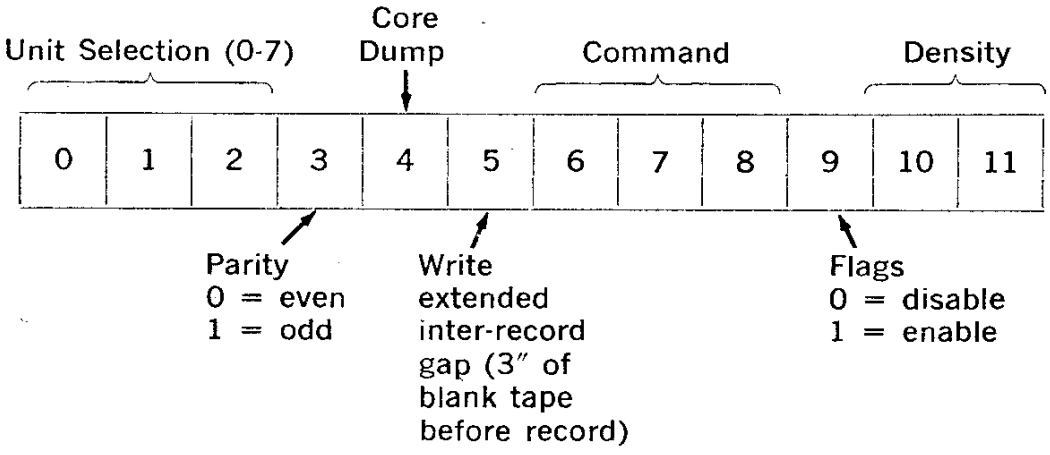


Figure 7-23

Unit Selection

Unit	Unit Selection Bits 0	1	2
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

Density	Density Selection Density Bits	
200 bpi	0	0
556 bpi	0	1
800 bpi	1	0
800 bpi		
9 channel	1	1

Command Selection

COMMAND	BITS 6	7	8
NO OP	0	0	0
Rewind	0	0	1
Read	0	1	0
Read/Compare	0	1	1
Write	1	0	0
Write EOF	1	0	1
Space Forward	1	1	1
Space Reverse	1	1	1

Magnetic Tape Function Summary

Legend: CA = Current Address Register = 7752$_8$
WC = Word Count Register = 7753$_8$
F = Forward
R = Reverse
DS = Density Setting
PR = Parity Setting
EN = Enable Interrupt

Function	Characteristics	Status of Error Types
NO-OP	CA: Ignored WC: Ignored DS: Ignored PR: Ignored EN: Ignored	Illegal BOT Tape Rewinding
SPACE FORWARD	CA: Ignored WC: two's complement of number of records to skip DS: Must be set PR: Must be set EN: Must be set	Illegal EOF Bad Tape MTF, BOT, EOT
SPACE REVERSE	Same as Space Forward	Illegal EOF Bad Tape BOT MTF

READ DATA	CA: Core Address —1 WC: two's complement of number of words to be transferred DS: Must be set PR: Must be set EN: Must be set	Illegal EOF Parity Bad Tape MTF EOT Data Request Late Record Length Incorrect
WRITE DATA	Same as READ DATA	Illegal EOT Parity MTF Bad Tape
WRITE EOF	CA: Ignored WC: Ignored DS: Must be set PR: Must be set EN: Must be set	Data Request Late Same as WRITE DATA plus EOF
READ/COMPARE	Same as READ DATA	Illegal EOF Read/Compare Error Bad Tape MTF EOT Data Late Record Length Incorrect
REWIND	CA: Ignored WC: Ignored DS: Ignored PR: Ignored EN: Must be set	Illegal Tape Rewinding MTF BOT

DATA ACQUISITION PERIPHERALS

Digital Equipment Corporation manufactures a number of data acquisition and control subsystems designed for use with the PDP-8/E. In subsystem can be purchased initially and expanded as required. Three major analog multiplexer subsystems are available, covering analog input ranges from 10 mv to 100 V and having channel capabilities from 4 to 1024 channels. In addition, two types of Digital-to-Analog Converter subsystems are provided as options. A Universal Digital Controller subsystem is also offered as an option. This subsystem provides capabilities for controlling or interrogating up to 3072 discrete digital loads or sources.

AD01-A 10 (or 11)-Bit Analog-to-Digital Converter

The AD01-A is a flexible, low cost, multichannel analog acquisition option for the PDP-8/E computer. The standard AD01-A consists of an expandable solid-state multiplexer, channel selection circuits for up to 32 channels, a programmable input range selector, a 10-bit A/D converter, control, and interface, and power supply. Module additions can be used to expand the multiplexer in four-channel groups up to 32 channels. (Optional 32 channel expander available).

The AD01-A interfaces with the external bus of the PDP-8/E computer to provide ten-bit digitization of unipolar analog signals having a full-scale range of 0V to +1.25V, +2.5V, +5V or +10V. An optional sign-bit addition permits eleven-bit bipolar operation. A programmable input range selection extends the dynamic range of the AD01-A (at moderate sampling rates) to an equivalent of 13 bits for unipolar inputs or to 14 bits for bipolar inputs. An optional sample and hold amplifier is also available to reduce the conversion aperture to 100 ns. Each multiplexer channel switch utilizes an enhancement mode MOSFET that is normally open when unselected, or when power is off. These switches provide overload protection for up to + or — 20V, and signal protection against short circuits.

Operation of the AD01-A is controlled by program instructions from the computer. An ADSC instruction selects the multiplexer channel, system gain, and the interrupt or noninterrupt mode as defined by an AC word.

The selected channel input is connected to the programmable-gain amplifier, which scales the analog input for a 0 to +10V output that is provided to the A/D converter summation junction, either directly or via the sample-and-hold. The ADSC instruction also clears the A/D Done Flag and initiates a conversion cycle. The conversion is performed by successive approximation. For standard unipolar AD01-A, the analog input results in a 10-bit binary output code. The sign-bit option permits conversion of bipolar inputs (0 to + or — 1.25V, + or — 2.5V, + or — 5V, or + or — 10V) to an eleven-bit, two's complement code with an extended sign format. For this format, AC bits 0 and1 are connected to the same source and denote polarity of the analog input (1 = —V, 0 = +V).

When the conversion is complete, the A/D converter sets its A/D Done Flag. This flag is sensed by an ADSF instruction and, if set, causes the next program instruction to be skipped so that the converter word can be transferred to the AC, using an ADRB instruction. The ADRB instruction also clears the A/D Done flag to ready the AD01-A for another cycle.

The above operations can also be implemented using microprogrammed IOT instructions for "best sampling" operation.

Specifications

Resolution	Unipolar 10 bits, or 1 part in 1024 Bipolar (option) sign + 10 bits
Converter Accuracy	± 0.1% of 10V full-scale input ± 0.125% of full-scale with sample & hold
Quantizing Error	± ½ least significant bit
Thruput Rate	22μs including channel & gain selection unipolar; 29μs bipolar.
S & H Aperture	100 ns 0.1μs with sample and hold
Sample & Hold Acquisition	5μs to .01% of full-scale step change
No. of Analog Inputs	4 minimum, expandable to 32 in groups of 4 Channels (64 channels available)
Input Impedance	1000 megohms in parallel with 20 pf.
Input Isolation	Enhancement mode MOSFET switches, "off" when unselected or power off.
Channel Selection (program selectable)	5-bit address
Input Voltage Range (program selectable)	Unipolar: 0 to + 1.25, +2.5, + 5.0, +10.0V full-scale Bipolar (option): 0 to ± 1.25, ± 2.5V, ± 5.0, ± 10.0V CRC and VRC (compute): 1.4 μs full-scale
Overload Capability	± 20V on all ranges without damage
Cross Channel Attenuation	78 db, DC-80Hz for 20 volts p-p signals, 100 ohm source impedance
Input Gain	Program selectable

Accuracy Specifications

Program Selected Gain	×1	×2	×4	×8	Comments
Full Scale Input	+10V	+5.0V	+2.5V	+1.25V	Selectable
Gain Accuracy	0.05%	0.05%	0.05%	0.05%	*Unipolar* % of full-scale for 30 days
Resolution	9.8mv	4.9mv	2.45mv	1.22mv	per bit
Zero Drift	550	300	175	110	*uv/degreeC RTI*

Zero drift w/sample & hold	750 2mv	400 1mv	225 1mv	140 1mv	uv/degreeC RTI 0-0 RTI
Noise w/sample & hold	+−.05% 2.4mv	1.2mv +−.05%	1.2mv +−.05%	1.2mv +−.1%	0-0 25 degree C and 3 sigma confidence
Repeatability					

Sign Bit	Adds one bit
Word Length	Control word 12 bits Output word 12 bits
Modes of Operation	Interrupting/noninterrupting (program-selectable)
Environmental	0 to + 55 degrees C operating −25 degrees to + 85 degrees C storage to 90% without condensation
Power	115/230V + or − 10% 47 to 470 Hz, single phase less than 50W

Programming

The following instructions are associated with AD01A operation:

Skip on A/D Done Flag (ADSF)

Octal Code:	6531
Execution Time:	2.6 μs
Operation:	Skips the next program instruction if the A/D Done Flag is set.

Read A/D Buffer (ADRB)

Octal Code:	6532
Execution Time:	2.6 μs
Operation:	Transfers the content of the A/D buffer to AC0 through AC11 and clears the A/D Done Flag.

Convert Analog Input (ADCV)

Octal Code:	6534
Execution Time:	2.6 μs
Operation:	Clears the A/D Done Flag and initiates a conversion operation.

Select Multiplexer Channel and Gain (ADSC)

Octal Code:	6535
Execution Time:	3.6μs
Operation:	Loads the content of AC into multiplexer input register and implements channel and gain selection. Also clears the A/D Done Flag and initiates a conversion.

Read A/D Buffer, Clear Flag and Start Conversion (ADRC)

Octal Code: 6536
Execution Time: 3.6μs
Operation: Jam transfers content of A/D buffer to AC0-AC11, clears flag, and starts a new conversion.

Select Channel and Gain and Read A/D Buffer (ADSR)

Octal Code: 6537
Execution Time: 4.6μs
Operation: Transfers the content of the AC to the multiplexer input register, clears the AC, and transfers the content of the A/D buffer to the AC. The A/D Done Flag is then cleared, and another conversion is initiated.

NOTE

The ACRC (6536) and ADSR (6537) instructions are used for "burst sampling" conversions in the noninterrupt mode.

A program service routine for the AD01-A can be written as follows:

```
ADSC    /SELECT CHANNEL, GAIN AND MODE AND START
        /CONVERSION
ADSF    /SKIP ON A/D DONE FLAG
JMP.—1  /JUMP BACK AND TEST FLAG AGAIN
EXIT    /READ A/D BUFFER
```

AFC8 Low-Level Analog Input Subsystem

The AFC8 is a computer-based unit that multiplexes up to 1024 analog inputs, selects gain, and performs a 12-bit analog-to-digital conversion. Analog inputs can be provided by thermocouples or strain-gain sources having a source impedance of 0 to 500 Ω. Both channel selection and gain of the multiplexer are under control of the computer; thus, the multiplexer can handle combinations of low-level and high-level analog inputs having a range from 10 mV dc full scale to 100V full scale. Channel sampling is performed at a maximum rate of 200 channels per second. Field wiring terminates on screw terminal connectors, and requires only simple 2-wire twisted pair inputs.

Analog inputs are converted to 12-bit digital words (11 bits plus sign) for transfer to the PDP-8/E computer. Sampling of analog inputs is initiated by the computer issuing IOT instructions. Two IOT instructions are normally required for each channel sample. The first instruction defines the multiplexer gain for the channel and loads a three-bit gain

word from the AC into a multiplexer register. The second instruction defines one of 1024 possible channels for the sample and loads an 11-bit address from the AC into a register. This instruction also starts a multiplexer timing cycle, in which system gain is established, the channel is selected, and the analog output is made available to the A/D converter. When the A/D converter completes the conversion, it loads the 12-bit word into a buffer register, terminates multiplexer sampling, and sets its device flag. The computer senses the state of the device flag by issuing ADSF instructions. When the flag is set, the A/D converter returns a skip request and the computer issues an ADRB instruction to transfer the digital word to AC00 through AC11. This instruction also clears the A/D flag to ready the device for another conversion.

The subsystem is housed in 19-inch industrial type (H964) cabinets that have their own cooling and low-voltage power supplied. Four of these cabinets are required for the maximum channel capability of 1024 channels. The multiplexer is connected to the PDP-8/E computer using a Positive I/O Bus Interface module.

Specifications

Analog Input Voltage Ranges	± 10 mV full scale ± 50 mV full scale ±100 mV full scale ±200 mV full scale −200 mV to + 500 mV full scale −200 mV to +1 V full scale −200 mV to +5 V full scale −200 mV to + 10 V full scale −200 mV to +100 V full scale
Analog Input Current Ranges	± 1 mA full scale ± 5 mA full scale ±10 mA full scale ±50 mA full scale
A/D Converter Output Word	Parallel, 12 bits (11 bits plus sign)
Resolution	5μV
Accuracy	± 25μV or ±0.05% of f.s., whichever is greater
Common Mode Rejection	120 dB or greater above 60 Hz
Common Mode Voltage	200V
Normal Mode Rejection	50 dB or greater at 60 Hz
System Sampling Rate	200 channels per second (max.)
Single Channel Sampling Rate	20 samples per second max. at stated accuracy
Expansion Capabilities	System can be expanded in groups of 8 channels up to a maximum of 1024 channels.

Internal A/D Conversion Time	50 μs
Resolution	Sign + 11 bits

Programming
Instructions for multiplexer operation are listed below.

Set Multiplexer Gain (ADSG)

Octal Code:	6542
Execution Time:	2.6 μs
Operation:	Loads a three-bit gain word from AC09 through 11 into a multiplexer register for selection of system gain for channel.

Set Multiplexer Address (ADSA)

Octal Code:	6544
Execution Time:	2.6 μs
Operation:	Loads an eleven-bit address from AC01 through 11 into a multiplexer register for selection of a channel. Also starts multiplexer timing to select gain and channel and provide analog sample to A/D converter.

Skip on A/D Flag (ADSF)

Octal Code:	6531
Execution Time:	2.6 μs
Operation:	Senses A/D converter flag. If flag is a one, increments the PC, and skips the next sequential instruction so the A/D converter can be serviced.

Read A/D Converter Buffer (ADRB)

Octal Code:	6534
Execution Time:	2.6 μs
Operation:	Transfers twelve-bit word in A/D buffer to AC00 through AC11, and resets A/D converter flag.

A program for selecting multiplexer gain and channel and transfer of digital word to the computer can be written as follows:

```
TAD XX
ADSG        /LOAD MULTIPLEXER GAIN WORD
CLA
TAD YY
ADSC        /LOAD MULTIPLEXER ADDRESS WORD
ADSF        /SKIP IF A/D CONVERTER FLAG IS 1
JMP .—1     /JUMP BACK AND SENSE FLAG AGAIN
```

ADRB /READ A/D BUFFER AND TRANSFER CONTENTS TO /AC00-11. CLEAR A/D CONVERTER FLAG.

Type AF04A Guarded Scanning Integrating Digital Voltmeter

The Type AF04A is a Guarded Scanning Integrating Digital Voltmeter system, with wide dynamic range and high common-mode rejection, and is capable of expansion to 1000 channels. The AF04A is used with the PDP-8/E to multiplex up to 1000 three-wire analog channels into a six-decimal digit integrating digital voltmeter (IDVM). Each digit is BCD-coded for input and display by the IDVM. Full scale ranges are from + or − 10 mV to + or − 300V, with automatic ranging, 300 percent overranging, and a usable 5 μs 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 of process monitoring where a wide range of signals requires large dynamic range. The 10mV range has 0.001 percent resolution, and, coupled with a common-mode noise rejection greater than 140 dB at all frequencies, allows accurate direct measurement of thermocouples, strain gauges, load cells, and other low-level transducers without additional amplification.

The AF04A IDVM, 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 auto-ranging, as well as the integration time of the integrating digital voltmeter.

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 IDVM allow for 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 may be manually controlled, completely independent of the computer.

Specifications

Full scale + or −	10mV, 100mV, 1V, 10V, 100V, 300V and automatic ranging
Over-ranging	300% on all but highest range
Maximum Input Voltage	300V
Resolution	5μV (usable), 0.1μV (LSB)
Accuracy (overall worst case with daily calibration)	+ or − 0.004% of reading or + or − 0.01% of full scale
Temperature Stability (RMS full scale and zero drift)	+ or − 0.006%/day

Temperature coefficient	+ or — 0.003% of reading/degrees C
Full scale	+ or — 0.002% of full scale/degrees C
Zero	(+ or — 0.006% of full scale/degrees C on 10mV and 1V range)
Line voltage stability	+ or — 0.005%/10% change
Maximum common-mode voltage	+ or — 300V from power line ground
Common-mode rejection (166.6ms 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 megohms/V
10, 100, 300V ranges	10 megohms
Internal secondary standard	
Value	+ or — 1.000V
Accuracy	+ or — 0.002% traceable to National Bureau of Standards
Stability	+ or — 0.005%/month
Temperature coefficient	negligible

Selected Resolution

DC Voltage Range	0.001% Maximum Reading	0.001% Resolution	0.01% Maximum Reading	0.01% Resolution	0.1% Maximum Reading	0.1% Resolution
10mV	30.0000mV	0.1μV	030.00mV	1μV	0030.00mV	10μV
100mV	300.000mV	1μV	0300.00mV	10μV	00300.0mV	100μV
1000mV	3000.00mV	10μV	03000.0mV	100μV	003000.mV	1mV
10V	30.0000V	100μV	030.000V	1mV	0030.00V	10mV
100V	0300.000V	1mV	0300.00V	10mV	00300.0V	100mV
1000V*	0300.00V	10mV	00300.0V	100mV	000300.V	1V

*1000V range is scanner-limited to 300V peak maximum.

Scanning Speed
(Programmed Range)

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

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

Programming

The IOT instructions associated with the scanning IDVM 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
Execution Time: 2.6 μs
Operation: Transfers the contents of the accumulator to the AF04A control register. Control Word 1 is used only if a range change is required (see Figure 7-17).

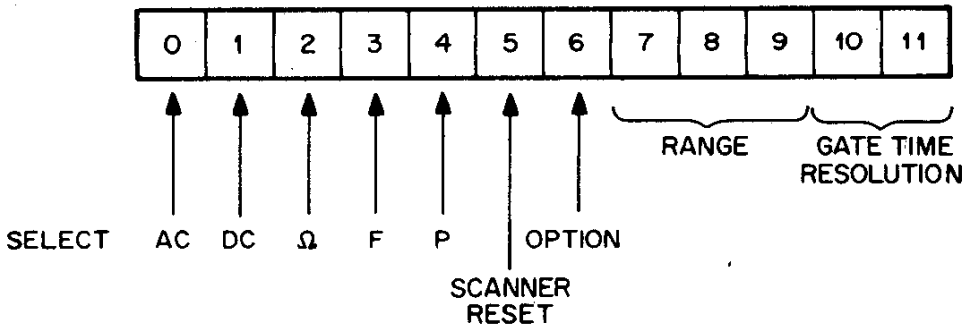


Figure 7-24 Control Word 1 (from Computer)

Select Channel and Convert (VCNV)

Octal Code: 6541
Execution Time: 2.6 μs
Operation: Transfers the contents of the accumulator to the AF04A channel address register. Automatically digitizes the analog signal on the selected channel (see Figure 7-18).

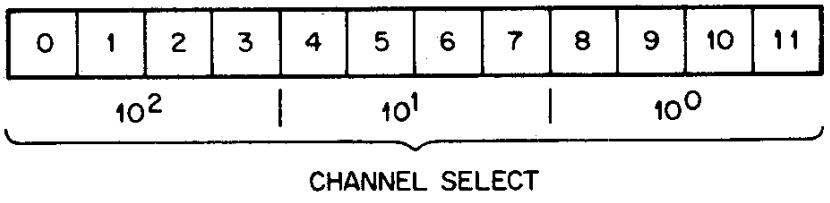


Figure 7-25 Control Word 2 (from Computer)

Index Channel and Convert (VINX)

Octal Code: 6544
Execution Time: *2.6 μs*
Operation: Increments the last channel address by one and automatically digitizes the analog signal on the selected channel. The contents of the control register are unchanged.

Skip on Data Ready (VSDR)

Octal Code: 6561
Execution Time: 2.6 μs
Operation: Set a data ready flag when the scanning voltmeter has selected a channel and digitized the analog signal. This instruction is used to test for the data ready flag.

Read Data and Clear Flag (VRD)

Octal Code: 6562
Execution Time: 2.6 μs
Operation: Transfers the content of the selected byte of the IDVM output word to the accumulator and clears the data ready flag. The first data available after the flag is set is always byte 1. Subsequent bytes are program-selected using the Byte Advance command (see Figure 7-26).

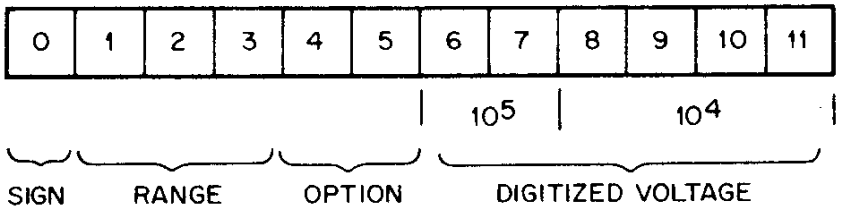


Figure 7-26 Data Word (to Computer)

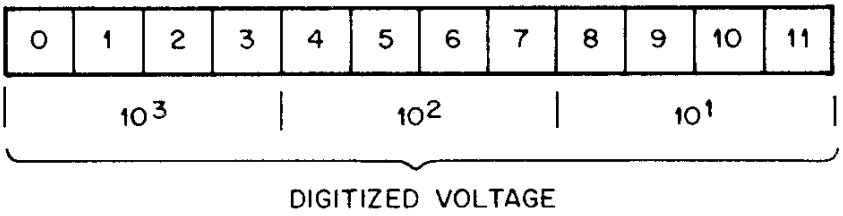


Figure 7-27 Data Word (to Computer)

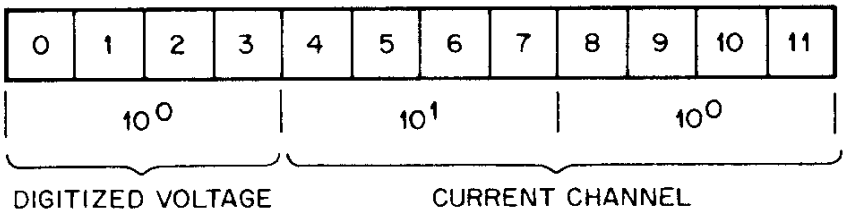


Figure 7-28 Data Word (to Computer)

All address and digitized data is in 8-4-2-1 BCD format.

Byte Advance (VBA)

Octal Code: 6564
Execution Time: 2.6 μs
Operation: The total data word from the AF04A is 36 bits long. The first data word after the flag is set is always the 12 most significant bits. The BYTE ADVANCE command requests the next 12 most significant bits. When the data is available, the data ready flag is set again. To select the 12 least significant bits, a second BYTE ADVANCE command is required. When the data is available, the data ready flag is set again.

Sample Current Channel (VSCC)

Octal Code: 6571
Execution Time: 2.6 μs
Operation: Digitizes the analog signal on the current channel. 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 ms per sample.)

Frequency and Period Measurement Options for AF04A

A separate input permits the IDVM to be used as a frequency counter capable of counting to 2 mHz with selectable gate times of 1, 10, and 100 ms, providing measurement resolution of 10 Hz. Increased accuracy at low frequencies (to 10 kHz 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. Frequently 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 IDMV for frequency or period measurements.

Specifications (See Figure 7-29)

Frequency Measurements
Range: 10 Hz to 2 mHz
Sensitivity: 100 mV rms or −1V pulses, at least 0.3 μs wide at 50% points. 100V rms maximum working voltage.
Input Impedance: 22K shunted by less than 1000 pF, including internal cabling.
Time Base: 100 kHz crystal oscillator with initial accuracy of + or − *0.0005%, long-term stability + or − 0.001%/wk; temp. coefficient* + or − 0.0002%/degrees C.
Period Measurements
Range: 1, 10, and 100 period average. Input frequency from 10 Hz to 25 kHz sine wave or 0.1 pps to 25,000 pps.

Sensitivity: 100 mV rms or – IV pulses, at least 0.3μs wide at 50% points. 100V rms maximum working voltage.

Input Impedance: 22K shunted by less than 1000 pF, including internal cabling.

Accuracy: $\pm$ 1 count + time base accuracy + trigger error. Trigger error $<$ $\pm$ 0.03% for 100 mV rms sine wave with 40 dB signal-to-noise ratio.

Time Base: 100 kHz crystal oscillator with initial accuracy of $\pm$ 0.0005%, long-term stability $\pm$ 0.0001T/wk; temp. coefficient $\pm$ 0.0002%/degrees C.

Selected Resolution

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

Figure 7-29

Additional AF04-A Options

A type AF04-X expansion Mounting Panel is available which provides an additional 200 channels. For each 10 channels implemented, the Type AF04-S 10-Channel Guarded Reed Relay Multiplexer Switch is required.

Thermocouple reference junctions
Extended scanner for more than 1000 channels

AA50-A Digital-To-Analog Conversion Subsystem

The AA50-A DAC is a general-purpose, program-controlled DAC subsystem that converts 12-bit (11 bits plus sign) words into analog outputs having a continuously adjustable full-scale range of 0 to ± 10V at 10 mA.

The AA50-A is housed in a H911 type mounting panel and is furnished complete with power supply, I/O cables, control and interface logic, and up to six DAC modules, each providing one analog output. The unit interfaces with the external bus of the PDP-8/E. All operations are controlled by IOT instructions, including the selection of the DAC module to receive the 12-bit output word. Each DAC module contains a buffer register and a scaling amplifier with reference mounted on the same module.

For an output function, the computer issues an IOT instruction that specifies the DAC module to receive the 12-bit word. The control logic of the AA50-A decodes the IOT, performs input gating for the 12-bit word from AC0-11, and loads the words into output buffer of the designated DAC module. The word remains in the output buffer until the buffer is updated by another input; thus, the resulting analog output is available until updating occurs.

Specifications

Digital Input	Parallel, 11 bits plus sign in two's complement form
Coding	3777 (octal) = + 10V 0000 (octal) = 0V 4000 (octal) = — 10V
Standard Analog Output	0 to ±10V@ 10 ma (adjustable)
Settling Time	20μs to ½ LSB (measured at output connector with no capacitive loading)
Accuracy	0.05% of full scale
Linearity	± ½ LSB (± 2.44mV for ± 10V DAC output)
Capacitive Loading	0.1 μf at output connector will not cause instability

Programming

The following instructions are associated with AA50-A operation:

Select DAC 0 (DACS0)

Octal Code: 6551
Execution Time: 2.6 μs
Operation: Transfers content of AC to DAC module 1 and converts it to analog output.

Select DAC 1 (DACS1)

Octal Code: 6552
Execution Time: 2.6 μs
Operation: Transfers content of AC to DAC module 2 and converts it to analog output.

Select DAC 2 (DACS2)

Octal Code: 6553
Execution Time: 3.6 μs
Operation: Transfers content of AC to DAC module 3 and converts it to analog output.

Select DAC 3 (DACS3)

Octal Code: 6554
Execution Time: 2.6 μs
Operation: Transfers content of AC to DAC module 4 and converts it to analog output.

Select DAC 4 (DACS4)

Octal Code: 6555
Execution Time: 3.6 μs
Operation: Transfers content of AC to DAC module 5 and converts it to analog output.

Select DAC 5 (DACS5)

Octal Code: 6556
Execution Time: 3.6 μs
Operation: Transfers content of AC to DAC module 6 and converts it to analog output.

Device codes 56 and 57 are used when additional (up to three total) AA50's are required.

AA05-A/AA07 Digital-to-Analog Converter and Control

The AA05 Digital-to-Analog Converter (DAC) provides housing power and control for up to 24 10-bit DAC modules. The AA07 Expansion Unit extends the capacity of the system to 64 channels of DAC.

Each conversion channel may use any of four printed circuit card DAC modules. These modules include two single-buffered units, Types A608 and A609, and two double-buffered units, Types A610 and A611. A608 is a single-buffered, 10-bit DAC, with unipolar output (0V to + 10V). Type A609 is a single-buffered, 10-bit DAC with bipolar output and variable offset. A610 and A611 are similar to A608 and A609, respectively, except that the former are double-buffered units.

The principal power supply furnishes all power for up to 64 DAC modules, with the exception of the −10V reference power. Reference power is furnished by the Type H706 Reference Power Supply, which is optional to the AA05/AA07 unit. A maximum of five H706 supplies can be allocated to the various DAC channels, two of which are in the AA05 and three of which are in the AA07.

Each DAC in the AA05/AA07 DAC and expansion unit are used with the PDP-8/E computer to control up to 64 DAC channels. Both the DAC address and the digital word to be converted are program-controlled as two I/O data words for 12-bit computers. The DAC address is stored in the AA05 and remains there until changed by the program for fast updating of any channel.

Six indicators on the front panel of this device indicate the binary address of the DAC channel currently being addressed. All data bits and I/O transfer commands are buffered to present a minimum load to the computer bus even with 64 DACs in use. The AA07 expansion assembly allows expansion to 64 single- or double-buffered DACs.

The AA05/AA07 consists of a 10-bit buffer register, level converters, a precision divider network, and a current-summing amplifier capable of driving large external loads. Provisions are made for double-buffering and bipolar output voltage where required. A precision reference voltage, supplied externally by the H706 power supply, ensures greater efficiency and optimum scale-factor matching in multiple-channel systems. The AA05/AA07 DAC utilizes four separate instructions. These instructions clear the DAC address register, transfer the contents of AC(0-9) to the input register of the selected DAC, and update all double-buffered channels (if applicable).

Specifications

Standard Output	Unipolar, 0V to + 10V at 10 ma
Optional Output	Bipolar, + or − 5V or + or − 10V
Output Impedance	Less than 1 ohm
Temperature Coefficient	0.1mV/degrees C plus temperature coefficient of reference supply (worst-case for DEC reference supply is 0.6mV/degrees C)
Resolution	0.1% of full-scale
Accuracy	+ or − 5mV
Settling Time (Full-scale)	5 μs for 1 DAC module. Less than 100 μs for up to 12 DAC modules
Environmental	0 degrees to 50 degrees C
Power	7A (max) at 115V, 60Hz

Programming

The following instructions are associated with the AA05A DAC:

Clear DAC Address (DACL)

Octal Code: 6551
Execution Time: 2.6 μs
Operation: Clears DAC address register.

Load DAC Address (DALD)

Octal Code: 6552
Execution Time: 2.6 μs
Operation: Loads content of AC in DAC address register.

Load DAC Input Register (DALI)

Octal Code: 6562
Execution Time: 2.6 μs
Operation: Loads content of AC in DAC input register specified by DAC address register.

Update All Channels (DAUP)

Octal Code: 6564
Execution Time: 2.6 μs
Operation: Updates all double-buffered channels to provide DAC outputs to loads.

Universal Digital Controller (UDC)

The UDC is a digital input/output system with a controller having 256 12-bit addressable channels. Each channel can be used as an input or output path. When used for output functions, a channel can control 12 discrete off/on devices such as relays, flip-flops, etc. When used for input functions, a channel can be used to interrogate the status of 12 discrete off/on sources such as switches, relays, and flip-flops. Thus, the UDC provides the capability for accessing a total of 3072 discrete digital points either for input (status) or output (control) functions in 12-bit combinations.

All input/output data is handled in the form of 12-bit words. The data is unstructured except for the generic module type and address word read to the computer after an interrupt. Accumulator bits 0 through 3 receive a four-bit code denoting the generic type or function performed by the module specified and by the eight-bit address in AC04 through AC11.

Any UDC channel or word can be input or output. When dedicated for an input function, the type of interrupt desired must be specified by the program. The type of interrupt is defined by AC10 and AC11 as follows:

AC10	AC11	TYPE OF INTERRUPT
0	0	None
0	1	Deferred processing
1	0	Immediate processing
1	1	Both

Once an interrupt type is selected and an interrupt occurs, the UDC locates the interrupting address, using an address scan cycle. This cycle requires to 20 μs. Once the interrupting address has been located, the address and module generic type are made available to the computer.

The functional capability to EXCLUSIVE OR an AC word with an I/O word is provided by Change-Of-State (COS) gating. The AC bits are loaded into the COS register with the Load Previous Status (octal 6357) IOT. Data from the word of I/O presently addressed is at the gates and the EXCLUSIVE-OR function is performed.

Two bits, pulse open and pulse close, are hard-wired at the I/O word in question; their purpose is to mask out data changes that are not pertinent. The EXCLUSIVE-OR function is defined by the following:

DATA BIT	AC BIT	PULSE OPEN	PULSE CLOSE	COS OUT
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	1
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0

The UDC is housed in an H964 industrial-type cabinet(s) complete with cooling fans and low-voltage supplies. The smallest configuration of the UDC consists of one file in a single cabinet. This file contains the IOT and interface control logic; the address scanner logic, and capabilities for handling up to four I/O channels or words. This basic system can be expanded from four channels to 256 channels in four-channel groups by adding system units, file and cabinets.

Specifications

Operating Modes	Digital Output Digital Input *Interrupt or Noninterrupt*
Data Format	Parallel, 12-bit unstructured
Addressing Capability	256 12-bit channel or words or 3072 discrete digital points

Input/Output Module Selection	Directly addressable and location independent
Interrupt Module Identification	4-bit Generic code type
Interrupt Structure	Immediate or deferred by module assignment and program
Interrupt Scan or Address Location Time	5 μs typical
I/O Data Rate	105 12-bit word per second
System Clock Rates	3 clock rates available to each I/O channel or word (1) 60 Hz, 6.3 VAC (line power) (2) 175 Hz, to 1.75 kHz (3) 1.75 Hz to 17.5 kHz
Standard Output Drive Capabilities	250 ma at up to +55V (suitable for relay drivers)
Standard Inputs	2 amps, 500V, 100VA, (Mercury-wetted relays)
Functional Modules Available	15 ma at + 6V. Contact Sense Contact Interrupt Flip-Flop Relay Single-Shot Relay Flip-Flop Driver Single Shot Driver Latching Relay Input/Output Counters Digital to Analog Converters

Programming

The following instructions are associated with UDC operation:

Skip on Scan Not Busy (UDSS)

Octal Code: 6351
Execution Time: 2.6 μs

Operation: Skips the next instruction if Scan Not Busy flag is a one, denoting that the address scanner has located the interrupt channel, so that UDC can be serviced.

Start Interrupt Scan (UDSC)

Octal Code: 6353
Execution Time: 3.6 μs
Operation: Enables address scan function if interrupt flag is set and interrupt type (immediate or deferred) is present.

Read Address and Generic Type (UDRA)

Octal Code: 6356
Execution Time: 3.6 μs
Operation: Transfers the generic type and address to the AC after interrupting address has been located.

Load Previous Status (UDLS)

Octal Code: 6357
Execution Time: 4.6 μs
Operation: Loads content of AC into COS register and reads the result of the EXCLUSIVE OR function of the COS logic to AC.

Skip On UDC Flag and Clear Flag (UDSF)

Octal Code: 6361
Execution Time: 2.6 μs
Operation: Skips the next instruction and clears the UDC flag if UDC interrupt Flag is set.

Load Address (UDLA)

Octal Code: 6363
Execution Time: 3.6 μs
Operation: Loads 8-bit address from AC into address register scanner.

Enable UDC Interrupt Flag (UDEI)

Octal Code: 6364
Execution Time: 2.6 μs
Operation: Sets the interrupt enable flip-flop so that UDC can generate interrupt requests.

Disable UDC Interrupt Flag (UDDI)

Octal Code: 6365
Execution Time: 3.6 μs
Operation: Clears interrupt enable flip-flop so that UDC cannot generate interrupt requests.

Clear AC and Read Data (UDRD)

Octal Code: 6366
Execution Time: 3.6 μs
Operation: Clears the AC and transfers data specified by address of address scanner register to AC.

Load Data and Clear AC (UDLD)

Octal Code: 6367
Execution Time: 4.6 μs
Operation: Transfers content of AC to address specified by address scanner register, then clears AC.

TYPE VW01 Writing Tablet

The VW01 Writing Tablet converts graphical information, in the form of X and Y coordinates, to digital data that can be input to a digital computer. The major components of the VW01 are the writing tablet, spark pen, component box, and computer interface logic.

The user places a sheet of paper on the writing tablet and draws sketches, schematics and hand-written symbols or characters using the special ball-point pen. The sound of the spark emitted by the pen is picked up by microphones located along the X- and Y-axes of the writing tablet. The time lapse, from spark emission until sound is picked up by each bank of microphones, is accurately measured to provide a digital record of the X and Y coordinates of the spark pen location on the paper.

The digitized graphic data is input to a digital computer for immediate or delayed processing.

The VW01 provides an efficient man/machine interface with digital systems that allows the user complete freedom of expression.

The VW01 consists of the writing tablet, VW01-AP interface and BC08B I/O cable. The KA8-E positive I/O Bus is required.

VW01-MX Multiplex Option

The VW01-MX Multiplex option allows up to four VW01 Writing Tablets to be used with a single VW01 computer interface. This option consists of the VW01-MX Multiplexer and up to four VW01-MA Writing Tablet assemblies. If the VW01-MX Multiplex option is included as part of the system, additional cabling and interface requirements must be considered.

Modes of Operation

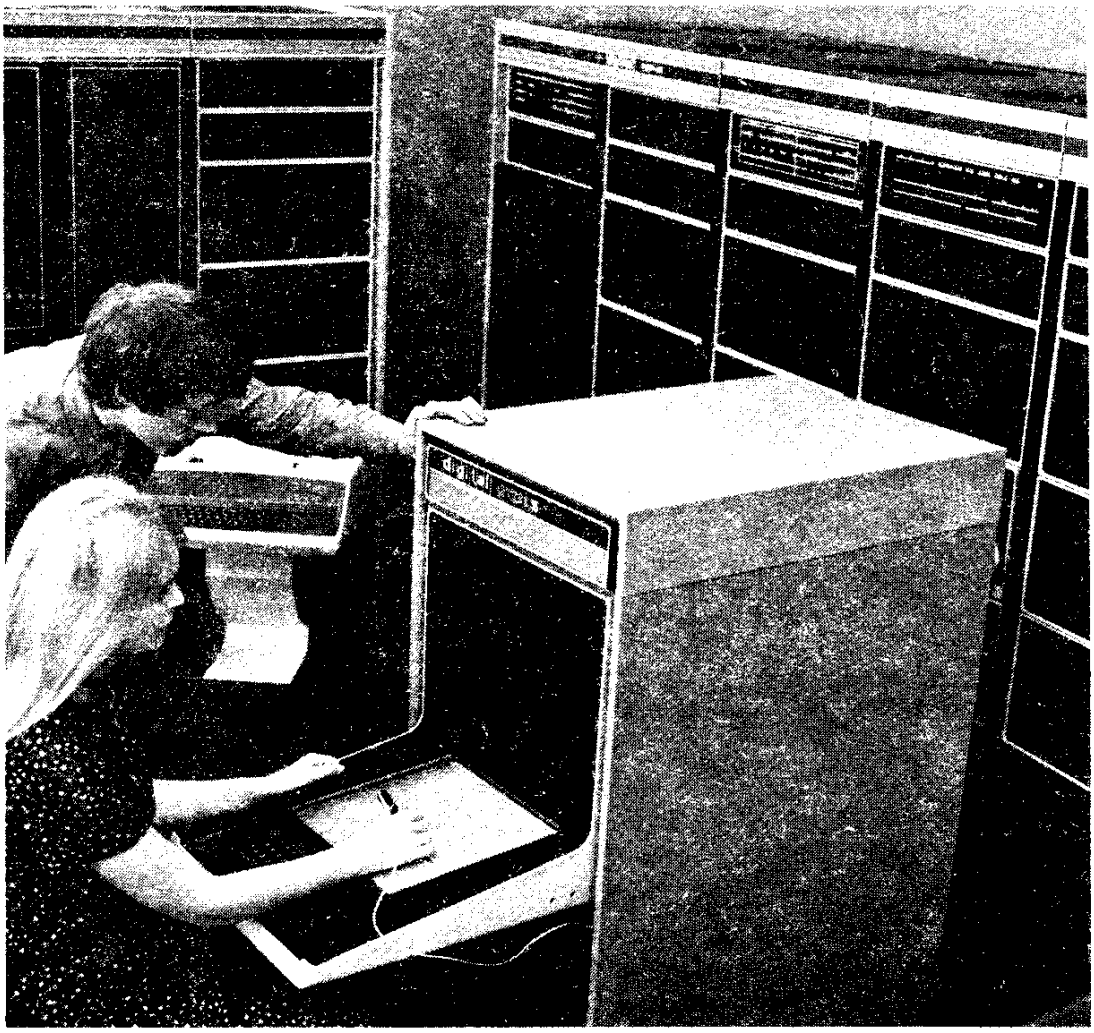
The VW01 operates in either of two modes: Single Point or Data Input.

Single Point Mode—In the Single Point mode of operation, a single spark is generated each time the spark pen is pressed against the writing surface. The spark is initiated by the closure of a microswitch within the spark pen. The Single Point mode is used if the operator desires to plot points. For example, to plot points at four different locations, he positions the pen point at each location. Then, by pressing and releasing the pen at each position, the corresponding X-Y coordinate pairs are sensed and digitized.

Data Input Mode—In the Data Input mode, a continuous series of sparks are generated at a constant rate, under control of clock pulses. The X-Y coordinate pairs are continuously generated and input to the computer. This mode allows the user to draw continuous lines, circles, curves, etc. that can be displayed on a CRT.

The normal data rate at which X-Y coordinate pairs can be generated is 200 Hz.

The Data Input mode can also be used for tracking applications. Tracking is a technique that is used to move a cursor, or other type of position indicator to a specific X-Y coordinate location on the display. With appropriate programming, the cursor will follow the spark pen movement. The spark pen is then pressed on the writing tablet at a specific X-Y coordinate location to draw on the display.



VW01 Writing Tablet

Digitizing the Graphic Data

At the time a spark is generated, X and Y clock pulses are initiated and the X and Y registers are incremented until the sound of the spark is received by the X and Y microphones. As soon as a microphone detects the sound, the associated X or Y clock pulses are inhibited and the register stops incrementing. The binary numbers contained in the X and Y registers will be directly proportional to the X and Y coordinates of the position at which the spark was emitted.

For example, if a spark is generated at a point on the writing tablet that is four inches from the X-axis microphone and another spark is generated at a point on the writing tablet that is eight inches from the X-axis microphone, the time for the sound wave to travel from the point of generation to the X-axis microphone would be twice as long as the time required for the first spark. Thus the binary number contained in the X register for the second spark generated would be approximately double the value of the binary number for the first spark generated. When the spark pen is moved, a different set of binary numbers, proportional to the new spark pen position are entered into the X and Y registers.

Computer Input

When the binary numbers that represent a pair of X-Y coordinates are settled in the 10-bit X and Y registers, the VW01 computer interface logic requests a program interrupt. When the computer services the interrupt request, the 10-bit digital words specifying each coordinate are successively read into the computer AC by IOT instructions.

Specifications

Component	Dimensions (in.)		
	Height	Width	Depth
Writing tablet	13	13	1.5
Interface logic rack	10.44	19	12
Component box	3.18	19	5

The spark pen is 5.5 inches long and 0.25 inch in diameter

Digital Resolution	10-bit resolution in both X and Y axes.
Graphic Resolution	1000 x 1000 line pairs; 90 lines per inch
Reproducibility	One (least significant) bit in 1000, in both X and Y axes.

Drift

Constant Temperature	With the spark pen stationary, the X and Y registers will not vary more than ± one bit in 1024.
4.4° to 32°C	With the spark pen stationary, the X and Y registers will vary ± two bits per thousand per degree change Centigrade.
+ 40° to 90°F	With the spark pen stationary, the X and Y register will vary ± 1.4 bits per thousand pen degree change Fahrenheit.

Data rate	
FAST SCAN	200 X-Y coordinate pairs per second. The data rate can be decreased to 1 X-Y coordinate pair per second.
SCAN	100 X-Y coordinate pairs per second per tablet; used only with VW01-MX Multiplex option.
Single Point	Determined by user's manual activation of the spark pen microswitch.
Multiplex latency	With the VW01-MX Multiplex option, the interval from each writing tablet DATA READY flag to the time the next writing tablet is enabled is 1.4 msec.
Spark pen longevity* (typical)	
Spark gap	50 x 10^7 discharges, minimum
Ink Cartridge	5000 ft. of inked lines
Writing tablet surface	11 x 11 inches
Input power requirements	115V, 50/60-Hz + 2%, single phase, 17-30A, or 230V, 50 Hz + 2%, single phase, 8-15A.
Operating temperature range	+40 to +90°F (4.4 to 32°C)
Operating humidity range	20 — 55% relative humidity, without condensation.

Programming

The following instructions are used to program the VW01.

Set Tablet Controls (WTSC)

Octal Code: 6054

Operation: The following functions are cleared by I/O Buffered Power Clear. The Set Tablet Controls IOT, with the appropriate bit set, sets or clears the following functions, depending upon the bit selected.

ACCUMULATOR BITS

0	1	2	3	4	5	6	7	8	9	10	11
		Single Point		Pen Data Intr En		Data Ready Intr En		FAST SCAN	SCAN	Writing Tablet En	
		CLR	SET	CLR	SET	CLR	SET			CLR	SET

* The spark gap length of service is extended by using the ON/OFF switch located on the writing tablet.

Writing Tablet EN—SET
AC bit 11=1
The writing tablet is initially enabled for operation in FAST SCAN. To change to Single Point, SCAN Multiplex, or Single Point, the appropriate function must be selected.

Writing Tablet EN—CLR
AC bit 10 = 1
The writing tablet is disabled from performing any control functions.

SCAN
AC bit 09 = 1
The writing tablets are enabled to operate in the multiplex mode. Up to four writing tablets can be multiplexed. Each tablet will have a data rate of 100 Hz and the tablets will operate in sequential order.

FAST SCAN
AC bit 08 = 1
Enables the logic for the operation of one tablet at a data rate of 200 Hz. Using the Select Tablet IOT with the appropriate bit set, a single writing tablet can be selected for Data Input operation. I/O Buffered Power Clear always selects writing tablets 01, and FAST SCAN.

Data Ready Intr EN—SET
AC bit 07 = 1
The DATA READY flag is enabled onto the I/O interrupt bus.

Data Ready Intr EN—CLR
AC bit 06 = 1
The DATA READY flag is disabled from the I/O interrupt bus.

Pen Data Intr EN—SET
AC bit 05 = 1
The PEN DATA flag is enabled onto the I/O interrupt bus.

Pen Data Intr EN—CLR
AC bit 04 = 1
The PEN DATA flag is disabled from the I/O interrupt bus.

Single Point—SET
AC bit 03 = 1
The writing tablet is enabled for a single pair of X-Y coordinates. The microswitch in the pen must be activated. An X-Y coordinate pair is present when the DATA READY flag is set.

Single Point can be selected for the multiplex of the writing tablets. When the microswitch in any of up to four pens is activated, the associated tablet takes control of the I/O bus and an X-Y coordinate pair is ready when the DATA READY flag is set. The tablet that set the DATA READY flag will then have to be cleared by using the Select Tablet IOT with the appropriate bit set. All tablets should be cleared before initiating Single Point (Multiplex) operation.

Right/Left
This bit indicates the current setting of the RIGHT/LEFT switch. A logical 1 indicates RIGHT and a logical 0 indicates LEFT.

Single Point
A logical 1 indicates Single Point mode of operation.

TAB 01 through TAB 04
Tablet 01 indicates the writing tablet 01 ON/OFF switch is in the ON position and writing tablet 01 is selected. With the multiplex option, tablet 01 is set by Buffered Power Clear or the Clear All Flags IOT, and writing tablets 02, 03, and 04 are cleared. TAB 02 through TAB 04 are *logical 1 only when the associated ON/OFF switch is ON and that writing* tablet is selected.

Pen Data Intr EN
Indicates the status of the pen data interrupt enable.

Data Ready Intr EN
Indicates the status of the DATA ready interrupt enable.

SCAN
Indicates whether SCAN or FAST SCAN has been selected.

Writ Tab EN
Indicates the status of the writing tablet enable.

Clear Data Ready Flag (WTCD)
Octal code: 6061
Operation: This IOT is issued to clear the DATA READY flag.

Single Point—CLR
AC bit 02 = 1
The Single Point mode operation will be disabled.

Read X (WTRX)
Octal code : 6052
Execution Time: 2.6 μs
Operation: The Read X IOT "OR"s 10 bits from the X register into the processor accumulator.

ACCUMULATOR BITS

0	1	2	3	4	5	6	7	8	9	10	11
		X0	X1	X2	X3	X4	X5	X6	X7	X8	X9

Read X Word Format

Read Y (WTRY)
Octal code: 6062
Execution *Time:* *2.6 μs*
Operation: The Read Y IOT "OR"s the 10-bit Y register into the processor accumulator. The Y coordinate bits are read into the same accumulator bit positions as indicated for the X coordinate bits.

Read Status (WTRS)
Octal code: 6072
Execution Time: 2.6 μs
Operation: The Read Status IOT reads the flag and status indicator bits into the processor accumulator as follows:

ACCUMULATOR BITS

0	1	2	3	4	5	6	7	8	9	10	11
DATA RDY FLAG	PEN DATA FLAG	RIGHT = 1 LEFT = 0	Single Point	TAB 01	PEN DATA INTR EN	TAB 02	DATA RDY INTR EN	TAB 03	FAST SCAN = 1 SCAN = 0	TAB 04	WRIT TAB EN

Status Word Format

Logical 1 bit indicates condition selected.

DATA READY Flag
The DATA READY flag is set when an X-Y coordinate pair is updated to the current position of the spark pen on the writing tablet surface.

PEN DATA Flag
The PEN DATA flag is set when an X-Y coordinate pair is updated to the current position of the spark pen and the spark pen microswitch is activated.

Clear Pen Data Flag (WTCP)
Octal code: 6051
Execution Time: 2.6 μs
Operation: This IOT is issued to clear the PEN DATA flag.

Writing Tablet Skip (WTSK)
Octal code: 6071
Execution Time: 2.6 μs
Operation: The writing Tablet Skip IOT can only be used to perform a computer program skip on a writing tablet I/O interrupt. The two writing tablet flags that can provide an I/O interrupt are the DATA READY flag and the PEN DATA flag. The appropriate flag has to be enabled onto the I/O interrupt bus using the data ready interrupt enable or the pen data interrupt enable.

Select Tablet (WTSE)
Octal code: 6074
Execution Time: 2.6 μs
Operation: The Select Tablet IOT is used when the VW01-MX multiplex option is implemented, in conjunction with the TAB 01 through TAB 04 control bits. When FAST SCAN is selected, only one writing tablet can be active, and this tablet can be selected by setting the appropriate tablet control bit in the accumulator.

ACCUMULATOR BITS								
0	4	5	6	7	8	9	10	11
		SET DATA READY	SET PEN DATA	CLR SET XY	TAB 01	TAB 02	TAB 03	TAB 04

Select Tablet Word Format

Clear Set XY (WTMN)

Octal code: 6064

Execution Time: 2.6 μs

Operation: The Clear Set XY IOT is used only for maintenance purposes. When the CLR SET XY bit position in the accumulator is cleared and the Clear Set XY IOT is issued, the X and Y registers will be cleared. When the CLR SET XY bit position is set and the IOT is issued, the X and Y registers will be set.

POSITIVE I/O BUS DATA COMMUNICATIONS EQUIPMENT OPTIONS

DC02-F 8-Channel Multiple Teletype Control Control

The DC02-F is a multi data station control allowing the user to add from one to eight serial to parallel, parallel to serial asynchronous data channels (Teletype, dataphone, or other serial data devices). A DC02-G module set, consisting of a bus driver, a receiver, and transmitter module, is used for each serial data device to be controlled. Up to four DC02-F controls can be used per system allowing the control of 32 serial data devices. This option operates on the positive I/O bus.

The 32 data stations are selected in the following manner:

Bits 8-11 of the accumulator select the DC02-F control which controls up to eight (8) stations (Figure 7-30). Bits 0-7 of the accumulator select the station within the DC02-F (Figure 1B). Data in the accumulator can be transmitted simultaneously to more than one station, but data from only one station can be received into the accumulator at any given time. Each station has two status flags, a receiver and a transmitter flag. When the receiver flag is set, it indicates that an eight (8) bit word has been assembled in the receiver register and is ready for transfer into the accumulator. When the transmitter flag is set, it indicates that an 8-bit word has been transmitted and the station is ready to transmit another word.

When a flag is set and the "interrupt on" flip flop is set, an interrupt request signal is generated. This level is sensed by the interrupt bus and with the IOT 6125 instruction.

The status of the station reader flags is read into AC bits 0-7 with IOT 6123 and bits 8-11 are cleared. The status of the transmitter flags is read into AC bits 0-7 with IOT 6113 and bits 8-11 are cleared. The status of the station select flip flop is read into AC bits 0-7, the "interrupt on" flip flop into AC bit 11, and AC 8-10 are cleared with IOT 6127. IOT's 6111 and 6121 check the flags individually. In all cases, a station must be selected to check its flag status.

The type BC01-A modem interface adapter is available for modifying the DC02-F for compatibility with EIA standard RS-232C interface logic levels.

