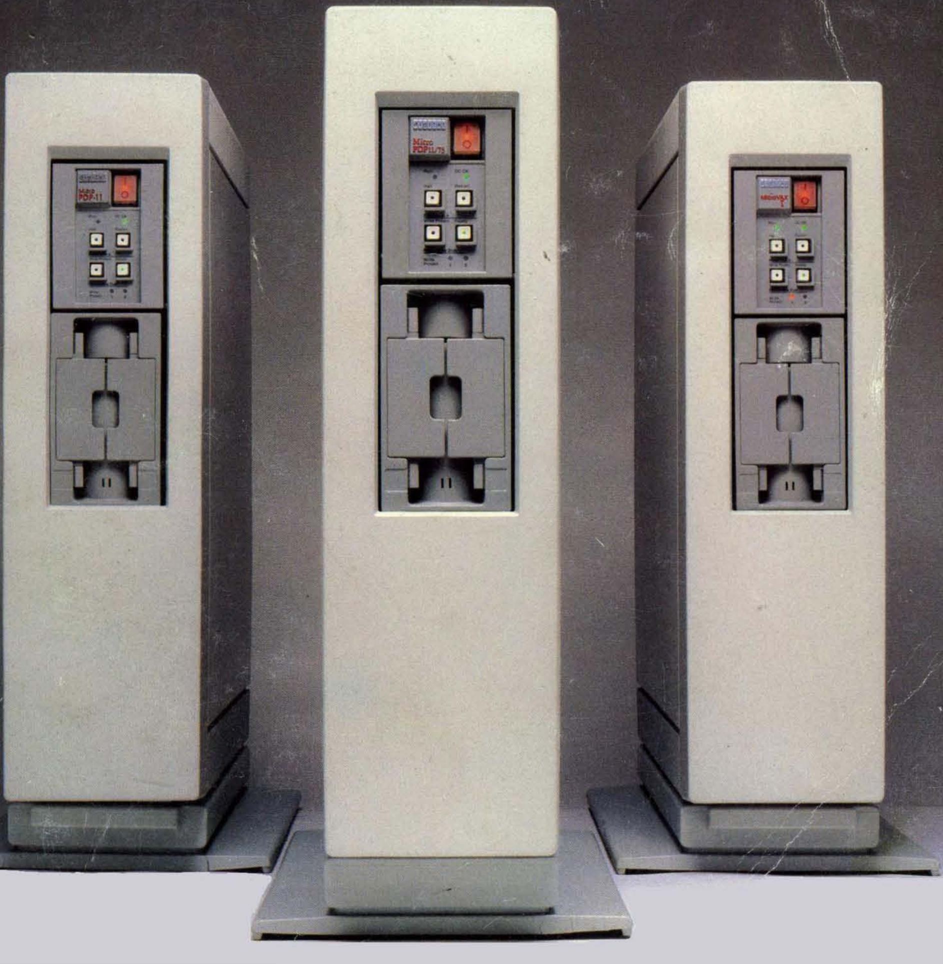
digital

Microsystems Handbook



Microsystems Handbook

digital

This handbook was produced by Digital Equipment Corporation, Corporate Communications Group, 200 Baker Avenue, West Concord, Massachusetts 01742.

Contents

Chapter 3 ▪ System Options

Preface

This *Microsystems Handbook* is a concise reference for Digital's 16-bit and 32-bit supermicrocomputers. It describes the functions of each member of the microsystems family—the MicroVAX I, MicroPDP-11/73, and MicroPDP-11/23—and their related options and software.

Chapter 1 provides a *family overview* with sections describing the similarities and differences among the various microsystems.

Chapter 2 outlines the *basic hardware components* of the family along with the types of configurations that are available.

Chapter 3 explains each of the *system options* that are available. These include memory, performance options, storage devices, communications devices, terminals, and printers.

Chapter 4 outlines *basic system software products* for the microsystems, including operating systems, high-level languages, information management products, productivity tools, and communications software.

Chapter 5 describes the *Digital Network Architecture* and how the microsystems implement the architecture with such products as DECnet, Ethernet, Internet, and Packetnet.

Chapter 6 summarizes the *VAX and PDP-11 architectures* and how they are implemented in the MicroVAX I and the MicroPDP-11 systems.

Appendix A gives a *technical description of the 22-bit Q-bus* that is the backbone of the microsystems family.

Appendix B gives *general information on the PDP-11/23-PLUS*, the original Q-bus PDP-11 system.

Appendix C provides help with the *preparation for the microsystem installation* and gives some *configuring guidelines.*

Appendix D covers Digital's extensive *Customer Services programs.*

Appendix E lists additional *reference documentation* for MicroVAX I and MicroPDP-11 hardware and software.

Your comments and suggestions will help us in our continuous effort to improve the quality and usefulness of this handbook. Please complete and return the Reader Comment Sheet that can be found at the back of this book.

Note

At the time this handbook went to print, Digital Equipment Corporation announced its newest microsystem, the MicroPDP-11/SV. As a result, it is not specifically referred to in the body of the handbook. However, a brief overview of the product is presented in Appendix F.

Chapter 1 • Microsystems Family Overview

• Definition of Microsystems

Digital's microsystems are a family of low-cost, medium- to high-performance, 16- and 32-bit supermicrocomputers that are designed for realtime and timesharing applications. They each offer a choice of operating systems, high-level languages, and information management tools. The three microsystems described in this handbook are the 32-bit MicroVAX I, the 16-bit MicroPDP-11/73, and the 16-bit MicroPDP11/23.

The MicroVAX I successfully implements a subset of the 32-bit VAX architecture and a modular version of the VAX/VMS operating system. This enables the MicroVAX I to be compatible with larger VAX systems and to run programs that have been developed on these systems.

The MicroPDP-11/73 and MicroPDP-11/23 successfully implement a subset of the 16-bit PDP-11 architecture and all of the many PDP-11 operating systems. This enables the MicroPDP-11 systems to be compatible with larger Q-bus and UNIBUS PDP-11 systems and to run programs that have been developed on these systems.

Markets/Applications

The microsystems serve the technical customer who requires data processing for a variety of applications in the laboratory, factory, medical, educational, and engineering markets. They also serve the commercial customer who requires data processing for such applications as business, office, banking, insurance, and administration. And because they are fully compatible with software developed for either the VAX or PDP-11, the microsystems offer Digital's existing customers easy growth and expansion capabilities.

The microsystems family allows users to build distributed processing networks entirely from Digital's systems and components. Each microsystem can be easily integrated into distributed processing environments with other Digital system products via DECnet or into local area networks with Ethernet.

Number of Users

A user is the person who is directly using the computer, either by terminal or batch. The number of users that can concurrently share each microsystem is based upon the type of application that each is running. The MicroVAX I can accommodate up to four concurrent, active users depending on the application. The MicroPDP-11/73 can accommodate up to 12 concurrent, active users, and the MicroPDP-11/23 can accommodate up to four concurrent, active users.

▪ Commonality Among the Microsystems

All of the microsystems use many of the same components. These components are the chassis, enclosures, memory options, storage devices, communications devices, video and printing terminals, and modems. Because they are common components, they allow for easy, cost-effective migration from one microsystem to another as the customer's application requires. There is no need for reinvestment in new packaging or options when the processing requirements grow.

Q-bus

The 22-bit Q-bus (or the Q22 bus) is implemented in the common, eight-slot backplane assembly of the microsystems. The Q22 bus enables a common set of low-cost interface options to be installed in the systems. The Q22 bus implements block-mode, direct-memory access addressing for up to 4 Mbytes of physical memory.

System Hardware

The microsystems are available as complete systems in various types of configurations. The microsystem chassis provides all of the internal assemblies necessary to configure a system. Included with each chassis are an eight-slot, Q22-bus backplane, dc power supply, front control panel, I/O distribution panel, and cooling fans.

There are three types of common system packaging methods—floorstand, tabletop, and rackmount. All three can accommodate a microsystem chassis along with a dual-diskette drive and/or a Winchester fixed-disk drive.

System Options

Besides the basic system hardware, there is a large assortment of system options that are available for the microsystems. These include memory and performance options, storage devices, communications devices, video and printing terminals, and modems. All are Q22-bus-compatible.

Because the microsystems share the same bus, chassis, and backplane, most of the options that are available for one microsystem are also compatible with the other microsystems. This makes system migration and software interchange easier and very economical.

Networks

Digital's Network Architecture (DNA) defines an overall approach to networking and includes a set of software and hardware protocols to support a range of requirements.

The microsystems and other Digital computers can use DECnet, one of the DNA protocols, to share files, programs, and resources. Systems can communicate over traditional interconnects and over Ethernet for high-speed, local communication. Digital's Ethernet program provides high-level software and advanced hardware to enable high-speed communications between computer systems and servers in Local Area Networks (LANs).

By emulating the protocol of other manufacturers' devices, the microsystems with Internet software can communicate with other vendors' equipment. IBM batch (2780/3780), interactive (3271), and SNA protocols are supported, as are those from CDC and UNIVAC.

The microsystems can interface with Digital's Packetnet system for communications through a public packet-switched network (X.25) with other systems, regardless of the manufacturer.

The Professional 300 series, DECmate II and III, and the Rainbow 100 series of single-user systems can also be connected to the microsystems for file transfer and terminal emulation.

▪ Differences Among the Microsystems

The MicroVAX I and the MicroPDP-11 systems are based on different, but related, family architectures. Although there are many architectural similarities between the VAX and PDP-11 system designs, there are some differences as well. Both families share such characteristics as physical-address space, virtual-address space, memory management, general registers, addressing modes, interrupts, and use of instruction sets. The families differ in the size and type of these characteristics.

For example, the MicroVAX I is based on 32-bit operand size, and the MicroPDP-11s are based on 16-bit operand size. They each use virtual-address space, but each can address varied sizes of this space. For a more detailed description of the two family architectures, refer to Chapter 6—Architecture Summary.

Central Processors

In order to implement each individual family's architecture, the MicroVAX I, MicroPDP-11/73, and MicroPDP-11/23 each have a unique central processing unit (CPU). All of these CPU models utilize the Q22 bus, which provides compatibility for all of the microsystems' options. These CPU models represent a wide range of performance and capabilities. When a microsystem is being selected, the central processor should be one basis for the customer's decision.

System Software
The MicroVAX I and the MicroPDP-11s each have system software and layered products designed specifically for their architectures. Each of these systems offers a wide range of proven operating systems, high-level languages, information management products, productivity tools, and networking products for supporting the customer's application. Many of the same types of system software have been developed for each of the microsystems. Refer to Chapter 4—System Software and Layered Products for descriptions of each of these software products.

Chapter 2 • System Hardware

Introduction

The basic system hardware that each microsystem requires is a CPU, chassis, enclosure, memory, and mass storage. Digital has developed a common chassis for all of the microsystems so that the user has to choose only the CPU, enclosure, memory, and storage that best fits the needs of the application.

The microsystem chassis, along with a variety of enclosures, is very compact. This gives the microsystems a trim, attractive appearance while also giving them more flexibility in location and in mounting. A complete microsystem can sit on a tabletop, stand on the floor under a desk or table, or be mounted into a standard 19-inch-wide rack.

Configuring methods will be discussed so that the accompanying system options can be planned and ordered in the easiest and most convenient manner. For detailed configuring information for all the microsystems, refer to the *PDP-11 Systems and Options Catalog.* Ordering information for this publication is provided in Appendix E—Documentation.

CPU Modules

Each microsystem has its own unique central processing unit.

- The KD32-A module set is the 32-bit MicroVAX I central processing unit.
- The KDJ11-B module is the 16-bit MicroPDP-11/73 central processing unit.
- The KDF11-B module is the 16-bit MicroPDP-11/23 central processing unit.

These CPU modules all utilize the Q22 bus that helps to protect previous hardware and software investments. Coupled with the VAX or PDP-11 architectures, these CPU modules represent a wide range of performance and capabilities within the microsystem family.

KD32-A CPU Module Set

The MicroVAX I processor logic is contained on two quad-height modules—the data path module (DAP) and memory controller module (MCT). The data path module and the memory controller module are electrically connected by a flat cable. These two modules implement the MicroVAX I architecture and are designated as the KD32-A MicroVAX I CPU.

The KD32-A CPU module set provides:

- MicroVAX I CPU functions
- Q22-bus interface that supports block-mode direct-memory access (DMA) transfers and up to 4 Mbytes of physical memory
- 8-Kbyte, direct-mapped cache memory 512-entry longword-translation buffer
- 10-millisecond interval timer
- One console terminal serial-line unit
- 8-Kbyte bootstrap programmable read-only memory (PROM)

The data path module contains the data path, instruction decoder, microsequencer logic, and the miscellaneous logic necessary to implement the MicroVAX I instruction set. It decodes and executes macroinstructions and controls the microinstruction flow. It controls the transfer of data within the module and processes program interrupts. Communications between the console terminal and the system are also controlled by the module logic.

The data path module is a quad-height module and occupies slot two of the Q22-bus backplane. The module is connected by flat cables to the memory controller module and to the panel insert on the I/O distribution panel. Three light-emitting diode (LED) indicators, mounted on the edge of the module, display an octal code during self-test operations.

The main functions of the data path module are:

- Decoding and executing macroinstructions
- Controlling microinstruction flow
- Processing program interrupts
- Communicating with the console terminal
- Communicating with the memory controller module

The memory controller module accepts memory request commands from the data path module and sequences the memory controller logic to perform the specified commands. It translates virtual addresses into physical addresses and implements the Q22-bus interface. The logic is contained on a quad-height module that occupies slot one of the Q22-bus backplane. The memory controller module is connected by a flat cable to the data path module and performs the following functions:

- Generates clock signals
- Controls the memory controller microinstruction flow
- Translates virtual addresses
- Accesses the data cache
- Communicates with the Q22 bus

KDJ11-B CPU Module

The KDJ11-B is a quad-height processor module that is implemented in the MicroPDP-11/73. It is designed for use in high-speed, realtime applications and for multiuser, multitasking environments. It has three times the relative performance of the KDF11-B module.

The KDJ11-B provides:

- Complete PDP-11 instruction set including hardware multiply/divide (Extended Instruction Set)
- Floating-point instruction set standard in microcode
- 8-Kbyte, direct-mapped cache memory
- Q22-bus interface that supports block-mode DMA transfers and up to 4 Mbytes of physical memory
- Line frequency clock
- Four levels of interrupts
- Powerfail/autorestart
- Console emulator in microcode
- One console terminal serial-line unit
- 2-Kbyte erasable read-only memory (ROM) for custom bootstraps

The KDJ11-B executes the complete PDP-11 integer and floating-point instruction sets. Full 22-bit memory management is provided for both instruction and data references (I&D space) in three protection modes—kernel, supervisor, and user. The KDJ11-B is fully downward compatible with older PDP-11s that have 18-bit memory management.

The three protection modes provide the ability to implement layered software protection. Memory management separately manages each mode allowing each to access different sections of main memory. Furthermore, each section can have different access protection rights. Each mode also uses a separate system stack pointer that offers an additional degree of isolation. The protection modes are organized so that a higher protection mode can always enter a lower protection mode, but a lower protection mode can never enter a higher protection mode. Kernel mode has full privileges and can execute all instructions. Supervisor mode and user mode, the two lower privileged modes, cannot execute certain instructions.

The module interfaces to the Q22 bus and can address up to 4 Mbytes of main memory. Block-mode DMA transfers are included, which are standard on the Q22 bus. The Q22 bus is fully downward compatible with the standard 18-bit Q-bus.

The KDJ11-B supports console emulation (micro octal debugging tool or ODT). This allows users to interrogate and write to main memory, CPU registers, and I/O devices.

The module contains an 8-Kbyte, write-through, direct-mapped cache (set size one, block size one). The cache is transparent to all programs and acts as a high-speed buffer between the processor and main memory. The data stored in the cache represents the most active portion of main memory being used. The processor accesses main memory only when data is not available in the cache.

Self-diagnostic LEDs are provided on the KDJ11-B and indicate the status of the module and system when the module is powered-up. The LEDs aid in troubleshooting module failures. The LEDs also appear on the I/O distribution panel.

The KDJ11-B is compatible with all of the MicroPDP-11 family operating systems.

KDF11-B CPU Module

The KDF11-B is a quad-height processor module that is implemented in the MicroPDP-11/23. It is designed for use in moderate-speed, realtime applications and for multiuser, multitasking environments.

The KDF11-B provides:

- Complete PDP-11 instruction set including hardware multiply/divide (EIS)
- Floating-point and commercial instruction sets (optional)
- Q22-bus interface that supports block-mode DMA transfers and up to 4 Mbytes of physical memory
- Line frequency clock
- Four levels of interrupts
- Powerfail/autorestart
- Console emulator in microcode
- Two console terminal serial-line units
- 8-Kbyte diagnostic/bootstrap ROM

The KDF11-B module supports up to 256 Kbytes of memory on an 18-bit Q-bus backplane or up to four Mbytes of memory on a Q22-bus backplane. When used with the Q22 bus, the KDF11-B can utilize four-level interrupt protocol. The 22-bit memory management is provided for both instruction and data references (I&D space) in two protection areas—kernel and user. The KDF11-B is fully downward compatible with older PDP-11s that have 18-bit memory management.

The KDF11-B supports console emulation (micro octal debugging tool or ODT). This allows users to interrogate and write to main memory, CPU registers, and I/O devices.

Self-diagnostic LEDs are provided and indicate the status of the module and system when the module is powered-up. The LEDs aid in troubleshooting module failures. The LEDs also appear on the I/O distribution panel.

The KDF11-B is compatible with all of the MicroPDP-11 family operating systems.

▪ System Chassis and Enclosures

Each microsystem chassis contains the components that are shown in Figure 2-1. These components include the following:

- 8-slot Q22-bus backplane
- dc power supply
- Front control panel
- I/O distribution panel (not shown)
- Two cooling fans
- CPU module(s)
- System modules (optional)
- 5.25-inch fixed-disk drive (optional)
- 5.25-inch dual-diskette drive (optional)

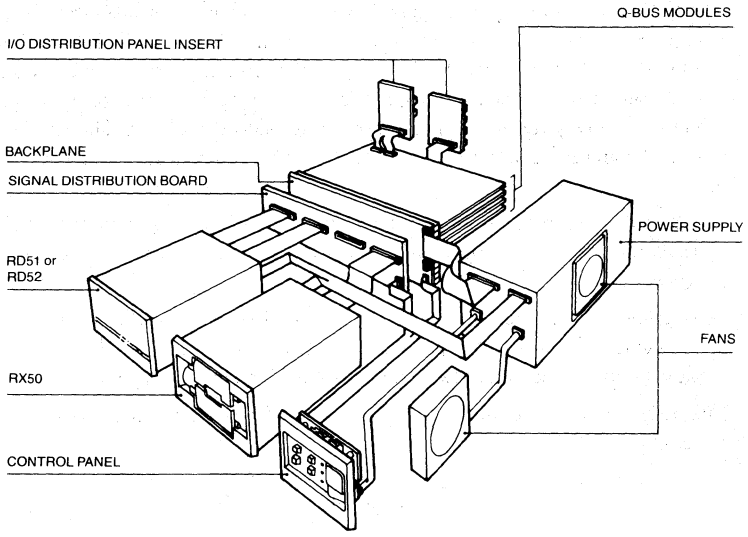


Figure 2-1 ▪ Microsystem Chassis

H9278-A Backplane

The logic modules for the MicroVAX I, MicroPDP-11/73, and MicroPDP-11/23 are installed into the H9278-A backplane assembly. The backplane is included with the system chassis. The assembly consists of four rows by eight slots of prewired connectors and a mounting frame that allows quad- or dual-height logic modules to be easily inserted and removed. A card guide also permits the latches on the quad-height modules to hold securely onto the backplane.

The backplane incorporates the Q22-bus wiring in rows A and B of connector slots one through eight and in rows C and D of connector slots four through eight. The Q22 bus supports an interrupt and DMA grant-continuity scheme for the logic modules installed in the backplane. Table 2-1 shows each microsystem's module placement assignments, and Figure 2-2 shows the backplane slot locations listed in Table 2-1.

The backplane provides the 220-ohm far-end termination for the MicroVAX I and the 120-ohm far-end termination for the MicroPDP-11 systems. Refer to Appendix A—Q-bus for detailed information on 120-ohm and 220-ohm bus termination on the backplane.

Four connectors on the backplane, J1 through J4, receive voltages and signals from the H7864 power supply and provide signals and voltages to the front control panel of the microsystem unit. Maximum ratings are 36 A at +5 V and 6 A at +12 V.

Table 2-1 ▪ Logic-module Assignments

Slot	Row	MicroVAX I	MicroPDP-11/73	MicroPDP-11/23
1	ABCD	CPU	CPU	CPU
2	ABCD	CPU	Memory	Memory
3	ABCD	Memory	Additional memory or communications option	Additional memory or communications option
4	ABCD	Additional memory or communications option	Two dual-height options or one quad-height option. RQDX1 if it is the last-used slot.	Two dual-height options or one quad-height option. RQDX1 if it is the last-used slot.
5	ABCD	Two dual-height options or one quad-height option. RQDX1 if it is the last-used slot.	See Slot 4	See Slot 4
6	ABCD	See Slot 5	See Slot 4	See Slot 4
7	ABCD	See Slot 5	See Slot 4	See Slot 4
8	ABCD	See Slot 5	See Slot 4	See Slot 4

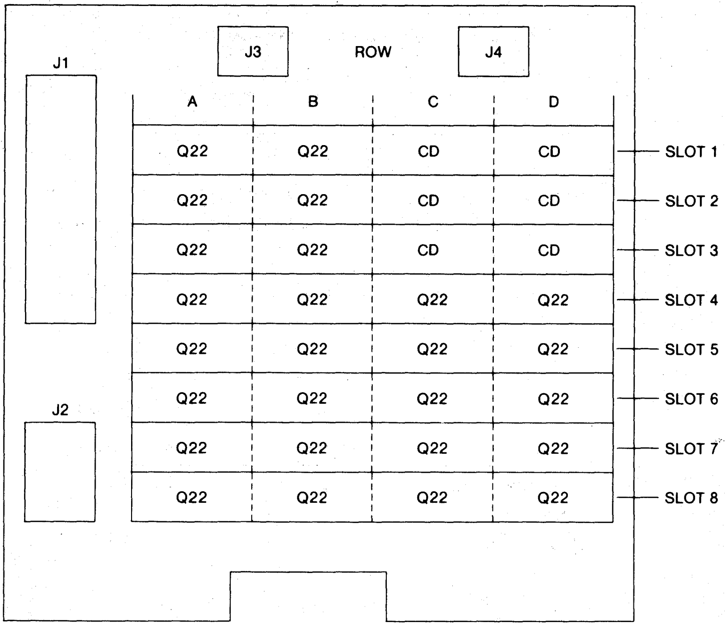


Figure 2-2 ▪ Q22-bus Backplane Assignments

H7864 Power Supply

The H7864 modular power supply, shown in Figure 2-1, is common to the basic microsystems chassis. It drives 230 watts of power to the logic modules mounted in the system backplane, to the disk-drive units, to the control-panel switches and indicators, and to the two dc fans.

Features of the H7864 power supply include:

- Universal supply with switchable inputs for 88-128 V RMS at 120 V/60 Hz and 176 - 256 V RMS at 240 V/50 Hz
- Separate output circuit for the two dc fans that provides 0.45 A at +12 V and +9 V
- Line voltage conditioning
- Q22-bus compatible power sequencing signals
- +5 V dc power rating of 4.5 A minimum to 36 A maximum
- +12 V dc power rating of 7 A maximum

Additional H7864 features include thermal shutdown, overvoltage and overcurrent protection, ac input transient suppression, and three signals for Q22-bus operation.

Control Panel

The microsystem control switches and indicator lights are located on the control panel at the front of the microsystem chassis. This control panel is common to each microsystem chassis. Figure 2-3 shows the pushbuttons and indicators for the floorstand unit. These controls allow you to apply and remove the ac power, to stop and start the current program operation, and to protect the data stored on the RD51 or RD52 Winchester disk drive.

- ac Power—is controlled by the ac power rocker switch marked 0 and 1. When power is applied, the pushbutton lights orange.
- Halt Pushbutton—halts the operation of the current program. When the switch is activated, the indicator on the pushbutton lights orange. Restart/bootstrap operations are inhibited when the Halt pushbutton is depressed.
- Restart Pushbutton—used to initiate a processor restart when the Halt pushbutton is not activated.
- Fixed Disk 0 Write Protect Pushbutton—used to prevent data from being written to the fixed disks. When the switch is activated, the orange indicator in the pushbutton lights.
- Fixed Disk 0 Ready Pushbutton—disables the fixed-disk drives to prevent data from being read from or written to the disk. When activated, the green indicator in the pushbutton lights. The indicator lights intermittently when the disk is reading or writing data during normal operation.
- Run Indicator—when the processor is operating, the run indicator lights green.
- DC OK Indicator—when the dc voltages in the system unit are correct, the DC OK indicator lights green.
- Removable Disk Write Protect 1—when data cannot be written to removable disk 1 in the RX50, the indicator lights orange.
- Removable Disk Write Protect 2—when data cannot be written to removable disk 2 in the RX50, the indicator lights orange.

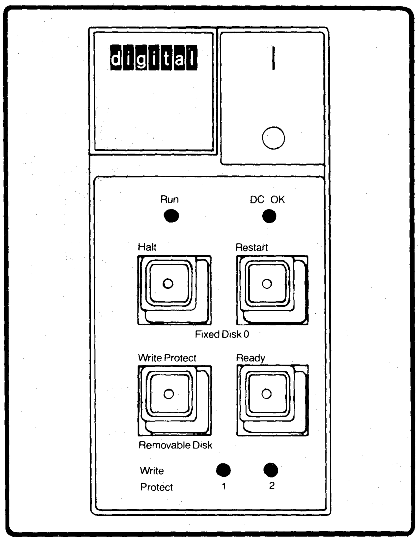


Figure 2-3 ▪ Control Panel

I/O Distribution Panel

The I/O distribution panel, located at the rear of the microsystem chassis, is used to connect the cables from the console terminal, printing terminal, and other external devices that operate with the system. The I/O distribution panel, shown in Figure 2-4, contains six areas to mount connectors for these devices, and also provides signal filtering and shielding against electromagnetic and radio frequency interference (EMI/RFI). Each module that sends a cable outside the chassis has a panel insert that mounts in the I/O distribution panel. The external cable attaches to a connector on this panel. Insert panels come in two standard sizes: small (1 × 4 inches) and large (2 × 3 inches). The I/O distribution panel has apertures for four large inserts and two small inserts, or two large inserts and five small inserts (two of the large apertures can be converted to three small apertures).

With each microsystem, one of the large panel inserts (the leftmost) is predefined. On the MicroVAX I and MicroPDP-11/73, this panel insert has one 25-pin EIA connector for the console terminal. It also includes a rotary switch for selection of the baud rate of the console device. Two seven-segment LED indicators are used to display an error code and its location during the self-test diagnostic program and bootstrap routine. The MicroPDP-11/23 has two 25-pin EIA connectors for the console terminal and video or printing terminal.

The remaining connector panels on all of the microsystems provide 25-pin EIA connectors to be used when additional device interfaces have been installed in the units.

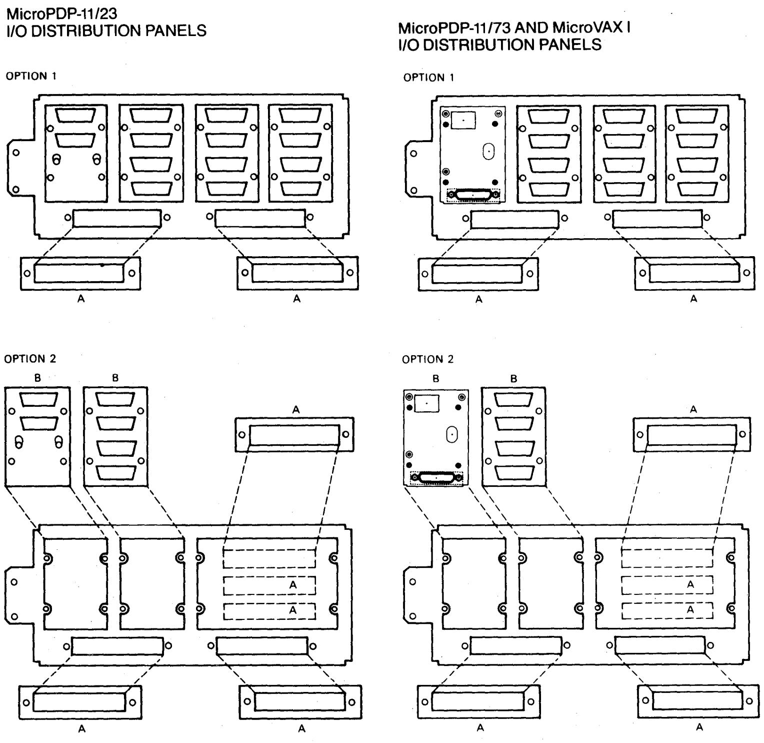


Figure 2-4 ▪ I/O Distribution Panel

Fans

Two brushless fans within the microsystem unit provide a flow of air from left to right (side-to-side) to cool the internal assemblies and modules as shown in Figure 2-1. Power for the fans is provided by the H7864 dc power supply.

Enclosures

The microsystems are available in three different system packages as shown in Figure 2-5. All three packages incorporate the same components that are shown in Figure 2-1. Each microsystem has a chassis, CPU, a choice of memory, a choice of integrated fixed-disk and/or dual-diskette drives, one or two serial-line units, and a Q22-bus backplane for additional interfaces and expansion.

The floorstand and tabletop units are enclosed in an outer shell and include a front cover and rear I/O distribution panel cover. The floorstand unit includes a base which is attached to the bottom of the microsystem unit.

The rackmount unit can be installed into a 19-inch-wide (48.26 centimeters) rack or cabinet and contains a front cover and chassis cover. The mounting hardware for installation into the rack is also included.

The MicroPDP-11/23 and MicroPDP-11/73 can also be mounted into a 42-inch-high (106 centimeters) H9642-series cabinet. These systems include all of the same components as the other enclosures except for the 5.25-inch dual-diskette drive and fixed-disk drives. In their place, there is space in the cabinet to add an RQC25 fixed/removable disk and an additional RC25 fixed/removable disk. The MicroPDP-11/23 system cabinet can also accommodate an RL02 removable-cartridge disk or a TSV05 streaming-tape drive. Refer to the *PDP-11 Systems and Options Catalog* for complete configuring information.

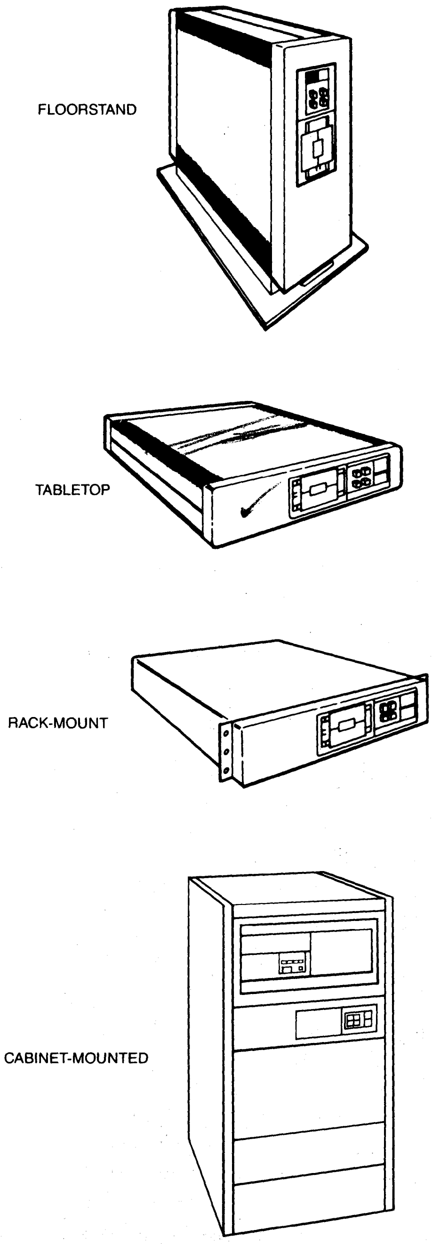


Figure 2-5 ▪ Microsystem Enclosures

▪ Configurations

Each of the microsystems is available in a variety of configurations. The MicroVAX I is available in System Building Block configurations. The MicroPDP-11/73 is available in Base System, Packaged System, and System Building Block methods. The MicroPDP-11/23 is available in Base and Packaged Systems only. The following section briefly describes each of these configuring methods. Detailed configurating information can be found in the *PDP-11 Systems and Options Catalog.* Copies of this publication can be obtained from your local Digital sales representative or by ordering them direct from Digital. Refer to Appendix E for information on how to order documentation.

Base Systems

The base system is a method of selecting a MicroPDP-11/73 and MicroPDP-11/23 system with a preconfigured CPU, microsystem chassis, enclosure, load device, main-storage device, backup-storage device, and memory.

Packaged Systems

The packaged system allows MicroPDP-11/73 and MicroPDP-11/23 customers to choose a pre-configured hardware system and software license. Each packaged system includes a CPU, microsystem chassis, enclosure, communications device, load and backup-storage device, main-storage device, memory, and a PDP-11 operating-system general license. In certain configurations, a U.S.A. country kit may also be included.

System Building Blocks

The system building block is the most popular way to select a hardware and software microsystem configuration. System building blocks are available for the MicroVAX I and the MicroPDP-11/73. Every microsystem component is offered as part of a menu. First, a CPU must be chosen with the appropriate amount of memory from the CPU menu, and then each of the remaining components must be chosen from their own specific menu. The menu categories are storage device, load device, enclosure, communications device, console terminal, and software license. System building blocks are flexible because they offer a wide range of components, are customer-tailorable, and also make configuring very easy.

Country Kits

For MicroPDP-11 systems, user documentation comes in a country kit which must be ordered with the system. Also included in the kit are customer-runnable diagnostics. User documentation in English is supplied with the MicroVAX I.

The appropriate country power cord is determined by the destination address of the system, and is automatically shipped with all of the microsystems.

▪ Additional Documentation

MicroVAX I CPU Technical Description	*EK-KD32A-TD*
KDJ11-A CPU Module User's Guide	*EK-KDJ1A-UG*
KDF11-B CPU Module User's Guide	*EK-KDFEB-UG*
PDP-11 Systems and Options Catalog	*ED-26051-41*

Chapter 3 • System Options

Introduction

Once a microsystem is selected, the hardware options that best suit the application must also be chosen. These options include memory, options which enhance the system's performance, storage devices, communications devices, terminals, printers, and modems. Most of these options are compatible with every member of the microsystems family. All of the options required to complete or expand a microsystem are described in this chapter.

Memory Options

Two types of memory options are available for the microsystems—the MSV11-QA and the MSV11-P series. These memory options can often improve system performance. Each provides the capability to perform direct-memory access. During direct-memory access operations, data is transferred without processor intervention using block-mode transfers. Up to 16 words or 36 bytes of information can be transferred by specifying a starting address.

MSV11-QA

The MSV11-QA memory module is a random-access memory (RAM) module that provides 1 Mbyte of main-memory storage. This memory is implemented on a single, quad-height module using 64-Kbit MOS RAM integrated circuits. MSV11-QA supports 22-bit addressing so that up to 4 Mbytes of physical memory can be configured.

The following are the main features of the MSV11-QA:

- Offers 1 Mbyte of MOS memory on a single, quad-height module
- Uses 64-Kbit MOS RAM chips
- Supports 22-bit addressing for up to 4 Mbytes of physical memory
- Supports block-mode DMA
- Has LEDs for parity-error checking

MSV11-P Series

The MSV11-P series comprises random-access memory (RAM) modules that provide 256 Kbytes (MSV11-PK) or 512 Kbytes (MSV11-PL) of main-memory storage. Each memory module is implemented on a single, quad-height module using 64-Kbit MOS RAM integrated circuits. The MSV11-P series supports 18-bit and 22-bit addressing so that up to 4 Mbytes of physical memory can be configured.

The following are the main features of the MSV11-P series:

- Offers 256 or 512 Kbytes of MOS memory on a single, quad-height module
- Uses 64-Kbit MOS RAM chips
- Supports 18-bit or 22-bit addressing for up to 4 Mbytes of physical memory
- Supports block-mode DMA transfers
- Has LEDs for parity-error checking

▪ Performance Options

Three options are available to enhance the performance of the MicroPDP-11/23. A floating-point processor and floating-point accelerator are available for applications that require a great deal of calculation. A character-string instruction set (Commercial Instruction Set) is also offered for business applications.

KEF11-AA Floating-point Processor

KEF11-AA is a single- and double-precision floating-point option. This option expands the capability of the MicroPDP-11/23 by adding the microcode to implement PDP-11 floating-point instructions. The microcode resides in two chips in one 40-pin package that mounts directly on the CPU module. It performs operations on 32-bit and 64-bit floating-point numbers and provides up to 17 digits of precision. KEF11-AA also provides integer to floating-point conversions.

FPF11 Floating-point Accelerator

FPF11 is a single-precision and double-precision fast floating-point hardware option that executes instructions approximately six times faster than the KEF11-AA. This option is a single, quad-height module for the MicroPDP-11/23 and mounts adjacent to the CPU. FPF11 performs hardware operations on 32-bit and 64-bit floating-point numbers and provides up to 17 digits of precision. Like the KEF11-AA, it provides integer to floating-point conversions.

KEF11-BB Character-string Instruction Set

The KEF11-BB implements a set of 27 commercial instructions on a variety of data types, including character-string, packed-decimal, and numeric formats. Because these data types closely resemble those used in COBOL, they are referred to as the Commercial Instruction Set (CIS). KEF11-BB mounts directly on the MicroPDP-11/23 CPU board.

▪ Storage Options

There are many storage options available for the microsystems. These options are offered in several different technologies so that the correct choice can be made for the storage application. Whether storage is being added for media backup, for loading software, for main storage, or for software interchange with another system, Digital has the appropriate storage device for the task.

RQDX1 Disk Controller

The RQDX1 is a single, quad-height, intelligent controller that provides data transfers between the Q22 bus and the RX50 dual-diskette drive, and one or two RD51 or RD52 fixed-disk drives. The RQDX1 logic provides the necessary data buffering and control to allow direct-memory access (DMA) transfers using the Mass Storage Control Protocol (MSCP).

A flat cable attaches to a 50-pin connector mounted at the edge of the module and to the signal distribution board located near the Q22-bus backplane. Signals and data are then transferred from the connectors on the distribution board to the disk-drive assemblies. Four LED indicators are also mounted near the edge of the module to display octal codes during the self-test program operation.

The following are the main features of the RQDX1:

- Controls up to four logical disk-drive units (two fixed-disk drives and one dual-diskette drive)
- Supports block-mode DMA data transfers
- Has maintenance self-test programs
- Uses LEDs for parity-error checking

RLV12 Disk Controller

The RLV12 controller is a single, quad-height module for the MicroPDP-11 family that interfaces the RL02 removable-disk drives to the Q22 bus. One RLV12 controls up to four RL02 disk drives. The RLV12 transfers data to and from the Q22 bus using direct-memory access transactions. It performs a cyclic redundancy check on data and headers. Memory is parity-checked, and the current command to the RLV12 is aborted when the error is detected.

The following are the main features of the RLV12:

- Controls up to four removable-disk drive units
- Supports burst-mode DMA data transfers
- Detects parity and flags error on current cycle
- Has maintenance self-test programs

RX50 Flexible Disk

Programs and data can be entered into and removed from the microsystems through the RX50 dual-diskette drives (shown in Figure 2-1 and in Figure 3-1). The RX50 is a single unit that contains two separate diskette drives. Each of the two drives in the RX50 operate with a 5.25-inch flexible diskette and provide a storage total on both diskettes of up to 819.2 Kbytes of formatted data. Access to the drive is through the two doors located at the front of the drive unit.

The RQDX1 controller module, located in the Q22-bus backplane, provides the interface between the Q22 bus and the RX50 dual-diskette drive. The controller implements the required MSCP protocol and controls the DMA data transfers. The RX50 operates with dc power supplied from the H7864 power supply.

The following are the main features of the RX50:

- 819.2 Kbytes total formatted capacity (409.6 Kbytes formatted capacity per diskette)
- 300 rotations/minute disk rotation
- Total of two surfaces on a single spindle
- 250 Kbits/second (31.25 Kbytes/second) average transfer rate
- 164 milliseconds average seek time (6 milliseconds track-track)
- 100 milliseconds average rotational latency
- Single-sided, 80 tracks/inch, preformatted diskettes

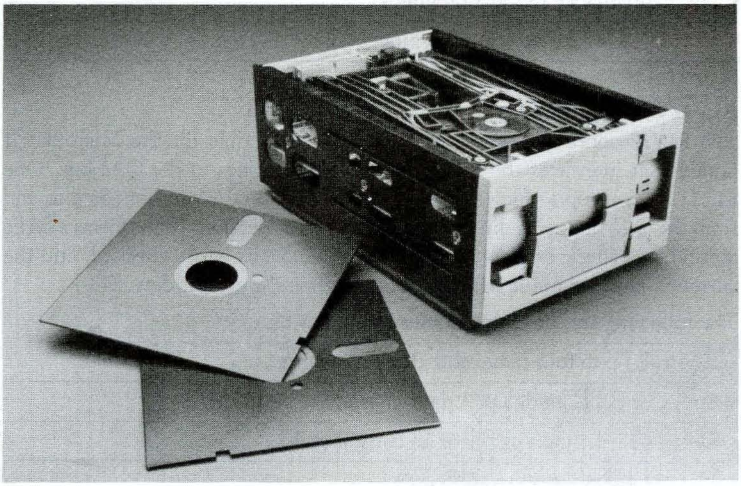


Figure 3-1 ▪ RX50 Dual-diskette Drive

RX02 Flexible Disk

The RX02, shown in Figure 3-2, is a double-density, flexible-diskette drive that operates with 8-inch diskettes. Direct-memory access is used to provide rapid data transfer and efficient utilization of the host processor. The RX02 includes two 0.5-Mbyte drives, for a total of 1 Mbyte, and a controller with interconnecting cables. The dual drives are packaged as a rackmount or tabletop unit. The RX02 also operates with its own power supply.