With this adapter, the bit rate can be increased to 100K baud for driving medium to high speed asynchronous modems. Transmission distance is 1500 ft. maximum (environment dependent) for standard Teletype levels. EIA transmission distance is limited only by characteristics of modem and associated facility (refer to DEC Communications Equipment Handbook for Selection of Modems).

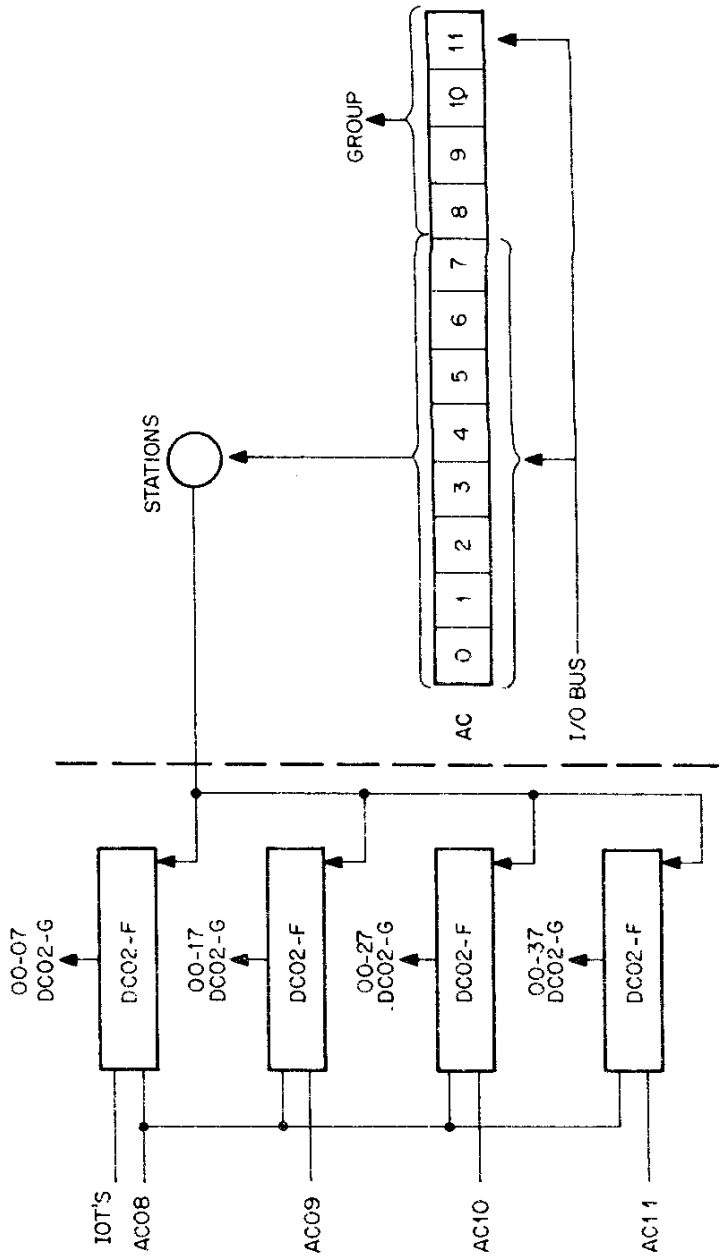


Figure 7-30 Station Selection Operation

DC02 Set Up Procedure

1. The jumper board in B05 is used to select the AC bit corresponding to the group the user desires.

AC BIT	JUMPER		GROUP	STATIONS
8	U1	S2	1	(0-7)
9	V1	S2	2	(10-17)
10	T2	S2	3	(20-27)
11	R2	S2	4	(30-37)

2. *The jumper board in B05 is also used to select the clock frequency desired. Refer to the TCF print for jumping information.*

3. The jumper board in B08 is used to select the frequency desired for the receivers and transmitters. Refer to the TTS print.

Clock Frequency Selection

It is recommended that the reader should carefully review the following facts before attempting to select baud rates. Baud rates are defined as "bits per second," for example, a device has 11 bits per character and transmits 10 characters per second; therefore:

$$\underset{\text{character}}{11 \text{ bits}} \times \underset{\text{second}}{10 \text{ characters}} = \underset{\text{second}}{110 \text{ bits}} = 110 \text{ baud}$$

The input clock frequency must be eight (8) times the baud rate for the M706 receiver and two (2) times the baud rate for the M707 transmitter. Timing for the M706 and M707 pairs is available from two different clock sources (they, of course, may be the same). The M452 is used with Teletype-like devices. It has two clock outputs, 220 Hz and 880 Hz, exactly suitable for 110 baud operation. The other clock source is a crystal oscillator (M405). Its frequency is extremely accurate and must be specified when ordering the DC02-F. Refer to Logic Handbook for further information regarding these modules. The DC02-F control has three clock frequency options designations, FA, FB, and FC. The FA has a variable RC control clock (M452) primarily used for 110 baud rate. The FB consists of two crystal control clocks (M405) and two frequency dividers (M216) for operations from 50 baud to 100K baud. The FC is a combination of the FA and FB. It has a maximum of five different frequencies for handling up to eight serial data devices (provided some are operating at the same baud rate).

Each M405 clock is divided into six different frequencies by a binary type downcounter. Two sets of taps are provided from each counter in the event that a clock must control two different baud rates. Referring to the TCF print (counter flow), the reader will observe that the frequencies at taps 2 and 4 are a binary multiple of 16 times the clock frequency at taps 1 and 3, respectively. Therefore, if two different bauds must be controlled by one clock, the second must be a binary multiple of sixteen times the first. However, for special applications, if the second frequency is available at another binary multiple (other than 16) an engineering modification to the wire assembly will be necessary.

EXAMPLE: A DC02-F is to control devices at the following baud rates: 110 baud ASR-33 Teletype), 2.4K baud, 80K baud, and 5K baud.

First determine what DC02-F clock frequency option is required. Since there are only two clocks with frequency dividers available to control the three fast bauds, two of the bauds must be a binary multiple of 16. The 80K and 5K bauds are a binary multiple of 16, therefore, they are controlled by one of the clocks and one frequency divider. The other clock is needed to control the 2.4K baud and the M452 is needed to control the 110 baud. Therefore, since all clocks are required, the DC02-F option is used.

The next step is to determine the clock frequencies. All clock frequencies are calculated with respect to the highest frequency (that being the receiver). The general equations A and B are used to determine the clock frequency.

$$\text{Receiver Frequency} = 8 \times \text{baud} \times \text{MF} \quad \text{A}$$
$$\text{Transmitter Frequency} = 2 \times \text{baud} \times \text{MF} \quad \text{B}$$

The multiple factor (MF) is the binary down count factor at the point of frequency availability (the taps on the frequency divider). The baud is the actual baud desired. The 8 and 2 are the requirements of the receiver (M706) and transmitter (M707) module, respectively.

For the 110 baud, the frequencies are available on the M452. Since there is no frequency divider, there is no MF.

$$\text{Receiver Frequency} = 8 \times \text{Baud} = 8 \times 110 = 880 \text{ Hz}$$
$$\text{Transmitter Frequency} = 2 \times \text{Baud} = 2 \times 110 = 220 \text{ Hz}$$

For the 2.4K baud the MF depends on the tap used.

$$\text{Rec. Freq. (taps 1 or 3)} = 8 \times \text{Baud} \times 1 = 8 \times 2.4\text{K} = 19.2\text{K Hz}$$
$$\text{Rec. Freq. (taps 2 or 4)} = 8 \times \text{Baud} \times 16 = 128 \times 2.4\text{K} = 3.07\text{M Hz}$$

For the 80K and 5K baud both taps are used, but the MF are the same.

$$\text{Rec. Freq. (tap 1 or 3)} = 8 \times \text{Baud} \times 1 = 8 \times 80\text{K} = 640\text{K}$$
$$\text{Rec. Freq. (tap 2 or 4)} = 8 \times \text{Baud} \times 16 = 128 \times 5\text{K} = 640\text{K}$$

The MF of 8 and 128 (above MF) are constants shown on the baud rate chart. The transmitter and the receiver frequencies are paired on the frequency divider so that only the receiver frequency is calculated.

DC02F Options

Figure 7-31 illustrates the necessary ordering information when configuring a multiple Teletype System.

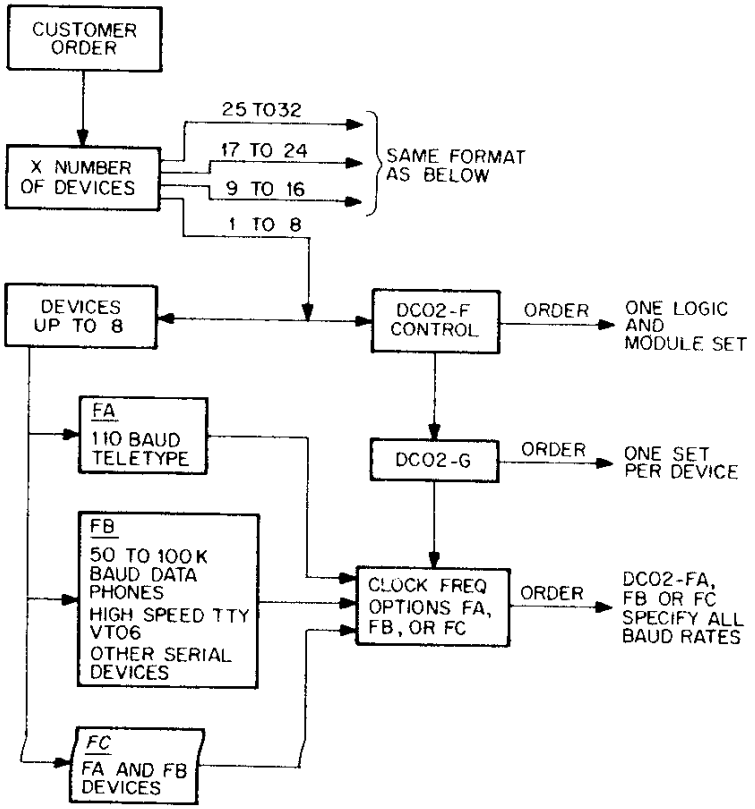


Figure 7-31 Multiple Teletype Configurations

Programming

The following instructions apply to group select, station select, receive and transmit functions with the DC02F 8-Channel Multiple Teletype Control.

Group & Station Select Operation and Status Instructions
* AC bits 8-11 select the group (Figure 7-30).
** AC bits 0-7 selects the stations.

Read Transmitter Flag (MTPF)

Octal Code: 6113 & Group*
Operation: Clear AC then read the specified group's transmitter flag status into AC bits 0-7.

Set Interrupt Flip-Flop (MINT)

Octal Code: 6115 & ACII
Operation: Set the "interrupt on" flip flop in all DC02-F controls.

Select Specified Station (MTON)

Octal Code: 6117 & Group & Station
Operation: Select the specified station in the specified group.

Read Receiver Flag Status (MTKF)

Octal Code: 6123 & Group*
Operation: Clear AC then read the specified group's receiver flag status into AC bits 0-7.

Skip on Interrupt Request (MINS)

Octal Code: 6125 & Group*
Operation: Skip if the DC02-F "interrupt request" is active in the specified group.

Read Station Status (MTRS)

Octal Code: 6127 & Group*
Operation: Clear AC then read the specified group's station/s status into AC bits 0-7 and the "interrupt on" status into AC11.

Housekeeping and Data Receive/Transmit Instructions.

The following IOT's function on station previously selected.

Skip on Keyboard Flag (MKSF)

Octal Code: 6111
Operation: Skip if keyboard flag is set.

Clear Receive Flag (MKCC)

Octal Code: 6112
Operation: Clear the Receive Flag.

Receive Operation (MKRS)

Octal Code: 6114
Operation: OR'S the receiver buffer with AC bits 4-11.

Combined MKRS & MKCC

Octal Code: 6116
Operation: Refer to 6112 and 6114.

Skip on Transmitter Flag (MTSF)
Octal Code: 6121
Operation: Skip if transmitter flag is set.

Clear Transmitter Flag (MTCF)
Octal Code: 6122
Operation: Clear Transmitter Flag.

Transmit Operation (MTPC)
Octal Code: 6124
Operation: Load AC bits 4-11 into the transmit register and transmit.

Combined MTCF & MTPC
Octal Code: 6126
Operation: Refer to 6122 & 6124.

DC02G Serial Line Interface Unit

A DC02-G is a module set which interfaces a serial type full duplex asynchronous data device. It is used in conjunction with the DC02-F, which is an eight channel multi-Teletype multiplex controller. For each serial data device controlled by the DC02-F, a DC02-G module set is added to control and transfer data to and from the device.

The module set consists of an M706 receiver module, an M707 transmitter module, and an M623 buffer module. The M706 receiver module independently assembles data, asynchronously, from a serial device and converts it to parallel form for transfer to the central processor. When a word of data is assembled in the receiver register, a flag is set. This flag is checked by a 6111 IOT instruction or by the DC02-F controller. The M707 module independently converts parallel data to serial form for transmission to an asynchronous serial data device. When a word is transmitted from the transmitter register, a flag is set. This flag is checked by a 6121 IOT instruction or by the DC02-F controller. The M623 buffer module buffers the receiver register onto the central processor data lines. Electrically speaking, the DC02-G is designed to supply transmitter and receiver keying currents that are intended for use with 20 ma DC-keyed devices.

FLOATING POINT PROCESSOR TYPE FPP-12

DEC's new floating point processor gives the PDP-8/E computer a dual processor capability. It also does calculations as much as 39 times faster than before, while maintaining seven-digit accuracy.

The unit (FPP-12) is designed for all types of floating point arithmetic. The computational speed of the PDP-8/E is dramatically increased because the floating point calculations are done by hardware rather than by software, which is usually the case. Typically, a three-word, 36-point floating point multiply took 1,100 microseconds when done by software, and 500 microseconds when done by software and an Extended Arithmetic Element. An FPP-12 equipped PDP-8/E can do the same calculation in 28 microseconds.

Adding the FPP-12 as a parallel processor decreases the time needed to run a specific program.

When a calculation has to be done, it is transfered from the central processor to the floating point processor, while the central processor continues with its program. Without the FPP-12, the calculation has to be done by the central processor unit, which interrupts the program until the calculation is done. Also, the FPP-12 simplifies programming by giving a programmer direct access to 32,768 words or core memory and by eliminating the paging steps usually required. Eliminating paging can also lead to further reductions in the time required to execute a program. These time-saving features, when combined with the time saved by using hardware to do floating point calculations, allow an FPP-12 equipped PDP-8/E to execute application programs as much as 100 times faster than they could be done by software alone.

Floating Point Number System

The term, floating point, implies a movable binary point in a similar manner to the movable decimal point in scientific notation. An exponent is used to keep track of the number of spaces the binary or decimal point is moved.

Examples of scientific notation:

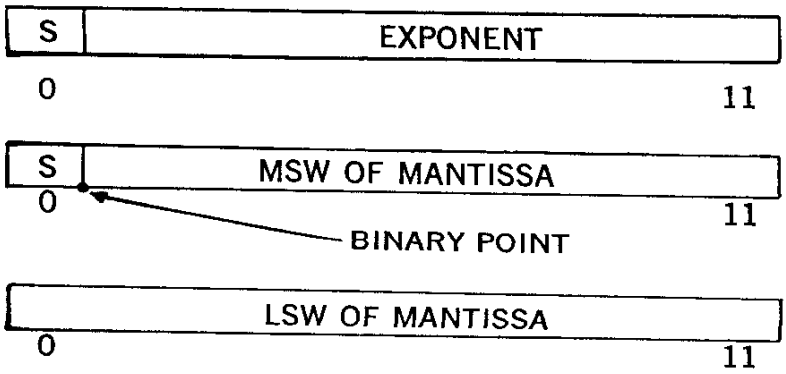
$$234 = 23.4 \times 10^1 = 2.34 \times 10^2$$

Examples of binary floating-point notation:

$$(1011) = (101.1) \times 2^1 = (10.11) \times 2^2 = (1.011) \times 2^3$$
$$= 0.1011 \times 2^4 = 0.01011 \times 2^5$$

Note that in all cases of binary floating-point notation given above, there are four significant bits. However, in the last example the mantissa which multiplies the exponent contains six bits. Given a fixed number of bits, it is desirable to adjust the exponent and the binary point to eliminate leading zeroes to retain the maximum significance for a given format length. The FPP12 normalizes or removes leading zeroes as the last step in every floating-point arithmetic operation.

The floating-point data format used by the FPP12 is identical to the format used by the PDP-8 floating-point system (DEC-08-YQYB-D). As shown below, there is a 12-bit signed 2's complement exponent and a 24-bit signed 2's complement mantissa.

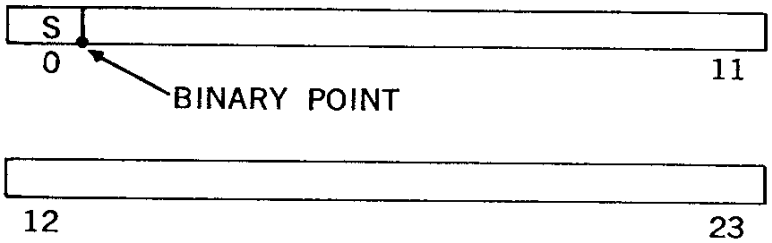


The FPP-12 carries all calculations to 28 bits of precision then rounds to 24 bits after normalization. After rounding, the results are rechecked for proper normalization prior to completion of the instruction.

In fixed point arithmetic, a calculation which results in a number whose magnitude cannot be expressed in 12 or 24 bits is an error. With the FPP-12, the number range is 2^{+2047} to 2^{-2048}. Exceeding the upper limit, 2^{+2047}, causes the FPP-12 to interrupt the CPU and set its exponent overflow status bits. A calculation resulting in an exponent smaller than 2^{-2048} is an exponent underflow which normally causes a program interrupt. The programmer has the option at initialization to request that the underflow trap be ignored, in which case, the result of a calculation in which underflow occurred is set to 0.

Double Precision

For those calculations where full 24-bit precision is not necessary and where core space is of a premium, the FPP-12 is used in fixed point double precision mode. Each operand consists of a 24-bit signed 2's complement fraction as shown below. As with the floating-point mode, each calculation is carried to 28 bits of precision and rounded to 24 bits. In this instance, normalization is not performed allowing the occurrence of leading zeroes which reduces the precision of subsequent calculations. The largest numbers that may be represented in double precision format are $+2^{23}-1$ and -2^{23}. Calculations producing numbers that exceed this range cause the floating point processor to initiate a program interrupt with the fraction overflow status bit set to a one.



Operation

The FPP-12 is initialized and interrogated as to its status through PDP-8/E IOT's. Once initialized, the FPP-12 operates much like a central processor fetching instructions and operands and storing results in memory. Data breaks are generally requested as needed. However, the usual number of breaks requested by the FPP-12 is two per instruction performed by the processor. This means that while the FPP-12 is "stealing cycles," programs can be run simultaneously at slightly reduced speed.

Active Parameter Table Format

Location

P	Field Bits of Operand Address	Field Bits of Base Reg.	Field Bits of Index Register Location	Field Bits of FPC
P+1	Lower 12 bits of FPC			
P+2	Lower 12 bits of index register location			
P+3	Lower 12 bits of Base Reg			
P+4	Lower 12 bits of operand address			
P+5	Exponent of FAC			
P+6	MSW of FAC			
P+7	LSW of FAC			
NOTE: APT address points to location P.				

It should be noted that once initialized the FPP-12 will execute programmed instructions until

1. an error condition occurs,
2. an exit instruction is reached,
3. an exit IOT is issued,
4. an I/O preset is issued by the PDP-CPU*,
5. the PDP-CPU encounters any type of halt.

Initialization

In order to execute the first instruction of any program the FPP-12 must have the following information:

1. The address of the first instruction (FPC)
2. The initial contents of the floating AC (FAC)
3. The core address of index register 0. (Index registers 1 through 8 are stored in the next 7 sequential 12 bit words.) (XO)
4. The base register which contains the core address of the first location in the data block. (The data block consists of 128 thirty-six bit words.)

To simplify initialization, the four parameters listed above are placed in core in an active parameter table (APT), shown above, by the CPU. Two initializing IOT's are then issued to the FPP-12. FPCOM (6553) loads a command register and the most significant 3 bits of the APT pointer. FPST (6555) loads the remaining 12 bits of the APT pointer and starts the floating-point processor. Whenever the floating-point processor performs an exit, the current values of the FPC, FAC, XO, base reg., and operand address are deposited in the APT to be used either for restarting the FPP-12 or for debugging.

IOT List

FPINT 6551 Skip on FPP "interrupt request" flag.

FPHLT 6554 Halt the processor at the end of the current instruction. Store active registers in core, set a status register bit, and the "interrupt request" flag.

*This operation while the FPP-12 is running might necessitate a program reload.

FPCOM	6553	If the FPP is not running and the FPP "interrupt request flag" has been reset, set the command register to the contents of the AC. The three least significant bits of the AC set the field bits of the "Active Parameter Table" address. If the FPP is running or the interrupt request flag is set, the instruction is ignored.
FPICL	6552	Clear the FPP "interrupt request" flag.
FPST	6555	If the FPP is not running and the FPP "interrupt request flag" is reset, set the location of the "Active Parameter Table" to the contents of the AC, initiate the FPP and skip the next instruction. If the FPP is not running or the FPP "interrupt request flag" has not been reset, the instruction is ignored.
FPRST	6556	Read the FPP status register into the AC.
FPIST	6557	Skip on FPP "interrupt request" flag. If the skip is granted, clear the flag and read the FPP status request into the AC.

CPU AC After Read Status Instruction

AC0	Double Precision Mode
AC1	Instruction Trap
AC2	C.P.U. Force Trap
AC3	Divide by Zero
AC4	Fraction Overflow (double precision mode only)
AC5	Exponent Overflow
AC6	Exponent Underflow
AC7	
AC8	Unused
AC9	
AC10	
AC11	Run

The following data are transferred to the FPP by issuing the FPCOM (load command register instruction 6553):

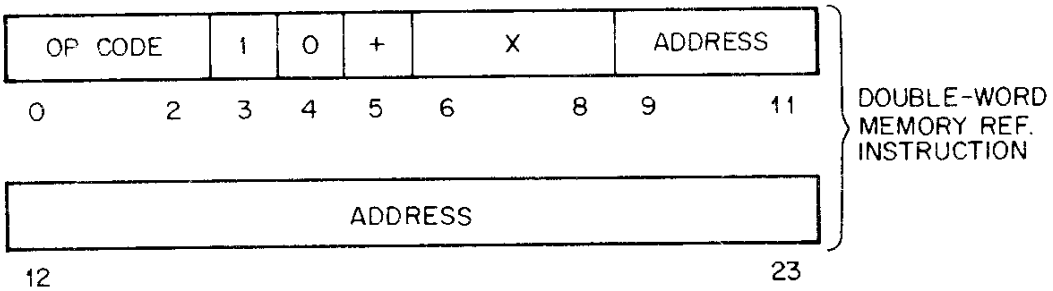
AC0	Select double precision mode
AC1	Exit of exponent underflow error
AC2	Enable memory protection
AC3	Enable interrupt
AC4	Do not store op address on exits
AC5	Do not store address of index registers on exits
AC6	Do not store address of indirect pointer list on exits
AC7	Do not store FAC of exists
AC8	Unused
AC9	
AC10	Data field of "Active Parameter Table"
AC11	

Methods for Memory Reference Instructions

The FPP-12 is capable of three modes of addressing for memory referencing instructions:

1. Double-word direct addressing
2. Single-word direct addressing
3. Single-word indirect addressing

A full indexing capability is available for both methods 1 and 3. The determined address for memory referencing instructions indicates the exponent in floating-point mode and generally directs to the most significant word in double precision mode. The format for double-word addressing is shown below:



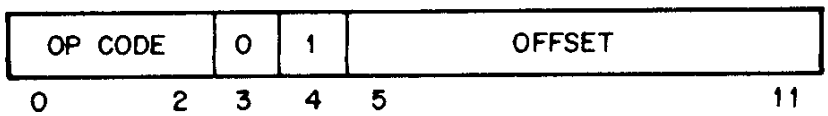
Example 1

If bit 4 is a 0, a double-word instruction is indicated. Setting bit 3 of double-word instruction to a 1 indicates a memory referencing instruction. A non-zero quantity in bits 6-8 causes the address contained in bits 9-23 to be modified by a specified index register. Setting bit 5 to a one causes the specified index register to be incremented prior to use in modification of the address. It should be noted that index register zero can be incremented and tested but is not used for address modification.

Single-Word Addressing Formats

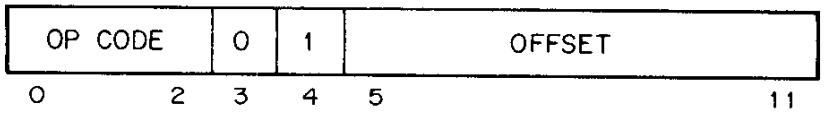
The two single-word address formats utilize a data block that is specified by a base address which is contained in the base register. The data block contains 128 3-word locations. In double-precision mode, the exponent of locations is ignored on the data block.

Single word formats are distinguished by bit 4 being a one. Bit 3 is the indirect indicator in a similar manner to PDP-8 code. The single-word direct address format example shown below the core address is equal to the sum of the 7-bit offset times 3 plus the contents of the base register.



Example 2

If bit 3 is a one, the following indirect format is specified:



Example 3

The effective address for Example 3 is given by the following equation:

$$\text{address} = C\,(\,(\text{offset} * 3) + \text{base address}) + \underbrace{[C\,(X+X0) + \text{bit } 5 * 1)]\ [2 \text{ or } 3]}_{\text{This term} = 0 \text{ if } X = 0}$$

Index Registers

Any core location may be used as an index register. Index register 0 is determined by the 15-bit X0 address. The X0 address is initially set from the active-parameter table, but may be altered by the MVX instruction. Index register X is in core location X0 + X where X = 0, . . . 7.

Accessing successive data points in floating-point mode requires incrementing the operand address by $(3)_8$ for each new data point. In double-precision mode, the proper increment is $(2)_8$ for each new data point. To account for the difference between the two modes, the selected index register is multiplied by 3 in floating-point mode or 2 in double-precision mode before it is used as an address modifier.

Instruction Set

OP CODE	MNEMONIC	MEMORY REFERENCE INSTRUCTIONS
0	FLDA	Load the FAC from the effective address.
1	FADD	Add the operand to the contents of FAC and store the result in the FAC.
5	FADDM	Add the operand to the contents of the FAC and store the results in the operand.
2	FSUB	Subtract the operand from the contents of the FAC and store the result in the FAC.
3	FDIV	Divide the operand into the contents of the FAC and store the results in the FAC.
4	FMUL	Multiply the contents of the FAC by the operand and store the result in the FAC.
7	FMULM	Multiply the contents of the FAC by the operand and store the results in memory.
6	FSTA	Replace the operand with the contents of the FAC.

Program Examples

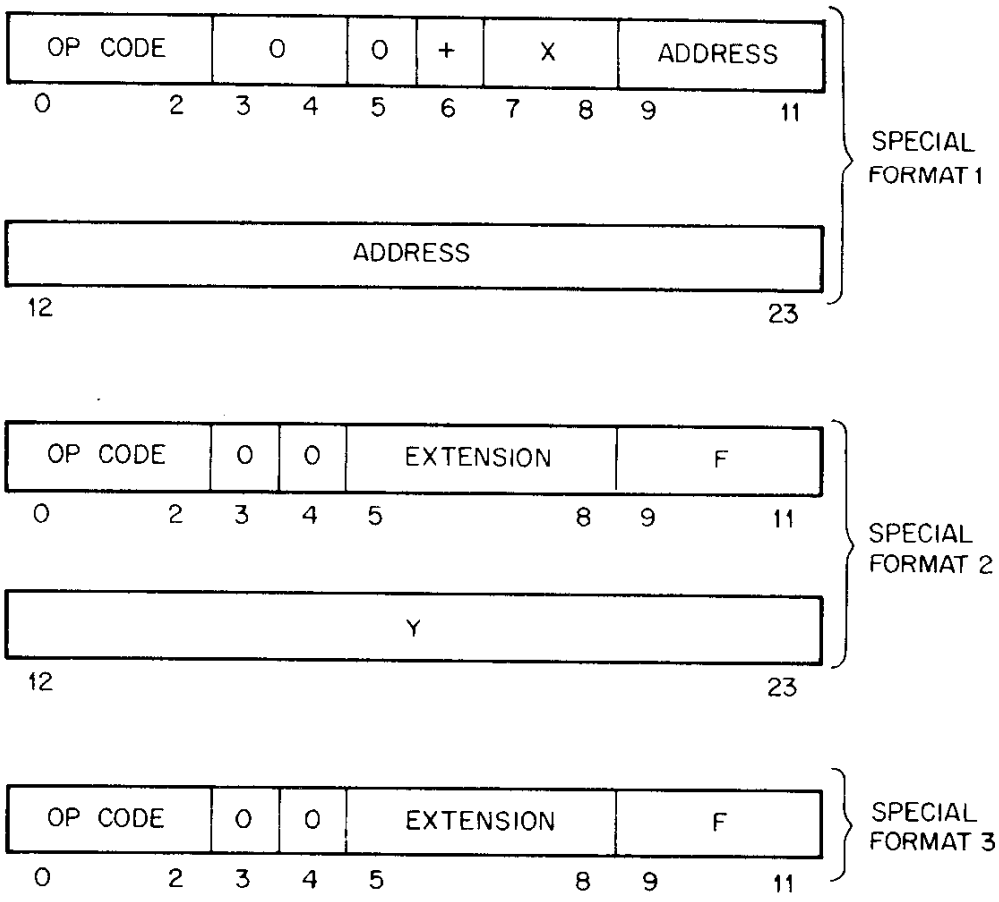
LOCATION	MNEMONIC	OCTAL CODE	
X	FSUB A	2401	/subtract 1 from the
		5432	/FAC
15432	A,	0001	
		2000	
		0000	
X	FSUB B	2205	/subtract 1 from the
			/FAC
Base Register + 5	B,	0001	
		2000	
		0000	
X	FSUB C, 2	2421	/subtract 1 from the
		5432	/FAC
15432 + 3	C',	0001	
(Index Reg 2)		2000	
		0000	
X	FSUB I D	2603	/subtract 1 from the
Base Register + 3	D,		/AC
		5412	
		0132	
		6724	
26724		0001	
		2000	
		0000	

Special Instructions

The FPP-12 special instructions are similar in nature to the nonmemory referencing instructions for the PDP-8. The set of special instructions includes conditional jumps, two types of subroutine calls, two types of unconditional jumps, several index register operations, a number of accumulator controls, two mode control instructions, and several operational instructions. Altogether, the special group has 26 defined instructions, five trapped instructions, and 14 undefined codes which do not perform any operation. Special instructions which may consist of 1 or 2 12-bit words are denoted by zeroes in bits 3 and 4 as shown below:

SPECIAL FORMAT 1		
OP CODE	MNEMONIC	
2	JXN	The index register X is incremented if bit 5 = 1 and a jump is executed to the address contained in bits 9-23 if index register X is nonzero.

The JNX instruction is similar to the following sequence of PDP-8 instructions.



		ISZ JMP TAG	
3 4 5 6 7	Trapped Instructions	The "instruction trap" status bit is set and the FPP-12 exits causing a PDP interrupt. The unindexed operand address is dumped into the active parameter table.	

SPECIAL FORMAT 2

OP CODE	EXTENSION	MNEMONIC	
0	10	FSTAX	The contents of the index register specified by bits 9-11 are replaced by the contents of bits 12-23.
0	11	ADDX	The contents of bits 12-23 are added to the index register specified by bits 9-11.
0	12-17	NOP	These codes are undefined single-word instructions and perform no operation.

Conditional Jumps—Jumps, if performed, are to the location specified by bits 9-23 of the instruction.

1	0	JEQ	Jump if the FAC = 0
1	1	JGE	Jump if the FAC 0
1	2	JLE	Jump if the FAC ⩽ 0
1	3	JA	Jump always
1	4	JNE	Jump if the FAC ± 0
1	5	JLT	Jump if the FAC < 0
1	6	JGT	Jump if the FAC > 0
1	7	JAL	Jump if impossible to fix the floating-point number contained in the FAC; i.e., if the exponent is greater than $(23)_{10}$.

POINTER MOVES

1	10	SETX	Set X0 the location of index register zero to the address contained in bits 9-23 of the instruction.
1	11	SETB	Set the base register to the address contained in bits 9-23.

SUBROUTINE CALLS

OP CODE	EXTENSION	MNEMONIC	
1	13	JSR	Jump and save return. The jump is to the location specified in bits 9-23 and the return is saved on the 1st location of the data block.

The JSR is used in writing re-entrant code as the return address is stored in the user's data block. A possible return from a re-entrant subroutine is via the two instruction sequences as follows:

```
        LDA 0       0200    /Load AL with contents
                            /of 1st location of the data
                            /block
        JAC         0007    /Jump to the location
                            /specified by the
                            /least significant 15 bits
                            /of the AC mantissa
                            /JAC is a special
                            /Format 3 instruction
```

1	12	JSA	An unconditional jump is deposited in the address and address + 1 where address is specified by bits 9-23. The FPC is set to address + 2.
1	14-17	NOP	These codes are single-word NOP's.

SPECIAL FORMAT 3 INSTRUCTIONS

OP CODE	EXTENSION	MNEMONIC	
1	1	ALN	The mantissa of the FAC is shifted until the FAC exponent equals the contents of the index register specified by bits 9-11. If bits 9-11 are zero, the FAC is aligned such that the exponent $= 23_{10}$.[1] In fixed-point mode an arithmetic shift is performed on the FAC fraction. The number of shifts is equal to the absolute value of the contents of the specified index register. If the contents of the index register is positive, shifting is towards the least significant bit; otherwise shifting is towards the most significant bit. In fixed-point mode the FAC exponent is not altered.

[1]Setting the exponent $= (23)_{10}$ intergerizes or fixes the floating point number. The JAL instruction tests to see if fixing is possible.

OP CODE	EXTENSION	MNEMONIC	
0	2	FLATX	The FAC is fixed and the least significant 12 bits of the mantissa are loaded into the index register specified by bits 9-11. In fixed-point mode the least significant 12 bits of the FAC is loaded into the specified index register. The FAC is not altered by the FLATX instruction.
0	3	FLDAX	The contents of the index register specified by bits 9-11 are loaded right justified into the FAC mantissa. The FAC exponent is loaded with $(23)_{10}$ and then the FAC is normalized. This operation is typically termed floating a 12-bit number. In fixed-point mode the FAC is not normalized.
0	4-7	NOP	These single-word instructions *perform no operation.*

OPERATE GROUP

OP CODE	EXTENSION	BITS 9-1	MNEMONIC	
0	0	0	FEXIT	Dump active registers into the active parameter table, reset the FPP-12 run flip flop to the 0 state, and interrupt the PDP-8 processor.
0	0	1	FPAUSE	Wait for external synchronizing signal. This instruction is designed to cooperate with the AIP-12 option.
0	0	2	FCLA	Zero the FAC mantissa and exponent.
0	0	3	FNEG	Form the two's complement of the FAC mantissa.
0	0	4	FNORM	Normalize the FAC. In fixed-point mode FNORM is a NOP.

OP CODE	EXTENSION	BITS 9-1	MNEMONIC	
0	0	5	START F	Start floating-point mode.
0	0	6	START D	Start double-precision mode.
0	0	7	JAC	Jump to the location specified by the least significant 15 bits of the FAC mantissa.

RT01 DEC-link® Data Entry Terminal

RTO1 DEC-link® Data Entry Terminal

DEC-link is a low-cost, self-contained data entry device which is remotely locatable. It features teletype and EIA serial line compatibility.

DEC-link offers 16 unique characters which a monitoring computer may use for either numeric data or control functions. It can display up to 12 digits of decimal data (plus decimal point) as well as status indicators.

Data is entered via an integral 16 character keyboard; numeric data is displayed on "Nixie"* tubes.

The status indicators are used to indicate non-numeric information such as "repeat transmission," "computer ready," etc. Four programmable status indicators are standard on DEC-link.

Interface to a computer is easily accomplished via any fully duplex, 4-wire data communications teletype interface.

Modem interface signals, corresponding to EIA RS-232C specifications, are also provided.

APPLICATIONS

DEC-link provides easy and economical access to numeric information in a computer. It lends itself to such applications as:

Stockroom Inventory Control
Data Logging
Information Retrieval
Production Line Monitoring
Quality Control Monitoring
Work Flow Monitoring
Security Systems
Machine Efficiency Reporting
Management Information Systems

DEC-link fills the gap in price, performance, and usage between full-scale, video displays and electro-mechanical, hard-copy devices.

SPECIFICATIONS

General

Line Votage: 115 VAC, 230 VAC 47-62 Hz.
Power: 30W
Size: 15" W x 12" D x 6" H
Weight: 12 lbs.
Aux. Switches: on-off

Display Options

Lamps: 4 Status Indicators (programmed control)
Digits: 4, 8, or 12 Nixie tubes
Decimal Point: Programmable over 12 digits

Control Functions

Clear Display: Code $(100)_8$
Load Status Indicators: Code $(129)_8$ to $(137)_8$ "P" through "←"

Data Input

Input Levels:

20 MA TTY Isolated Current Loop

EIA RS232C

Receive Rate: 110 or 300 Baud

Character Format:

8 level asychronous serial ASCII

1 or 2 stop bits

Data Output

Output Levels: Isolated Transistor switch capable of passing 20MA

EIA RS232 Levels:

Data Terminal Ready

Transmitted Data

Received Data

Protective Ground

Signal Ground

Transmission Rate: 110 or 300 Baud

Character Format: 8 level asynchronous serial ASCII

Character Rate:

10 Characters/Second (110 Baud)

30 Characters/Second (300 Baud)

Output Connectors:

4 lug Jones Strip (TTY)

Cinch DB 25P (EIA)

Character Set

Number of Characters: 16

Code: ASCII 8 Level

Character Codes:

ASCII 0 through 9

A through F

PROGRAMMING

The RT01 DEC-link Data Entry Terminal utilizes the standard Teletype control interface. Therefore, the same instructions used to program the Teletype are also used to program this Data Entry Terminal.

DW08-A I/O conversion panel

Digital's DW08-A Conversion panel enables any PDP-8/E computer to economically communicate with I/O devices of opposite logic levels. The DW08 contains its own integral power supply and takes up only 5-1/4 inches in height in a standard 19 inch rack.

The DW08 Positive-to-Negative Bus Converter accepts the positive I/O bus of a PDP-8/E and KA8/E, KD8/E option. Outputs consist of a Negative Bus, as well as a continuation of the Positive Bus. Positive Bus signal levels are defined (see figure 7-32) as high (+3 volts) and low (0 volts); Negative Bus signals are defined as high (0 volts) and low (—3 volts). The name bus denotes a combination of input (received by the computer) and output (sent by the computer) signals.

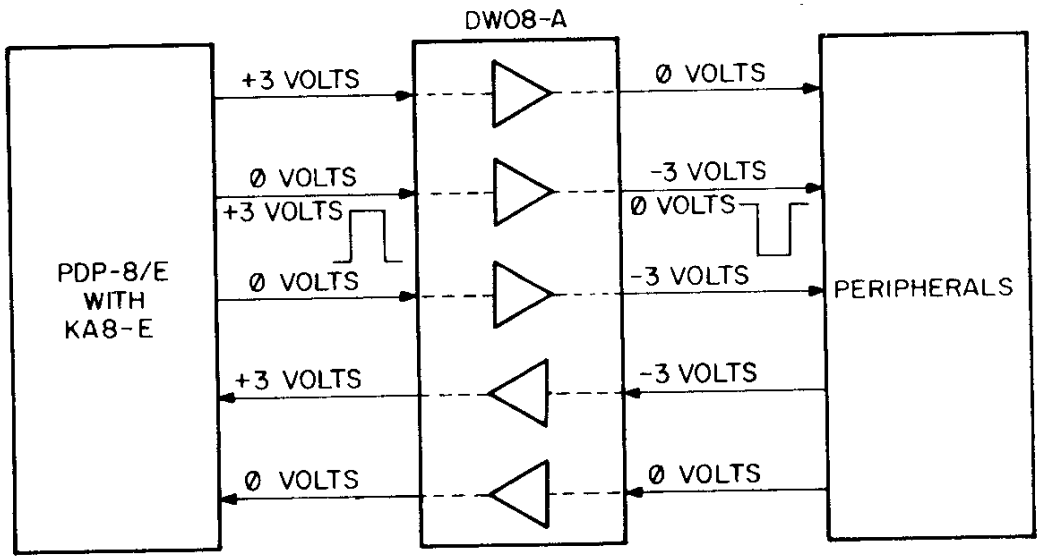


Figure 7-32 DW08-A conversion levels

The Positive Bus (with TTL logic levels of 0 volts and +3 volts) inputs on five M904 cable connectors at locations A01 to A05. The continued positive bus outputs are obtained on five M903 or M904 cable connections at locations B01 to B05. Input level conversion produces a high level out (0 volts) for a high level in (+3 volts); and a low level out (—3 volts) for a low level in (0 volts). Certain timing signals used on PDP-8/E computers (BIOP1,BIOP2,BIOP4,BTS3 (1), BTS 1 (1), and B INITIALIZE are clamped and inverted before level conversion.

The Negative Bus inputs and outputs are obtained on eleven WO11 or WO31 cable connections at location A13 to A23. Input signals to the computer on the Negative Bus are level converted to produce a low level out (0 volts) on the Positive Bus for a high level in (0 volts) and a high level out (+3 volts) for a low level in (—3 volts).

Cable Lengths

Delays occur within the DW08A unit due to level conversion, which effects the maximum length of the I/O bus of the PDP-8/E. The effective I/O

cable length consumed by the DW08A is 10 feet, which must be subtracted from the maximum permissible bus length from the PDP-8/E to the farthest device on the converted bus.

Specifications

Dimensions:	5 1/4" high for 19" mounting. A cabinet depth of 15" is required because of the power supply.
AC Input:	120 VAC or 240 VAC ±10%, 50 Hz to 60 Hz
Power Consumption:	115 watts
Heat Dissipation:	300 BTU/hour
Weight:	20 lbs.

part 3

INTERFACING AND INSTALLATION

CHAPTER 8
DIGITAL LOGIC CIRCUITS

INTRODUCTION

The digital logic circuits in this chapter are used to interface I/O devices to the computer using Digital Equipment Corporation FLIP CHIP Modules. Logic handbooks published by DEC describe hundreds of FLIP CHIP Modules with their component circuits, associated accessories, hardware, power supplies, and mounting panels. The designer should study the logic handbooks carefully before beginning on interface design for a special I/O device.

The basic logic circuits used for interfacing to the computer are: AND, OR, NAND, NOR, Flip-Flop, Single-Shot, Schmitt Trigger, Inverter, Amplifier, and Bus Driver. A brief discussion of these circuits and their logic symbology follows.

The symbology employed with the PDP-8 family of computers and M-series modules is similar to MIL-STD-806B. This chapter describes DEC symbology with definitions of logic functions, graphic representations of the functions, and examples of their application. A Table of Combinations is also shown.

LOGIC SYMBOLS

The following description of logic symbols contains truth tables that show graph representations of the logic functions. In the truth tables, the letter H stands for HIGH (+ 3V), and the letter L stands for LOW (0V). Examples of DEC symbology are shown along with figures and truth tables.

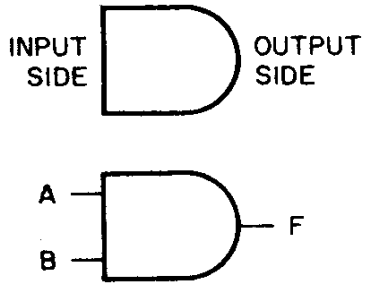
State Indicator

The presence of the small circular symbol at the input(s) of a function indicates that an L input signal activates the function. The absence of this small circle indicates that an H input signal activates the function. Similarly, a small circle at the output of a function indicates that the output terminal of the activated function is relatively low, and the absence of the circle indicates that the output is relatively high.

STATE INDICATOR ABSENT

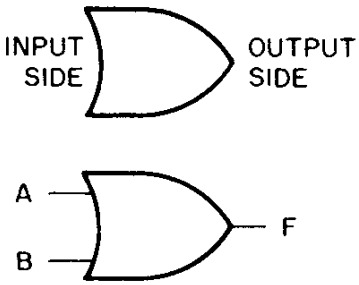
a. AND—The symbol in Figure 8-1 represents the AND function. The output (F) is high only if both inputs (A and B) are high.

b. OR—The symbol in Figure 8-2 represents the OR function. The OR output (F) is high if any input (A or B) is high.



INPUT		OUTPUT
A	B	F
L	L	L
L	H	L
H	L	L
H	H	H

Figure 8-1 Symbol, AND Function



INPUT		OUTPUT
A	B	F
L	L	L
L	H	H
H	L	H
H	H	H

Figure 8-2 Symbol, OR Function

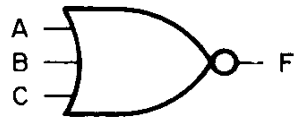
STATE INDICATOR PRESENT

a. NAND—The symbol in Figure 8-3 represents the NAND function. The output (F) is low only when all inputs (A, B, and C) are high. NAND logic is the major gate configuration in the PDP-8 family.

INPUT			OUTPUT
A	B	C	F
L	L	L	H
L	L	H	H
L	H	L	H
L	H	H	H
H	L	L	H
H	L	H	H
H	H	L	H
H	H	H	L

Figure 8-3 Symbol, NAND Function

b. NOR—The symbol in Figure 8-4 represents one version of the NOR function. The output (F) is low if one or more of the inputs (A, B, and C) are high.

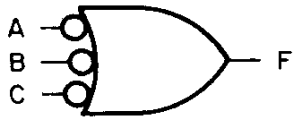


INPUT			OUTPUT
A	B	C	F
L	L	L	H
L	L	H	L
L	H	L	L
L	H	H	L
H	L	L	L
H	L	H	L
H	H	L	L
H	H	H	L

Figure 8-4 Symbol, NOR Function

c. NOR—The symbol in Figure 8-5 represents another version of the NOR function. The output (F) is high if one or more of the inputs

(A, B, or C) are low. The NOR version for this function is identical to one version of the NAND function shown in Figure 8-3.



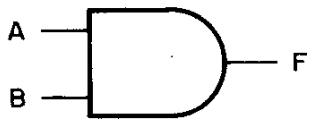
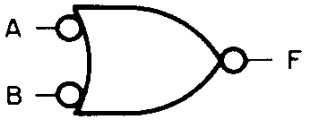
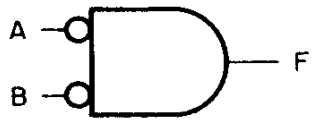
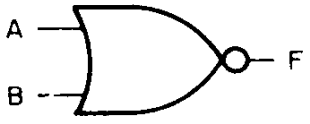
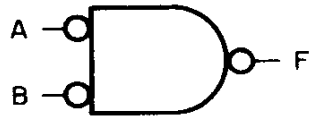
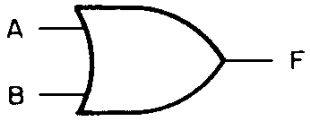
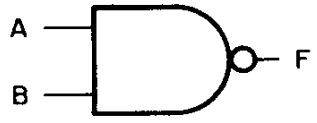
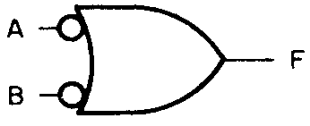
INPUT			OUTPUT
A	B	C	F
L	L	L	H
L	L	H	H
L	H	L	H
L	H	H	H
H	L	L	H
H	L	H	H
H	H	L	H
H	H	H	L

Figure 8-5 Symbol, NOR Function

Table of Combinations

Table 8-1 illustrates the applications, functions, and truth tables of two variables and their equivalents, as well as their relationship to DEC logic.

Table 8-1 Table of Combinations

AND	OR	A	B	F
		H	H	H
		H	L	L
		L	H	L
		L	L	L
		H	H	L
		H	L	L
		L	H	L
		L	L	H
		H	H	H
		H	L	H
		L	H	H
		L	L	L
		H	H	L
		H	L	H
		L	H	H
		L	L	H

The symbol in Figure 8-6 shows the flip-flop function. The pins are numbered counterclockwise on a standard flip-flop. The flip-flop has four possible inputs: set (S), reset(R), data(D), and clock (C); and two data outputs, logic 0 (low) and logic 1 (high). If the data input is high and a pulse is applied to the clock input, the flip-flop sets to the logic 1 state. If the data input is low and a pulse is applied to the clock input, the

flip-flop resets to the logic 0 state. When the data input is shown with a small circle (redefined flip-flop), the opposite of the above is true; that is, if the data input is high and a pulse is applied to the clock input, the flip-flop goes to its logic 0 or reset state, etc.

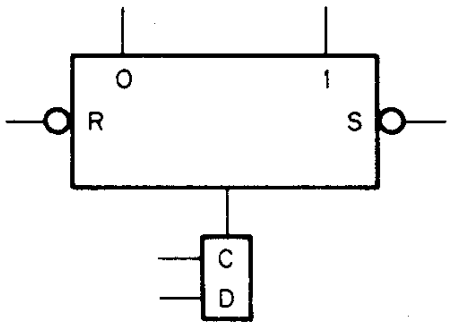


Figure 8-6 Symbol, Flip-Flop Function

One-Shot Function

The symbol in Figure 8-7 shows the one-shot (OS) function. The output signal shape, amplitude, duration, and polarity are determined by the circuit characteristics of the OS device. When it is not activated, the one-shot device is in either a 0 or 1 state. When the input is pulsed by a high-to-low level change, the 1 output goes high and remains high; and the 0 output goes low and remains low for the specific time of the device.

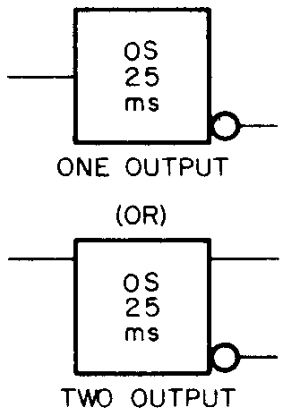


Figure 8-7 Symbol, One-Shot Function

Schmitt Trigger

The symbol in Figure 8-8 shows the Schmitt trigger (ST) function. The Schmitt trigger is normally either in a 0 or a 1 state (inactivated). When the input signal crosses a predetermined voltage threshold, the Schmitt

trigger changes to its opposite state and remains there until the input signal falls below the threshold.

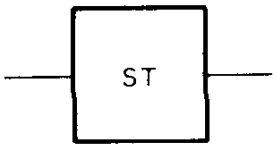


Figure 8-8 Symbol, Schmitt Trigger Function

General Logic Symbols

The symbol in Figure 8-9 is used for functions not specified elsewhere. An example of this symbology is the inhibit driver used in the PDP-8 family. When possible, an explanation or abbriviation of the function is contained within the box.

Figure 8-9 Symbol, General Logic

Amplifier

The symbol in Figure 8-10 shows a linear or nonlinear current or voltage amplifier. This symbol is used to represent level changers, inverters, emitter followers, and lamp drivers.

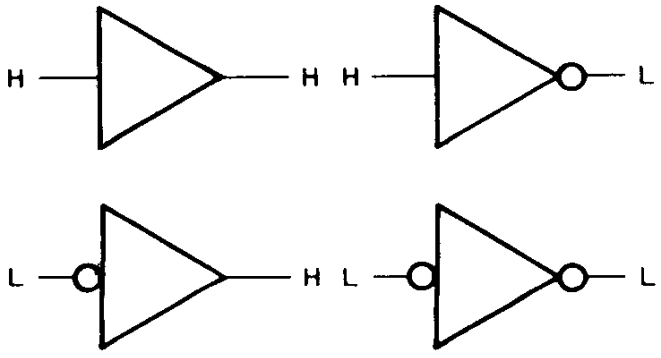


Figure 8-10 Symbol, Amplifier

Time Delay

The symbol in Figure 8-11 shows a time-delay device. The time-delay duration is specified inside the symbol unless there are two or more outputs. When there are two or more outputs, the delay time of each output is marked adjacent to that output.

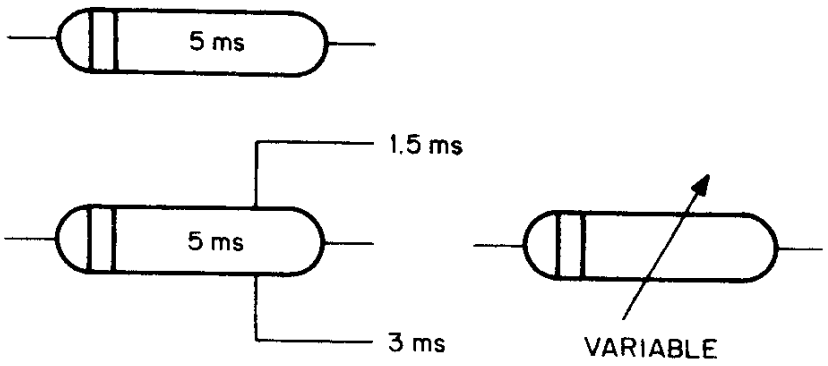


Figure 8-11 Symbol, Time Delay Function

CHAPTER 9
THE OMNIBUS INTERFACE

INTRODUCTION

Interfacing to the PDP-8/E OMNIBUS is accomplished with both hardware and software. For standard interfaces, DEC supplies necessary option modules and the equivalent programs (subroutines and MAINDECs) necessary to perform those DEC defined functions. This software is quite adequate to satisfy the operational requirements of each standard option as defined in Chapter 7. However, in the event that the user desires program functions different from those given in the software packages, he must change the software using the standard input/output devices such as a teleprinter, a card reader, or the console switches. Before undertaking this, however, the user should acquaint himself with the software routines (good documentation of the software is provided with each program tape) and read INTRODUCTION TO PROGRAMMING, a volume in the DIGITAL small computer handbook series.