The following are the main features of the RX02:

- Two 0.5-Mbyte drives for a total formatted capacity of 1 Mbyte
- Supports DMA data transfers
- Two surfaces, two spindles
- 61 Kbytes/second peak transfer rate
- 154 milliseconds average seek time
- 83 milliseconds average rotational latency
- Single-sided, 77 tracks/inch, preformatted diskettes
- Will read RX01 formatted diskettes

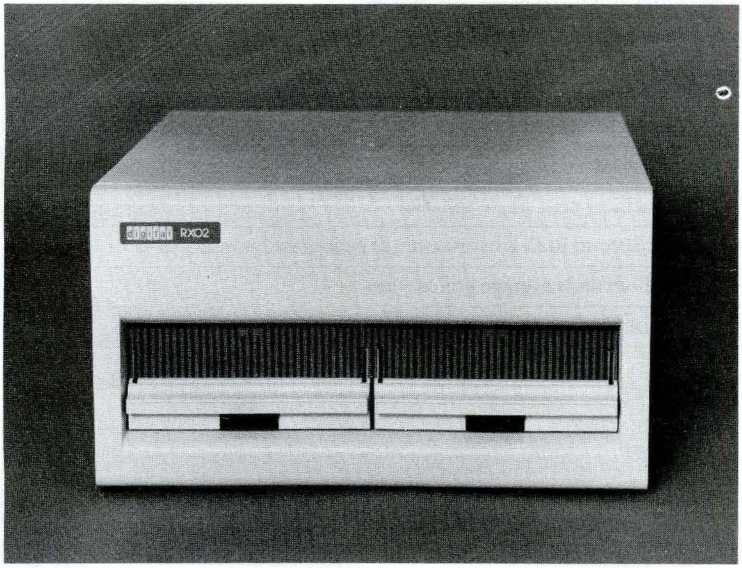


Figure 3-2 ▪ RX02 Diskette-Drive Subsystem

RD51 and RD52 Fixed Disks

The RD51 and RD52 fixed-disk drives are compact, Winchester disk drives (shown in Figure 2-1 and in Figure 3-3), that provide reliable mass-data storage for the microsystems. The drives contain double-sided, 5.25-inch, nonremovable disks enclosed in a sealed assembly. A microprocessor within the unit controls the data transfers. The RD51 disk drive can store up to 11 Mbytes of formatted data, and the RD52 disk drive can store up to 31 Mbytes of formatted data.

The RQDX1 controller module, located in the Q22-bus backplane, is the interface between the Q22 bus and the disk-drive units. The controller module establishes the required MSCP protocol to allow direct-memory access between the CPU, memory, and the disk-drive units. Both the RD51 and RD52 units operate from the dc power of the H7864 power supply.

The following are the main features of the RD51:

- 11 Mbytes formatted capacity
- Supports DMA data transfers
- Four surfaces on a single spindle
- 5 Mbits/second (625 Kbytes/second) peak transfer rate
- 93.3 milliseconds average access time
- 8.33 milliseconds average rotational latency

The following are the main features of the RD52:

- 31 Mbytes formatted capacity
- Supports DMA data transfers
- Eight surfaces on a single spindle
- 5 Mbits/second (625 Kbytes/second) peak transfer rate
- 57.5 milliseconds average access time
- 8.5 milliseconds average rotational latency

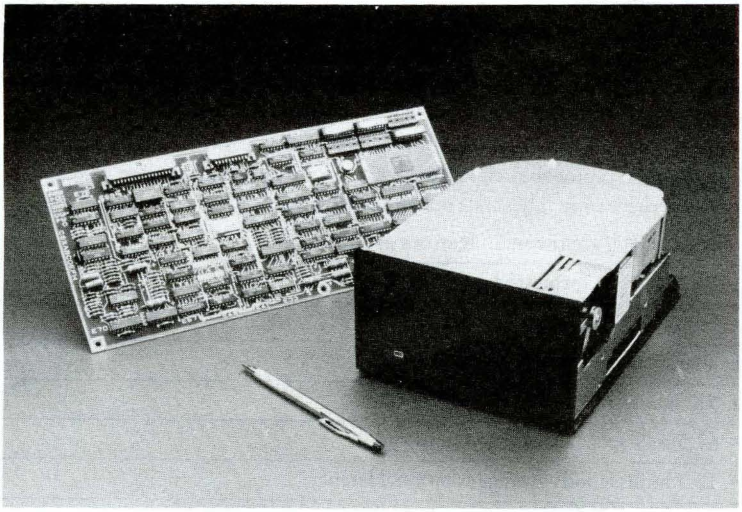


Figure 3-3 ▪ RD51 and RD52 Fixed-disk Drives

RL02 Removable Disk

The RL02, shown in Figure 3-4, is a 10.4-Mbyte, removable-cartridge disk drive for the MicroPDP-11 family. Direct-memory access is used to provide rapid data transfer and efficient utilization of the host processor. The RLV12 controller module, located in the Q22-bus backplane, is the interface between the Q22 bus and the removable-disk unit. The controller module establishes the required protocol to allow direct-memory access between the CPU, memory, and drive unit. The RL02 operates with its own power supply.

The following are the main features of the RL02:

- 10.4 Mbytes formatted capacity
- Supports DMA data transfers
- Two surfaces on a single spindle
- 512 Kbytes/second peak transfer rate
- 67.5 milliseconds average access time
- 12.5 milliseconds average rotational latency

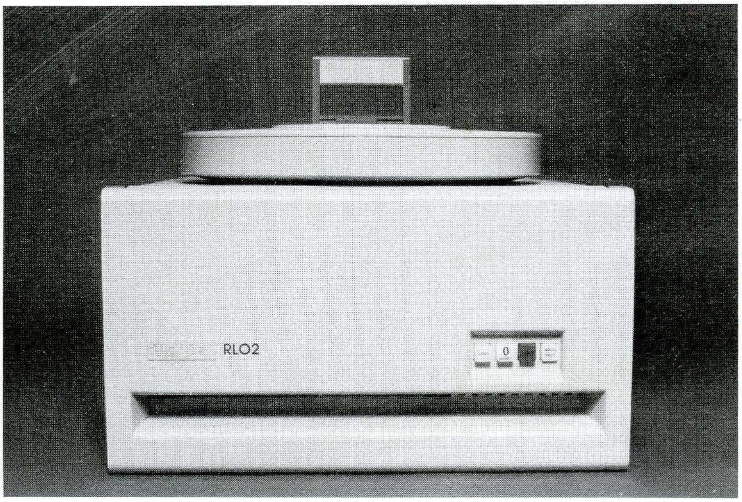


Figure 3-4 ▪ RL02 Removable-disk Drive

RQC25 Fixed/Removable Disk

The RQC25, shown in Figure 3-5, is a 26-Mbyte, Winchester fixed-disk unit combined with a 26-Mbyte, sealed removable-cartridge disk unit for a total of 52 Mbytes. The RQC25 contains its own intelligent controller (KLESI) and onboard microdiagnostics for maintenance. The removable-cartridge portion of the disk provides a one-to-one backup ratio, and is a low-cost alternative to traditional disk/tape configurations. The RC25 is also a 52-Mbyte, Winchester, fixed/removable disk drive but without a controller. It can be added onto the RQC25 to give the microsystem a total of 104 Mbytes of storage.

Because it uses the Mass Storage Control Protocol, the RQC25 is compatible with other Digital Storage Architecture disk subsystems. Exceptional data reliability and integrity features include a powerful 170-bit error detection and correction code, automatic retry and revectoring, embedded servos, and bad-block replacement.

The following are the main features of the RQC25:

- 52 Mbytes formatted capacity
- Four surfaces, single spindle
- 1025 Kbytes/second peak transfer rate
- 45.5 milliseconds peak access time
- 10.5 milliseconds average rotational latency

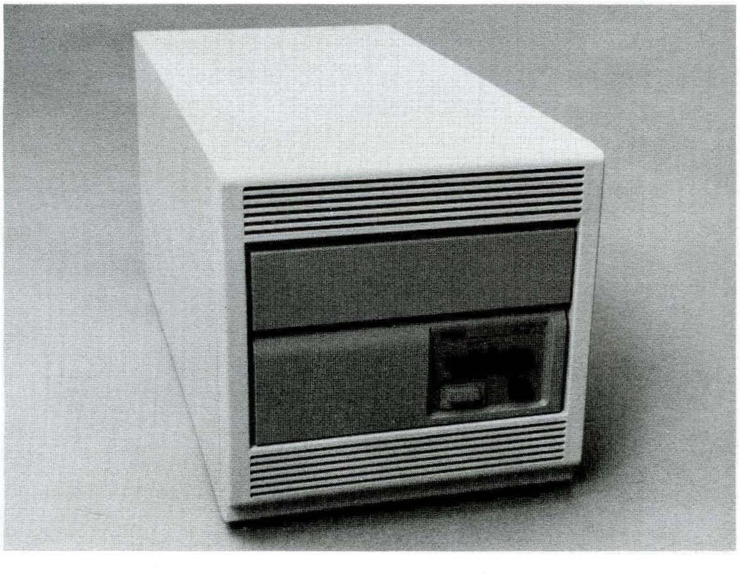


Figure 3-5 • RQC25 Fixed/Removable-disk Drive

TQK25 Cartridge Tape

The TQK25, shown in Figure 3-6, is a cartridge-tape drive packaged in a standalone tabletop enclosure for the MicroPDP-11 family. It is designed for fast backup of the RD51 and RD52 fixed disks. The TQK25 will serially record up to 60 Mbytes on a DC600A 0.25-inch tape that is enclosed in a sealed cartridge. It can copy any of today's mini-Winchester disks onto a single cartridge. The TQK25 cartridge-tape drive also comes with its own universal power supply with cooling, a quad-slot Q22-bus controller module (TS11 emulation), a 16- or 32-inch (41- or 81-centimeter) internal CPU cable, and a 9-foot (2.74-meter) CPU-to-drive cable.

The following are the main features of the TQK25:

- 60 Mbytes formatted capacity
- 8000 bits/inch recording density
- 55 Kbytes/second peak transfer rate
- 55 inches/second read/write speed
- 10 tracks/inch on 0.25-inch DC600A tape

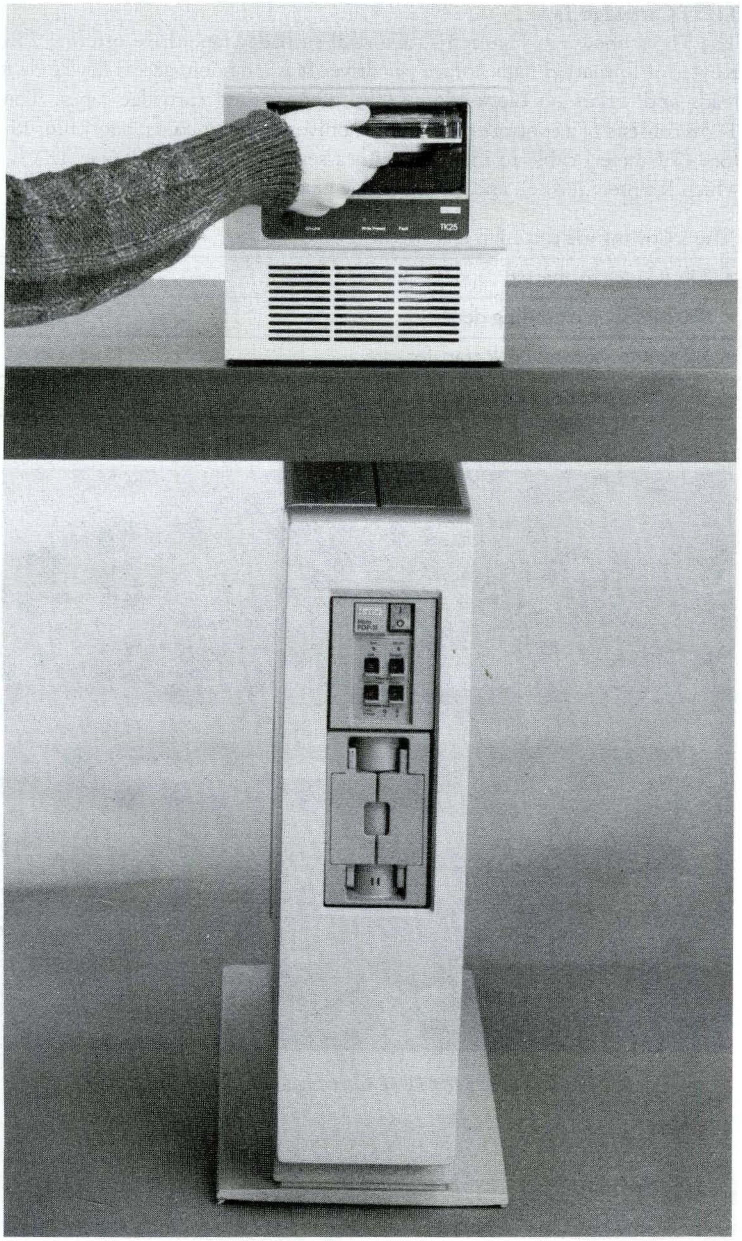


Figure 3-6 ▪ TQK25 Cartridge-tape Drive

TU58 Cartridge Tape

The TU58, shown in Figure 3-7, is a dual cartridge-tape drive offering 256 Kbytes of formatted data storage per drive. It is a random-access device that reads and writes on block-addressable, preformatted cartridge tapes. The TU58 consists of a controller, two drives, universal power cords, boot chip, 18-foot (5.5-meter) cable to interface with a serial-line unit (DLVE1 or DLVJ1, which is a prerequisite), and two cartridge tapes.

The following are the main features of the TU58:

- 256 Kbytes formatted capacity per cartridge
- 800 bits/inch recording density
- 3.7 Kbytes/second peak transfer rate
- 30 inches/second read/write speed

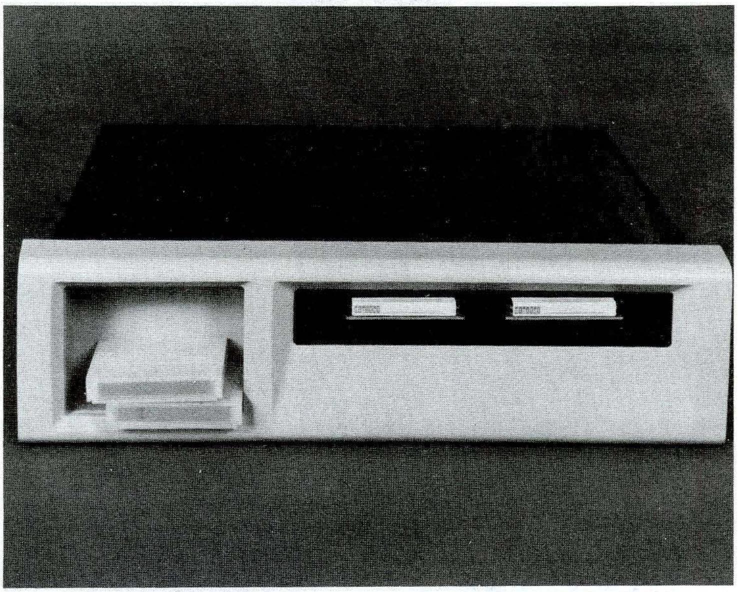


Figure 3-7 ▪ TU58 Cartridge-tape Drive

TSV05 Streaming Tape

The TSV05, shown in Figure 3-8, is a magnetic-tape drive for the MicroPDP-11 family that includes a tape transport with an integral formatter and a single, quad-height controller module. It features a storage capacity of 40 Mbytes using 8-Kbyte blocks, 25/100 inches/second streaming-tape backup, and front-loading automatic tape threading. The TSV05 supports industry-standard 1600 bits/inch phase-encoded format and ANSI compatibility. The tape transport occupies only 8.7 inches (22 centimeters) in a H9642-type 41.7-inch-high (106-centimeter) cabinet, allowing ample room for expansion. The TSV05 is also available without the cabinet for system integrators.

The following are the main features of the TSV05:

- 40 Mbytes formatted capacity (2400-foot reel)
- Front loading
- 1600 bits/inch recording density
- Automatic tape threading
- 40 or 160 Kbytes/second peak transfer rate
- 8.7 inches high
- 25/100 inches/second read/write speed (preset by user)

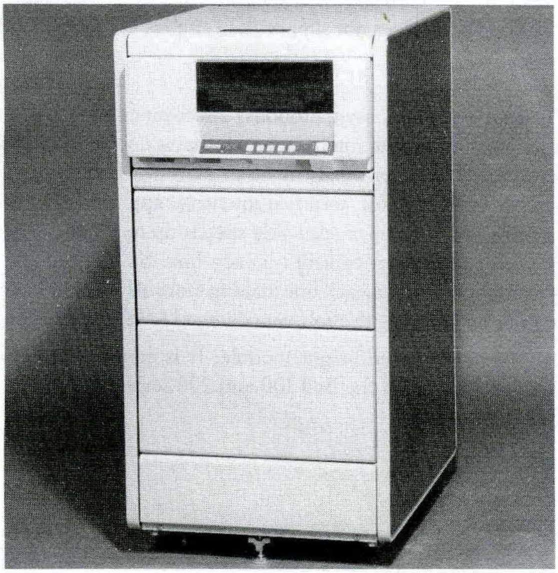


Figure 3-8 ▪ TSV05 Streaming-tape Drive

▪ Communications Options

The communications capability of the each of the microsystems can be expanded by communications interface options. These options enable asynchronous, synchronous, and realtime data transfers between two or more systems, and also between a host system and its user terminals, modems, and other external devices. Each option consists of an interface module, internal cables, and a panel insert. The interface module installs into a slot in the Q22-bus backplane, and the device connector panel insert is mounted in the I/0 distribution panel at the rear of the microsystem chassis. Refer to the *PDP-11 Microcomputer Interfaces Handbook* for detailed information on the following communications options. Appendix E has the ordering information for this handbook.

Asynchronous Interfaces

▪ DZV11

The DZV11 is a four-line, asynchronous multiplexer that provides local or remote interconnection between the microsystems and EIA RS232-C/CCITT V.28 terminals or other systems. The DZV11 operates at program-selectable speeds up to 9600 bits/second at full-duplex with limited-modem control on each line.

The DZV11 is a single, quad-height module. It is compatible with Digital's family of modems and with the Bell 100 and 200 series of modems and their equivalents. The DZV11 does not support half-duplex operations on modems.

▪ DHV11

The DHV11 is an eight-line, asynchronous, direct-memory access multiplexer that provides local or remote interconnection between the microsystems and EIA RS232-C/CCITT V.28 terminals or other systems. Direct-memory access reduces system overhead for terminal-intensive applications. The DHV11 operates at program- or jumper-selectable speeds up to 38,400 bits/second at full-duplex with full-modem control on each line. Split-speed transmit and receive rates are supported on each line making more efficient use of communications facilities by reducing the software demand for the receive line.

The DHV11 is a single, quad-height module. It is compatible with Digital's family of modems and with the Bell 100 and 200 series of modems and their equivalents.

- DZQ11

  The DZQ11 is a four-line, asynchronous multiplexer that provides local or remote interconnection between the microsystems and EIA RS232-C/CCITT V.10 terminals or other systems. The DZQ11 operates at program-selectable speeds up to 9600 bits/second at full-duplex with limited-modem control on each line.

  The DZQ11 is a single, dual-height module. It is compatible with Digital's family of modems and with the Bell 100 and 200 series of modems and their equivalents.

- DLVJ1

  The DLVJ1 is a four-line, asynchronous interface that provides local or remote interconnection between the MicroPDP-11 systems and EIA RS232-C/CCITT V.8, EIA RS422/CCITT V.11, and EIA RS423/CCITT V.10 terminals. The DLVJ1 acts as four separate devices, making program operations more convenient than they are with a multiplexer. The DLVJ1 operates at program- or jumper-selectable speeds from 150 to 38,400 bits/second at full-duplex. Limited-modem control is included. Split-speed transmit and receive rates are supported on each line making more efficient use of communications facilities by reducing the software demand for the receive line.

  The DLVJ1 is a single, dual-height module. It is compatible with Digital's family of modems and with the Bell 100 and 200 series of modems and their equivalents.

  Diagnostic support is available for the DLVJ1 on the MicroVAX I, which is supported only as an OEM add-on device.

- DLVE1

  The DLVE1 is a single-line, asynchronous interface that provides local or remote interconnection between the MicroPDP-11 systems and EIA RS232-C/CCITT V.28 terminals. The DLVE1 operates at program- or jumper-selectable speeds from 50 to 19,200 bits/second at full-duplex. Limited-modem control is included. Split-speed transmit and receive rates are supported making more efficient use of communications facilities by reducing the software demand for the receive line.

  The DLVE1 is a single, dual-height module. It is compatible with Digital's family of modems and with the Bell 100 and 200 series of modems and their equivalents.

Synchronous Interfaces

- DEQNA

The DEQNA is a high-performance, synchronous, communications controller that connects the microsystems to an Ethernet Local Area Network (LAN). The DEQNA complies fully with the Ethernet specification and operates at 10 Mbits/second.

The DEQNA provides Ethernet data-link layer functions and a portion of the physical channel functions. The DEQNA is supported under DECnet Phase IV software. Digital also provides documentation and device drivers so that users can write their own higher-level protocols for specialized applications and communications in multivendor environments. The DEQNA allows communications with up to 1023 addressable devices on an Ethernet. It physically and electrically connects to the Ethernet coaxial cable via Ethernet transceiver cables and an H4000 Ethernet transceiver, or a Local Area Interconnect (DELNI). The Physical Address ROM (DEXMR) is required to downline load software to a diskless Ethernet node with a DEQNA.

The DEQNA is a single, dual-height module.

- DMV11

The DMV11 is a microprocessor-controlled, single-line, synchronous interface that provides local or remote interconnection between the microsystems and other computer systems with EIA RS232-C/CCITT V.28, or RS423/RS449 interfaces. The DMV11 implements DDCMP in hardware and supports direct-memory access data transfers, DECnet point-to-point or multipoint configurations, and full-modem control. It operates at speeds from 19,200 bits/second to 56,200 bits/second (depending on the version selected) at half- or full-duplex.

Depending on the operating system and layered software, the DMV11 can support up to 12 tributaries. In multipoint configurations, these tributaries can be other DMV11s or DMP11s. In point-to-point configurations, the DMV11 can communicate with other DMV11s, DUP11s, DMR11s, or DMP11s.

The DMV11 is a single, quad-height module. It is compatible with Digital's family of modems and with the Bell 200 series of modems and their equivalents.

- DPV11

  The DPV11 is a single-line, synchronous, programmable interface that provides local or remote interconnection between the microsystems and other computer systems with EIA RS232-C/CCITT V.28 or EIA RS232-C/CCITT V.11 interfaces. It operates at speeds up to 56,000 bits/second at half- or full-duplex with full-modem control. The DPV11 is programmable for either byte-oriented protocols (DDCMP or BISYNC) or bit-oriented protocols (SDLC or HDLC). The DPV11 is suited for interfacing to medium-speed synchronous lines for remote batch and remote job-entry applications.

  The DPV11 is a single, dual-height module. It is compatible with Digital's family of modems and with the Bell 200 series of modems and their equivalents.

- KMV11

  The KMV11 is a high-performance, direct-memory access, single-line, programmable, communications controller that provides local or remote interconnection between MicroPDP-11 systems and other computer systems with EIA RS232-C/CCITT V.28, EIA RS422/CCITT V.1, or EIA RS423/CCITT V.10 interfaces. It is capable of communications speeds up to 64,000 bits/second. The KMV11 utilizes the MICRO/T11 processor to perform user-defined communications functions, thereby freeing the host to do more application computations.

  The KMV11 can be programmed in synchronous or asynchronous modes. It is implemented as a single, quad-height module. The KMV11 also provides full-modem support for Digital's family of modems, the Bell 200 series of modems or their equivalents, and European-PPT approved modems.

Realtime Interfaces

- DRV11-WA

  The DRV11-WA is a general-purpose, direct-memory access, parallel-line, interface unit with 22-bit addressing capability. It permits block-mode DMA data transfers at rates up to 250 Kwords/second in single-cycle mode, and up to 500 Kwords/second in burst mode. The DRV11-WA is a single, dual-height module.

- DRV11-J

  The DRV11-J is a general-purpose, program-controlled, parallel-line interface. It contains 64 bidirectional input/output lines configured as four 16-bit ports. It is also bit interruptible on up to 16 lines. Interrupt vectors may have fixed or rotating priorities. The DRV11-J is a single, dual-height module.

- AAV11

  The AAV11 is a four-channel, digital-to-analog converter module for the MicroPDP-11 family that includes control and interfacing circuits. It has four D/A converters, a dc-to-dc converter that provides power to the analog circuits, and a precision voltage reference. Each channel has its own holding register that can be addressed separately and provides 12 bits of resolution. The AAV11 is a single, dual-height module and is available as an add-on option only.

- ADV11

  The ADV11 is a 12-bit, successive approximation, analog-to-digital converter module for the MicroPDP-11 family that samples analog data at specified rates and stores the digital equivalent value for processing. A multiplexer section can accommodate up to 16 single-ended or 8 quasi-differential inputs. The converter section uses a patented auto-zeroing design that measures the sample data with respect to its own circuitry offset and, therefore, cancels out its own offset error. The ADV11 is a single, dual-height module and is available as an add-on option only.

- AXV11

  The AXV11 is an analog input/output module for the MicroPDP-11 family that accepts up to 16 single-ended inputs or up to eight differential inputs, either unipolar or bipolar. The AXV11 also has two separate digital-to-analog (D/A) converters. Each D/A converter has a write-only register that provides 12-bit input data resolution. On receiving the data, the AXV11 changes it to an analog output voltage. The AXV11 is a single, dual-height module and is available as an add-on option only.

- KWV11

  The KWV11 is a 16-bit, programmable clock/counter for the MicroPDP-11 family that provides a variety of means for determining time intervals or counting events. It can be used to generate interrupts to the processor at predetermined intervals, or to synchronize the processor ratios between input and output events. It can also be used to start the ADV11 analog-to-digital converter either by clock counter overflow or by the firing the Schmitt trigger. The KWV11 can be operated in any of four programmable modes—single interval, repeated interval, external event timing, and external event timing from zero base. The KWV11 is a single, dual-height module and is available as an add-on option only.

▪ Terminals, Printers, and Modems

Digital offers a complete line of video terminals, printers, and modems for the microsystems. A terminal or device can be selected that incorporates the features that the application requires. A detailed description of all the video terminals and can be found in the *Terminals & Printers Handbook*. Refer to Appendix E for ordering information for the handbook.

Video Terminals

Video-display terminals use a cathode-ray tube (CRT) screen for output and a typewriterlike keyboard for input. Alphanumeric-video terminals are capable of displaying letters, numbers, and special characters in a fixed format. Graphics-video terminals can individually manipulate picture elements on the display screen and can represent graphs, charts, and pictures. Typically, a graphics terminal can also emulate an alphanumeric terminal. Table 3-1 is a comparison chart of the alphanumeric and graphics video terminals.

Table 3-1 ▪ Video-terminal Features

Model	Universal VT100 Family Features	Keyboard	Advanced Video Features	Graphics	Printer Port	Local Echo	Asychro-nous Communi-cations	Integral Modem
VT101	X	ANSI Numeric	X			X	Full Duplex	OPT
VT102	X	ANSI Numeric Word Processing	X		X	X	Half/full Duplex	OPT
VT125	X	ANSI Numeric Word Processing	X	X	X		Full Duplex	OPT
VT220	X	ANSI Numeric	X		X	X	Full Duplex	
VT240	X	ANSI Numeric	X	X	X	X	Full Duplex	
VT241	X	ANSI Numeric	X	X	X	X	Full Duplex	

▪ VT100 VIDEO-TERMINAL SERIES

The VT100 series of video-display terminals, shown in Figure 3-9, includes a sculptured typewriterlike detachable keyboard that connects to the video-display unit by a 6-foot (1.9-meter) coiled cord. The keyboard is used to set the terminal's functions. The display can be customized by setting tab stops, by reversing the video image, and by changing the cursor from underline to block to suit the application.

The advanced video features, optional on some models, gives three additional capabilities: a combination of 24 display lines and 132 columns, space and connections for an extra character set that resides in the terminal, and a character display in any combination of blinking, underlined, bold, or reversed video.

The VT100 series of terminals operate on full-duplex, asynchronous communications lines and are equipped with a standard EIA RS232-C interface.

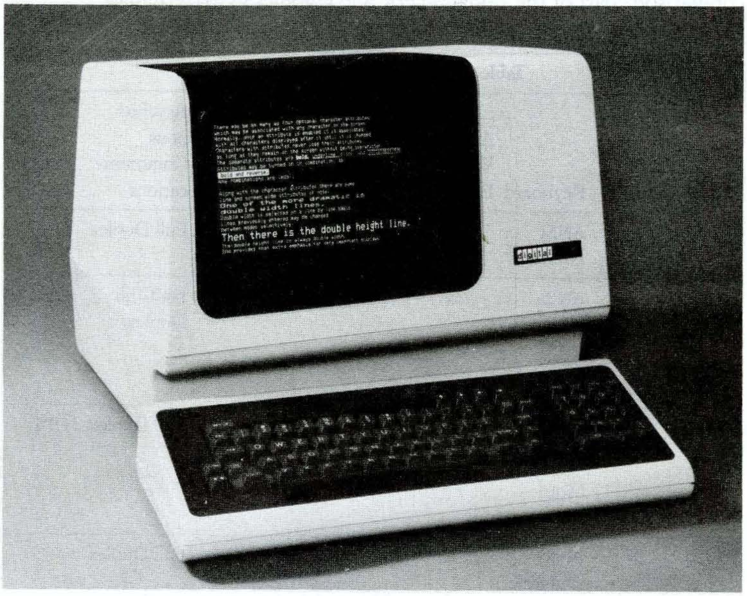


Figure 3-9 ▪ VT100 Video-display Terminal

- VT101

  The VT101, a low-cost video terminal, provides 24 lines of 80 characters or 14 lines of 132 characters. Characters can be underlined or represented in reverse video. The display includes smooth scrolling and split-screen operation. The operator can set the terminal characteristics from the keyboard.

  Communications with the system are full-duplex and asynchronous.

- VT102

  The VT102 terminal functions like a VT100 terminal with the advanced video and printer port options. The monitor displays 24 lines of 80 or 132 characters with individually selectable attributes including bold, blink, underline, or reverse video. Double-width, double-height characters can be used for legibility at a distance. Text can scroll smoothly up the screen rather than jump a full line at a time. The screen can be split into scrolling and nonscrolling regions. Many of the terminal's operating characteristics can be changed from the keyboard.

  Asynchronous communication with the system can be full-duplex, full-duplex with asymmetric speeds, or half-duplex with modem control. Automatic synchronization codes (XOFF tells the computer to stop sending data, XON to start again) allow fast data transfers without the loss of information. The VT102 can connect to a dedicated printer, such as an LA100 or LA50, to produce a hardcopy printout of the video screen.

  The VT102 video terminal is also available in a rugged version for use in harsh environments. The RT102 has a sheet-steel case, a heavy-duty filtration system, and a sealed keyboard.

- VT125

  The basic VT125 combines bit-mapped, monochrome graphics and a printer port with VT100 functionality. The advanced video option increases the number of 132-character lines to 24 and allows selection of any combination of bold, blink, underline, and reverse video attributes on a character-by-character basis.

  VT125 graphics are described by the Remote Graphics Instruction Set (ReGIS), which is executed by the VT125's microprocessor. ReGIS is the high-level language interface to the VT125. ReGIS considers a picture to be a group of graphic objects where each object is a geometric shape that can be described by a few characters of information. For example, ReGIS understands that the shape of a circle applies to any circle that can be drawn and that it can be described on the screen by locating its center and a point on its circumference. The same kind of understanding for other graphic objects allows ReGIS to be a compact descriptor of graphic images. The VT125 terminal's screen resolution is 768 (horizontal) × 240 (vertical). There are two graphics planes that can be displayed one at a time or at the same time. When displayed together, the two planes provide four monochrome intensities on the VT125 terminal screen—black, white, and two shades of gray. When attached to an external color monitor, the VT125 can generate four colors from a palette of 64 using both graphics planes. The VT125 can be connected directly to a graphics printer such as the LA100 or LA50 for a hardcopy printout of the screen.

▪ VT200 VIDEO-TERMINAL SERIES

The VT200 series is Digital's newest offering of video terminals. The VT200 terminals include all of the universal features of the VT100 series of video terminals. The VT200 units are smaller in size than the VT100 units and include low-profile packaging. A series of setup menus make tailoring the terminal to the application very easy.

▪ VT220

The VT220, a monochromatic, text-only terminal, shown in Figure 3-10, incorporates full VT100 functionality. The terminal features include a 12-inch nonglare screen, low-profile packaging, and an adjustable monitor. The VT220 terminal includes VT52 terminal emulation, advanced-video features, built-in printer port, and U.S.A./European modem controls. The international capabilities include a multinational character set, universal power supply, and both 20 milliampere and EIA interfaces. A plain-language set-up menu, programmable function keys, and selective-erase feature combine to make the VT220 terminal easy to use. Operator-oriented features such as split screen, bidirectional smooth scrolling, double-height and double-width characters, and reverse video allow the VT220 terminal to be used for many applications.

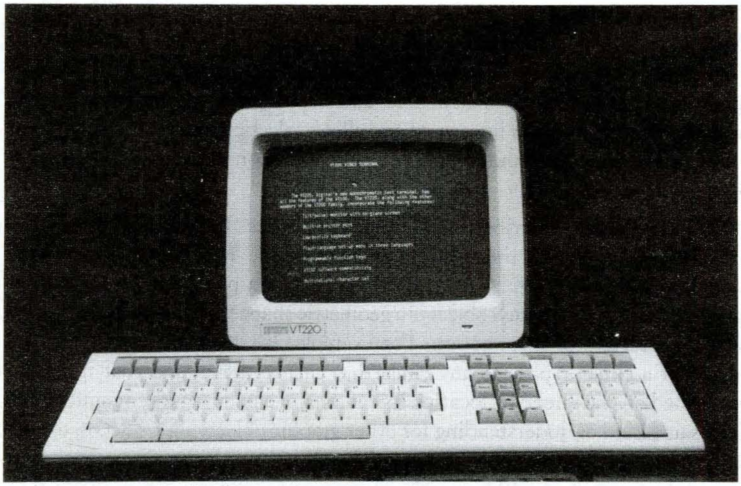


Figure 3-10 ▪ VT220 Video-display Terminal

▪ VT240 AND VT241

The VT240 is a monochromatic, text and graphics video-display terminal. It incorporates all the features of the VT100 family and the VT220 terminal, is completely self-contained, and requires no upgrading.

The VT240 terminal supports the industry's graphics standards by generating full bit-mapped graphics in both ReGIS and Tektronics 4010/4014 emulation. ReGIS is Digital's general-purpose graphics descriptor. It allows pictorial data to be created and stored very easily. By connecting this terminal to a graphics printer, such as the LA100, the contents of the display can be reproduced. The VT240 terminal consists of the same keyboard that is used for the VT220 terminal, a monitor, and a system box. The system box contains the power supply, control-circuit board, and electrical connectors.

The VT241 terminal offers all the capabilities of the VT240 terminal and includes a color monitor for four-color text and graphics output. Both the VT240 and the VT241 are shown in Figure 3-11.

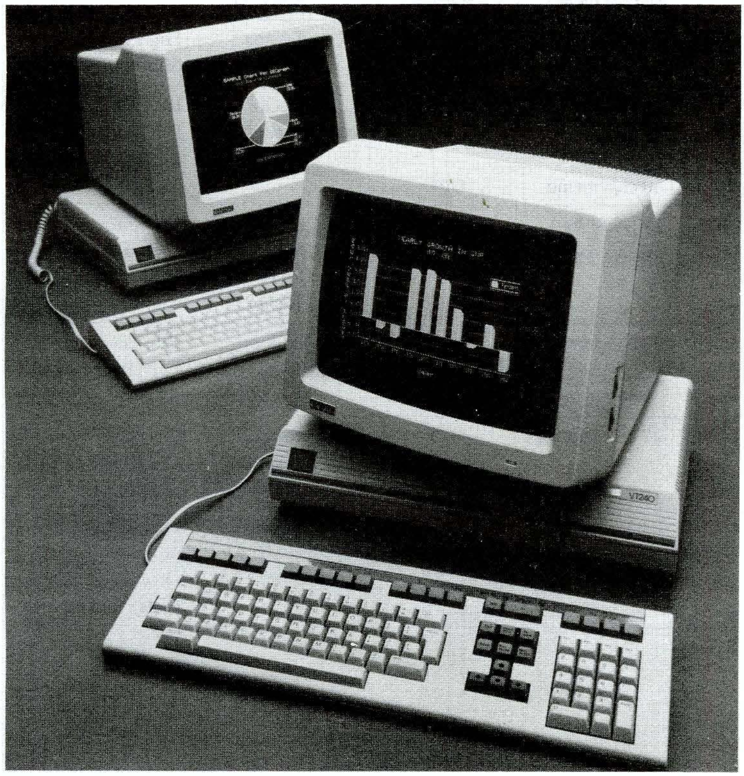


Figure 3-11 ▪ VT240 and VT241 Video- and Graphics-display Terminals

Printing Terminals

Several printing terminals are available for operation with the microsystems. These terminals include large, free-standing units and small, desktop units. Table 3-2 provides a comparison of the features available with the printing terminals.

Table 3-2 ▪ Printing-terminal Features

Model	Print Speed and Quality	Graphics	Parts per Form	Paper Feed	Special Features
LA50	100 ch/s draft 50 ch/s memo	X	3	Friction Low tear-off tractor	DEC PC-compatible
LA100	240 ch/s draft 30 ch/s letter 80 ch/s memo	X OPT	4	Friction Tractor Sheet OPT Roll OPT	DEC PC-compatible Plug-in fonts
LA120	180 ch/s draft		9	Tractor	High-volume printing
LA210	240 ch/s draft 40 ch/s letter 80 ch/s memo	X OPT	4	Tractor Sheet OPT	IBM PC-compatible DEC PC-compatible 10 languages
LA12	150 ch/s draft 80 ch/s memo	X	2	Friction Pin Roll Tractor OPT	Dial through the keyboard modem and/or acoustic coupler
LQP02	32 ch/s letter	OPT	4	Friction Sheet OPT Tractor OPT	WPS-compatible Full-character printing
LQP03	27 ch/s letter	OPT	4	Friction Sheet OPT Tractor OPT	PC-compatible Full-character printing
LN03	8 pages/min		1	Cutsheet	High-resolution printing Downline-loadable fonts

▪ LA50 PERSONAL PRINTER

The LA50, shown in Figure 3-12, is a low-cost, tabletop printer. Its impact dot-matrix mechanism prints 7 × 9 matrix characters at 100 characters/second in draft mode, 13 × 9 characters at 50 characters/second in memo mode, as well as bit-map graphics.

Characters can be printed at pica (10) or elite (12) pitch, as well as compressed (16.5) for 132 characters on a line. Each pitch can also be printed in double-width characters. The LA50 has 250 printable characters, including:

- The standard 96-character ASCII set
- VT100 terminal special-character set
- 8-bit multinational-character set
- 11 national-character sets

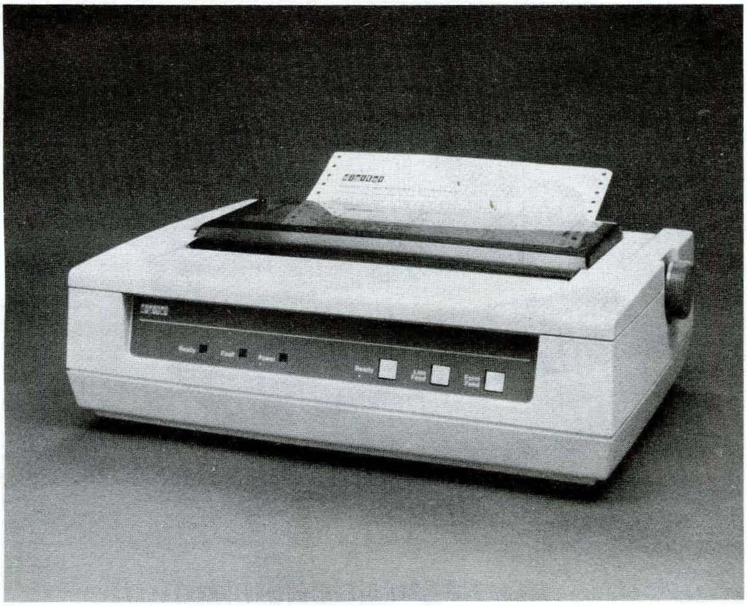


Figure 3-12 ▪ LA50 Personal Printer

▪ LA100 (LETTERPRINTER 100 AND LETTERWRITER 100)

The LA100 printing terminals, shown in Figure 3-13, provide flexibility in a tabletop unit about the size of an electric typewriter. The Letterwriter 100 is a keyboard send/receive (KSR) terminal, and the Letterprinter 100 is receive-only (RO). Both can print on friction-fed paper or with an optional tractor-fed paper.

The impact dot-matrix printer offers you a choice of print quality and speed. In draft mode, it prints 240 characters/second maximum with a 7 × 9 print matrix. At the other extreme, letter-quality mode gives you a 33 × 18 print matrix at 30 characters/second. In between these modes is the optional memo-quality mode with its 33 × 9 print matrix and 80 characters/second speed. Standard fonts included with the LA100 are Courier-10 and Orator-10. Optional fonts include Gothic-10, Symbol-10, Courier-12, and many others. Fonts can be selected by the system or from the keyboard.

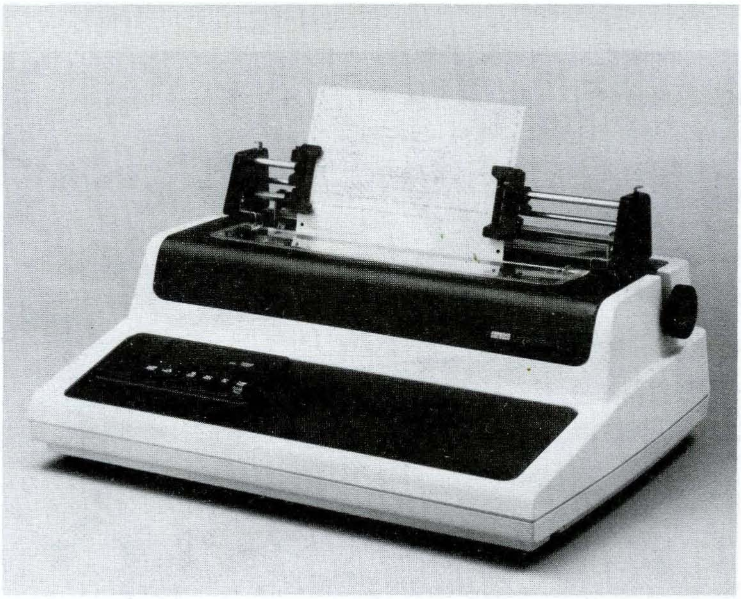


Figure 3-13 ▪ LA100 (Letterprinter 100)

▪ LA120 DECWRITER III PRINTER

The LA120 freestanding printing terminal, shown in Figure 3-14, is a sturdy, high-performance device with a reputation for reliability. It prints on multiple-copy, tractor-fed paper from 3 inches to 14.8 inches wide, at a maximum speed of 180 characters/second—fast enough to print a typical one-page memo in 20 seconds. Because of its 1000-character buffer, bidirectional printing, and ability to skip quickly across spaces, the LA120 printer maximizes the printing throughput.

The standard LA120 printer character set is U.S.A./United Kingdom. Character sets for Europe or the APL programming language are optional. Characters can be printed in eight sizes and in six choices of vertical-line spacing. Characters are formed by an impact dot-matrix printhead in a 7 × 7 dot format. Both keyboard send/receive (KSR) and receive-only (RO) versions are available.

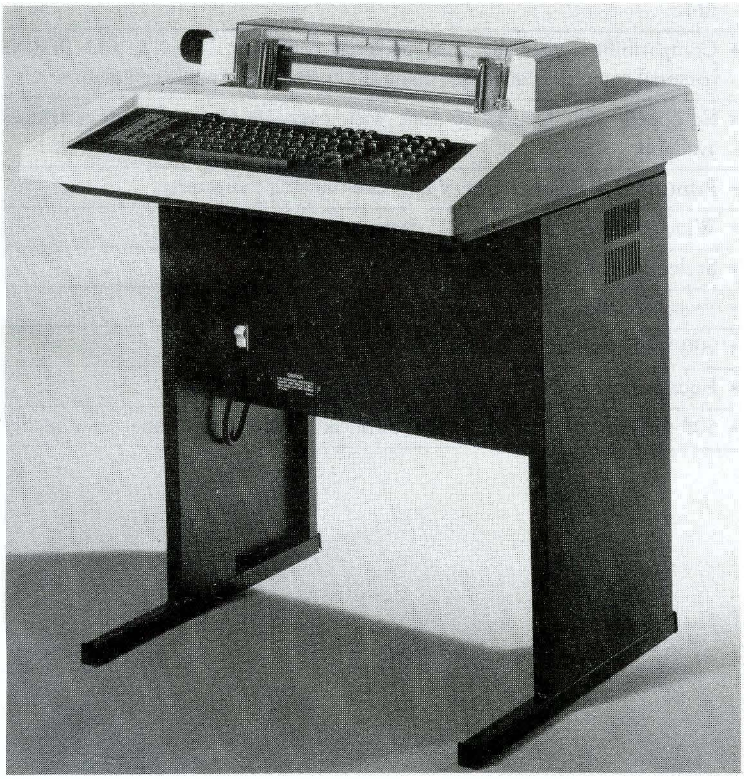


Figure 3-14 ▪ LA120 DECwriter III Printer

▪ LA210 LETTERPRINTER

The LA210 Letterprinter, shown in Figure 3-15, is a microprocessor-driven, medium-speed, wide-carriage, receive-only (RO), multimode printer that offers similar functionality to the LA100 personal-computer model plus the addition of IBM personal computer compatibility. The LA210 uses conventional impact dot-matrix printing technology. Dot formations may be of either letter quality (lower speed/higher density), draft quality (higher speed/lower density) and a binary-print mode that allows the host computer to define all dots printed (graphics mode). The three basic modes of printer operation cover a wide range of applications. The LA210 has been designed with the following standard capabilities:

- 240 characters/second draft printing speed
- 40 character/second letter-quality printing speed
- Compatibility with Digital and most IBM bit-map graphics printing applications
- Compatibility with the IBM personal computer, IBM-XT, and IBM-AT personal computers and with many IBM personal-computer emulators
- Equipped with Courier-10 font with over 30 optional font cartridges available
- Prints in ten languages plus VT100 line-drawing characters
- Wide carriage prints up to 217 characters on 15-inch paper
- Styled for office environment
- Forms-handling tractor with acoustic shield
- 2000-character input buffer
- Equipped with EIA RS232-C interface
- 500-million character laminated printhead delivers exceptional print quality

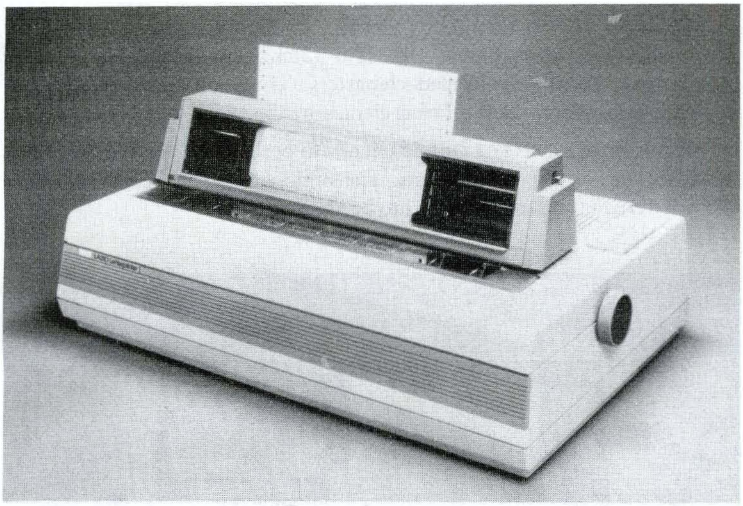


Figure 3-15 ▪ LA210 Letterprinter

▪ LA12 CORRESPONDENT

The LA12 Correspondent, shown in Figure 3-16, is a high-quality, portable printing terminal, with an integral modem and an acoustic coupler. The LA12 includes 9 × 9 print dot-matrix characters at a maximum of 150 characters/second, as well as bit-map graphics. Unlike most portable teleprinters, the LA12 uses plain (not thermal) paper. Character sets and formatting features are similar to those of the LA120 and LA100 printers.

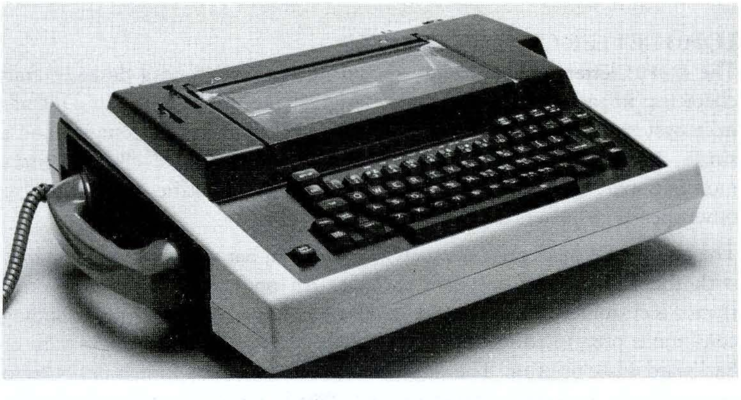


Figure 3-16 ▪ LA12 Correspondent

▪ LQP02 LETTER-QUALITY PRINTER

The LQP02, shown in Figure 3-17, is a 32-character/second letter-quality printer that prints fully formed characters. There are over 100 different interchangeable daisywheel printing elements from which to select.

LQP02 uses single sheets or fanfold paper. An optional tractor allows bidirectional feeding of preprinted forms. The optional dual-tray cutsheet feeder allows letterhead and second sheets to be alternated.

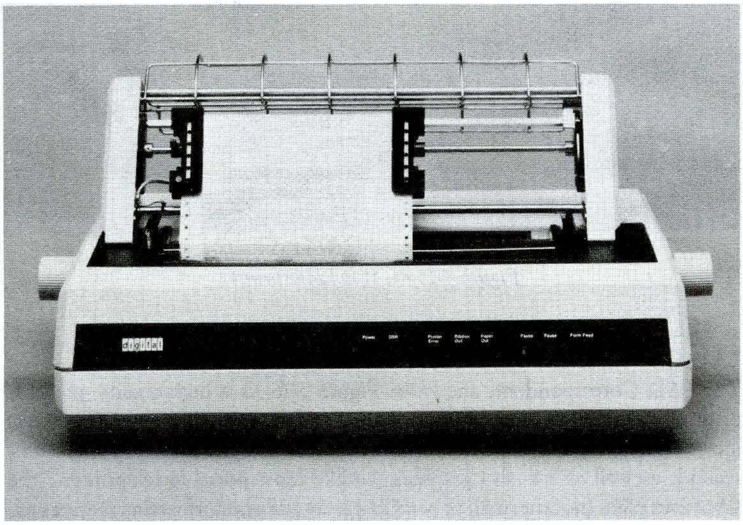


Figure 3-17 ▪ LQP02 Letter-quality Printer

▪ LQP03 LETTER-QUALITY PRINTER

The LQP03 letter-quality printer, shown in Figure 3-18, is a desktop, full-character, impact printer especially designed for use with all of Digital's microsystems. The LQP03 includes an expanded character set contained on a single, 130-petal daisywheel. The daisywheel allows use of all of Digital's multinational characters on one wheel, and provides scientific, mathematic, or other special characters on another.

The printer produces graphics such as pie charts, bar charts, and line graphs with the Daisy-Aids™ software package. An optional bidirectional forms tractor is customer-installable and handles a variety of fanfold paper including continuous preprinted forms. It permits the paper to be scrolled forward or backward while printing. The single-tray cutsheet feeder option is designed to automatically feed precut paper to the LQP03 in either portrait or landscape fashion.

The LQP03 runs on a variety of Digital's operating systems and layered software products, plus software packages from other manufacturers. Depending on the software support, the LQP03 can also perform overprinting, bold facing, underlining, subscripting, and superscripting. The LQP03 includes a standard serial interface for compatibility with existing Digital printers.

TMDaisy-Aids is a trademark of Escape Computer Software, Inc.

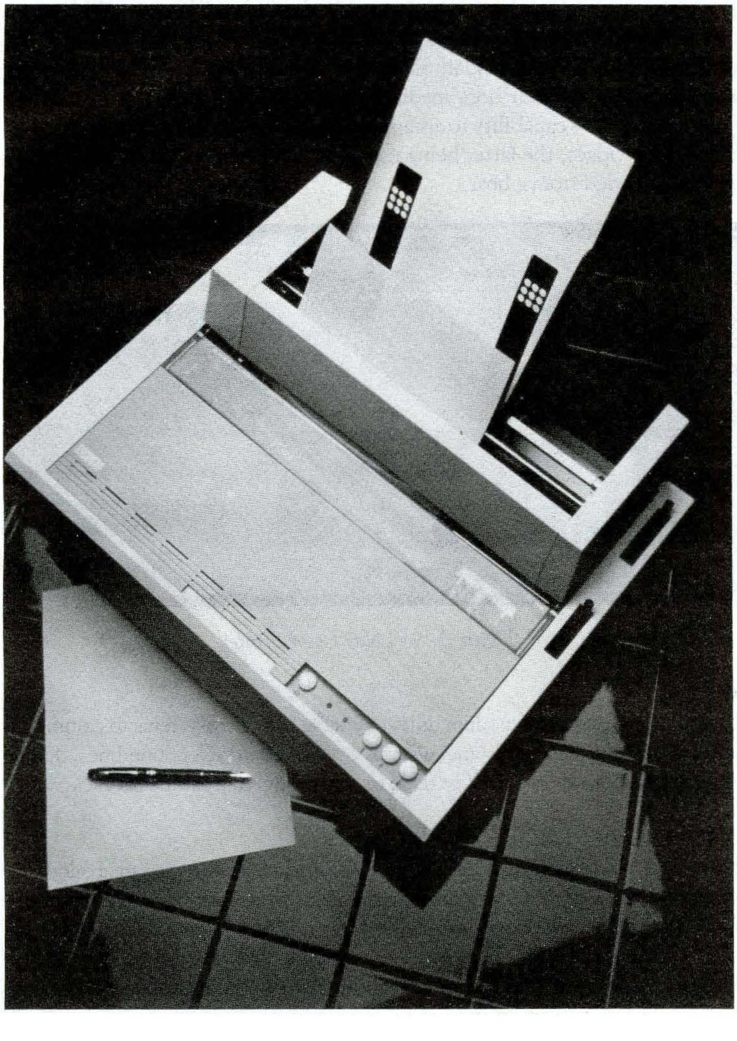


Figure 3-18 • LQP03 Letter-quality Printer

▪ LN03 LASER PRINTER

The LN03, shown in Figure 3-19, is an eight-page/minute laser printer that can be used as a shared resource for microsystems users and as a remote printer for local area networks.

The LN03 is a compact printer, packaging the print engine and the controller in one tabletop unit. With resolution of 300 dots × 300 dots/inch, the print quality of the LN03 is outstanding on both cutsheet paper and transparencies. The input paper capacity of the LN03 is 250 sheets, complemented by a paper path that automatically collates the pages in a 250-sheet output tray. In addition to the 16 fonts resident in the LN03, a wide variety of optional fonts will be offered in both host media and cartridge formats. This flexibility reflects the LN03's capability to accept two ROM (i.e., precoded typefaces) and/or RAM cartridges, the latter being used to receive fonts or forms that can be downline-loaded from a host.

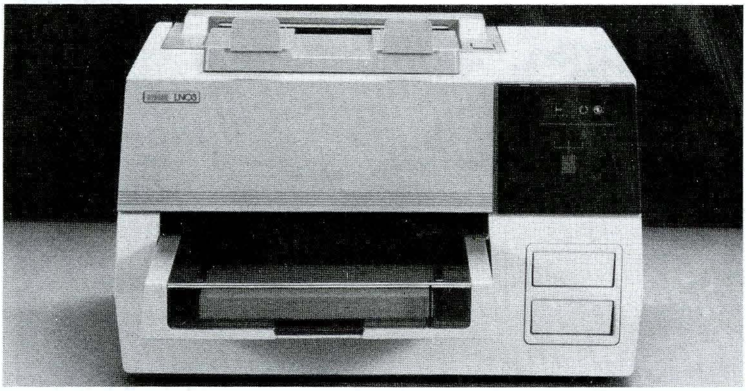


Figure 3-19 ▪ LN03 Laser Printer

Modems

Digital provides several modem units to enable the microsystems to communicate with remote terminals and systems over standard telephone lines. These units can be placed on a desk or table or mounted into a cabinet or rack.

▪ DF02 AND DF03 MODEM UNITS

The DF03 modem, shown in Figure 3-20, connects directly to a modular telephone jack and provides both synchronous and asynchronous communications over dialup telephone lines. The DF03 unit is compatible with the Bell 212A and 103J data sets. It provides synchronous communication from 0-300 bits/second and synchronous or asynchronous communications at 1200 bits/second. The DF03 can be used with any standard telephone for manual-call origination or is available with serial asynchronous autodial capability that can be controlled by a computer system or terminal connected to the EIA port.

The following are the main features of the DF03:

- Dual application modem—supports 0 to 300 and 1200-bit/second asynchronous operation, or 1200-bit/second synchronous operation
- Conforms to FCC Part 15 requirements
- Approved for direct connection to public-switched telephone network by FCC Part 68
- Quick and easy installation, maintenance, and operation
- Automation originate, answer, and disconnect capabilities
- Single, serial-data port for both automatic dialing with autocall feature and normal data
- Multiple-modem modules certified for operation in Canada by the Department of Communication
- Multiple-modem modules support EIA cables up to 200 feet

The DF02 is physically similar to the DF03 modem, transfers at 300 bits/second, and has both the autodial and autoanswer capability.

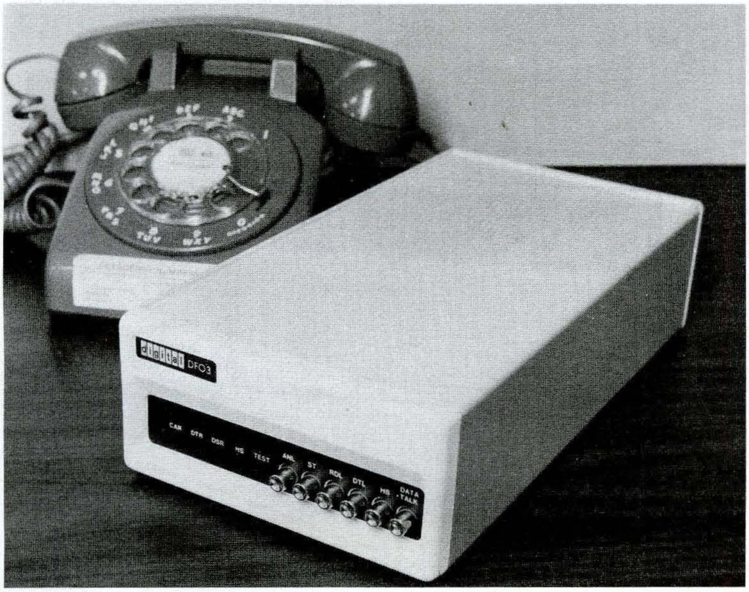


Figure 3-20 ▪ DF03 Modem Unit

▪ DF100 MODEM ENCLOSURE

The same functionality provided by the DF03 unit is available as a module that mounts into the DF100 enclosure shown in Figure 3-21. This enclosure houses up to 12 DF03 modules and can be installed into a 19-inch (48.26-centimeter) cabinet or rack. The modem modules are available with or without the autodial capability.

The following are the main features of the DF100:

- Houses up to 12 modem modules for cost reduction and space savings
- Single, complete unit that is easily installed in a cabinet or rack
- Contains internal power supply and provides space for optional power regulator
- Permits easy servicing with online replacement of modem modules
- Device and communication line cables are easily connected by the user

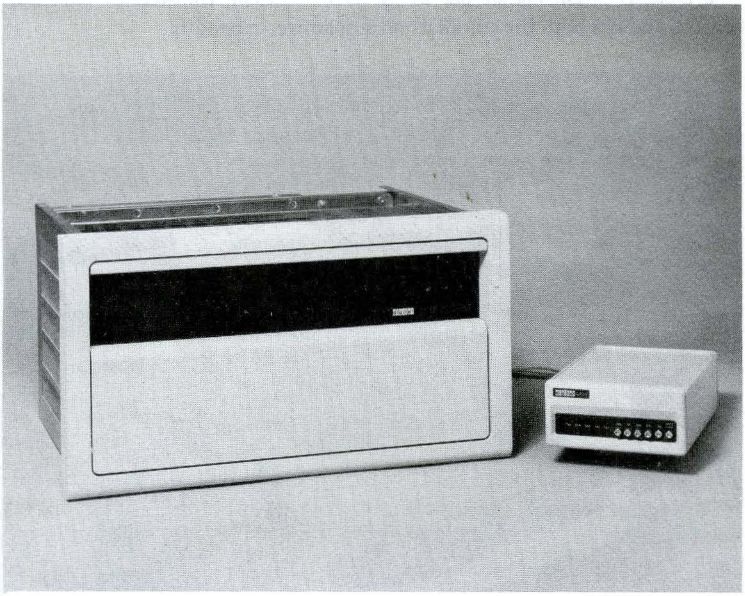


Figure 3-21 ▪ DF100 Modem Enclosure

▪ Additional Documentation

PDP-11 Systems and Options Catalog	*ED-26051-41*
Microcomponents Product Catalog	*EA-30248-68*
PDP-11 Microcomputer Interfaces Handbook	*EB-23144-18*
Terminals & Printers Handbook	*EB-23909-54*

Chapter 4 • System Software and Layered Products

• Introduction

Software is a collection of programs and routines that allow the computer user to utilize the computer hardware. System software comprises the operating system, editors, and utilities that are created for use on a specific processor or processor family. Layered products are made up of high-level languages, information management products, programmer productivity tools, and communications software products. Layered products work in conjunction with a specific operating system.

Although the MicroVAX and MicroPDP-11 are architecturally similar in many respects, they each have software designed specifically for their architectures. This chapter will briefly describe the many operating systems and layered products that run on each of the microsystems.

• MicroVAX Operating and Development Systems

The full capabilities of the MicroVAX I system can be reached through the MicroVMS operating system, the ULTRIX-32m operating system, and the VAXELN development tool kit.

The MicroVMS and ULTRIX-32m operating systems organize and allocate system resources and files, protect data, and monitor system operations. A large set of programming languages, utilities, and application software is available to support these operating systems. The VAXELN development tool kit helps the programmer to develop dedicated applications for the MicroVAX.

The key attributes of these systems are summarized in Table 4-1.

MicroVMS Operating System

MicroVMS is a general-purpose, virtual-memory operating system for the MicroVAX I. It simultaneously supports realtime, batch, and interactive timesharing applications. Through virtual memory, the hardware and software interact to allow the physical memory of the system to be extended onto disk space. Some of the advantages of this system are:

- The 32-bit architecture supports 4 Gbytes of virtual address space to permit large programs to be created and executed.
- Paging and swapping enable more programs to be executed than can be executed in systems with only physical memory.
- Efficient memory rearrangement selects stored programs to be mapped into physical memory as required.
- Extensive system management capabilities control program behavior to minimize disk access and maximize program performance.

▪ MICROVMS FEATURES

The MicroVMS operating system provides the same native-mode runtime environment offered by the VAX/VMS operating system. All of the basic VAX/VMS operating system features are included except for support of the PDP-11 compatibility mode. MicroVMS is a multifunction, virtual-memory system with extensive VAX/VMS system capabilities in file organization and access, program development, and information management, as well as the basic system utilities and the Digital Command Language. A wide range of applications and operational environments can be accommodated. Program development is fast and simple and there is virtually unlimited room for growth or migration to larger VAX/VMS systems.

All native-mode, nonprivileged VAX/VMS applications will execute on MicroVAX I without modification. Many of the Digital-supplied language compilers, productivity tools, information management products, and communications products are available as optional products with MicroVMS software. Applications can easily coexist with, and migrate between, MicroVMS and VMS systems so that the user's investment in applications development and support is protected.

MicroVMS software is in modular form and starts with an *Extended Base Kit* for running applications. The Extended Base Kit consists of three pieces—the base system, the secure user environment option, and the common utilities option. The last two pieces may be installed if necessary but are not required. In addition to the Extended Base System Kit is a *Program Development Kit* that is required to do program development. Because the MicroVMS software is modular, it is possible to effectively support systems with small secondary storage capacities, thus reducing the total system cost.

The MicroVMS operating system and optional communications hardware and software products provide extensive communications capabilities for the MicroVAX I system. These capabilities include remote system and local area network communications. Data access to any VAX system in the network, including clusters, is possible. When used with the VAX/VMS operating system, these facilities form a fast and efficient means of transferring information in large commercial and technical applications.

- MICROVMS PROGRAM DEVELOPMENT

  Program development support is provided by Digital's high-level languages. These languages use the Common Language Environment (CLE) that makes them conform to a specific set of standards. CLE allows each MicroVMS language to interact easily with system services, libraries, and applications written in other MicroVMS languages.

  VAX RMS (Record Management Services) provides a standard for I/O from all languages. This allows files written by one application to be read and used by another application, even if it is written in a different language. Use of the VMS Runtime Library (RTL) provides an integrated, functional base for all MicroVMS languages. A symbolic debugger is included with MicroVMS that can be used to debug programs written in any MicroVMS language. Unique devices can also be interfaced to the system by writing a driver to support the device.

  Many application programs written by Digital and third-party vendors are available for program development.

ULTRIX-32m Operating System

ULTRIX-32m is Digital's enhanced native-mode UNIX™ operating system for the MicroVAX I. This software product is derived from the UNIX Version 4.2 from the University of California at Berkeley. The ULTRIX-32m operating system is compatible with the ULTRIX-32 operating system for larger VAX processors. The product is complementary to MicroVMS and VAXELN and addresses the needs of users seeking a commodity operating system on low-end 32-bit systems.

Although no UNIX system standard yet exists in the industry, the Berkeley implementation is widely recognized as the premier UNIX system adaptation on 32-bit systems. Because the Berkeley system was specifically developed to take advantage of the virtual architecture of VAX systems, it optimizes the hardware so that it performs more efficiently than other versions of the UNIX system.

The ULTRIX-32m system is a repackaging, not a subset, of the full ULTRIX-32 product for larger VAX processors. It is designed to provide the complete set of ULTRIX-32 operating system capabilities for the smaller MicroVAX I system.

™UNIX is a trademark of AT&T Bell Laboratories.

▪ ULTRIX-32m FEATURES

The ULTRIX-32m operating system offers a wide range of programmer productivity tools and networking capabilities.

The Bourne and C shells serve as the command-language interfaces into the system. They are both programmable and thus allow for a tailorable user environment.

The ULTRIX-32m system provides a hierarchy of named directories and subdirectories. The number of levels is limited only by physical space. Utilities that aid in system and file maintenance include line and screen editors, file interchange, print spooler, and system installation and maintenance.

The C programming language interfaces with ULTRIX-32m and the UNIX system in the same manner. As a result, ULTRIX-32m or UNIX system applications that are written in C can be easily moved from one UNIX machine to another. ULTRIX-32m can also be programmed in assembly language.

ULTRIX-32m's communications facilities include TIP (remote login), UUCP (phone network), mail, and Ethernet.

VAXELN Development Tool Kit

The VAXELN development tool kit is a layered product that supports VAX/VMS program development with a high-level implementation language (PASCAL). A finished VAXELN program will run on a VAX or a MicroVAX without an operating system. Typical applications include industrial automation, workstations, Ethernet server networks, and any other application in which individual processors have dedicated functions. These applications should not be needed simultaneously for general computing, program development, or other uses for which a general operating system is more appropriate.

Typically, VAXELN applications are realtime applications. VAXELN simplifies the design and implementation of such applications by offering the PASCAL language, a kernel executive, and optional, pregenerated service programs and device drivers.

▪ VAXELN FEATURES

Both multitasking and multiprogramming are supported by the VAXELN kernel and VAXELN PASCAL programming language. With multitasking, the execution of a task's instructions is interleaved with the instructions of all other tasks on the same CPU, giving the effect of concurrent execution. With multiprogramming, entire programs including multitasking programs can be scheduled concurrently on the same CPU.

™UNIX is a trademark of AT&T Bell Laboratories.

VAXELN systems can run on autonomous VAX or MicroVAX computers or, with networking software provided in the tool kit, they can be connected in an Ethernet local area network. The network may include VAX/VMS nodes or other nodes using DECnet-VAX services and protocols. Once written, VAXELN PASCAL programs can be redistributed among the nodes in a network without changing their code.

Finished VAXELN systems can be loaded onto portable Files-11 B storage volumes and booted from them on the MicroVAX I. If the user has the optional DECnet-VAX license and Ethernet hardware, VAXELN software can be loaded downline from the development computer to the MicroVAX I.

VAXELN system images can also be stored in and booted from read-only memories (ROMs).

▪ VAXELN SYSTEM DEVELOPMENT

For the development of VAXELN systems, the tool kit can be used on any VAX computer running a current version of the VAX/VMS operating system. Together with other software modules, the user receives the following development utilities (VAX/VMS program images) in the VAXELN tool kit:

- VAXELN PASCAL compiler
- VAXELN debugger
- VAXELN system builder

Although previously existing VAX programming languages can be used in VAXELN system development, the *VAXELN PASCAL compiler* is provided in the tool kit. VAXELN PASCAL is a highly optimizing, native-mode compiler extended to eliminate the need for other programming languages including assembly language. It provides a consistent PASCAL language for all system programming problems, including the writing of interrupt-service routines and blocks for concurrent processes. It is the primary implementation language of the VAXELN tool kit.

User programs are written with the aid of the usual VAX/VMS text editors and utilities and then compiled. The compiled code is linked, using the standard VAX/VMS Linker, to a special runtime library also supplied with the tool kit. The runtime library provides special support for VAXELN PASCAL I/O operations, standard PASCAL routines such as SIN, and certain procedures used in system programming. It is provided both in object-library and shareable-image forms in the tool kit. Only those shareable images containing code called by applications programs are included in the finished VAXELN system. This makes a finished system with a minimal amount of unused code while still maintaining maximum ease of use in program development.

The *VAXELN debugger* is used to debug programs in a developed, executing VAXELN system. It can be used to debug a VAXELN system locally using the target computer's console terminal. If the user has the optional DECnet-VAX license and Ethernet hardware, the debugger can be used remotely to debug VAXELN systems running on Ethernet nodes from a programmer's terminal on a VAX/VMS node.

The *VAXELN system builder* combines program images, the VAXELN kernel image, and runtime routines to produce an image of the finished VAXELN system. The program images can be any user-written programs developed in VAXELN PASCAL or any of the images supplied with the tool kit. The program images can also be the routines written in other VAX languages provided that they do not call VAX/VMS system services or language-specific runtime routines, such as I/O.

Also included in the tool kit are a number of images ready for inclusion in the user's VAXELN system:

- The *VAXELN kernel* is included in every VAXELN system. It manages the system's processes and data, providing the controlled sharing of the system's resources. The kernel operates on a special class of objects (processes, events, semaphores, messages, message ports, devices, and message-port names), which are represented as predeclared data types in the language. The operations of the kernel are reflected in VAXELN PASCAL programs by calls to special predeclared procedures.
- The *VAXELN file service* supports I/O operations from VAXELN PASCAL programs to file-storage devices such as disks, as well as remote file access from DECnet-VAX nodes. I/O requests from the user's PASCAL programs (calls to the I/O runtime routines) are interpreted by the file service and performed by the appropriate device-driver program. The file service and the tool kit's disk-driver programs use the Digital Data Access Protocol (DAP) Version 7.0 for their internal I/O interface. User-written drivers can run compatibly with the file service by following this protocol, and programming tools are supplied in the tool kit for this purpose.
- The *VAXELN network service* provides completely transparent network communications between VAXELN nodes in a local area network, and between VAXELN nodes and DECnet-VAX nodes. In network applications, each VAXELN node runs its own VAXELN system, and each system is built including the network service. Given such a configuration, the network locations of VAXELN applications programs are completely invisible to each other. A program can communicate with a program on another node using precisely the same statements as if both programs were on the same node.

- Device drivers are supplied for the commonly used UNIBUS (VAX) and Q22-bus (MicroVAX) devices. All are implemented in VAXELN PASCAL and are supplied both in source form and in image (binary) form. The driver sources can be used as templates for user-written drivers.
- Programming aids are supplied, such as template-device drivers, PASCAL declarations of Data Access Protocol (DAP) interfaces, and declarations of exception arguments.

▪ MicroPDP-11 Operating and Development Systems

The MicroPDP-11 family has been able to take advantage of the broad base of PDP-11 software. The MicroPDP-11 provides a very easy and direct migration path from other Digital products by supporting existing PDP-11 software. The long list of PDP-11 system software allows the MicroPDP-11 to address tasks found in realtime, multiuser timesharing, and batch environments. The MicroPDP-11 is supported by eleven operating systems, of which two were developed exclusively for the MicroPDP-11 family. All of these make the MicroPDP-11 suitable for both development and applications environments. The key attributes of these systems are summarized in Table 4-1.

RSX-11 Family of Operating Systems

RSX-11 operating systems are designed to execute multiple programs concurrently. A program is allowed to execute (use the CPU as a resource) until its immediate need for the CPU is completed or until an external event with a higher-priority program takes its place. If the higher-priority program needs memory space, the lower-priority program is swapped out to a disk. This ability to respond rapidly to external events makes the RSX-11 family of operating systems an ideal choice where realtime reaction is important, as in industrial process control or data acquisition. When tasks of equal priority are eligible to execute, a round-robin scheduler rotates their selection so that all receive an equal share of CPU time.

RSX-11 operating systems are not limited to realtime tasks. They are designed to provide login/logout procedures with passwords, access protection, user accounting, and other capabilities to support multiple users developing programs on the system. Communication with the operating system is made through the easy-to-use Digital Command Language (DCL) or with the traditional RSX-11 Monitor Console Routine (MCR) command language.

When doing program development under Micro/RSX, RSX-11M-PLUS or RSX-11M, a broad range of Digital high-level languages, software tools, and applications programs is available. The integral RMS-11 file system maintains files in a sequential, random, relative, or multikeyed ISAM organization.

Micro/RSX, RSX-11M-PLUS, and RSX-11M offer unsurpassed software compatibility. All nonprivileged tasks that run on RSX-11M can run on RSX-11M-PLUS and Micro/RSX without change or reassembly. Privileged tasks usually require little or no change. Micro/RSX and RSX-11M-PLUS also provide significant support for migrating applications to VMS environments.

There are four members of the RSX-11 family—Micro/RSX, RSX-11M-PLUS, RSX-11M, and RSX-11S. The following section describes each in more detail and makes comparisons of their features.

▪ MICRO/RSX

Micro/RSX is an extended subset of the multiuser, multitasking RSX-11M-PLUS operating system. As the newest member of the RSX-11 family, Micro/RSX was designed for use with the MicroPDP-11 and is a customer-installable, easy-to-use system that can support up to 12 users in both realtime and timesharing environments.

Micro/RSX is offered in two packages. The *Base Kit* provides the full RSX-11M-PLUS executive, appropriate utilities and device drivers, support for user-mode program development in high-level languages, and a user documentation kit. The *Advanced Programmer's Kit* is an add-on to the Base Kit and includes the software and documentation necessary for MACRO-11 and privileged program development. This includes a MACRO assembler, a librarian, an online debugging tool, and system libraries specifically designed to support privileged programming. The Advanced Programmer's Kit also include the *data terminal emulator* and *file transfer utility* which allows for easy file transfer between a Micro/RSX system and any other RSX, VMS, or P/OS system. Communications between any of these systems is established through a terminal line.

Micro/RSX also has Professional 350 diskette-exchange capability. With the use of the Professional Tool Kit, it allows programs to be developed for use on the Professional 350. Like RSX-11M-PLUS, Micro/RSX also provides a migration path directly to VMS.

▪ RSX-11M-PLUS

RSX-11M-PLUS provides the optimal multiuser system software for Digital's newest processors in the PDP-11 family. As the superset member of the RSX-11 family of operating systems, RSX-11M-PLUS offers all of this family's capabilities.

RSX-11M-PLUS takes advantage of the expanded addressing capabilities of Digital's newest PDP-11 processors while retaining the superior reliability and the successful architecture of RSX-11M. RSX-11M-PLUS uses hardware features in these PDP-11 processors that are not available in other members of the PDP-11 family. With the use of supervisor-mode library routines and separate user-mode instruction and data space, an RSX-11M-PLUS task can address up to 196

Kbytes of memory. In addition, RSX-11M-PLUS supports multistream batch, system accounting, dynamic dual-ported disks, addition memory management capabilities, and more simultaneous tasks and terminals than RSX-11M.

- RSX-11M
  The RSX-11M operating system is the original member of the RSX-11 family. It offers a large portion of the capabilities contained in RSX-11M-PLUS. RSX-11M excels in performance on small- and medium-size PDP-11 systems. It is designed to support factory automation, laboratory data acquisition and control, graphics, process monitoring, process control, communications, and other applications that demand immediate response. Its multiprogramming capabilities permit realtime activities to execute concurrently with such activities as program development, text editing, and data management.

- RSX-11S
  RSX-11S is a memory-resident subset of RSX-11M. As a result, a file system is not supported on RSX-11S. RSX-11S is used as an extremely efficient execute-only system. RSX-11S is generally used under conditions on which a disk cannot safely operate, such as on the floor of a manufacturing plant.

  RSX-11S provides excellent online process control. Because all programs are memory-resident, response is extremely fast. Tasks for an RSX-11S system are developed on computers running the RSX-11M, RSX-11M-PLUS, or VAX/VMS operating system. Such tasks are then loaded into the RSX-11S system image by using a supplied host utility, or the RSX-11S Online Task Loader (OTL), or by downline loading if both the host and the RSX-11S system have DECnet or DECdataway support.

RSTS Family of Operating Systems

The RSTS family is one of Digital's most popular, multiuser timesharing systems for general use. Since it was introduced, over 350 man-years of effort have contributed to its development into a versatile and easy-to-use operating system.

The RSTS family comprises two multiuser, timesharing operating systems—RSTS/E and Micro/RSTS. Both operating systems are compatible with a wide range of programming languages, information management tools, and applications.

The RSTS family's multiprogramming environment allows concurrent data processing, word processing, program development, and batch-mode processing (RSTS/E only). Highly developed system security protects both data and system resources.

The system programmer using RSTS can benefit from its interactive text editors, program debuggers, trace facilities, dump analyzers, and high-level forms languages. The system manager can oversee every aspect of daily operation on a RSTS system. In addition, RSTS allows the inexperienced user to perform word processing, access databases with only a minimum of training, and implement special applications.

▪ RSTS/E

RSTS/E is a multiuser, general-purpose timesharing system that can implement up to 63 jobs and up to 127 terminals. Each one of the multiple users can count on an almost immediate response to requests for access to programs, utilities and data, and transactions in process.

The user is associated with a job and interacts with that job through a terminal. A timesharing job scheduler allows the job to execute until its immediate need for the CPU is completed, or until an allocated period of time expires. The eligible job with the highest priority will then be executed. If several eligible jobs have the same priority, a round-robin scheduler rotates their selection so each gets an equal share of CPU time. For example, a batch job can also be submitted to run at a future time after working hours and the batch processor job will supervise its execution in the submitter's absence.

Each individual's files and file directory can be protected from unauthorized access by other users. Each user can specify who can read a file and who is allowed to modify or update it.

Communications with the operating system are accomplished through the easy-to-use Digital Command Language (DCL). Individual installations can add their own commands with the Concise Command Language (CCL). And with optional MENU-11, customized menus can be built as a command interface for novice or infrequent users. BASIC-PLUS is also included with the RSTS/E operating system.

Because RSTS/E can migrate from one PDP-11 processor to another, system growth is easy. In addition, distributed processing networks can be built using DECnet/E for Digital-only networks and Internets for connection to other vendor's systems.

▪ MICRO/RSTS

Micro/RSTS is a pregenerated subset of RSTS/E and supports all RSTS/E system calls. Micro/RSTS was designed primarily for use with the MicroPDP-11 and is a customer-installable, easy-to-use system that can support up to 20 jobs on the MicroPDP-11/73 and up to 10 jobs on the MicroPDP-11/23. It can also support up to 14 terminals in a timesharing environment.

Micro/RSTS consists of two separate kits. The *Base System Kit* is required for all users, and the *Application Development Kit* is optional. The Base System Kit includes software and documentation for the Micro/RSTS runtime system, for device support, and for BASIC-PLUS program development. BASIC-PLUS is included with the kit. The Application Development Kit includes software and documentation to support native MACRO-11 and high-level language program development. MACRO-11 is included with the Application Development Kit.

Micro/RSTS supports programs developed on any RSTS/E system. Host development requires that files be transferred to and from the Micro/RSTS system. This can be done by using any transfer medium that is available on both systems.

MicroPower/Pascal Development Tool Kit

MicroPower/Pascal is an advanced software tool kit for developing Q-bus-based microcomputer applications. It includes a high-performance PASCAL compiler, a modular executive, and a variety of tools to create concurrent, realtime applications programs.

▪ MICROPOWER/PASCAL FEATURES

MicroPower/Pascal has two system environments to accomplish this development. The *host system* creates and builds the software. The *target system* executes the software. Each application is custom-designed for its target system and includes the appropriate set of operating system services. The host, using the symbolic debugger, controls the execution of the target application during development.

There are four MicroPower/Pascal products. MicroPower/Pascal-RT, MicroPower/Pascal-Micro/RSX, and MicroPower/Pascal-RSX develop applications using a PDP-11 host system. MicroPower/Pascal-VMS develops applications using the VAX family.

The host development environment for each of these products includes an extended, realtime PASCAL compiler, a symbolic debugger, several build utilities, and a MACRO-11 interface. The target environment includes a library of software modules for process synchronization, communications, scheduling, exception and interrupt handling, timer services, and device and file I/O.

The application program is created and linked with the appropriate runtime software in the host system. It is then transported to the target system by one of three methods—writing it into read-only memory, downline loading it over a serial line, or recording it onto removable storage media such as a floppy disk or tape cartridge and then bootstrapping it on the target system.

MicroPower/Pascal is very compact and can reside in as little as 8 Kbytes of memory for small application programs. For complex applications, MicroPower/Pascal can address as much as 4 Mbytes of memory.

ULTRIX-11 Operating System

Digital's enhanced native-mode UNIX system software, ULTRIX-11, provides a flexible, interactive programming environment for multiple users.

▪ ULTRIX-11 FEATURES

ULTRIX-11 includes all of the features found in Version 7, such as the Bourne shell, C shell, shell scripts, pipes, C compiler, and Assembler. In addition, it includes a hierarchical file system that can provide a directory of files. Subdirectories can also be created to manage groups of similar files. Input/output can be performed by reading or writing into a special file that is associated with an I/O device. This makes file and device I/O similar for ease of programming. It also allows a program to accept either a file or device without changes, and extends the file protection mechanism to the I/O. The creator of a file can permit or deny read, write, and access protection to other users.

Digital has written several enhancements for ULTRIX-11 to provide better performance and maintainability. These enhancements include:

- Improved fault tolerance
- Disk bad-block replacement
- Automated installation and system generation
- System tuning
- Processor and peripheral device support
- VI full-screen editor
- Overlay kernel for CPUs with combined instruction and data space
- Special files
- File-system table
- Crash-dump analyzer
- System-management commands
- C shell
- Kernel floating-point simulator
- TIP remote login and file transfer
- Source code control system with job control
- Certain System V features
- Certain University of California at Berkeley Version 2.9 features

The ULTRIX-11 operating system is interfaced through the Bourne or C shell command-line interpreter. Shell commands create processes that can communicate through pipes, create subsidiary processes, and synchronize the offspring processes.

ULTRIX-11 languages include:

- C—a high-level language conducive to structured programming
- FORTRAN-77 and RATFOR—adds a C-type control structure to FORTRAN
- BASIC-like interpretive language
- Programmable desk calculator

Software tools include a compiler writing system, document preparation programs, information handling routines, and graphics support.

The customer can easily install ULTRIX-11 and also run the System Exerciser Package to verify that it is functioning properly.

RT-11 and CTS-300 Operating Systems

RT-11 is an operating system for realtime, single-user applications. It is well suited for such applications as laboratory and factory instrument control, manufacturing process control, flight management, mapping, and numerous other technical jobs. CTS-300 is a complete, multiuser software environment layered on top of RT-11. It is designed to support commercial applications. CTS-300 includes DIBOL, a programming language designed for writing business applications.

- RT-11 FEATURES

RT-11 uses the Digital Command Language (DCL). This makes access to operating system services as easy as typing English-like commands. Instead of having to manage system calls directly, services can be called through DCL commands that will prompt for any missing parameters and will offer help if a problem or question arises.

The keypad editor, KED/KEX, is specially designed for a wide range of video terminals and takes advantage of all their advanced features. Screen-oriented editing immediately points out any editing problems and makes quick changes to correct errors or to accommodate altered program needs.

RT-11 systems offer a choice of three different operating-system monitors to accommodate a range of RT-11 users. Digital supplies the system with a single-job monitor, a foreground/background monitor, and an extended-memory monitor. A single-job monitor, called SJ, organizes the system for single-user, single-program conditions. The foreground/background monitor, called FB, takes advantage of the fact that much central processor time is spent waiting for external events such as I/O transfer or realtime interrupts. In the FB monitor, this waiting time is put to good use by allowing the CPU to be used for other jobs while the principal (foreground) job is pausing. The extended-memory monitor, XM, allows both foreground and background jobs to extend their effective logical program space beyond the 64-Kbyte space imposed by

16-bit addresses on PDP-11 computers. The XM monitor contains all the features of FB plus the capability of accessing up to 4 Mbytes of memory.