This chapter provides the necessary information for users desiring to build a special interface. It deals primarily with the hardware consideration and it is understood that the user must create his own program for his own defined functions. If the user lacks sufficient experience to design his interface, he can contact his local DEC Sales Office for special assistance.

The means of transferring commands and signals from module to module is accomplished on what is called the "OMNIBUS." All PDP-8/E modules, including options, plug into the OMNIBUS in a significantly accessible manner.

The OMNIBUS is an etched board with rows of connectors soldered to the board. The pin assignment is the same on all connectors. Thus, the OMNIBUS accommodates 96 signals, which feed to 96 pins on the connectors. The user is generally only concerned with those signals that control data transfers, address memory, or contain the data to be transferred. However, the additional signals, such as timing, are readily available on the OMNIBUS to accommodate any tailor-made requirement in the event that the user should design and build his own interface module.

Many advantages are derived from the OMNIBUS approach. Because all connectors on the OMNIBUS contain the same signals, a module can be placed anywhere on the bus at the convenience of the user. All random wiring is eliminated with this type of arrangement. This feature provides greater performance, and reliability. Considerable space is conserved; thus providing a unique packaging capability that allows a high density of electronic circuitry in a small area.

The information in chapter 9 is presented in two sections. The first section describes the many aspects of the OMNIBUS in terms of its physical qualities, the types of interfaces placed on the bus, and the input/output transfer schemes. This information is intended to provide the reader with sufficient background to begin designing an interface control module.

When interfacing to the PDP-8/E, the designer may consider the OMNIBUS as his interface. If he follows the rules specified in section 1 of this chapter, he is more than half way toward designing his own interface. The nature of the OMNIBUS and all 96 signals are defined in a manner that makes interfacing relatively easy to accomplish.

Section 2 identifies the Data Transfer types and some guidelines to help the designer choose the transfer techniques for his needs; section 3 provides a general guideline for the designer building a Programmed I/O Interface Control Module; section 4 provides a general guideline for the designer building either a single-cycle or a three-cycle Data Break Interface; section 5 provides general design and construction guidelines. Section 6 includes some PDP-8/E interface hardware.

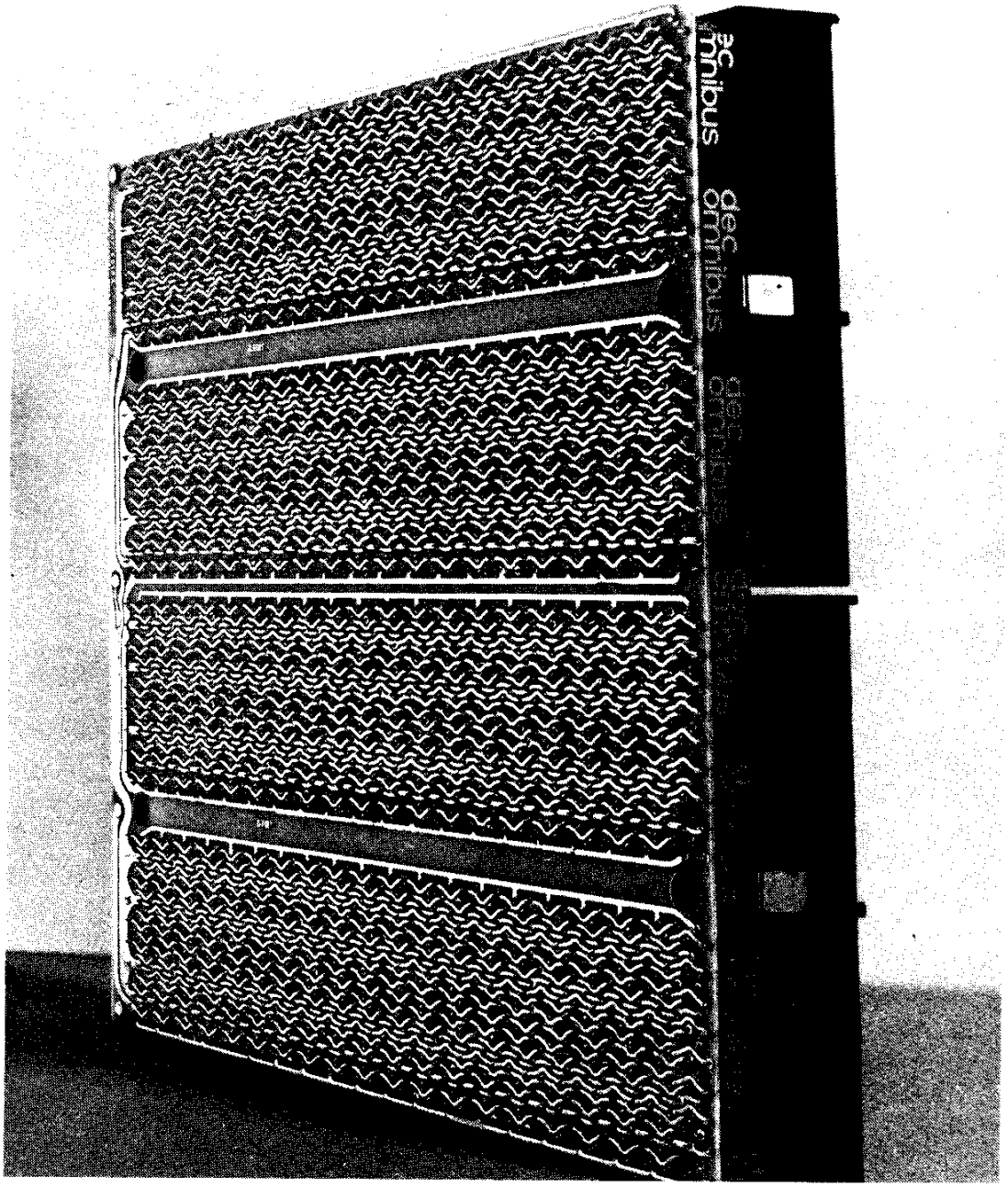


Figure 9-1 PDP-8/E OMNIBUS

SECTION 1 OMNIBUS DESCRIPTION

BUS STRUCTURE

The OMNIBUS (H919 OMNIBUS Assembly) is a back plane etched circuit board with ten H803 connectors mounted onto the board and wave soldered. The OMNIBUS is 10½ inches by 10½ inches with a 1⅝ inch thickness. The OMNIBUS is attached to the bottom of the PDP-8/E mounting box and is the means by which all modules are connected. Figure 9-1 shows the OMNIBUS with all connectors mounted. A single assembly accommodates 20 PDP-8/E modules.

The OMNIBUS is designed so that all back plane wiring is eliminated and so that every pin in a given connector slot is defined. All modules plugging into the bus are PDP-8/E modules. If a functional unit on the bus requires more than one module, Type H851 edge type connectors on the top of the board connect multiple boards together. For cables to the "outside world," connectors on the side of the module connect to a shielded coaxial or flat ribbon cable. In this arrangement, up to 2 connectors for each module may be used.

Figure 9-2 shows the OMNIBUS with modules plugged into it. Each module functional unit can be placed anywhere on the bus or removed from the bus without affecting the operation or performance of the rest of the system not requiring that module.

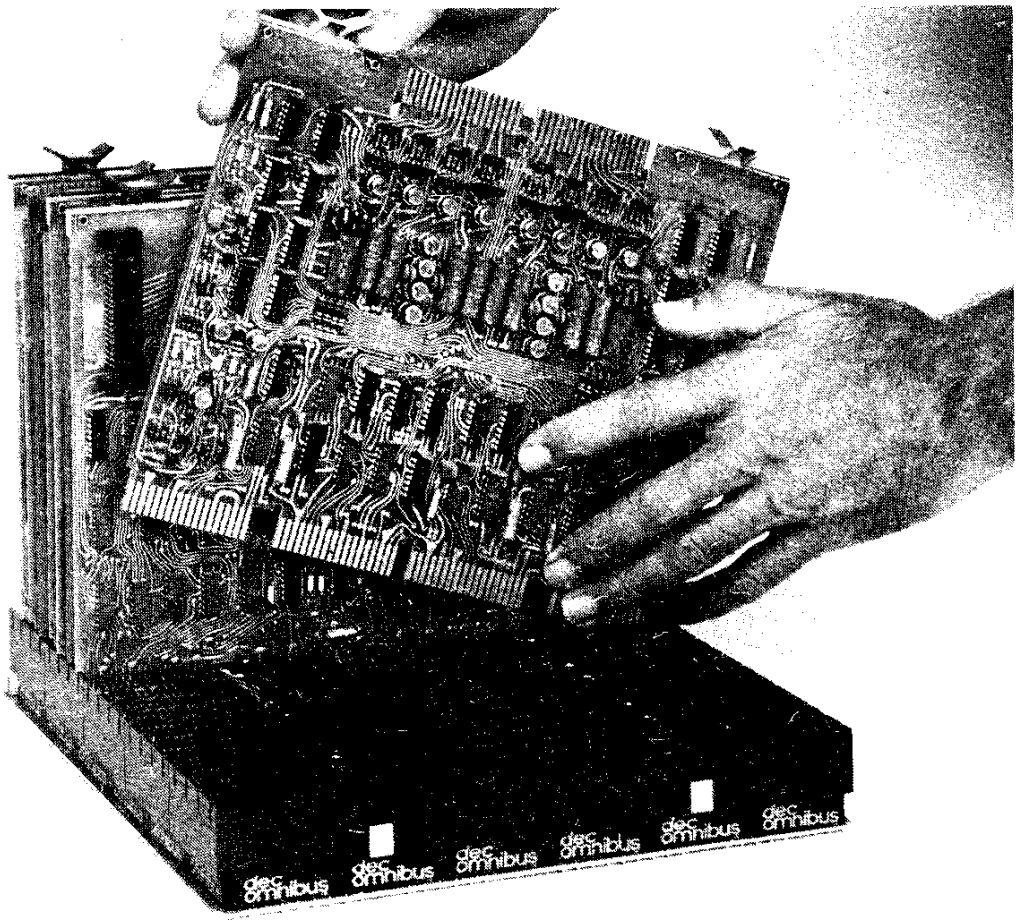


Figure 9-2 PDP-8/E Modules mounted on the OMNIBUS

Each OMNIBUS contains 20 slots; two assemblies with a M935 bus connector provides 40 slots, two of which are used to interconnect the assemblies. Thus, 38 slots are available when the OMNIBUS Expansion unit is used. However, the OMNIBUS can be expanded to accommodate an additional 37 modules. This is illustrated in Figure 9-3, which shows the basic OMNIBUS connected to the OMNIBUS expansion unit.

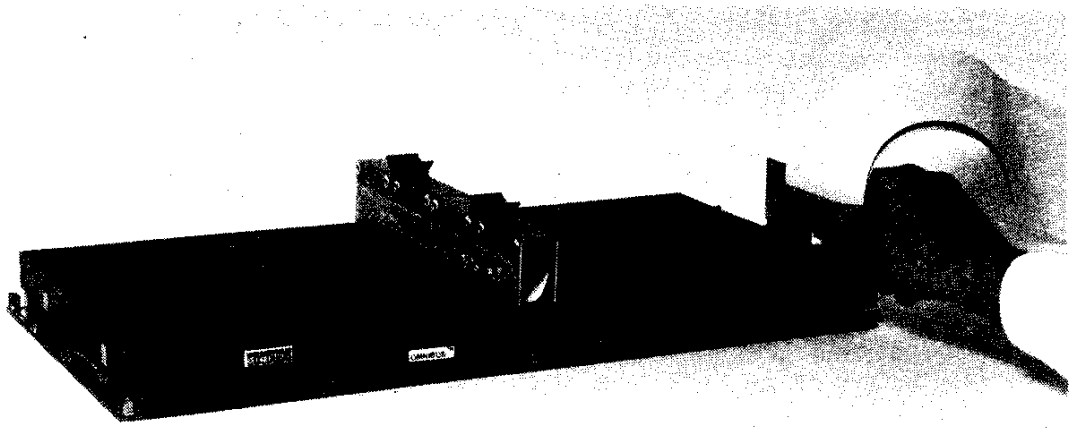


Figure 9-3 The System Expansion Capability.

BUS SPECIFICATIONS

Logic Levels
Logic 1—Max Voltage: 0.4V
Min Voltage: -0.5V
Logic 0—Max Voltage: 5.0V
Min Voltage: 3.0V

SYSTEMS CONFIGURATION

The PDP-8/E with all primary options is identified in Figure 9-4. The basic system contains the central processor (4 modules) the programmer's console (1 module), 4K memory (3 modules), a shield (1 module) and a console Teletype control (1 module).

System expansion is easily accomplished simply by adding "off-the-shelf" control units necessary to accommodate the corresponding peripheral equipment, if additional machine capability is desired. For example, if it were desired to add additional memory capability to a basic PDP-8/E, a Memory Extension Control and Time Share Option, type KM8-E, could be added. Then 4K memory units could be added, up to *a maximum 32K capability.* For those customers who wish to use PDP-8/I or PDP-8/L compatible peripherals, an external bus option such as the Positive I/O Bus Interface Module, type KA8-E, and the Data Break Interface Module, type KD8-E, connects to the OMNIBUS to provide interfacing capabilities.

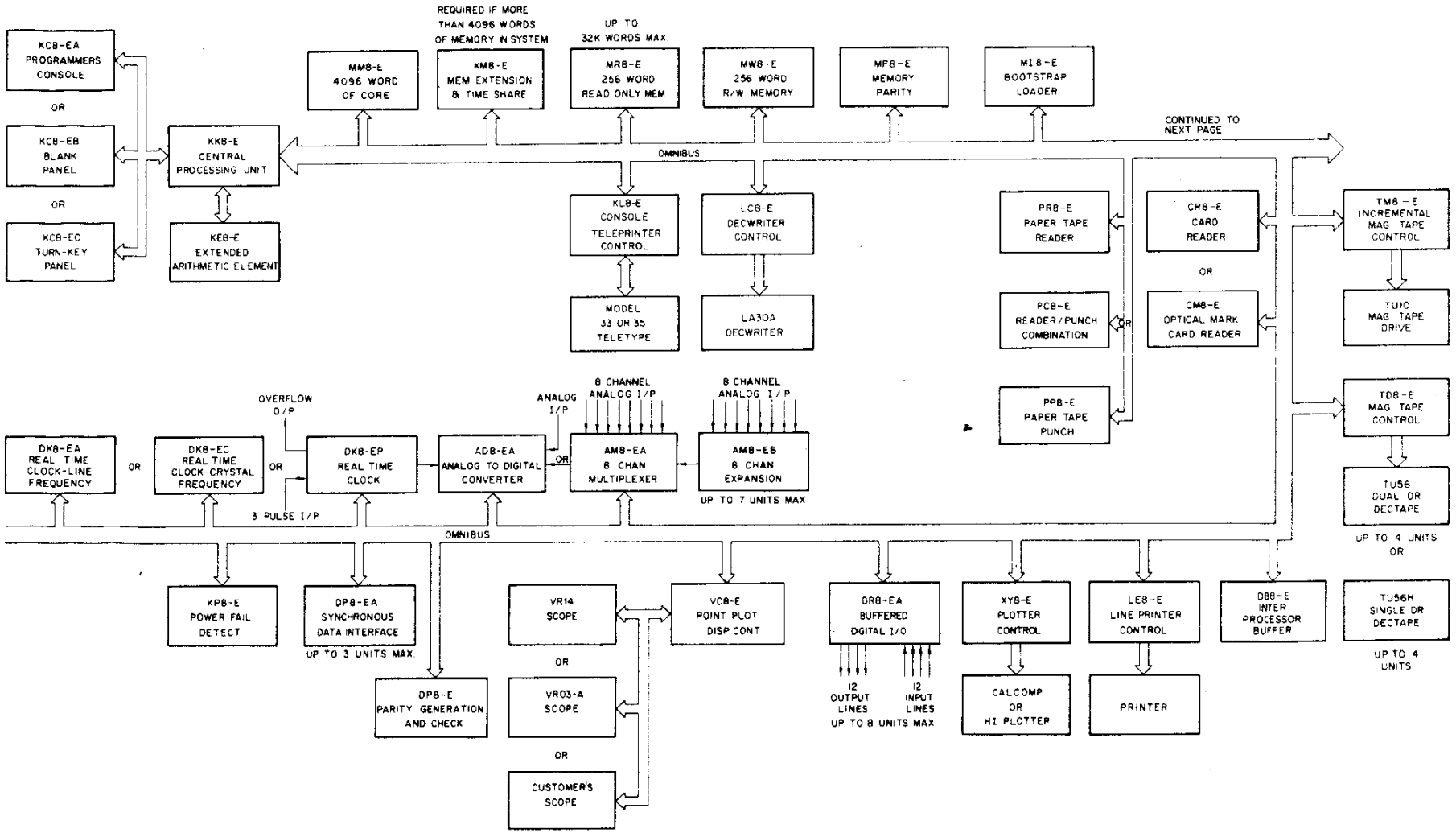


Figure 9-4 PDP-8/E—OMNIBUS Configurator

Almost all types of peripherals are included as an option to allow the user to expand his system to his own requirements. However, in the event that the user has a unique requirement such as a special control system, he may build his own control module by following the rules specified in this chapter. Refer to Chapter 11 for planning and installation.

RELATIONSHIP OF THE EXTERNAL BUS TO THE OMNIBUS

The External Bus, which is mechanically and electrically organized the same as the I/O bus on the PDP-8/L or the PDP-8/I with KA8-I, plugs into the OMNIBUS by way of the Positive I/O Bus interface and the Data Break interface. Each of these modules receives the same signal on the same pins as any other module plugged into the OMNIBUS. The interfacing details to the External Bus are given in Chapter 10 of this handbook.

OMNIBUS SIGNALS

The signals and pin assignments of the OMNIBUS are given in Figure 9-5. The L and H after the signal name identifies the most common assertion level. Bus Loads are provided in Figure 9-6. Each load corresponds to a description of each signal that is provided in Tables 9-1 through 9-4. The tables also identify the specific circuit by type 1 through 10 under the column heading "TYPE LOAD", "TYPE DRIVER". The corresponding circuit type is illustrated in Figure 9-6. The loading rules presented later in this chapter provide information on the electrical properties of these lines.

PIN	D1	D2	C1	C2	B1	B2	A1	A2
A	TP	+15V	TP	+5V	TP	+5V	TP	+5V
B	TP	- 15V	TP	- 15	TP	- 15V	TP	-15V
C	GND	GND	GND	GND	GND	GND	SP GND *	GND
D	MA8L	IRØ L	I/O PAUSE L	TP1 H	MA4 L	INT STROBE H	MAØ L	EMAØ L
E	MA9 L	IR1L	CØL	TP2 H	MA5 L	BRK IN PROG L	MA1 L	EMA 1L
F	GND	GND	GND	GND	GND	GND	GND	GND
H	MA10 L	IR2 L	C1 L	TP3 H	MA6 L	MA,MS LOAD CONT L	MA2 L	EMA2 L
J	MA11 L	F L	C2 L	TP4 H	MA7 L	OVERFLOW L	MA3 L	MEM START L
K	MD8 L	D L	BUS STROBE L	TS1 L	MD4 L	BREAK DATA CONT L	MDØ L	MD DIR L
L	MD9 L	E L	INTERNAL I/O L	TS2 L	MD5 L	BREAK CYCLE L	MD1 L	SOURCE H
M	MD10 L	USER MODE H	NOT LAST XFER L	TS3 L	MD6 L	LA ENABLE L	MD2 L	STROBE H
N	GND	GND	GND	GND	GND	GND	GND	GND
P	MD11 L	F SET L	INT ROST L	TS4 L	MD7 L	INT IN PROG H	MD3 L	INHIBIT H
R	DATA 8 L	PULSE LA H	INITIALIZE H	LINK DATA L	DATA 4 L	RES 1 H	DATA Ø L	RETURN H
S	DATA 9 L	STOP L	SKIP L	LINK LOAD L	DATA 5 L	RES 2 H	DATA 1 L	WRITE H
T	GND	GND	GND	GND	GND	GND	GND	GND
U	DATA 10 L	KEY CONTROL L	CPMA DISABLE L	IND 1 L	DATA 6 L	RUN L	DATA 2 L	ROM ADDRESS L
V	DATA 11 L	SW	MS,IR DISABLE L	IND2 L	DATA 7L	POWER OK H	DATA 3 L	LINK L

* THIS PIN IS CONNECTED TO GROUND ON THE BUS, BUT SERVES AS A LOGIC SIGNAL WITHIN MODULES TO FACILITATE TESTING

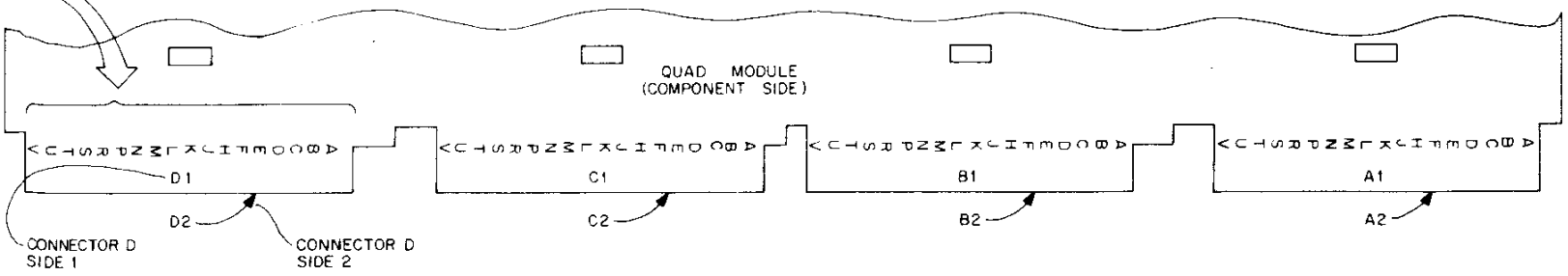
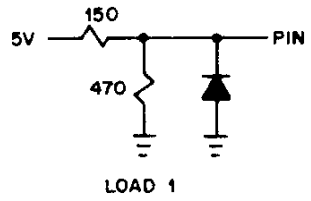
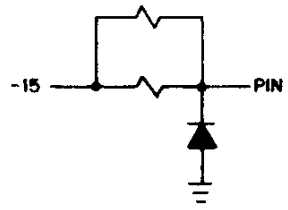


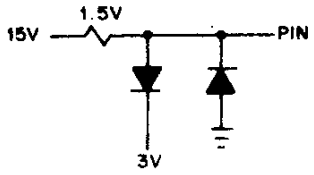
Figure 9-5 OMNIBUS Pin Assignment



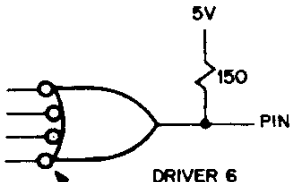
LOAD 1



LOAD .5

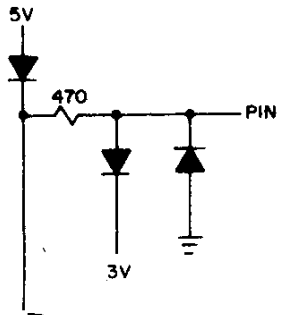


LOAD 2



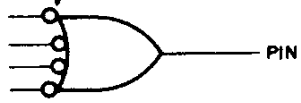
DRIVER 6

TTL GATE, USUALLY SN744D
NO ADDITIONAL DRIVERS
ALLOWED ON THE SAME LINE

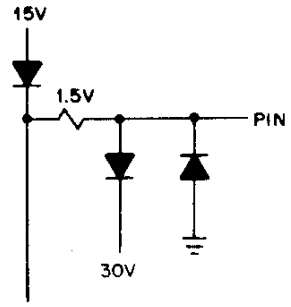


COMMON LINE WHICH IS
+1.5V DURING TP4
(DURING TP2 RESPECTIVELY)
AND OPEN CIRCUITED
OTHERWISE

LOAD 3

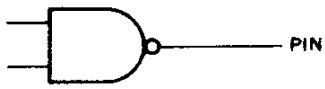


DRIVER



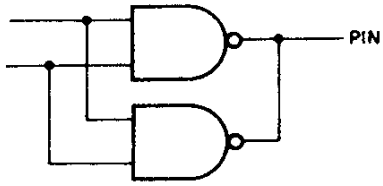
COMMON LINE WHICH IS
+30V DURING "INT STROBE"
AND OPEN CIRCUITED
OTHERWISE

LOAD 4



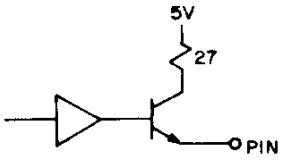
OPEN COLLECTOR
DRIVER USUALLY 8881

DRIVER 8



TWO OPEN COLLECTOR DRIVER
IN PARALLEL ALWAYS 8881

DRIVER 9



DRIVER 10

Figure 9-6 OMNIBUS Loads

Table 9-1 Programmed I/O OMNIBUS Signals

SIGNAL	ORIGIN	FUNCTION	.TYPE LOAD	TYPE DRIVER
MD0-11	Memory	Provides IOT instruction as follows: 6_8 (used by processor) Device operation code 0 1 2 3 4 5 6 7 8 9 10 11 Device Select Code LOGIC STATES: Ground = 1 +3V = 0	2	8
I/O PAUSE L	Processor	Used to gate the device select and device operation codes into the programmed I/O interface decoders and generate BUS STROBE at TP3 and NOT LAST XFER H. I/O PAUSE is grounded when MD0-2 equals 6 (octal) during FETCH and not USER MODE. PAUSE begins 150 ns after the start of TP1 and continues until 150 ns after the start of TP3 if INT STROBE is high.	1	6
TP 3 H	Processor	TP3H is used to clear the flag and clock the output buffer of a Programmed I/O interface. It is generated in the timing generator as a positive-going 100 ns pulse. (See timing pulses in Table 9-1c)	1	6
INTERNAL I/O L	Interface	Signal INTERNAL I/O is grounded by the device selector decoder. The Positive I/O Bus Interface cannot generate IOP's when this line is grounded. This inhibits decoding any Internal OMNIBUS IOT instructions. Failure to ground this line will result in long IOT timing.	2	8
DATA0-11	Processor and Interface	The 12 DATA lines called DATA BUS serves as a bidirectional bus for both input and output data, between the AC register in the processor and the interface buffer register. The proces-	4	8

Table 9-1 Programmed I/O OMNIBUS Signals (Continued)

SIGNAL	ORIGIN	FUNCTION	TYPE LOAD	TYPE DRIVER
		sor internal gating and loading is controlled by C0, C1, and C2 signals. During TS3 of an IOT instruction, the contents of the DATA BUS is applied to the processor's major register gating in accordance with the C lines. For output transfers, information must be taken from the DATA BUS by edge triggering only, using the leading edge of TP3.		
C lines C0, C1, C2	Interface	Signals C0, C1, C2 determine the type of transfer between a device and the processor. These lines control the data path within the processor and determine if data is to be placed onto the DATA BUS or received from the DATA BUS. They are also used to develop the necessary load control signals required to load either the AC register or the PC register. When it is time for a device to make either an input or output transfer, the device will ground the appropriate combination of C control lines so that Major Register gating and Register loading is made possible. Refer to the Table below for Control line combinations and type of transfer. When the C Control lines are grounded at the Interface, the time required for the bus lines to settle must be considered. If, for example, data is to be transferred from a device to the PC Register, data must be transferred from the DATA BUS (see *Table 9-1a*) to the adders. From the adders, data is loaded into the PC with a PC load signal. PC load is developed from	2	8

Table 9-1 Programmed I/O OMNIBUS Signals (Continued)

SIGNAL	ORIGIN	FUNCTION	TYPE LOAD	TYPE DRIVER
		C2L•BUS STROBE. Since BUS STROBE is generated by the processor during a normal IOT, C2L should be grounded not less than 280 ns before BUS STROBE. If the PC register is to be modified (both the PC and DATA applied to the adders), C2 L should be grounded not less than 400 ns before Bus STROBE L is generated.		
SKIP L	Interface	An IOT checks the flag for a ONE state and causes the device logic to ground the SKIP line if the flag is set. The result (PC + 1) is loaded into the CPMA. The SKIP line is sampled by the processor at TP3, and must be grounded 50 ns before TP3 in order for the skip to occur.	2	8
INT RQST L	Interface	Signal INT RQST is part of the Interrupt System. It is the method by which the device signals the processor that it has data to be serviced. When the device flag is set, signal INT RQST is immediately grounded. The processor samples the INT RQST line at INT STROBE time. If all the conditions for an interrupt are met, the processor then asserts signal INT IN PROG H.	2	8

Table 9-1a Table of Transfer Control Signals

Type of Transfer	Transfer Control Lines C0	C1	C2	Information Gated onto the Data Bus	Bus Set-up Time with respect to BUS STROBE	Action Required by Peripheral at Interface	Action by Processor**	Contents of Data Bus During Transfer
Output AC→ Data BUS AC un-changed	H	H	H	AC Reg.	280 ns	Load data bus into buffer.	Transfers AC to Data Bus. AC remains un-changed.	AC register only. User modifica-tion of this type of transfer may bring undesir-able results.
Output AC→ DATA Bus	L	H	H	AC Reg.	280 ns	Ground C0.	Transfers AC to Data Bus and clears AC.	AC Register.
AC Cleared						Load data bus into buffer.		
Input AC V Peripheral Data	H	L	H	Peripheral Data & Contents of AC reg.	280 ns	Gate peripheral data to data bus. Ground C1.	Transfers con-tents of AC to the data bus. The ORed result loaded into the AC.	AC ORed with Peripheral Data.
Input Jam-Data Bus →AC	L	L	H	Peripheral data	280 ns	Gate peripheral data to data bus. Ground C0 & C1.	Transfer data bus to AC reg-ister.	Peripheral Data
Relative Jump Data Plus PC→ PC	*	H	L	Peripheral data	400 ns	Gate peripheral data to data bus. Ground C2.	Transfer con-tents of PC and Data Bus to ad-ders. Load the added result in-to the PC.	Peripheral Data
Absolute Jump Data Bus →PC	*	L	L	Peripheral data	280 ns	Gate peripheral data to data bus. Ground C1 and C2.	Transfer con-tents of data bus to PC.	Peripheral Data

* Don't Care
** Bus Strobe loads AC or PC.

Table 9-2 Additional Programmed I/O OMNIBUS Signals for the Sophisticated User

SIGNAL	ORIGIN	FUNCTION	TYPE LOAD	TYPE DRIVER
BUS STROBE L	Interface during an extended I/O	Signal BUS STROBE is used to load the AC and PC registers. It is normally grounded by the processor during an I/O PAUSE and NOT LAST XFER H at TP3 time. Consequently, unless special I/O operations are being performed, the designer of an interface need not concern himself with BUS STROBE. BUS STROBE is a 100 ns negative-going pulse. For input transfers to the AC, or Absolute Jumps, data must be placed on the DATA BUS a minimum of 280 ns prior to BUS STROBE. For Relative Jumps, data must be placed on the DATA BUS a minimum of 400 ns prior to BUS STROBE. Input transfers to the AC and Absolute Jumps can take place within a normal IOT. However, Relative Jumps, which require 400 ns, present a timing problem. As with any operation requiring more than 280 ns, the problem is dealt with by stopping machine timing and grounding BUS STROBE at the interface. Allow 400 ns after data is applied to the DATA BUS before grounding BUS STROBE for 100 ns. Ground NOT LAST XFER at least 50 ns before TP3. This stops the processor timing at TP3 until NOT LAST XFER is again high thereby extending the length of TS3. Timing will continue only if NOT LAST XFER is high and BUS STROBE is generated.		

Table 9-2 Additional Programmed I/O OMNIBUS Signals for the Sophisticated User (Continued)

SIGNAL	ORIGIN	FUNCTION	TYPE LOAD	TYPE DRIVER
NOT LAST XFER L	I/O Interface	A ground level on this line indicates to the processor that the next BUS STROBE does not terminate the I/O transaction. Since most internal I/O devices use only one transaction per IOT, this signal is normally not grounded by the internal I/O devices. Thus, the internal devices usually only asserts its "C" lines and INTERNAL I/O. However, if the transfer is such that more than 280 ns are required between the time the device data is applied to the DATA BUS and signal BUS STROBE is grounded, or if multiple transfers are being made in a single IOT, the processor timing may be stalled long enough to complete the transfer. If for example the contents of the PC is to be added to the contents of the device data, additional time beyond the 280 ns is required to allow the ripple action of the adders to be completed. In this case, 120 ns more are needed. The device must ground NOT LAST XFER at least 50 ns before TP3. At TP3, the processor timing stalls. When device data is applied to the DATA BUS, the device must wait 400 ns and then ground BUS STROBE for 100 ns. Signal NOT LAST XFER should be brought high before the time when BUS STROBE is generated. This will restart timing with TS4 and negate signal I/O PAUSE L. *As indicated in the following* flow diagram (Figure 9-7), NOT LAST XFER accomplishes three basic tasks:	2	8

Table 9-2 Additional Programmed I/O OMNIBUS Signals for the Sophisticated User (Continued)

SIGNAL	ORIGIN	FUNCTION	TYPE LOAD	TYPE DRIVER
		a) Determines if the timing is to stop, b) Determines when BUS STROBE is generated, (if extended I/O), c) Determines when timing is to resume.		

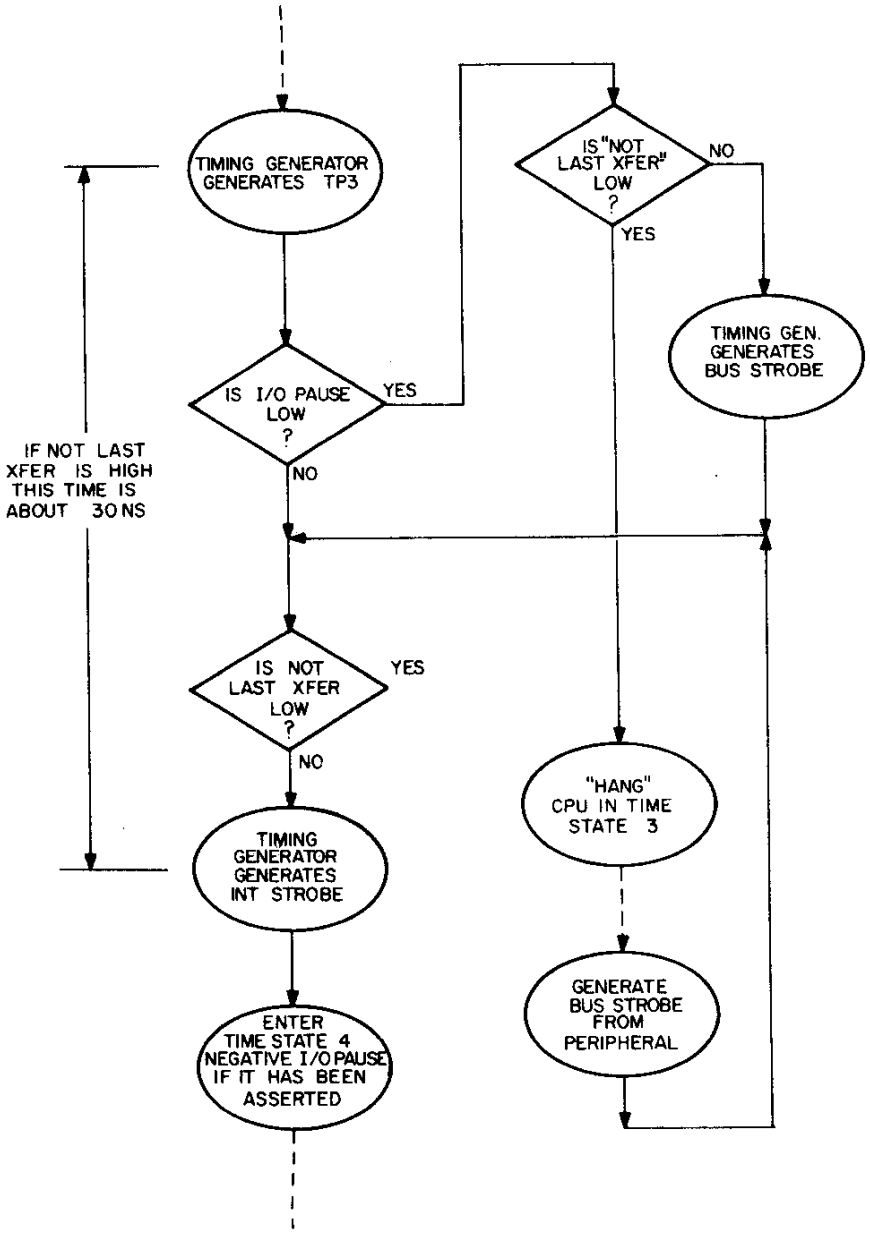


Figure 9-7 NLT Flow Diagram

Table 9-3 Data Break OMNIBUS Signals

SIGNAL	ORIGIN	FUNCTION	TYPE LOAD	TYPE DRIVER
MD0-11	Memory and Processor	In addition to its function under I/O, MD provides a one-way, 12-line data path between memory and the Data Break Interface. LOGIC STATES: high = 0 ground = 1	2	8
DATA0-11	Data Break Interface	The 12 DATA lines called DATA BUS are used to determine Break Priority, and to carry input data during a Data Break Cycle. LOGIC STATES: high = 0 ground = 1		
INT STROBE H	Processor	This 100 ns positive-going pulse occurs at TP3 (except for extended I/O and long EAE cycles) and is a necessary input to the timing chain to continue timing into TS4. For extended I/O and long EAE cycles, INT STROBE H is generated by BUS STROBE with NOT LAST XFER H. The leading edge of INT STROBE is the latest time in the machine cycle at which a break request can be accepted.	1	6
BREAK IN PROG L	Data Break Interface	This line is grounded at INT STROBE time if a break request is being made. This signal causes the BRK PROG lamp on the front panel to be lit during the next TS1 to indicate that a data break device has an active break request.	2	8
CPMA DISABLE L	Data Break Interface	This line is grounded by the Data Break Device at INT STROBE time if a break request is detected. CPMA BUS L causes the CP's Memory Address register to be disconnected from the MA lines at the next TP4. At the same time, the BKMA register of the highest priority device must be gated onto the MA BUS within 50 ns of the leading edge of TP4.	2	8

Table 9-3 Data Break OMNIBUS Signals (Continued)

SIGNAL	ORIGIN	FUNCTION	TYPE LOAD	TYPE DRIVER
OVER-FLOW L	Processor	This line is driven by a flip-flop that senses the carry from the adders at TP2. The flip-flop is set each time there is a carry or borrow out of the MB and is "ANDed" with TS3 before going to the OMNIBUS. This line is used, for example, during 3-cycle Breaks to indicate that the word count overflow has occurred.	2	8
BREAK DATA CONT L	Data Break Interface	BREAK DATA CONT is grounded when the contents of the MD are to be placed into the adders during a break cycle. This signal is used when a ONE is to be added to memory via the DATA BUS to increment either the Word Count or Current Address memory location. It is also used to perform an Add to Memory (ADM) type of Break. This line must not be changed during TS2 and is usually changed at TP4. Because MD DIR controls the transfer direction of memory data, the following truth table relates MD DIR and BREAK DATA CONT to the type of data break transfer.	2	8

BREAK DATA CONT USAGE

Type of Transfer	MD DIR	BREAK DATA CONT	INFO ON DATA BUS
Device→ Memory	H	H	DEVICE INFO
Memory→ Device	L*	X	X
	H	L	0
Memory PLUS Device→ Memory	H	L	DEVICE INFO

* Preferred Method

Table 9-3 Data Break OMNIBUS Signals (Continued)

SIGNAL	ORIGIN	FUNCTION	TYPE LOAD	TYPE DRIVER
MS, IR DIS L	Data Break Interface	This line is grounded at TP4 by the Break Device having the highest priority. This signal is used to disconnect the outputs of the MAJOR STATE and IR register outputs from the OMNIBUS and from all circuitry within the CP. The processing is terminated at the end of the current instruction cycle and resumes when MS, IR DIS is again high. The start of MS, IR DIS L is the start of the DMA state. In addition, MS, IR DIS L also enables a data path from the DATA BUS to the adders to provide DATA to the MB or DATA + MD to the MB.	2	8
BREAK CYCLE L	Data Break Interface	BREAK CYCLE is grounded at TP4 by the break device having the highest priority. This signal causes the BRK lamp on the front panel to be lit during the next TS1 to indicate that the Break Cycle has started.	2	8
MA,MS LOAD CONT L	Data Break Interface	This line is grounded at TP1 by the device having the highest priority and remains grounded during Break until the TP4 following the last Break Cycle. MA,MS LOAD CONT L prevents the CPMA and MS registers from being loaded at TP4.	2	8
MD DIR	Processor or Data Break Interface	Refer to table 9-4	3	8

Table 9-4 Basic System OMNIBUS Data and Control Signals

SIGNAL	ORIGIN	FUNCTION	TYPE LOAD	TYPE DRIVER
MD0-11	Processor and memory	The 12 Memory Data lines carry information to and from memory of the currently addressed location. The contents of the addressed memory location are applied to the Memory Data lines beginning in the last half of TS1. If the major state is FETCH or DEFER (non-auto index), the contents of the MD lines will not change for the remainder of the cycle. The MD lines serve as the input into memory during every write operation (which occurs during TS3 and TS4). If the Major State is DEFER (auto index) or EXECUTE, the contents of the MD can change at TP2. This change is controlled by signal MD DIR which allows data to be applied to the MD lines from the memory register when MD DIR is grounded and inhibits the transfer of MB data to memory. The MB register provides the only external means of inputting into memory and can do so only when MD DIR is high. For normal machine operation, the MD lines provide instructions, addresses, operands and data. For I/O devices, the MD lines provide the device select and operation codes. For data break devices, the MD lines carry data into the device. LOGIC STATES: high = 0 grounded = 1	2	8

Table 9-4 Basic System OMNIBUS Data and Control Signals (Cont.)

SIGNAL	ORIGIN	FUNCTION	TYPE LOAD	TYPE DRIVER
MD DIR	Processor or Programmer's Console or Data Break Device	The control of external data to memory and data received from memory is provided by signal MD DIR. When high, MD DIR gates the contents of the MB register onto the MD lines and is thereby applied to memory during memory WRITE time. When grounded, MD DIR gates the contents of the memory Sense Amps out to the MD lines during the memory READ time. Thus, MD DIR can control only the place at which data is applied to the MD lines. When data is applied to the MD lines from the MB Register, data cannot be applied to the MD lines from the Sense Amps. During FETCH and DEFER (non-autoindex), the contents of the MD lines cannot change. The instruction or address read from memory is written back into the same memory addressed location. MD DIR L assures that these lines will not change. During EXECUTE and DEFER (autoindex), the contents of the addressed memory location are applied to the MD lines until TP2. At this time, MD DIR is brought high so that the contents of the MB Register can be applied to the MD lines and subsequently written into the same memory location during the WRITE portion of the memory cycle. During the manual operation of the processor from the Programmer's Console, MD DIR can be changed without considering the time states.	3	8

Table 9-4 Basic System OMNIBUS Data and Control Signals (Cont.)

SIGNAL	ORIGIN	FUNCTION	TYPE LOAD	TYPE DRIVER
		During a Data Break Operation, MD DIR is controlled by the Data Break device depending upon the type of transfer (input or output). MD DIR should be changed at TP1 time.		
MA 0-11	Processor or Data Break Interface	Used to address memory to any one of 4096 possible locations. This address is changed only at TP4 time. The address is normally developed in the processor. However, during a data break, the MA lines can be used for break addresses, which originate in the data break module. When the processor is executing an instruction, the address always originates in the processor. LOGICAL STATES: 1 = low 0 = high	3	8
EMA 0-2	Processor or Data Break Interface	Used only when the extended memory is provided. These 3 bits are combined with the 12 bit Memory address to form a 15-bit memory address. This is necessary to specify one location out of 32,768 possible locations. The extended address bits specify the memory field in use. MA11 is the least significant bit and EMA0 is the most significant bit. All 12 or 15 lines are high for a zero and low for a one. Thus, if the machine does not contain a Memory Extension Control, the EMA bits are automatically zero (high), selecting the lowest 4K of memory. LOGICAL STATES: 1 = low 0 = high	3	8

Table 9-4 Basic System OMNIBUS Data and Control Signals (Cont.)

SIGNAL	ORIGIN	FUNCTION	TYPE LOAD	TYPE DRIVER
DATA0-11	Nearly all portions of the machine except memory	The 12 DATA lines called DATA BUS serve as a multipurpose bidirectional bus. Generally, the DATA BUS is the in/out path between the peripheral and the AC register. However, the DATA BUS is also between the AC Register/MQ Register and the processor adders; and therefore capable of applying peripheral data or AC/MQ data to the adders. The DATA BUS usage with respect to processor timing is illustrated in the following table.	4	8

DATA BUS USAGE

Machine Timing Within Major States	Major States			
	F	D	E	DMA
TS1		Indicators		
TS2	CPU	NOT	CPU	DATA → MB
TS3	I/O DIALOGUE (Only if an IOT Otherwise CPU)	USED	CPU	NOT USED
TS4		Priority Determination		

MEM START	Programmer's Console or Power Fail Option	This line is grounded for a minimum of 100 ns to initiate a memory cycle. It must not be grounded after TP2. Memory cycles continue automatically until STOP is grounded.	2	
ROM ADDRESS	Rom	When this line is high, the Read/Write memory runs normally. When this line is low, the Read/Write memory does not function, despite memory timing signals on the bus. This line is used when a small ROM is used	3	8

Table 9-4 Basic System OMNIBUS Data and Control Signals (Cont.)

SIGNAL	ORIGIN	FUNCTION	TYPE LOAD	TYPE DRIVER
		with addresses that overlap the main memory. If the ROM sees its address on the MA lines, this line is immediately grounded. If ROM ADDRESS is grounded during the Execute portion of a JMS instruction, the next instruction will be taken from the addressed location, thus saving a ROM location.		
MEMORY WRITE, SOURCE, STROBE, INHIBIT, RETURN	Timing Generator	These five signals control the memory, and may vary if different memory and timing modules are used. MEMORY WRITE is high during the write portion of the memory cycle. SOURCE is used to turn on memory current. It is high when read or write current is to be turned on. RETURN and SOURCE turn on (go high) at the same time, but RETURN turns off 50 ns later to insure that the stack does not remain capacitively charged. INHIBIT turns on the inhibit drivers when positive. STROBE provides a time reference from which the output of the sense amplifiers are sampled. STROBE goes positive before data is actually ready in the sense amplifiers. Each memory then delays this leading edge by the optimum amount. This precaution allows for stack variation. If data is read from memory, the negative going edge of STROBE indicates the data on the MD lines is valid.	3	8
IRO, 1 & 2	Processor	These 3 lines indicate the effective instruction being processed. They usually, (but not always), indicate the contents of the IR. Lines are low for a 1 and high		

Table 9-4 Basic System OMNIBUS Data and Control Signals (Cont.)

SIGNAL	ORIGIN	FUNCTION	TYPE LOAD	TYPE DRIVER
		for a 0. All instruction processing and major state gating is derived from these lines.		
F, D, E	Processor	The 3 major states lines, like the IR lines, indicate the effective major state. The appropriate line is low to indicate its major state. Only one of these lines should be grounded at any time. A fourth major state called Direct Memory Access (DMA) is activated when MS, IR DISABLE is low and F,D or E are not grounded by an external device.	3	8
F SET		F SET is similar to major states described above. This line must not be grounded by modules external to the CP. F SET indicates the next machine cycle is a fetch unless disabled. Note that DMA State causes F SET.	2	8
LINK LOAD LINK DATA	Timing Generator, Peripherals	These two lines may be used to jam one bit of information into the LINK. LINK DATA is low for a 1 and high for a zero. LINK LOAD is normally high and is brought to ground for 100 ns (minimum) to cause loading.	LINK LOAD 1 on both ends of bus LINK DATA: 2	9 8
LINK	Processor	This line indicates the state of the link bit in the processor. It is high if the link is 0, and low if the link is a 1.	2	
LD ADD EN	Programmer's Console	When the ADDR LOAD key is pressed this signal is asserted (grounded). A data path is enabled to allow the DATA BUS to be transferred to the major register bus (see Pulse LA).	2	8

Table 9-4 Basic System OMNIBUS Data and Control Signals (Cont.)

SIGNAL	ORIGIN	FUNCTION	TYPE LOAD	TYPE DRIVER
KEY CONTROL	Programmer's Console	This line is grounded by the Programmer's Console when the operator depresses the EXTD ADDR LOAD, EXAM, or DEP key. When EXTD ADDR LOAD key is depressed, CPMA LOAD is inhibited preventing an address, intended for the Memory Extension Control, from being loaded into the CPMA. If the DEP or EXAM key is used, CPMA LOAD is again inhibited so that data will not be loaded into the CPMA. KEY CONTROL L also generates STOP which resets RUN so that the timing will stop at the next TS1 and causes MA + 1 to go to the PC register. KEY CONTROL L is negated at TP4. KEY CONTROL L also prevents interrupts from occurring. When ADDR LOAD is depressed, KEY CONTROL H remains high so that CPMA LOAD may be developed.	2	8
STOP	Timing Generator or Programmer's Console	STOP is asserted (grounded) by the STOP key and F SET, by the SINGLE STEP key, by KEY CONTROL (low), or by the HLT instruction. It is sampled at TP3. If this line is low, processing is stopped at the end of the current memory cycle.	2	8
PULSE LA	Programmer's Console	When ADDR LOAD or EXTD ADDR LOAD key is pressed, this positive pulse either causes the contents of the DATA BUS to be loaded into the CPMA (if KEY CONTROL is high), or causes DATA 6 thru 11 to go to the Extended Memory IB, IF, DF (if Key Control is low). Note that PULSE LA does not initiate a memory cycle (see LD ADDR EN).	5	10

Table 9-4 Basic System OMNIBUS Data and Control Signals (Cont.)

SIGNAL	ORIGIN	FUNCTION	TYPE LOAD	TYPE DRIVER
IND 1 & IND 2	Programmer's Console	These 2 lines control the data placed on the DATA lines at TS1 time. IND 1 IND 2 Effect High High Status word goes to DATA BUS at TS1 High Low C (MQ) goes to Data Bus at TS1 Low High Data Bus Low Low C (AC) goes to Data lines at TS1 Status word format: Bit Function — Front Panel Abbr. 0 Link — L 1 "Greater than" flip-flop — GT 2 Interrupt Bus — INT BUS 3 No Int. Allowed — NO INT. 4 Interrupt On — ION 5 User Mode — UM 6 Instruction Field 0 — IF 0 7 Instruction Field 1 — IF 1 8 Instruction Field 2 — IF 2 9 Data Field 0 — DF 0 10 Data Field 1 — DF 1 11 Data Field 2 — DF 2	2	8
SW	Programmer's Console	SW is a line controlled by front panel switch SW. When the switch lever is up, the line is low. When the lever is down, the line is high.	2	8
INT IN PROG H	Timing Generator	INT IN PROG signifies that the CP is in the process of honoring an interrupt request. This line is asserted (brought to +3V) at INT STROBE time if all	4	8

Table 9-4 Basic System OMNIBUS Data and Control Signals (Cont.)

SIGNAL	ORIGIN	FUNCTION	TYPE LOAD	TYPE DRIVER
		conditions for granting an interrupt request are present. It is negated at TP1 of the next cycle. Grounding this line prevents honoring an interrupt, even though all other conditions are met.		
OVER-FLOW L	Processor or Data Break Interface	This line is driven from a flip-flop that senses the carry from the adders at TP2. The flip-flop is set each time there is a carry or borrow out of the MB and is "ANDed" with TS3 before going to the bus. For the basic system, the flip-flop is set by a carry from an ISZ instruction. Refer to Table 9-1b for use in Data Breaks.	2	8
RUN L	Timing Generator	When low, RUN indicates that the machine is executing instructions. The Run Flip-Flop is set by Mem Start and cleared at TP3 if STOP is asserted.	NO LOAD	7
TS1 L, TS2 L, TS3 L, TS4 L	Timing Generator	These time state lines are high if negated, and low if asserted. Each time state precedes its corresponding time pulse. Time states are always 200 ns or more in duration, and change 50 ns after the leading edge of the time pulse. The machine is in TS1 when stopped.	1	6
TP1 H, TP2 H, TP3 H, TP4 H	Timing Generator	These 100 ns positive-going pulses originate in the timing module. The exact spacing of the timing pulses is a function of fast or slow cycle. The source of all timing is a 20 MHz crystal clock and a frequency divider. The timing generator	1	6

Table 9-4 Basic System OMNIBUS Data and Control Signals (Cont.)

SIGNAL	ORIGIN	FUNCTION	TYPE LOAD	TYPE DRIVER
		starts running any time Mem Start is issued, and continues to run until TP4 occurs. At that time, if STOP was negated (high) at TP3, the timing generator continues to run. Each Time Pulse except TP3 overlaps 2 Time States. For example, TP1 begins 50 ns before the end of TS1 and ends 50 ns after TS2 has started.		
POWER OK H	Power Supply	POWER OK, when high, indicates that the dc voltage from the power supply is adequate to allow proper functioning of the machine. If this line becomes negated, no new memory cycles will be started. Also, after a delay long enough to complete the current cycle, the memory drive current is inhibited. When this line is negated, INITIALIZE is generated.	5	10
INITIALIZE H	Timing Generator	INITIALIZE is a positive-going pulse of at least 600 ns duration. This pulse is used to clear AC and LINK, and to clear all flags in peripherals. It is generated if POWER OK is negated, by the Clear Key on the front panel, and by an IOT (6007).	5	10
USER MODE L	Extended Memory Control	This signal originates in the Time Share portion of the Extended Memory Control. When asserted (ground), it disables OSR, LAS, IOT, and HLT instructions. OSR and LAS are disabled at the panel by inhibiting the placing of SR on the DATA lines. The IOT and HLT instructions are disabled in the Central Processor.	2	8

SECTION 2 HOW TO CHOOSE THE TYPE OF I/O TRANSFER

The type of I/O transfer must be first considered before beginning the task of designing the I/O interface.