There are three communications utilities that come with RT-11. VTCOM (Virtual Terminal Communications) allows RT-11 to connect to any host system via a serial line and transfer ASCII files between the two systems. TRANSF (Binary File Transfer) uses VTCOM and allows binary files to be transferred between RT-11-based (RT-11, CTS-300, RTEM-11) systems via a serial line. Ethernet Handlers for DEQNA allow users to write their own software to communicate over Ethernet hardware.

RT-11 provides even more tasks to make the system accessible to both novice and experienced users alike. RT-11 offers an automatic-installation procedure that installs the operating system simply by conducting an interactive dialog with the user.

Programs can be written without explicitly identifying the output device. For example, the device selection can be deferred until the program is run so that printer output can be directed to the disk. When a new device is added to the system, any old programs can be adapted easily.

RT-11 programs can also be developed as one of the tasks on an RSX-11 system using RTEM-11. Programs developed with RTEM-11 can execute on appropriately configured RT-11 systems in the same manner as if they had been developed on RT-11. Most programs developed on RTEM-11 can be debugged and tested on RTEM-11. The execution environment supplied with RTEM-11 is foreground/background (FB).

Although RT-11 supports only one command terminal, multiterminal support capability allows programs to control up to 16 additional terminals.

▪ CTS-300 FEATURES

CTS-300 is designed to support commercial applications. It consists of the RT-11 operating system, described before, plus DIBOL (Digital's Business-Oriented Language), and a number of utilities. Most RT-11-dependent software products can also be run on CTS-300.

CTS-300, like RT-11, is a single-user system in the sense that there can be only one system command terminal. However, multiple terminals running multiple DIBOL jobs or developing multiple DIBOL programs are supported under the three DIBOL runtime systems—single-user DIBOL (SUD), timeshared DIBOL (TSD), and extended-memory timeshared DIBOL (XMTSD).

DIBOL is an easy-to-learn and easy-to-use language that allows commercial applications to be developed in minimal time. DIBOL has a Data Division and a Procedure Division, like COBOL, and provides the ability to manipulate data, evaluate arithmetic expressions, redefine records, call other programs, spool output, and access files.

Utilities included with CTS-300 include:

- DECform—defines video screen formats, checks entered data for range and type, and totals and validates entered fields. It also supports additions, inquiries, changes, and verifications to DMS-300 files.
- DMS-300—a data management utility that supports sequential access, random access, and keyed access to ISAM files.
- SORT/MERGE—a data management utility that permits users to easily define the parameters for sorting and merging data files.
- Line Printer Spooler Utility—queues and manages files for printed output.

DSM (Digital Standard MUMPS)

DSM is a complete, multiuser system environment with data management capability and the interactive, high-level language MUMPS. With DSM, programs can be quickly written, tested, debugged, or modified to establish a working application.

The MUMPS language, originally developed at Massachusetts General Hospital, has syntax and semantics oriented toward solving database-related applications. A novice programmer can very quickly produce useful working code, although using the full range of MUMPS capabilities does require some programming experience.

MUMPS' text-handling capabilities allow the inspection of any data item for content (such as particular keywords) or for format (letters, numbers, or punctuation characters in a string of text). The capabilities are useful for online data-entry checking and correction.

The DSM hierarchical file structure allows data files to be designed to suit the needs of a particular environment. Dynamic file storage simplifies expansion or modification of the database. The database handler maintains an in-memory cache of disk data for high-performance data access and data sharing.

DSM implements an extension of the 1983 ANSI Standard MUMPS language. DSM allows a MUMPS application to define independent error handlers for each execution level. A MUMPS debugger allows the DSM programmer to set or clear breakpoints, single-step through MUMPS commands, and trace program execution.

▪ DSM-11 FEATURES

DSM-11 is a complete multiuser operating system layered on the MicroPDP-11 systems that includes the MUMPS language interpreter and a powerful, high-performance data management capability. DSM-11 supports the following features on the MicroPDP-11:

- Mountable volume sets
- Inter-job communications
- Memory-resident applications
- Magnetic-tape streaming
- IBM binary synchronous communications
- Autoconfiguration
- Unattended backup
- System-level, transparent journal of database modifications can be maintained on either disk or magnetic tape.
- Output to devices (such as a printer) can be spooled.
- Bad-block management for all disk media
- Online, high-speed database backup, disk-media preparation, and tape-to-tape copying
- Hardware device-error reporting, system patching utility, and an executive debugger for system maintenance
- System installation and generation procedures

▪ VAX DSM FEATURES

VAX DSM is an implementation of Digital Standard MUMPS layered on VMS and MicroVMS. VAX DSM includes the MUMPS language and a high-performance database handling capability. Additional VAX DSM features include:

- Use of VMS facilities (batch, spooling, backup)
- Access to VMS I/O capabilities (sequential, relative, and indexed files, mailboxes, DECnet)
- External call interface to nonMUMPS procedures ($ZCALL function)
- Transparent journal of database modifications
- Mapping of DSM routines in virtual memory

Table 4-1 ▪ Operating-system Summary

Feature	Micro VMS	ULTRIX -32m	Micro/ RSX	RSX-11M-PLUS	RSX-11M	RSX-11S	Micro/ RSTS	RSTS/E	ULTRIX -11	RT-11	CTS-300*	DSM
User Interface												
Shell		X							X			
DCL	X		X	X	X	X	X	X		X	X	
MCR				X	X	X						
CCL							X	X		X	X	
User-written			X	X	X					X	X	
Text Editors												
Keypad	X		X	X	X		X	X		X	X	
Line		X	X	X	X		X	X	X	X	X	X
Screen		X	X	X	X			X	X	X	X	
Batch Processing	X		X	X				X		X	X	
File Management												
Multikey ISAM			X	X	X		X	X				
Single-key ISAM			X	X	X		X	X			X	
Sequential		X	X	X	X		X	X	X	X		
Relative			X	X	X		X	X		X		
Random			X	X	X		X	X		X	X	

*Includes RT-11, DIBOL-83, and DECform.

▪ High-level Languages

High-level languages take the words and symbols entered by the programmer and translate them into a series of binary codes that the computer can understand. When the computer is ready to output information, high-level languages take the binary codes and return them to a format that can be read and understood by a person.

Different applications may require different methods of programming. This accounts for the growth of many different programming languages. Digital offers a variety of industry-standard languages that solve these programming problems.

Table 4-2 summarizes the languages supported by each operating system.

VAX Ada*

VAX Ada is Digital's implementation of the government-validated compiler called Ada* for VAX systems. VAX Ada is suitable for systems, computation, general purposes, and in particular for realtime applications and multitasking applications. Aside from government applications, VAX Ada is also strong in the industrial and educational areas as well. VAX Ada fully conforms to the ANSI-83 Ada standard. VAX Ada is available on MicroVAX I under MicroVMS.

VAX APL

VAX APL (A Programming Language) is a concise programming language that simplifies the handling of numeric and character data organized as lists and tables. VAX APL is a native-mode, shareable, reentrant interpreter that is available on the MicroVAX I under the MicroVMS operating system. It provides a built-in function editor, debugging aids, system communication facilities, and a file system. VAX APL can execute lines of code immediately or store the code for later execution.

BASIC

Four versions of BASIC are available for the microsystems. All four versions use simple English commands, understandable abbreviations, and familiar symbols for mathematical and logical operations. They are readily accessible to programmers who are not computer specialists. They are used extensively in education, small businesses, laboratories, and for personal use.

VAX BASIC is an interactive, shareable language processor for the MicroVMS operating system. VAX BASIC takes full advantage of the VAX-11 floating point, decimal, and character instructions.

BASIC-PLUS-2 is an extended BASIC compiler that runs under the Micro/RSX, RSX-11M-PLUS, RSX-11M, Micro/RSTS, and RSTS/E operating systems. It takes full advantage of the PDP-11 floating-point and integer instruction sets.

*Ada is a registered trademark of the U.S. government.

BASIC-PLUS and BASIC-11 are conversational programming languages developed at Dartmouth College that use simple English-like statements and familiar mathematical notations to perform operations. BASIC-PLUS is an integral part of the RSTS/E operating system. BASIC-11 is optional for the RT-11 and CTS-300 operating systems.

VAX BLISS

VAX BLISS is a high-level systems implementation language for VAX systems only. VAX BLISS supports development of modular software according to structured programming concepts by providing an advanced set of language features. VAX BLISS provides access to most of the hardware features of the VAX to facilitate programming of realtime and/or hardware-dependent applications. It is especially intended for the development of operating systems, compilers, runtime system components, database file systems, communications software, and utilities. VAX BLISS is available on MicroVAX I under MicroVMS.

C

C is a concise, expressive, structured programming language designed by AT&T Bell Laboratories for program development on UNIX systems. It is included with ULTRIX-11 for the MicroPDP-11 family. VAX C is an extended implementation of the C programming language developed by AT&T Bell Laboratories and is available as an option on MicroVMS.

COBOL

COBOL is an industry-standard, data-processing language used extensively for business applications because of its orientation toward character, string, and file processing. Digital provides two versions of COBOL that are based on the 1974 ANSI COBOL standard and implements many items from the proposed ANSI standard.

COBOL-81 runs on the MicroPDP-11 family under Micro/RSX, RSX-11M-PLUS, RSX-11M, Micro/RSTS, and RSTS/E. COBOL-81 is a subset of VAX COBOL. Performance can be improved by using COBOL-81 with the optional KEF11-BB Commercial Instruction Set option.

VAX COBOL is a fully featured COBOL compiler that meets the highest level of federal requirements. It runs on the MicroVAX I under MicroVMS. Because COBOL-81 is a subset of VAX COBOL, COBOL-81 programs can, in most cases, be compiled and executed on a MicroVAX or VAX without source-code changes.

CORAL-66

CORAL-66 is a block-structured language developed by the British government for realtime and process-control applications. The language is designed to replace assembly-level programming in modern industrial and commercial applications. It is used for long-life products where ease of maintenance and flexibility are required. CORAL-66 is available only on MicroPDP-11 systems with RSX-11M-PLUS, RSX-11M, or RSX-11S.

DIBOL-83

DIBOL-83 (Digital's Business-Oriented Language) is a high-level procedural language designed specifically for interactive business data processing. DIBOL-83 is based on the DIBOL Standards Organization definition.

It is represented in two segments: a Data Division and a Procedure Division. The Data Division defines the data that is used by the program. The Procedure Division contains the executable statements. DIBOL-83 is available on MicroPDP-11 systems under Micro/RSX, RSX-11M-PLUS, RSTS/E, and as part of CTS-300. It is also available on the MicroVAX I under MicroVMS.

FORTRAN

FORTRAN is the most widely used programming language for developing programs dealing with scientific applications. Three FORTRAN compilers are available for the microsystems:

- VAX FORTRAN for MicroVMS
- PDP-11 FORTRAN-77 for Micro/RSX, RSX-11M-PLUS, RSX-11M, RSX-11S, Micro/RSTS, and RSTS/E systems. FORTRAN-77 is available for RT-11 from the Digital Classified Software Library.
- FORTRAN IV for RSX-11M-PLUS, RSX-11M, RSTS/E, and RT-11 systems
- VAX FORTRAN conforms at the full-language level to the ANSI FORTRAN X.39 1978 standard and is upward compatible from PDP-11 FORTRAN-77.

PDP-11 FORTRAN-77 combines the efficient numeric computation for which FORTRAN is known with access to sequential, relative, and indexed files. This makes PDP-11 FORTRAN-77 ideal for writing software that must manipulate and perform calculations on structures of numeric data, as in accounting or statistical packages. PDP-11 FORTRAN-77 is built on an ANSI subset of the ANSI FORTRAN 1978 standard.

FORTRAN IV is a fast, one-pass, optimizing compiler that implements an extended superset of the 1966 ANSI standard for FORTRAN. FORTRAN IV works efficiently in small-memory environments and is capable of producing absolute binary code for standalone MicroPDP-11 systems or for loading into ROM or PROM memory.

MUMPS

MUMPS is a language oriented toward database applications. It is an integral part of DSM and is described in the DSM section.

PASCAL

PASCAL is a block-structured language that contains control statements, data types, and predeclared procedures and functions. There are two versions of PASCAL for the microsystems: PDP-11 PASCAL/RSX and VAX PASCAL.

PDP-11 PASCAL/RSX provides all standard PASCAL features as well as extensions that are designed to improve the productivity of the PASCAL programmer. These extensions make it simple to divide programs into easily managed problem sections and to enhance the computing power of the language. PDP-11 PASCAL/RSX is available as an option on Micro/RSX, RSX-11M-PLUS, and RSX-11M.

VAX PASCAL generates optimized, shareable code that takes full advantage of the VAX hardware floating point and character instruction sets and the virtual memory capabilities of MicroVMS. VAX PASCAL is available on MicroVAX I under MicroVMS.

VAX PL/I

VAX PL/I is a block-structured, comprehensive programming language that supports scientific computation, commercial data handling and organization, and extensive string manipulation. VAX PL/I is available on MicroVAX I under MicroVMS.

VAX RPG II

VAX RPG II is an extended implementation of the RPG II language developed by IBM as a problem-oriented language for commercial applications. VAX RPG II includes extensions for integration with the VAX architecture. It provides a convenient means of preparing a wide variety of reports and other commercial data processing applications. VAX RPG II is available on MicroVAX I under MicroVMS.

Table 4-2 ▪ High-level Language Availability

Language	Micro VMS	ULTRIX -32m	Micro/ RSX	RSX- 11M- PLUS	RSX- 11M	RSX- 11S	Micro/ RSTS	RSTS/E	ULTRIX -11	RT-11	CTS- 300	DSM
VAX Ada	X											
VAX APL	X											
BASIC	X		X	X	X		X	X		X	X	
C	X								X			
COBOL	X		X	X	X		X	X				
CORAL-66				X	X	X						
DIBOL-83	X		X	X				X			X	
FORTRAN	X		X	X	X	X	X	X		X		
MUMPS	X											X
PASCAL	X		X	X	X							
VAX PL/I	X											
VAX RPG II	X											

▪ Information Management

Information management software ensures users that they have an integrated system of data management capabilities to help them organize their data. Information management products help users do more for themselves and give programmers more time to plan and develop new applications. This section briefly describes each of the information management products that are available on each of the microsystems.

VAX CDD (Common Data Dictionary)

The Common Data Dictionary provides a single, logical data dictionary for the MicroVAX I under MicroVMS. This dictionary is built from one or more physical dictionary files. Within this dictionary, the CDD maintains a hierarchical directory of the user's data descriptions. CDD contains definitions for VAX DATATRIEVE, VAX TDMS, VAX COBOL, VAX BASIC, VAX DIBOL, and VAX PL/I. VAX CDD is prerequisite software for these products and must be installed prior to their installation.

DATATRIEVE

DATATRIEVE is an interactive, query report generation and data maintenance system designed for the less experienced computer user. DATATRIEVE provides facilities for selective data retrieval, sorting, formatting, updating, and report generation without the need for programming. VAX DATATRIEVE is available on MicroVAX I running MicroVMS. PDP-11 DATATRIEVE is also available on MicroPDP-11 systems running Micro/RSX, RSX-11M-PLUS, RSX-11M, and RSTS/E.

VAX Rdb/ELN

VAX Rdb/ELN is a relational database management system designed for dedicated applications on systems running in the VAXELN application environment. VAX Rdb/ELN applications are developed using the VAXELN Development Tool Kit on a host VAX/VMS system. The resulting bootable, VAXELN-based Rdb/ELN application is then moved to the VAXELN target system using disk media or an Ethernet local area network link. The application program executes on the target system as a dedicated database system. The network link to the host development system may be used for remote debugging. VAX Rdb/ELN is available on MicroVAX I running MicroVMS.

RMS (Record Management System)

RMS is a straightforward method of creating, updating, and modifying files using sequential, relative, or multikey indexed access methods. RMS is an integral part of Micro/RSX, RSX-11M-PLUS, RSX-11M, Micro/RSTS, and RSTS/E on MicroPDP-11 systems. It is also an integral part of MicroVMS on MicroVAX I.

VAX TDMS (Terminal Data Management System)

VAX TDMS is a product designed for the implementation of interactive, forms-intensive applications running on MicroVAX I under MicroVMS. As a terminal subsystem, VAX TDMS can reduce applications development and maintenance effort. VAX TDMS replaces applications program logic specific to terminal interactions with definitions that are external to the program.

▪ Productivity Tools

Productivity tools allow program developers to simplify their programming and information-management processes. By eliminating extensive maintenance and documentation tasks, more time is left for creating new applications or changing existing ones. This section briefly describes each of the productivity tools that are available on each of the microsystems.

VAX DEC/CMS (Code Management System)

VAX DEC/CMS is a program library for use as an aid in program organization, development, and maintenance. It is a tool that allows programmers to manipulate information relating to their software project. Each CMS command invokes a certain function, such as reserving a file for modification or obtaining a report listing development status. VAX DEC/CMS is available on MicroVAX I running under MicroVMS.

DECmail-11

DECmail-11 is an electronic-message system that can create, edit, send, and process messages, as well as store, search for, and retrieve messages held in user folders. DECmail-11 is command-driven and includes an online help facility that provides the user with information that covers most normal situations. DECmail-11 can be used in a network environment. The Message Router is an optional product for the network environment and provides a store-and-forward transport mechanism between RSTS/E, RSX-11M-PLUS, and VAX/VMS nodes in a network. It can also communicate with DECmail/VAX and with the All-In-One office software system. DECmail-11 is available on the MicroPDP-11 systems under Micro/RSX, RSX-11M-PLUS, Micro/RSTS, and RSTS/E.

VAX DEC/MMS (Module Management System)

VAX DEC/MMS is a software tool that automates and simplifies the building of software systems. It can determine what components in a described software system have changed and rebuild the system according to these changes. When some modules of a software system are modified, dependent modules may need to be recompiled. VAX DEC/MMS determines which modules need to be recompiled, and performs the appropriate actions to ensure that the software system is recompiled and linked with all the latest changes. VAX DEC/MMS is available on the MicroVAX I under MicroVMS.

VAX DECslide

VAX DECslide is a menu-driven graphic representation utility that creates slides. A combination of symbols and text is used in the menu selection. VAX DECslide uses an interactive interface so that diagrams and text are displayed as they are entered. Editing functions allow changes to slides as they are created or after they have been saved. A text menu and message area at the bottom of the screen displays user options, help messages, and operation status. A directory feature lists the slides, including the date and time of creation, slide name, and comments. After slides are created, they can be colored with available color palettes. They can be printed in a single-size or double-size format, saved, copied, exported, changed, or deleted. VAX DECslide is available on MicroVAX I under MicroVMS.

VAX DEC/Test Manager

VAX DEC/Test Manager is an automated regression testing system. It allows the programmer to organize tests, selected tests for execution, and verify and review the test results. With VAX DEC/Test Manager, one can describe a set of tests, classify the tests by assigning them to groups, and choose the combination of tests and/or groups to be run. The selected tests are executed and the results compared with the known correct output. During the execution of a test, VAX DEC/Test Manager provides a summary of a test's status. It also allows the programmer to view the test results interactively, evaluate the test run, and use the results to make modifications to the code being tested. VAX DEC/Test Manager is available on MicroVAX I under MicroVMS.

VAX DEC/Shell

VAX DEC/Shell is a command language that provides a means of communicating with the VMS and MicroVMS operating systems. It is similar to the user interface on a UNIX Version 7 system. It appears to its users as the Bourne Shell and enables them to perform many of the same tasks done on a UNIX Version 7 system.

VAX DEC/Shell consists of two parts—the command interface and the Shell programming language. It also provides many of the most common UNIX utilities and commands and treats most files as stream files, a type of organization that is familiar to UNIX users.

FMS (Form Management System)

FMS is designed to aid in the development of application programs that use video forms. FMS manages the forms for application programs that use the VT100 and VT200-compatible terminals. FMS provides the user with the character attributes of reverse video, bold face, blinking, and underline. It provides line attributes of double width, double height, and scrolling. Screenwide attributes such as 80 or 132 column lines and reverse video are included. Alternate character sets including the VT100 special graphics character set for line drawing are supported. FMS is available for MicroPDP-11 systems running under RSX-11M-PLUS, RSX-11M, RSTS/E, and RT-11. It also is available for MicroVAX I under MicroVMS.

INDENT

INDENT is a data entry and forms management product for commercial applications written in DIBOL, COBOL, or BASIC-PLUS-2. INDENT uses reverse video, bold face, underline, 132-column lines, scroll, split-screen, reverse screen, and the line-drawing character set. INDENT form definitions are created using a text editor. The resulting English-like source code is then compiled using the INDENT compiler and task builder. A forms driver is linked to the host application program during task building. After task building the form object module, the form task image is executed by the application program and the INDENT runtime system. Data from forms is returned to the application program when an entire form is completed or when an individual field is completed. INDENT is available for MicroPDP-11 systems under RSTS/E.

MENU-11

MENU-11 provides the RSTS/E applications developer with the ability to present a simple, menu-driven interface to the user on a video terminal for both system and application functions. MENU-11 accepts and executes commands related to the menu items, to perform specific functions, without extensive user training in the RSTS/E operating system. Three types of commands provide screen formatting, program execution, and security access. These commands allow the developer to format the menu screen for the user's terminal and to control the interfacing of the user with the system. MENU-11 is available on MicroPDP-11 systems under RSTS/E.

▪ Communications Software

After all of the necessary network hardware is set up, communications software makes the network functional within the Digital Network Architecture (DNA) framework. Digital's network software is broadly divided into three categories:

- DECnet—for communications among Digital systems
- Internet—for communications with other manufacturers' equipment
- Packetnet—for communications with other participants in public packet-switched networks

For more detailed information on these communications software areas, refer to Chapter 5—Networks. For descriptions of the software that is available in each of these areas, refer to the *Networks and Communications Buyer's Guide*.

▪ Additional Documentation

VAXELN Technical Summary	*EJ-30083-47*
VAX/VMS Technial Summary (includes MicroVMS)	*EJ-26070-48*
PDP-11 Software Handbook	*EB-25398-41*
RSX-11 Handbook	*EB-25742-41*
RSTS/E Handbook	*EJ-23534-18*
ULTRIX Software Guidebook	*EJ-26153-20*
Networks and Communications Buyer's Guide	*ED-26127-42*

Chapter 5 · Networks

Introduction

Digital offers extensive capabilities that permit the linking of computers and terminals into flexible configurations called networks. Networks increase the efficiency and cost-effectiveness of data processing operations.

Networking allows computer systems and terminals, whether located around a facility or around the world, to share resources and exchange information, files, and programs. The smaller computers in a network have access to the powerful capabilities of larger systems, while the larger computers can take advantage of smaller dedicated systems chosen for specific application environments.

Distributed processing is the general term used to describe the physical placement of computers where they are needed. As organizations become more complex and develop more sophisticated demands for computer resources, the ability to network processors and share computing resources becomes increasingly important.

Types of Systems

Two general types of systems can be implemented for most processing functions—the standalone system and systems connected by a network. With the standalone system, all data is entered manually at the system by an operator or from locally connected machines or instruments. In a network-connected system, information can be entered locally or transferred to or from other systems through an electrically connected network link.

Standalone Systems

A single-user standalone system, shown in Figure 5-1, can process information received from several sources. Data can be entered from the console terminal keyboard, read from a diskette in the disk drive, or received from an external machine or instrument. Only one user at a time can operate the system.

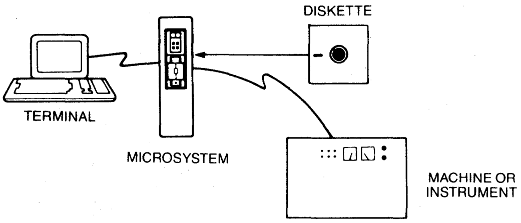


Figure 5-1 ▪ Single-user Standalone System

A multiuser standalone system, an expansion of the single-user system, allows several users to concurrently share a single processor. Several terminals can be connected to the processor and the processing is timeshared between users as shown in Figure 5-2. Data is entered from the same sources as the single-user system.

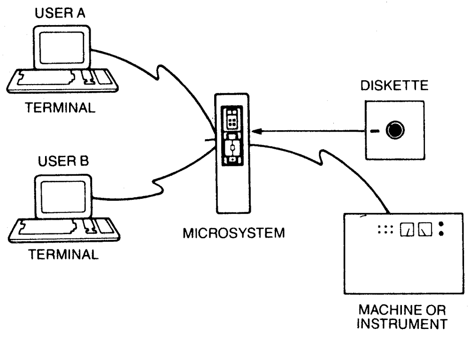


Figure 5-2 ▪ Multiuser Standalone System

For realtime or runtime applications requiring that information be shared between systems, there are several methods of communications.

As shown in Figure 5-3, Departments A and B of a small company both need sales figures, and Department B also needs the inventory-level information of Department A. Normally each department would be required to enter the sales figures from the terminal keyboard of each system. Department B would also be required to read the display of Department A and to enter the information manually into the Department B system from the keyboard.

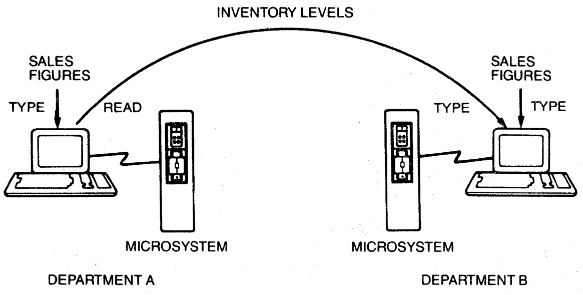


Figure 5-3 ▪ Manual Data Entry

A more efficient method of communications is shown in Figure 5-4. Department A manually enters the sales figures, processes the information, and records both the sales figures and inventory levels onto the diskette. No manual entry would be required by Department B. Once the diskette is received by Department B, the information can be processed.

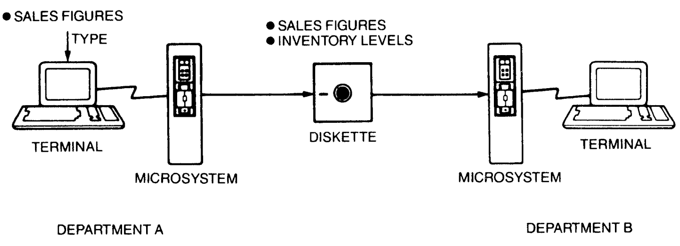


Figure 5-4 ▪ Recorded Data Entry

Network-connected Systems
Two or more standalone systems can be electronically connected together by a network. The network enables the efficient transfer of information between systems (shown in Figure 5-5).

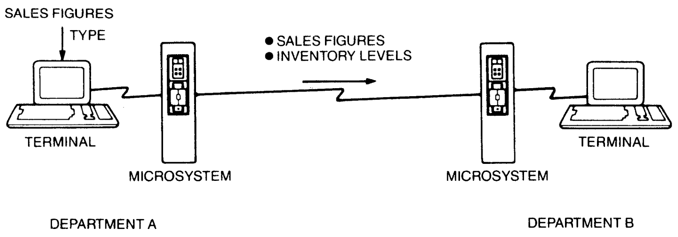


Figure 5-5 ▪ Two-system Network

Networks also enable systems located in different cities to communicate with each other as shown in Figure 5-6. Through the use of modems, information is transferred between offices in different locations over the standard telephone lines.

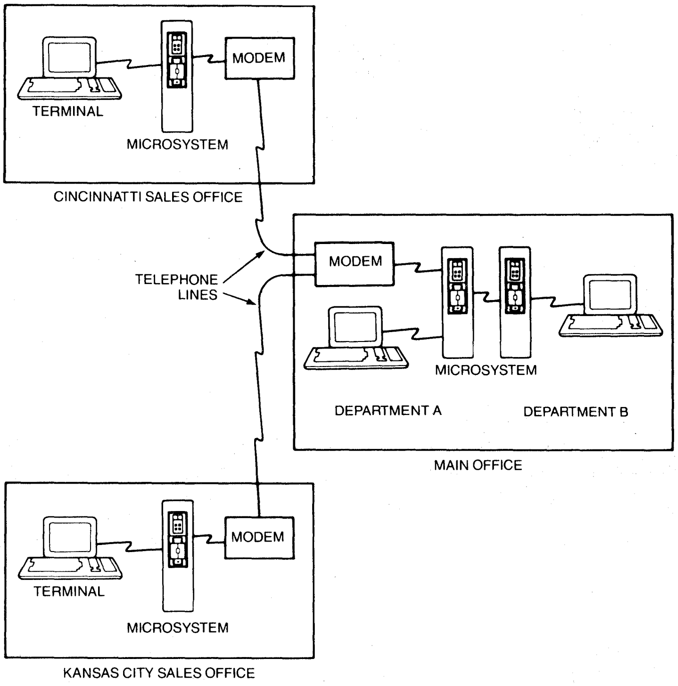


Figure 5-6 ▪ Remote Communications Network

▪ Digital Network Architecture

Digital Network Architecture (DNA) is a set of hardware and software networking capabilities that support communications between Digital's systems, and between Digital's systems and other manufacturers' systems.

Digital-to-Digital communications are permitted through protocols, or rules, that are defined by the DNA. DNA protocols are based on the architectural models for open systems interconnection created by the International Standards Organization (ISO). These rules govern the format, control, and sequencing of message exchange among Digital computers.

Internet products provide a means for Digital's systems to communicate with systems built by other manufacturers. These products emulate common communications protocols and are data transfer facilitators rather than hardware emulators.

DNA Structures

The lowest layer of the DNA structure, shown in Figure 5-7, is the *physical link layer*. This layer governs electrical and mechanical transport of information between systems that are connected. Computer systems can be physically connected by cables, fiber optic lines, microwave transmissions, or switched networks such as telephone lines. In addition to the physical connection, the electrical signals on the lines must be properly defined. The signal characteristics and data rates are all defined by the hardware comprising the interface module and the transmission link.

ISO SEVEN LAYERS	DNA LAYERS	DNA FUNCTIONS		
APPLICATION	USER	FILE TRANSFER REMOTE RESOURCE ACCESS DOWN LINE SYSTEM LOAD REMOTE COMMAND FILE SUBMISSION VIRTUAL TERMINALS		
	NETWORK MANAGEMENT			
PRESENTATION	NETWORK APPLICATION			
SESSION	SESSION CONTROL	TASK TO TASK		
TRANSPORT	END COMMUNICATIONS			
NETWORK	ROUTING	ADAPTIVE ROUTING		
DATA LINK	DATA LINK	DDCMP POINT TO POINT MULTIPOINT	X.25	ETHERNET
PHYSICAL	PHYSICAL LINK			

Figure 5-7 ▪ Digital Network Architecture Structure

The next highest layer of the DNA structure is the *data link layer*. The data link layer can prepare messages for transmission according to a specified protocol, check the integrity of received messages, and manage access to the channel. The data link is usually implemented by hardware and software. For a simple asynchronous interface, the hardware contribution to the data link is minimal. With other devices, such as the DEQNA Ethernet interface, almost all of the data link layer is implemented in hardware.

Because of the data link layer, the *routing layer* can rely on error-free connection to adjacent nodes. It addresses messages, routes them across intervening nodes, and controls the flow of messages between nodes. The routing layer and higher layers are implemented in software.

Because the routing layer establishes the path, the *end communications layer* can address the end machine without concern for route, and can perform end-to-end error recovery.

The *session control layer* manages the system-dependent aspects of a communications session. For instance, when the end communications layer reliably delivers a message from another manufacturer's system in the network, the session control layer interprets that message for acceptance by the system software.

The *network application layer* converts data for display on terminal screens and printers.

The *network management layer* monitors network operations by logging events and collecting statistical and error information. It also controls network operations by tuning network parameters and testing nodes, lines, modems, and interfaces.

The *user layer* provides services that directly support the user and application tasks such as resource sharing, file transfers, and remote file access.

▪ Typical Network Configurations

Systems can be interconnected in a network in many different configurations. Some of these configurations are described in this section.

Each of the participating systems in a network is called a *node*. Two nodes communicate through a link or connection. Figure 5-8 shows microsystem nodes linked in a network.

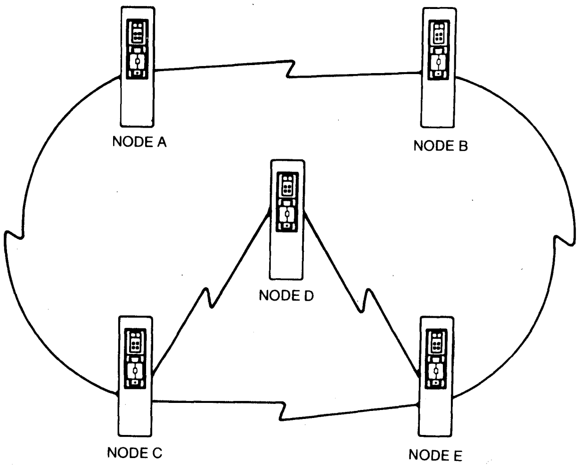


Figure 5-8 ▪ Nodes Linked in a Network

The physical links can be cables, fiber optic lines, microwave transmissions, and other conductors that form the data paths between two nodes such as Node A and B. Logical nodes exist whenever two nodes can communicate through a physical link or through another node such as between Node A and D.

Node C is required to route and transfer the message. This configuration increases the transmission time between nodes but decreases the cost because fewer physical links are required. As the number of physical links increase in a fully connected network, the resulting costs increase. Figure 5-9 shows the relationship of the number of nodes to the physical links.

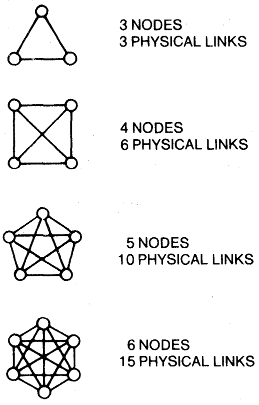


Figure 5-9 ▪ Fully Connected Network Nodes

One method of reducing the number of physical links is the multidrop or multipoint network shown in Figure 5-10. More than two nodes can be connected to the same physical link. Each node is required to determine which of the messages are dedicated to that node and to manage access to the links thus avoiding conflict between nodes.

In a network with centralized control, one node such as a mainframe computer is required to determine which of the remaining nodes can send messages, where the messages will be sent, and the length of the messages. In a network with distributed control, each node recognizes a procedure that allows the node to access the network independently.

In a star network, as shown in Figure 5-10, one node is designated as the central node and all other outlying nodes are physically connected to it. This network is efficient because most of the communications are between the central node and one outlying node, such as a timesharing network or a shared word processor system. Because all messages must be transferred through the central node, it must process many transactions when the message rate is high. If the central node fails, all transactions halt. In some networks, distributed control may be implemented, thereby decreasing the load on the central node.

In a ring network, each node is physically linked to two adjacent nodes, as shown in Figure 5-10. Messages circulate around the ring and each node retransmits the messages not addressed to itself. This method is less complex than the generalized routing method because each node is required to transmit the message only to the adjacent node.

Distributed control in a ring network may be implemented through token passing. A special token message circulates around the ring and a node can claim access to the network by receiving the token message as it passes through.

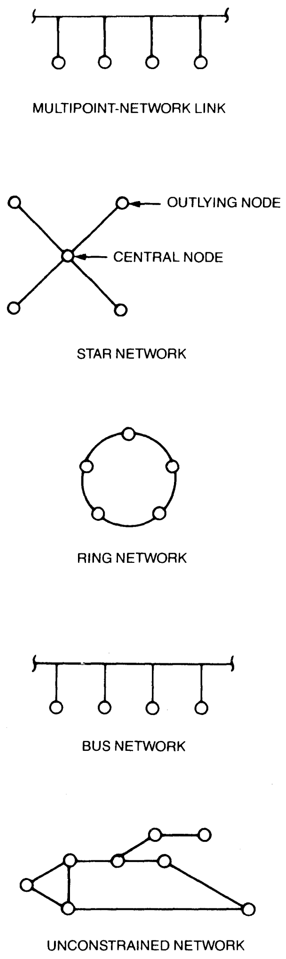


Figure 5-10 ▪ Network Configuration Types

The bus network, shown in Figure 5-10, is similar to a multidrop link. Messages placed on the shared physical link reach all nodes, and the intended receiver must recognize the message's address in order to receive the transmission. None of the nodes, however, have to route or retransmit messages intended for other nodes. A bus network typically uses distributed control. There is no single point of failure. The Ethernet local area network is a bus configuration.

An unconstrained network is shown in Figure 5-10. The placement of physical links is usually determined by the cost of the physical connections, by the number of messages to be transferred, and by the network reliability requirements. Some of the nodes in this network may have the capability to route messages to other nodes. Long-distance packet-switched networks are often unconstrained.

▪ Types of Links

Several interface options are provided for the microsystems to support the implementation of the physical and data link levels of the DNA.

Ethernet Link

As computer systems such as word processors, workstations, personal computers, and departmental systems become more numerous and accessible, an integrated communications network can increase productivity and reduce data processing costs.

Ethernet provides local area network technology through a hardware and software combination to create a physical communications channel between systems. This allows large amounts of data to be exchanged at high rates between systems located within limited distances.

Figure 5-11 shows the physical links and the DNA layers that perform the networking functions of Ethernet. The DEQNA is the interface module that provides the internal connection to the Q22 bus of the microsystems. The hardware elements include the DEQNA Ethernet interface, the transceiver cable, and the H4000 Ethernet transceiver and coaxial cable. The coaxial cable can be in lengths of up to 500 meters (1640.5 feet). The transmission rate can be up to 10 Mbits per second.

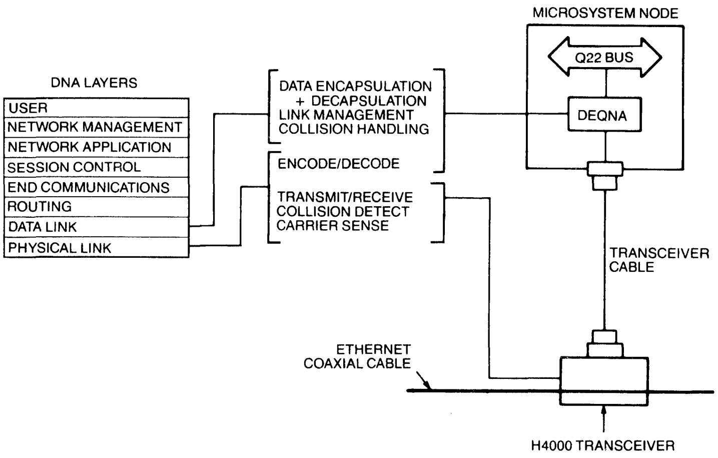


Figure 5-11 ▪ Ethernet Physical Link and Data Link Layers

Figure 5-12 shows a small-scale Ethernet configuration using a single coaxial cable. Each cable segment can include up to 100 transceivers or nodes.

A transceiver cable, which can be up to 50 meters (164 feet) in length, connects the H4000 Transceiver to the DEQNA Ethernet interface.

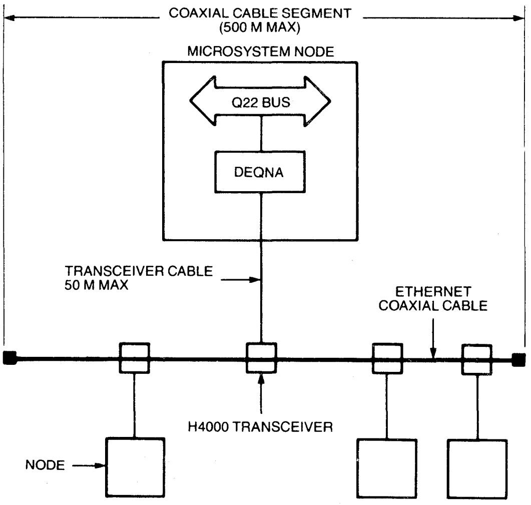


Figure 5-12 ▪ Small-scale Ethernet Configuration

A medium-scale Ethernet configuration is shown in Figure 5-13 and includes a repeater that connects two cable segments. Each coaxial cable can be a maximum of 500 meters (1640.5 feet) in length and can operate with up to 100 transceivers, including the repeater transceiver.

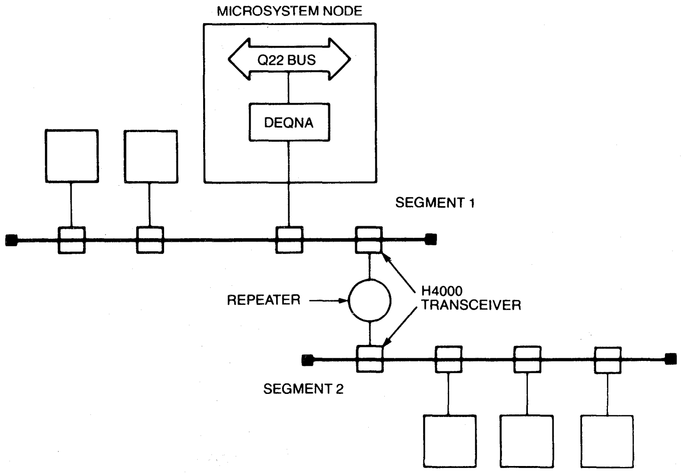


Figure 5-13 ▪ Medium-scale Ethernet Configuration

A large-scale Ethernet configuration is shown in Figure 5-14 and consists of five coaxial cable segments. The segments are connected by repeaters and can attach up to 1024 nodes. Each segment is connected by remote repeaters with up to a maximum distance of 1000 meters (3281 feet) between repeaters.

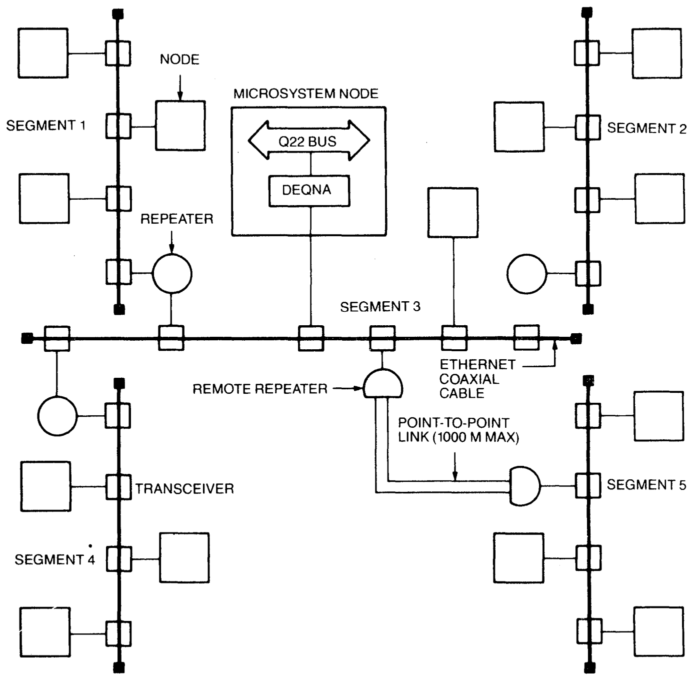


Figure 5-14 ▪ *Large-scale Ethernet Configuration*

The H4000 transceiver and tap can be installed on an operating cable with no disruption of the system operations. The transmit, receive, carrier sense, and collision-detection functions of the physical link layer are implemented in the H4000 transceiver.