The basic types of peripherals are used with the PDP-8/E: one that is designed to transmit or receive one character (12-bit word) per service routine by the processor; and one that is designed to transmit or receive a block of characters (a series of 12-bit words) per service routine by the processor.

DATA TRANSFER TYPES

Data transfer can occur in any one of three data transfer facilities. These are: Programmed I/O Transfers, Interrupt facility, and Data Break facility.

PROGRAMMED I/O TRANSFER—The simplest method of accomplishing an input/output transfer is the Programmed I/O Transfer. This method relies upon the processor to occasionally check the Status Flag and service the flag with a subroutine.

INTERRUPT FACILITY—A more efficient method of input/output transfers is to employ the Interrupt System. This method includes all of the elements in the Programmed I/O transfer except the time of transfer. The device decides when to transfer by grounding an INTERRUPT REQUEST line. The processor responds at the end of the current instruction.

DATA BREAK TRANSFER—A still more efficient method of transfer is to the Data Break System. Whenever the data break device decides that it is time to transfer, it generates MS, IR DIS to force the processor into a Direct Memory Access State and CPMA DIS to disable the CPMA register. This leaves the data break device free to supply its own address and to manipulate the Major Registers Control logic so that it can input and output data at will. The processor responds to a break at the end of the current cycle. Note that, in general, data break requires more hardware than Programmed I/O. Additional logic is necessary to handle addressing, etc., and some programmed I/O is necessary to initialize and check status of the device.

INTERFACING TO THE PROCESSOR

Two sides of the interfacing must be considered: the processor side and the interface control side. It is necessary to understand that both the processor and the interface control share common lines on the OMNIBUS. Furthermore, although the interface control may place information on these common lines, only when certain control lines are asserted is information loaded into the processor. This requirement is necessary for both Data Break and Programmed I/O Transfers.

Because each line on the OMNIBUS is shared by a number of devices as well as elements of the processor, it is most important that each line be used by only one device at any given time except where specified in this handbook.

The OMNIBUS/Basic System Interface is illustrated in figure 9-8. For most I/O operations, data is received and outputted by the Major Registers module. Data is received from the OMNIBUS 12 data lines called DATA BUS and applied to the input gates. By asserting the appropriate control lines, data can be loaded into the MA, MB, MQ, AC or PC register; or be added to existing data in one of the registers and correspondingly become new data for the DATA BUS or Memory Data or a new memory address. Thus for I/O operations, the user's path into the basic system is via the DATA BUS. Data transmitted from the processor is also via the DATA BUS except when using a Data Break Device. Notice that the MD lines on the OMNIBUS are connected to memory via a two-way path. When a Data Break Controller is connected to the MD lines, information may be transmitted from memory without having to go through any Major Register. However, from the Data Break device, data is always received on the DATA BUS and applied to memory via the Major Registers.

A more comprehensive picture of both Data Break Control and I/O Interface Control operating with the basic system is provided in figure 9-9. The logical sections of both I/O interface and Data Break interface are identified. Each of these sections consists of simple logic and are expanded upon in sections 3 and 4 of this chapter.

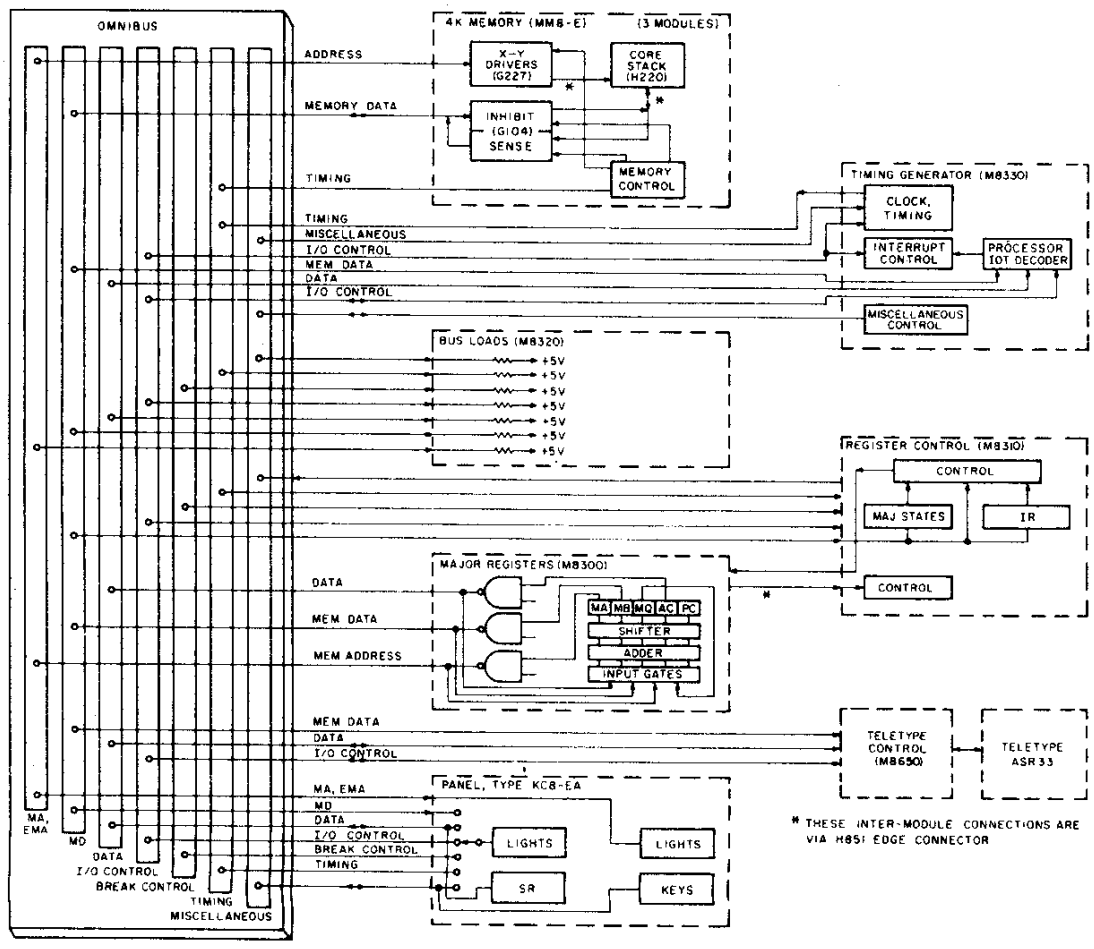


Figure 9-8 OMNIBUS/Basic System Interface

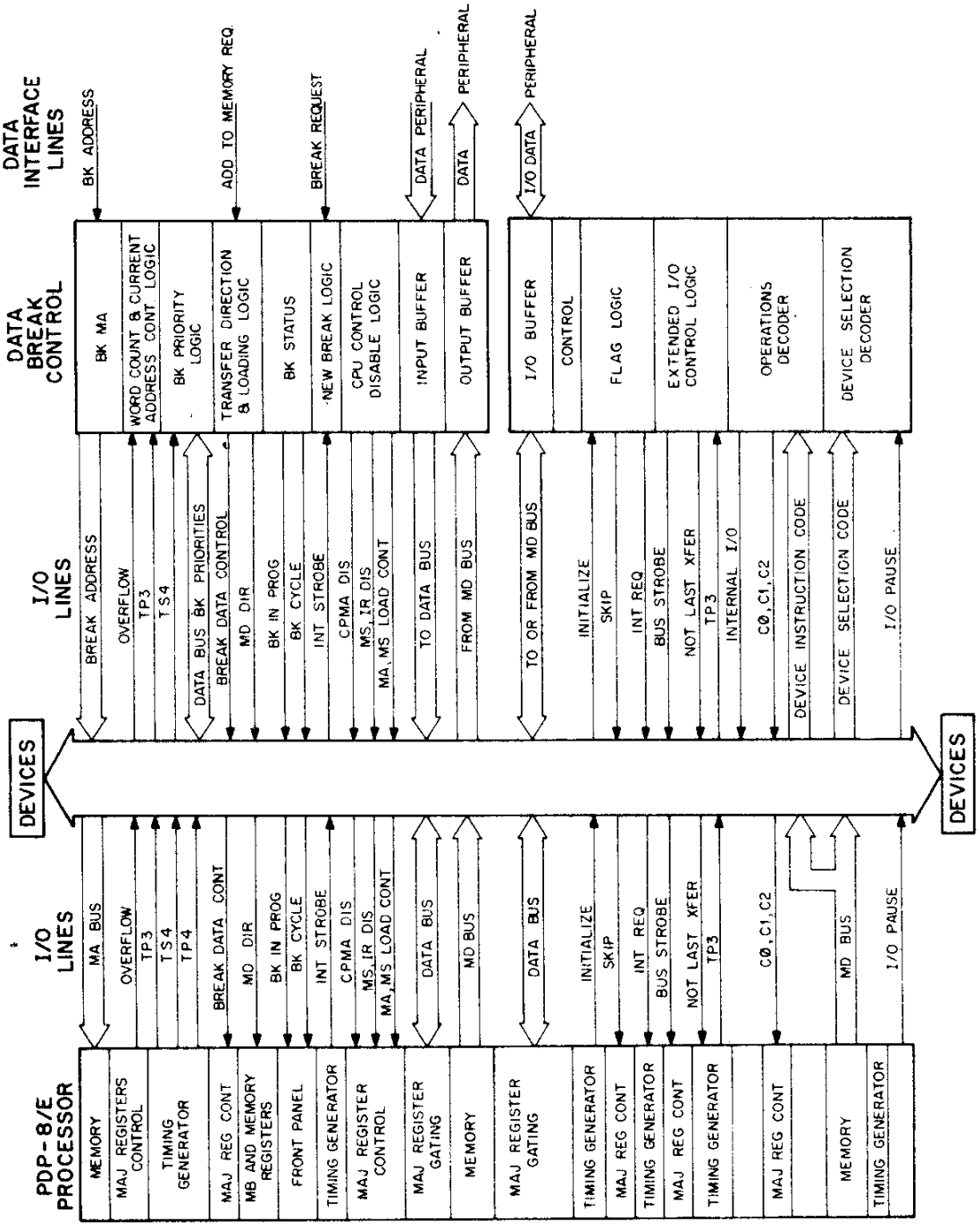


Figure 9-9 OMNIBUS/ I/O Signal Interface Block Diagram

DATA TRANSFER RATES

One means of determining the type of data transfer is by examining your projected data transfer rates. Figure 9-10 illustrates a general guideline based upon transfer rates. However, it does not consider the number of I/O devices or the amount of calculations expected of the processor.

Any device with transfer rates up to approximately 60 characters per second can be easily serviced by a simple programmed I/O. However, adding more interfaces requires a closer examination. The Programmed Interrupt System should be employed when using transfer rates above 60 characters per second up to 5K characters per second. For data rates above 5K characters per second, the Data Break Facility should be employed.

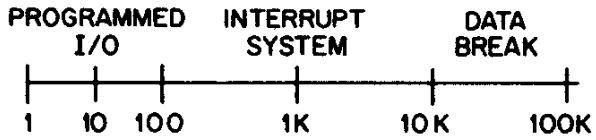


Figure 9-10 Data Transfer Rates (Characters Per Second)

DEVICE CODES

The device codes for selecting each device are given in Table 9-5.

Table 9-5 PDP-8/E Device Codes

00	Central Processor, Type KK8-E
01, 02	High Speed Tape Reader/Punch & Control, Type PC8-E, PP8-E, PR8-E
03, 04	Console Teleprinter Control, Type KL8-E
05, 07	Oscilloscope Display Control, Type VC8-E
10	Power Fail Detect and Auto Restart, Type KP8-E
10	Memory Parity, Type MP8-E
11	Redundancy Check Control, Type DP8-EP
12	Reserved for Card Punch and Control
13	Real Time Clock, Type DK8-EA, DK8-EC, DK8-EP
14-17	Reserved for Special Systems and Customer's use
20-27	Memory Extension and Time Share Control, Type KM8-E
30-37	
36, 37	General Purpose Interface, Type BB08
40-47	Synchronous Data Interface, Type DP8-EA, DP8-EB
50-51	
52	Analog Multiplexer, Type AM8-EA
53	A/D Converter, Type AD8-EA
50-57	Buffered Digital I/O, Type DR8-EA
60-62	Disk and Control, Type DF32-D
60-62, 64	Disk Control, Type RS08
63, 67	Card Reader and Control, Type CR8-E, CM8-E
65	Plotter Control, Type XY8-E
66	Line Printer and Control, Type LE8
70-72	Industry Standard Magnetic Tape and Control, Type TM8-E
70-77	DECtape Control, Type TD8-E
73-75	Disk File and Control, Type RK8
76-77	DECtape Control, Type TC08

SECTION 3 DESIGNING BASIC PROGRAMMED I/O INTERFACE CONTROL CIRCUITS

The basic interface control circuits to either transfer data in or transfer data out consists of: 1) a device selection circuit, 2) a device operations decoder, 3) I/O control logic, and 4) input/output buffers. An example of a basic programmed I/O interface control is illustrated in figure 9-11 followed by the related timing.

A general rule to follow for interfacing with the OMNIBUS is given as follows:

SIGNALS TAKEN OFF THE OMNIBUS: Use DEC380, DEC314, or DEC384 ICs (or equivalent). This is recommended to minimize bus loading.

SIGNALS PLACED ONTO THE OMNIBUS: Gate signals onto the OMNIBUS with DEC8881 ICs (or equivalent). This is recommended because of the low leakage current characteristic that allows a greater number of devices to be "wired ORed" together.

DEVICE SELECTION CIRCUIT—MD3-8 bits are used to carry the device select information. The example given in figure 9-11 shows the DEC380 and DEC314 being used as a simple decoder. When octal 52 is received and signal PAUSE is grounded by the processor, gate 314 is qualified. The output is used to assert signals INTERNAL I/O L and MY DEVICE L. No operation can occur unless signal MY DEVICE is grounded by the device selection decoder.

OPERATIONS DECODER—MD9-11 bits determine the type of operation to be performed. Three DEC380's are shown (see figure 9-11) receiving these bits. The outputs of these gates are in turn presented to a binary-to-octal decoder type 8251 and the decoded results control the interface in the manner shown in table 9-6.

Table 9-6 Operations Decoder IOT's

IOT	ACTION REQUIRED BY INTERFACE	RESULT
6521	CLOCK OUTPUT BUFFER	AC→DATA BUS→DEVICE AC UNCHANGED
6522	GROUND C0	CLEAR AC
6523	CLOCK OUTPUT BUFFER & GROUND C0	DATA BUS→DEVICE; CLEAR AC
6524	GATE DEVICE DATA ONTO DATA BUS AND GROUND C1	DEVICE DATA ORed with AC→AC
6525	CLEAR FLAG	CLEAR FLAG
6526	GATE DEVICE DATA ONTO DATA BUS AND GROUND C0 and C1	JAM INPUT OF DEVICE DATA→AC
6527	GROUND SKIP line if flag(1)	PC + 1→CPMA

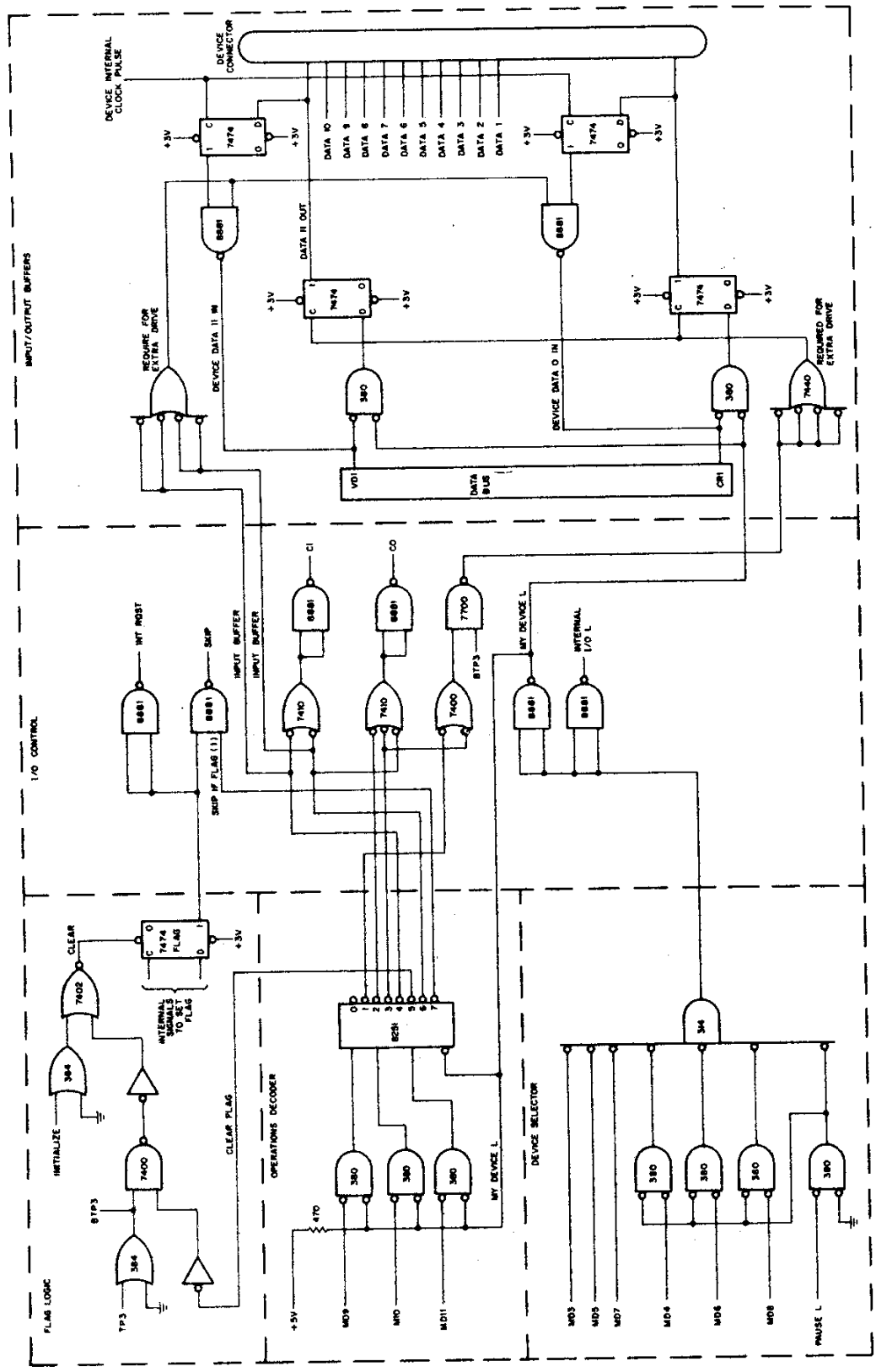


Figure 9-11 Sample Programmed I/O INTERFACE Control Logic

FLAG LOGIC—The flag is represented as a 7474 D-type flip-flop. For an input transfer, the flag may be clocked by the device internal clock pulse with a DATA IN signal used as the data input. If the transfer is to be an output transfer, the clock input could be a timing pulse and the data input could be the output of the operations decoder. The flag represented in the example is used for an input transfer. For both input and output transfers, two flags are required.

INTERRUPT REQUEST—The processor responds to the INT RQST line by completing the current instruction and then executing a JMS to location 0. Simultaneously, the interrupt system is turned off. The execution of the JMS instruction saves the current program count in location 0. It is up to the program to identify the interrupting device by polling sequentially (testing) device flags. After the device has been serviced, the interrupt service routine returns to the main program with a JMP I 0 instruction.

OUTPUT BUFFER—The output buffer serves to receive processor data during an IOT instruction and outputs data to a device at the device timing. Two types of output transfers can be made depending upon the device. A parallel to parallel transfer will transfer the parallel data from the DATA BUS to an equal number of parallel lines to the device. A parallel to serial transfer will load the parallel data from the DATA BUS into the buffer and shift the data out to a single output line to the device. Figure 9-12 illustrates a parallel to serial output buffer. Signal LOAD BUFFER gates the contents of the DATA BUS onto the set side of the register flip/flops. This illustration shows 8 data bit flip/flops in between an ENABLE and LINE flip/flops. A shift control circuit must also be added to provide the necessary buffer control.

A more simple version is the parallel to parallel transfer. The illustration in Figure 9-12 could be slightly modified to include only the input gates and the flip/flops; data can be applied to the output circuit from the one side of each flip/flop.

In the sample Programmed I/O Interface Control (figure 9-11), each input flip-flop is a type 7474 and is represented on the illustration as a 12-bit buffer register. The data OUTPUT is clocked by IOT 6521 or 6523 and TP3. However, for a parallel to serial conversion (figure 9-12), the LOAD IOT loads the buffer with TP3H and shifts the data with each internal clock & shift pulse.

INPUT BUFFER—The input buffer serves to receive device data at the device timing and applies the data to the DATA BUS during an IOT instruction. Figure 9-13 illustrates a serial to parallel input buffer representing 8 bits applied to DATA 4 through 11 of the DATA BUS. A slight modification eliminating the shift function and having each data input applied to the corresponding flip/flop makes the serial input into a parallel input type buffer.

In the sample Programmed I/O Interface Control (figure 9-11), each buffer register is a type 7474 and is represented on the illustration as a 12-bit buffer register. The data input is clocked in by the device's internal timing and gated out to the DATA BUS by IOT 6524.

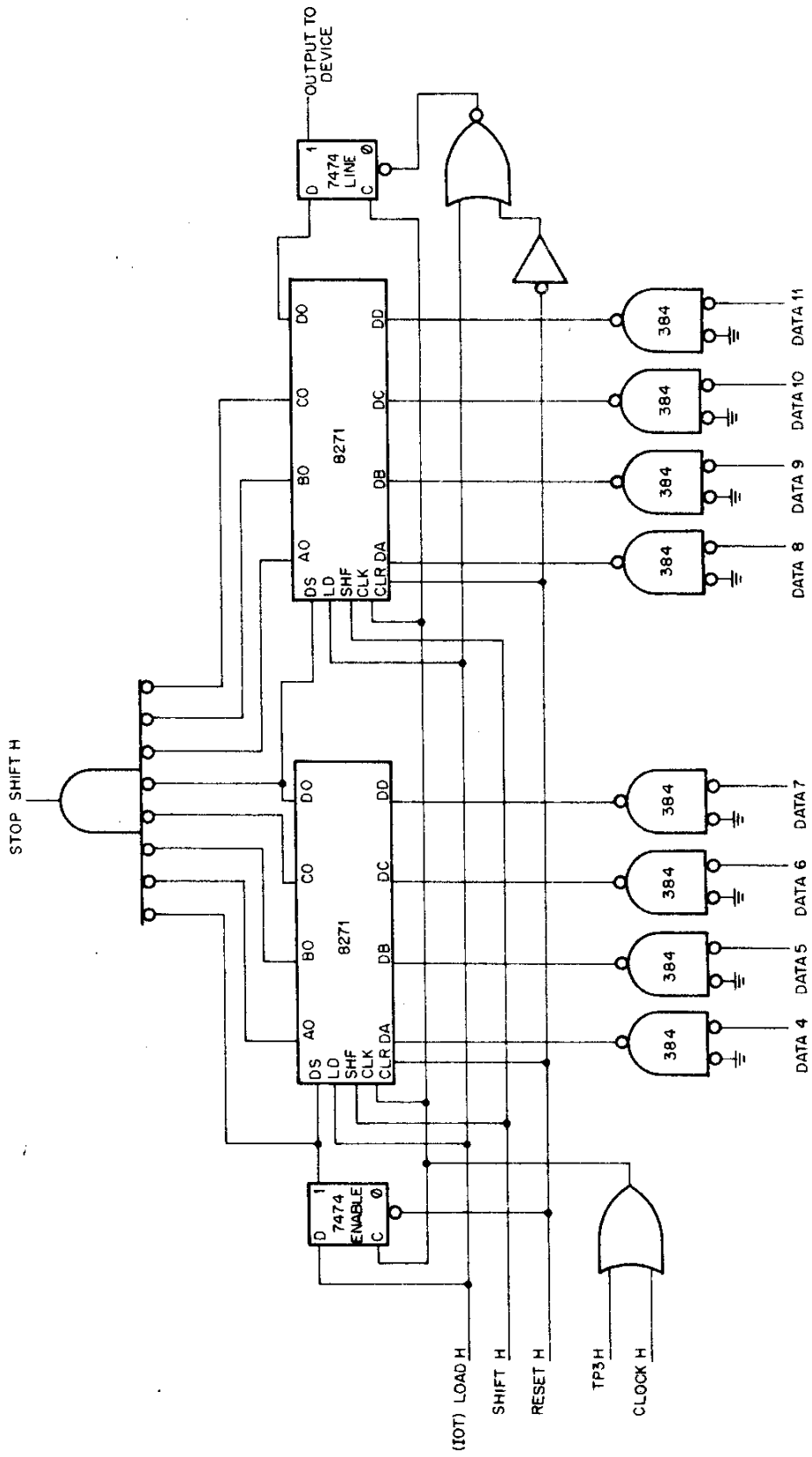


Figure 9-12 Parallel to Serial Output Buffer

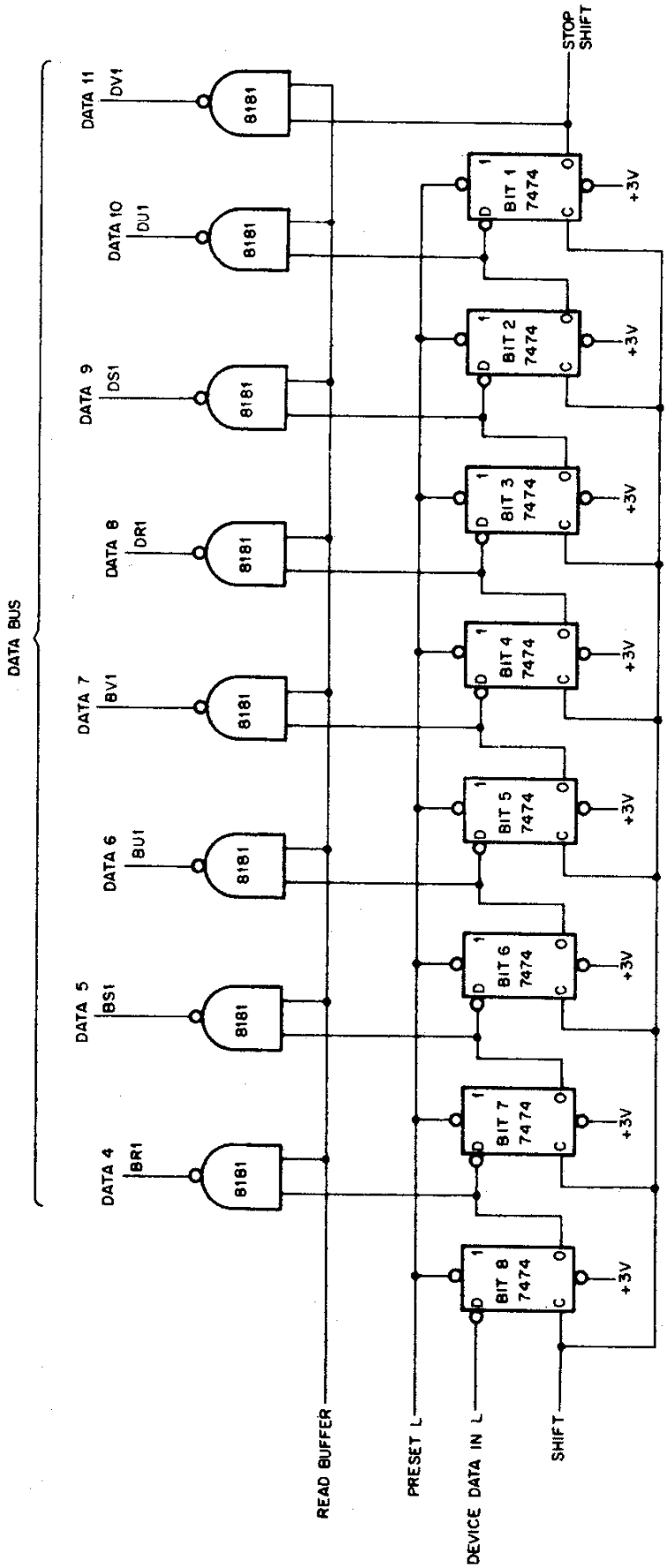


Figure 9-13 Serial to Parallel Input Buffer

I/O CONTROL—The I/O Control includes INT RQST which immediately responds when the flag is set; SKIP which is grounded when IOT 6527 is decoded and the flag is set; C0 and C1 which may be grounded by the operations decoder during various conditions of data transfer and input/output enabling logic which responds to the operations decoder and controls the I/O buffers.

INPUT/OUTPUT TIMING FOR PROGRAMMED I/O INTERFACES—A timing diagram corresponding to the Programmed I/O interface example is illustrated in figure 9-14. An explanation of the time periods from A to J is given in the following.

PERIOD	TIME	FUNCTION
A—D & E—J	350ns	Time required to perform the transfer (PAUSE)
A—B & E—F	$\leqslant$ 70ns	Time required to decode the device selection and assert INTERNAL I/O.
A—C & E—F	$\leqslant$ 100ns	Time required to decode the IOT and assert the necessary "C" lines or SKIP and supply data if needed.
D & J		The time when the transfer takes place. Note that the DATA BUS will change at this time. This is the reason that edge triggering must be used.

EXTENDED I/O—Only if the data input time is a problem should the extended I/O functions be employed. The two control signals that allow extended I/O are NOT LAST XFER and BUS STROBE. When NOT LAST XFER is grounded, the processor timing is stalled at TP3. BUS STROBE, which is normally asserted by the processor, must be grounded when the data is ready to be loaded into one of the major registers. When BUS STROBE is asserted and NOT LAST XFER is not asserted, the processor timing resumes. The net result is the extension of Time State 3.

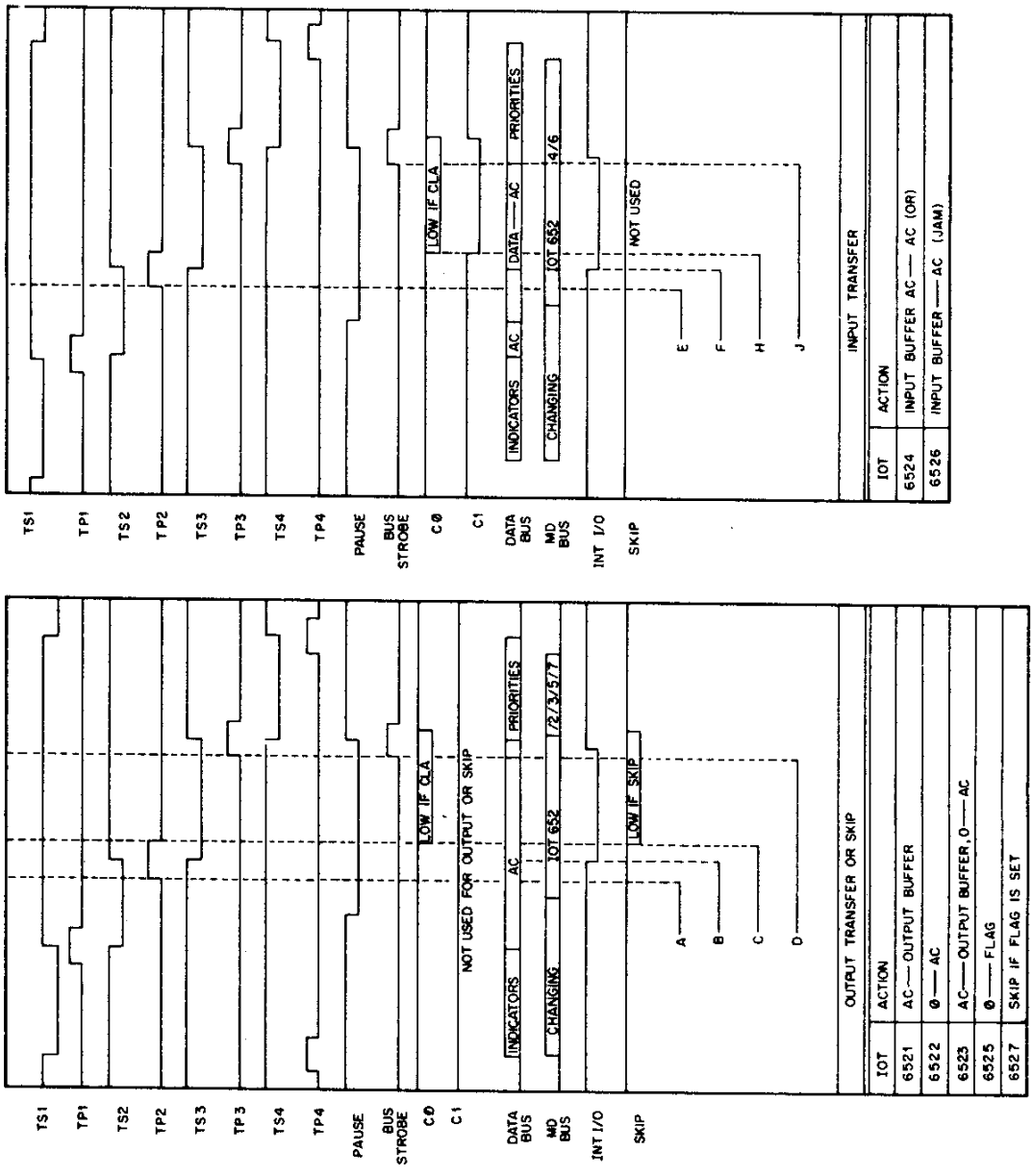


Figure 9-14 Timing for Sample Programmed I/O Interface Control

SECTION 4 DESIGNING A BASIC DATA BREAK INTERFACE

GENERAL

The data break control (refer to figure 9-15) consists of logic mounted on an OMNIBUS compatible QUAD type module. The communications link to the processor is via the OMNIBUS into which the module plugs. The communication to the controlled peripheral is via a connector (mounted on the data break interface module) and corresponding cable to the peripheral.

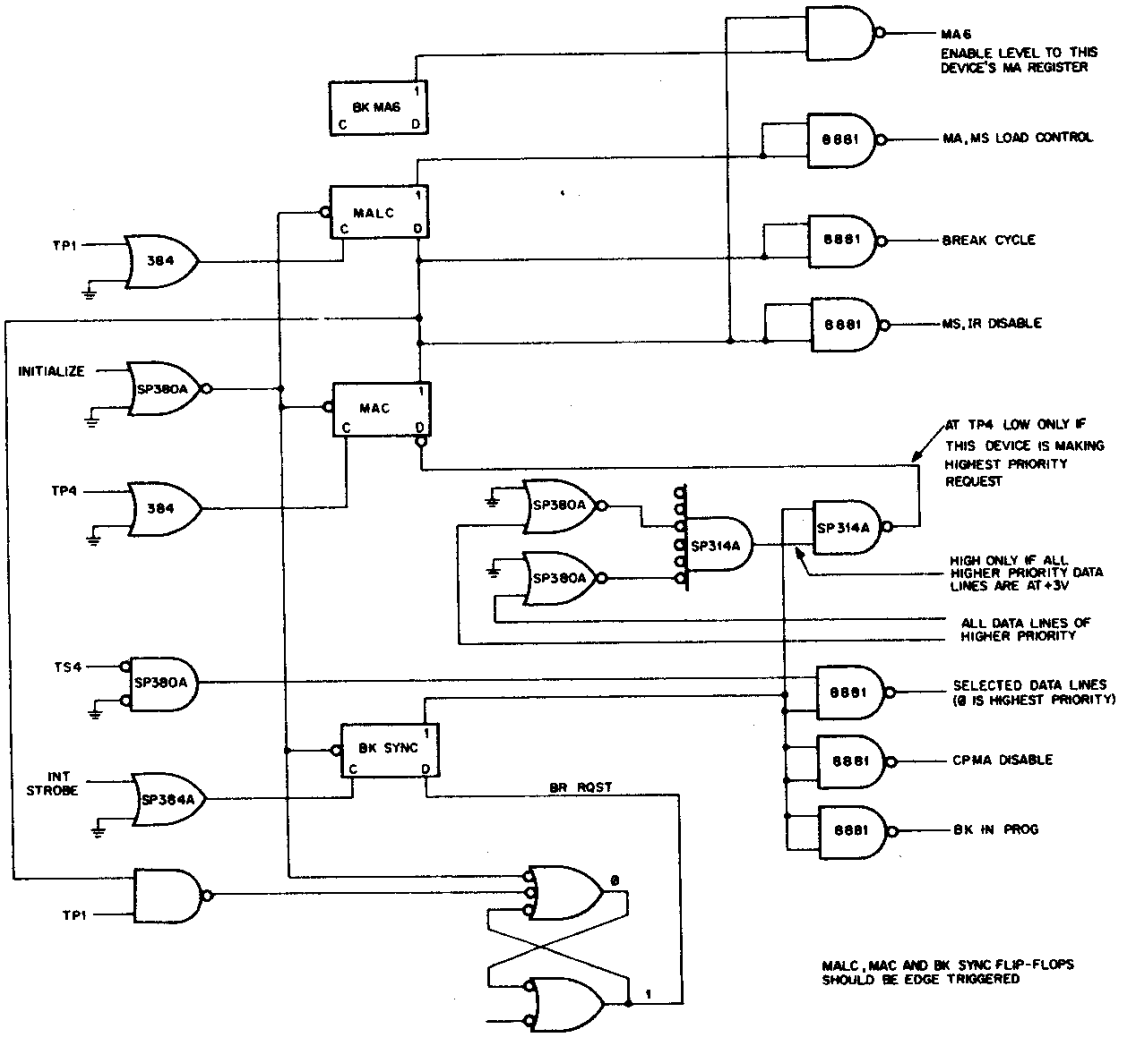


Figure 9-15 Basic Data Break Control Logic

The data break control breaks into the processor sequence of events at the end of a processor cycle whenever a data break peripheral has data to be transferred into or out of the processor. When a new break request is received by the data break control, the data break control waits for INT STROBE and then initiates disabling of the CPMA register. It waits for MY PRIORITY to be established and then disables the Major State and Instruction registers. This places the processor in the DMA state by disqualifying the output gates and in the same manner inhibits any outputs of the Instruction Register.

BREAK ADDRESS—The data break control contains a 15-bit Break Memory Address (BKMA) Register (12 for 4K and 3 for extended memory). When applied to the MA lines, the BKMA can directly address any memory location in any one of 8 memory fields. If the device is a 3-cycle break device, the control must be concerned with 3 addresses.

DATA PATHS—For all data break output transfers, the data path is via the MD lines. For all input transfers, the data path is via the DATA BUS.

STATUS REGISTERS—The status registers of a data break device may be read into the AC register by means of an IOT instruction. To accomplish this, a separate interface control, meeting the requirements of a programmed I/O operation is required.

BREAK PRIORITIES—Twelve break priorities are available via the DATA BUS during TS4. The highest priority is DATA 0. A break priority decoding network in each data break device checks all higher-order bits to make sure they are all at +3v and grounds the DATA line of its priority to inhibit any lower order devices. The device doing the decoding executes its break cycle. PRIORITY MUST BE TESTED PRIOR TO EVERY DMA CYCLE.

TRANSFER DIRECTION AND LOADING LOGIC—A method of controlling the type of transfer (input, output, or add to memory) must be provided on the data break control interface. To transfer data into memory via the DATA BUS, the device data must be applied to the memory buffers. This is accomplished by leaving signals BREAK DATA CONT H and MD DIR H so that the device data can be applied to memory. When it is necessary to add the device data to memory data, MD DIR is left high and the BREAK DATA CONT line is grounded. For output transfers, MD DIR must be grounded. The MB register is automatically loaded every TP2. A summary of the transfer types and the signals required by the data break control are summarized in table 9-7.

Table 9-7 Data Break Control Signals

TYPE OF XFER	MD DIR	BREAK DATA CONT	INFO ON DATA BUS
DEVICE→MEMORY	H	H	DEVICE INFO
MEMORY→DEVICE*	L*	X	X
	H	L	0
MEMORY PLUS DEVICE→MEMORY	H	L	DEVICE INFO

* PREFERRED METHOD

DATA BREAK INTERNAL LOGIC AND TIMING—Refer to the example provided in this section for the internal logic and break timing.

BASIC ONE-CYCLE DATA BREAK INTERFACE

The basic one-cycle break interface required to transfer data consists of a Break Memory Address Register (BKMA) to address memory independently of the processor; a Break Priority Network to assure the activation of the device with the highest priority; Input/Output buffers and Break Control Logic. A sample data break interface is illustrated in figure 9-16. The data break sequence of events are described in terms of the primary data break control signals and the processor timing is given in table 9-8

Table 9-8 One-cycle Data Break Sequence of Events

DATA BREAK EVENT	PROCESSOR TIMING	DESCRIPTION
BREAK REQUEST	Any time. Sampled by the leading edge of INT STROBE	Signal BREAK REQUEST is developed by the device any time a input or output transfer is to be made. It is loaded into a New Break (NBR) flip-flop by INTERRUPT STROBE, sets the flip-flop, and causes the start of a series of events leading to data break transfers.
ADD TO MEMORY	Any time. Must be asserted not later than TP1.	Signal ADD TO MEMORY is generated at the same time as BREAK REQUEST whenever data is to be transferred into memory. It is loaded into the ADM flip-flop by TP1.
DATA IN	TP3 Any time. Must be asserted not later than TP1.	Signal DATA IN is enabled only when the data transfer is to memory and at the same time as BREAK REQUEST. It is loaded into a flip-flop by TP1.
INT STROBE L		The following signals are generated as the result of INT STROBE loading the NBR: a) BK IN PROG L (IF BREAK REQUEST) b) CPMA DIS L (IF BREAK REQUEST) c) DEVICE PRIORITY L (IF BREAK REQUEST)

Table 9-8 One-cycle Data Break Sequence of Events (Cont.)

DATA BREAK EVENT	PROCESSOR TIMING	DESCRIPTION
BREAK PRIORITY	TS4	Since each device priority was applied to the DATA BUS at TP3, all priorities are tested during TS4. With the sample data break interface having a 3rd priority, signal MY PRIORITY is developed if DATA 0 and DATA 1 are high. The condition of MY PRIORITY L and NBR (0) L will cause the MA CONTROL flip-flop to set at TP4.
BREAK ADDRESS	TP4	The break address is supplied by the data break device. The contents of the Address lines are loaded into the BKMA by TP4 and the 1 output of the MA CONT is used to gate the Break Address onto the MA lines.
PROCESSOR DMA STATE	TP4	The designer should watch the propagation delays of circuits so that not more than 50ns elapses between the start of TP4 and the arrival of MAC(1) to the BKMA output gates. When the MA CONT flip-flop is set, signal MS, IR DIS is grounded. This disconnects the outputs of the processor's Major State and Instruction registers and thereby causes the processor to enter into the DMA state. Signal BK CYCLE is also grounded. If the transfer direction is from memory to the device, MD DIR is grounded at TP1. If the transfer direction is from the device to memory, BREAK DATA CONT is grounded at TP1.

Table 9-8 One-cycle Data Break Sequence of Events (cont)

DATA BREAK EVENT	PROCESSOR TIMING	DESCRIPTION
INHIBIT MS & MA register loading	TP1	When MA CONT is set, the MALC is loaded at TP1. This grounds the MA, MS LOAD CONT line which prevents the MS and the MA registers from being loaded. The processor is now conditioned so that data break transfers will in no way affect the previous or the next processor instruction. At TP1 the ADM and/or OUT flip-flops are loaded to control the type of data transfer. The break request may be cleared by TP1. This allows the MA control flip-flop to be set at TP4.
INPUT TRANSFER	TS2	Device data is gated in by DATA IN (0) L and applied to the DATA BUS by DATA EN H and TS2 L.
OUTPUT TRANSFER	TP3	Memory data is gated into the data break interface when a DATA IN L signal is present and loaded into the input buffer by TP3.
NEXT WORD	TP3	At TP3 of the Data Break Cycle, signal INT STROBE L is again generated in the processor. If signal BREAK REQUEST is asserted at this time indicating that another data word is to be transferred, the break priorities will again be tested during TS4 and a new break address will be applied to the MA lines at TP4. Otherwise, those signals that disabled the processor during the last break cycle will be negated and the processor continues with the current instruction.

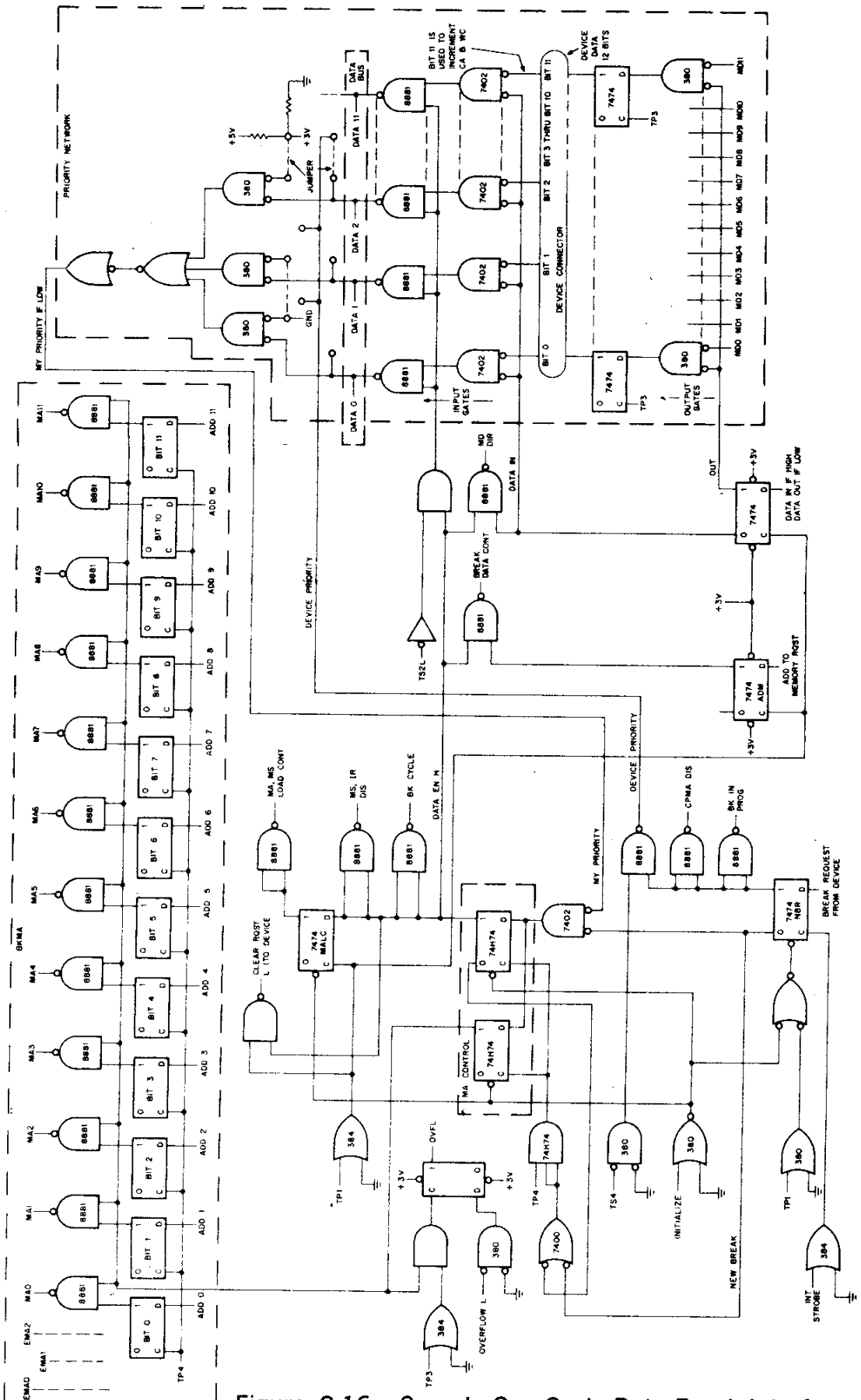


Figure 9-16 Sample One-Cycle Data Break Interface

Timing for Sample Data Break Interface—The data break control timing with respect to the processor timing is illustrated in figure 9-17. The diagram illustrates 2 complete processor cycles and a portion of a third cycle. For the first cycle, only that portion beginning with TP3 is of interest. This is the time required by the data break device to assert the control signals necessary to halt the processor, address memory, and be ready for input or output transfer. If there are to be no additional transfers, the break control signals are negated at the end of the break cycle as shown on the diagram. Otherwise, the break control signals will continue into the next cycle.

THREE-CYCLE DATA BREAKS

All of the previous information has dealt with one-cycle breaks. Three-cycle breaks consist of three break cycles in succession, the first two of which are used to control word count and current address. See Chapter 6 for a detailed discussion of three-cycle data break theory.

The data break hardware for a three-cycle break is more complicated than that for a one-cycle data break. In addition to the normal data break equipment, the three-cycle control requires internal major state control to accommodate word count, current address and data transfer cycles. A means for loading the BKMA register from the MD lines during the current address cycle must also be provided. Priority must be checked three times: once each before WC, CA and B cycles in order to allow higher priority devices access to memory in the minimum amount of time.

The hardware implementation for a three-cycle break can be accomplished using the previous one-cycle break example and the flow diagram for a three-cycle break illustrated in figure 9-18 as a guide.