The access control method used by Ethernet is called Carrier Sense—Multiple Access with Collision Detection (CSMA/CD). To transfer a message, a node monitors the transmissions on the cable to sense a pause between the data packets. The node continues to sense the data while initiating the transmission. If another node transmits simultaneously, both nodes will stop transmission and wait for a random interval of time before initiating another transmission.

The DEQNA Ethernet interface encodes and decodes the data exchanged with the H4000 transceiver. In addition, it implements the functions of the data link layer by encapsulating and de-encapsulating messages, handling collisions, and filtering received messages. The DECnet software provides the remaining functions of the message transfer such as routing, end communications, and session control.

Up to eight Ethernet nodes can be connected to a single H4000 transceiver using the DELNI Ethernet concentrator as shown in Figure 5-15. These nodes can include Q-bus processors with the DEQNA interface, or UNIBUS PDP-11 or VAX processors with the DEUNA Ethernet interface. For localized connections, up to eight processor nodes can be interconnected using the DELNI concentrator without physical connection to the H4000 Ethernet transceiver.

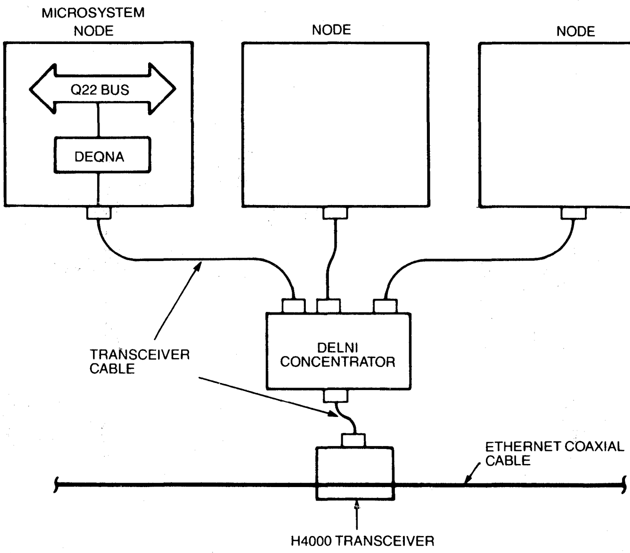


Figure 5-15 ▪ Ethernet Node with DELNI Concentrator

Asynchronous Links

The nodes in a network can be connected by asynchronous links when high-speed transfers and efficiency are not required. Data transmitted through the link is character-oriented. The characters can be five to eight bits in length, preceded by a start bit, and followed by stop bits. The time interval between the stop bits of one character and the start bit of the next character can vary, thereby reducing the efficiency of the data communications. The electrical and mechanical characteristics of the signals and interfaces are defined by standards created by the Electrical Industries Association (EIA) and the International Consultative Committee on Telegraphy and Telephony (CCITT).

The physical link can take one of two forms:

- EIA Standard Signals, Remote Connection—Communications between computer systems and devices over long distances can be implemented using modems and telephone lines as shown in Figure 5-16. The modems convert the system and device signal levels into signals acceptable by the telephone lines. The telephone lines can be private, leased, or part of the public-switched network. The remote device can be a terminal or any asynchronous interface of another computer system. The modems of each node must be of a compatible type.
- EIA Standard Signals, Local Connection—For local communications in the same area or within the same building, computer systems and terminals can be connected by EIA null modem cables as shown in Figure 5-17. The null modem cable transfers the serial EIA data and control signals between the nodes. The local device can be a terminal or an asynchronous interface of another computer system.

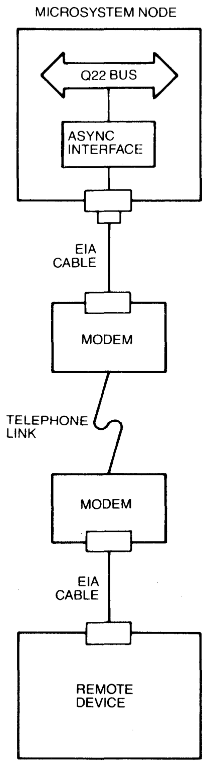


Figure 5-16 ▪ Remote Connection, Asynchronous Link

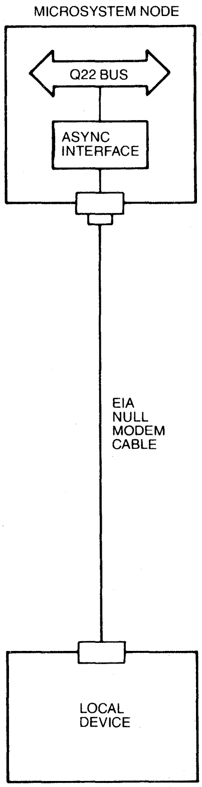


Figure 5-17 ▪ Local Connection, Asynchronous Link

For descriptions of optional asynchronous interface modules, refer to Chapter 3—System Options.

Synchronous Links

Synchronous links are used for communicating where speed and efficiency are important. Synchronous communications send a block of characters enclosed in a frame. The contents of the frame vary from one protocol to another, but they typically consist of text, identification of the beginning and the end of the frame, and information ensuring reliable reception of the text. Because the amount of extra information needed to complete the frame is fixed, the efficiency of synchronous transmission increases as the size of the textblock increases.

Some synchronous protocols, such as Digital's DDCMP and IBM's BISYNC, require the length of the text to be an integral number of characters or bytes. These are called character-oriented protocols. Others, including IBM's SDLC and the International Standards Organization's HDLC, allow the text length to be any number of bits. These are referred to as bit-oriented protocols.

The physical line can take one of three forms:

- Modem Connection, Remote Connection—For synchronous communications over long distances, the interface module is connected to a modem as shown in Figure 5-18. The modem connection can be specified by one of the EIA standards (RS232-C, RS422, or RS423) or by a CCITT standard (V.24, V.28, or V.35). The modem connects to a private line, a leased telephone line, or to the public-switched telephone network. The choice of modem and the line connecting the modems will depend on the speed of the data communications.
- Modem Eliminator, Local Connection—Connecting two local synchronous devices that interface via the EIA or CCITT standards requires a modem eliminator as shown in Figure 5-19. The distance between devices can be from several hundred feet to a few miles, depending on the speed of transmission and other factors.
- Integral Modem, Local Connection—The DMV11 series of synchronous interface modules contains an integrated modem that is compatible with the Digital DDCMP protocol. These interfaces can be used for point-to-point or multidrop-network configurations. Figure 5-20 shows the connections to the local-device interfaces.

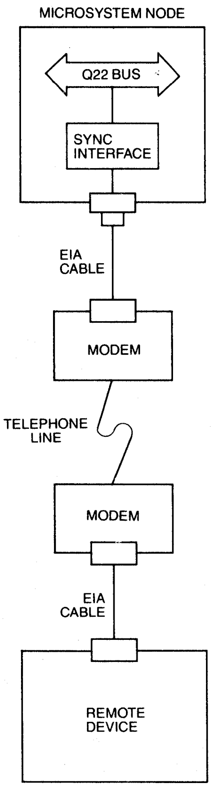


Figure 5-18 ▪ Remote Connection, Synchronous Link

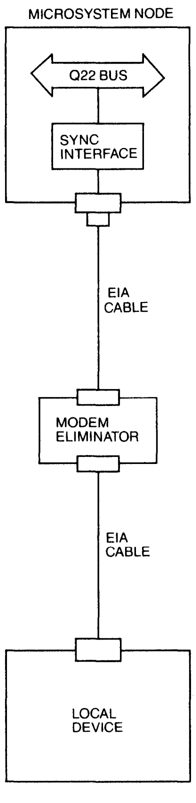


Figure 5-19 ▪ Local Connection, Synchronous Link

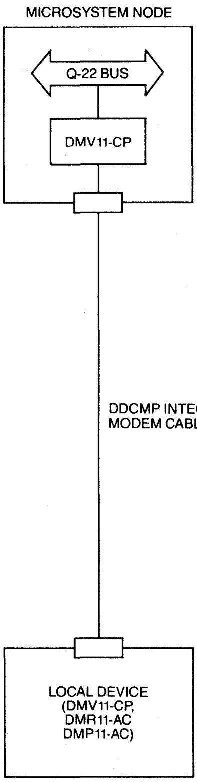

MICROSYSTEM NODE
Q-22 BUS
DMV11-CP
DDCMP INTEGRAL MODEM CABLE
LOCAL DEVICE (DMV11-CP, DMR11-AC DMP11-AC)

Figure 5-20 ▪ DDCMP Local Connection, Synchronous Link

For descriptions of optional synchronous interface modules, refer to Chapter 4—System Options.

▪ Network Software

After the network links are established, communications software makes the network functional within the DNA framework.

Digital's network software is broadly divided into three categories—DECnet, for communications among Digital systems; Internet, for communications with other manufacturers' equipment; and Packetnet, for communications with other participants in public packet-switched networks.

DECnet Communications

DECnet software supports communications among Digital computer systems. Data on the physical links is independent of system type, and the DECnet software converts the received data into formats that the operating system is prepared to accept. DECnet allows full use of the network by providing higher-level network functions. The following are the key elements:

- *Task-to-task communications* allow programs that are executing in different systems, under different operating systems, and written in different languages, to exchange information.
- *File transfer* supports the exchange of sequential ASCII or binary files between different operating systems.
- *Remote file access* allows the user to read, write, or modify files on another system.
- *Remote command file submission and execution* allow one computer system to direct another to execute commands that are resident on the remote system or sent as part of the request.
- *Downline loading* allows programs developed on a system with appropriate peripherals and resources to be transmitted to another system such as a small, memory-only system, for execution.
- The *network command terminal* gives a terminal user logical connection to a remote system with the same operating system; the terminal operates as if it were directly connected to the remote system.
- *Network management* provides the tools for monitoring and controlling network operation.

Refer to Table 5-1 for a complete comparison of DECnet products.

Internet Communications

Digital's microsystems can communicate with another vendor's equipment through software that emulates a protocol supported by that vendor. Although the name of the protocol may correspond to a specific device made by another manufacturer, the microsystems emulate only the communications protocol used by that device and not the capabilities of the device.

Microsystem nodes in an Ethernet link can also communicate with IBM systems through a Systems Network Architecture (SNA) gateway. The SNA gateway is an Ethernet node solely responsible for interfacing to an IBM system using IBM's System Network Architecture. Figure 5-21 shows the SNA gateway connections.

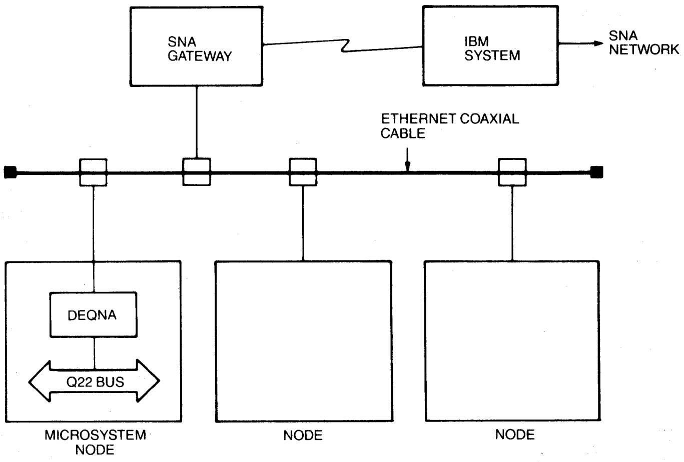


Figure 5-21 ▪ Ethernet to SNA Gateway Node

Packetnet System Interface

Public packet-switched networks can be an alternative to leased or dialup telephone lines for long-distance communications.

The charge is based on the volume of data transmitted rather than the fixed charge for a leased line. Access, speed, and reliability are better than that provided by a dialup line. The network also compensates for differences in transmission speeds between nodes and may offer services in addition to communications.

The Packetnet System Interface (PSI) software can coexist with, or operate as a layered product under, DECnet software. This allows DECnet facilities to be used between nodes connected through the packet-switched network and through leased or dialup lines. PSI makes communications possible with any other system (Digital or non-Digital) connected to the packet-switched network. PSI software supports task-to-task communications and remote terminal access to the microsystems.

The communications protocol, part of the data link layer, is implemented in the hardware of some interface modules. For other interface modules, this protocol is provided by communications software.

Table 5-1 ▪ DECnet Products and Features

	DECnet Products					
Capability	-VAX	-11M-PLUS	-11M	-11S	/E	-RT
Task-to-task communications	X	X	X	X	X	X
File transfer	X	X	X	X	X	X
Remote file access	X	X	X	X[1]	X	X
Remote command file submission	X	X	X		X	X[2]
Remote command file execution	X		X	X	X	X[2]
Downline loading	X	X	X			
Network command terminal	X	X	X	X[3]	X	X
Network management	X	X	X		X	X

[1] Offers local users network access to remote file systems. Does not allow users on remote systems to access local files.

[2] Requester-only functions.

[3] DECnet-11S does not support connection from remote command terminals.

▪ Single-user System Interconnection

The Professional 300 series, DECmate II and III, and the Rainbow 100 series can be connected to the microsystems over an asynchronous line. Although these communications do not use DECnet software, they do fall into the Digital-to-Digital communications category.

Two modes of communications are supported:

- *File transfer* supervises the transmission of an entire file between a single-user system and a microsystem without operator intervention.
- For file transfer with a Professional, the microsystems must have the Host File Transfer Package running under RSX-11M, RSX-11M-PLUS, or VMS. The Professional must have the PRO/Communications Package.
- For file transfer with a DECmate or a Rainbow, the microsystems must have DX/11M running under RSX-11M or RSX-11M-PLUS, DX/RSTS running under RSTS/E or Micro/RSTS, or DX/VMS running under VMS. DECmate must have the WPS-8 Communications Package. Rainbow must have the VT102 or VT125 Communications Package.
- *Terminal emulation* provides character-by-character communications that looks to the microsystems like a terminal.
- The Professional, along with the PRO/Communications Package, can emulate an alphanumeric terminal (VT102) or, with the extended bit-map module, a graphics terminal (VT125). DECmate with the WPS-8 Communications Package emulates an alphanumeric terminal (VT100 or VT52). The Rainbow emulates an alphanumeric terminal with the VT102 Communications Package, and a graphics terminal with the VT125 Communications Package.

▪ Additional Documentation

Networks and Communications Buyer's Guide	*ED-26127-42*
Networks Handbook	*EB-26013-42*
PDP-11 Microcomputer Interfaces Handbook	*EB-23144-18*

Chapter 6 • Architecture Summary

Introduction

Computer architecture is defined as the characteristics of the computer that are observed by the operator and programmer at the assembly-language level. These characteristics, which exist in both the VAX and PDP-11 architectures, include instructions sets, data types, addressing modes, registers, address space, and memory management. This chapter discusses the similarities of and differences between these architectural characteristics.

Multiple system implementations of common computer architectures have allowed Digital's customers to continue to upgrade and expand, at the lowest possible cost, as their needs have changed. Within each of the VAX and PDP-11 families, customers can move to other computers without reinvesting in software, peripherals, communications devices, or training. Common computer architectures ensure computer compatibility.

For a simple overview listing of the architectural characteristics of these two families, refer to Table 6-1. For detailed descriptions of the VAX and PDP-11 architectures, refer to the *VAX Architecture Handbook* and to the *PDP-11 Architecture Handbook*. Ordering information for these books is provided in Appendix E—Documentation.

Table 6-1 • Architectural Characteristics Overview

Characteristic	MicroVAX I	MicroPDP-11/73	MicroPDP-11/23
Physical Address Space	Model dependent: 4 Mbytes	4 Mbytes	4 Mbytes
Virtual Address Space	4 Gbytes total instruction and data space	64 Kbytes instruction space 64 Kbytes data space	64 Kbytes instruction space 64 Kbytes data space
I/O System	Memory-mapped 8 Kbytes I/O space	Memory-mapped 8 Kbytes I/O space	Memory-mapped 8 Kbytes I/O space

Characteristic	Micro-VAX I	Micro-PDP-11/73	Micro-PDP-11/23
Operand Types	Bit field	—	—
	8-bit byte	8-bit byte	8-bit byte
	16-bit word	16-bit word	16-bit word
	32-bit longword	—	—
	32-bit floating[1]	32-bit floating	32-bit floating
	64-bit floating[1]	64-bit floating	64-bit floating
	64-bit floating with different precision[1]	—	—
	128-bit floating[1]	—	—
Registers	12 general	6 general	6 general
	3 stack pointers	3 stack pointers	3 stack pointers
	1 program counter	1 program counter	1 program counter
CPU Execution State	32-bit processor status longword	16-bit processor status word	16-bit processor status word
Addressing Modes	Register[2]	Register	Register
	Register-deferred	Register-deferred	Register-deferred
	Auto-increment	Auto-increment	Auto-increment
	Auto-decrement	Auto-decrement	Auto-decrement
	Auto-increment-deferred	Auto-increment-deferred	Auto-increment-deferred
	Auto-decrement-deferred	Auto-decrement-deferred	Auto-decrement-deferred
	Index	Index	—
	Displacement-deferred	—	—
	Literal	—	—

Characteristic	Micro-VAX I	Micro-PDP-11/73	Micro-PDP-11/23
Interrupt System	Automatically vectored interrupts	Automatically vectored interrupts	Automatically vectored interrupts
	4 hardware levels	4 hardware levels	4 hardware levels
	32 software levels	8 software levels	8 software levels
Operating Modes	Kernel	Kernel	Kernel
	Executive	—	—
	Supervisor	Supervisor	Supervisor
	User	User	User
Data Types	Bit field	—	—
	Byte	Byte	Byte
	Word	Word	Word
	32-bit longword	—	—
	64-bit longword	—	—
	—	—	Character

Characteristic	Micro-VAX I	Micro-PDP-11/73	Micro-PDP-11/23
Instruction Sets	PDP-11 superset VAX-11 subset	Basic PDP-11 with EIS	Basic PDP-11 with EIS
Instruction Classes	Integer arithmetic	Integer arithmetic	Integer arithmetic
	Integer logical	Integer logical	Integer logical
	Floating point	Floating point	Floating point
	Byte string[1]	Byte string (including decimal byte string)	Byte string (including decimal byte string)
	Basic flow control (branch, jump)	Basic flow control (branch, jump)	Basic flow control (branch, jump)
	Basic call	Basic call	—
	Enhanced flow control (do loops)	—	—
	Queue management	—	—
	Polynomial (Taylor Series)	—	—
	Enhanced procedure call	—	—
Instruction Length	1^{-n} bytes	2, 4, or 6 bytes (CIS instructions longer)	2, 4, or 6 bytes (CIS instructions longer)
Floating Point	D and F or F and G standard	D and F standard	D and F optional
CIS	—	—	Optional

[1] Emulated

[2] Indexed variations

MicroVAX I Architectural Characteristics

The MicroVAX I architecture includes the following characteristics:

- 4-Gbyte virtual-address space
- 32-bit operand size
- Memory management
- 16 32-bit general registers
- Multiple addressing modes
- 32 processor-priority levels
- Vectored hardware and software interrupts
- Variable instruction size
- Subset of the VAX instruction set
- Emulation support for the unimplemented VAX instruction set except for PDP-11 compatibility mode
- Subset of the VAX privileged registers
- Subset of the VAX data types

The MicroVAX instruction set uses 32-bit addressing that enables the processor to address up to four billion bytes of virtual-address space. The processor's memory-management hardware includes mapping registers used by the operating system that provide page protection by operating mode.

The MicroVAX provides 16 32-bit general registers that can be used for high-speed, temporary storage or as accumulators, index registers, or base registers. One of these registers is a program counter and three registers provide Procedure Call instructions.

The processor also offers a variety of addressing modes, including an indexed-addressing mode, that use the general registers to identify instruction operand locations.

The instruction set includes character-string and floating-point instructions, as well as integer, logical, queue, and bit-field instructions. Instructions and data can start at any arbitrary byte boundary in memory or, in the case of bit fields, at any arbitrary bit in memory.

For further reference, the section "KD32-A CPU Modules" in Chapter 2—System Hardware describes the MicroVAX I CPU as an implementation of the VAX architecture.

MicroPDP-11 Architectural Characteristics

The MicroPDP-11 architecture includes the following characteristics:

- 64-Kbyte virtual-address space
- 4-Mbyte physical-address space
- 16-bit operand size
- Memory management
- One or two sets of eight 16-bit general registers
- Multiple addressing modes
- Eight processor-priority levels
- Variable instruction size
- Full set of PDP-11 privileged registers
- Full set of PDP-11 data types
- Full set of PDP-11 instructions

The MicroPDP-11 instruction set uses 16-bit addressing that provides a directly addressable virtual-address space of 65,536 (64K) bytes. Actual memory capacity in the MicroPDP-11 is 4096K (4M) bytes. Memory management translates the 16-bit virtual addresses into the full 22-bit physical addresses needed to address 4 Mbytes. The processor's memory-management hardware includes mapping registers used by the operating system that provide page protection by operating mode.

The MicroPDP-11/23 provides eight 16-bit general registers that can be used for high-speed, temporary storage or as accumulators, index registers, or base registers. Two registers with special purposes are the program counter and the stack pointer. The MicroPDP-11/73 has a second set of the eight 16-bit general registers with a program counter and stack pointer for each set.

The MicroPDP-11 processors offer a variety of addressing modes, including an indexed-addressing mode, that uses the general registers to identify instruction operand locations.

Floating-point instructions are standard on the MicroPDP-11/73 and available as an option on the MicroPDP-11/23. Character-string instructions are offered as an option on the MicroPDP-11/23 only.

For further reference, the sections "KDJ11 CPU Module" and "KDF11 CPU Module" in Chapter 2—System Hardware describe the MicroPDP-11 CPUs as an implementation of the PDP-11 architecture.

▪ Address Space and Memory

The MicroVAX I uses the 8-bit byte for addressing. MicroVAX I instructions use a 32-bit virtual address to identify these byte locations. It is called a virtual address because it is not the real address of a physical-memory location. It is translated into a real address by the processor under operating-system control. A virtual address is not a unique address of a location in memory, as are physical-memory addresses. Two programs using the same virtual address might refer to two different physical memory locations, the same physical memory location, or the same physical memory location using different virtual addresses. The set of all possible 32-bit virtual addresses is called virtual-address space.

The MicroPDP-11 also uses the 8-bit byte for addressing. PDP-11 instructions use a 16-bit virtual address to identify a byte location. Memory management translates 16-bit virtual addresses into the 22-bit physical addresses needed to address 4 Mbytes of memory.

MicroPDP-11 instructions can address memory using either direct addressing or indirect (deferred) addressing. With direct addressing, the 16-bit address in one of the general registers points directly to the data in memory (Figure 6-1). With indirect addressing, the 16-bit address in a general register points to an address stored in memory, which in turn points to the data (Figure 6-2).

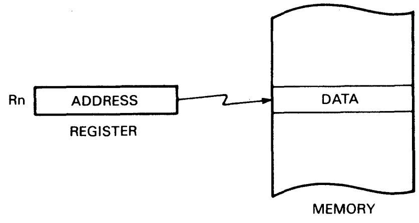


Figure 6-1 ▪ Direct Addressing

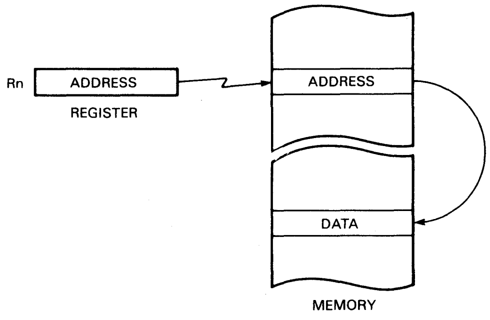


Figure 6-2 ▪ Indirect or Deferred Addressing

MicroVAX and MicroPDP-11 memory locations and peripheral-device (I/O-device) registers are addressed in the same manner. The upper 8 Kbytes of physical-address space are reserved for I/O-device addressing. Other physical-memory locations have been reserved for interrupt and trap handling.

Physical-address Space

Physical-address space is a contiguous series of word-addressable hardware locations used to define memory and I/O-device registers.

A physical address in the MicroVAX I is 23 bits long and provides a physical-address space of 8 Mbytes. Of these, 4 Mbytes are in memory space and 4 Mbytes are in I/O space. The I/O space is largely empty usually utilizing only the first 8 Kbytes.

The MicroPDP-11 architecture specifies that physical addresses may be up to 22 bits long and provides a physical-address space of 4 Mbytes. As in the MicroVAX I, the first 8 Kbytes are used for I/O-device addressing.

Virtual-address Space

Through the MicroVAX and MicroPDP-11 memory-management hardware, the operating system provides an execution environment in which users can write programs without having to know where the programs are loaded in physical memory. In this environment, users can write programs that are too large to fit into the allocated physical memory. This environment is called virtual-address space.

A virtual address is a 16-bit or 32-bit integer that a program uses to identify a storage location in virtual memory. Virtual memory may be the set of all physical-memory locations in the system plus the set of disk blocks that the operating system designates as extensions of physical memory.

A program written for a MicroVAX processor sees a 32-bit address space that references up to four billion bytes. This 4-Gbyte window is known as the program's virtual-address space. Each MicroVAX program's virtual-address space begins with address 0 and can extend upward to a maximum of 4 Gbytes.

A program written for a MicroPDP-11 processor sees a 16-bit address space that references up to 64 Kbytes of memory. This 64-Kbyte window is known as the program's virtual-address space. Each MicroPDP-11 program's virtual-address space begins with address 0 and can extend upward to a maximum of 64 Kbytes.

Memory Management

Memory management enables the operating system to map virtual addresses into physical addresses. This physical address is then used to specify a location in the storage device.

The MicroPDP-11 system has to convert a fairly small virtual address to a large physical address. This allows the 16-bit MicroPDP-11 processor to access a very large physical memory a little at a time. Figure 6-3 shows the MicroPDP-11 translating the 16-bit virtual addresses into the 22-bit physical addresses.

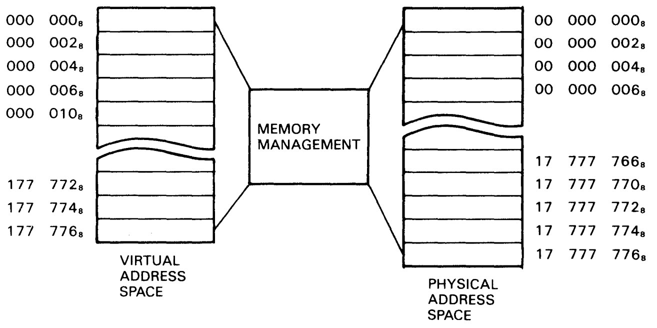


Figure 6-3 ▪ MicroPDP-11 Memory Management

The MicroVAX I memory management is almost exactly the reverse of the process for the MicroPDP-11. The MicroVAX processor can express virtual addresses over a 4-Gbyte range, yet no real memory exists that can hold such a program. A scheme must be used to allow the physical memory to contain only those parts of the virtual-address space that are currently in use.

A combination of MicroVAX microcode and operating-system software creates this memory-management environment. Initially, none of the user's program is in physical memory. Instead, it is all on the storage device. As the user's program references its virtual-address space, the necessary "pages" of the user's program are brought into physical memory and the user's virtual addresses are mapped (pointed or translated) to the appropriate physical addresses. Eventually, physical memory becomes full and the operating system must create space for new pages. The oldest pages are kicked out of physical memory (i.e., copied back out to the storage device) allowing the newest pages to fit in. This entire process is called demand paging and is central to how the MicroVAX can run programs that are much larger than the existing physical memory. Figure 6-4 shows the MicroVAX translating the 32-bit virtual addresses into the 23-bit physical addresses.

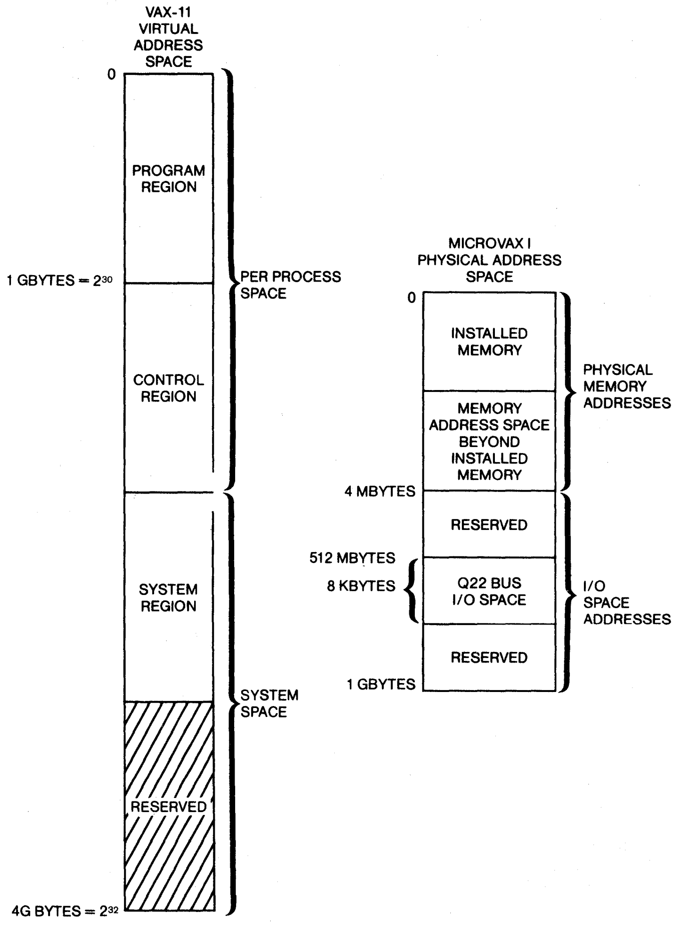


Figure 6-4 ▪ MicroVAX Memory Management

Memory Protection

The MicroVAX and MicroPDP-11 memory management provides a second important feature beyond managing where the code and data should go. Memory management also controls who may have access to the code and data. This is important if a multiuser system is to protect the operating-system software from the users as well as protect individual programs from one another.

The view that each running program has of physical memory is completely controlled by the operating system. To the program, sections of physical memory can be labeled read/write, read-only, or invisible. For example:

- The program code should be marked read-only if the programmer has written pure code that does not modify itself.
- Fixed program data can also be marked read-only so that it cannot be damaged.
- Variable program data is marked read/write.
- Sections of memory the program has no need to know about are made invisible.

Both the MicroVAX and the MicroPDP-11 families assign these protection attributes on a per-page basis.

Registers and Stacks

A register is a location within the processor that can be used for high-speed, temporary data storage, and addressing, or as an accumulator during computation.

The MicroVAX has 16 32-bit registers available for use with the native instruction set. Twelve of these registers are for general purposes. The rest of these registers have special purposes. One of these special-purpose registers is designated as the program counter and contains the address of the next instruction to be executed. The other three special-purpose registers are designated for use with procedure call instructions. They are the stack pointer, argument pointer, and frame pointer.

The MicroPDP-11/23 uses one set of eight 16-bit registers. Six of these are general-purpose registers and the remaining two are special-purpose registers. The special-purpose registers are the program counter and the stack pointer. The MicroPDP-11/73 uses two sets of eight 16-bit registers. Each set has the six general-purpose registers, a program counter, and a stack pointer.

A stack is an array of consecutively addressed data items that are referenced on a last-in, first-out basis using a register. Data items are added to and removed from the low address end of the stack. A stack grows toward lower addresses as items are added and shrinks toward higher addresses as items are removed.

A stack can be created anywhere in the user's program address space. Any register can be used to point to the current item on the stack. The operating system, however, automatically reserves portions of each process address space for stack data structures. User software references its stack data structure, called the user stack, through a general register designated as the stack pointer. When the user runs a program image, the operating system automatically provides the address of the area designated for the user stack.

MicroVAX Registers

The 16 32-bit registers in the MicroVAX (shown in Figure 6-5) are labeled R0 through R15 (in decimal). Registers can be used for temporary data storage, or as accumulators, base registers, or index registers. A base register contains the address of the base of a software data structure such as a table, and an index register contains a logical instruction into a data structure.

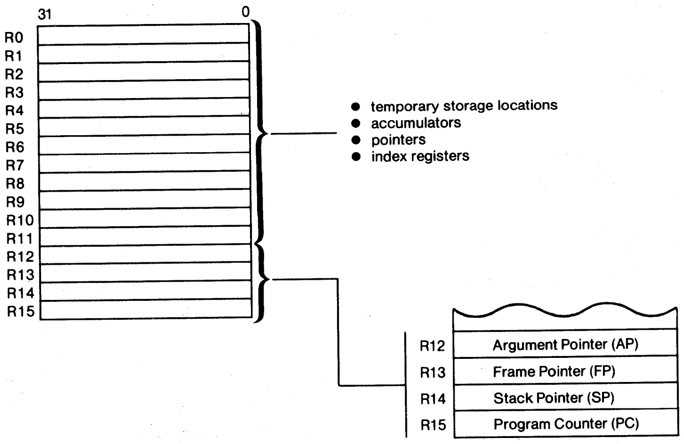


Figure 6-5 ▪ MicroVAX Registers

Some registers have special significance, depending on the instruction being executed. Registers R12 through R15 have special significance for many instructions and subsequently have special labels. These registers are described below.

- *Program counter* (PC or R15) contains the address of the next byte to be processed in the instruction stream.
- *Stack pointer* (SP or R14) contains the address of the top of a stack maintained for subroutine and procedure calls.
- *Frame pointer* (FP or R13) contains the address of the base of a software data structure stored in the stack.
- *Argument pointer* (AP or R12) contains the address of the base of a software data structure called the argument list.

MicroVAX Program Counter

A MicroVAX native-mode instruction has a variable-length format and instructions are byte-aligned. A variable-length format not only makes code more compact, but it also can easily extend the instruction set. The opcode for the operation is either one or two bytes long and is followed by zero to six operand specifiers, depending on the instruction. An operand specifier can be one or several bytes long, depending on the addressing mode. Figure 6-6 illustrates the representation of an instruction as a string of bytes. Just before the processor begins to execute an instruction, the program counter contains the address of the first byte of the next instruction. The way in which the program counter is updated is totally transparent to the programmer.

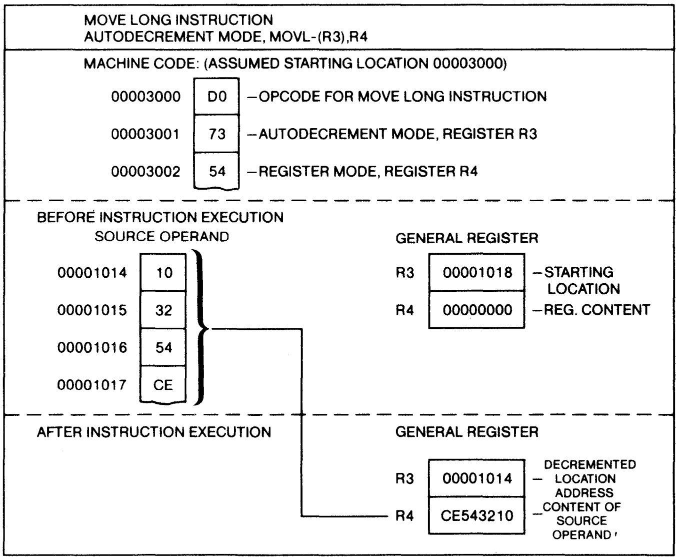

MOVE LONG INSTRUCTION
AUTODECREMENT MODE, MOVL-(R3),R4
MACHINE CODE: (ASSUMED STARTING LOCATION 00003000)
00003000 D0 –OPCODE FOR MOVE LONG INSTRUCTION
00003001 73 –AUTODECREMENT MODE, REGISTER R3
00003002 54 –REGISTER MODE, REGISTER R4
BEFORE INSTRUCTION EXECUTION
SOURCE OPERAND
00001014 10
00001015 32
00001016 54
00001017 CE
GENERAL REGISTER
R3 00001018 –STARTING LOCATION
R4 00000000 –REG. CONTENT
AFTER INSTRUCTION EXECUTION
GENERAL REGISTER
R3 00001014 – DECREMENTED LOCATION ADDRESS
R4 CE543210 – CONTENT OF SOURCE OPERAND

Figure 6-6 ▪ MicroVAX Instruction Representation

MicroVAX Stack Pointer, Argument Pointer, and Frame Pointer

The stack pointer is a register specifically designated for use with stack structures. The stack pointer can place items on, or remove items from, the stack.

The argument pointer is used to pass the address of the argument list to a called procedure, and the frame pointer is used to keep track of the nested call instructions.

An argument list is a formal data structure containing the arguments required by the procedure being called. Arguments can be actual values, addresses of data structures, or addresses of other procedures.

The call instructions always keep track of nested calls with the frame pointer register. The frame pointer contains the address on the stack of the items pushed on the stack during the procedure call. The set of items pushed on the stack during a procedure call is known as a call frame or stack frame.

MicroVAX Processor Status Longword

A processor register in the MicroVAX called the processor status longword (PSL) determines the execution state of the processor at any time. The low-order 16 bits of the processor status longword constitute the processor status word available to the user process. The high-order 16 bits provide privileged control of the system. The fields can be grouped together by function to control the operating mode of the current instruction, and the interrupt processing. Figure 6-7 shows the processor status longword.

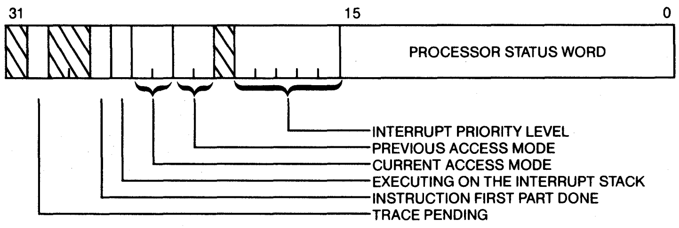


Figure 6-7 ▪ Processor Status Longword

MicroPDP-11 Registers

The MicroPDP-11 registers can be used as operands for arithmetic and logical operations or for addressing in memory. Register operations are internal to the processor and do not require bus cycles (except for instruction fetch). All memory and peripheral device data transfers require bus cycles and longer execution time. Thus general-purpose registers used for processor operations result in faster execution times.

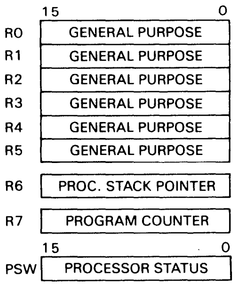


Figure 6-8 ▪ MicroPDP-11 Registers

The program counter (PC or R7) contains the address of the next instruction to be executed. Normally, the PC is used only for addressing and not for arithmetic or logical operations.

When an interrupt or trap occurs, the processor status word (PSW) and the program counter are saved on the processor stack. The stack pointer (SP or R6) contains the address of the top of this stack in memory. The PSW and PC contain all the information needed for the processor to resume execution where it left off. The last-in, first-out stack allows orderly processing of interrupts and traps even when the processor is already processing.

Processor Status Word

The processor status word (PSW) is a special processor register found in the MicroVAX and MicroPDP-11 that is used to check a program's status and to control synchronous error conditions. The processor status word, shown in Figure 6-9, contains two sets of bit fields—condition codes and trap enable flags.

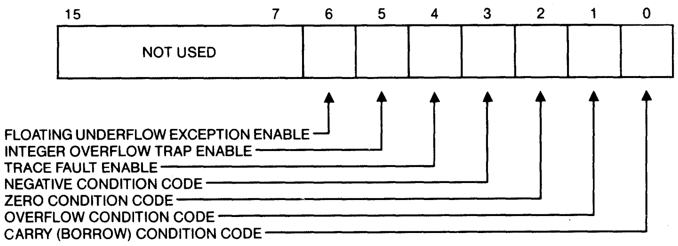


Figure 6-9 ▪ Processor Status Word

The condition codes indicate the outcome of a particular logical or arithmetic operation. The branch-on-condition instructions can be used to transfer control to a code sequence that handles the condition.

There are two kinds of exceptions that concern the user process—trace faults and arithmetic exceptions. The trace fault is used to debug programs or evaluate performance. Arithmetic exceptions include:

- Integer or floating-point overflow, in which the result was too large to be stored in the given format
- Integer or floating-point divide-by-zero, in which the divisor supplied was zero
- Floating-point underflow, in which the result was too small to be expressed in the given format

When an exception occurs, the processor immediately saves the current state of execution and traps to the operating system. The operating system automatically searches for a procedure that wants to handle the exception.

▪ Addressing Modes

The processor's addressing modes allow almost any operand to be stored in a register or in memory, or as an immediate constant. Table 6-2 summarizes the addressing modes.

Table 6-2 ▪ Addressing Modes

Addressing Modes	Micro-VAX I	Micro-PDP-11/73	Micro-PDP-11/23
Register (General-purpose register contains data)	X	X	X
Register-deferred (General-purpose registers contains address of data)	X (plus indexed variation[1])	X	X
Auto-increment (General-purpose register contains address of data. After access, GPR is incremented to point to next address of data.)	X (plus indexed variation[1])	X	X
Auto-increment-deferred	X (plus indexed variation[1])	X	X
Auto-decrement	X (plus indexed variation[1])	X	X
Auto-decrement-deferred	—	X	X
Displacement (MicroPDP-11 Index)	X (plus indexed variation[1])	X	X
Displacement-deferred (MicroPDP-11 Index-deferred)	X (plus indexed variation[1])	X	X

[1] Second register (multiplied by data size) is added to address.

MicroVAX Addressing Modes

There are seven basic MicroVAX addressing modes that use the general registers to identify the operand location. They include:

- *Register mode* (mode 0)—the register contains the operand.
- *Register-deferred mode* (mode 1)—the register contains the address of the operand.
- *Autodecrement mode* (mode 2)—the contents of the register are first decremented by the size of the operand and then used as the address of the operand. The size of the operand (in bytes), given by the data type of the instruction operand, depends on the instruction. For example, the Clear Word instruction uses a size of two, because there are two bytes per word.
- *Autoincrement mode* (mode 3)—the contents of the register are used as the address of the operand, and then incremented by the size of the operand. If the program counter is the specified register, the mode is called immediate mode.
- *Autoincrement-deferred mode* (mode 4)—the contents of the register are used as the address of a location in memory containing the address of the operand and then are incremented by four (the size of the address). If the program counter is the specified register, the mode is called absolute mode.
- *Displacement mode* (mode 5)—the value stored in the register is used as a base address. A byte, word, or longword signed constant is added to the base address, and the resulting sum is the effective address of the operand.
- *Displacement-deferred mode* (mode 6)—the value stored in the register is used as the base address of a data structure. A byte, word, or longword signed constant is added to the base address, and the resulting sum is the address of the location that contains the actual address of the operand.