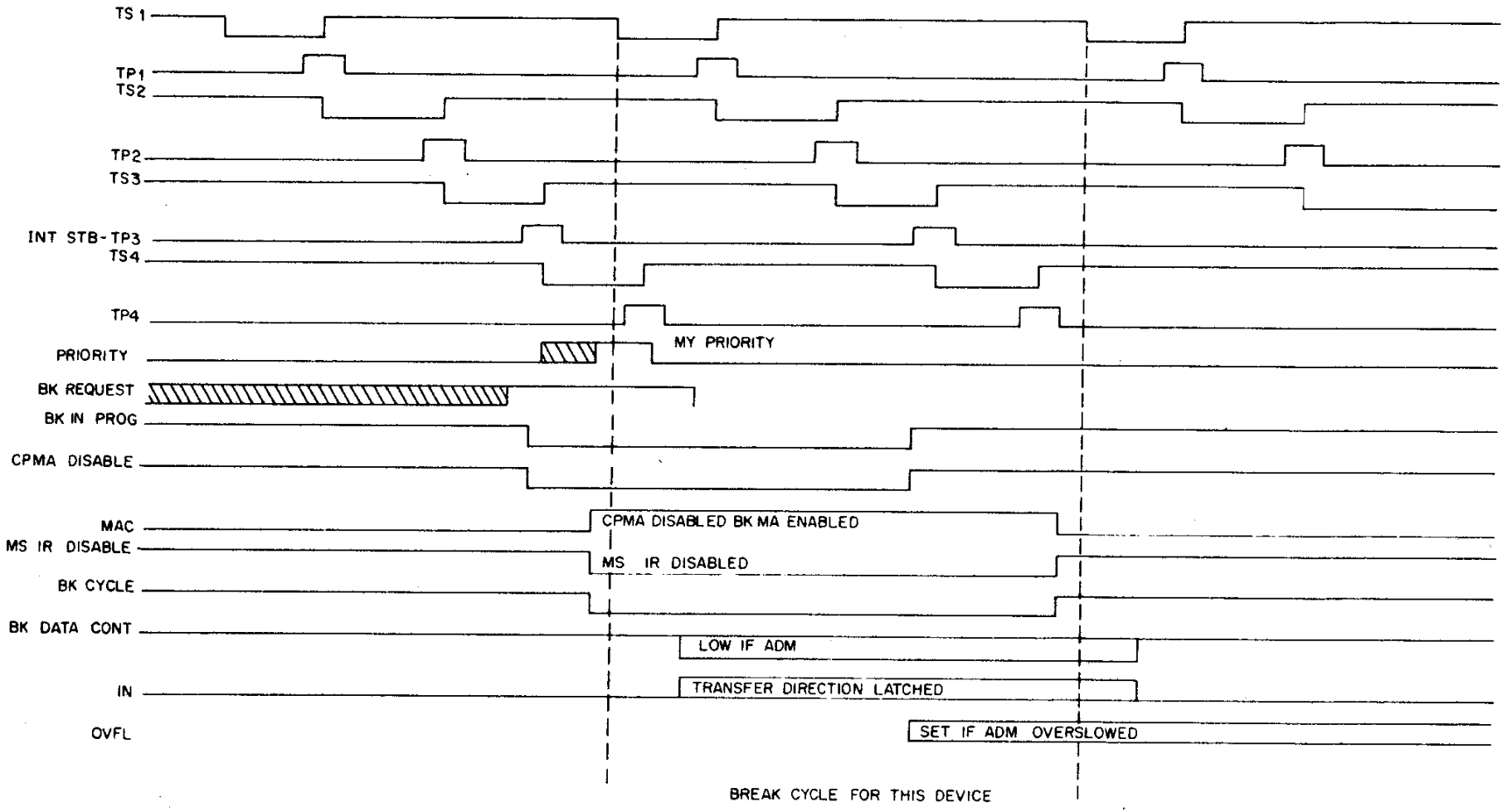


Figure 9-17 Data Break Control Timing Diagram

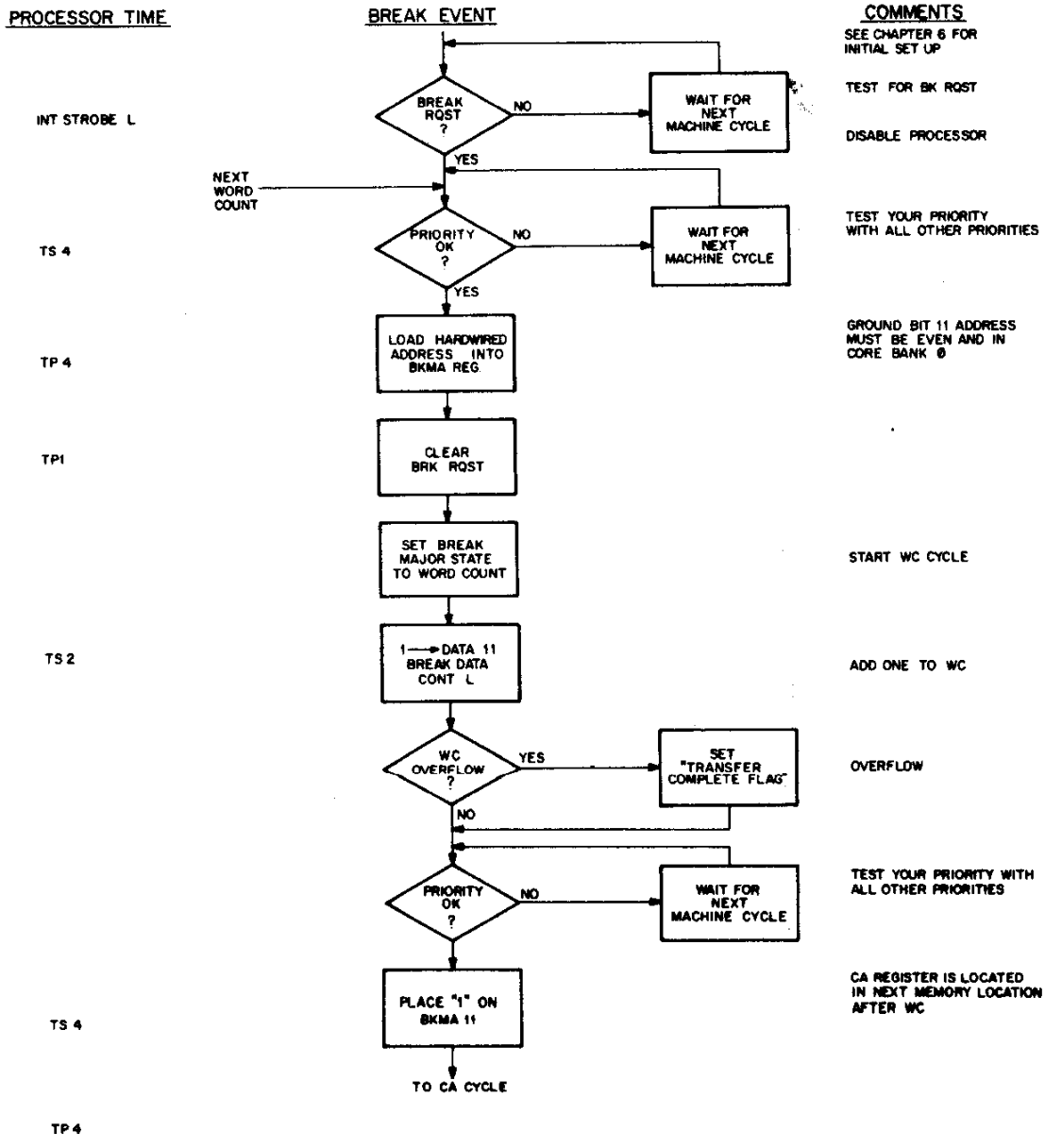


Figure 9-18 3-Cycle Data Break Implementation Flow Diagram

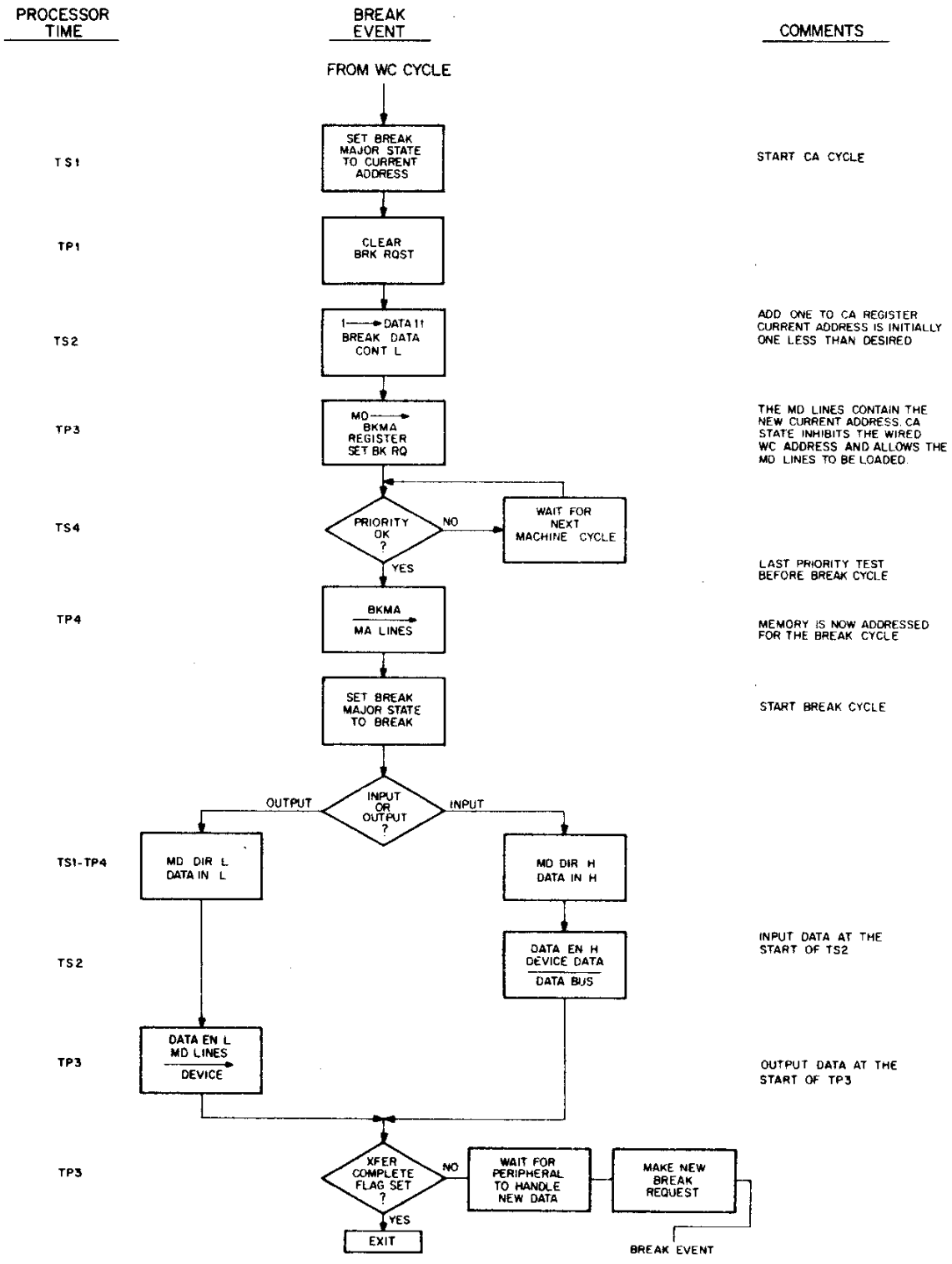


Figure 9-18 3-Cycle Data Break Implementation Flow Diagram (continued)

DESIGN CHECK LIST FOR SINGLE CYCLE DATA BREAK INTERFACE

The following information summarizes the important design considerations of a single cycle Data Break Interface.

PDP-8/E CONTROL LINES

a. To ground BK IN PROG and CPMA DIS, use the leading edge of INT STROBE H. ☐

b. Ground the bit on the DATA BUS corresponding to its priority while examining all other priority bits only during the TS4 Time Period. ☐

c. At the leading edge of TP4, provided all higher-order PRIORITY bits on the DATA BUS are high: ☐

   1. Place the break address on the MA lines within 50ns after the leading edge of TP4,
   2. Ground BREAK CYCLE,
   3. Ground MS, IR DIS,

d. At the leading edge of TP1; if all conditions within C above were met: ☐

   1. Ground MA, MS LOAD CONTROL,
   2. Set the ADM flip-flop and ground BK DATA CONT and/or MD DIR to establish the data transfer path,

NOTE

Control lines are generally negated in the same order in which they were asserted.

DATA TRANSFER PATHS

a. For data input transfers or add to memory:
   Place input or modifying data on the 12-bit DATA BUS for the duration of TS2. ☐

b. For Data output transfers:
   Use a Time Pulse to gate data from the output buffer to the device. Data is available on the MD lines at TP2, TP3, or TP4 time. ☐

c. Overflow:
   If a modification of memory was made, the OVERFLOW line will be low during TS3. Sample this line with TP3 if you wish. ☐

SECTION 5
GENERAL DESIGN & CONSTRUCTION GUIDELINES

INTERFACE DESIGN OPTIONS

Basically, the user has two options in designing his interface. One option is to build an interface module to the external bus, and the other is to build an interface module to the OMNIBUS.

External bus interfacing allows the designer to deal with the wire-wrap system, which by definition is easier for him to alter. The user then does not have to be concerned with the rigid pin assignment imposed upon the OMNIBUS. Chapter 10 explains how to do such interfacing.

Interfacing to the OMNIBUS is simple and direct, providing that the designer conforms to the bus pin assignment. He has several options in selecting the type of module that he wishes to place on the bus. He may, for instance, construct a wire-wrap assembly and place it at the far end of the OMNIBUS. This assembly can be connected directly to the OMNIBUS. There is enough room to place a fairly complicated controller with wire-wrap pins. The restriction is, of course, that the pin assignment must conform to the OMNIBUS. The user may purchase a single quad board containing wire-wrap pins and IC sockets from DEC. Upon this board he may easily construct any type of interface to his specifications.

Another method that provides the highest density of components is the use of an etched board. The user can build blank boards and purchase from DEC most of the necessary IC's, connectors and cables. The advantages of this approach are fully realized when large numbers of duplicate interfaces are to be made. The cost of building an etched board is, of all methods, the lowest—provided that enough interfaces are to be constructed so the designer can recoup his rather high initial engineering costs.

Board Layout

When connecting the +5V supply, the designer should try to split up the runs into three separate parts, and connect each run to a different pin. Then, if there is a short somewhere from +5V to ground, it is three times as easy to find. A good rule to follow is to limit the number of IC's to be mounted on any one board to 50. If the designer has a requirement for more than 50 IC's, he should consider using a second board. A typical layout of components is shown in Figure 9-19.

Uninsulated components should not project more than $\frac{3}{8}$ in. above the board, insulated components should not project more than $\frac{11}{32}$ in. above the board. The grounding scheme should be carefully planned. Pins C, F, N, and T (except AC1) provide the ground lines needed to operate the board. It is good practice to use as many ground lines as possible; connect all ground pins together, and make runs as short as possible to the ground pins of the ICs. Poor grounding can cause occasional annoying malfunctions.

Etched Circuit Layout and Construction Rules

The following layout and construction rules should be used as a guide to assure optimum performance of the interface module:

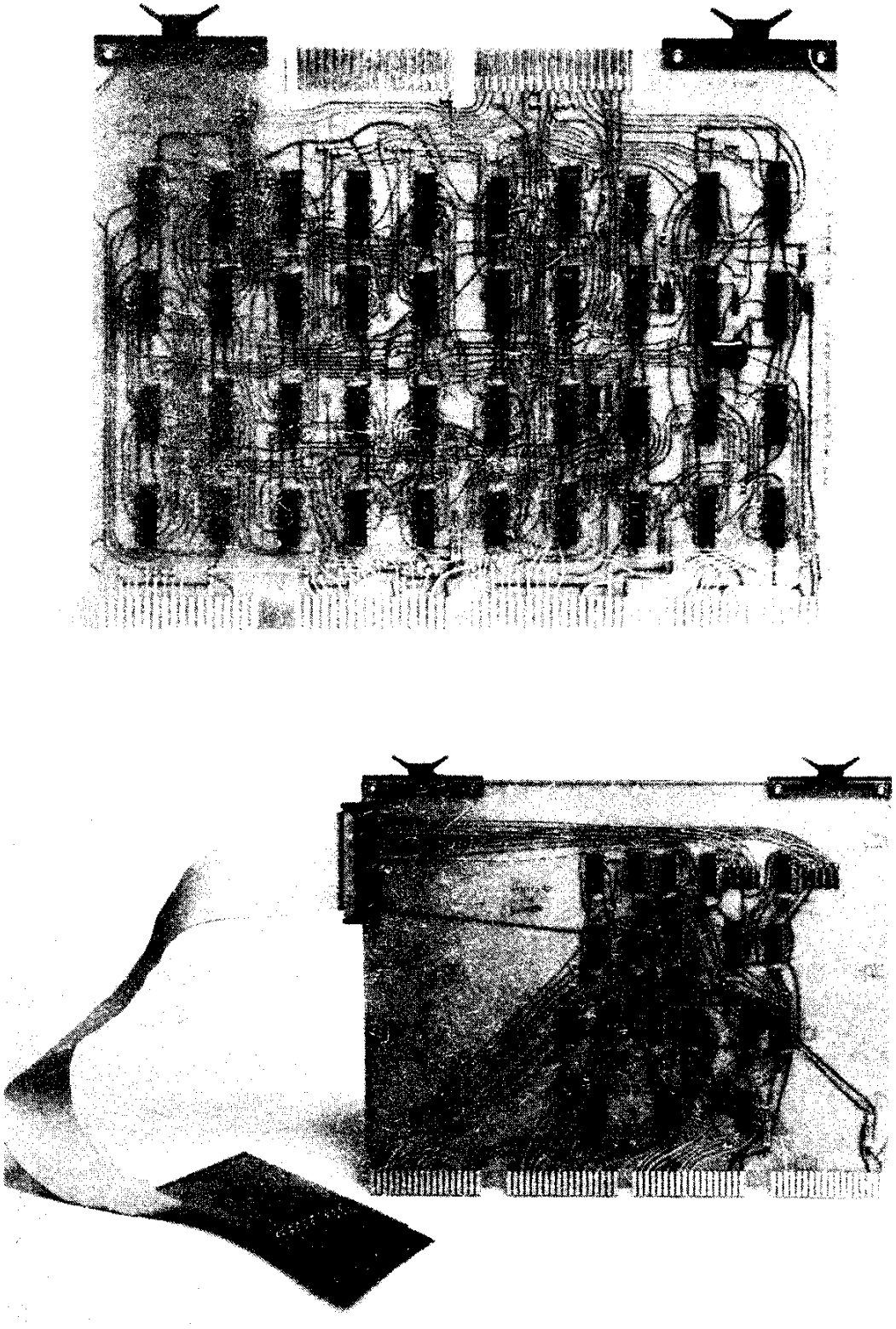


Figure 9-19 Typical Component Layout

General Cable Rules and Suggestions

Cabling is an important consideration when designing and constructing an interface control module that plugs into the OMNIBUS. The designer's only concern is that his cabling is adequate between his peripheral and the interface control module that he is designing to plug into the OMNIBUS. No additional cabling is required. The interface cable connects to a 40-pin connector type H854 shown mounted on the upper left side of the module. The peripheral connector is a 40-pin type connector. A channel "cut-away" along the length of the power supply allows bundling of many cables and allows them to be clamped onto the side of the channel.

DEC Supplied Interface Cables

The user may purchase from DEC the necessary interface cables. Two standard lengths are provided—a six ft. cable (part number BC08J-6) and a ten ft. cable (part number BC08J-10). Each cable contains a 40 pin connector type H856, which connects to the interface module and a 36-pin module type M953 which in turn connects to the users peripheral.

Each cable provides 18 signal lines and 22 ground lines. A ground line follows each signal line with several ground lines on each end of the cable. This arrangement is illustrated in Figure 9-20.

The design is intended to provide an electrical shield between lines, and to provide adequate grounding between the two ends of the cable. The use of ribbon cable is convenient, saving space and eliminating bundle problems. However, round coaxial cabling can be used. The coaxial cable then has to be larger because it consists of several layers of conductor and insulator material. Where the environment is considered hostile (such as may be the case in a factory), coaxial cabling is recommended. However, in most cases, the flat cable will prove adequate.

Cabling Rules

There are cabling precautions that the designer of any interface control module should follow. These include:

a. Do not run the AC line immediately adjacent to a low level signal.

b. Do not attempt to drive a line directly with the output of a flip-flop. The flip-flop may be triggered by noise being sent back along the line.

c. Tie all grounds together at the board and at the far end of the cable.

d. If there are two or more cables running parallel, there must be an intervening electrostatic shield.

e. Use as low an impedance as possible, but not lower than 100 ohms on the lines, and terminate the lines at the far end to eliminate ring; or drive them with a higher impedance and wait for them to settle down.

Module M953 Pins		Module H856 Pins	
Signal	Ground	Signal	Ground
	A1		A, B, C, E
B1		D	
	C1		P, S
D1		J	
E1		N	
	F1		U
H1		T	
J1		X	
	K1		Y, AA
L1		BB	
M1		FF	
	N1		CC
P1		LL	
	R1		HH, KK, MM
S1		RR	
	T1		
*U1			
*V1			
*A2			
*B2			
	C2		SS
D2		F	
E2		L	
	F2		H, K
H2		R	
	J2		M
K2		V	
	L2		W
M2		Z	
	N2		EE
P2		DD	
	R2		PP
S2		JJ	
T2		NN	
	U2		UU, VV
V2		TT	

* not used

Figure 9-20 Interface Cable Pin Assignment

If the user requires a more complicated interface controller that requires two boards, a connector type H851 is used to interconnect the signals from one board to the next. This is shown in Figure 9-2. The connector receives 36 etched finger type pins from both modules and slides onto both modules. Pin A1 connects to pin A1, etc.

INTERFACE TIMING CRITERIA

In nearly all instances, the user need not concern himself with the details of timing. This section is included to assist the person with an exceptional case.

There are basically three types of timing with which the user may be concerned. First, there is the data exchange time between the computer itself and the peripheral (a function of the I/O structure of the machine). Secondly, there is data break timing, which is a function of the break priority system and the data break peripherals. The third consideration is Interrupt timing.

General Timing Rules

General timing rules are as follows:

a. If maximum machine speed is necessary, do not use the positive I/O Bus interface, type KA8-E. Instead, design all peripherals so that they plug directly into the OMNIBUS.
b. When the above is not feasible, do not microcode IOTs. For example, replace the KRB instruction with its equivalent KCC and KRS instruction. The result is a slightly longer time for the overall IOT; however, the processor is stopped for two shorter periods of time, rather than one long period of time. Hence, the data break and interrupt systems have access to the processor and memory sooner.

Interrupt Timing

Interrupt timing is concerned with the interval from the time that a flag is seen until the time it can be serviced. Although this time is fairly easily calculated, the exact details depend upon the number of flags the processor checks before it finds a flag that is set.

Timing Example:

An interrupt request is asserted by one of the control modules. The processor must finish the current instruction upon which it is operating, possibly a TAD indirect through an auto index register. TAD indirect through an auto index register takes 1.2 microseconds plus 2 times 1.4 microseconds, or 4.0 microseconds total. In addition, the processor might not have seen the flag as much as 300 ns before that time, so it is a maximum of 4.3 microseconds from the time that the flag got set until the time the processor can start executing the JMS to location zero. This assumes no EAE option and that no device is using the break facility at that time. Thus, up to this time, it has taken the processor 4.3 microseconds to recognize that there is a flag to be serviced. This example assumes that the KM8-E Memory Extension control is installed.

TIMING EXAMPLE

TAD I	4.0 μs	Through an Auto Index register
	0.3 μs	Worst case time for processor to see flag
(JMS)	1.4 μs	Jump to location 0, store PC
DCA AC	2.6 μs	Store AC and Link
GTF	1.2 μs	Pick up flags in memory extension control unit
DCA FLAGS	2.6 μs	Store flags
MQA	1.2 μs	Load AC with MQ
DCA MQ	2.6 μs	Store MQ
JMP I .+1	2.2 μs	Jump somewhere to Interrupt flag scan
IOT FLAG SKIP	1.2 μs	Skip on flag—Start of Flag Scan Routine
IOT	1.2 μs	Perform I/O Transfer
Total	20.5 μs	Total time required to get to IOT and perform instruction.
DCA I 10	4.0 μs	DCA indirect through some auto index register
CLR FLAG	1.2 μs	
ISZ	2.6 μs	Bookkeeping—To see if last transfer
JMP	1.2 μs	Dismiss
TAD MQ	2.6 μs	Restore MQ
MQL	1.2 μs	Transfer the contents of AC into MQ
TAD FLAGS	2.6 μs	Check status of flags
RTF	1.2 μs	Restore flags
TAD AC	2.6 μs	Restore AC
JMP I O	2.4 μs	Go back to location 0
Total	21.6 μs	Total time required to get back to main program again

Thus 20.5 μs + 21.6 μs = 42.1 μs total for this example.

The interrupt timing requirements are a function of the amount of coding that it takes to determine what has to be done and how much time is available to do it. There are two times that must always be considered. The first is the length of time from the time of the interrupt request until the source of the interrupt request has been recognized and the data retrieved. The second time is the length of time it takes to finish processing the data, including all of the housekeeping routines and the time it takes to restore the major registers to their original state.

A special instruction, SRQ (octal 6003), allows the programmer to test for possible interrupts before actually restoring the machine. One may save considerable time entering and exiting the interrupt program by merely returning to the flag scan routine, if the SRQ instruction indicates the presence of a second interrupt.

The program interrupt system is satisfactory for data rates less than 10 KHz (one word every 100 microseconds). Above this rate, the user should examine each individual case for its merits and decide whether or not he should use the data break facility or possibly tie the machine up with high data rates.

Timing Requirements for Data Break Facilities
The important timing consideration in data break as in program interrupt, is whether sufficient time is available from the time the flag (in this case, the break request) is set to the time all the data is moved in or out of memory. The first item in question is the period of time before operation actually starts on the break request. The break request timing for the PDP-8/E has improved considerably in that a break request can be honored between major states of an instruction, whereas in the older model machines, the break system had to wait until an instruction was completely processed. The break system is synchronized 300 nanoseconds before the end of every memory cycle. At the same time the processor tests for the possibility of interrupt, it tests for the possibility of break. Unlike the case of an interrupt, the break system need not wait until the processor is ready to go into a FETCH. If the processor is programmed to do only machine instructions (assuming no EAE or external I/O), it would take the processor no more than 1.7 microseconds to begin servicing the break request. For the external bus, this time would be 4.9 microseconds, maximum. The choice of one-cycle or three-cycle breaks is an important timing consideration. Basically, the internal operation of the PDP-8/E is such that it readily adapts to one cycle break. Users who are planning on constructing their own break device should think in terms of one cycle break.

Timing and Break Priorities—Timing problems result when the break priority system has not been carefully planned. Suppose, for example, the highest speed break device is a complex parallel disk that serves up a word once every 5 microseconds. Suppose two break devices request a break simultaneously (e.g., the disk and a slower device such as the DECtape that has 50 microseconds for which the word is available). If the priorities of these two devices are not correct (i.e., DECtape assigned first priority), timing problems are inevitable. If the DECtape and disk simultaneously request a break, both devices must wait 1.7 microseconds for the current memory cycle to finish. The DECtape then makes a 3-cycle break request since it (incorrectly) has the highest priority.

Therefore, the waiting time of the disk is 3 times 1.4 microseconds or 4.2 microseconds plus 1.7 microseconds (a total of 5.9 microseconds), to service the DECtape. The result is that the DECtape was serviced and the disc has lost its data because 5 microseconds have expired. This is an obvious situation where the disk must have a higher priority. Other situations may not be obvious without examining the timing requirements before assigning a priority to each device. As a general rule, the user should set up his priority based upon the device that has data available for the shortest amount of time.

GENERAL PROPAGATION DELAY GUIDELINES
When designing an interface module, the designer should consider the individual propagation delays of such logic elements as NAND gates, flip-flops, J-K flip-flops, one shot delays, etc. He should add each delay in a logic chain to determine the overall delay of his module.

2 Input NAND Gate Delay

Typical characteristics of a NAND gate used with the PDP-8/E logic are illustrated in Figure 9-21. Where high fan out is required, a SN7440 type gate is preferred.

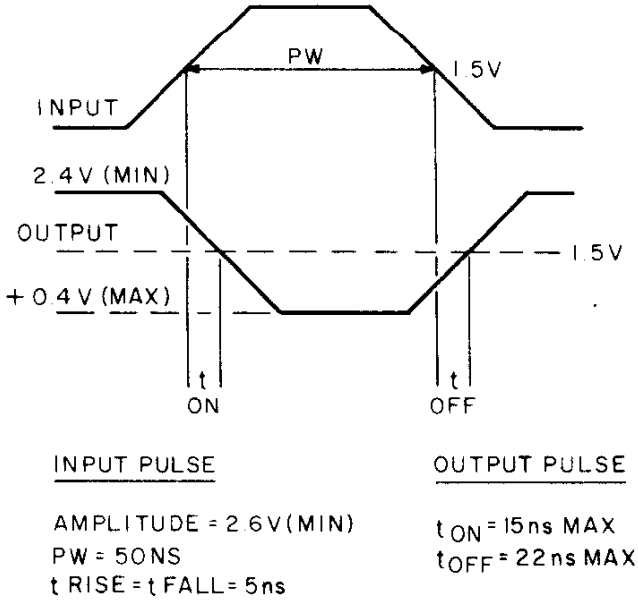


Figure 9-21 2 Input NAND Gate Typical Characteristics Example

Flip-Flop Propagation Delays

Typical D-type flip-flops trigger on the leading or rising edge of a positive clock pulse; the propagation delay is measured from the threshold point of this edge. The set-up time of the flop is also measured from this threshold point. Data on the input must be settled at least 20 nanoseconds prior to the clock transition.

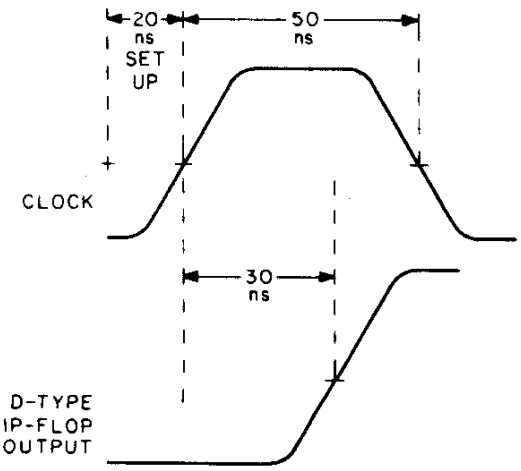


Figure 9-22 Typical D-Type Flip-Flop Timing Example

J-K flip-flops

J-K type flip-flops are, in effect, trailing edge triggering devices as explained previously. The only restriction on the J and K inputs is that they must be settled by the time that the rising edge occurs. Timing is shown in Figure 9-23.

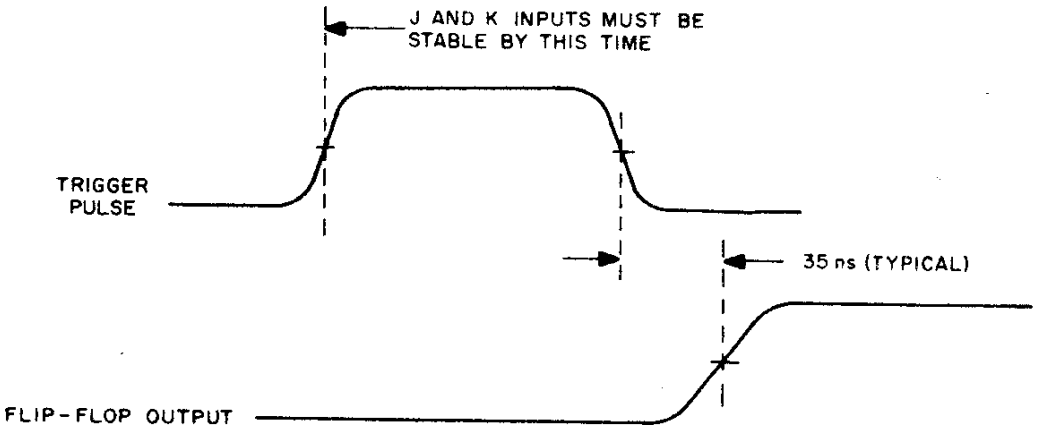


Figure 9-23. J-K Flip-Flop Timing Example

When using the dc Set or Reset inputs of either flip-flop type, propagation delays are referenced to the falling edge of the pulse. This is due to the inverted sense of these inputs. When resetting ripple type counters (where the output of one flip-flop is used as the trigger input to the next stage), the Reset pulse must be longer than the maximum propagation delay of a single stage. This ensures that a slow flip-flop does not introduce a false transition, which could ripple through and result in an erroneous count.

One-Shot Delay—Calibrated time delays of adjustable duration are generated by the Delay Multivibrator such as the M302. When triggered by a level change from a logical one to a logical zero, this module produces a positive output pulse that is adjustable in duration from 50 to 750 ns with no added capacitance. Delays up to 7.5 milliseconds are possible without external capacitance. Basic timing and the logic symbol are shown in Figure 9-24. The 100 picofarad internal capacitance produces a recovery time of 30 ns. Recovery time with additional capacitance can be calculated using the formula:

$$t(r)\ \text{Nanoseconds} = 100.3\ C\ \text{Total (Picofarands)}$$

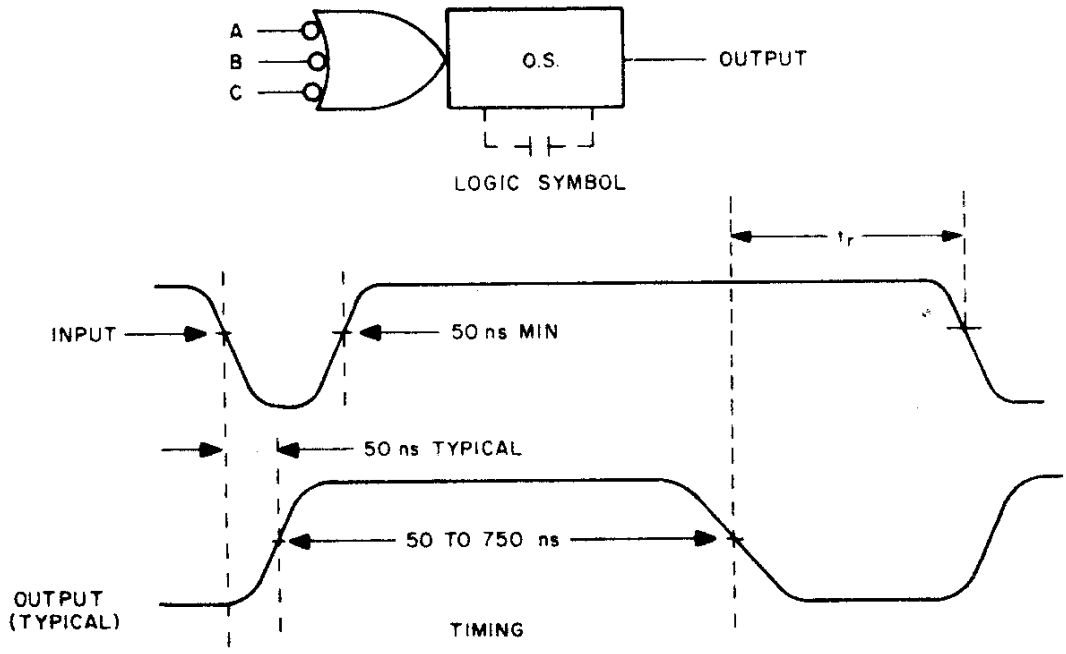


Figure 9-24 One-Shot Delay Timing and Logic Symbol Example

Maximum Operating Frequency

Once the designer has determined the individual propagation delays in each logic element, he must then add these delays corresponding to a simple logic chain. He then compares the results with the system frequency to assure that his logic circuit can meet the requirements imposed by the system frequency. Figure 9-25 illustrates a situation in which various logic components in a given chain are examined and all delays are added. The following assumptions are made:

a. A standard clock pulse width of 50 nanoseconds is assumed. This period is measured from the threshold point of the leading edge to the threshold point of the trailing edge.
b. One flip-flop propagation delay of 35 nanoseconds from the trailing edge of the clock pulse to the threshold point of the final state of the flip-flop is allowed.
c. Two gate-pair delays of 30 ns each are assumed. (A gate-pair consists of two inverting gates in series.) Two gate-pair delays are usually required to perform a significant logic function with a minimum of parallel operations. The two gate-pair delays total 60 ns.

The time necessary to perform these operations before the next occurrence of the clock pulse is the sum of the delays; 50 + 35 + 60, or 145 nanoseconds. Allowing 20 ns for variations within the system, the resulting period is 165 ns, corresponding to a 6 MHz clock rate.

Note that the D-type flip-flop triggers on the leading edge of the clock pulse and the J-K flip-flop triggers on the trailing edge. When calculating system timing using flip-flops, remember that the flip-flop inputs must be settled at least 20 nanoseconds prior to the occurrence of the clock pulse.

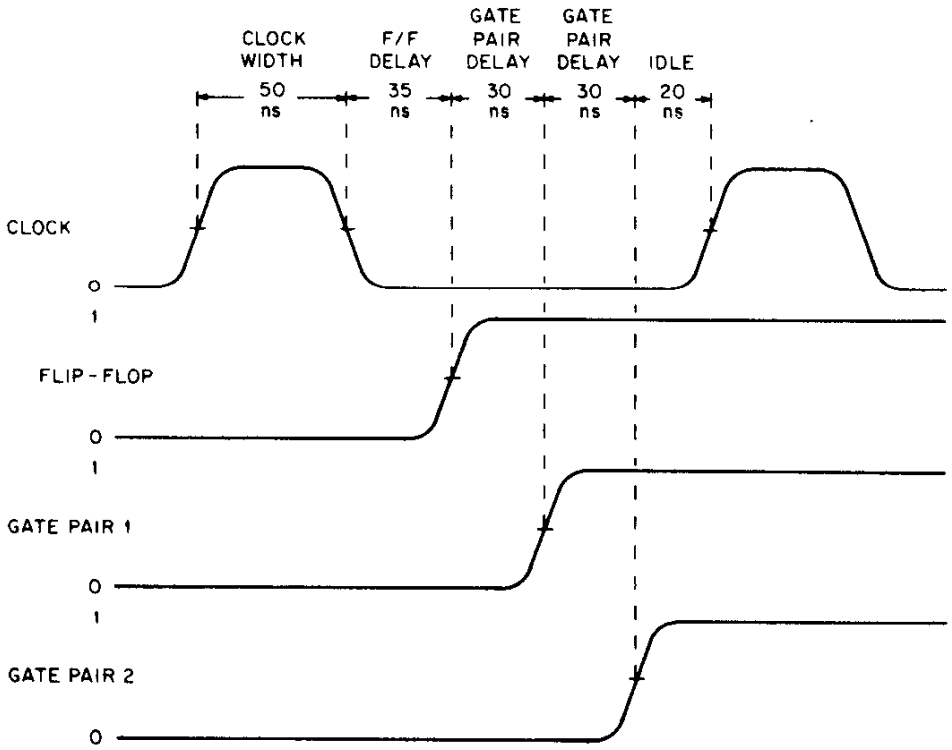


Figure 9-25 **Delays Determining Total Operating Frequency Example**

The preparation of a timing diagram that considers delays introduced by all logic elements aids the designer in achieving predictable system performance. Don't forget that minimum (including zero) delays are possible, and that no good design should rely on a finite gate delay. Using a similar approach, the designer must ensure that his selection and operation decoding scheme works within the allotted time, and that data can be made available in time for TP3 to either strobe the data in or strobe the data out, depending upon the type of transfer. I/O PAUSE is used to gate the device and operation decoders and TP3 strobes the data either into the OMNIBUS or onto the data lines to the peripheral. A total of 280 nanoseconds is allowed to bring the contents of either the input or output data line into the interface controller. All input signals to the processor must be established at the processor at least 130 nanoseconds before TP3. This discussion assumes either an I/O transfer or a programmed Interrupt. With data break devices, the preceding information does not apply.

LOADING RULES

Almost all signal lines on the OMNIBUS are driven positive by a load resistor (refer to Figure 9-6) and pulled to ground by open collector ICs. The designer should use a gate such as the DEC380, DEC314, DEC348 ICs or their equivalent as an input device. These gates have a fairly high input impedance and, therefore, do not load down the bus. Wherever possible, one of the input of a gate should be held positive by I/O PAUSE or some other logic signal in order to further reduce the input load. Conversely, to drive output lines (such as the bus lines) the user should consider DEC8881 or its equivalent (paying careful attention to the leakage current).

Device Selection Inputs

When designing the device selection decoder, a NAND gate such as the DEC380, which is a two-input gate, should be strobed by I/O PAUSE. This helps to remove the load from the MD lines.

Skip and Interrupt Request Lines

As a general rule, the DEC8881 gate outputs should be limited to one per device code for skip line and one per device code for the Interrupt Request Line. A potential problem exists when too many gates are tied into these lines.

Electrical Considerations of Driving a Line

Most signal lines on the OMNIBUS are tied through a load register to +5V. Users who want to look at any one line must do so with the DEC380 gate. Users desiring to drive any bus line to ground must do so with the output of the DEC8881 gate. The limitation of the amount this gate can carry must be considered. A major factor is the output leakage, as no switch is a perfect open circuit nor a perfect short circuit. This is a fundamental limitation. Tolerance for ground level should be considered up to .4V, as defined by the TTL logic. The high signal must be a minimum of 2.6V; however, 3.0V is recommended. An additional consideration is that the DEC380 requires some current at its input.

GROUNDING

Pins C,F,N, and T on the OMNIBUS are used for ground signals. The user who is making modules that are designed to plug into the OMNIBUS should utilize all of the ground pins and tie all of the ground pins together. He should make the connecting lines as short as possible. The user should also attempt to keep the leads from the ground pins to the ground terminals on the ICs as short as possible. The shorter the ground runs from the integrated circuits on the module to the ground pins and the more duplication there is (parallel paths), the quieter the ground system is within the module. The designer should pay careful attention to the recommendations of the ICs to ensure that good construction practices are followed. (Do not overlook the local bypass capacitors required at an IC.) The designer should use as much as possible a .01 microfarad ceramic disk capacitor across Vcc to ground for every IC used.

TESTING TECHNIQUES

When the interface module has been completely assembled and checked, the designer should perform an initial checkout and then proceed to test his interface in the system. This should be followed by a complete peripheral system integration.

Initial Checkout

The tester should remove the power source connections prior to performing his initial test. He should then make an electrical test with an ohmmeter from +5V to ground, from −15V to ground, and from +15V to ground. He must ensure that there are no power shorts prior to connecting power. A short can damage the etched circuits.

System Test

The next test step is to plug the module into the OMNIBUS and generate an IOT to check out the device selection capability, logic levels, opera-

tion of the flag, and capability of the interface controller to receive and transfer data. A combination of 1's and 0's may be placed in the AC and transferred to the interface. With an oscilloscope or voltmeter, the tester can check each of the connector terminals corresponding to each of the data bits to ensure that the right information is being transferred. Another useful test is to generate a count pattern in the AC and observe that bit 11 is moving twice as fast as bit 10, which is moving twice as fast as bit 9, and so on. These waveforms can be examined for each line. This test indicates shorts existing between data lines.

Final Testing
Before actual operation, the final test includes connecting the peripheral to the interface and transferring data into and out of the peripheral. For example, the 1's and 0's can be checked at the peripheral end for input to the peripheral and checked at the AC register for data transferred from the peripheral to the processor. Whenever possible, the programs used for testing should be collected into a Diagnostic Program for the device. A properly designed diagnostic, since it tests only one peripheral, is a powerful tool for finding system failures.

PROGRAMMING RULES

The most successful method of programming is to begin a program as simply as possible, test it, and then add to the program until it performs the required job. Before beginning the programming, the programmer should become familiar with the programs that he will be using. Refer to Chapter 4 for a description of the standard programs and refer to Appendix A for a complete list of the PDP-8/E compatible programs.
For best results, the programmer should avoid the use of the following device codes:

1. Devide code 0 (reserved for processor)
2. Device code 3
3. Device code 4
4. Avoid all codes in the 20 through 27 series (reserved for the extended memory control)
5. Avoid the Disk and DECtape device code series

Device codes 14-17 have been made available for the programmer's special use.

DESIGN CHECK LIST

When interfacing to the Omnibus, certain things must be done and others should be done. The following is a check list to summarize the requirements.

a. Omnibus Compatibility

1. In looking at the bus, do you use only 380, 384, 314 type IC's?
2. Are DEC8881's or 7438's selected for 25 μA leakage used to gate onto the EMA, MA, ROM ADDRESS and MS IR DISABLE lines? ☐
3. Are two DEC8881's in parallel used to gate onto Link Load and Bus Strobe? ☐

4. Are N8881, 8235 or 97401's used to gate all other signals onto the bus? ☐
5. Are all flip-flops receiving information from the bus leading edge triggered? ☐
6. When using direct clear or presets, do you gate a level (IOT) against TP3? ☐
7. Never gate TP3 and a data bit into a direct clear or preset. ☐
8. Never gate information onto the MD lines unless you're a memory. ☐

b. Timing

1. Is the path that pulls internal I/O less than 70 nsec. from pause? ☐
2. Are the paths that assert the "C" lines, data and skip, less than 100 nsec. from pause? ☐
3. Are the data, skip and "C" lines asserted by levels (IOT's), not pulses? ☐
4. If using long cycles is LAST XFER asserted at least 100 nsec. before TP3? ☐
5. If using long cycles, check the timing requirements in this chapter. ☐

c. Loading

1. Are MD 3-8 gated against pause? ☐
2. Are MD 9-10 gated against option select? (The decoded device code.) ☐
3. Are the Data Lines loaded only during an IOT? ☐
   One input to the receiver should be option select.
   Only one receiver per device code is allowed per bit.
4. Do you drive any 380, 384, 314 IC's with TTL? If so, do you have the proper pull up? ☐
5. Did you check the loading of long runs, such as Data Enables? ☐

d. Noise and Interference

1. Are all unused direct clear and presets tied off? ☐
2. Are all grounds used and tied together at both the front and back of the board? ☐
3. Are all signals to the OMNIBUS and control flip flops disabled by INITIALIZE? In some cases, such as magnetic recording, PWR OK may be preferred. ☐
4. Is there an .01 μf. capacitor across pwr at each IC? ☐
5. Are there 6.8 μf. capacitors between each +5V and ground? ☐
6. Does power and ground go to the correct pins on each IC? For instance, 380 power is pin 8, ground pin 1. ☐
7. Check Chapter 9 for lines which should not be used. ☐

e. For Convenience

1. Keep +5V runs separate (three runs). This makes finding shorts easier. ☐
2. Label all jumpers in etch. ☐

SECTION 6 PDP-8/E INTERFACE HARDWARE

The following Interface accessories are available to make interfacing to the OMNIBUS a simple task.

H9190 M935 Kit—contains the H9190 assembly with M Series connector blocks for standard M Series modules, power wiring harness, and power bus board. It includes M Series power bussing for all but the four slots in the first column. Also included are two M935 bus connectors. Four mounting spacers allow the H9190 to be easily mounted in the second half of an 8/E chassis.

H803 Connector Block—a high density, 8-slot connector block with wire wrap pins. This connector is designed to be used with M Series modules.

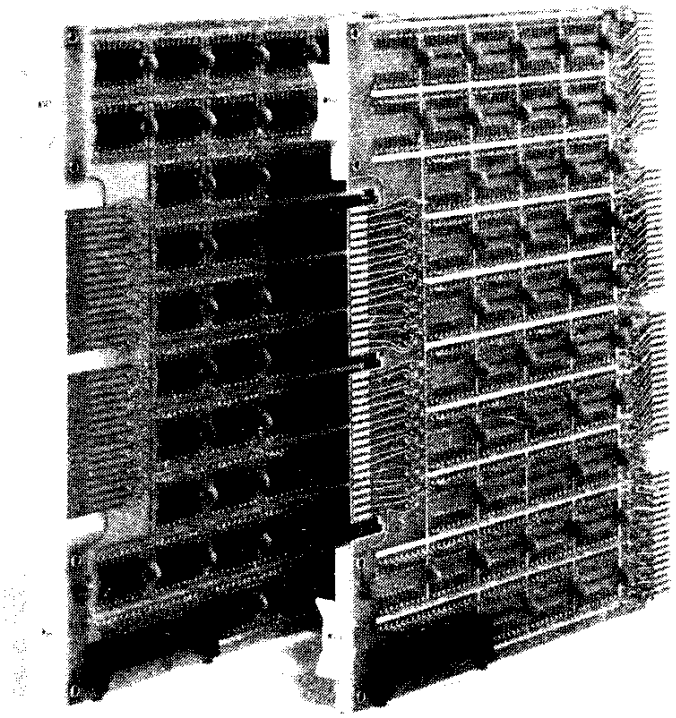
M935 Bus Connector—used to interconnect 8/E assemblies. The H9190 may be connected to the 8/E OMNIBUS using two M935's.

H9190 Mounting Panel—contains M Series connector blocks with 8/E-type packaging for standard M Series modules. Also included are the 8/E power wiring harness and power bus board. There is M Series power bussing for all but the four slots in the first column. Four mounting spacers allow the H9190 to be easily mounted in the second half of an 8/E chassis.

H019 Mounting Bar—an aluminum casting with the power bus board and power wiring harness. It also includes four mounting spacers for mounting in an 8/E chassis. Up to ten connector blocks of any type may be accommodated by this frame.

H811-A Hand Wire Wrapping Tool (pencil type, 30-gauge)

H812-A Hand Unwrapping Tool (pencil type, 30-gauge)



W966 W967

The W966 is the 8/E collage mounting board. It is double sided, extended length, and quad height with wire wrappable pins. It will accommodate 14- and/or 16- pin dual in-line IC's with or without 16-pin sockets. Two separate leads may be wire wrapped to each pin. Up to 42 IC's can be mounted on the W966. Discrete components may be directly soldered onto the board. The top center of the W966 board has 72 terminal fingers with terminating wire wrap pins. An I/O connector (male) terminating in wire wrap pins is mounted on the left side of the W966 board to provide access to the "outside world" when using BC08J-XX cable with a double sided connector board or a BC08K-XX single sided connector board. Both connector boards have 18 conductor lines.

All power and ground lines are common to the 8/E OMNIBUS.

AA2, BA2, CA2	+5
AC1, AC2, AF1, AF2, AN1, AN2, AT1, AT2 BC1, BC2, BF1, BF2, BN1, BN2, BT1, BT2 CC1, CC2, CF1, CF2, CN1, CN2, CT1, CT2 DC1, DC2, DF1, DF2, DNI, DN2, DT1, DT2	GND

The W967 is similar in all details to the W966 except that the W967 is supplied with 42 low profile IC sockets.

H851 Edge Connector
The H851 edge connector is used to bus signals from the top center terminal fingers to an adjacent quad board with similar terminals.

H852

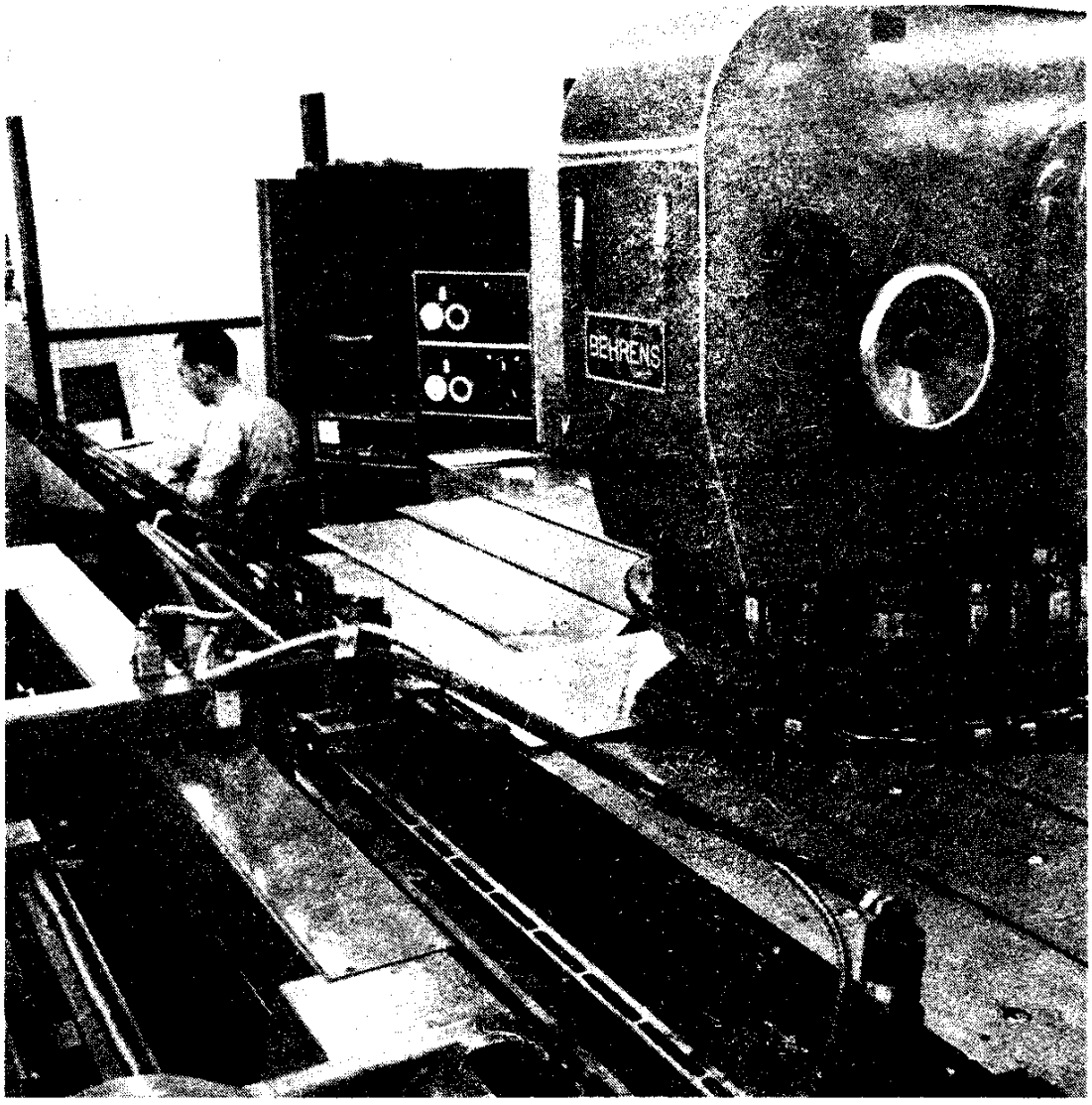
H853

H852, H853 Module Holders
When using two or more W940, W941, W942, W943, W950 or W951 boards in parallel in logic connector blocks, rigidity of the boards is maintained by using the H852 rib type holder between board handles 1 and 2, 3 and 4, and using the H853 non-rib type holder between board handles 2 and 3.

935—1000 foot roll, 30-gauge insulated wire.

For additional information consult the latest edition of

Digital's LOGIC HANDBOOK.



PDP-8 controlling a sheet metal punch.

CHAPTER 10

I/O EXPANSION TECHNIQUES

GENERAL

The degree of versatility of a computer is determined by the type and number of peripheral devices that can be interfaced with it. The PDP-8/E was designed with this idea in mind. Consequently, it can be easily interfaced with a variety of peripherals in a variety of methods.

Chapter 9 was concerned with interfacing directly to the OMNIBUS. However, the OMNIBUS was designed to allow other bus systems to be added and therefore providing a very flexible expansion capability.

DEC is now offering two additional basic techniques of receiving and sending data to the OMNIBUS. These are:

1. Using the Positive I/O Bus Interface option,
2. Using standard M series Bus Receivers and Bus Transmitters.

Section 1 deals with the Positive I/O Bus technique of interfacing to the OMNIBUS and Section 2 describes the method of interfacing using standard M Series Modules and companion hardware.

SECTION 1 POSITIVE I/O BUS INTERFACING TECHNIQUES

Previous discussions have brought out the fact that peripherals can be be interfaced with either the OMNIBUS or the external bus. This means that a PDP-8/E user can utilize not only devices designed exclusively for the PDP-8/E, but also devices originally designed for use with the PDP-8/I and PDP-8/L computers, and even devices of his own manufacture. This also means that the entire catalog of M and K Series modules may be applied to satisy any high speed and control application.

This section deals with the external bus, its applications, and the technique of interfacing peripherals to the bus. The user may wish to interface a DF32-D Disk File and Control unit with the PDP-8/E, for instance. This equipment was designed for the PDP-8/I and PDP-8/L; but, by interfacing to the external bus, it can also be used with the PDP-8/E. The user may want to transfer data between the PDP-8/E and a remote location. The external bus, which is designed to drive long interconnecting lines, is ideally suited for this application.

The first part of the chapter answers general questions about the external bus—What is it? How does it differ from the OMNIBUS? How does one use it? The remainder of the chapter guides the user through the initial uncertainties of interfacing by covering such topics as: types of connectors and cables to use with the external bus; timing criteria, loading rules, and voltage levels; and hardware and wiring techniques. DEC hope that this information will be helpful in designing and implementing any interface that the user might require.