The autoincrement and autodecrement modes enable automatic stepping through tables. The displacement mode enables the generation of offsets into a table, with a choice of either short or long displacements. The deferred modes enable the user to maintain tables of operand addresses instead of the operands themselves. The indexed-addressing modes allow indexing into tables with a step size automatically determined by the operand. All except register mode can be modified by indexed addressing.

The MicroVAX processor also provides literal mode, in which an unsigned six-bit field in the instruction is interpreted as an integer or floating-point constant.

The MicroVAX processor's addressing modes allow considerable flexibility in the arrangement and processing of data structures. A data structure's design does not have to be tied to its processing method to be efficient. The variety of addressing modes enables the assembly language programmer to write, and high-level language compilers to produce, very compact code. For example, literal mode is a very efficient way to specify small constants.

MicroPDP-11 Addressing Modes

There are eight basic PDP-11 addressing modes that use the general registers to identify the operand location.

- *Register mode* (mode 0)—contains the operand in the register.
- *Register-deferred mode* (mode 1)—contains the address of the operand in the register.
- *Autoincrement mode* (mode 2)—interprets the contents of the register as the address of the operand in memory and, in addition, increments the contents of the register by 2 (word instructions) or by 1 (byte instructions) after the operand is accessed in memory. This leaves the register pointing to the next consecutive word or byte and makes stepping through a list of operands easier. Because both R6 (stack pointer) and R7 (program counter) normally contain addresses, they are always autoincremented by 2.
- *Autoincrement-deferred mode* (mode 3)—uses the contents of the register as a pointer to the address of the operand. The pointer in the register is then incremented by 2 after the address is located. Where mode 2 steps through a list of sequential operands, mode 3 steps through a list of sequential addresses that in turn point to operands stored anywhere in memory.
- *Autodecrement mode* (mode 4)—decrements the contents of the register by 2 (word instructions) or by 1 (byte instruction) before using the register as the address of an operand in memory. Where mode 2 steps through a list of operands at ascending addresses, mode 4 does the same by descending addresses.
- *Autodecrement-deferred mode* (mode 5)—interprets the contents of the register as the address of a word in memory, which in turn points to the operand. The register is decremented by 2 before accessing the address in memory. Where mode 3 steps through a list of addresses in ascending memory order, mode 5 steps through the list in descending memory order.

- *Index mode* (mode 6)—adds the contents of the word immediately following the instruction to the contents of the register, and uses the resulting sum as the address of an operand in memory. This allows you to specify the starting address of a list independently of the offset of an entry in the list. By changing the starting address but not the index, you can move the fifteenth entry in list A to the fifteenth entry in list B. By changing the index but not the starting address, you can move from the fifteenth entry in list A to the twentieth entry in list A. The starting address can be specified in the register and the offset in the word following the instruction, or vice versa.
- *Index-deferred mode* (mode 7)—adds the word following the instruction to the register in the same way as mode 6, but the resulting sum is used as the address of a word in memory that in turn points to the operand. Where mode 6 accesses operands stored in a list or table, mode 7 uses addresses stored in a list or table to access operands stored anywhere in memory.

Modes 2, 3, 6, and 7 can be particularly useful in conjunction with the program counter. The resulting addressing will be independent of where in memory the instruction is executed. Each time the processor implicitly uses the program counter to fetch a word from memory, the program counter is automatically incremented by 2 after the fetch is completed.

PC immediate mode is a special case of mode 2, using the program counter (R7) as the register. It accesses the word immediately following the instruction and is a fast way to read a constant operand.

PC absolute mode is a special case of mode 3, where an absolute address (i.e., constant regardless of where in memory the instruction is executed) is stored in the word immediately following the instruction. This absolute address is used as a pointer to the operand.

PC relative mode is a special case of mode 6, using the program counter (R7) as the register. The updated contents of the program counter (instruction address + 4) are added to the contents of the word immediately following the instruction (the offset) and the sum is used as a pointer to the operand in memory. The offset and the position of the operand relative to the instruction are independent of where they are located in memory, so PC relative mode is helpful in writing position-independent code.

PC relative-deferred mode is a special case of mode 7, using the program counter as the register. The word following the instruction (the offset) is added to the updated program counter (instruction address + 4), and the resulting sum is used as a pointer to a location which in turn contains the address of the operand.

▪ Exceptions and Interrupts

While running one process, the processor executes instructions and controls data flow to and from peripherals and main memory. To share processor, memory, and peripheral resources among many processes, the processor provides two arbitration mechanisms called exceptions and interrupts. Exceptions are events that occur synchronously with respect to instruction execution; interrupts are external events that occur asynchronously.

The flow of execution can change at any time. The processor distinguishes between changes in flow that are local to a process and those that are of systemwide context and independent of any particular process. Process-local changes occur as the result of a user software error or when user software calls operating-system services. Process-local changes in program flow are handled through the processor's exception-detecting mechanism and the operating system's exception dispatcher.

Systemwide changes in flow generally occur as the result of interrupts from devices or interrupts generated by the operating-system software. Interrupts are handled by the processor's interrupt-detection mechanism and the operating system's interrupt-service routines (systemwide changes in flow may also occur as the result of severe hardware errors; these are handled either as special exceptions or high-priority interrupts).

Systemwide changes in flow generally take priority over process-local changes in flow. The processor uses a priority system for servicing interrupts. To arbitrate between all possible interrupts, each kind of interrupt is assigned a priority, and the processor responds to the highest-priority pending interrupt. For example, interrupts from realtime I/O devices would take precedence over interrupts from mass-storage devices, terminals, lineprinters, and other less time-critical devices.

The processor services interrupts between instructions or at well-defined points during the execution of long iterative instructions. When the processor acknowledges an interrupt, it switches rapidly to a special systemwide context so that the operating system can service the interrupt. Systemwide changes in the flow of execution are handled in a way that makes them totally transparent to individual processes.

Exception and Interrupt Vectors

The processor can automatically initiate changes in the normal flow of program execution. The processor recognizes two kinds of events that cause it to invoke conditional software—exceptions and interrupts. Some exceptions, such as arithmetic traps, affect an individual process only. Others affect the system as a whole, such as a machine check. Interrupts include both device interrupts, such as those signaling I/O completion, and software-requested interrupts, such as those signaling the need for a context-switch operation.

The processor knows which software to invoke when an exception or interrupt occurs because it references specific locations, called vectors, to obtain the starting address of the exception or interrupt dispatcher. Each vector tells the processor how to service the event.

Processor-priority Levels

To arbitrate between interrupt requests that can occur simultaneously, the MicroVAX processor recognizes 32 processor-priority levels. The highest 16 processor-priority levels are reserved for interrupts generated by hardware, and the lowest 16 processor-priority levels are reserved for interrupts requested by software.

The MicroPDP-11 processors recognize eight processor-priority levels. The highest four processor-priority levels are reserved for interrupts generated by hardware, and the lowest four levels are reserved for interrupts requested by software.

Context Switching

In the multiprogramming environment, several individual streams of code can be ready to execute at the same time. Instead of allowing each stream to execute one at a time, the operating system can intervene and switch between the streams of code that are ready to execute. The stream of code the processor is executing at any one time is determined by its hardware context.

The hardware context contains the information that is loaded in the processor's registers that identify where the stream of instructions and data are located, which instruction to execute next, and what the processor is doing during execution.

The process is the stream of instructions and data defined by the hardware context. Each process has a unique identification in the system. The operating system switches between processes by requesting the processor to save one process hardware context and load another.

▪ Processor Operating Modes

In a high-performance, multiprogramming system, the processor must provide the basis for protection and sharing among the processes competing for the system's resources. The basis for protection in the system is the processor's operating mode. The operating mode in which the processor executes determines:

- Instruction-execution privileges—which instructions the processor will execute
- Memory-access privileges—which locations in memory the current instruction can access

At any one time, the processor is either executing code in the context of a particular process, or it is executing code in the systemwide interrupt-service context.

Kernel mode allows execution of all instructions. In a multiprogramming environment, the most privileged functions of the operating system—physical I/O operations, resource management, and job scheduling—are implemented in code that runs in kernel mode. The access-control provisions of memory management protect these elements from tampering by programs running in less privileged modes.

Executive mode allows Record Management Services (RMS) any many of the operating system's programmed service procedures to execute.

User mode prohibits the execution of instructions, such as halt and reset, that would allow one program in a multiprogramming environment to harm the system as a whole. Each user's virtual-address space permits writing only into its own areas in memory.

Supervisor mode has the same level of privilege as user mode and can be useful for programs being shared among users but still requiring protection.

Table 6-3 shows both the MicroVAX and MicroPDP-11 processor operating modes.

Table 6-3 ▪ Processor Operating Modes

Operating Modes	MicroVAX I	MicroPDP-11/73	MicroPDP-11/23
Kernel	X	X	X
Executive	X	—	—
Supervisor	X	X	X
User	X	X	X

MicroVAX Operating Modes

In the MicroVAX context of a process, the processor recognizes four operating modes—kernel, executive, supervisor, and user. Kernel is the most privileged mode, and user, the least privileged. When in interrupt-service context, the processor recognizes only kernel mode.

The processor spends most of its time executing in user mode. When user software needs the services of the operating system—whether for acquisition of a resource, for I/O processing, or for information—the processor executes those services in the same operating mode or in one of the more privileged operating modes.

To execute code in one of the more privileged modes, the system manager must grant access and the operating system controls the operation. The memory protection that the privileged mode gives is enforced by the processor. In general, code executing in one mode can protect itself and any portion of its data structures from read and/or write accesses by code executing in any less privileged mode. This memory-protection mechanism provides the basis for data structure integrity.

MicroPDP-11 Operating Modes

In the MicroPDP-11 context of a process, the processor recognizes three access modes—kernel, user, and supervisor.

▪ Data Types

The data type of an instruction operand determines the number of bits of storage to be treated as a unit and what the interpretation of that unit is. Each microsystem's instruction set operates on integer, floating-point, and character-string data types. For each of these data types, the selection of an operation immediately tells the processor the size of the data and its interpretation.

The MicroVAX instruction set can also manipulate variable-length bit fields where the user defines the size of the field and its relative position. There are several variations of these primary data types. Table 6-4 provides a summary of the data types available.

Table 6-4 ▪ Data Type Representation

Data Types	Micro-VAX I	Micro-PDP-11/73	Micro-PDP-11/23
Integer (byte)	8 bits	8 bits	8 bits
	−128 to +127 signed	−128 to +127 signed	−128 to +127 signed
	0 to 255 unsigned	0 to 255 unsigned	0 to 255 unsigned
Integer (word)	16 bits	16 bits	16 bits
	−32768 to +32762 signed	−32768 to +32762 signed	−32768 to +32762 signed
	0 to 65535 unsigned	0 to 65535 unsigned	0 to 65535 unsigned

Data Types	Micro-VAX I	Micro-PDP-11/73	Micro-PDP-11/23
Integer (longword)	32 bits	—	—
	–231 to +231^{-1} signed	—	—
	0 to 232^{-1} unsigned	—	—
Integer (quadword)	64 bits	—	—
	–263 to +263^{-1}	—	—
	0 to 264^{-1} unsigned	—	—
F_floating point	4 bytes 7 decimal digits of precision	4 bytes 7 decimal digits of precision	4 bytes 7 decimal digits of precision
D_floating point	8 bytes 16 decimal digits of precision	8 bytes 16 decimal digits of precision	8 bytes 16 decimal digits of precision
G_floating point	8 bytes	—	—
	15 decimal digits of precision	—	—
Character string	0 to 65535 bytes	0 to 65535 bytes	0 to 65535 bytes
	One character per byte	One character per byte	One character per byte
Decimal string	—	0 to 31 bytes	0 to 31 bytes
	—	One digit per byte	One digit per byte
Bit field	0 to 32 bits	—	—
	Dependent on interpretation	—	—
Queue	$\geqq$2 longwords per queue entry	—	—

Integer data is stored as binary values in either byte or word formats on a MicroPDP-11, and in byte, word, longword, quadword, or octaword formats on a MicroVAX. A byte is 8 bits, a word is 2 bytes, a longword is 4 bytes, a quadword is 8 bytes, and an octaword is 16 bytes. The microsystem interprets an integer as either a signed value or an unsigned value. The sign in signed values is determined by the high-order bit.

Floating-point values are stored using a signed exponent and a binary, normalized fraction. Floating-point data types can represent positive and negative numbers with a much greater absolute value than integer data (as large as 1.7×10^{38}), or with a fractional value (as small as $.29 \times 10^{-38}$). PDP-11-based systems use F_floating-point and D_floating-point data types. MicroVAX I uses F_floating-point, D_floating-point, and G_floating-point data types in hardware and supports the H_floating-point data type in software.

- Single-precision F_floating-point data is 4 bytes long with an 8-bit excess 128 exponent. The effective 24-bit fraction yields approximately seven decimal digits of precision.
- Double-precision D_floating-point data is 8 bytes long with an 8-bit excess 128 exponent. The effective 56-bit fraction yields approximately 16 decimal digits of precision.
- G_floating-point data is 8 bytes long with an 11-bit excess 1024 exponent. The effective 53-bit fraction yields approximately 15 decimal digits of precision.
- H_floating-point data is 16 bytes long with a 15-bit excess 16,384 exponent. The effective 113-bit fraction yields approximately 33 decimal digits of precision.

Floating-point instructions are standard in the microcode of the MicroVAX and MicroPDP-11/73. Optional floating-point units, FPF11 and KEF11-AA, are available for the MicroPDP-11/23. These units implement 46 microcoded instructions that perform arithmetic, logical, and conversion operations, and operate six to ten times faster than equivalent software routines. For more information on the FPF11 and the KEF11-AA, refer to Chapter 2—System Hardware.

Character data are strings of bytes containing any binary data such as ASCII codes. Various standards, the most common of which is ASCII, assign an interpretation to some or all of the 256 different codes that can be represented by this data type.

The first character in the string is stored in the first byte, the second character is stored in the second byte, and so on in ascending order. An 8-bit character is stored at any addressable byte in memory or in the low-order byte of a general register.

A character-string is a sequence of up to 65,535 bytes in memory which can be located in two consecutive registers or two consecutive words in memory. The first word is the length of the character string (unsigned integer format), and the second word is the address of the most significant character (MSC). Subsequent characters through the least significant character (LSC) are stored in ascending memory locations.

Several kinds of decimal-string data formats are used in business applications where their correspondence to COBOL data types, keypunch codes, or printable characters is used. Decimal-string data formats are only available on the MicroPDP-11 systems. All represent numbers consisting of 0 to 31 decimal digits, with an implied decimal point to the right of the least significant digit (LSD). All are stored in memory as contiguous bytes.

There are separate instructions for packed-decimal string operations and zoned-numeric string operations, and for the decimal conversions between the two.

The address of any data item is the address of the first byte in which the item resides. All integer, floating-point, and character-string data can be stored starting at any address in memory. A bit field, however, does not necessarily start at a byte boundary in memory. A bit field is simply a set of contiguous bits between 0 and 32 bits in length. The starting bit location is identified relative to a given byte address or register. The instruction set can interpret a bit field as a signed or unsigned integer.

The MicroVAX processor also provides for two types of queue data. These consist of circular double-linked lists. A queue entry is specified by its address. Each queue entry is linked via a pair of longwords. The first longword is the forward link; it specifies the location of the succeeding entry. The second longword is the backward is the backward link; it specifies the location of the preceding entry.Two queue types are differentiated according to the nature of the links—absolute and self-relative. An absolute link contains the absolute address of the entry that it points to. A self-relative link contains a displacement from the address of the queue entry. Also, the instructions for use on a self-relative queue are interlocked.

▪ Instruction Sets

An instruction consists of an operation code (opcode) and zero or more operands that are described by a data type and addressing mode. The microsystem instruction sets are based on over 100 different kinds of operations, each addressable in several ways.

To choose the appropriate instruction, it is necessary only to become familiar with the operations, data types, and addressing modes. For example, the ADD operation can be applied to any of several sizes of integer or floating-point operands, and each operand can be addressed directly in a register, directly in memory, or indirectly through pointers stored in registers or memory locations.

MicroVAX Instruction Set

The MicroVAX instruction set is a subset of the VAX instruction set. It executes a large set of variable-length instructions, recognizes a variety of data types, and uses 16 32-bit general registers. The opcodes can be grouped into classes based on their function and use.

Instructions used to manipulate the MicroVAX data types include:

- Integer and logical instructions
- Floating-point instructions
- Character-string instructions
- Bit-field instructions
- Queue instructions
- Address-manipulation instructions
- General-register manipulation instructions

Instructions that provide basic program flow and enable the user to call procedures are:

- Branch, jump, and case instructions
- Subroutine call instructions
- Procedure call instructions
- Additional miscellaneous instructions

The section called "Microsystem Instruction Set Summary" lists these basic instruction operations.

MicroPDP-11 Instruction Set

The MicroPDP-11 also executes a large set of variable-length instructions, recognizes a variety of data types, and uses 8 16-bit registers. The following are the groupings of instructions:

- Load, store, and move instructions copy data between registers, memory, and I/O devices.
- Arithmetic instructions perform operations that interpret the numeric data, such as add, multiply, and negate.
- Shift and rotate instructions manipulate data within its original location.
- Data conversion instructions translate one data type to another.
- Logical instructions perform operations that manipulate bits and compare operands.
- Program control instructions redirect the flow of execution.
- Miscellaneous instructions include all of the remaining instructions.

Most instructions, other than floating-point and string-data instructions, use one of three basic formats—single operand, double operand, and branch. The single operand for instructions such as CLR (clear) and NEG (negate) is specified by the destination-address field, and the result is left in the same location. Instructions such as ADD and SUBtract use two operands specified by the source address and the destination address as input. These instructions leave the result at the destination address.

A user program can test the outcome of an arithmetic or logical operation. The processor provides a set of condition codes and branch instructions for this purpose. The condition codes indicate whether the previous arithmetic or logical operation produced a negative or zero result or whether there was a carry, borrow, or overflow. There is a variety of branch-on-condition instructions—those for overflow and carry or borrow, and those for signed and unsigned relational tests.

Microsystem Instruction Set Summary

Instruction	MicroVAX I	MicroPDP-11/73 MicroPDP-11/23
ABSD	—	Take absolute value of D_floating
ABSF	—	Take absolute value of F_floating
ACBB	Add, compare, and branch to byte	—
ACBF	Add, compare, and branch to F_floating	—
ACBG	Add, compare, and branch to G_floating	—
ACBL	Add, compare, and branch to longword	—
ACBW	Add, compare, and branch to word	—
ADAWI	Add aligned word interlocked	—
ADC	—	Add carry bit to word
ADCB	—	Add carry bit to byte
ADD	—	Add
ADDB2	Add byte 2-operand	—
ADDB3	Add byte 3-operand	—
ADDD	—	Add D_floating
ADDF2	Add F_floating 2-operand	—
ADDF3	Add F_floating 3-operand	—
ADDF	—	Add F_floating
ADDG2	Add G_floating 2-operand	—
ADDG3	Add G_floating 3-operand	—
ADDL2	Add longword 2-operand	—
ADDL3	Add longword 3-operand	—
ADDN(I)	—	Add numeric decimal strings
ADDP(I)	—	Add packed decimal strings
ADDW2	Add word 2-operand	—

Instruction	MicroVAX I	MicroPDP-11/73 MicroPDP-11/23
ADWC	Add with carry	—
AOBLEQ	Add one and branch less than or equal	—
AOBLSS	Add one and branch less than	—
ASH	—	Arithmetic shift register
ASHC	—	Arithmetic shift two combined registers
ASHL	Arithmetic shift longword	—
ASHN(I)	—	Arithmetic shift numeric decimal string
ASHP(I)	—	Arithmetic shift packed decimal string
ASHQ	Arithmetic shift quadword	—
ASL	—	Arithmetic shift word left
ASLB	—	Arithmetic shift byte left
ASR	—	Arithmetic shift word right
ASRB	—	Arithmetic shift byte right
BB	Branch on bit (set/clear)	—
BBC	Branch on bit clear and (set/clear) bit	—
BBCCI	Branch on bit clear and clear bit interlocked	—
BBS	Branch on bit set and (set/clear) bit	—
BBSSI	Branch on bit set and set bit interlocked	—
BCC	Branch if carry bit is clear	Branch if carry bit is clear
BCS	Branch if carry bit is set	Branch if carry bit is set
BEQ	—	Branch if equal to zero
BEQL	Branch if equal	—
BEQLU	Branch if equal unsigned	—

Instruction	MicroVAX I	MicroPDP-11/73 MicroPDP-11/23
BGE	—	Branch if greater than or equal to zero
BGEQ	Branch if greater than or equal to	—
BGEQU	Branch if greater than or equal to unsigned	—
BGT	—	Branch if greater than zero
BGTR	Branch if greater than	—
BGTRU	Branch if greater than unsigned	—
BHI	—	Branch if higher
BHIS	—	Branch if higher or same
BIC	—	Bit clear word
BICB2	Bit clear byte 2-operand	—
BICB3	Bit clear byte 3-operand	—
BICB	—	Bit clear byte
BICL2	Bit clear longword 2-operand	—
BICL3	Bit clear longword 3-operand	—
BICPSW	Bit clear processor status word	—
BICW2	Bit clear word 2-operand	—
BICW3	Bit clear word 3-operand	—
BIS	—	Bit set word
BISB2	Bit set byte 2-operand	—
BISB3	Bit set byte 3-operand	—
BISB	—	Bit set byte
BISF2	Bit set F_floating 2-operand	—
BISG2	Bit set G_floating 2-operand	—
BISL2	Bit set longword 2-operand	—
BISL3	Bit set longword 3-operand	—
BISPWS	Bit set processor status word	—
BISW2	Bit set word 2-operand	—

Instruction	MicroVAX I	MicroPDP-11/73 MicroPDP-11/23
BISW3	Bit set word 3-operand	—
BIT	—	Bit test word
BITB	Bit test byte	Bit test byte
BITL	Bit test longword	—
BITW	Bit test word	—
BLB	Branch on low bit (set/clear)	—
BLE	—	Branch if less than or equal to zero
BLEQ	Branch if less than or equal to	—
BLEQU	Branch if less than or equal unsigned	—
BLO	—	Branch if lower
BLOS	—	Branch if lower or same
BLSS	Branch if less than	—
BLSSU	Branch if less than unsigned	—
BLT	—	Branch if less than zero
BMI	—	Branch if minus
BNE	—	Branch if not equal to zero
BNEQ	Branch if not equal	—
BNEQU	Branch if not equal unsigned	—
BPL	—	Branch if plus
BPT	Breakpoint fault	Breakpoint trap
BR	—	Branch (unconditional)
BRB	Branch with byte displacement	—
BRW	Branch with word displacement	—
BSBB	Branch to subroutine with byte displacement	—
BSBW	Branch to subroutine with word displacement	—
BVC	Branch if overflow bit clear	Branch if overflow bit clear

Instruction	MicroVAX I	MicroPDP-11/73 MicroPDP-11/23
BVS	Branch if overflow bit set	Branch if overflow bit set
CALLG	Call procedure with general argument list	—
CALLS	Call procedure with stack argument list	—
CASEB	Case on byte	—
CASEL	Case on longword	—
CASEW	Case on word	—
CCC	—	Clear all condition codes
CFCC	—	Copy floating condition codes
CHM_	Change mode to kernel, executive, supervisor, or user	—
CLC	—	Clear carry condition code
CLN	—	Clear negative condition code
CLR	—	Clear word
CLRB	Clear byte	Clear byte
CLRD	Clear D_floating	Clear D_floating
CLRF	Clear F_floating	Clear F_floating
CLRG	Clear G_floating	—
CLRL	Clear longword	—
CLRQ	Clear quadword	—
CLRW	Clear word	—
CLV	—	Clear overflow condition code
CLZ	—	Clear zero condition code
CMP	—	Compare word
CMPB	Compare byte	Compare byte
CMPC3	Compare characters 3-operand	—
CMPC(I)	—	Compare character strings
CMPD	—	Compare D_floating
CMPF	Compare F_floating	Compare F_floating
CMPG	Compare G_floating	—

Instruction	MicroVAX I	MicroPDP-11/73 MicroPDP-11/23
CMPL	Compare longword	—
CMPN(I)	—	Compare numeric decimal strings
CMPP(I)	—	Compare packed decimal strings
CMPV	Compare field	—
CMPW	Compare word	—
CMPZV	Compare zero-extended field	—
COM	—	Take one's complement of word
COMB	—	Take one's complement of byte
CSM	—	Call supervisor mode
CVTBF	Convert byte to F_floating	—
CVTBG	Convert byte to G_floating	—
CVTBL	Convert byte to longword	—
CVTBW	Convert byte to word	—
CVTFB	Convert F_floating to byte	—
CVTFG	Convert F_floating to G_floating	—
CVTFL	Convert F_floating to longword	—
CVTFW	Convert F_floating to word	—
CVTGB	Convert G_floating to byte	—
CVTGF	Convert G_floating to F_floating	—
CVTGL	Convert G_floating to longword	—
CVTGW	Convert G_floating to word	—
CVTLB	Convert longword to byte	—
CVTLF	Convert longword to F_floating	—

Instruction	MicroVAX I	MicroPDP-11/73 MicroPDP-11/23
CVTLG	Convert longword to G_floating	—
CVTLN	—	Long integer to numeric decimal
CVTLP	—	Long integer to packed decimal
CVTLW	Convert longword to word	—
CVTNL	—	Numeric decimal to long integer
CVTNP	—	Numeric decimal to packed decimal
CVTPL	—	Packed decimal to long integer
CVTPN	—	Packed decimal to numeric decimal
CVTRFL	Convert rounded F_floating to longword	—
CVTRGL	Convert rounded G_floating to longword	—
CVTWB	Convert word to byte	—
CVTWF	Convert word to F_floating	—
CVTWG	Convert word to G_floating	—
CVTWL	Convert word to longword	—
DEC	—	Decrement word
DECB	Decrement byte	Decrement byte
DECL	Decrement longword	—
DECW	Decrement word	—
DIV	—	Divide
DIVB2	Divide byte 2-operand	—
DIVB3	Divide byte 3-operand	—
DIVD	—	Divide D_floating
DIVF2	Divide F_floating 2-operand	—
DIVF3	Divide F_floating 3-operand	—
DIVF	—	Divide F_floating

Instruction	MicroVAX I	MicroPDP-11/73 MicroPDP-11/23
DIVG2	Divide G_floating 2-operand	—
DIVG3	Divide G_floating 3-operand	—
DIVL2	Divide longword 2-operand	—
DIVL3	Divide longword 3-operand	—
DIVP(I)	—	Divide packed decimal strings
DIVW2	Divide word 2-operand	—
DIVW3	Divide word 3-operand	—
EDIV	Extended divide	—
EMODF	Extended modulus F_floating	—
EMODG	Extended modulus G_floating	—
EMT	—	Emulator trap
EMUL	Extended multiply	—
EXTV	Extract field	—
EXTZV	Extract zero-extended field	—
FFC	Find first clear	—
FFS	Find first set	—
HALT	Halt	Halt
INC	—	Increment word
INCB	Increment byte	Increment byte
INCL	Increment longword	—
INCW	Increment word	—
INDEX	Computer index	—
INSQUE	Insert entry in queue	—
INSQHI	Insert entry in queue at head interlocked	—
INSQTI	Insert entry in queue at tail interlocked	—
INSV	Insert field	—
IOT	—	Input/output trap
JMP	Jump	Jump

Instruction	**MicroVAX I**	**MicroPDP-11/73 MicroPDP-11/23**
JSB	Jump to subroutine	—
JSR	—	Jump to subroutine
L2D	—	Load two string descriptors
L3D	—	Load three string descriptors
LDCDF	—	Load and convert D_floating to F_floating
LDCFD	—	Load and convert F_floating to D_floating
LDCID	—	Load and convert integer to D_floating
LDCIF	—	Load and convert integer to F_floating
LDCLD	—	Load and convert long integer to D_floating
LDCLF	—	Load and convert long integer to F_floating
LDD	—	Load D_floating
LDEXP	—	Load exponent
LDF	—	Load F_floating
LDPCTX	Load process context	—
LOCC	Locate character	—
LOCC(I)	—	Locate character
MARK	—	Facilitates stack cleanup
MATC(I)	—	Match character string
MCOMB	Move complemented byte	—
MCOML	Move complemented longword	—
MCOMW	Move complemented word	—
MFPD	—	Move from previous data space
MFPI	—	Move from previous instruction space
MFPR	Move from processor register	—

Instruction	MicroVAX I	MicroPDP-11/73 MicroPDP-11/23
MFPS	—	Move from processor status word
MFPT	—	Move from processor type
MNEGB	Move negated byte	—
MNEGF	Move negated F_floating	—
MNEGG	Move negated G_floating	—
MNEGL	Move negated longword	—
MNEGW	Move negated word	—
MODD	—	Multiply and separate integer and D_floating
MODF	—	Multiply and separate integer and F_floating
MOV	—	Move word
MOVAF	Move byte, word, longword	—
MOVAG	Move quadword	—
MOVB	Move byte	Move byte
MOVC(I)	—	Move character string
MOVF	Move F_floating	—
MOVG	Move G_floating	—
MOVL	Move longword	—
MOVPSL	Move from processor status longword	—
MOVQ	Move quadword	—
MOVRC(I)	—	Move reverse-justified character string
MOVTC(I)	—	Move translated character string
MOVW	Move word	—
MOVZBL	Move zero-extended byte to longword	—
MOVZBW	Move zero-extended byte to word	—

Instruction	MicroVAX I	MicroPDP-11/73 MicroPDP-11/23
MOVZWL	Move zer-extended word to longword	—
MTPD	—	Move to previous data space
MTPI	—	Move to previous instruction space
MTPR	Move to processor register	—
MTPS	—	Move to processor status word
MUL	—	Multiply
MULB2	Multiply byte 2-operand	—
MULB3	Multiply byte 3-operand	—
MULD	—	Multiply D_floating
MULF	—	Multily F_floating
MULF2	Multiply F_floating 2-operand	—
MULF3	Multiply F_floating 3-operand	—
MULG2	Multiply G_floating 2-operand	—
MULG3	Multiply G_floating 3-operand	— —
MULL2	Multiply longword 2-operand	—
MULL3	Multiply longword 3-operand	—
MULP(I)	—	Multiply packed decimal strings
MULW2	Multiply word 2-operand	—
MULW3	Multiply word 3-operand	—
NEG	—	Negate (take 2's complement) of word
NEGB	—	Negate (take 2's complement) of byte
NEGD	—	Negate D_floating
NEGF	—	Negate F_floating
NOP	No operation	No operation

Instruction	MicroVAX I	MicroPDP-11/73 MicroPDP-11/23
POLYF	Polynomial evaluation F_floating	—
POLYG	Polynomial evaluation G_floating	—
POPR	Pop registers from stack	—
PROBER	Probe read	—
PROBEW	Probe write	—
PUSHAF	Push address byte, longword, or word on stack	—
PUSHAG	Push address quadword on stack	—
PUSHL	Push longword on stack	—
PUSHR	Push registers on stack	—
REI	Return from exception or interrupt	—
REMQHI	Remove entry from queue at head interlocked	—
REMQUE	Remove entry from queue	—
REMQTI	Remove entry from queue at tail interlocked	—
RESET	—	Reset bus
RET	Return from procedure	—
ROL	—	Rotate register word left
ROLB	—	Rotate register byte left
ROR	—	Rotate register word right
RORB	—	Rotate register word right
ROTL	Rotate longword	—
RSB	Return from subroutine	—
RTI	—	Return from interrupts
RTS	—	Return from subroutine
RTT	—	Return from interrupts

Instruction	**MicroVAX I**	**MicroPDP-11/73 MicroPDP-11/23**
SBC	—	Subtract carry bit from word
SBCB	—	Subtract carry bit from byte
SBWC	Subtract with carry	—
SCANC	Scan characters	—
SCANC(I)	—	Scan character string
SCC	—	Set all condition code bits
SEC	—	Set carry condition code
SEN	—	Set negative condition code
SETD	—	Set D_floating mode
SETF	—	Set F_floating mode
SETI	—	Set floating for integer mode
SETL	—	Set floating for long integer mode
SEV	—	Set overflow condition code
SEZ	—	Set zero condition code
SKPC	Skip character	—
SKPC(I)	—	Skip character string
SOBGEQ	Subtract one and branch greater than or equal to	—
SOBGTR	Subtract one and branch greater than	—
SPANC	Span characters	—
SPANC(I)	—	Span character string
STCDF	—	Store and convert D_floating to F_floating
STCDI	—	Store and convert D_floating to integer
STCDL	—	Store and convert D_floating to long integer
STCFD	—	Store and convert F_floating to D_floating

Instruction	MicroVAX I	MicroPDP-11/73 MicroPDP-11/23
STCFI	—	Store and convert F_floating to integer
STCFL	—	Store and convert F_floating to long integer
STD	—	Store D_floating
STEXP	—	Store exponent
STF	—	Store F_floating
STFPS	—	Store floating-point program status word
STST	—	Store floating-point status
SUB	—	Subtract word
SUBB2	Subtract byte 2-operand	—
SUBB3	Subtract byte 3-operand	—
SUBD	—	Subtract D_floating
SUBF2	Subtract F_floating 2-operand	—
SUBF3	Subtract F_floating 3-operand	—
SUBF	—	Subtract F_floating
SUBG2	Subtract G_floating 2-operand	—
SUBG3	Subtract G_floating 3-operand	—
SUBL2	Subtract longword 2-operand	—
SUBL3	Subtract longword 3-operand	—
SUBN(I)	—	Subtract numeric decimal strings
SUBP(I)	—	Subtract packed decimal strings
SUBW2	Subtract word 2-operand	—
SUBW3	Subtract word 3-operand	—
SVPCTX	Save process context	—
SWAB	—	Swap bytes in word
SXT	—	Sign extend
TRAP	—	Trap

Instruction	**MicroVAX I**	**MicroPDP-11/73 MicroPDP-11/23**
TST	—	Test word
TSTB	Test byte	Test byte
TSTD	—	Test D_floating
TSTF	Test F_floating	Test F_floating
TSTG	Test G_floating	—
TSTSET	—	Test word, set low bit
TSTW	Test word	—
WAIT	—	Wait for interrupt
WRTLCK	—	Read/lock destination, write/ unlock RO
XFC	Extended function call	—
XOR	—	Exclusive OR word
XORB2	Exclusive OR byte 2-operand	—
XORB3	Exclusive OR byte 3-operand	—
XORL2	Exclusive OR longword 2-operand	—
XORL3	Exclusive OR longword 3-operand	—
XORW2	Exclusive OR word 2-operand	—
XORW3	Exclusive OR word 3-operand	—

▪ Additional Documentation

VAX Architecture Handbook	*EB-19580-20*
PDP-11 Architecture Handbook	*EB-23657-18*
MicroVAX I CPU Technical Description	*EK-KD32A-TD*
KDJ11-A CPU Module User's Guide	*EK-KDJ1A-UG*
KDF11-BA CPU Module User's Guide	*EK-KDFEB-UG*

Appendix A ▪ Q-bus

▪ Introduction

The Q-bus is the common communications path for the data, address, and control information that is transferred between the CPU, memory, and device interfaces. Each of the microsystems use only the Q-bus for these communications.

The 22-bit Q-bus consists of 42 bidirectional and two unidirectional signal lines that are built into the backplane assembly. Logic modules are installed in the backplane and connected to these signal lines with backplane connectors. The signal lines are defined as follows:

- Sixteen multiplexed data/address lines (BDAL< 15:00 >)
- Two multiplexed address/parity lines (BDAL< 17:16 >)
- Four nonmultiplexed extended address lines (BDAL< 21:18 >)
- Six data transfer control lines (BBS7, BDIN, BDOUT, BRPLY, BSYNC, BWTBT)
- Six system control lines (BHALT, BREF, BEVNT, BINIT, BDCOK, BPOK)
- Ten interrupt control and direct memory access control lines (BIAKO, BIAKI, BIRQ4, BIRQ5, BIRQ6, BIRQ7, BDMGO, BDMR, BSACK, BDMGI)

In addition to the data, address, and control signal lines, a number of power, ground, and spare lines have been defined for the 22-bit Q-bus (hereafter referred to as the Q22 bus). For a detailed description of these lines, refer to Table A-1.

▪ Bus Communications

All communications on the bus are performed asynchronously to allow some devices to transfer at data rates greater than those of other devices. The bus operates with a *master and slave relationship.* When more than one device requests the use of the bus, the device with the highest priority gains access. It becomes the bus master and controls the data transfers until it releases the bus. In performing the transfers, it addresses another device that is designated as a slave.

The current data cycle is overlapped with the arbitration for the next cycle, enhancing the system performance. The upper eight Kbytes of address space are reserved for I/O devices. Some of the addresses are fixed within this space, and others are allowed to float, depending on the system configuration.

The bus transactions consist of initialization, arbitration, data transmission, and miscellaneous:

- The *initialization* lines of the bus provide the information required to start the processor after powerup and cause an orderly shutdown of the processor during power failures. In addition, they allow the processor to reset the I/O subsystem.
- The *arbitration* lines control access to the data transmission portion of the bus.
- The *data transmission* lines allow words or bytes to be moved about on the bus. Transmission of data is always accomplished with one device acting as master and the other acting as slave. The master controls the direction and length of transmission.
- The *miscellaneous* lines provide other functions, including processor control and memory refresh.

Master/Slave Relationship

A master/slave relationship exists throughout each bus transaction. At any time, one device has control of the bus and is termed the bus master, and the other device is termed the slave. The bus master, which is typically the processor or a direct-memory access (DMA) device, initiates a bus transaction. The slave device responds by acknowledging the transaction in progress and by receiving data from, or transmitting data to, the bus master. The bus control signals transmitted or received by the bus master or the bus slave device must complete the sequence according to bus protocol.

The processor controls bus arbitration to determine which device becomes bus master at any given time. A typical example of this relationship is the processor, as master, fetching an instruction from memory, which is always a slave. Another example is a disk-drive device as master transferring data to memory as slave. Communications on the bus are interlocked so that, for certain control signals issued by the master device, there must be a response from the slave in order to complete the transfer. It is the master/slave signal protocol that precludes the need for synchronizing clock pulses.

Since bus-cycle completion by the bus master requires response from the slave device, each bus master includes a time-out error circuit that will abort the bus cycle if the slave device does not respond to the bus transaction within 10 microseconds.

Bus Signals and Pin Assignments

The connectors on the backplane assembly and the logic modules have corresponding pin and signal assignments. Table A-1 lists the signal and pin designations for the data/address, control, power/ground, and spare lines. All Q22-bus signals are asserted low and negated high, except BPOK and BDCOK, which are asserted high. Most signals are bidirectional, and transactions on the bus are performed asynchronously so that devices can exchange data at their own rates and the same interfaces can be used for different devices. The data and address lines are time-multiplexed.

Table A-1 ▪ Bus Signals and Pin Assignments

Data and Address

Signal	Pin	Signal	Pin
BDAL1	AV2	BDAL12	BS2
BDAL2	BE2	BDAL13	BT2
BDAL3	BF2	BDAL14	BU2
BDAL4	BH2	BDAL15	BV2
BDAL5	BJ2	BDAL16	AC1
BDAL6	BK2	BDAL17	AD1
BDAL7	BL2	BDAL18	AD1
BDAL8	BM2	BDAL19	BD1
BDAL9	BN2	BDAL20	BE1
BDAL10	BP2	BDAL21	BF1
BDAL11	BR2	BDAL14	BU2

Data Transfer Control

Signal	Pin	Signal	Pin
BDOUT	AE2	BSYNC	AJ2
BRPLY	AF2	BWTBT	AK2
BDIN	AH2	BBS7	AP2

Interrupt Control

Signal	Pin	Signal	Pin
BIRQ7	BP1	BIRQ4	AL2
BIRQ6	AB1	BIAKO	AN2
BIRQ5	AA1	BIAKI	AM2

Direct Memory Access Control				
Signal	Pin		Signal	Pin
BDMR		AN1	BDMGO	AS2
BSACK	BN1		BMDGI	AR2
System Control				
Signal	Pin		Signal	Pin
BHALT	AP1		BINIT	AT2
BREF	AR1		BDCOK	BA1
BEVNT	BR1		BPOK	BB1
Power and Ground				
Voltage	Pin		Voltage	Pin
+5B (battery)	AS1		−12	BB2
+12B (battery)	AS1	(not supplied	GND	AC2
+12B	BS1	by Digital)	GND	AJ1
+5B	AV1		GND	AM1
+5	AA2		GND	AT1
+5	BA2		GND	BC2
+5	BV1		GND	BJ1
+12	AD2		GND	BM1
+12	BD2		GND	BT1
Spares				
Signal	Pin		Signal	Pin
SSpare1	AE1		MSpareB	AL1
SSpare3	AH1		MSpareB	BL1
SSpare8	BH1		PSpare1	AU1
SSpare2	AF1		ASpare2	BU1
MSpareA	AK1			

▪ Data Transfer Bus Cycles

Seven types of data transfer operations can occur and are listed in Table A-2. During the bus cycle, executed by the bus master, 8-bit bytes or 16-bit words can be written or read. In block mode, multiple words can be transfer d to or from sequential word addresses, starting from a single bus address. The bus signals used for the data transfer operations are listed in Table A-3.

Table A-2 ▪ Data Transfer Operations

Mnemonic	Description	Bus Master Function
DATI	Data word input	Read
DATO	Data word output	Write
DATOB	Data byte output	Write byte
DATIO	Data word input/output	Read-modify-write
DATIOB	Data word input/byte output	Read-modify-write byte
DATBI	Data block input	Read block
DATBO	Data block output	Write block

Table A-3 ▪ Bus Signals for Data Transfers

Mnemonic	Description	Function
BDAL<21:00> L	22 data/address lines	BDAL<15:00> L are used for word and byte transfers. BDAL<17:16> L are used for extended addressing, memory parity error (16), and memory parity error enable (17) functions. BDAL<21:18> L are used for extended addressing beyond 256 Kbytes.
BSYNC L	Bus-cycle control	Indicates bus transaction in progress.
BDIN L	Data-input indicator	Indicates bus transaction in progress.
BDIN L	Data-input indicator	Strobe signal
BRPLY L	Slave's acknowledgment of bus cycle	Strobe signal
BWTBT L	Write/byte control	Control signal
BBS7 L	I/O device select	Indicates address is in the I/O page.

Data transfer bus cycles can be reduced to five basic types: DATI, DATO(B), DATIO(B), DATBI, and DATBO. These transactions occur between the bus master and one slave device selected during the addressing portion of the bus cycle.