Should questions arise regarding computer interface characteristics, the design of interfaces using DEC modules, or installation planning, customers are invited to telephone any of the DEC sales offices or the main plant in Maynard, Massachusetts. 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 licenses to use, manufacture, or sell equipment constructed in accordance therewith.

THE NATURE OF THE EXTERNAL BUS

What is the external bus? It is simply a number of signal lines (88, excluding grounds) that enable data transfers between the CP and peripherals. These lines carry data and control signals between the peripheral and two interface boards—the Positive I/O Bus interface (KA8-A) and the Data Break interface (KD8-E)—that plug into the PDP-8/E OMNIBUS. These two boards convert the internal bus signals into PDP-8/I and PDP-8/L-type bus signals. For instance, PDP-8/I peripherals need IOP pulses to perform instructions. The PDP-8/E does not generate internal IOP pulses, but it does provide signals (MD bits 09, 10, and 11) that can be converted into IOP pulses by the Positive I/O Bus interface. Other signals normally required by these peripherals are, in essence, available on the OMNIBUS. For example, BAC (buffered accumulator) bits must be supplied for the PDP-8/I peripherals. The PDP-8/E Data lines carry the necessary accumulator information. The Positive I/O Bus interface merely buffers the DATA bits and, thus, provides the external bus BAC signals.

Although the external bus consists of signal lines from both the positive I/O Bus interface and the Data Break interface, it is not always necessary to use both boards. When only programmed I/O transfer peripherals are used, the Positive I/O Bus interface provides all the necessary signals. However, if data break peripherals are to be connected, both interfaces must be used. Because each data break peripheral requires its own data break interface board, the number of signal lines comprising the bus may vary. There may be as many as 12 of these data break peripherals connected in the system, each contributing 36 signal lines to the external bus. Figure 10-1 illustrates the bus and its use when applied to a series of peripherals.

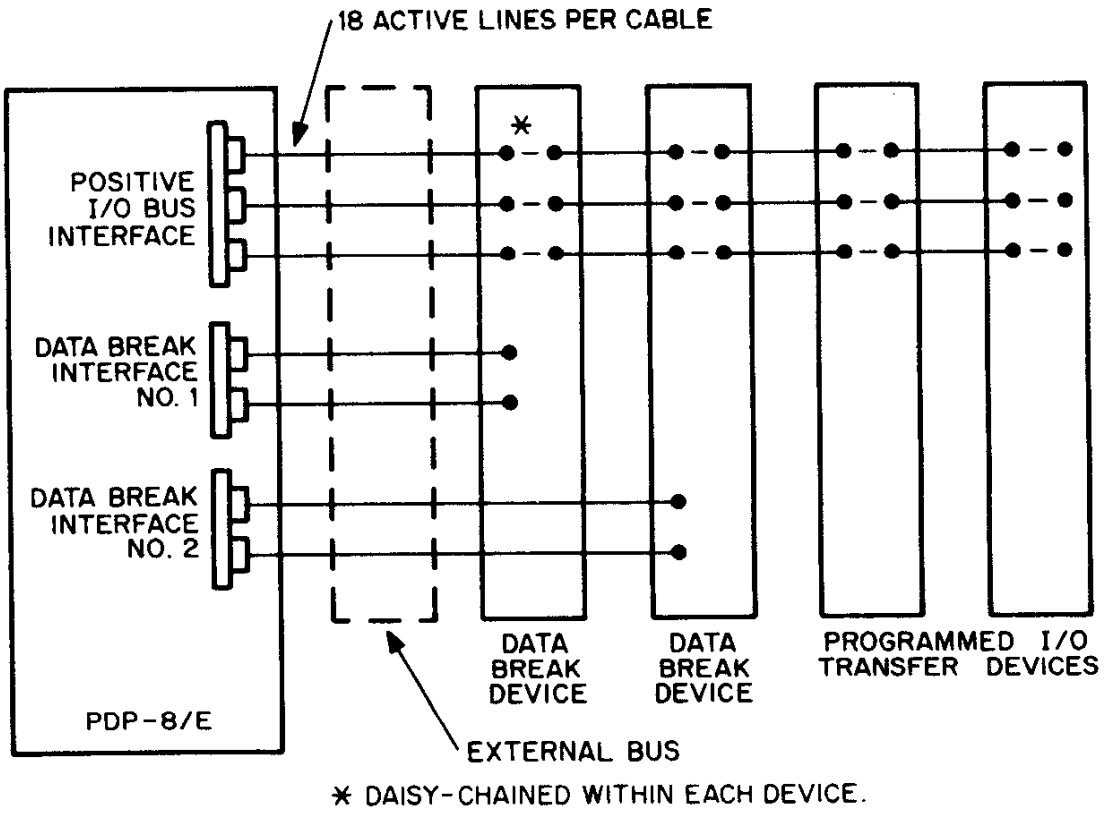


Figure 10-1 Parallel Connection of Peripherals

EXTERNAL BUS SIGNALS

Figure 10-2 shows not only the external bus signals, but also those OMNIBUS signals that are used by the two interfaces. Signal directions are shown for both buses. Some of the OMNIBUS signals—DATA 0-11, for instance—are common to both interfaces, but for clarity this commonality has been disregarded. The external bus signals are grouped according to the interface connector where they originate (Table 10-1 in paragraph 10.5 lists the bus signals and the connector and pin where each may be found). When similar signal lines are represented by one line of the drawing, as BAC 00-11, the actual number of lines is indicated in parentheses. The external bus signals are discussed in detail in the following section with emphasis on the relationship between these signals and the OMNIBUS signals.

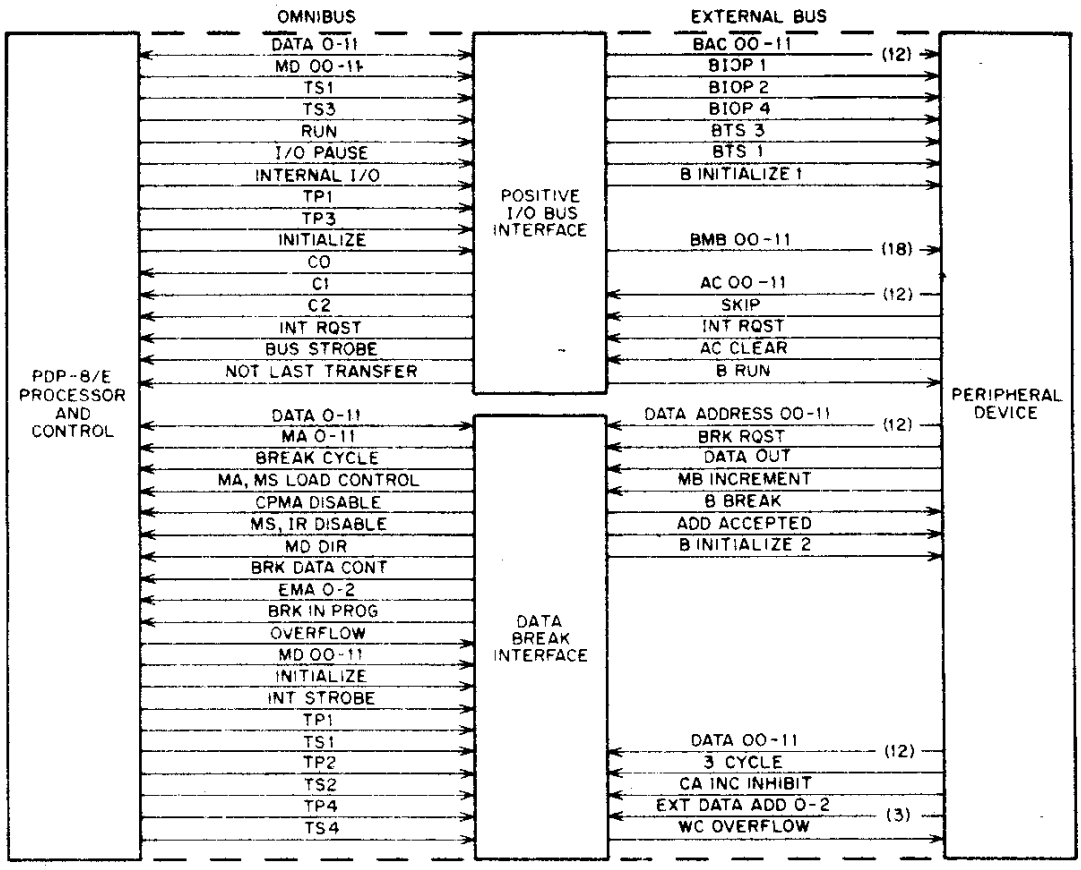


Figure 10-2 External Bus Signals and Related OMNIBUS Signals

SIGNAL NAME	DESCRIPTION
BAC 00-11	These signals represent the content of the PDP-8/E Accumulator register (AC). Information in the AC is transferred on the OMNIBUS DATA lines to the Positive I/O Bus interface. The interface buffers the signal and provides the BAC output. The BAC bits are strobed into registers in the peripheral when an IOT instruction is generated. 1 = +3V.
AC 00-11	The signals on these lines represent the contents of a register in the peripheral. This information is transferred to the Positive I/O Bus interface where it is put on the data lines for transfer to the PDP-8/E AC. 1 = GND.
BMB 00-11	The signals on these lines represent the content of the Memory Sense registers. This information is transferred from memory on the Memory Data (MD) lines. The MD lines are monitored by the Positive I/O Bus interface and the signals are converted to the BMB bits. These bits are used during IOT instructions; BMB03-08 carry the device selection code, while BMB09-11 are converted to BIOP pulses. 1 = +3V.

SIGNAL NAME	DESCRIPTION
BIOP 1, 2, & 4	These pulses are generated in response to the voltage levels on MD09-11 (BIOP4-1, respectively). These pulses generate IOT pulses within the peripheral, causing it to perform a certain operation. The width of the BIOP pulses and the interval between pulses are variable and can be adjusted on the Positive I/O Bus interface. Pulse = +3V.
BTS1, BTS3	These signals represent the TS1 and TS3 signals of the OMNIBUS. They synchronize operations in the peripheral with those in the computer and perform functions peculiar to the peripheral. They are primarily used in data break timing. Pulse = +3V.
B RUN	If this signal is GND, the computer is executing instructions.
AC CLEAR	When this signal is asserted (brought to GND) along with the AC bits, the result is a jam transfer of data to the AC. The signal may also be asserted by a separate IOP, clearing the AC.
SKIP	SKIP is asserted (grounded) by an IOT instruction. It causes the next sequential instruction to be skipped. If the SKIP bus is asserted during more than one IOT of an I/O instruction, the program skips a corresponding number of instructions. No more than three skips can be made by a single instruction.
B INITIALIZE 1	This 600 nanosecond-duration positive pulse is used to clear AC and link and to clear all flags in peripherals. It is generated at power turn on, and by the Clear All Flags (CAF) IOT, 6007.
DATA 00-11*	These lines transfer data from a data break peripheral to the data break interface. The peripheral transfers the information when it receives the B BREAK signal from the interface, indicating the start of the true break cycle. At TS2 of this break cycle, the data

SIGNAL NAME	DESCRIPTION
	break interface transfers the data to the OMNIBUS DATA 0-11 lines, which carry the data to the CP's memory buffer. 1 = GND.
B BREAK*	This signal is generated in the data break interface and transferred to the peripheral, where it enables a parallel loading of data, either into or out of the peripheral. The data break interface, in addition to generating B BREAK, asserts the OMNIBUS BREAK CYCLE line, notifying the computer that the break cycle has begun. 1 = GND.
DATA OUT*	This signal is produced by the peripheral and sampled by the data break interface. When DATA OUT is asserted (grounded) during the break cycle, data is transferred from the computer's memory to the peripheral.
DATA ADD 00-11*	These lines transfer address information from the peripheral to the OMNIBUS MA lines. If the peripheral is a 3-cycle break device, the address represents the memory location of the word count. Since this location is always the same for a 3-cycle device, the DATA ADDRESS lines are hard-wired in the peripheral. This address must be even (ending in 0, 2, 4, or 6) for word count. The data stored in this location represents the 2's complement of the number of data words to be transferred. The next sequential location is read from memory as the Current Address register. The data stored in this location represents the memory address of the data to be transferred. If the peripheral is a 1-cycle break device, the address on the DATA ADDRESS lines is provided by a register in the peripheral and represents the memory address of the data to be transferred. The address on the DATA ADDRESS lines is sampled by the TP4 pulse. The OMNIBUS CPMA DISABLE line is asserted by the data break interface at TP4 to enable the DATA ADDRESS information to be placed on the MA lines. 1 = GND.
BRK RQST*	This signal is asserted (brought to ground) by the peripheral when it is ready for a word transfer. When BRK RQST is present at INT STROBE time, the data break operation is entered. The OMNIBUS INT IN PROG line is asserted, and a load enable signal is provided for the data break interface break memory address (BKMA) register.

*Pertains to Data Break interface only.

SIGNAL NAME	DESCRIPTION
ADD ACCEPTED*	This signal is generated by the data break interface when a BRK RQST signal has initiated the data break operation. ADD ACCEPTED is used in the peripheral to clear the BRK RQST flip-flop. Pulse = GND.
MB INCREMENT*	When this signal is at ground level during the true break cycle, the contents of the memory location are acted upon as outlined in the following table.

MB INCREMENT	DATA OUT	Operation Performed	Descriptive Term Used for Operation
Low	Low	Contents of the memory location are incremented.	MB Inc
Low	High	Data on the DATA 00-11 lines is added to the contents of the memory location.	Add to Memory (ADM)

CA INCREMENT INH*	When this signal is asserted (grounded) during the CA cycle of a 3-cycle data break, the CA is not incremented.
3-CYCLE*	This signal is transferred from the peripheral to the data break interface to notify the interface logic to set either the WC flip-flop (3-cycle transfer) (ground input) or the B flip-flop (1-cycle transfer).
WC OVERFLOW*	The interface transfers this signal to the peripheral to notify it that the word count location in memory has become zero and that the data transfer should end. The signal is also present when overflow occurs during MB increment or ADM. Pulse = GND.
EXT DATA ADD 0-2*	These three lines are used when a KM8-E Memory Extension and Time Share interface is included in the basic PDP-8/E. The peripheral uses the lines to indicate the particular memory field involved in the transfer. During a 3-cycle data break, WC and CA cycles always occur in field 0, while only the B cycle occurs in the field specified by the extended data address. 1 = GND.
B INITIALIZE 2*	This positive signal clears all flags in the peripheral and is essentially the INITIALIZE signal of the OMNIBUS. It is used by the break device in lieu of B INITIALIZE 1 so as to reduce loading on the latter.

* Pertains to Data Break interface only.

APPLICATION

The nature of the external bus and its relationship to the OMNIBUS have been presented. Now, the use of the bus must be fully explored. First of all, the user wants to transfer data between his peripheral and the computer's memory. He can do this in any one of three ways—programmed I/O transfers, program interrupt transfers, or data break transfers. The basic ideas behind all three methods of data transfer have been discussed in previous chapters, and the user should be familiar with these before proceeding any further in this chapter.

Programmed I/O Transfers

Figure 10-3 is a logic block diagram that shows the more important signals involved in a programmed I/O transfer. The transfer process is, of course, similar to that which takes place between the computer and a peripheral interfaced to the OMNIBUS. Each peripheral has a flag flip-flop that is set when the peripheral is ready to receive or send information. A programmed IOT instruction is used to check this flip-flop. The program enters a waiting loop until the peripheral is ready. When the flag flip-flop is set, IOT XXA (illustrated on the block diagram) asserts the SKIP bus. The program then skips to an instruction that transfers program control to a servicing subroutine. The subroutine carries on the IOT dialogue between peripheral and processor. Although the general process is similar, both the method of peripheral selection and the use of the SKIP function differ for external bus peripherals.

The method of peripheral selection will be examined first. As shown on the block diagram, the peripheral contains a device selector and an IOT generator. The device selector monitors the BMB03 through BMB08 lines. These lines are merely the buffered OMNIBUS MD03 through MD08 lines. Thus, when an IOT instruction is issued, BMB03 through BMB08 carry the code for a particular peripheral. The device selector responds to the code by producing a DEVICE SELECTED signal. This signal is applied to the IOT generator. Unlike the OMNIBUS peripherals, external bus peripherals require BIOP pulses to carry out the operations specified by the IOT instructions. These pulses are generated in the Positive I/O Bus interface and reflect the information present on the OMNIBUS MD09, 10, and 11 lines. BIOP pulses are applied to the IOT generator where they are regenerated as IOT pulses that initiate operations within the peripheral. Figure 10-3 shows that the BIOP pulses (1, 2, and 4) are generated only when the following conditions are met: the OMNIBUS I/O PAUSE line is asserted, indicating that an I/O transfer is to take place; the INTERNAL I/O line is negated, indicating that the I/O transfer is to or from an external bus peripheral; the MD09, or MD10, or MD11 line is asserted. Thus, if the first two conditions are met, and MD09 is asserted, BIOP4 is generated. Similarly, BIOP1 is generated when MD11 is asserted. Each BIOP pulse is regenerated as an IOT pulse within the peripheral. Thus, BIOP1 becomes IOT XX1; BIOP4 becomes IOTXX4. (In the preceding notation, XX represents the particular device selection code).

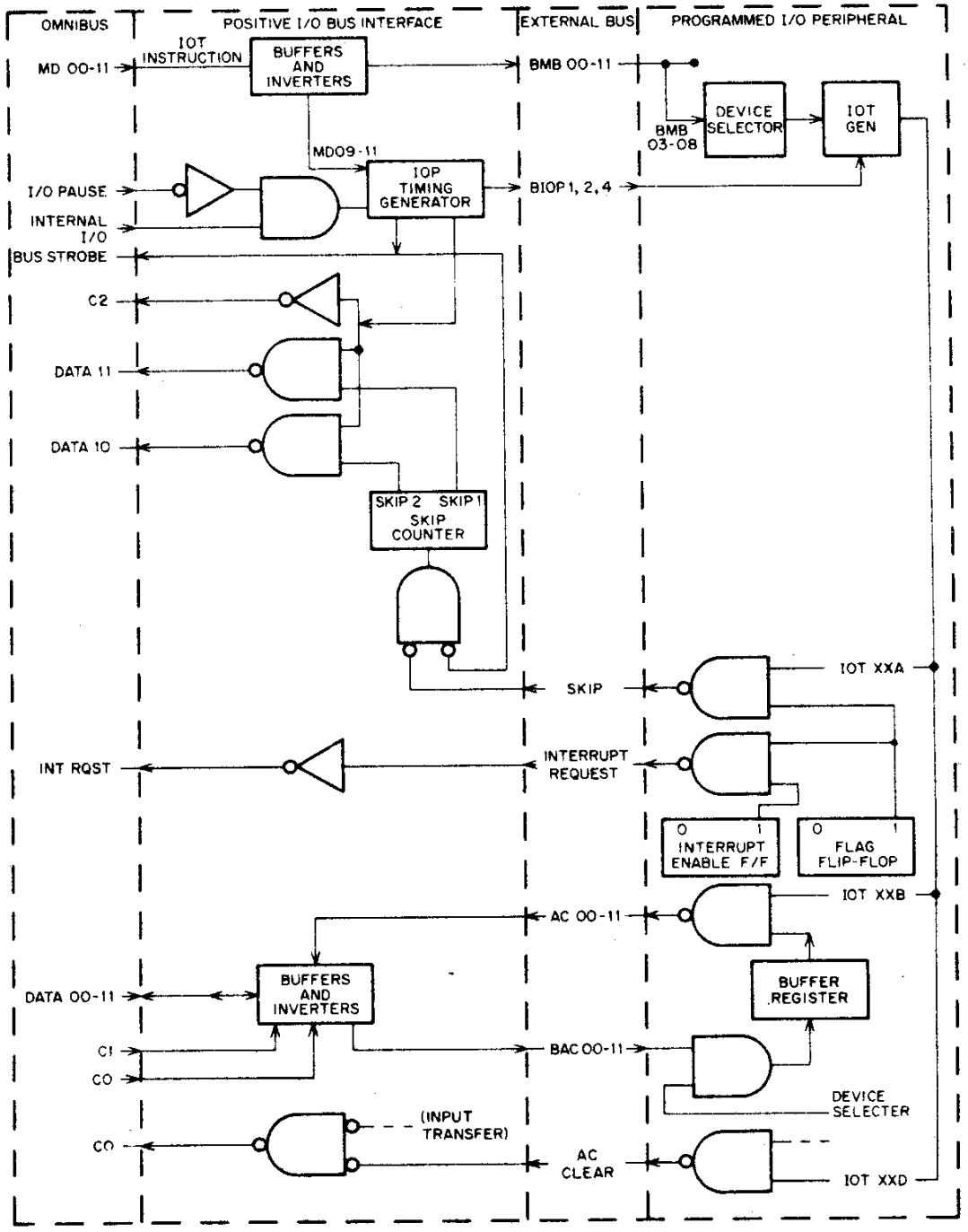


Figure 10-3 Block Diagram, Programmed Data Transfer

Specific IOT pulses perform specific operations in the peripheral. The primary use of specific IOT pulses, as well as the relationship between IOT, BIOP, and MD bits, is as follows:

MD Bit	BIOP	IOT	Primary Use
09	BIOP4	IOT XX4	Reading, loading, and clearing buffers.
10	BIOP2	IOT XX2	Clearing flags; clearing AC.
11	BIOP1	IOT XX1	Sampling flags; skipping.

These relationships are standard within DEC and are certainly not mandatory for the user. He may decide to use IOT XX1 to clear a flag, or he might wish to use IOT XX4 to sample a flag. He may even use combinations such as IOT XX5 by generating both BIOP1 and BIOP4 with the *same IOT instruction. In any event, it is wise to develop some standard* such as that expressed above.

The discussion so far has shown how the peripheral is selected and how operations are initiated by the IOT instruction. The use of the SKIP function must now be explained. One should recall that a flag flip-flop in the peripheral is sampled by an IOT pulse and a skip is effected. IOT XXA checks the status of this flip-flop (XXA is used here, rather than XX1, to emphasize the fact that "A" may be 1, 2, 5, or whatever the user wishes). When the peripheral is ready, the SKIP bus is asserted. A strobe signal from the IOP timing generator clocks the skip counter, a two-stage binary counter. This strobe signal is generated near the end of the BIOP pulse. If one BIOP pulse is generated by the IOT instruction, the skip counter is clocked only once and the SKIP 1 line is asserted. A control signal, produced at the end of the BIOP pulse, then asserts both the C2 line and the DATA 11 line of the OMNIBUS. When the C2 line is asserted, with C1 negated, DATA + PC goes to PC. In other words, the PC is incremented by one and the program skips one instruction. Figure 10-4 is a timing diagram of the SKIP function and is helpful in visualizing the process. Note that two strobes are generated, each performing the function shown on the diagram.

It should also be noted on the timing diagram that the CP operation is halted from TP3 to the beginning of the second BUS STROBE. This BUS STROBE, which occurs after NOT LAST TRANSFER has been negated, generates INT STROBE, which restarts the CP. This is the disadvantage inherent in external bus interfacing—the CP must remain inactive while BIOP pulses are generated and control lines are activated. Thus, while OMNIBUS I/O transfers are accomplished in 1.2 microseconds, the minimum time required for external bus I/O transfers is 2.6 microseconds when only one BIOP pulse is generated by the IOT instruction. Suppose two BIOP pulses are generated. The user might want to skip two instructions in the program, for example. In this case, the IOP timing generator produces three BUS STROBES. Each of the first two BUS STROBES clocks the skip counter. Thus, the SKIP 2 line is asserted. Again, C2 is asserted by a control signal, but now DATA 10 rather than DATA 11 is brought low. BUS STROBE three then enables DATA + PC *to the PC, and the program skips the next two instructions. This may be* advantageous for a specific application, but note the increased transfer processing time—it is now 3.6 microseconds. If three BIOP pulses are used, the CP is halted for 4.6 microseconds. It must be remembered that these are minimum values. Figure 10-4 shows that the time from

TP3 to the beginning of BIOP1 is variable, as is the width and the pulse separation when more than one BIOP is issued by the IOT instruction.

The user may want to increase IOP width and/or separation for specific applications. He should bear in mind, however, that this affects processing time and may slow the computer appreciably. The width and separation are controlled by separate potentiometers on the interface.

The minimum allowable values are 800 nanoseconds for pulse width and 200 nanoseconds for pulse separation. The resistor values in the timing circuits allow the user to increase these values to five times the minimum.

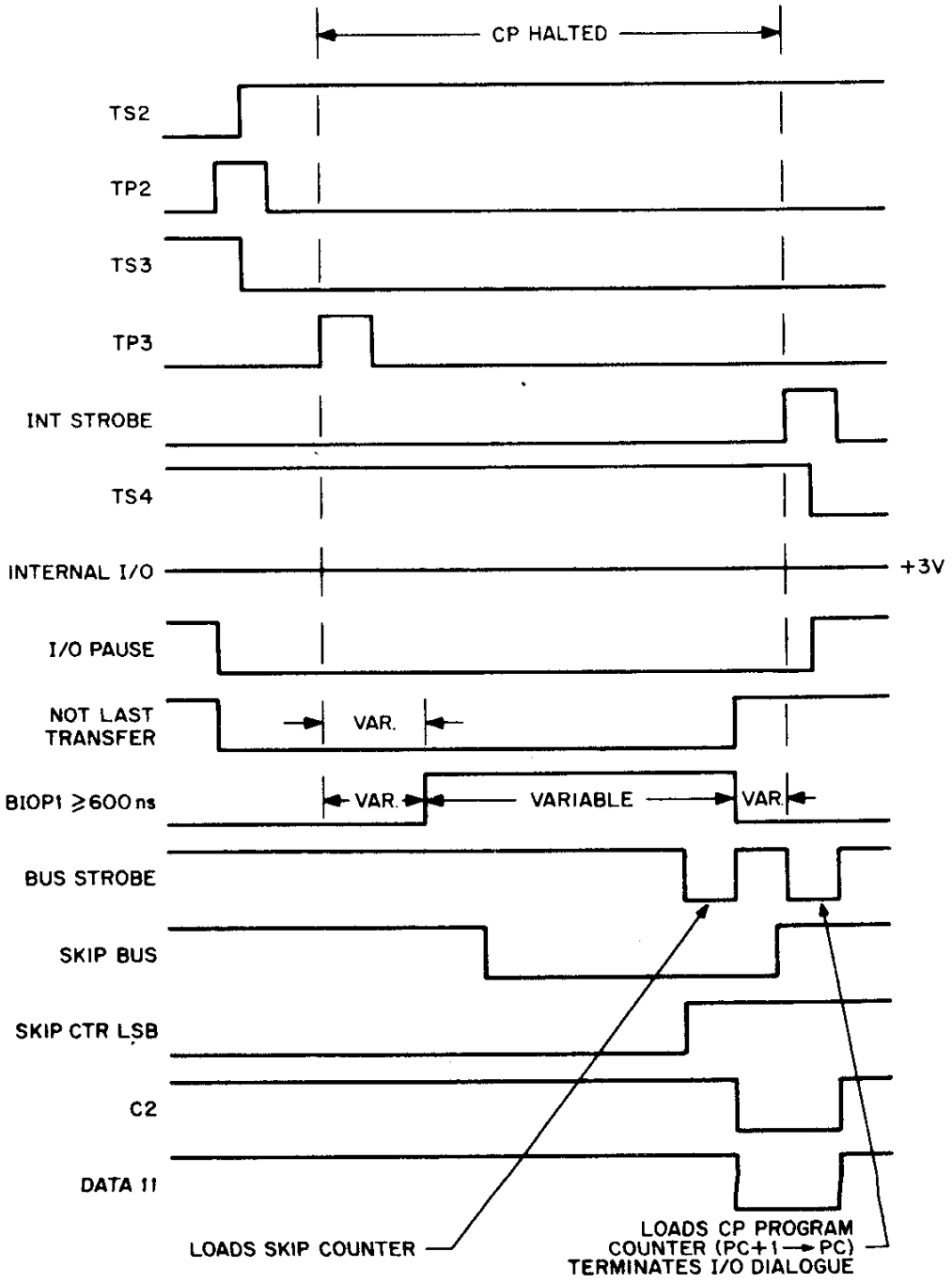


Figure 10-4 Timing Diagram, Skip Bus Application

Figure 10-5 shows another timing diagram, this one illustrating a data word transfer. Conventionally, BIOP4 is being used to accomplish the transfer. Note that the BIOP pulses do not occupy specific time slots. Thus, if BIOP1 and BIOP2 are not required, as in this example, the IOP timing generator produces BIOP4 without any delay. Again two BUS STROBES are generated, each performing the indicated function. The C1 control line must be asserted to indicate an input data transfer. This is done if data is placed on any external bus AC line. The data is buffered and inverted, and placed on the OMNIBUS DATA 0-11 lines (one of these lines, DATA XX, is illustrated). With only C1 asserted, a 1's transfer to the AC is carried out. If a jam transfer to the AC is desired, the C0 control line must be asserted along with C1. This can be done only if the external bus AC CLEAR line is brought to ground, which can be accomplished either by gating IOT4 in the peripheral or by other means that the user might select. Do not permanently ground AC CLEAR.

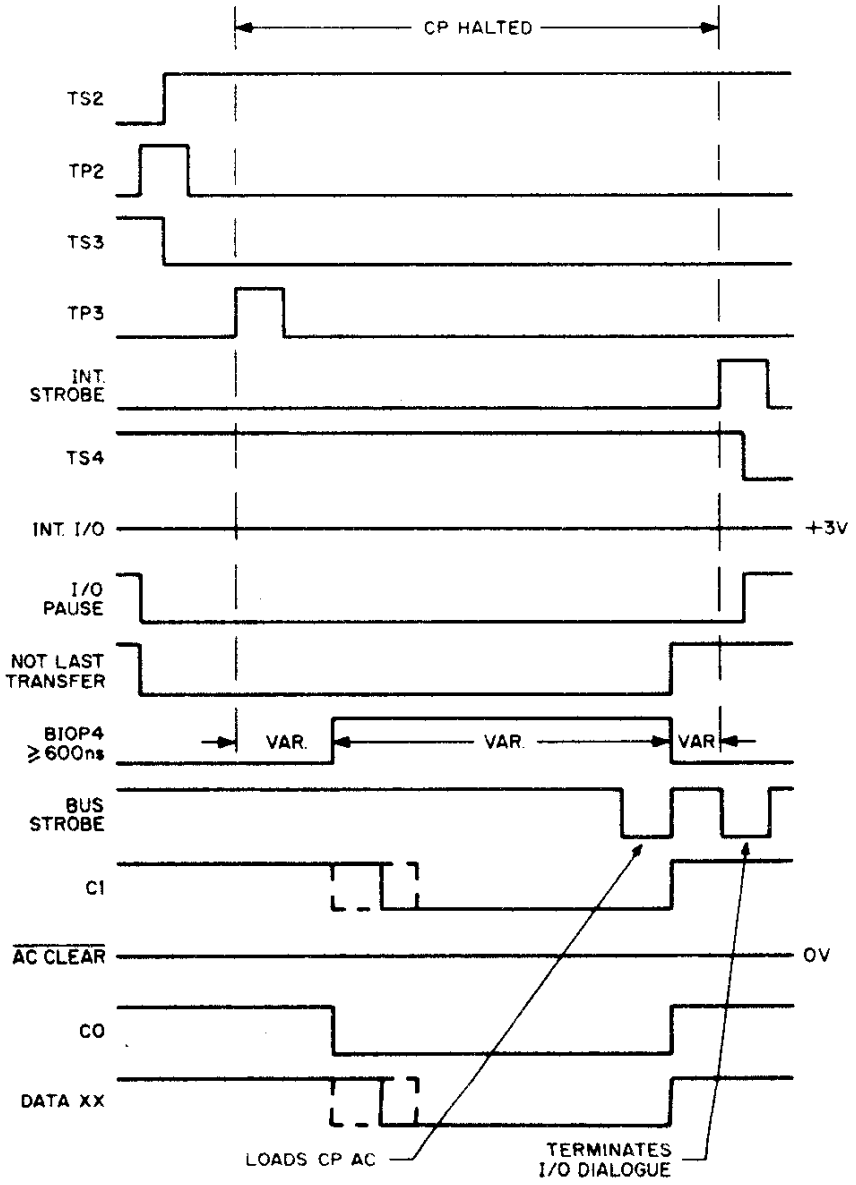


Figure 10-5 Timing Diagram, Programmed Data Transfer

Program Interrupt Transfers

Program interrupt devices are connected to the external bus exactly as are programmed data transfer devices, and use the very same signals. The entire program interrupt process proceeds just as it does for OMNIBUS peripherals. The peripheral requests an interrupt by setting its flag flip-flop. The processor honors the request and services the peripheral in a program subroutine. The data transfer is accomplished by programmed transfers, as explained in the preceding section. The user must provide means of asserting (grounding) the external bus INTERRUPT REQUEST line when the peripheral is ready for a transfer. Figure 10-3 shows the peripherals flag flip-flop controlling the INTERRUPT REQUEST line. The SKIP line must also be used when more than one such interrupt device is connected onto the external bus and is utilized as detailed in the preceding section.

Data Break Transfers

Data break transfers involving peripherals on the external bus are accomplished in much the same way as transfers involving OMNIBUS peripherals. Refer to figure 10-6, which is a logic block diagram that shows the more important functional blocks and signals involved in a data break transfer. Figure 10-7 shows the timing relationship of the major signals and, along with the block diagram, should be referred to throughout this discussion.

An important feature that is illustrated on the block diagram is the circular flow of data between OMNIBUS and peripheral. The peripheral receives data from the computer by way of the OMNIBUS MD lines and the external bus BMB lines. It sends data to the computer by way of the external bus DATA 00-11 lines and the OMNIBUS DATA 0-11 lines. Thus, there is a need for both interface boards when data break peripherals are used. In addition to providing a data path, the Positive I/O Bus interface provides BIOP pulses. These pulses, as in programmed data transfers, are used in conjunction with a device selector to produce IOT pulses in the peripheral. However, data break IOT pulses are used only to initiate the data break operation and, once the operation has started, program control of the peripheral ceases. Therefore, the program provides IOT instructions that tell the peripheral, via the IOT pulses, to perform any preliminary operations that may be required. When the peripheral has followed these instructions and is ready to either send or receive data, it requests a Data Break. The data break interface logic then assumes control over the operation.

When the Data Break interface receives the BRK RQST signal from the peripheral, it uses the next INT STROBE from the OMNIBUS to assert the OMNIBUS BRK IN PROG line. At TS4 the interface checks the priority network to make sure no higher priority device is present in the system. If none is present, the interface asserts two OMNIBUS processor-control lines at TP4. Assertion of these lines—CPMA DISABLE and MS, IR DISABLE—causes the processor to load its CPMA register and suspend operation, while the peripheral interface BKMA register assumes control of the OMNIBUS MA lines (as with OMNIBUS data break peripherals, the break can occur at the end of any processor major state). At the same time—TP4—the interface major state control logic enters either

the word count cycle or the break cycle, depending on the state of the 3 CYCLE signal from the peripheral. If this peripheral is a single-cycle break device, the interface logic enters the B (break) cycle. The B BREAK signal is generated and sent to the peripheral where it either loads the output buffer register with data or places data from this register on the lines to the interface logic. If the transfer direction is from the peripheral to memory, data from the buffer register is placed on the DATA 00-11 lines. At TS2 this data is placed onto the OMNIBUS DATA 0-11 lines. The next TP2 loads the data into the MB, providing the BREAK DATA CONT has not been asserted, and the MB contents are then written in memory.

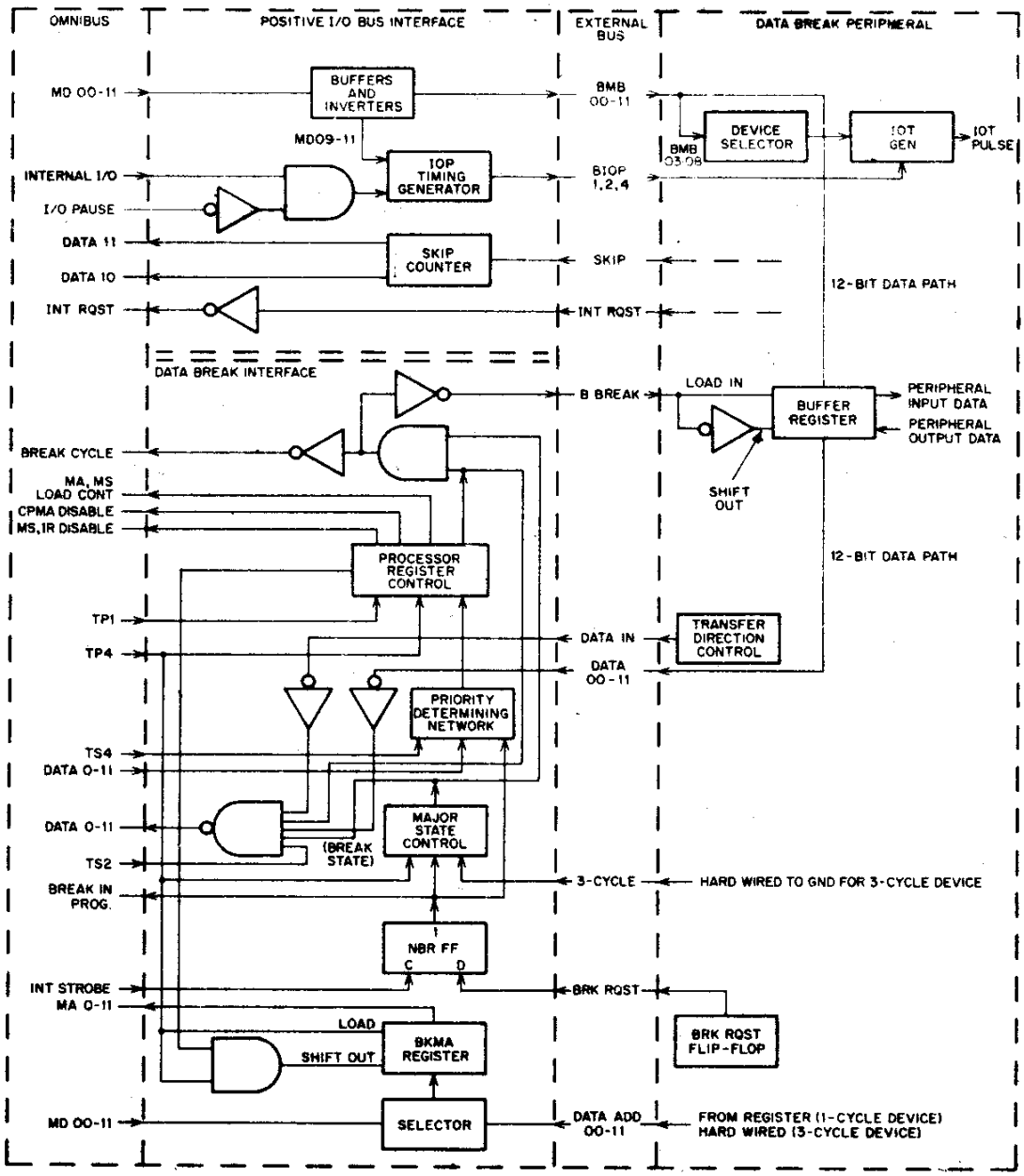


Figure 10-6 Block Diagram, Data Break Transfer

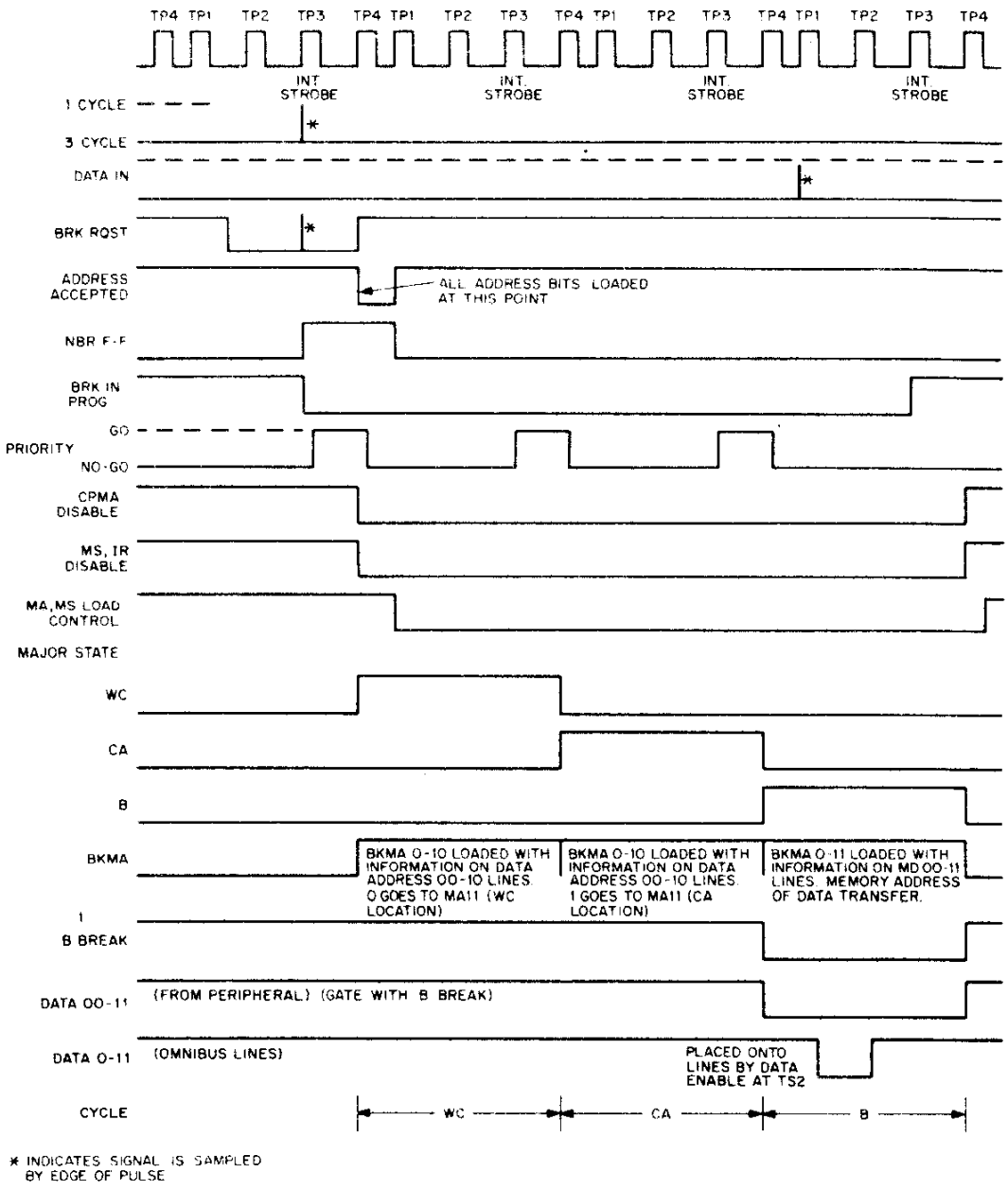


Figure 10-7 Timing Diagram, Data Break Transfer

Figure 10-7 indicates a peripheral-to-computer data transfer. It also shows the timing for a 3-cycle, rather than a single-cycle, break device, and indicates that the WC and CA cycles must be completed before B BREAK is generated at the beginning of the B cycle. In addition, the timing illustrates a very important and advantageous feature of the interface logic—priority is checked not only before entering the WC cycle, but also before entering the CA and B cycles. This means that high priority devices can override lower priority devices even in the middle of a break operation. As many as 12 Data Break peripheral interface boards can be inserted into the OMNIBUS. Each peripheral must first be assigned a priority level. Each interface contains an identical priority network that is wired in a unique fashion, so as to reflect the priority

level assigned to the peripheral. During any TS4 of a data break cycle, the highest priority interface, of those interfaces whose peripherals have made break requests, receives a 'go' signal from its priority network. All other interfaces whose peripherals have made requests receive a 'no-go' signal, and must wait until the next TS4 for another priority check.

This unique priority system allows the user to utilize his data transfer system in a most efficient manner. However, he must pay careful attention to the manner in which he assigns priorities. An essential rule should be: assign priorities in relation to the length of time data is available at the peripheral. For example: peripheral A and peripheral B both assert their BRK RQST lines at some time between TP1 and TP3 of a particular processor cycle. Peripheral A has been assigned a higher priority than peripheral B; and, thus, takes control of the break operation. If 'A' is a 3-cycle device, assuming no other higher priority devices are present, it does not relinquish control until it has transferred the data word during the B-cycle. During TS4 of this cycle, 'B' receives a priority 'go' signal and takes control of the operation at TP4. If 'B' is also a 3-cycle break device it must now go through WC and CA before the actual transfer can begin.

Seven to eight microseconds has elapsed from the moment that 'B' requested a break until the moment when its interface generates B BREAK. If 'B' is a very high speed device, the data that was present in its output register at the time of the break request may no longer be present when B BREAK is finally generated. The solution in this example is obvious (if 'A' is a slower device than 'B', that is): assign 'B' a higher priority than 'A'. OMNIBUS data break peripherals are also equipped with priority networks, although these generally differ from the network of the Data Break interface. Nevertheless, all types work together, and the only limitation is on the total number of priority networks, 12.

While OMNIBUS data break peripherals are generally single-cycle devices, those data break peripherals interfaced to the external bus are, for the most part, 3-cycle devices. Because the WC and CA registers of a 3-cycle device are located within core memory, the Data Break interface BKMA register is used differently from its counterpart in an OMNIBUS peripheral. Figure 10-7 indicates the use of the BKMA register during each cycle of the 3-cycle break operation. During WC, the hard-wired memory address of the word count register is loaded into the BKMA. The only restrictions on this address are that bit 11 be a '0' and that the memory location be in memory field 0.

The word count in memory is brought out, incremented, and deposited back in memory. During CA, the same hard-wired address is loaded into BKMA 0-10, and BKMA11 is asserted. The result is the memory address of the current address register. The current address is brought out of memory, incremented, and placed on the MD lines. Not only is this address deposited back in the CA register, but it is also loaded into the BKMA register at the beginning of the B cycle. Thus, the current address specifies the memory address to which, or from which, the data is to be transferred.

In the preceding explanation, data has been presented as a 12-bit data word that is transferred unaltered to or from a specified memory location. However, the interface logic provides the user with two features that somewhat alter this idea. The first, MB INCREMENT, allows the user to transfer a single bit of information via the OMNIBUS DATA 11 line. The result is an incrementation of the data contained in the specified memory location. The peripheral must cause the external bus MB INCREMENT line to be asserted with the DATA OUT line asserted. Each time MB INCREMENT is asserted, the interface logic asserts the OMNIBUS DATA 11 line and the data in the memory location is incremented.

The second feature, Add to Memory (ADM), allows the user to alter the 12-bit data word in memory. Specifically, the incoming data word and the data contained in the addressed memory location are added in the processor memory buffer. The result is then returned to the addressed location. As with MB INCREMENT, the peripheral must cause the MB INCREMENT line to be asserted; but now the DATA OUT line must be negated. The result is ADM. Table 10-1 lists the possible states of the DATA IN line and the MB INCREMENT line, and the resulting data transfer. L indicates that the line is asserted (grounded), while H indicates that the line is negated.

Table 10-1. Data Control Lines (Data Break Interface)

DATA OUT	MB INCREMENT	TYPE AND DISPOSITION OF DATA
L	H	12-bit data word transferred from the addressed memory location to the peripheral buffer via the BMB lines.
H	H	12-bit data word transferred from the peripheral buffer to the addressed memory location via the DATA00-11 lines.
L	L	MB INCREMENT. The contents of the addressed memory location are incremented.
H	L	Add to Memory (ADM). The twelve-bit data word on the DATA 00-11 lines is added to the contents of the addressed memory location, and the result stored in the addressed memory location.

INTERFACING TECHNIQUES

At this point the user should have a good understanding of what the external bus is and the way that he wants to use it. The remainder of this chapter is devoted to the technique of interfacing the bus to the user's peripheral.

When the user purchases a PDP-8/E, he selects the options he requires. Assume that he wants both interfaces. Digital Equipment Corporation then supplies the boards, the interconnecting cables, and the connectors on either end of each cable. The customer specifies whether the cable should be shielded flat cable or round or flat coaxial cable. In addition, he selects the cable length from a variety of standard lengths made available by DEC. The connectors on either end of the cable are standard DEC connectors and those on the free end will connect into any peripheral control manufactured by DEC.

PDP-8/I and 8L-type Peripherals

If the user has a PDP-8/I or PDP-8/L-type peripheral control, the PDP-8/E can be easily interfaced to it. If the control is a data break peripheral control, the user first plugs the two interface boards (with the cables already connected to the board) into the OMNIBUS, in any available slots. He then inserts the peripheral control connectors into the appropriate slots in the peripheral control connector block. In general, the appropriate slot is determined as follows:

Cable Connector		Peripheral Module Slot	Signals
I/O Bus Interface	Data Break Interface		
Cable 1		A1	BAC
Cable 2		A2	BMB
Cable 3		A3	AC INPUT
	Cable 4	A4	DATA ADDRESS
	Cable 5	A5	DATA BITS

(The above information is generally true; however, because peripheral control module arrangements sometimes vary, the user should check the peripheral control interconnection information to see that the stated correspondence is correct.)

If the peripheral is a programmed I/O device, only the Positive I/O Bus interface must be inserted into the OMNIBUS. In this instance, cables 1, 2, and 3 of the board connect as indicated above. If several peripherals are being connected to the bus, they are connected as indicated in figure 10-1. Table 10-2 lists the external bus signals and provides the connector pin assignment for each signal line. The location of a particular pin can be determined as follows: if the connector board is held so that the connector pins are to the left and the keying cutouts appear as shown in Figure 10-8 (the longer cutout on the bottom), pin A is the topmost pin, pin V is the bottommost, and side 2 is the nearer side. Pins are lettered A through V (excluding G, I, O, and Q). This convention of pin identification also applies to the M-series and G-series connectors and modules.

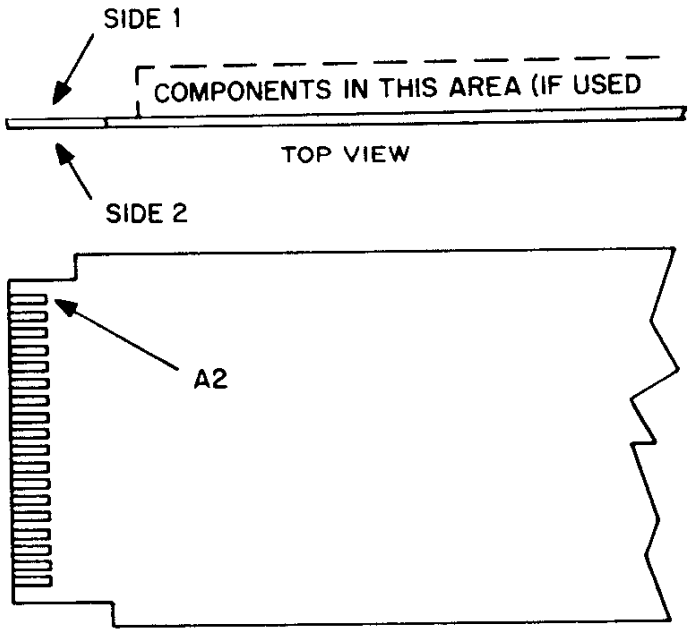


Figure 10-8 Connector Pin Identification

Table 10-2 External Bus Connector Pin Assignments

POSITIVE I/O BUS INTERFACE

Signal Name			Connector Pin
Cable 1	Cable 2	Cable 3	
BAC 00	BMB00(1)	AC 00*	B1
BAC 01	BMB01(1)	AC 01*	D1
BAC 02	BMB02(1)	AC 02*	E1
BAC 03	BMB03(0)	AC 03*	H1
BAC 04	BMB03(1)	AC 04*	J1
BAC 05	BMB04(0)	AC 05*	L1
BAC 06	BMB04(1)	AC 06*	M1
BAC 07	BMB05(0)	AC 07*	P1
BAC 08	BMB05(1)	AC 08*	S1
BAC 09	BMB06(0)	AC 09 *	D2
BAC 10	BMB06(1)	AC 10*	E2
BAC 11	BMB07(0)	AC 11*	H2
BIOP1	BMB07(1)	SKIP BUS*	K2
BIOP2	BMB08(0)	INT RQST BUS*	M2
BIOP4	BMB08(1)	AC CLEAR BUS*	P2
BTS3	BMB09(1)	B RUN (0)	S2
BTS1	BMB10(1)		T2
B INITIALIZE 1	BMB11(1)		V2

DATA BREAK INTERFACE

Signal Name		Connector Pin
Cable 1	Cable 2	
DATA ADD 00*	DATA 00*	B1
DATA ADD 01*	DATA 01*	D1
DATA ADD 02*	DATA 02*	E1
DATA ADD 03*	DATA 03*	H1
DATA ADD 04*	DATA 04*	J1
DATA ADD 05*	DATA 05*	L1
DATA ADD 06*	DATA 06*	M1
DATA ADD 07*	DATA 07*	P1
DATA ADD 08*	DATA 08*	S1
DATA ADD 09*	DATA 09*	D2
DATA ADD 10*	DATA 10*	E2
DATA ADD 11*	DATA 11*	H2
BRK RQST*	3 CYCLE*	K2
DATA OUT*	CA INCREMENT INH*	M2
B BREAK (0)	BWC OVERFLOW	P2
ADD ACCEPTED	EXT DATA ADD 02*	S2
MB INCREMENT*	EXT DATA ADD 01*	T2
B INITIALIZE 2	EXT DATA ADD 00*	V2

Signals marked * are input signals, and are asserted at ground. Unmarked signals are output signals, and are asserted at +3V.

Note: Be sure to ground all other pins except U1, V1, A2, and B2.

Customer Peripherals

The interfacing technique becomes more complicated when the user desires to connect his own peripheral. The user must decide if he wants to use the cables and connectors that are supplied with the interface board. If he elects to do so, he must use DEC connector blocks that accommodate the BC08J interface cables. He must then provide a mechanical interface between the DEC connector block and the peripheral input connectors, and, in addition, must provide an electrical interface between the external bus and the peripheral. The designer of the interface has to consider questions of voltage levels, loading criteria, mechanical compatibility, etc. The task of interfacing is greatly simplified if the customer makes use of DEC's line of M-series modules and compatible H-series connector blocks and mounting panels.

DEC Logic Module Interfacing

As an example of the application of DEC modules to the user's interfacing problem, consider the following: the peripheral, whether a data break device or a programmed transfer device, must be able to recognize its device code and respond in some way to that code. The device code is contained in the BMB03-08 bits, which are provided by cable 2 from the Positive I/O Bus interface. The user can design his own integrated circuit or discrete component interface to decode the BMB03-08 bits and provide a Device Selected signal. He can then breadboard the circuit, test it, build it, and install it in some kind of connector,

which he then has to connect to the interface cable. Alternately, he could use a DEC M103 Device Selector module, which, in addition to providing the correct Device Selected signal, accomplishes most of the interfacing, provides diode clamp protection, and even provides extra logic gates for the customers use.

The M103 module is shown in Figure 10-9. Assume that the user assigns device code 14 to his peripheral. The selected BMB lines provide an enabling signal for gate 2 whenever device 14 is selected. The module also provides the peripheral with the necessary IOT pulses, as is indicated in Figure 10-9. The output signals are available for immediate use.

The M-series modules use TTL logic, and the input loading/output drive requirements are given in number of unit loads. A unit load is defined as follows: in the logic 0 state, the driver must be able to sink 1.6 milliamps (maximum) from the unit load's input circuit, while maintaining an output voltage equal to or less than 0.4 volts; in the logic 1 state, the driver must maintain an output voltage equal to or greater than 2.4 volts, while supplying the unit load with a leakage current of no more than 40 microamps. The M103 is capable of driving 37 TTL unit loads at the IOT outputs, while the DEVICE SELECTED output can drive 16 TTL unit loads.

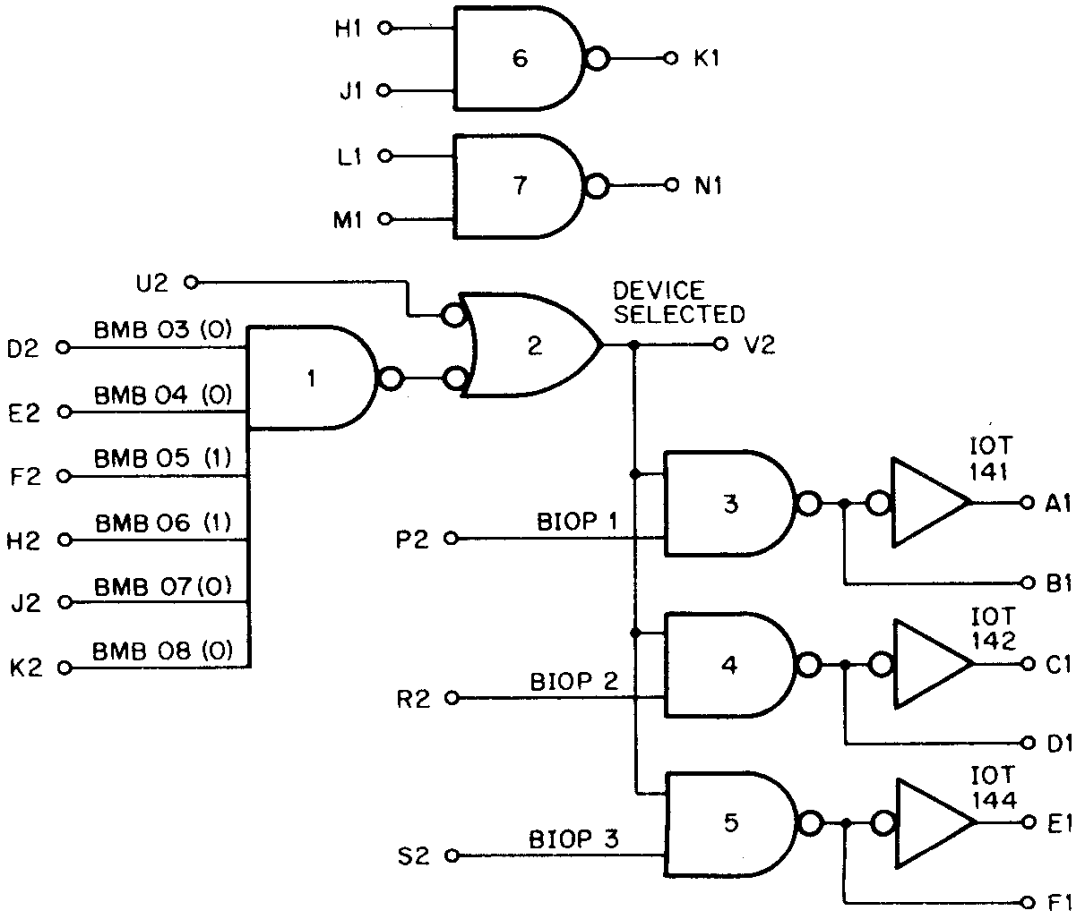


Figure 10-9 M103 Device Selector Module

Besides getting the peripheral selected, it is also necessary to transfer data. If, for example, data is to be sent to the PDP-8/E accumulator, it must be fed onto the external bus AC00-11 lines. The M624 Bus Driver module can provide this capability. This module, shown in Figure 10-10, contains 15 bus drivers. Twelve have a common gate line and can be used as shown in Figure 10-11. An output register in the peripheral is loaded with the data that must be transferred. The IOT pulse can then clock the data onto the AC lines. The user provides the proper input drive for the bus drivers. The M624 presents the following input TTL unit loads; 12 loads at the clock input (IOT 14X line); 1 load at the data inputs.

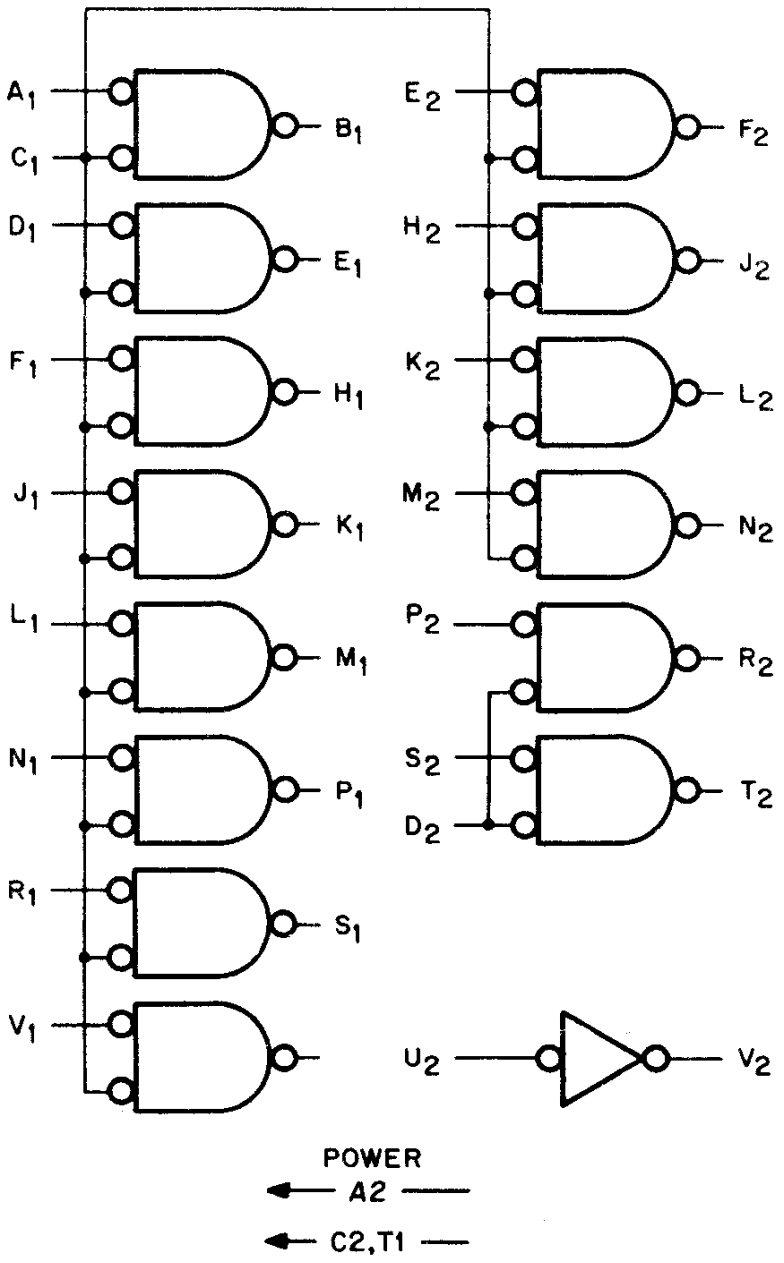


Figure 10-10 M624 Bus Driver Module

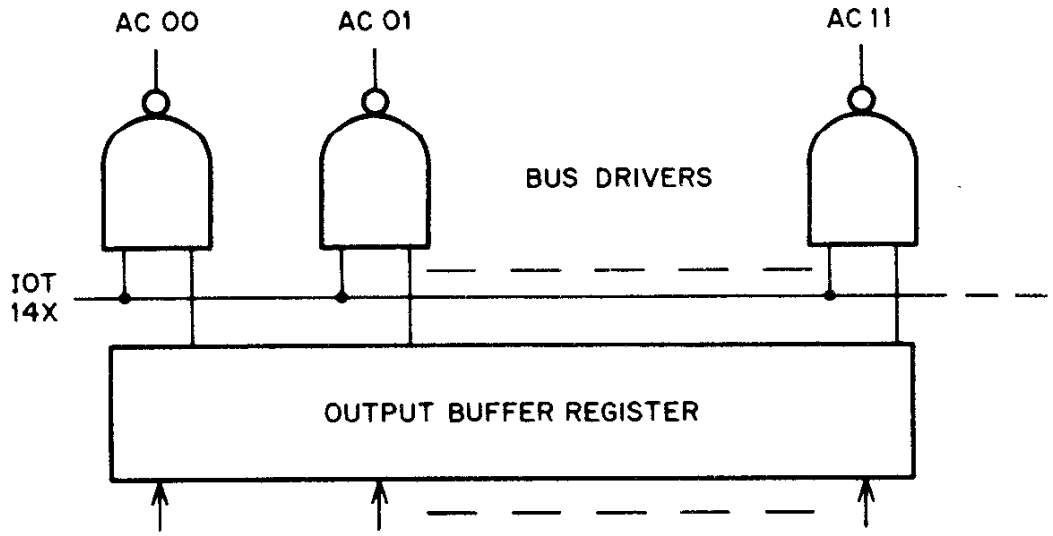


Figure 10-11 Bus Driver Application

The user also has to transfer data from the accumulator. Therefore, he must get the data on the BAC 00-11 lines into the peripheral buffer register. While doing this, he must take care that he does not place an excessive load on any BAC line. If he uses an M101 Bus Data interface, shown in figure 10-12, he can be sure of maintaining proper loading of these lines. Figure 10-13 shows how this interface must be used. Twelve of the gates are connected to the 'device select' line. When the device (XX) is selected, the BAC 00-11 data is applied to the input lines of the buffer register. IOT XXY then loads the register with the data word. The M101 presents the following input TTL unit loads: 15 loads at the clock input (device select line); 1 load at the data inputs.

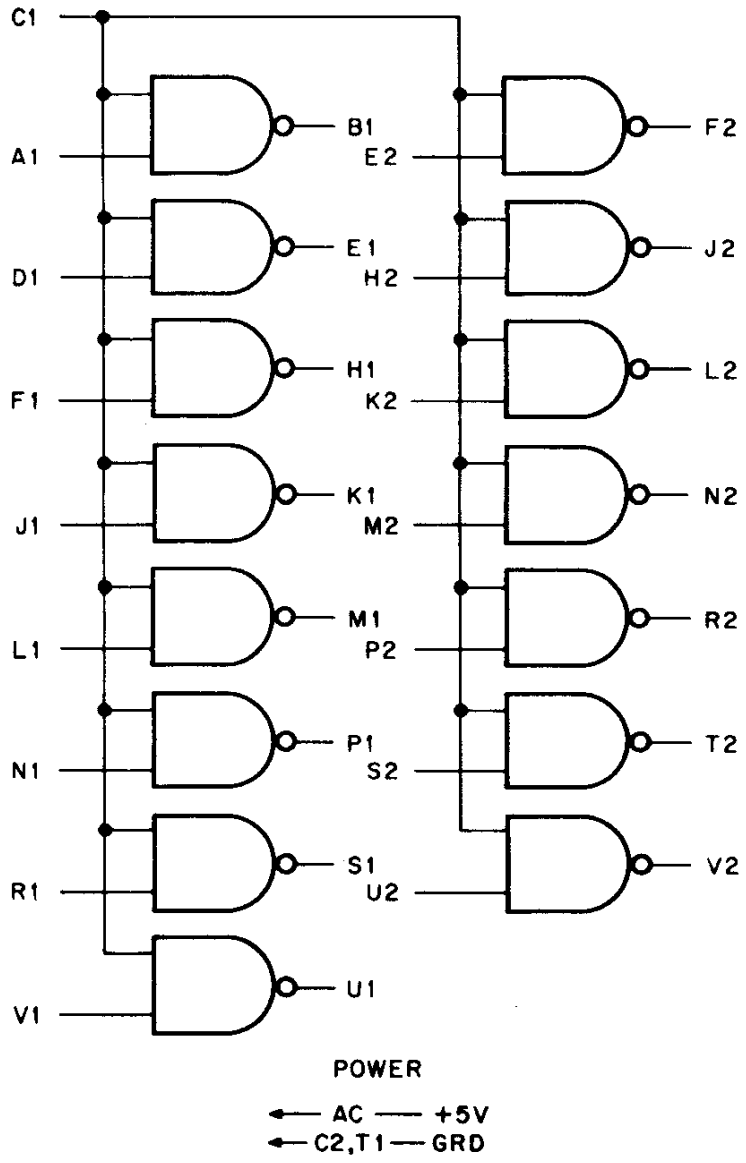


Figure 10-12 M101 Bus Data Interface

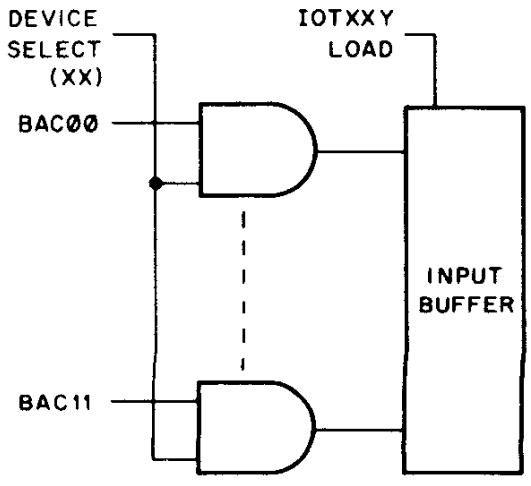


Figure 10-13 Bus Data Application

A variety of M-series modules are available from which the user can select to meet his needs. A list of these modules is given at the end of this chapter. This list should prove helpful in the design of interfaces. Attention is also directed to DEC's DIGITAL LOGIC HANDBOOK available from all sales offices.

M-series modules greatly simplify the task of electrical interfacing. However, this is only part of the problem. The external bus and the peripheral must also be mechanically interfaced. DEC's H-series of connector blocks and mounting panels are extremely valuable in this application. The following example takes the user through a mechanical interfacing procedure to acquaint him with the technique.

The user wants to interface a programmed transfer type of peripheral to the external bus. He wants to use the cables provided with the Positive I/O Bus interface, and he intends to solve part of the interfacing problem by using M-series modules. Not only does he need a connector block into which he can insert either an M-series module, or the connector on cables 1, 2, and 3, but he also needs something on which to mount each connector block. DEC's H803 connector block, providing slots for eight modules, accepts the connectors, the M-series modules, and even DEC's A-, K-, and W-series of modules. To mount these connector blocks, H911 mounting panels can be used. This mounting panel contains eight H803 connector blocks and can be mounted in a standard 19-in. equipment rack. Figure 10-14 shows an H911 mounting panel containing two H803 connector blocks. The wire-wrap pins of the connector blocks face the front of the mounting panel. Cable 1 from the Positive I/O Bus interface is pictured as connecting into slot A1. This is in accordance with the convention presented previously. Cable 2 and cable 3 (omitted for clarity) would be inserted into slots A2 and A3, respectively.

If a data break peripheral were being used, cable 1 from the KD8-E interface would be inserted into slot A4, while data break cable 2 would be inserted into slot A5. The M-series module is shown as being inserted into slot A6. It may be inserted into any available slot, except B1 through B5. Finally, the peripheral I/O cable is shown as being inserted into slot A7 (again just an arbitrary assignment). The peripheral I/O cable must be equipped with a connector that is compatible with the H803 connector block. Either an M903 connector (for use with shielded Mylar) or an M904 connector (for use with coaxial cable) is recommended.

Slots B1, B2, and B3 should be wired in parallel with A1, A2, and A3 so that the I/O bus can be continued to additional peripheral controls. In general, module layout is primarily a matter of common sense. The convention just given is a standard with DEC and may or may not meet with the user's approval. Customers should, nevertheless, always attempt to make parallel connections of the Positive I/O bus interface cables to facilitate possible future expansion.

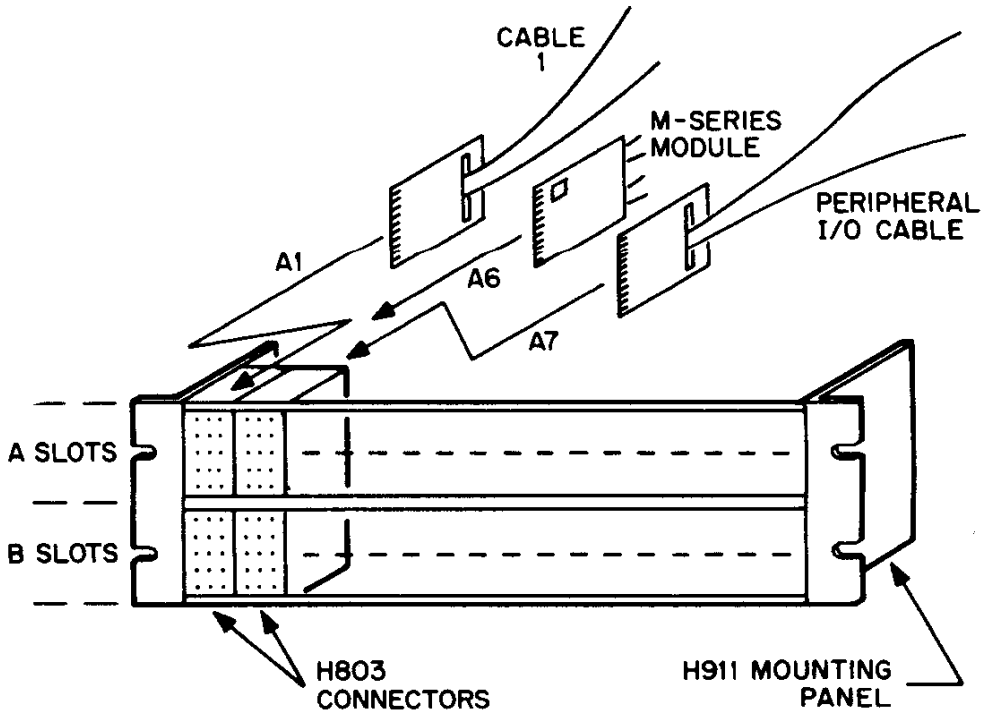


Figure 10-14 Interface Hardware

Connections between the various cable connectors and the M-series modules are made by selective wiring of the H803 connector blocks. The following suggestions and requirements are provided to help reduce the mounting panel wiring time. The connectors should be wired in the order given in steps 1 through 4.

1. The H803 mounting blocks have wire-wrap pins, thereby eliminating the problems associated with soldering, while at the same time providing highly reliable, long lasting connections. DEC recommends #30 AWG, Teflon-coated solid wire for connector block wiring. Smaller wire may be used if many connections are to be made to a single lug: A wire-wrapping tool must be used to wire the connector pins. DEC type H810 pistol grip hand wire-wrapping tool is designed for wrapping #24 or #30 solid wire on Digital-type connector pins (check the DIGITAL LOGIC HANDBOOK for further information about this tool and any other hardware mentioned in this chapter).

CAUTION

Whenever a wire-wrapping tool is used on a mounting panel containing modules, steps must be taken to avoid voltage transients that can burn out components. A battery-powered or air-operated tool is preferred, but even with these tools static charge can build up and burn out semiconductors. If the modules remain in the connector panel during wiring, ensure that the wire wrap tool is electrically grounded. Whenever soldering is done on a mounting panel containing modules, a 6V soldering iron should be used.

2. Certain connector pins on cable connectors and modules are reserved for specific functions. Cable connector pins are reserved as follows.

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, U2.
Not Used:	U1, V1, A2, B2.

Module pins are reserved as follows:

Positive dc voltage:	A2 (usually +5V)
Negative dc voltage:	B2 (usually −15V)
Ground:	C2, T1.

Some form of bus strip (such as DEC 933 Horizontal Bussing Strips) should be used to make all connector power connections and all horizontally bussed signal connections. Negative DC voltage should be wired to pin B2 of module connectors only if the voltage is required by the module.

3. Adequate grounding must be provided. The user should not be concerned with the question of ground loops. At the frequencies dealt with in digital logic, many parallel paths are of utmost importance. There must be ground continuity between cabinets, and between the logic assembly and any equipment with which the logic connects. Continuity between mounting panels and between ground pins on the various connector blocks is achieved when the following instructions have been carried out. These instructions are illustrated in Figure 10-15.
   a. Vertical grounding wires must interconnect each chassis ground lug with pin C2 and pin T1 grounds. Start these wires at the uppermost mounting panel and continue to the bottom panel. Begin by connecting C2 pins, then T1 pins, alternating thereafter. Space the wires about two inches apart, so that each of the chassis-ground lugs is in line with one of the wires. Each vertical wire should make three connections at each mounting panel.
   b. Connect pin C2 of each module to T1 of the same module, making connections to all other pins to be grounded along the way. Connect T1 of each module to C2 of the module beneath. Ensure that connections are made to the ground pins on the signal connectors.
   c. Bus ground pins horizontally wherever possible.

4. After ground connections have been made, connect all signal wires in any convenient order. Point-to-point wiring produces short wire lengths, installs quickly, is easy to trace and change, and generally results in better appearance and performance than cabled wiring.

   Certain restrictions and criteria must be observed when interfacing to the external bus. These are encountered when interfacing both with DEC modules and connector blocks, and with cus-

tomer-designed circuits. These requirements are contained in the section dealing with restrictions and criteria. Users are strongly advised to consult this section prior to completing their interface package.

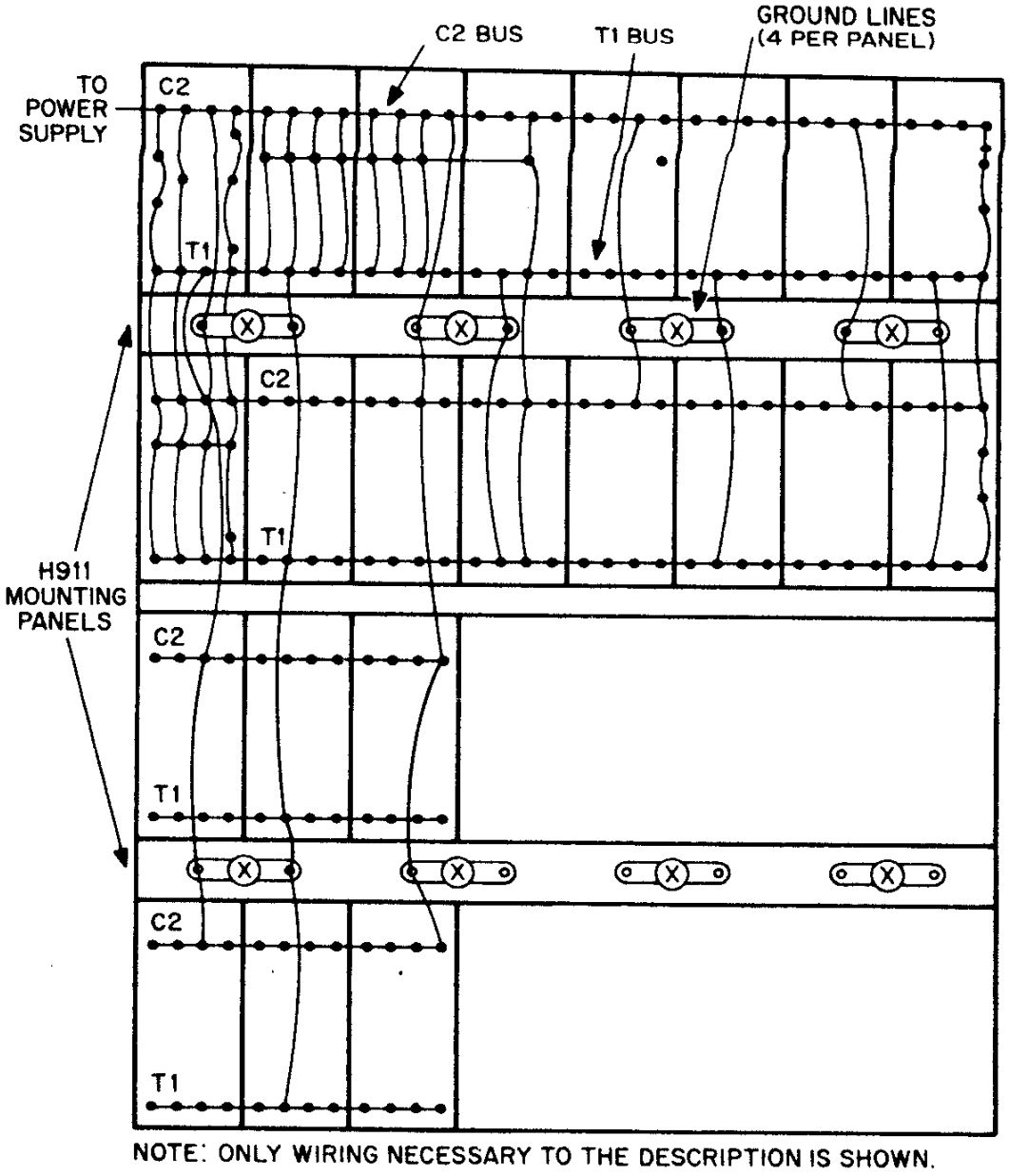


Figure 10-15 Mounting Panel Wiring

Customer Designed Interfaces

The customer who elects to design his own interface package must provide both electrical and mechanical interfaces. In addition to the requirements of the following section, the user should keep in mind the following definitions:

1. External bus signals are positive pulses or positive levels, allowing direct TTL-logic interfacing with appropriate diode clamp protection.

2. These positive pulses and levels change from ground (0V to 0.4V) to a positive voltage between 2.4V and 3.6V.
3. All signal lines are loaded within the two external bus interfaces in the PDP-8/E. Signals coming from the peripheral are inactive when a voltage potential is applied to them; conversely, signals going to the peripheral are inactive when no voltage potential is applied to them.

Figure 10-16 provides logic diagrams of the circuits to which the user must interface. The details of the method used are left to the customer.

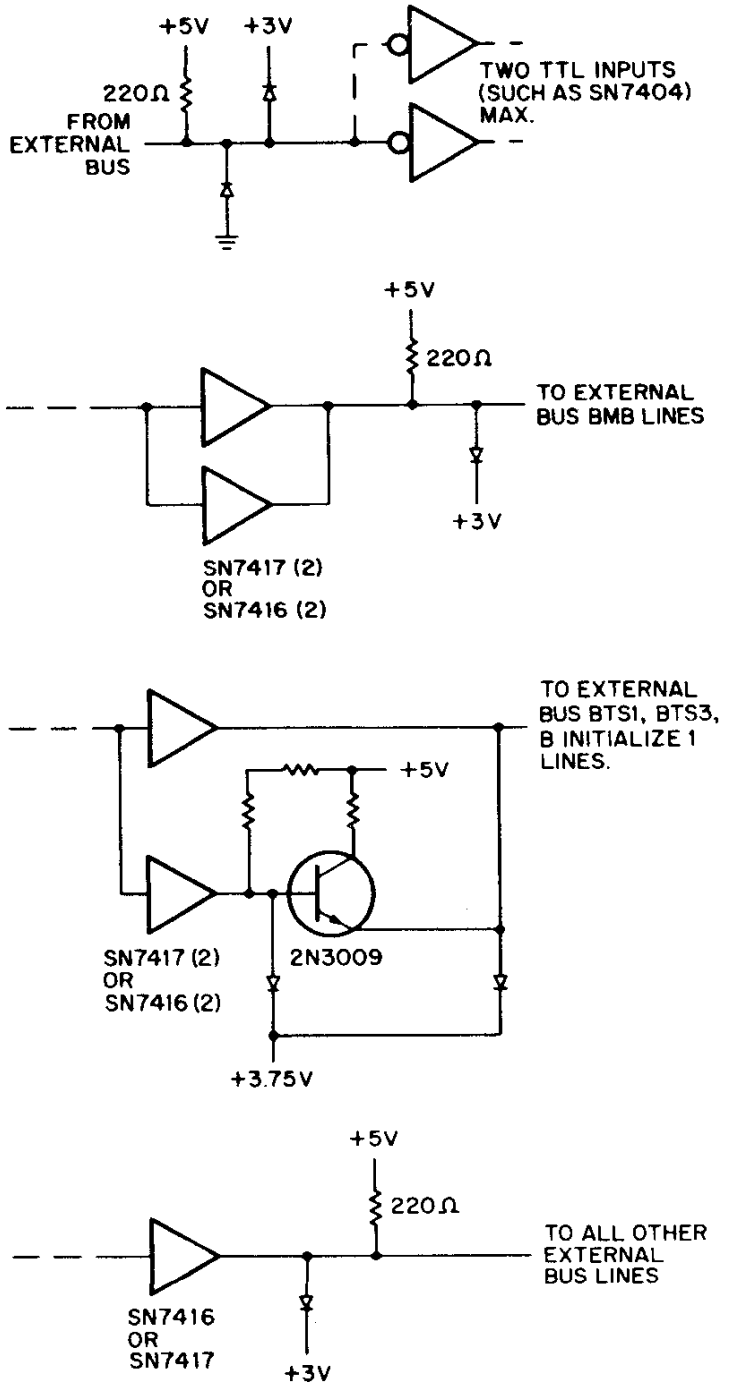


Figure 10-16 Interface Logic Diagrams

RESTRICTIONS AND CRITERIA

Cooling

The low power consumption of M-series modules results in a total dissipation of about 15W in a typical H911 mounting panel containing 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.

Signal Terminating

Termination is required on Positive I/O Bus interface cables longer than 20 feet, and may be desirable on shorter cables. The following signals should be shunted to ground by a 100 ohm resistor: IOP1; IOP2; IOP4; BTS1; BTS3; B INITIALIZE 1. If a series of peripherals is being used, the termination is inserted in the last peripheral. A DEC G717 resistor terminator module is available for this purpose.

Timing Criteria

Timing criteria must be considered only when peripherals having high operating speeds (over 5kHz) are being interfaced. The following information concerning interrupt processing must be understood.

a. The interrupt feature must be turned on via the ION instruction in order for the device to be allowed to interrupt the processor.

b. In order to honor the interrupt, the central processor must have completed the instruction it is presently doing.

c. When an interrupt request is honored, the hardware of the machine executes an effective JMS to location 0 in memory field 0, and also disables the interrupt system.

d. An interrupt servicing routine must be resident in memory and the starting address of this routine must be defined in the memory location immediately following location 0.

The longest time required to honor an interrupt request is approximately the time duration of the slowest instruction. Thus, for a PDP-8/E without the EAE option, this time would be 4.3 microseconds (the time required to complete a 3-cycle instruction). For the PDP-8/E with the EAE option, the time would be 9.8 microseconds. These times assume an interrupt request just after the processor enters the FETCH state.

The following examples illustrate the use of this timing criterion.

EAMPLE 1—PDP-8/E without EAE option.

Time between interrupts	:	50.0 μs
Maximum processor time before interrupt	:	—4.3 μs
Time for hardware JMS to location 0	:	—1.4 μs
Maximum time allowed for servicing before possible error arises	:	44.3 μs

EXAMPLE 2—PDP-8/E with EAE option.

Time between interrupts	:	50.0 μs
Maximum processor time before interrupt (with EAE option installed—24-bit long shift)	:	—9.8 μs

Time for hardware JMS to location 0	:	—4.3 μs
Maximum time allowed for servicing before possible error arises	:	35.9 μs

Another timing criterion is concerned with peripheral gating time from IOP to SKIP, from IOP to AC input signals, and from IOP to AC CLEAR. To avoid time delay problems, these gating times must be limited to 100 nanoseconds.

The third timing criterion is concerned with the delays inherent in interconnecting cabling. DEC logic generates waveforms with rising edges containing frequencies of over 100 MHz. At these frequencies the inductance, mutual inductance, capacitance, and tramission line properties of the external bus cabling become significant. To avoid problems, consider the following when interfacing.

a. The propagation delay of typical wiring (1.5 nanoseconds/ft) is often significant when overshoot and reflections are considered.

b. The current carrying capacity of a wire is only V/Z(0) until the wave has propagated along the wire three times. Typical wiring has a characteristic impedance of approximately 150 ohms, so that the current available at the end of the wire for rising waveforms is only 20 milliamps until reflections have propogated, regardless of the source current available.

c. The inductance and capacitance of wiring combine to produce high frequency ringing on the transitions of waveforms. This ringing can be controlled by resistively terminating the line with approximately 100 ohms.

d. The mutual inductance and capacitance of the wiring causes high-frequency crosstalk which may produce false operation of the logic. This crosstalk can be reduced in one of the following ways; minimizing the number of high frequency signal components by clipping or clamping high-frequency ringing with a level terminator circuit, or wiring with short wires and/or twisted pairs, thereby reducing coupling. Clamping can also be used to prevent the excursion of the output or input voltages beyond certain predetermined limits. This is sometimes necessary to prevent false triggering or electrical damage to gates.

CABLING RULES AND SUGGESTIONS

1. Round and flat coaxial cable are electrically interchangeable and may be intermixed in a system. If cables will be subjected to extraordinary abuse, round coaxial cable is preferable when connecting free-standing cabinets.

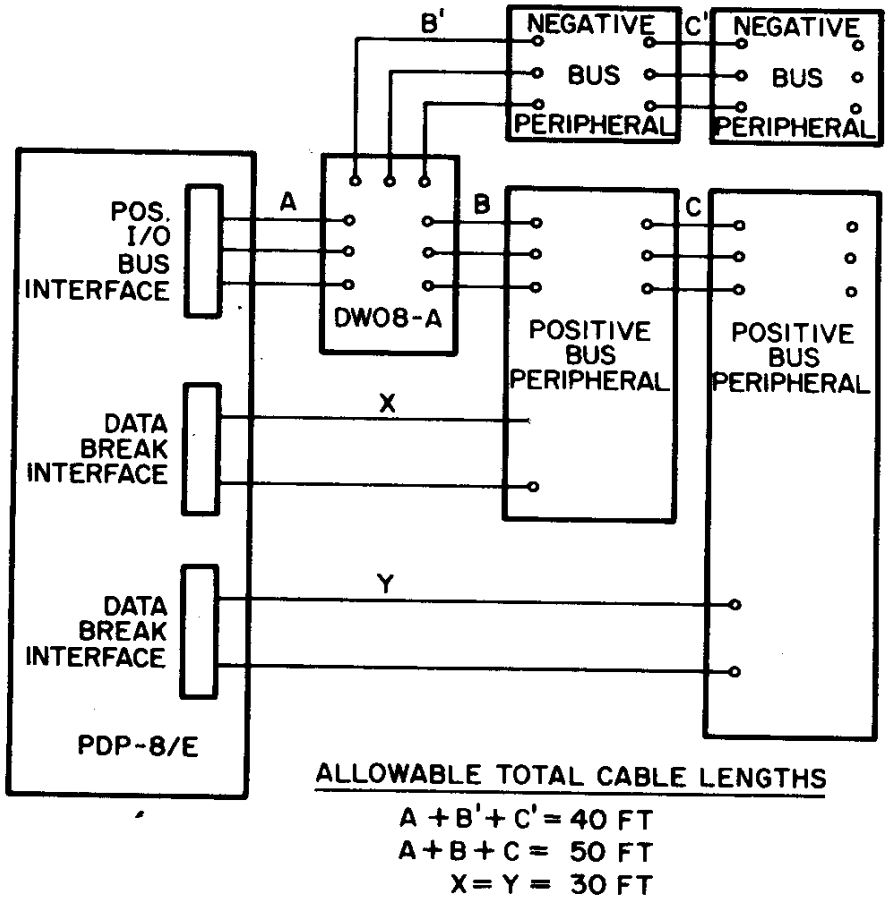
2. Indiscriminate mixing of shielded flat cable and coaxial cable is not advised. DEC recommends that all cables be shielded flat cables, except when the user is trying to gain maximum length or is connecting free-standing cabinets. Not more than one change from BC08-J to coaxial, or vice-versa, should be made over the length of a bus.

3. The following cable length restrictions should be observed:

CABLE	TYPE	MAXIMUM LENGTH
1, 2, and 3 from the Positive I/O Bus interface	Coaxial	50 ft

If a DW08-A I/O converter panel is connected onto cables 1, 2, and 3, the system can accommodate negative bus periphorals. The normal positive bus maximum cable lengths remain as indicated; the maximum cable lengths for the converted bus (negative) are 10 feet shorter than that indicated. See Figure 10-17.

1 and 2 from the Data break interface	Coaxial	30 ft



NOTE: LENGTHS GIVEN FOR COAXIAL CABLE; EACH IS A MAXIMUM ALLOWABLE LENGTH.

Figure 10-17 Maximum Bus Lengths

4. DEC cable has the nominal characteristics listed below. The user should ensure that whatever cable he uses exhibits approximately the same characteristics.

$Z = 95 +$ or $-$ 5 ohms
$C = 13.75$ pF/ft (unterminated)
$L = 124$ nH/ft
$R = 0.095$ ohm/ft
$V(p) = 1.5$ ns/ft

5. The cables supplied with the Positive I/O Bus interface and the Data Break interface can be obtained in standard length, as outlined in table 10-3.

Table 10-3 Table Of Standard Cable Lengths

CABLE	LENGTH
BC08J-6	6 feet
BC08J-10	10 feet

SECTION 2 OMNIBUS INTERFACING USING "OFF THE SHELF" MODULES

Interfacing to the PDP-8/E OMNIBUS can be accomplished conveniently by utilizing M Series driver and receiver modules. The low-leakage current requirements for devices "wire ORed" to the OMNIBUS signals are met completely when the M783, M784, and M785 modules are used.

This approach takes advantage of Digital's broad line of proven interface modules. Customers who are designing new interfaces or who have existing interfaces using TTL Logic levels may tie them directly to the OMNIBUS via this procedure. The attractiveness of this procedure lies in the availability of fully engineered and warranted modules.

OMNIBUS SIGNAL SUMMARY:

Most OMNIBUS lines are considered by the system to be inactive (voltage level high) until the line level is pulled to ground. Logic levels on these lines are defined as:

Logic 1—Max. Voltage: 0.4 V
Min. Voltage: −0.5 V
Logic 0—Max. Voltage: 5.0 V
Min. Voltage: 3.0 V

Signal levels on the OMNIBUS may be converted to or from TTL levels with the following modules:

Bus Receiver —M784
Bus Driver —M783
Bus Transceiver —M785

THE "BUILDING BLOCK" APPROACH

A block diagram illustrating a system relationship is shown in figure 10-18. The M783 and M784 Driver and Receiver or the counterpart—the M785 Transceiver are shown as the necessary prerequisites to establish OMNIBUS compatibility. The connecting link between the OMNIBUS and the interfacing M modules is with two M935 Bus connectors which interconnect the last slot of the OMNIBUS to the first slot of the H9190 assembly. The user interfacing options are immediately expanded by interconnecting the slots on the H9190 assembly to Bus Drivers and Bus Receivers and interconnecting the Bus Drivers and Bus Receiver modules to other M or K series modules. There are more than fifty M series modules to choose from and scores of K series modules.

M783—BUS DRIVERS

The M783 module (represented in figure 10-19) consists of 12 two-input NAND gates with open-collector outputs. The gates are grouped into a set of 8 with a common enable line and 4 individual gates. Each output is capable of sinking 50 mA while maintaining a collector voltage of $\leq$0.8V. The output leakage current is $<$25 μA. All gate inputs are TTL compatible.

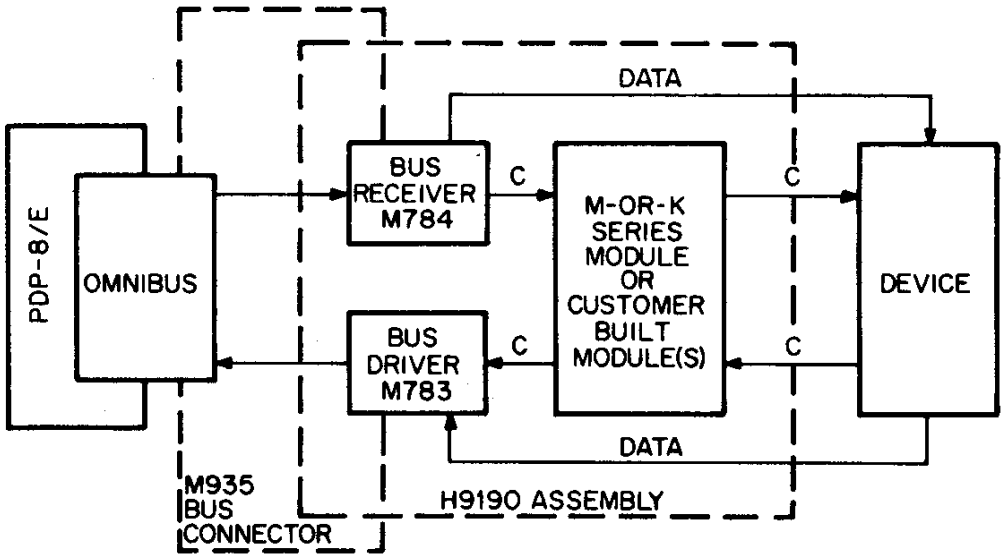


Figure 10-18 Typical System Interface Block Diagram

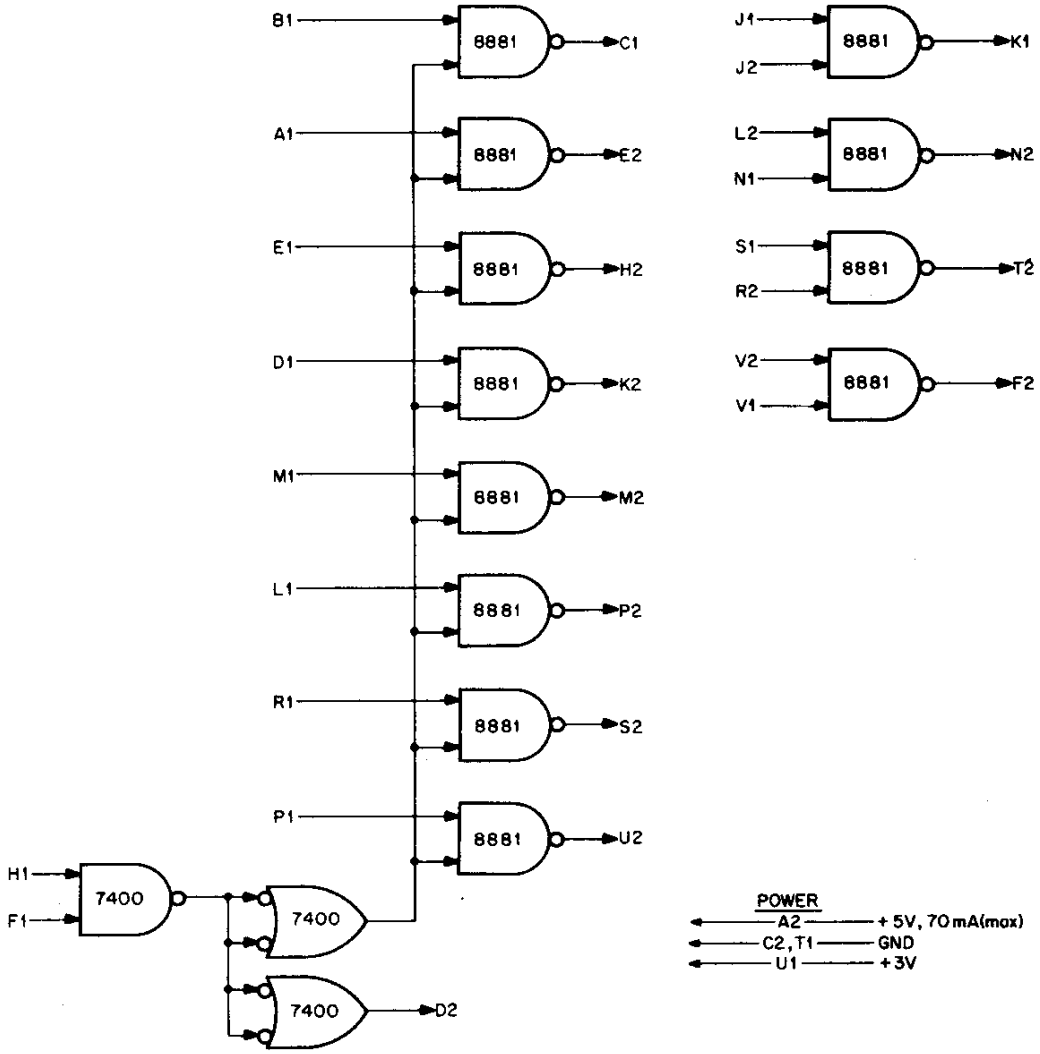


Figure 10-19 M783 Bus Drivers

M784—BUS RECEIVERS

The M784 module (represented in figure 10-20) has 16 inverting receiver circuits constructed of two-input NOR gates with the common enable line grounded. Inputs are characterized as:

Low Level: $<$ 1.4 V at 25μA (max)
High Level: $>$ 2.5 V at 160 μA (max)

All gate outputs are TTL compatible with a fan-out from each of 7 TTL loads.*

*One unit load is defined as:
Logic 0—sink 1.6mA with V out $\leqslant$ 0.4 V
Logic 1—supply 40.0 μA with V out $\geqslant$ 2.4 V

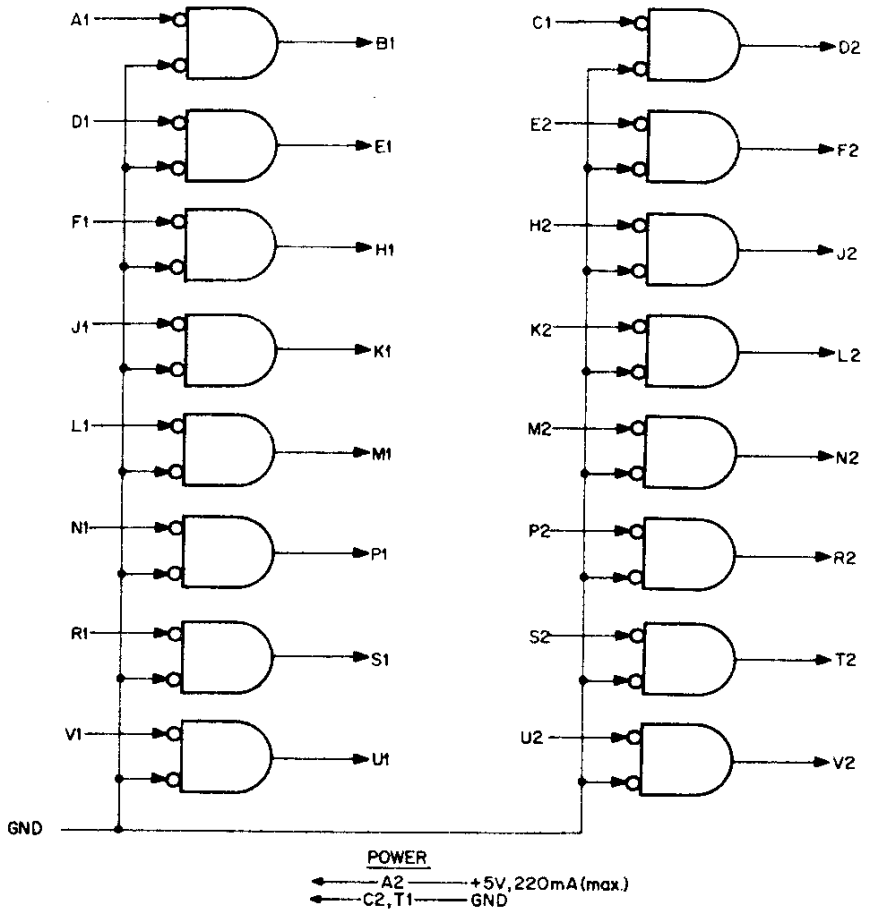


Figure 10-20 M784 Bus Receivers

M785—BUS TRANSCEIVER

This composite module consists of 8 drivers and 8 receivers. Each set of 8 gates has a common enable line, convenient for strobing data to and from the OMNIBUS. The loading characteristics of the devices are identical to their M783 and M784 counterparts.