Bus-cycle Protocol

Before initiating a bus cycle, the previous bus transaction must have been completed (BSYNC L negated) and the device must become bus master. The bus cycle can be divided into an addressing portion and a data transfer portion. During the addressing portion, the bus master outputs the address for the desired slave device, memory location, or device register. The selected slave device responds by latching onto the address bits and holding this condition for the duration of the bus cycle until BSYNC L becomes negated. During the data transfer portion, the actual data transfer occurs.

Device Addressing

The device-addressing portion of a data transfer bus cycle comprises an address setup and deskew time and an address hold and deskew time. During the address setup and deskew time, the bus master performs the following:

- Asserts BDAL<21:00> L with the desired slave-device address bits
- Asserts BBS7 L if a device in the I/O page is being addressed
- Asserts BWTBT L if the cycle is a DATO(B) or DATBO

During this time, the received address BBS7 L and BWTBT L signals are asserted at the slave-bus receiver for at least 75 nanoseconds before BSYNC goes active. Devices in the I/O page ignore the nine high-order address bits BDAL<21:13> and instead decode BBS7 L along with the 13 low-order address bits. An active BWTBT L signal during address setup time indicates that a DATO(B) or DATBO operation will follow, while an inactive BWTBT L indicates a DATI, DATBI, or DATIO(B) operation will follow.

The address hold and deskew time begins after BSYNC L is asserted. The slave device uses the active BSYNC L bus receiver output to clock BDAL address bits, BBS7 L, and BWTBT L into its internal logic. BDAL<21:00> L, BBS7 L, and BWTBT L will remain active for 25 nanoseconds minimum after BSYNC L bus receiver goes active. BSYNC L remains active for the duration of the bus cycle. Memory and peripheral devices are addressed similarly except for the way the slave device responds to BBS7 L. Addressed peripheral devices do not decode address bits on BDAL<21:13> L. Addressed peripheral devices may respond to a bus cycle when BBS7 L is asserted (low) during the addressing portion of the cycle. When asserted, BBS7 L indicates that the device address resides in the I/O page (the upper 4000 bytes of address space). Memory devices generally do not respond to addresses in the I/O page; however, some system applications may permit memory to reside in the I/O page for use as DMA buffers, read-only memory bootstraps, or diagnostics.

DATI Bus Cycle

The DATI bus cycle, shown in Figure A-1, is a read operation. During DATI, data is input to the bus master. Data consists of 16-bit word transfers over the bus. During the data transfer portion of the DATI bus cycle, the bus master asserts BDIN L for 100 nanoseconds minimum after BSYNC L is asserted. Figure A-2 shows the DATI bus cycle timing. The slave device responds to BDIN L active as follows:

- Asserts BRPLY L for 0 nanoseconds minimum (8 microseconds maximum to avoid bus timeout) after receiving BDIN L for 125 nanoseconds maximum before BDAL bus driver data bits are valid.
- Asserts BDAL< 17:00 > L on the MicroVAX I and BDAL< 21:00 > L on the MicroPDP-11 with the addressed data and error information for 0 nanoseconds minimum after receiving BDIN and 125 nanoseconds maximum after assertion of BRPLY.

When the bus master receives BRPLY L, it does the following:

- Waits at least 200 nanoseconds deskew time and then accepts input data at BDAL< 17:00 > L bus receivers. BDAL< 17:16 > L are used for transmitting parity errors to the master.
- Negates BDIN L for 200 nanoseconds minimum to 2 microseconds maximum after BRPLY L goes active.

The slave device responds to BDIN L negation by negating BRPLY L and removing read data from BDAL bus drivers. BRPLY L must be negated 100 nanoseconds maximum prior to removal of read data. The bus master responds to the negated BRPLY L by negating BSYNC L.

Conditions for the next BSYNC L assertion are as follows:

- BSYNC L must remain negated for 200 nanoseconds minimum.
- BSYNC L must not become asserted within 300 nanoseconds of previous BRPLY L negation.

NOTE

Continuous assertion of BSYNC L retains control of the bus by the bus master, and the previously addressed slave device remains selected. This is done for DATIO(B) bus cycles where DATO or DATOB follows a DATI without BSYNC L negation and a second device addressing operation. Also, a slow slave device can hold off data transfers to itself by keeping BRPLY L asserted, which will cause the master to keep BSYNC L asserted.

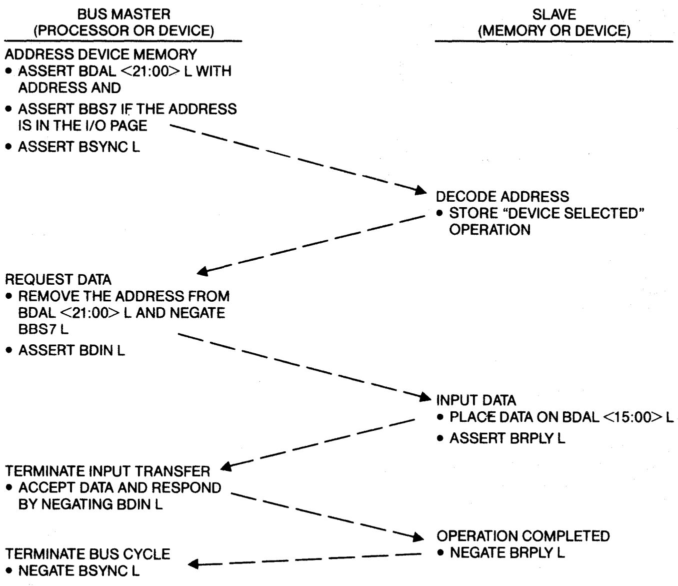
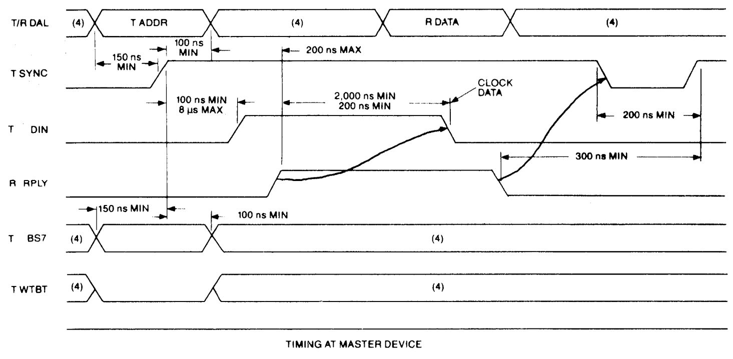
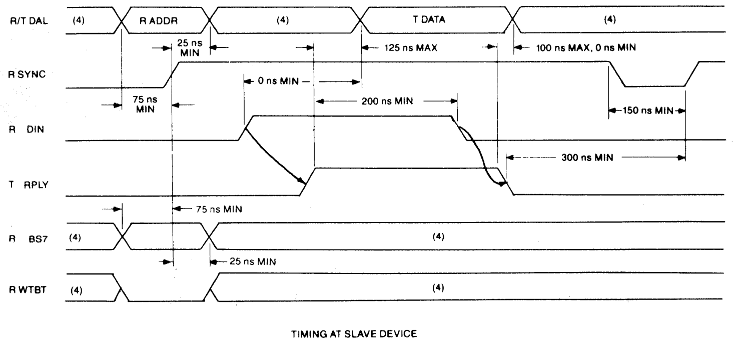

BUS MASTER
(PROCESSOR OR DEVICE)
SLAVE
(MEMORY OR DEVICE)
ADDRESS DEVICE MEMORY
• ASSERT BDAL <21:00> L WITH ADDRESS AND
• ASSERT BBS7 IF THE ADDRESS IS IN THE I/O PAGE
• ASSERT BSYNC L
DECODE ADDRESS
• STORE "DEVICE SELECTED" OPERATION
REQUEST DATA
• REMOVE THE ADDRESS FROM BDAL <21:00> L AND NEGATE BBS7 L
• ASSERT BDIN L
INPUT DATA
• PLACE DATA ON BDAL <15:00> L
• ASSERT BRPLY L
TERMINATE INPUT TRANSFER
• ACCEPT DATA AND RESPOND BY NEGATING BDIN L
OPERATION COMPLETED
• NEGATE BRPLY L
TERMINATE BUS CYCLE
• NEGATE BSYNC L

Figure A-1 ▪ DATI Bus Cycle





NOTES

1 Timing shown at Master and Slave Device Bus Driver inputs and Bus Receiver Outputs

2 Signal name prefixes are defined below:

T = Bus Driver input
R = Bus Receiver Output

3 Bus Driver Output and Bus Receiver input signal names include a "B" prefix

4 Don't care condition

Figure A-2 ▪ DATI Bus-cycle Timing

DATO(B) Bus Cycle

The DATO(B) bus cycle, shown in Figure A-3, is a write operation. Data is transferred in 16-bit words (DATO) or 8-bit bytes (DATOB) from the bus master to the slave device. The data transfer output can occur after the addressing portion of a bus cycle when BWTBT L has been asserted by the bus master, or immediately following an input transfer part of a DATIO(B) bus cycle.

The data transfer portion of a DATO(B) bus cycle comprises a data setup and deskew time and a data hold and deskew time.

During the data setup and deskew time, the bus master outputs the data on BDAL < 15:00 > L at least 100 nanoseconds after BSYNC L is asserted. If the transfer is a word transfer, the bus master negates BWTBT L at least 100 nanoseconds after BSYNC L assertion. BWTBT L remains negated for the length of the bus cycle. If the transfer is a byte transfer, BWTBT L remains asserted. If the transfer is the output of a DATIOB, BTWBT L becomes asserted and lasts the duration of the bus cycle.

During a byte transfer, BDAL < 00 > L selects the high or low byte. This occurs while in the addressing portion of the cycle. If asserted, the high byte BDAL < 15:08 > L is selected; otherwise, the low byte BDAL < 07:00 > L is selected. An asserted BDAL16 L at this time will force a parity error to be written into memory if the memory is a parity-type memory. BDAL17 L is not used for write operations. The bus master asserts BDOUT L at least 100 nanoseconds after BDAL and BWTBT L bus drivers are stable. The slave device responds by asserting BRPLY L within 10 microseconds to avoid bus timeout. This completes the data setup and deskew time.

During the data hold and deskew time the bus master receives BRPLY L and negates BDOUT L. BDOUT L must remain asserted for at least 150 nanoseconds from the receipt of BRPLY L before being negated by the bus master. BDAL < 17:00 > L bus drivers remain asserted for at least 100 nanoseconds after BDOUT L negation. The bus master then negates BDAL inputs.

During this time, the slave device senses BDOUT L negation. The data are accepted, and the slave device negates BRPLY L. The bus master responds by negating BSYNC L. However, the processor will not negate BSYNC L for at least 175 nanoseconds after negating BDOUT L. This completes the DATO(B) bus cycle. Before the next cycle, BSYNC L must remain unasserted for at least 200 nanoseconds. Figure A-4 shows DATO(B) bus-cycle timing.

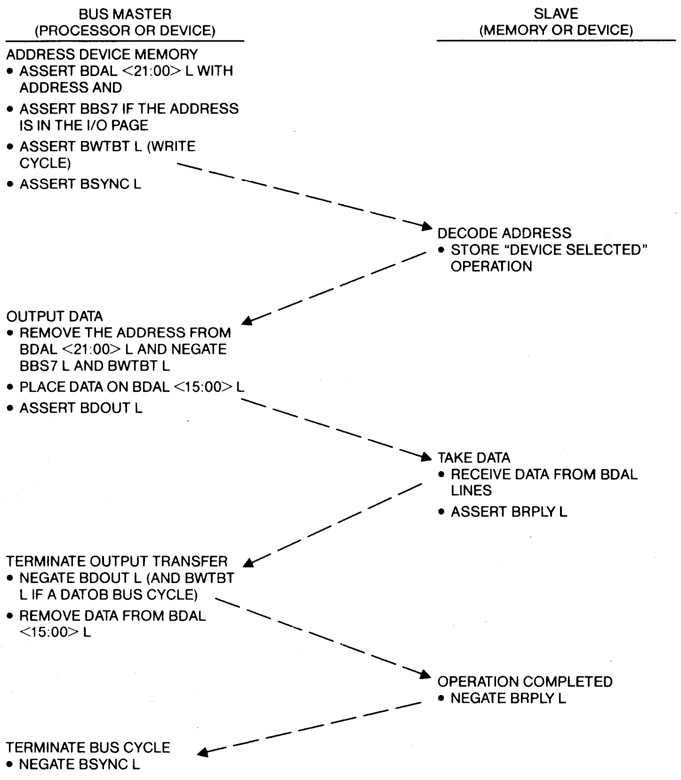
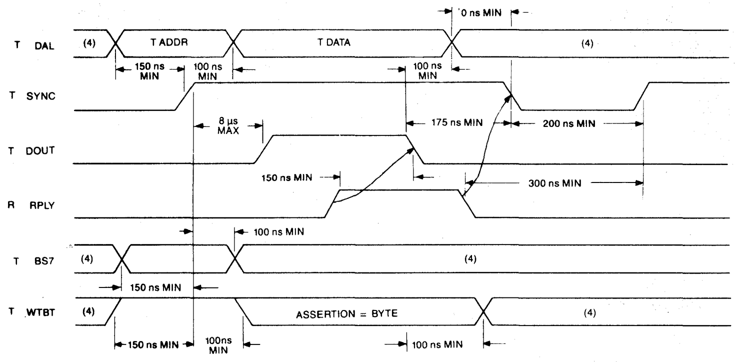
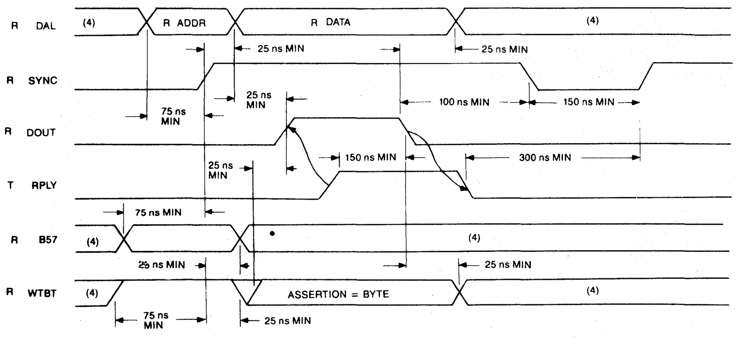

BUS MASTER
(PROCESSOR OR DEVICE)
SLAVE
(MEMORY OR DEVICE)
ADDRESS DEVICE MEMORY
• ASSERT BDAL <21:00> L WITH ADDRESS AND
• ASSERT BBS7 IF THE ADDRESS IS IN THE I/O PAGE
• ASSERT BWTBT L (WRITE CYCLE)
• ASSERT BSYNC L
DECODE ADDRESS
• STORE "DEVICE SELECTED" OPERATION
OUTPUT DATA
• REMOVE THE ADDRESS FROM BDAL <21:00> L AND NEGATE BBS7 L AND BWTBT L
• PLACE DATA ON BDAL <15:00> L
• ASSERT BDOUT L
TAKE DATA
• RECEIVE DATA FROM BDAL LINES
• ASSERT BRPLY L
TERMINATE OUTPUT TRANSFER
• NEGATE BDOUT L (AND BWTBT L IF A DATOB BUS CYCLE)
• REMOVE DATA FROM BDAL <15:00> L
OPERATION COMPLETED
• NEGATE BRPLY L
TERMINATE BUS CYCLE
• NEGATE BSYNC L

Figure A-3 ▪ DATO or DATOB Bus Cycle



TIMING AT MASTER DEVICE



TIMING AT SLAVE DEVICE

NOTES
1 Timing shown at Master and Slave Device Bus Driver Inputs and Bus Receiver Outputs

2 Signal name prefixes are defined below

T = Bus Driver Input
R = Bus Receiver Output

3 Bus Driver Output and Bus Receiver Input signal names include a "B" prefix

4 Don't care condition

Figure A-4 ▪ DATO or DATOB Bus-cycle Timing

DATIO(B) Bus Cycle

The protocol for a DATIO(B) bus cycle is identical to the addressing and data transfer portions of the DATI and DATO(B) bus cycles; it is illustrated in Figure A-5. After the bus addresses the device, a DATI cycle is performed; however, BSYNC L is not negated. BSYNC L remains active for an output word or byte transfer DATO(B). The bus master maintains at least 200 nanoseconds between BRPLY L negation during the DATI cycle and BDOUT L assertion. The cycle is terminated when the bus master negates BSYNC L, as described for DATO(B). Figure A-6 shows DATIO(B) bus-cycle timing.

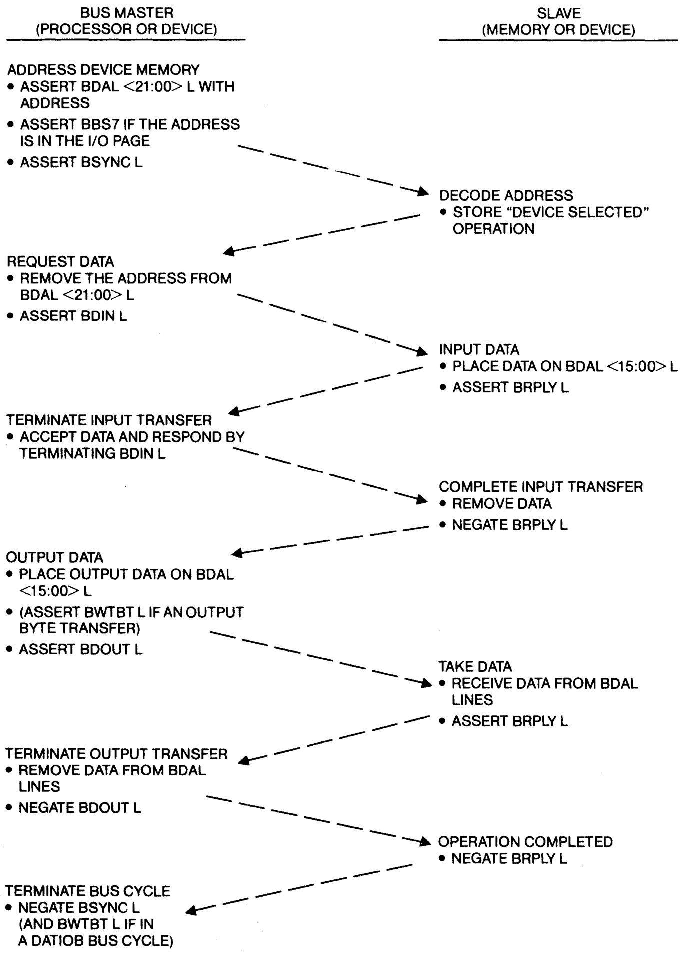
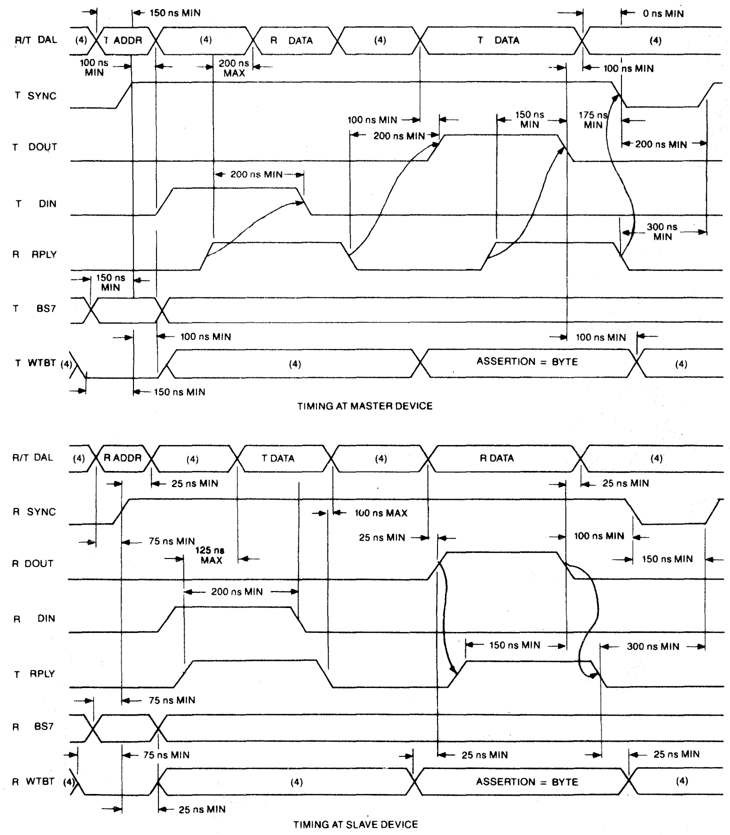

BUS MASTER (PROCESSOR OR DEVICE)
SLAVE (MEMORY OR DEVICE)
ADDRESS DEVICE MEMORY
• ASSERT BDAL <21:00> L WITH ADDRESS
• ASSERT BBS7 IF THE ADDRESS IS IN THE I/O PAGE
• ASSERT BSYNC L
DECODE ADDRESS
• STORE "DEVICE SELECTED" OPERATION
REQUEST DATA
• REMOVE THE ADDRESS FROM BDAL <21:00> L
• ASSERT BDIN L
INPUT DATA
• PLACE DATA ON BDAL <15:00> L
• ASSERT BRPLY L
TERMINATE INPUT TRANSFER
• ACCEPT DATA AND RESPOND BY TERMINATING BDIN L
COMPLETE INPUT TRANSFER
• REMOVE DATA
• NEGATE BRPLY L
OUTPUT DATA
• PLACE OUTPUT DATA ON BDAL <15:00> L
• (ASSERT BWTBT L IF AN OUTPUT BYTE TRANSFER)
• ASSERT BDOUT L
TAKE DATA
• RECEIVE DATA FROM BDAL LINES
• ASSERT BRPLY L
TERMINATE OUTPUT TRANSFER
• REMOVE DATA FROM BDAL LINES
• NEGATE BDOUT L
OPERATION COMPLETED
• NEGATE BRPLY L
TERMINATE BUS CYCLE
• NEGATE BSYNC L (AND BWTBT L IF IN A DATIOB BUS CYCLE)

Figure A-5 ▪ DATIO or DATIOB Bus Cycle



NOTES

1 Timing shown at Requesting Device Bus Driver inputs and Bus Receiver Outputs

2 Signal name prefixes are defined below

T = Bus Driver Input
R = Bus Receiver Output

3 Bus Driver Output and Bus Receiver Input signal names include a "B" prefix

4 Don't care condition

Figure A-6 ▪ DATIO or DATIOB Bus-cycle Timing

Direct Memory Access

The direct memory access (DMA) capability allows direct data transfer between I/O devices and memory. This is useful when using mass storage devices such as disk drives that move large blocks of data to and from memory. A DMA device needs only the starting address in memory, the starting address in mass storage, the length of the transfer, and whether the operation is read or write. When this information is available, the DMA device can transfer data directly to or from memory. Since most DMA devices must perform data transfers in rapid succession or else lose data, they are provided the highest priority.

DMA transfer is accomplished after the processor (normally bus master) has passed bus mastership to the highest-priority DMA device that is requesting the bus. The processor arbitrates all requests and grants the bus to the DMA device located electrically closest to it. A DMA device remains bus master indefinitely until it relinquishes its mastership.

The following control signals are used during bus arbitration:

BDMGI L	DMA grant input
BDGMO L	DMA grant output
BDMR L	DMA request line
BSACK L	Bus grant acknowledge

DMA Protocol

A DMA transaction can be divided into three phases:

- *Bus mastership acquisition* phase
- *Data transfer* phase
- *Bus mastership relinquish* phase

During the *bus mastership acquisition* phase, a DMA device requests the bus by asserting BDMR L. The processor arbitrates the request and initiates the transfer of bus mastership by asserting BDMGO L. The maximum time between BDMR L and BDMGO L assertion is DMA latency. This time is processor-dependent. BDMGO L/BDMGI L is one signal that is daisychained through each module in the backplane. It is driven out of the processor on the BDMGO L pin, enters each module on the BDMGI L pin, and exits on the BDMGO L pin. This signal passes through the modules in descending order of priority until it is stopped by the requesting device. The requesting device blocks the output of BDMGO L and asserts BSACK L. If BDMR L is continuously asserted, the bus will be jammed.

During the *data transfer* phase, the DMA device continues asserting BSACK L. The actual data transfer is performed as described earlier.

The DMA device can assert BSYNC L for a data transfer 250 nanoseconds minimum after it receives BDMGI L and its BSYNC L bus receiver becomes negated.

During the *bus mastership relinquish* phase, the DMA device relinquishes the bus by negating BSACK L. This occurs after completing (or aborting) the last data transfer cycle (BRPLY L negated). BSACK L can be negated up to a maximum of 300 nanoseconds before negating BSYNC L. Figure A-7 shows the DMA protocol and Figure A-8 shows DMA request/grant timing.

NOTE

If multiple data transfers are performed during this phase, considerations must be given to the use of the bus for other system functions, such as memory refresh (if required).

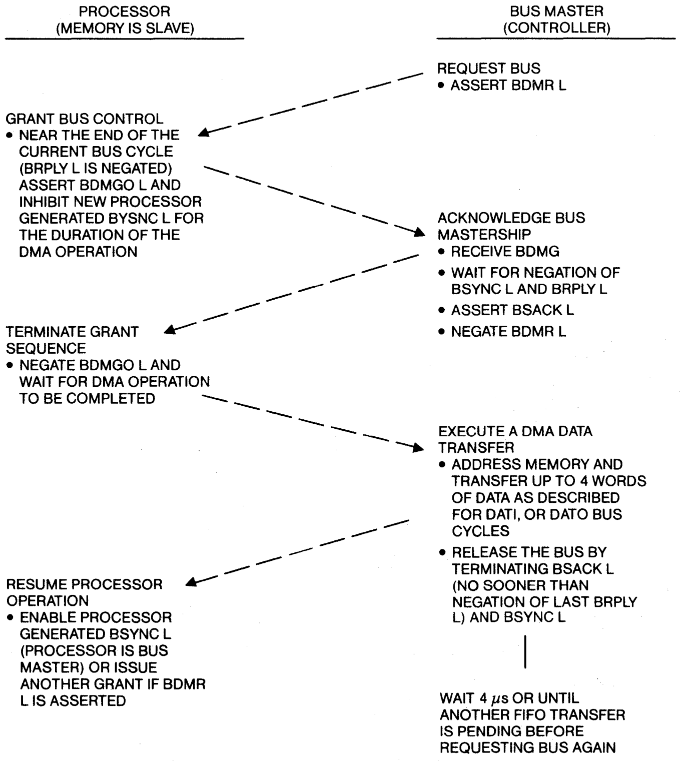
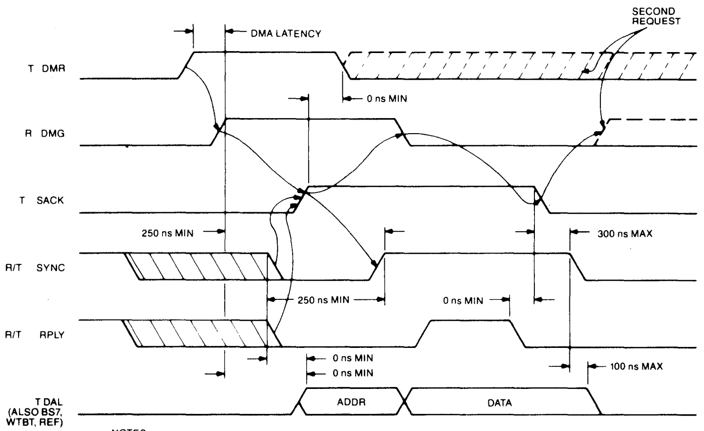

PROCESSOR (MEMORY IS SLAVE)
BUS MASTER (CONTROLLER)
REQUEST BUS
• ASSERT BDMR L
GRANT BUS CONTROL
• NEAR THE END OF THE CURRENT BUS CYCLE (BRPLY L IS NEGATED) ASSERT BDMGO L AND INHIBIT NEW PROCESSOR GENERATED BYSNC L FOR THE DURATION OF THE DMA OPERATION
ACKNOWLEDGE BUS MASTERSHIP
• RECEIVE BDMG
• WAIT FOR NEGATION OF BSYNC L AND BRPLY L
• ASSERT BSACK L
• NEGATE BDMR L
TERMINATE GRANT SEQUENCE
• NEGATE BDMGO L AND WAIT FOR DMA OPERATION TO BE COMPLETED
EXECUTE A DMA DATA TRANSFER
• ADDRESS MEMORY AND TRANSFER UP TO 4 WORDS OF DATA AS DESCRIBED FOR DATI, OR DATO BUS CYCLES
• RELEASE THE BUS BY TERMINATING BSACK L (NO SOONER THAN NEGATION OF LAST BRPLY L) AND BSYNC L
RESUME PROCESSOR OPERATION
• ENABLE PROCESSOR GENERATED BSYNC L (PROCESSOR IS BUS MASTER) OR ISSUE ANOTHER GRANT IF BDMR L IS ASSERTED
WAIT 4 μs OR UNTIL ANOTHER FIFO TRANSFER IS PENDING BEFORE REQUESTING BUS AGAIN

Figure A-7 ▪ DMA Protocol



NOTES
1 Timing shown at Requesting Device Bus Driver Inputs and Bus Receiver Outputs
2 Signal name prefixes are defined below
T = Bus Driver Input
R = Bus Receiver Output
3 Bus Driver Output and Bus Receiver Input signal names include a "B" prefix

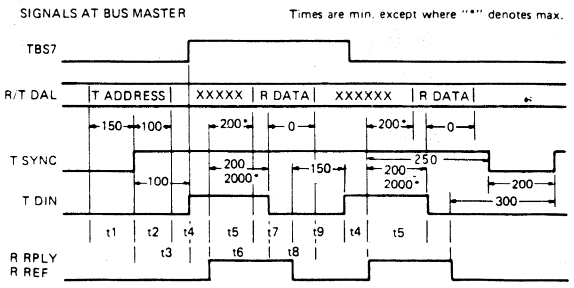
Figure A-8 ▪ DMA Request/Grant Timing

Block Mode

For increased throughput, block mode can be implemented on a device. In a block-mode transaction, the starting memory address is asserted, followed by data for that address and data for consecutive addresses. By eliminating the assertion of the address for each data word, the transfer rate is almost doubled.

DATBI Bus Cycle

The device-addressing portion of the DATBI bus cycle is the same as previously described for other bus cycles. The bus master gates BDAL <21:00>, BBS7, and the negation of BWTBT onto the bus. Figure A-9 shows the DATBI bus-cycle timing.



t1	=	address to T SYNC 150ns MIN.	
t2	=	address hold	100ns min
t3	=	T SYNC to T DIN	100ns min
t4	=	T DIN to R RPLY T (drive) + T (prop) + T (receive) + T (delay) + T (drive) + T (prop) + T (receive)	
t5	=	R RPLY to data	200ns max
t6	=	R RPLY to T DIN	200ns min
t7	= =	T DIN to R RPLY T (drive) + (prop) + T (receive) + T (delay) + T (drive + T (prop) + T (receive)	
t8	=	R RPLY to data	0ns min
t9	=	R RPLY to T DIN	150ns min
T cell	=	t4 + t6 + t7 + t9	–since t6 must be > t5 for master to have valid data · and t9 >t8

Figure A-9 ▪ DATBI Bus-cycle Timing

The master asserts the first BDIN 100 nanoseconds after BSYNC and asserts BBS7 a maximum of 50 nanoseconds after asserting BDIN for the first time. BBS7 is a request to the slave for a block-mode transfer. BBS7 remains asserted until a maximum of 50 nanoseconds after the assertion of BDIN for the last time. BBS7 can be gated as soon as the conditions for asserting BDIN are met.

The slave asserts BRPLY a minimum of 0 nanoseconds (8 milliseconds maximum to avoid bus timeout) after receiving BDIN. It asserts BREF concurrently with BRPLY if it is a block-mode device capable of supporting another BDIN after the current one. The slave gates BDAL < 15:00 > onto the bus a minimum of 0 nanoseconds after the assertion of BDIN and 125 nanoseconds maximum after the assertion of BRPLY.

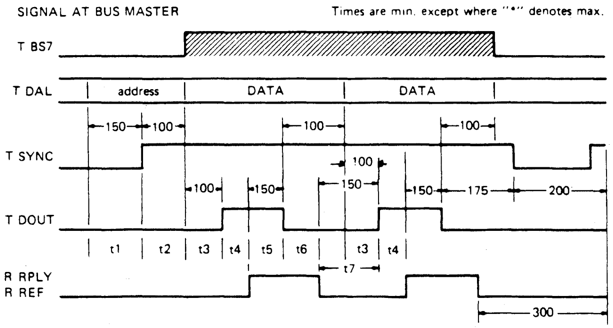
The master receives the stable data from 200 nanoseconds maximum after the assertion of BRPLY until 20 nanoseconds minimum after the negation of BDIN. It negates BDIN a minimum of 200 nanoseconds after the assertion of BRPLY.

The slave negates BRPLY a minimum of 0 nanoseconds after the negation of BDIN. If BBS7 and BREF are both asserted when BRPLY is negated, the slave prepares for another BDIN cycle. BBS7 is stable from 125 nanoseconds after BDIN is asserted until 150 nanoseconds after BRPLY is negated. The master asserts BDIN a minimum of 150 nanoseconds after BRPLY is negated, and the cycle is continued as before (BBS7 remains asserted, and the slave responds to BDIN with BRPLY and BREF). BREF is stable from 75 nanoseconds after BRPLY is asserted until a minimum of 20 nanoseconds after BDIN is negated.

If BBS7 and BREF are not both asserted when BRPLY is negated, the slave removes the data from the bus 0 nanoseconds minimum and 100 nanoseconds maximum after negating BRPLY. The master negates BSYNC a minimum of 250 nanoseconds after the assertion of the last BRPLY and a minimum of 0 nanoseconds after the negation of that BRPLY.

DATBO Bus Cycle

The device-addressing portion of the DATBO bus cycle is the same as described earlier. The bus master gates BDAL<21:00>, BBS7, and the assertion of BWTBT onto the bus. Figure A-10 shows the DATBO bus-cycle timing.



t1	=	address to T SYNC	150ns min
t2	=	address hold	100ns min
t3	=	data to T DOUT	100ns min
t4	=	T DOUT to R RPLY	
	=	T (drive) + T (prop) + T (receive) + T (delay) + T (drive) + T (prop) + T (receive)	
t5	=	R RPLY to T DOUT	150ns min
t6	=	T DOUT to R RPLY	
	=	T (drive) + T (prop) + T (receive) + T (delay) + T (drive) + T (prop) + T (receive)	
t7	=	R RPLY to T DOUT	150ns min
T cell	=	t3 + t4 + t5 + t6 + t7	–since t3 < t7

Figure A-10 ▪ DATBO Bus-cycle Timing

A minimum of 100 nanoseconds after BSYNC is asserted, data on BDAL <15:00> and the negated BWTBT are put onto the bus. The master then asserts BDOUT a minimum of 100 nanoseconds after gating the data.

The slave receives stable data and BWTBT from a minimum of 25 nanoseconds before the assertion of BDOUT to a minimum of 25 nanoseconds after the negation of BDOUT. The slave asserts BRPLY a minimum of 0 nanoseconds after receiving BDOUT. It also asserts BREF concurrently with BRPLY if it is a block-mode device capable of supporting another BDOUT after the current one.

The master negates BDOUT 150 nanoseconds minimum after the assertion of BRPLY. If BREF was asserted when BDOUT was negated, and if the master wants to transmit more data in this block-mode cycle, then the new data is gated onto the bus 100 nanoseconds minimum after BDOUT is negated. BREF is stable from 75 nanoseconds maximum after BRPLY is asserted until 20 nanoseconds minimum after BDOUT is negated. The master asserts BDOUT for 100 nanoseconds minimum after gating new data onto the bus and 150 nanoseconds minimum after BRPLY negates. The cycle continues as before.

If BREF was not asserted when BDOUT was negated, or if the bus master does not want to transmit more data in this cycle, then the master removes data from the bus a minimum of 100 nanoseconds after negating BDOUT. The slave negates BRPLY a minimum of 0 nanoseconds after negating BDOUT. The bus master negates BSYNC a minimum of 175 nanoseconds after negating BDOUT and a minimum of 0 nanoseconds after the negation of BRPLY.

DMA Guidelines

- Bus masters that do not use block mode are limited to four DATI, four DATO, or two DATIO transfers per acquisition.
- Block-mode bus masters that do not monitor BDMR are limited to eight transfers per acquisition.
- If BDMR is not asserted after the seventh transfer, block-mode bus masters that do monitor BDMR can continue making transfers until the bus slave fails to assert BREF or until they reach the total maximum of 16 transfers. Otherwise, they stop after eight transfers.

▪ Interrupts

The interrupt capability of the Q22 bus allows any I/O device to temporarily suspend (interrupt) current program execution and divert processor operation to service the requesting device. The processor inputs a vector address from the device to start the service routine (handler). Like the device-register address, the device vector is set by the hardware at locations within a designated range below location 001000 (octal).

The interrupt vector in the MicroVAX I is determined by adding 200 (hex) to the vector supplied by the device and using this as a longword offset into the system-control block (SCB). This process yields the starting physical address of the Q22-bus device interrupt routine.

The interrupt vector in the MicroPDP-11 indicates the first of a pair of addresses. The content of the first address is read by the processor and is the starting address of the interrupt handler. The content of the second address is a new processor status word. The new processor status word can raise the interrupt priority level, thereby preventing lower-level interrupts from breaking into the current interrupt service routine. Control is returned to the interrupted program when the interrupt handler is ended. The original interrupted program's address and its associated processor status word are stored on a stack. The original program address and processor status word are restored by a return-from-interrupt instruction at the end of the handler. The use of the stack and the Q-bus interrupt scheme can allow interrupts to occur within nested interrupts, depending on the processor status word.

Interrupts can be caused by Q22-bus options or by the CPU. Interrupts that originate from within the processor are called *traps*. Traps are caused by programming errors, hardware errors, special instructions, and maintenance features.

The Q22-bus signals used in interrupt transactions are:

BIRQ4	Interrupt request priority level 4
BIRQ5	Interrupt request priority level 5
BIRQ6 L	Interrupt request priority level 6
BIRQ7 L	Interrupt request priority level 7
BIAKI L	Interrupt acknowledge output
BIAK L	Interrupt acknowledge output
BDAL <21:00> L	Data/address lines
BDIN L	Data input strobe
BRPLY L	Reply

Device Priority

The MicroPDP-11 on the Q22 bus supports the following two methods of device priority. The MicroVAX I, however, uses distributed arbitration only.

- Distributed Arbitration—priority levels are implemented on the hardware. When devices of equal priority level request an interrupt, priority is given to the device electrically closest to the processor.
- Position-Defined Arbitration—priority is determined solely by electrical position on the bus. The closer a device is to the processor, the higher its priority is.

Interrupt Protocol

The interrupt protocol has three phases:

- The *interrupt request* phase
- The *interrupt acknowledge and priority arbitration* phase
- The *interrupt vector transfer* phase

Figure A-11 shows the interrupt request/acknowledge sequence and Figure A-12 shows the interrupt protocol timing.

The *interrupt request* phase begins when a device meets its specific conditions for interrupt requests, such as when the device is ready, done, or an error has occurred. The interrupt enable bit in a device status register must be set. The device then initiates the interrupt by asserting the interrupt request line(s). BIRQ4 L is the lowest hardware priority level and is asserted for all interrupt requests. The level at which a device is configured must also be asserted. A special case exists for level 7 devices which must also assert level 6. This is described in the following four-level interrupt discussion:

Interrupt Level	Lines Asserted by Device
4	BIRQ4 L
5	BIRQ4 L, BIRQ5 L
6	BIRQ4 L, BIRQ6 L
7	BIRQ4 L, BIRQ6 L, BIRQ7 L

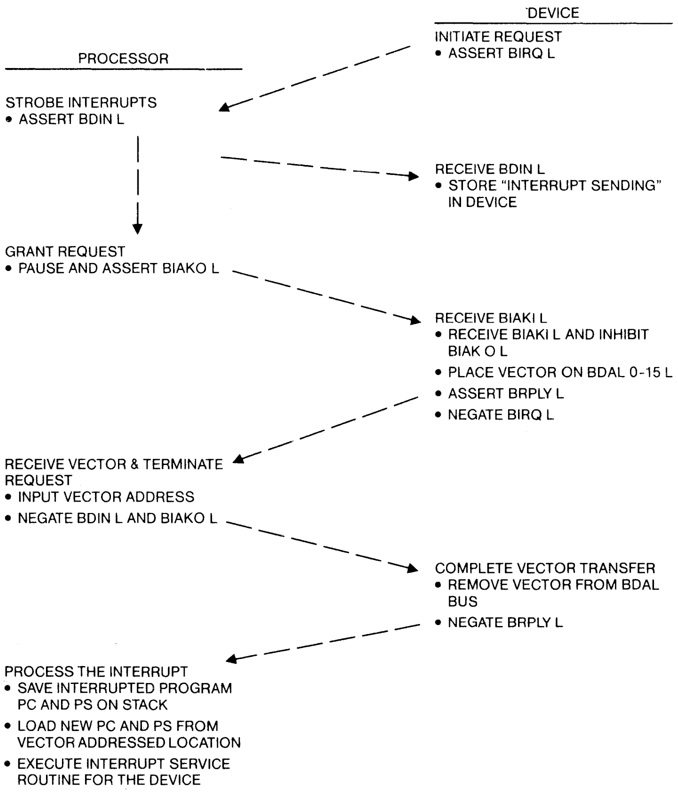

DEVICE
INITIATE REQUEST
• ASSERT BIRQ L
PROCESSOR
STROBE INTERRUPTS
• ASSERT BDIN L
RECEIVE BDIN L
• STORE "INTERRUPT SENDING" IN DEVICE
GRANT REQUEST
• PAUSE AND ASSERT BIAKO L
RECEIVE BIAKI L
• RECEIVE BIAKI L AND INHIBIT BIAK O L
• PLACE VECTOR ON BDAL 0-15 L
• ASSERT BRPLY L
• NEGATE BIRQ L
RECEIVE VECTOR & TERMINATE REQUEST
• INPUT VECTOR ADDRESS
• NEGATE BDIN L AND BIAKO L
COMPLETE VECTOR TRANSFER
• REMOVE VECTOR FROM BDAL BUS
• NEGATE BRPLY L
PROCESS THE INTERRUPT
• SAVE INTERRUPTED PROGRAM PC AND PS ON STACK
• LOAD NEW PC AND PS FROM VECTOR ADDRESSED LOCATION
• EXECUTE INTERRUPT SERVICE ROUTINE FOR THE DEVICE

Figure A-11 ▪ Interrupt Request/Acknowledge Sequence

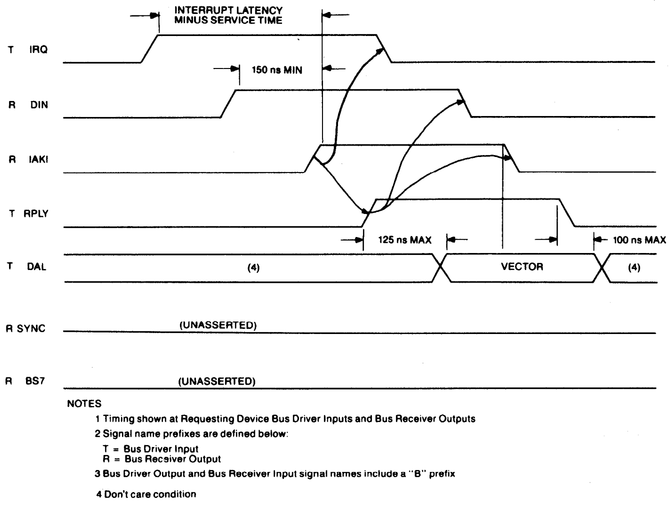


Figure A-12 ▪ Interrupt Protocol Timing

The interrupt request line remains asserted until the request is acknowledged.