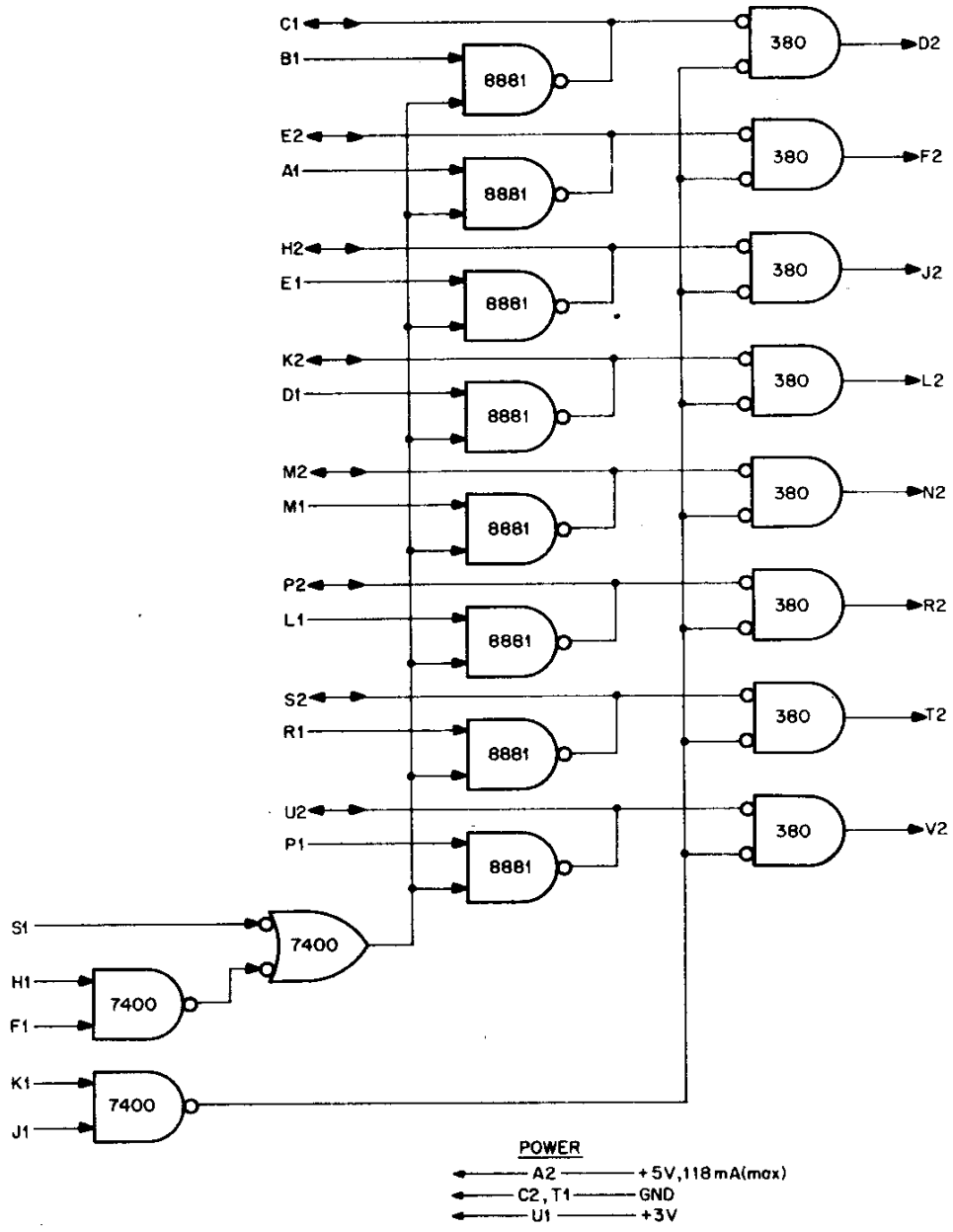
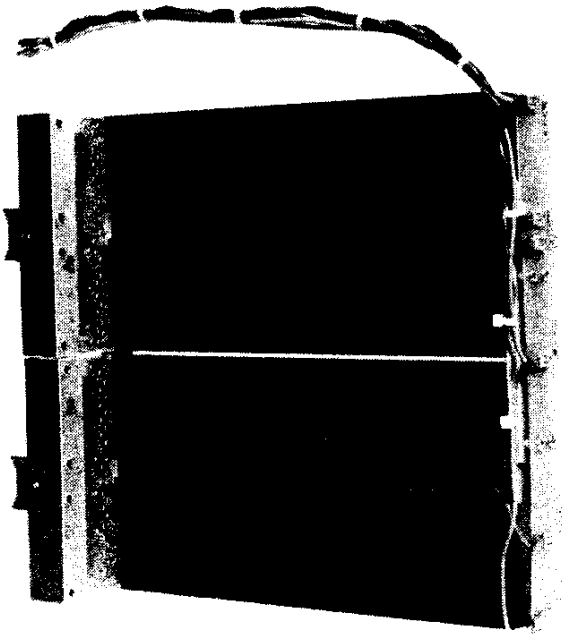
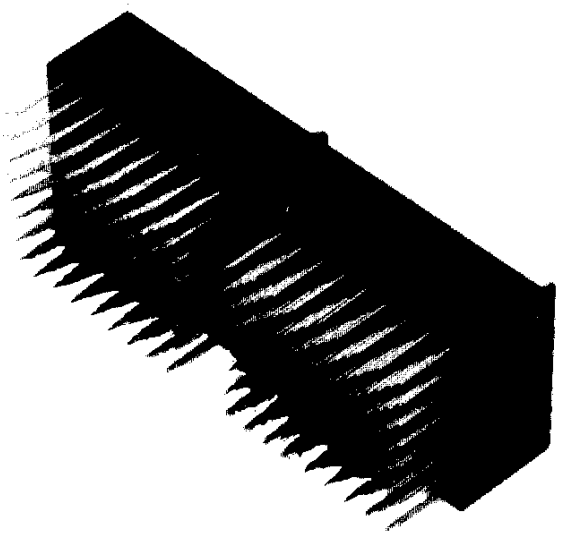


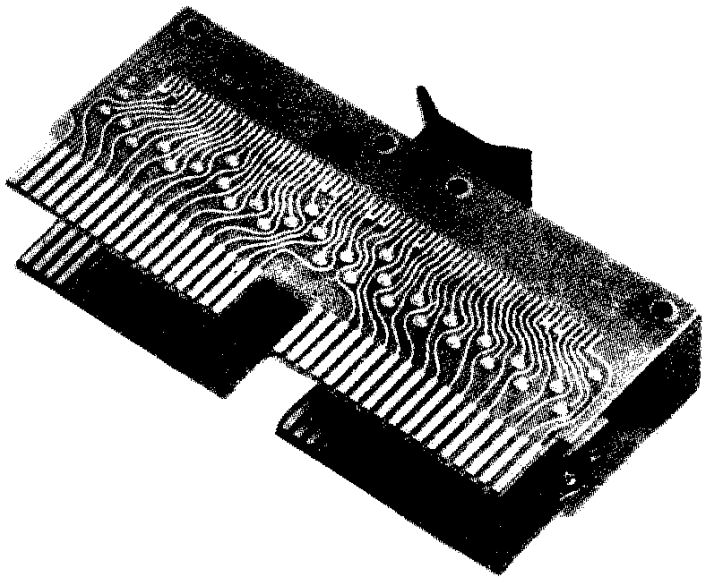
Figure 10-21 M785 Bus Transceiver



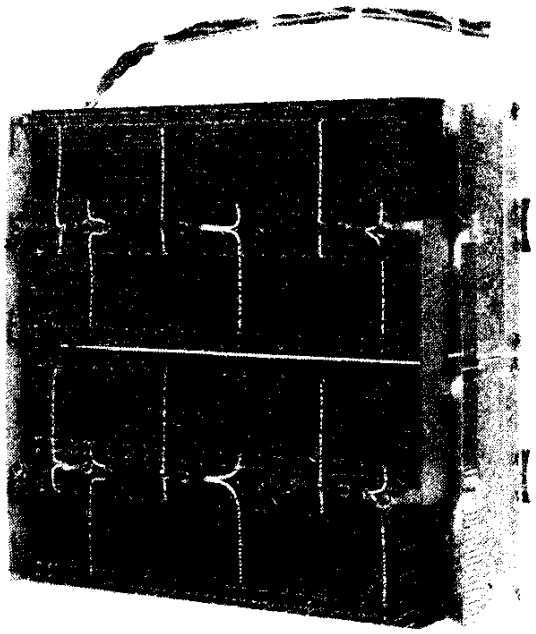
H9190 M935 Kit—contains the H9190 assembly with M Series connector blocks for standard M Series modules, power wiring harness, and power bus board. It includes M Series power bussing for all but the four lots in the first column. Also included are two M935 bus connectors. Four mounting spacers allow the H9190 to be easily mounted in the second half of an 8/E chassis.



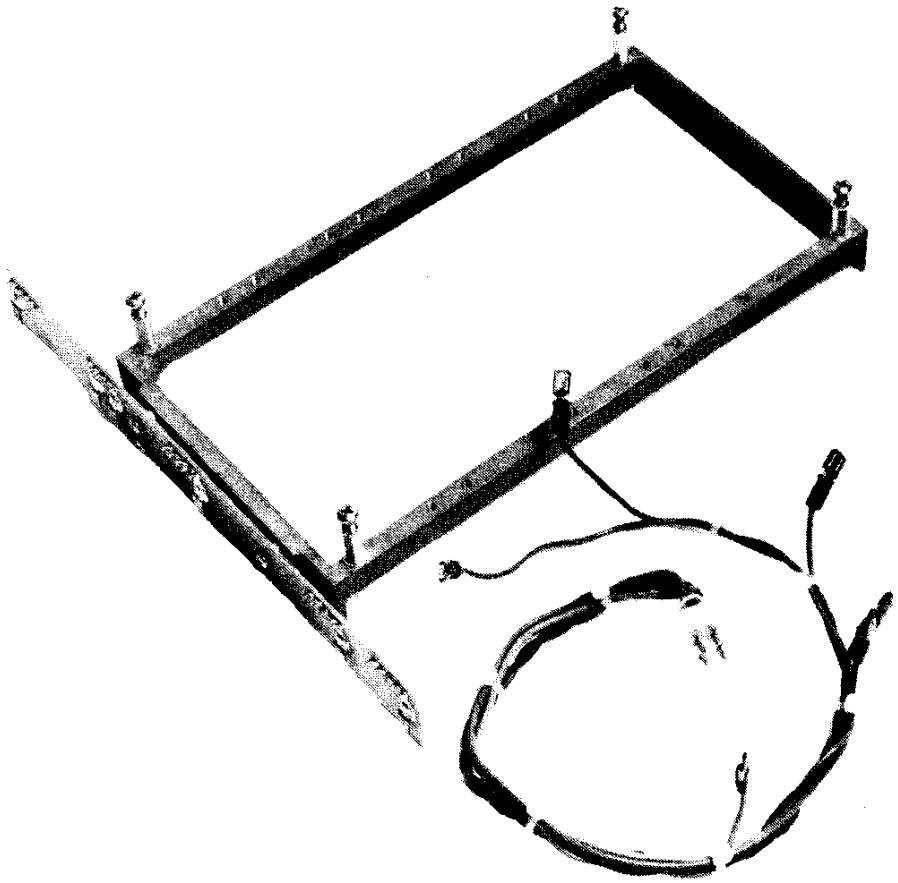
H803 Connector Block—a high density, 8-slot connector block with wire wrap pins. This connector is designed to be used with M Series modules.



M935 Bus Connector—used to interconnect 8/E assemblies. The H9190 may be connected to the 8/E OMNIBUS using two M935's.



H9190 Mounting Panel—contains M Series connector blocks with 8/E type packaging for standard M Series modules. Also included are the 8/E power wiring harness and power bus board. There is M Series power bussing for all but the four slots in the first column. Four mounting spacers allow the H9190 to be easily mounted in the second half of an 8/E chassis.



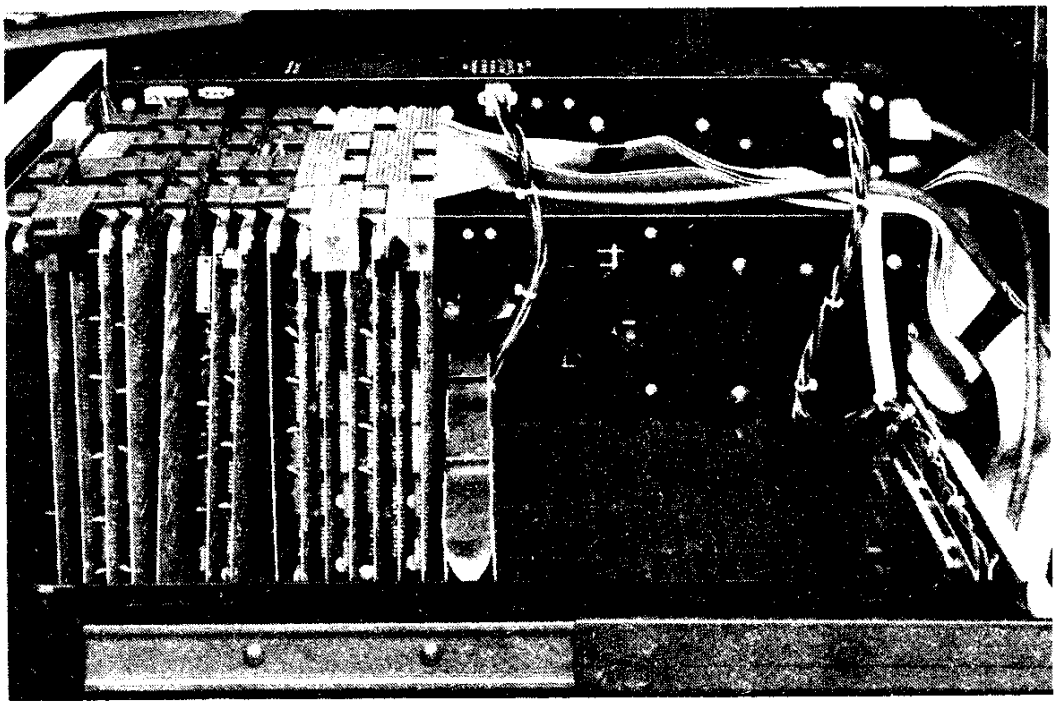
H019 Mounting Bar—an aluminum casting with the power bus board and power wiring harness. It also includes four mounting spacers for mounting in an 8/E chassis. Up to ten connector blocks of any type may be accommodated by this frame.

PHYSICAL PLACEMENT OF INTERFACE MODULES

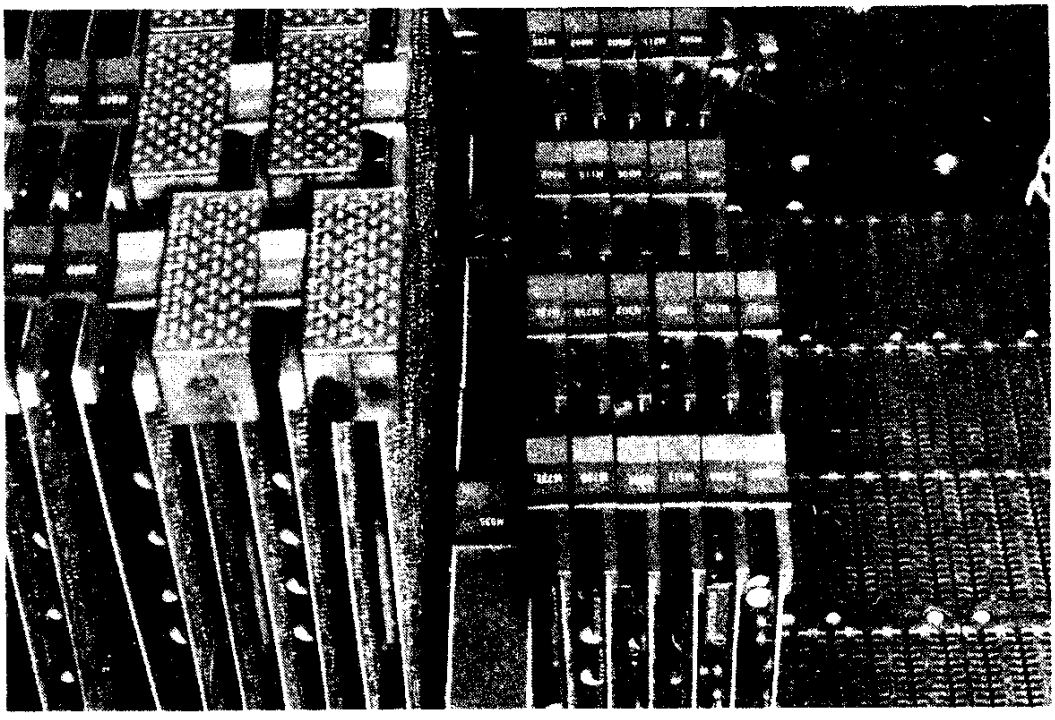
All pins of the OMNIBUS back-panel are dedicated to specific signal lines. For this reason any interface modules that are not pin-compatible cannot be inserted directly into the OMNIBUS. These modules should be plugged into type H803 connector blocks mounted externally to the OMNIBUS. This does not imply mounting externally to the PDP-8/E box, however. There is normally adequate mounting space avaliable within the box itself for medium sized interfaces.

There are two general methods of interconnection to the OMNIBUS from the external logic. One method is to use a H9190 Mounting Panel connected to the OMNIBUS via 2 M935 bus connectors. This panel contains 10 H803 connectors and is physically similar to the BE8-A OMNIBUS Expander assembly, the difference being that the back-panel is wire wrappable rather than being bussed. Mechanical mounting is to existing BE8-A supports within the PDP-8/e box. A power cable is provided for connection to the PDP-8/E power supply. This supply was designed to support internal interfaces. It can supply up to 13 amps at +5 Vdc. 3.5 amps at −15 Vdc and 0.2 amps at + 15 Vdc.

The H019 Mounting Bar will mount up to 10 H803 connector blocks. It mounts in the same manner as the H9190 and includes the power cable.



The H9190 Assembly shown connected to the OMNIBUS via the M935 Bus connector



The H9190 Assembly shown with 21 M series modules and many unused slots.

A second method of connection is to mount the H9190 in a BA8-AB expander box. Connection to the OMNIBUS in this case is via type BC08H-3F flexprint cables. Use of this mounting method is necessary only with large systems that mount the BE8-A OMNIBUS Expander in the 8/E box.

INTERFACE EXAMPLE-PAPER TAPE READER

The objective is to allow a PDP-8/E to read 8-bit code from a paper tape reader. The design utilizes the computer's Interrupt and Skip facility in order to minimize the time required to service the reader. All logic functions are performed using "off the shelf" M Series modules.

Input/Output Transfer (IOT) Instruction Usage

IOT 6641—Skip if RDR FLAG set by DATA STROBE
IOT 6642—Load Data onto DATA and C0-C1 lines and reset RDR FLAG: (e.g. load data into AC and clear interrupt request)
IOT 6643—Reset RDR FLAG (e.g. clear interrupt request)
IOT 6644—Unused
IOT 6645—Unused
IOT 6646—Set RDR RUN flip-flop (e.g. initiate read char.)

System Operation

Initially the RDR FLAG is reset by IOT 6643 to clear the interrupt line. IOT 6646 initiates a chain of operations that issues the reader motor drive signals and times the DATA STROBE to load hole sense data into the interface register. The DATA STROBE signal also sets the RDR FLAG which generates an interrupt to the 8/E. The condition of RDR FLAG can be monitored by using IOT 6641 as a Skip IOT. Once the RDR FLAG is set, IOT 6642 will load the register data into the accumulator of the 8/E and clear the interrupt line. The above procedure is repeated for reading of each character from the tape.

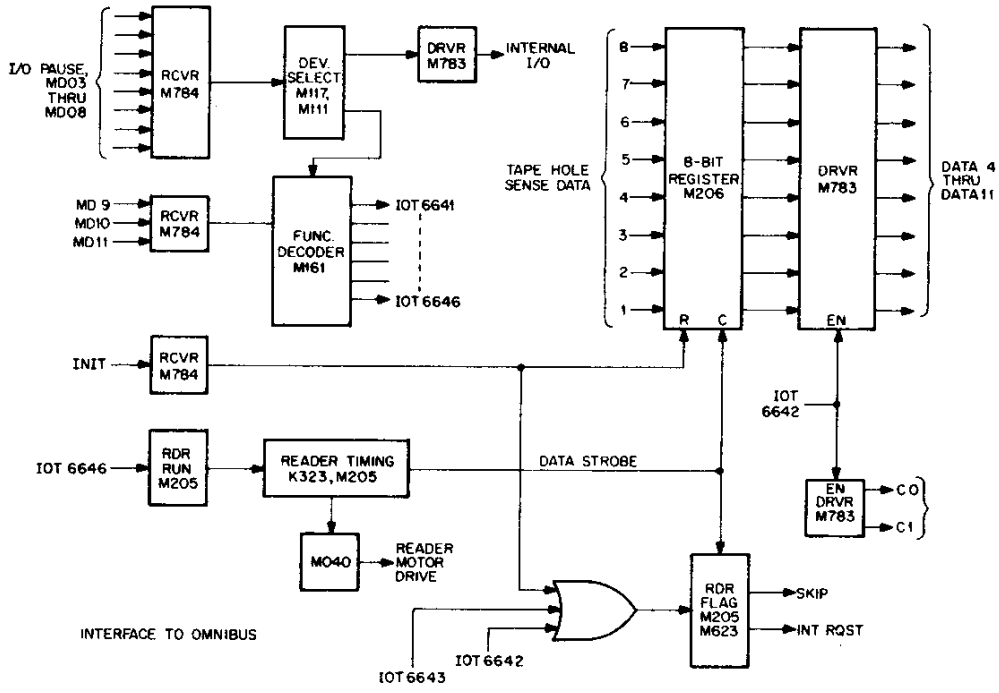


Figure 10-21 Block Diagram Paper Tape Reader Interface to OMNIBUS

M-SERIES MODULE SUMMARY

The M Series TTL integrated circuit modules consists of more than 95 modules ranging from basic and functional logic modules to self-contained computer interfacing modules for applications such as instrumentation, computer interfacing, data gathering, control, etc.

DEC is also offering a new M Series Logic Lab for use in breadboarding M Series logic designs. This new M Series Logic Lab is used in education as a training device which offers the user an easy step-by-step method to gain an understanding of various logic functions such as AND, OR, NAND, NOR, etc. The breadboard and testing capability of the Logic Lab is an effective tool for bridging the gap between paper design and a fully tested, marketable product. For detailed information, the reader should acquire a free copy of DEC's 300 page Logic Handbook. Please write to Direct Mail, Digital Equipment Corporation, 146 Main Street, Maynard, Massachusetts 01754 and ask for a copy of the Logic Handbook if you do not already have one.

The following is a partial 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 DEC's DIGITAL LOGIC HANDBOOK. For modules that cannot be found in the handbook, contact the nearest Digital representative for information.

TYPE	FUNCTION	DESCRIPTION
M002	15 Loads	Fifteen +3V sources each capable of driving 10 unit loads. Can be used for tying off unused inputs.
M040	Solenoid Driver	Output ratings of −70V and 0.6A allow these 2 drivers to be used with a variety of medium current loads.
M050	50 ma Indicator and Driver	Output ratings of −20V and 50 milliamps allow any of the 12 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 −3V system.
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 from the PDP-8/I, PDP-8/L, or PDP-8/E I/O bus. Pins are compatible with the M111 module.
M103	Device Selector	Diode gate, buffering and clamping circuits necessary to decode IOT's from the PDP-8/I, PDP-8/L, or PDP 8/E positive bus. Output pulses are not regenerated, only buffered.
M111	Inverter	Sixteen inverter circuits with a fan-in of 1 unit load and fan-out of 10 unit loads.

TYPE	FUNCTION	DESCRIPTION
M112	NOR Gate	Ten positive NOR gates with a fan-in of 1 unit load and fan-out of 10 unit loads.
M113	Ten 2-Input NAND Gates	Ten 2-input positive NAND gates with a fan-in of 1 unit load and fan-out of 10 unit loads.
M115	Eight 3-input NAND Gates	Eight 3-input positive NAND gates with a fan-in of 1 unit load and a fan-out of 10 unit loads.
M117	Six 4-Input NAND Gates	Six 4-input positive NAND gates with a fan-in of 1 unit load and a fan-out of 10 unit loads.
M119	Three 8-Input NAND Gates	Three 8-input positive NAND gates with a fan-in of 1 unit load and a fan-out of 10 unit loads.
M121	AND/NOR Gates	Six gates that perform the positive logic function AB + CD. Fan-in on each input is 1 unit load and gate fan-out is 10 unit loads.
M141	NAND/OR Gates	Twelve 2-input positive NAND gates that can be used in a wired OR manner. Gates are grouped in a 4-4-3-1 configuration with a fan-in of 1 unit load and a fan-out that depends on the number of gates ORed together.
M160	Gate Module	Three general purpose multi-input gates that can be used for system input selection. Fan-in is 1 unit load and fan-out is 10 unit loads.
M161	Binary to Octal/Decimal Decoder	A binary to 8-line or BCD to 10-line decoder. Gating is provided so that up to 6 binary bits can be decoded using only M161s. 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 that can be used for input selection. Each circuit is of an AND/OR configuration with four 2-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 on module pins.

TYPE	FUNCTION	DESCRIPTION
M203	Set-Reset Flip-flops	Eight single input set-reset flip-flops for use as buffer storage. Each circuit has a fan-in of 1 unit load and a fan-out of 10 unit loads.
M204	Counter-Buffer	Four J-K flip-flops that can be interconnected as a ripple or synchronous counter or used as general control elements.
M206	Six Flip-flops	6 D-type flip-flops that can be used in shift registers, counters, buffer registers, and general purpose control functions.
M207	Flip-flops	Six single-input J-K flip-flops for use as shift registers, ripple counters, and general purpose control functions.
M208	Buffer Shift	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 6-bit binary up/down ripple counter with control gates for direction changes via a single control line.
M212	6-Bit 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.
M230	Binary to BCD Shift Register Converter	One decade of a modified shift register that allows high speed conversion (100 nanoseconds per binary bit) of binary data to 8421 BCD code. System use of this module requires additional modules.
M302	One Shot Delay	Two pulse-or-level-triggered one-shot delays with output delay adjustable from 50 nanoseconds to 7.5 milliseconds. Fan-in is 2 unit loads and fan-out is 25 unit loads.
M310	Delay Line	Fixed tapped delay line with delay adjustable in 50 nanoseconds increments from 50 nanoseconds to 500 nanoseconds. Two digital output amplifiers and one driver are included.

TYPE	FUNCTION	DESCRIPTION
M360	Variable Delay	Continuously variable delay line with a range of 50 nanoseconds to 500 nanoseconds. Module includes delay line drivers and digital output amplifiers.
M401	Clock	A gateable RC clock with both positive and negative pulse outputs. The output frequency is adjustable from 10 MHz to below 100 Hz.
M405	Crystal Clock	Stable system clock frequencies from 1 kHz to 10 MHz are available with this module. Frequency drift at either the positive or negative pulse output is less than 0.01 percent of the specified frequency.
M410	Reed Clock	A stable low frequency reed clock similar to the M452. Stability in the range 10 degrees C to 70 degrees C is better than 0.15 percent. For use with communications systems and available with only standard teletype and data set frequencies.
M452	Variable Clock	Provides 880Hz, 440Hz, and 220Hz square waves necessary for clocking and for the M706 and M707 modules in a 110-baud teletype system.
M501	Schmitt Trigger	Provides 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 nanoseconds can be level shifted from —3V 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 levels shift pulses as short as 100 nanoseconds from —3V systems to M-Series levels. Each of the 6 circuits on this module has a 10 milliamp load resistor on the negative input.
M507	Bus Converter	Six inverting level shifters that accept —3V and GND as inputs and have an open collector NPN transistor at the output. Output rise is delayed by 100 nanoseconds for pulse spreading.

TYPE	FUNCTION	DESCRIPTION
M516	Positive Bus	Six 4-input NOR gates with overshoot and undershoot clamps on one input of each gate. In addition, one input of each gate is tied to +3V with the lead brought out to a connector pin.
M602	Pulse Generator	The two pulse amplifiers in this module provide standard 50 nanoseconds or 110 nanoseconds pulses for M-Series systems.
M617	Six 4 Input NOR Buffers	Six 4-input positive NOR gates with a fan-in of 1 unit load and a fan-out of 30 unit loads.
M627	Power Amplifier	Six 4-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	The three non-inverting level shifters on this module can be used to interface the positive levels or pulses (duration greater than 100 nanoseconds) of K- and M-Series to −3V logic systems.
M652	Negative Output Converter	These two circuits provide high-speed, non-inverting level shifting for pulses as short as 35 nanoseconds or levels from M-Series to −3V systems. The output can drive low impedance terminated cables.
M660	Positive Level Driver	Three circuits provide low impedance, 100-ohm, terminated cable driving capability, using M-Series levels or pulses of duration greater than 100 nanoseconds. Output drive capability is 50 milliamps at +3V or ground.
M661	Positive Level Driver	Three circuits provide low-impedance unterminated cable driving. Characteristics are similar to M660 with the exception that +3V drive is 5 milliamps.
M730	8 Bus Positive Output Interface	General-purpose positive bus output module for use in interfacing many positive level (0 to +20V) systems to the PDP-8/I, PDP-8/L, or PDP-8 E. Module includes device selector, 12 bit parallel output buffer, and adjustable timing pulses.
M731	8 Bus Negative Output Interface	Identical to M730, except that outputs are level shifted for 0 to −20V systems.

TYPE	FUNCTION	DESCRIPTION
M732	8 Bus Positive Input Interface	General purpose positive bus input module for interfacing many positive level (0 to +20V) systems to the PDP-8/I, PDP-8/L, or PDP-8/E. Module includes device selector, 12 bit parallel input buffer, and adjustable timing pulses.
M733	8 Bus Negative Input Interfacer	Identical to M732 except that inputs are level shifted from negative voltage systems.
M734	I/O Bus Input Multiplexer	The M734 is a double height, single width module and is a three-word multiplexer used for strobing twelve-bit words on the positive voltage input bus; usually the input of the PDP8/I or the PDP8/L. Device selector gating is provided. The data outputs of the M734 Multiplexer consist of open collector npn transistors which allow these outputs to be directly connected to the bus. All inputs present one TTL unit load and function as follows:
M735	I/O Bus Transfer Register	The M735 provides one 12-bit input bus driver and one 12-bit output buffer register for input and output data transfers on the positive I/O bus of either a PDP8/I or a PDP8/L. Device selector gating plus additional signal lines provide the flexibility necessary for a completet interface with the exception of flag sense signals. Use of the M735 is not restricted to a computer, as it can be used in many systems to provide reception and transmission of data over cables.
M737	12-Bit Bus Receiver Interface	The M737 was designed primarily to receive and store in a buffer register twelve parallel data bits from the positive bus of the PDP-8/I or PDP-8/L. The M737 is pin compatible with the M738 Counter-Buffer Interface, the M107 Device Selector, the M108 Flag Module, and the 12-Bit Bus Paneloid E100. The 12-bit Bus Receiver Interface, M737, consists of three basic sections: device selector, flag, and buffer register section.
M738	Counter-Buffer Interface	The M738 was designed primarily to strobe twelve parallel bits onto the positive bus of the PDP-8/I or PDP-8/L. This module consists of three basic sections:

TYPE	FUNCTION	DESCRIPTION
		1) A twelve-bit bus driver. 2) A twelve-bit Up Counter which is presetable by jam transfer, and 3) A clock input gate circuit twelve-bit bus driver.
M783	Bus Drivers	The M783 consists of 12 two-input NAND gates with open-collector outputs. The gates are grouped into a set of 8 with a common enable line and 4 individual gates. Each output is capable of sinking 50 mA while maintaining a collector voltage of $\leq$0.8V. The output leakage current is $<$25 μA. All gate inputs are TTL compatible.
M784	Bus Receivers	This module has 16 inverting receiver circuits constructed of two-input NOR gates with the common enable line grounded. Inputs are characterized as: Low Level: $<$ 1.4 V at 25 μA(max) High Level: $>$ 2.5 V at 160μA(max) All gate outputs are TTL compatible with a fan-out from each of 7 TTL loads.*
M785	Bus Transceiver	This composite module consists of 8 drivers and 8 receivers. Each set of 8 gates has a common enable line, convenient for strobing data to and from the OMNIBUS. The loading characteristics of the devices are identical to their M783 and M784 counterparts.
M901	Flexprint Cable Connector	Double-sided 36-pin shielded mylar cable connector. All pins are available for signals or grounds. Pins A2, B2, U1, and V1 have 10 ohm resistors in series.
M902	Resistor Terminator	Double-sided 36-pin terminator module with 100 ohm terminations on signal leads. Alternate grounds are provided as in the M903 and M904.
M903	Connector	Double-sided 36-pin shielded mylar cable connector with alternate grounds for I/O bus cables.
M906	Cable	Eighteen load resistors clamped to prevent excursions beyond +3V and ground. This terminator can be used in conjunction with the M623 to provide cable driving ability.

K SERIES MODULES

Another very important variety of "off-the-shelf" modules is the K series module. These are used in, but not limited to control applications. The number of applications using these modules runs into the hundreds. Representative applications include:

- Computer Based Data Aquisition
- Computer Based Control Systems
- Multiprocessor Systems
- Industrial Data Acquisition and Control
- Analog-to-Digital Conversion and Multiplexer Subsystems
- Digital Input and Output Subsystems
- Gas Chromatography Systems
- N/C Tape Preparation Systems

The combination of the M and K series modules using the "building block" approach with "off-the-shelf" modules is an ideal method of interfacing to the PDP-8/E processor for control applications. For more information and detailed examples, the reader should acquire a free copy of DEC's Control Handbook containing more than 200 pages of instructive material in the field of industrial control.

CHAPTER 11

INSTALLATION AND PLANNING

SPACE REQUIREMENTS

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

The PDP-8/E is available in two configurations, rack-mounted and table-top. Each offers many advantages that should be considered when planning the type of system desired.

The table-top PDP-8/E (see Figure 11-1) requires much less space than the rack-mounted version. The user may, if he desires, place the table-top model on a corner of his desk and operate the computer right in his office. This places the control panel within easy reach of someone seated at the desk. No special cooling is required, and, in most cases, a standard office electrical outlet provides adequate power. This table-top version is approximately 10½ in. high, 19 in. wide and 24 in. deep. It weighs approximately 100 pounds, and is made up of the central processor, memory, console, power supply, and chassis assembly with top cover. The chassis contains an OMNIBUS, an H724 Power Supply, a 15-foot Power Cord, and a Table-Top cover.

Expansion capability is included within the box for many additional options that can be included or added at a later date without requiring additional space. The basic machine occupies 9 OMNIBUS slots. Up to 11 additional modules can be added on the basic OMNIBUS. Use of the OMNIBUS Expander permits up to 18 additional modules to be added, providing a total of 38 slots. (Two slots are used to interconnect the OMNIBUS and 9 slots are used for the basic system leaving 29 slots available for options.) The system can be further expanded with the addition of the Expander Box, which provides slots for 18 or 36 additional modules.

The System Expander Box, type BA8-A, includes one KC8-EB Blank Front Panel, one Power Chassis Assembler, one OMNIBUS, and a BC08H Cable Set (cover is included). One additional OMNIBUS Expander can be added to increase Bus capability.

The rack-mounted PDP-8/E (see Figure 11-2) can be installed in a DEC cabinet or mounted in a customer cabinet. The Chassis contains an H919 Bus Assembly, an H724 Power Supply, a 15 ft. Power Cord, and Chassis Slides. The rack-mounted version is approximately 10-½ in. high, 19 in. wide, and 23-¼ in. deep, and weighs approximately 90 pounds.

Expansion capability of the rack-mounted PDP-8/E is the same as that of the table-top model. The rack-mounted System Expander Box, type BA8-B, includes a DC8-EB Blank Front Panel, a Power Supply, Chassis Assembly with one OMNIBUS, and a BC08H Cable Set (rack-mountable

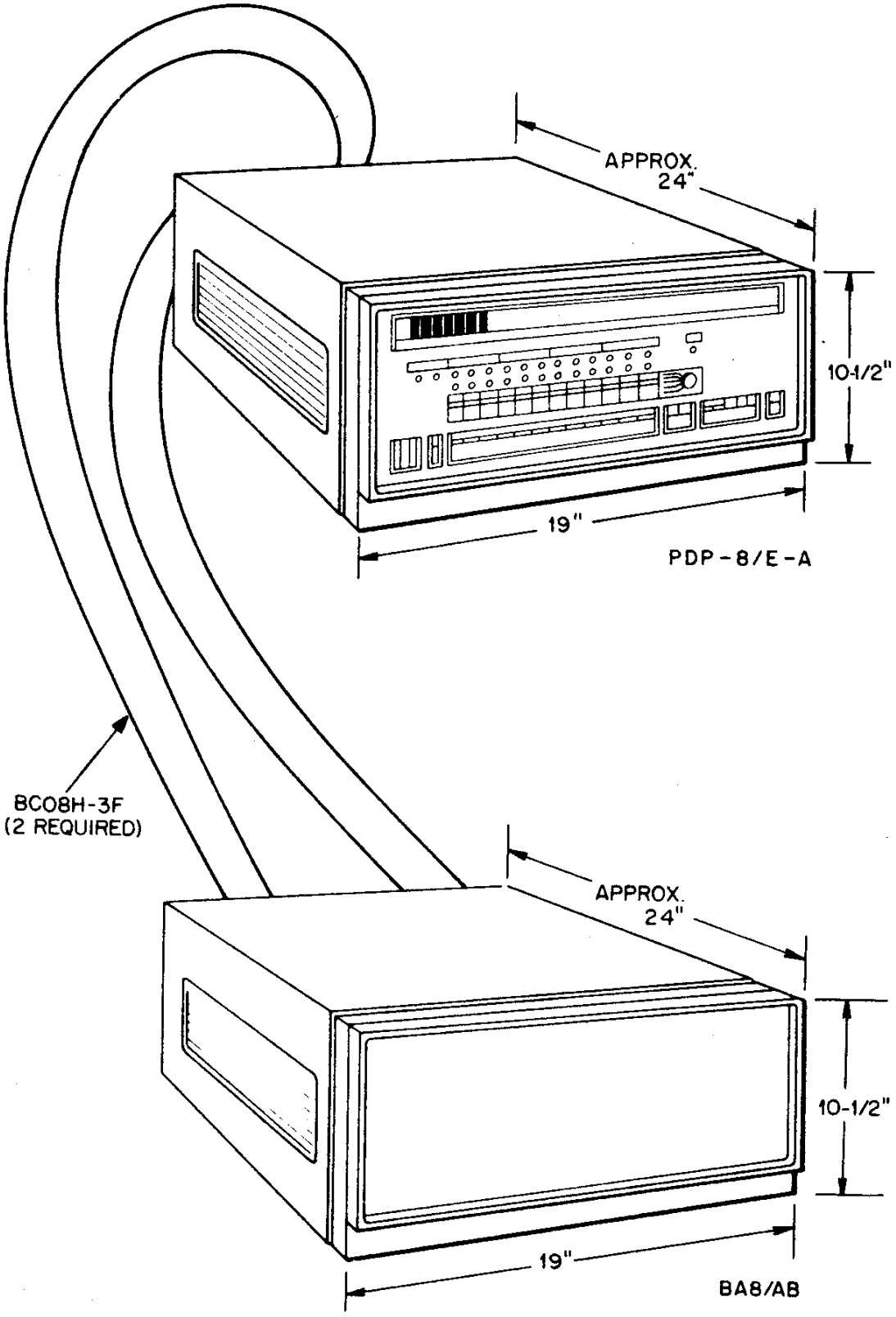


Figure 11-1 Table-Top PDP-8/E System and Expander Box

slides are included). An additional OMNIBUS Expander can be added to increase the Bus capabilities.

This cabinet arrangement allows the user to place the processor and an assortment of peripherals in one area. The basic cabinet (see Figures 11-3 and 11-4) is a Type H960B. Additional option cabinets (Type H961A) can be purchased and connected to the basic cabinet. Each cabinet is approximately 71-7/16 in. high, 21-11/16 in. wide, and 30 in. deep. The door swing is approximately 22 inches (both front and back doors). Thus, overall space requirement for each cabinet is approximately 22-in. by 74-in.

The standard Teletype Model ASR33 requires floor space approximately 22-1/4 in. wide by 18-1/2 in. deep. Standard signal cable length restricts the location of the Teletype to within eight feet of the computer equipment cabinet.

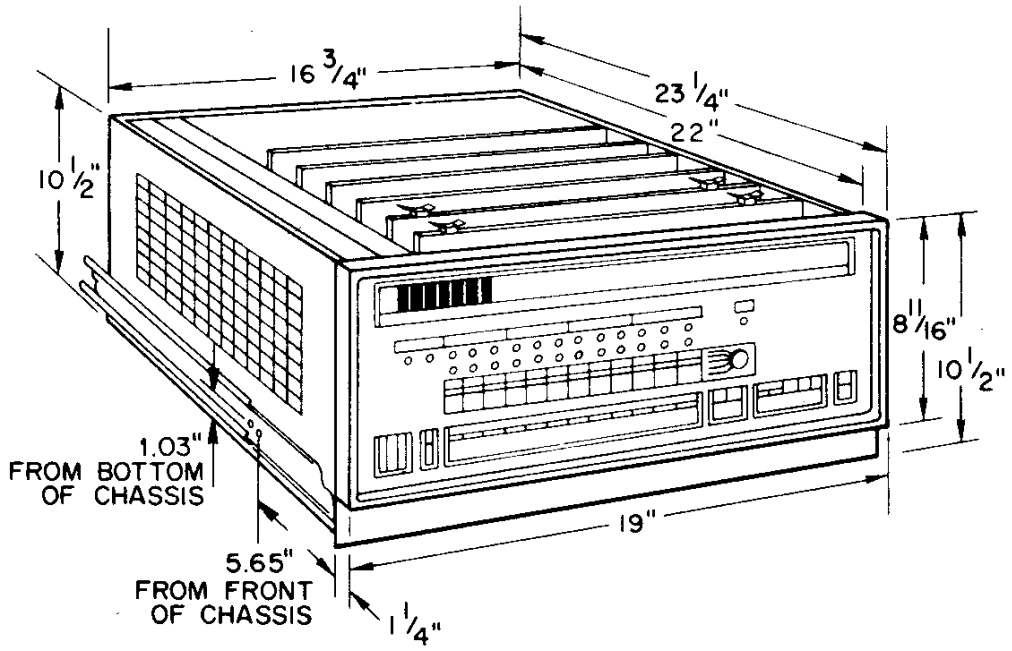


Figure 11-2 Rack-Mounted PDP-8/E Dimensions

Digital Equipment Corporation manufactures a standard 19-in. mounting frame assembly that offers the customer complete flexibility in selecting hardware to design the cabinet. It is a complete enclosure designed to house module racks, power supplies, computer systems, and peripherals.

The frame assembly, which includes a filter cover, is designed for sophisticated computer systems. It is constructed of rugged 12- and 13-gauge steel. The two pairs of frame uprights have 9/32 in. holes drilled at standard EIA spacings (5/8-5/8-1/2) the full length of the 63-in. mounting panel height.

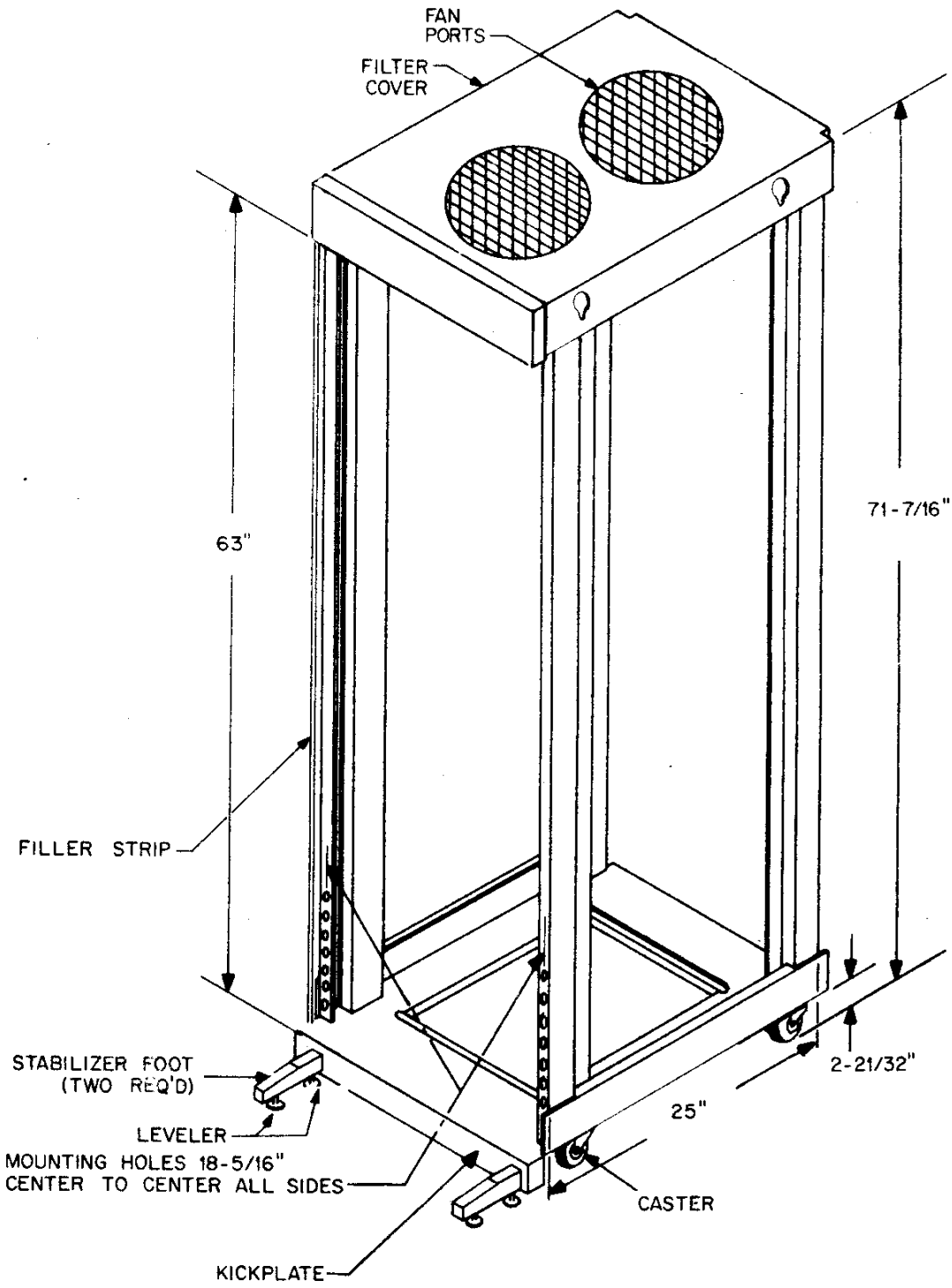


Figure 11-3 Typical H960-Series Cabinet Frame—Dimensions

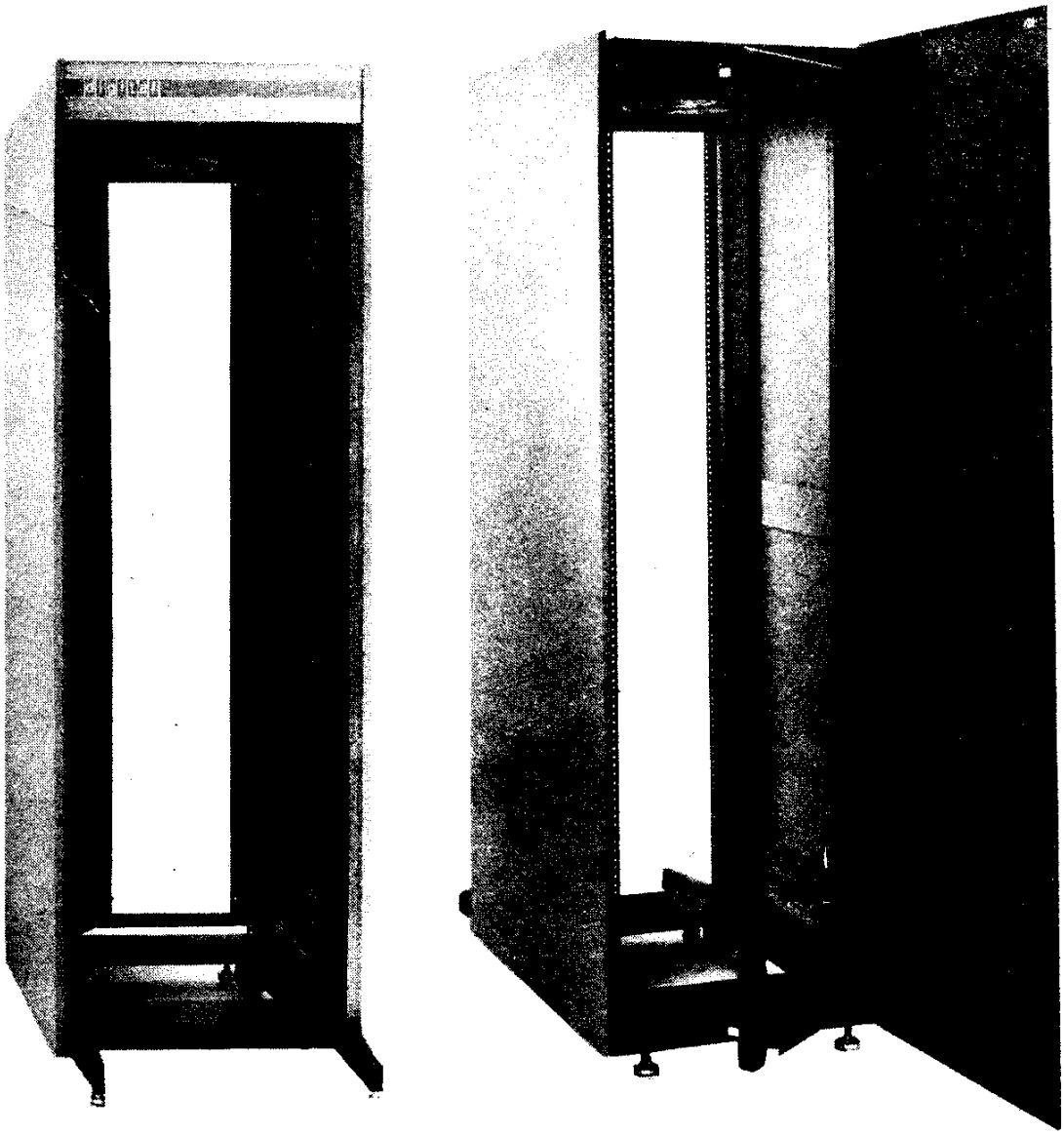


Figure 11-4 DEC H960B Cabinet

ENVIRONMENTAL REQUIREMENTS

Ambient temperature at the installation site can vary between 30 degrees F and 130 degrees F (between 0 degrees C and 55 degrees C) with no adverse effect on computer operation. To extend the life expectancy of the system, however, it is recommended that ambient temperature be maintained between 70 degrees F and 85 degrees F (between 21 degrees C and 30 degrees C). Humidity requirement is from 10 percent to 90 percent without condensation.

During shipping or storing of the system, the ambient temperature may vary between —4 degrees F and 150 degrees F (—20 degrees C and 65 degrees C). Although all exposed surfaces of all DEC cabinets and hardware are treated to prevent corrosion, prolonged exposure to extreme humidity should be avoided.

POWER REQUIREMENTS

PDP-8/E—	Power Dissipation	450 W
	Input Power	95V to 130V 47 to 63 Hz, 6A
	or	
		185V to 250V 47 to 63 Hz, 3A
Teletype—	Input Voltage	104V to 126V, 60Hz + or — 0.45 Hz 207V to 253V, 50Hz + or — 0.50 Hz
	Line Current Drain	2.0A
	Power Dissipation	150 W

For machines to be installed in the U.S.A. - This equipment requires a 120 volt, 60 cycle, single phase, two wire plus ground electrical supply. The computer is equipped with a Hubbell No. 3331 male plug which mates with a No. 3330 receptacle; a 30 amp. circuit is recommended.

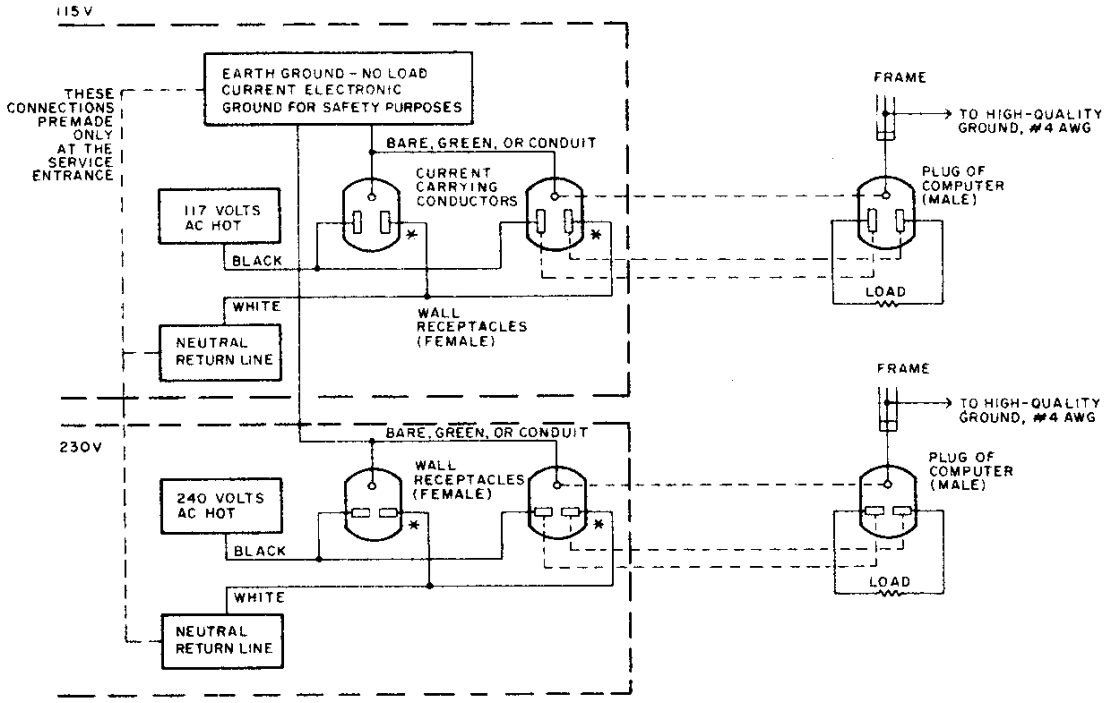
Cable Requirements—The PDP-8/E External I/O cable is a combination shield and ribbon cable. (The maximum length of the Data Break Cable, that can be used is 25 feet. The maximum length which can be used on the Positive I/O Bus Interface cables is 45 feet.) The cable between the PDP-8/E and the Expander Box is 3-½ feet long, and cannot be lengthened.

INSTALLATION PROCEDURE

Power Installation—The ac power requirements of the PDP-8/E computer are consistent with good electrical practice. The importance of correct electrical connections cannot be overstressed. Voltage readings must be made at the receptacle before the computer is plugged in, and careful checks must also be made after it is plugged in. It is an extremely wise precaution to take a voltage reading from the frame of the computer to the nearest grounded metal object before touching the computer. Figure 11-5 shows the recommended wall receptacle wiring.

It is generally advisable to provide a separate central load breaker panel for the computer system, with a breaker for the computer and for each peripheral receptacle. DEC recommends that the wiring include a run of No. 4 gauge wire from the computer frame to a substantial earth ground. A large water pipe or a steel building beam is adequate in most instances, although some systems may require a direct connection to a grounding stake or other high-quality earth ground. Significant operational difficulties are likely in the event of either a poor neutral or poor ground circuit.

Voltage readings should be made at each receptacle in the computer power system to ensure adherence to these power requirements. A checkout procedure for testing the electrical system is provided by DEC on request.



* A VOLTAGE BETWEEN WHITE WIRE AND FRAME INDICATES THAT THE RECEPTACLE IS ON A LINE WHICH IS A CONSIDERABLE DISTANCE FROM THE SERVICE ENTRANCE (THE WATT-HOUR METER AND FUSE BOX) AND IS CARRYING SIGNIFICANT CURRENT. THIS VOLTAGE SHOULD NOT EXCEED 5% OF THE NOMINAL LINE VOLTAGE.

Figure 11-5 Wall Receptacle Wiring

System Installation—The PDP-8/E is crated for shipment to the customer site to prevent damage. Installation is provided by DEC personnel at the customer site.

Computer customers may send personnel to instruction courses on computer operation, programming, and maintenance conducted regularly in Maynard, Massachusetts; Palo Alto, California; and Reading, England.

Table 11-1 provides installation data and space requirements which should be considered when installing a PDP-8/E system and related equipment. Figure 11-6 provides the necessary cabinet layout dimensions for those who are installing DEC Cabinet (H960B or H961B).

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