During the *interrupt acknowledge and priority arbitration* phase, the processor will acknowledge interrupts under the following conditions:

- The device interrupt priority is higher than the current interrupt priority level.
- The processor has completed instruction execution and no additional bus cycles are pending.

The processor acknowledges the interrupt request by asserting BDIN L and, 150 nanoseconds minimum later, asserting BIAKO L. The device electrically closest to the processor receives the acknowledgment on its BIAKI L bus receiver.

The two types of arbitration are discussed separately. If the device that receives the acknowledgement uses the four-level interrupt scheme, the following occurs:

- If not requesting an interrupt, the device asserts BIAKO L, and the acknowledgment propagates to the next device on the bus.
- If the device is requesting an interrupt, it checks that no higher-level device is currently requesting an interrupt. This is done by monitoring higher-level request lines. The following table lists the lines that are monitored by devices at each priority level.

Device Priority Level	Lines(s) Monitored
4	BIRQ5, BIRQ6
5	BIRQ6
6	BIRQ7
7	—

In addition to asserting levels 7 and 4, level 7 devices must assert level 6. This is done to simplify the monitoring and arbitration by level 4 and 5 devices. In this protocol, level 4 and 5 devices need not monitor level 7 since level 7 devices assert level 6. Level 4 and 5 devices will become aware of a level 7 request because they monitor the level 6 request. This protocol has been optimized for level 4, 5, and 6 devices, because level 7 devices seldom are used.

- If no higher-level device is requesting an interrupt, the acknowledgment is blocked by the device (BIAKO L is not asserted). Arbitration logic within the device uses the leading edge of BDIN L to clock a flip-flop that blocks BIAKO L.
- If a higher-level request line is active, the device disqualifies itself and asserts BIAKO L to propagate the acknowledgment to the next device along the bus.

Signal timing must be considered when implementing four-level interrupts. Note Figure A-12.

If a single-level interrupt device receives the acknowledgment, it reacts as follows:

- If not requesting an interrupt, the acknowledgment is blocked using the leading edge of BDIN L, and arbitration is won. The interrupt vector transfer phase begins.
- If the device was requesting an interrupt, the acknowledgment is blocked using the leading edge of BDIN L, and arbitration is won. The interrupt vector transfer phase begins.

The *interrupt vector transfer* phase is enabled by BDIN L and BIAKI L. The device responds by asserting BRPLY L and its BDAL<15:00> L bus driver inputs with the vector address bits. The BDAL bus-driver inputs must be stable within 125 nanoseconds maximum after BRPLY L is asserted. The processor then inputs the vector address and negates BDIN L and BIAKO L. The device then negates BRPLY L and, 100 nanoseconds maximum later, removes the vector address bits. The processor then enters the device's service routine.

NOTE

Propagation delay from BIAKI L to BIAKO L must not be greater than 500 nanoseconds per Q-bus slot. The device must assert BRPLY L within 10 microseconds maximum after the processor asserts BIAKI L.

Four-level Interrupt Configurations
High-speed peripheral devices can attain better software performance using the four-level interrupt scheme. Both position-independent and position-dependent configurations can be used.

The position-independent configuration is shown in Figure A-13. This allows peripheral devices that use the four-level interrupt scheme to be placed in the backplane in any order. These devices must send out interrupt requests and monitor higher-level request lines, as described. The level 4 request is always asserted by a requesting device regardless of priority. If two or more devices of equally high priority request an interrupt, the device physically closest to the processor will win arbitration. To function properly, devices that use the single-level interrupt scheme must be modified or placed at the end of the bus for arbitration.

The position-dependent configuration, shown in Figure A-14, is simpler to implement. A constraint is that peripheral devices must be inserted with the highest-priority device located closest to the processor and the remaining devices placed in the backplane in decreasing order of priority, with the lowest-priority devices farthest from the processor. With this configuration, each device has to assert only its own level and level 4. Monitoring higher-level request lines is unnecessary. Arbitration is achieved through the physical positioning of each device on the bus. Single-level interrupt devices on level 4 should be positioned last on the bus.

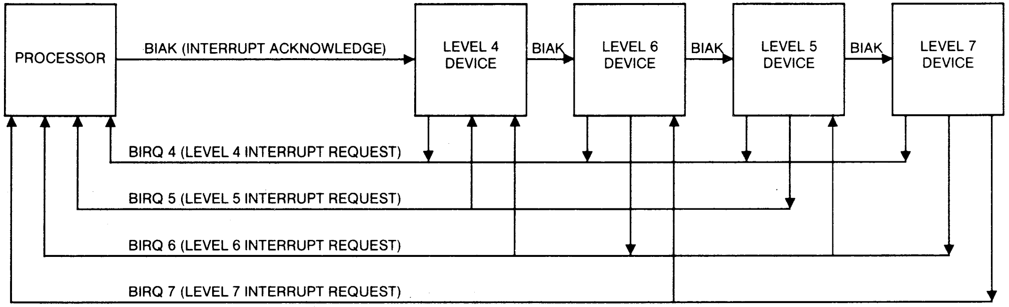

PROCESSOR
BIAK (INTERRUPT ACKNOWLEDGE)
LEVEL 4 DEVICE
BIAK
LEVEL 6 DEVICE
BIAK
LEVEL 5 DEVICE
BIAK
LEVEL 7 DEVICE
BIRQ 4 (LEVEL 4 INTERRUPT REQUEST)
BIRQ 5 (LEVEL 5 INTERRUPT REQUEST)
BIRQ 6 (LEVEL 6 INTERRUPT REQUEST)
BIRQ 7 (LEVEL 7 INTERRUPT REQUEST)

Figure A-13 ▪ Position-independent Configuration

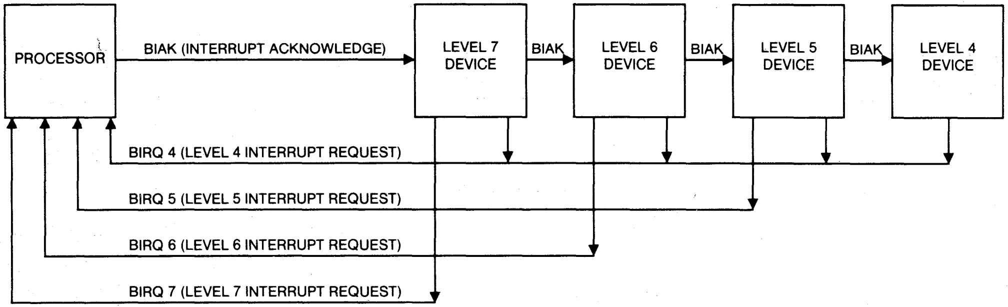

PROCESSOR
BIAK (INTERRUPT ACKNOWLEDGE)
LEVEL 7 DEVICE
BIAK
LEVEL 6 DEVICE
BIAK
LEVEL 5 DEVICE
BIAK
LEVEL 4 DEVICE
BIRQ 4 (LEVEL 4 INTERRUPT REQUEST)
BIRQ 5 (LEVEL 5 INTERRUPT REQUEST)
BIRQ 6 (LEVEL 6 INTERRUPT REQUEST)
BIRQ 7 (LEVEL 7 INTERRUPT REQUEST)

Figure A-14 ▪ Position-dependent Configuration

▪ Control Functions

The following Q22-bus signals provide control functions:

BREF L	Memory refresh (also block mode)
BHALT L	Processor halt
BINIT L	Initialize
BPOK H	Power OK
BDCOK H	dc power OK

Memory Refresh

If BREF is asserted during the address portion of a bus data transfer cycle, it causes all dynamic MOS memories to be addressed simultaneously. The sequence of addresses required for refreshing the memories is determined by the specific requirements for each memory. The complete memory-refresh cycle consists of a series of refresh-bus transactions. A new address is used for each transaction. A complete memory-refresh cycle must be completed within 1 or 2 milliseconds. Multiple data transfers by DMA devices must be avoided since they could delay memory-refresh cycles. This type of refresh is done only for memories that do not perform onboard refresh. The microsystems all use MSV11-P and MSV11-QA memories, which have the refresh logic included on the memory module. The BREF signal is used as a control signal for block mode.

Halt

Assertion of BHALT L for at least 25 microseconds interrupts the processor, which stops program execution and forces the processor unconditionally into console mode.

Initialization

Devices along the bus are initialized when BINIT L is asserted. The processor asserts BINIT L as a result of executing a powerup or powerdown sequence or after detection of a G character in console mode (MicroPDP-11) or an S character (MicroVAX I).

Power Status

Power-status protocol is controlled by two signals, BDCOK H and BPOK H. These signals are driven by some external device (usually the power supply).

- BDCOK H—When asserted, this signal indicates that dc power has been stable for at least 3 milliseconds. Once asserted, this line remains asserted until the power fails. It indicates that only 5 microseconds of dc power reserve remains.
- BPOK H—When asserted, this signal indicates that there is at least an 8-millisecond reserve of dc power and that BDCOK H has been asserted for at least 70 milliseconds. Once BPOK H has been asserted, it must remain asserted for at least 3 milliseconds. The negation of this line, the first event in the powerfail sequence, indicates that power is failing and that only 4 milliseconds of dc power reserve remain.

Powerup/Powerdown Protocol

Powerup protocol begins when the power supply applies power with BDCOK H negated. This forces the processor to assert BINIT L. When the dc voltages are stable, the power supply or other external device asserts BDCOK H. BINIT L is negated as a result of BDCOK H being asserted. The processor continues to test for BPOK H until it is asserted. The power supply asserts BPOK H 70 milliseconds minimum after BDCOK H is asserted. The processor then performs its powerup sequence. Normal power must be maintained at least 3 milliseconds before a powerdown sequence can begin.

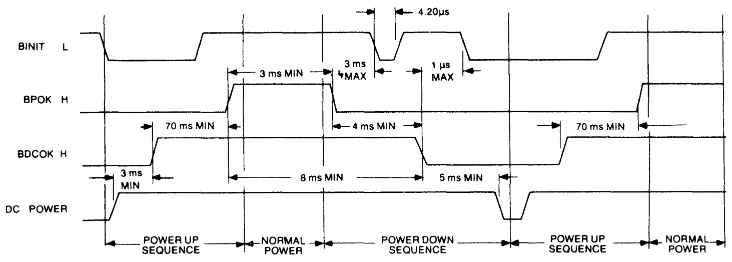
A powerdown sequence begins on the MicroVAX I when the power supply negates BPOK H. A powerfail interrupt is initiated if the current interrupt priority level is less than 1E (hex).

No later than 3 milliseconds after BPOK H negates, the processor asserts BINIT L for 8 to 20 microseconds. The power supply must negate BDCOK H no sooner than 4 milliseconds after the negation of BPOK H. This allows mass-storage devices to protect themselves against erasures and erroneous writes during a power failure.

The processor asserts BINIT L again no later than 1 microsecond after the negation of BDCOK H. The dc power must remain stable for a minimum of 5 microseconds after BDCOK H negates. BDCOK H must remain negated for a minimum of 3 milliseconds.

A powerdown sequence begins on the MicroPDP-11 when the power supply negates BPOK H. When the current instruction is completed, the processor traps to a powerdown routine at location 248. The end of the routine is terminated with a HALT instruction to avoid any possible memory corruption as the dc voltages decay.

When the processor executes the HALT instruction, it tests the BPOK H signal. If BPOK H is negated, the processor enters the powerup sequence. It clears internal registers, generates BINIT L, and continues to check for the assertion of BPOK H. If it is asserted and dc voltages are still stable, the processor will perform the rest of the powerup sequence. Figure A-15 illustrates powerup/powerdown timing.



NOTE
Once a power down sequence is started, it must be completed before a power-up sequence is started.

Figure A-15 ▪ Powerup/Powerdown Timing

▪ Bus Electrical Characteristics

The electrical specifications for the signals on the Q22 bus are as follows:

Input Logic Levels	
TTL Logical Low:	0.8 V dc maximum
TTL Logical High:	2.V dc minimum
Output Logic Levels	
TTL Logical Low:	0.4 V dc maximum
TTL Logical High:	2.4 V dc minimum

Load Definition

The ac loads are the maximum capacitance allowed per signal line to ground. A unit load is defined as 9.35 pF (picoFarads) of capacitance. The dc loads are defined as maximum current allowed with a signal-line driver asserted or unasserted. A unit load is defined as 210 microamperes in the unasserted state.

120-Ohm Bus

The electrical conductors interconnecting the bus-device slots are treated as transmission lines. A uniform transmission line, terminated in its characteristic impedance, will propagate an electrical signal without reflections. Since bus drivers, receivers, and wiring connected to the bus have finite resistance and nonzero reactance, the transmission line impedance is not uniform and therefore introduces distortions into pulses propagated along it. Passive components of the bus, such as wiring, cabling, and etched signal conductors, are designed to have a nominal characteristic impedance of 120 ohms.

The maximum length of interconnecting cable excluding wiring within the backplane is limited to 4.88 meters (16 feet).

Bus Drivers

Devices driving the 120-ohm bus must have open collector outputs and meet the following specifications:

- dc SPECIFICATIONS
  - Output low voltage when sinking 70 milliamps of current: 0.7 V maximum
  - Output high leakage current when connected to 3.8 V dc: 25 microamperes (even if no power is applied, except for BDCOK H and BPOK H)

These specifications must be met at worst-case supply voltage, temperature, and input-signal levels.

- ac SPECIFICATIONS
  - Bus-driver output-pin capacitive load: Not to exceed 10 picoFarads
  - Propagation delay: Not to exceed 35 nanoseconds
  - Skew (difference in propagation time between slowest and fastest gate): Not to exceed 25 nanoseconds
  - Rise/Fall Times: Transition time (from 10 - 90 percent for positive transition, and from 90 - 10 percent for negative transition) must be no faster than 10 nanoseconds

Bus Receivers

Devices that receive signals from the 120-ohm Q22 bus must meet the following requirements:

- dc SPECIFICATIONS
  - Input low voltage (maximum): 1.3 V
  - Input high voltage (minimum): 1.7 V
  - Maximum input current when connected to 3.8 V dc: 80 microamperes (even if no power is supplied)

These specifications must be met at worst-case supply voltage, temperature, and output-signal conditions.

- ac SPECIFICATIONS

Bus-receiver input-pin capacitance load: Not to exceed 10 picoFarads

- Propagation delay: Not to exceed 35 nanoseconds
- Skew (difference in propagation time between slowest and fastest gate): Not to exceed 25 nanoseconds

Bus Termination

The 120-ohm bus must be terminated at each end by an appropriate terminator, as shown in Figure A-16. This is a voltage divider with its Thevenin equivalent equal to 120 ohms and 3.4 V dc nominal. The MicroVAX I and MicroPDP-11 processor terminations are provided by the processor and backplane.

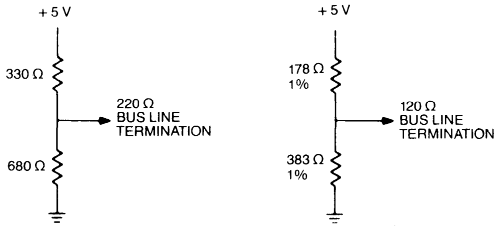


Figure A-16 ▪ 120-ohm Bus Terminations

Each of the bus signals starting with the letter B must see an equivalent network with the following characteristics at each end of the bus:

Input Impedance (with respect to ground)	12 ohms +5%, −15%
Open Circuit Voltage	3.4 V dc +5%
Capacitance Load	Not to exceed 30 pF

NOTE

The resistive termination can be provided by the combination of two modules (that is, the processor module supplies 220 ohms to ground. This, in parallel with another 220-ohm module, provides 120 ohms). Both of these terminators must be physically resident within the backplane.

▪ Bus Interconnect Wiring

The electrical characteristics of the wiring that connects the slots on the Q22-bus backplane must conform to certain requirements. These requirements ensure the proper operation of the installed modules.

Backplane Wiring

The wiring that connects all device interface slots on the backplane must meet the following specifications:

- The conductors must be arranged so that each line exhibits a characteristic impedance of 120 ohms, measured with respect to the bus common return.
- Crosstalk between any two lines must be no greater than 5 percent. Note that worst-case crosstalk is manifested by simultaneously driving all but one signal line and measuring the effect on the undriven line.
- The dc resistance of the signal path, as measured between the near-end terminator module and the far-end terminator module (including all intervening connectors, cables, backplane wiring, or connector module) must not exceed 2 ohms.
- The dc resistance of the common return path, as measured between the near-end terminator module and the far-end terminator module (including all intervening connectors, cables, backplane wiring, or connector module) must not exceed an equivalent of 2 ohms per signal path. Thus the dc resistance of the composite-signal return path must not exceed 2 ohms divided by 40 bus lines (or 50 milliohms). Although this common return path is nominally at ground potential, the conductance must be part of the bus wiring. The specified low-impedance return path must be provided by the bus wiring as distinguished from the common system or power ground path.

Intrabackplane Wiring

The wiring that connects the bus connector slots within one contiguous backplane is part of the overall bus transmission line. Because of implementation constraints, the nominal characteristic impedance of 120 ohms may not be achievable. Distributed wiring capacitance in excess of the amount required to achieve the nominal 120-ohm impedance may not exceed 60 picoFarads per signal line.

Power and Ground

Each bus interface slot has connector pins assigned for the following dc voltages. The maximum allowable current per pin is 1.5 amperes. +5 V dc must be regulated to +3 percent with a maximum ripple of 100 mV pp. +12 V dc must be regulated to +3 percent with a maximum ripple of 200 mV pp.

- +5 V dc—Three pins (4.5 amperes maximum per bus device slot)
- +12 V dc—Two pins (3.0 amperes maximum per bus device slot)
- Ground—Eight pins (shared by power return and signal return)

▪ Bus-system Configurations

Before configuring any system, three characteristics for each module in the system must be known. These characteristics are:

- Power consumption— +5 V dc and +12 V dc current requirements.
- ac bus loading—the amount of capacitance a module presents to a bus signal line. ac loading is expressed in terms of ac loads; one ac load equals 9.35 picoFarads of capacitance.
- dc bus loading—the amount of dc leakage current a module presents to a bus signal when the line is high (undriven). dc loading is expressed in terms of dc loads; one dc load equals 210 microamperes (nominal).

Power-consumption, ac-loading, and dc-loading specifications for each module are included in the *PDP-11 Systems and Options Catalog.*

NOTE

The ac and dc loads and the power consumption of the processor module and backplane must be included in determining the total loading of a backplane. Single-backplane systems are implemented in the MicroVAX I and MicroPDP-11. A single-backplane system is shown in Figure A-17.

- When using the MicroVAX I with 220-ohm processor termination, the bus can accommodate modules that have up to 20 ac loads total before additional termination is required. If more than 20 ac loads are included, the other end of the bus must be terminated with 120 ohms, and then up to 35 ac loads may be present.
- When using the MicroPDP-11 systems with 120-ohm processor termination, up to 35 ac loads can be used without additional termination. If 120-ohm bus termination is added, up to 45 ac loads can be configured in the backplane.
- The bus can accommodate modules for up to 20 total dc loads.
- The bus signal lines on the backplane can be up to 35.6 centimeters (14 inches) long.

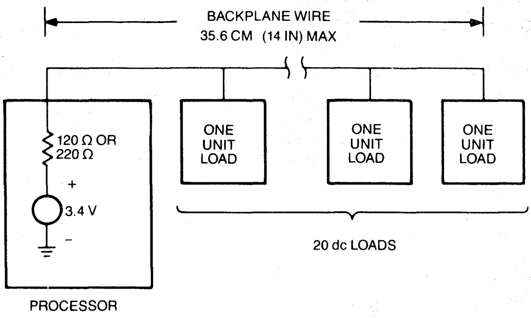


Figure A-17 ▪ Single-backplane Configuration

Power Supply Loading

Total power requirements for each backplane can be determined by obtaining the total power requirements for each module in the backplane. Obtain separate totals for +5 V and +12 V power. Power requirements for each module are specified in the *PDP-11 Systems and Options Catalog.*

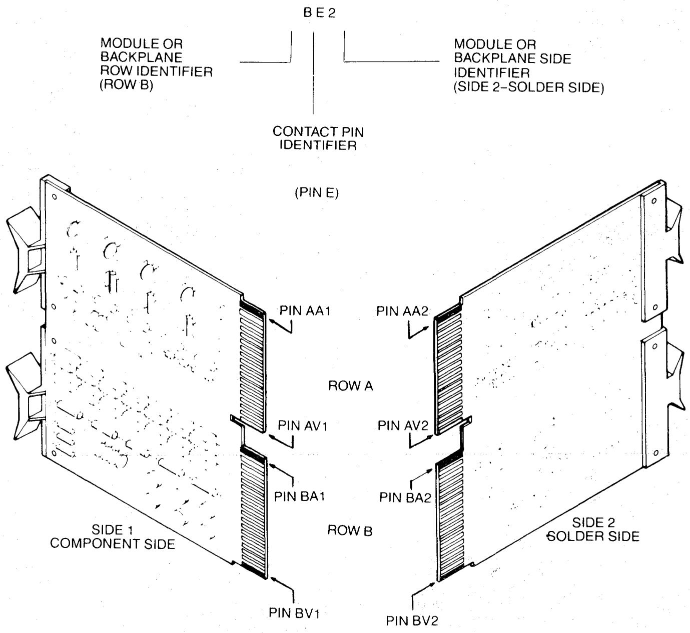


Figure A-18 ▪ Dual-height Module Contact Finger Identification

Module Connector Pin Identification

The microsystems use both dual-height and quad-height modules. The convention used to identify the pin connectors on the modules and backplane is shown in Figures A-18 and A-19. A dual-height module has two rows of connector pins, A and B, as shown in Figure A-18. A quad-height module has four rows labeled A, B, C, and D, which are shown in Figure A-19.

Each row has two sides. The side of the module where the components are located is designated as Side 1. The opposite, or soldered side, is designated as Side 2. Each row on each side consists of 18 contacts, for a total of 36 contacts per row. The contact designations are A through V, excluding G, I, O, and Q. The positioning notch between the two rows of pins mates with a protrusion on the connector block for correct module positioning. Examples of typical pin identifications on a module are shown in Figures A-18 and A-19.

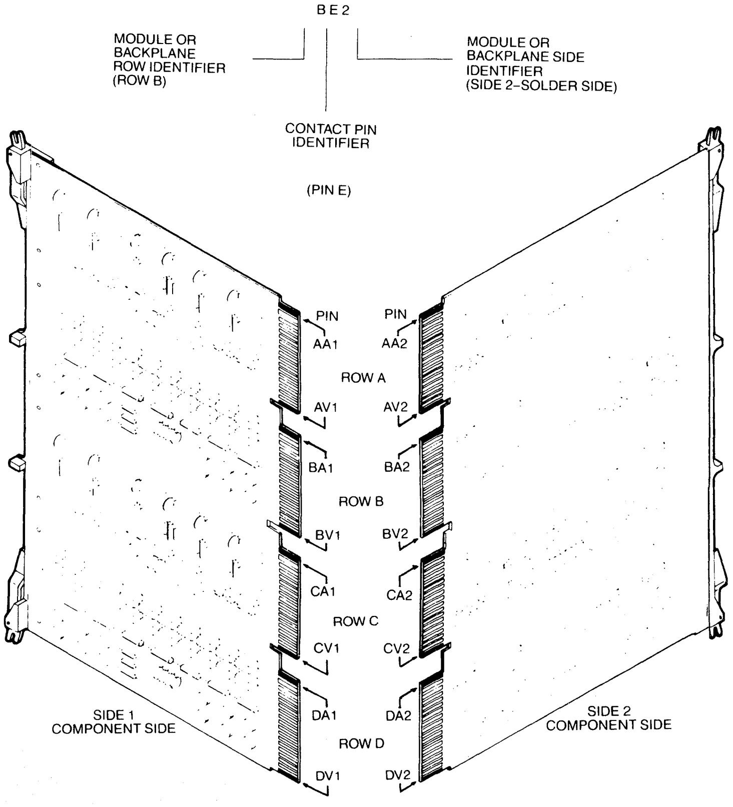


Figure A-19 ▪ Quad-height Module Contact Finger Identification

Table A-4 ▪ Bus-pin Identifiers

Bus Pin	Mnemonics	Description
AA1	BIRQ5 L	Interrupt Request Priority Level 5
AB1	BIRQ6 L	Interrupt Request Priority Level 6
AC1	BDAL16 L	Extended-address bit during addressing protocol; memory-error data line during data transfer protocol.
AD1	BDAL17 L	Extended-address bit during addressing protocol; memory-error logic enable during data transfer protocol.
AE1	SSPARE1 (Alternate +5B)	Special Spare—Not assigned or bused in Digital's cable or backplane assemblies; available for user connection. Optionally, this pin can be used for +5 V battery (+5B) backup power to keep critical circuits alive during power failures. A jumper is required on some bus options to disconnect the +5B circuit in systems that use this line as SSPARE1.
AF1	SSPARE2	Special Spare—Not assigned or bused in Digital's cable or backplane assemblies; available for user interconnection. In the highest-priority device slot, the processor can use this pin for a signal to indicate its RUN state.
AH1	SSPARE3 (SRUN simul-taneously)	Special Spare—Not assigned or bused in Digital's cable or backplane assemblies; available for user interconnection. An alternate SRUN signal may be connected in the highest-priority set.
AJ1	GND	Ground—System signal ground and dc return.
AK1	MSPAREA	Maintenance Spare--Normally connected together on the backplane at each option location (not a bused connection).
AL1	MSPAREB	Maintenance Spare—Normally connected together on the backplane at each option location (not a bused connection).
AM1	GND	Ground—System signal ground and dc return.

Bus Pin	Mnemonics	Description
AN1	BDMR L	Direct Memory Access (DMA) Request—A device asserts this signal to request bus mastership. The processor arbitrates bus mastership between itself and all DMA devices on the bus. If the processor is not bus master (it has completed a bus cycle and BSYNC L is not being asserted by the processor), it grants bus mastership to the requesting device by asserting BDMGO L. The device responds by negating BDMR L and asserting BSACK L.
AP1	BHALT L	Processor Halt—When BHALT L is asserted for at least 25 microseconds, the processor responds by halting normal program execution and entering console mode.
AR1	BREF L	Memory Refresh—Asserted by a DMA device. This signal forces all dynamic MOS memory units requiring bus refresh signals to be activated for each BSYNC L/BDIN L bus transaction. It is also used as a control signal for block mode. CAUTION The user must avoid multiple DMA data transfers (burst or "hog" mode) that could delay refresh operation if using DMA refresh. Complete refresh cycles must occur once every 1.6 milliseconds if required.
AS1	+12B or +5B	+12 V dc or +5 V battery backup power to keep critical circuits alive during power failures. This signal is not bused to BS1 in all of Digital's backplanes. A jumper is required on all bus options to disconnect the backup circuit from the bus in systems that use this line at the alternate voltage.
AT1	GND	Ground—System signal ground and dc return.
AU1	PSPARE1	Spare (not assigned; customer usage not recommended). Prevents damage when modules are inserted upside down.
AV1	+5B	+5 V Battery Power—Secondary +5 V power connection. Battery power can be used with certain devices.

Bus Pin	Mnemonics	Description
BA1	BDCOK H	dc Power OK—Power supply-generated signal that is asserted when there is sufficient dc voltage available to sustain reliable system operation.
BB1	BPOK H	Power OK—Asserted by the power supply 70 milliseconds after BDCOK is negated (when ac power drops below the value required to sustain power; approximately 75 percent of nominal). When negated during processor operation, a power-fail trap sequence is initiated.
BC1	SSPARE4 BDAL 18L (22-bit only)	Special Spare—Not assigned. Bused in cable and backplane assemblies; available for user interconnection.
BD1	SSPARE5 BDAL 19L (22-bit only)	Special Spare—Caution—these pins can be used as test points in some options.
BE1	SSPARE6 BDAL 20L	These bused address lines are address lines <21:18> and are used only during the address portion of the bus operation.
BF1	SSPARE7 BDAL 21L	These bused address lines are address lines <21:18> and are only used during the address portion of the bus operation.
BH1	SSPARE8	Special Spare—Not assigned or bused in Digital's cable and backplane assemblies; available for user interconnection.
BJ1	GND	Ground—System signal ground and dc return.
BK1 BL1	MSPAREB MSPAREB	Maintenance Spare—Normally connected together on the backplane at each option location (not a bused connection).
BM1	GND	Ground—System signal ground and dc return.
BN1	BSACK L	This signal is asserted by a DMA device in response to the processor's BDMGO L signal, indicating that the DMA device is bus master.
BP1	BIRQ7 L	Interrupt Request Priority Level 7

Bus Pin	Mnemonics	Description
BR1	BEVNT L	External Event Interrupt Request—When asserted, the processor responds by entering a service routine via vector address 1008. A typical use of this signal is a line time-clock interrupt. This signal is not used in the MicroVAX I processor.
BS1	+12B	+12 V dc battery-backup power (not bused to AS1 in all of Digital's backplanes). Not supplied by Digital.
BT1	GND	Ground—System signal ground and dc return.
BU1	PSPARE2	Power Spare 2—Not assigned a function. Not recommended for use. If a module is using −12 V (on pin AB2), and if the module is accidentally inserted upside down in the backplane, −12 V dc appears on pin BU1.
BV1	+5	+5 V Power—Normal +5 V dc system power.
AA2	+5	+5 V Power—Normal +5 V dc system power.
AB2	−12	−12 V Power— −12 V dc (optional) power for devices requiring this voltage. NOTE Modules that require negative voltages contain an inverter circuit on each module that generates the required voltages(s). The −12 V power is not required with Digital-supplied options.
AC2	GND	Ground—System signal ground and dc return.
AD2	+12	+12 V Power—+12 V dc system power.
AE2	BDOUT L	Data Output—BDOUT, when asserted, implies that valid data is available on BDAL<0:15> L and that an output transfer, with respect to the bus-master device, is taking place. BDOUT L is deskewed with respect to data on the bus. The slave device responding to the BDOUT L signal must assert BRPLY L to complete the transfer.
AF2	BRPLY L	Reply—BRPLY L is asserted in response to BDIN L OR BDOUT L and during IAK transactions. It is generated by a slave device to indicate that it has placed its data on the BDAL bus or that it has accepted output data from the bus.

Bus Pin	Mnemonics	Description
AF2	BDIN L	Data Input—BDIN L is used for two types of bus operations: When asserted during BSYNC L time, BDIN L implies an input transfer with respect to the current bus master, and requires a response (BRPLY L). BDIN L is asserted when the master device is ready to accept data from a slave device.
AJ2	BSYNC L	Synchronize—BSYNC L is asserted by the bus-master device to indicate that it has placed an address on BDAL<0:17> L. The transfer is in process until BSYNC L is negated.
AK2	BWTBT L	Write/Byte—BWTBT L is used in two ways to control a bus cycle: It is asserted at the leading edge of BSYNC L to indicate that an output sequence is to follow (DATO or DATOB), rather than an input sequence. It is asserted during BDOUT L, in a DATOB bus cycle, for byte addressing.
AL2	BIRQ4 L	Interrupt Request Priority Level 4
AM2 AN2	BIAKI L BIAKO L	Interrupt Acknowledge—In accordance with interrupt protocol, the processor asserts BIAKO L to acknowledge receipt of an interrupt. The bus transmits this to BIAKI L of the device electrically closest to the processor. This device accepts the interrupt acknowledgment under two conditions: The device requested the bus by asserting BIRQXL. The device has the highest-priority interrupt request on the bus at that time. If these conditions are not met, the device asserts BIAKO L to the next device on the bus. This process continues in a daisychain fashion until the device with the highest interrupt priority receives the interrupt acknowledge signal.

Bus Pin	Mnemonics	Description
AP2	BBS7 L	Bank 7 Select—The bus master asserts this signal to reference the I/O page (including that portion of the I/O page reserved for nonexistent memory). The address in BDAL<0:12> L when BBS7 L is asserted is the address within the I/O page.
AR2 AS2	BDMGI L BDMGO L	Direct Memory Access Grant—The bus arbitrator asserts this signal to grant bus mastership to a requesting device, according to bus mastership protocol. The signal is passed in a daisychain from the arbitrator (as BDMGO L) through the bus to BDMGI L of the next priority device (electrically closest device on the bus). This device accepts the grant only if it requested to be bus master (by a BDMR L). If not, the device passes the grant (asserts BDMGO L) to the next device on the bus. This process continues until the requesting device acknowledges the grant. CAUTION DMA device transfers must not interfere with the memory-refresh cycle.
AT2	BINIT L	Initialize—This signal is used for system reset. All devices on the bus are to return to a known, initial state. That is, registers are reset to 0, and logic is reset to state 0. Exceptions should be completely documented in programming and engineering specifications for the device.
AU2 AV2	BDAL0 L BDAL1 L	Data/address Lines—These two lines are part of the 16-line data/address bus over which address and data information are communicated. Address information is first placed on the bus by the bus master device. The same device then either receives input data from, or outputs data to, the addressed slave device or memory over the same bus lines.
BA2	+5	+5 V Power—Normal +5 V dc system power.
BB2	−12	−12 V Power— −12 V dc (optional) power for devices requiring this voltage. Voltages normally not supplied by Digital.
BC2	GND	Ground—System signal ground and dc return.

Bus Pin	Mnemonics	Description
BD2	+12	+12 V Power—+12 V system power.
BE2	BDAL2 L	Data/address Lines—These 14 lines are part of the 16-line data/address bus previously described.
BF2	BDAL3 L	
BH2	BDAL4 L	
BJ2	BDAL5 L	
BK2	BDAL6 L	
BL2	BDAL7 L	
BM2	BDAL8 L	
BN2	BDAL9 L	
BP2	BDAL10 L	
BR2	BDAL11 L	
BS2	BDAL12 L	
BT2	BDAL13 L	
BU2	BDAL14 L	
BV2	BDAL15 L	

▪ Additional Documentation

PDP-11 Architecture Handbook	*EB-23657-18*
PDP-11 Systems and Options Catalog	*ED-26050-41*

Appendix B • PDP-11/23-PLUS Overview

The PDP-11/23-PLUS is the original 16-bit Q-bus microcomputer. It is compatible with PDP-11 hardware design and PDP-11 system software. The PDP-11/23-PLUS uses the same CPU module as the MicroPDP-11/23 system—the KDF11-B. It can address up to 4 Mbytes of parity MOS memory, and can be packaged with up to 40 Mbytes of disk storage. The PDP-11/23-PLUS accepts all of the same options as the MicroPDP-11 family.

The PDP-11/23-PLUS can support up to eight concurrent, active users depending on the application. It is an ideal system for both commercial and technical users.

PDP-11/23-PLUS Features

- Utilizes complete PDP-11 instruction set including EIS
- Contains Q22-bus interface that supports block-mode DMA and up to 4 Mbytes of physical memory
- Uses optional floating-point and commercial instruction sets
- Supports memory management in kernel and user modes
- Supports four levels of vectored interrupts
- Includes a line-frequency clock
- Contains a 2-Kbyte diagnostic/bootstrap ROM
- Supports the ODT console emulator
- Includes powerfail/autorestart hardware
- Uses a nine-slot 22-bit backplane (H9276-A)
- Includes two asynchronous serial line units
- Includes an I/O distribution panel (H349) on cabinet-mounted systems
- Available in a cabinet-mounted or rackmount configuration
- Uses all of the MicroPDP-11 family operating systems

PDP-11/23-PLUS Configurations

The PDP-11/23-PLUS can be ordered as a cabinet-mounted system. This package contains the PDP-11/23-PLUS CPU and chassis mounted in a 42-inch-high H9642-series cabinet with two RL02 removable-disk drives and a controller, or with two RX02 flexible-diskette drives and a controller. For more information on the RL02 and RX02 storage devices, refer to Chapter 3—System Options.

The PDP-11/23-PLUS can also be ordered as a rackmount system. This configuration includes a PDP-11/23-PLUS CPU and chassis ready for mounting into a customer-supplied cabinet. No mass storage is included with the rackmount system. The RX50 dual-diskette drive can be added as an external device for software interchange with the other microsystems. Any of the other Q-bus peripherals may also be added.

▪ Cabinet-mounted System Specifications

- Height: 41.7 inches (106 centimeters)
- Width:21.3 inches (54 centimeters)
- Depth:30.0 inches (76.2 centimeters)
- Weight: 375 pounds (170.3 kilograms) for the dual-RL02 model
  288 pounds (139.6 kilograms) for the dual-RX02 model
- Watts: 840 for the dual-RL02 model
  380 for the dual-RX02 model
- Btu/hr: 2864 for the dual-RL02 model
  1297 for the dual-RX02 model

▪ Rackmount System Specifications

- Height: 5.25 inches (13.3 centimeters)
- Width:19.0 inches (48.3 centimeters)
- Depth:26.8 inches (68.0 centimeters)
- Weight: 45 pounds (20.0 kilograms)
- Watts:540
- Btu/hr: 1800

▪ Additional Documentation

PDP-11/23-PLUS System Manual	*EK-1T23B-OP*
Microcomputers and Memories Handbook	*EB-20912-20*
KDF11-B CPU Module User's Guide	*EK-KDFEB-UG*
PDP-11 Systems and Options Catalog	*ED-26051-41*

Appendix C • Site Preparation and Installation

Before the microsystem arrives, adequate planning and preparation of the area where the system will be located will simplify the installation and ensure the reliable operation of the system. The considerations are:

- Space to install the system
- Proper environmental conditions
- Correct power outlets and adequate power

Space

Sufficient space must be provided around the microsystem unit and terminals to allow access to the units, and to enable the proper circulation of air through the units. The surroundings should be comfortable for the operating personnel, and the area should be free from traffic and spills. The microsystem units should be positioned to allow the operator access to the diskette drives located at the front of the unit.

Table C-1 shows the overall dimensions of the microsystem units.

System

The MicroVAX I and MicroPDP-11 system units are available in floorstand, tabletop, or rackmount versions. The floorstand version is designed to fit under a standard 30-inch (76.2-centimeter) desk or table. It can easily be converted to a tabletop version by removing the pedestal mount. The rackmount version is designed to be installed cabinet such as the H9642.

Table C-1 • Microsystem-unit Dimensions

Dimension	Floorstand	Tabletop	Rackmount	Cabinet-mounted
Height	24.5 inches 62.2 cm	6.0 inches 15.2 cm	5.25 inches 13.3 cm	41.7 inches 106.0 cm
Width	10.0 inches 25.4 cm	22.25 inches 56.5 cm	19.0 inches 48.3 cm	21.2 inches 53.9 cm
Depth	28.5 inches 72.4 cm	38.5 inches 72.4 cm	25.5 inches 64.8 cm	31.5 inches 80.0 cm
Weight	70 pounds 32 kg	70 pounds 32 kg	55 pounds 25 kg	332 pounds 150.4 kg

Console Terminal

The console terminal is usually positioned close to the system operator. The console terminal keyboard is movable and can be placed in front of the display or close to the terminal.

Printers

Printers require space for the paper supply and for access to load the paper into the printer. The space requirements depend on the type of paper used (stationery or fanfold), and the location of the trays. The printer unit can be positioned at any location in the area that is convenient to the operator.

Cables

The power cord and the signal cables that connect the units in a system should be routed away from where the operator will be or from areas of traffic. No heavy objects should be placed on the cables and the cables should be long enough to prevent strain on the cable connectors. Sharp cable angles should also be avoided.

▪ Environment

The microsystems will operate in most normal working environments that provide comfortable surroundings for the operator. These areas include offices, schools, hospitals, or manufacturing areas that have controlled environments. There are, however, certain environmental conditions which may not be suitable for microsystem operation. Areas that are subject to extreme temperature and humidity changes, areas of high contamination in the air, and areas with electrical interference conditions are not recommended.

Temperature and Humidity

Table C-2 lists the acceptable temperature ranges and humidity levels for operating each of the microsystems. These conditions are specified for the operation of the microsystem unit, storage devices, console terminal, and printing devices supplied by Digital as part of the system. These conditions may not apply to devices and equipment supplied by other manufacturers.

Table C-2 ▪ Operating Temperature and Humidity

	MicroVAX I	**MicroPDP-11**
Temperature Range	59–90°F 15-32°C	59–90°F 15–32°C
Relative Humidity	20–80%	20–80%

NOTE

Some Field Service contracts require specific temperature and humidity limits that may differ from those listed.

Air Contamination

Dust and dirt particles that are suspended in air can block the air filters and prevent the proper circulation of air through the microsystem unit and the system devices. Disk drives and other mechanical devices may also be adversely affected by dust and dirt. To prevent these conditions, the system area should be kept reasonably clean by periodic vacuum cleaning. Where possible, the area in which the microsystem is operating should be environmentally separated from other areas that produce contaminants.

Electrical Interference

Digital's microsystems, terminals, and cables are provided with shielding and electrical filters to limit the effects of electrical interference. Electrical interference can be transmitted through power lines or through the air. High levels of interference from electrical power equipment, x-ray equipment, or radio devices can adversely affect the system operation. If any of these conditions exist at the operating site, additional filters and shielding may be necessary for reliable system operation.

Room Lighting

Reduced lighting levels in the area where the video-display terminals (VDTs) are operated can prevent excessive reflection from the display screen. Light levels can be varied by electrical dimming controls or by shielding the direct light.

NOTE

Electrical dimming controls are frequently the cause of electrical noise. Dimming devices that are designed only for computer applications should be used.

Screen filters for the VDTs are available to reduce reflections and to improve the contrast of the characters on the display.

Shock and Vibration

Digital's microsystems are designed to withstand the normal shock and vibrations that occur during shipment and under normal operation. If the system will be subjected to excessive shock or vibration conditions, Field Service should evaluate the location before the system is installed.

Static Electricity

Static electricity, generated in the area in which the system equipment is operating, can result in data loss, program errors, and other system failures. Static electricity is usually caused by a low humidity level and carpeted floor surfaces. A static charge occurs when people walk on the carpeted surface or when they move in vinyl- or fabric-covered chairs. The static electricity is discharged when a person comes in contact with the equipment, which is grounded by the power cord line or other leads.

Static electricity can be reduced by keeping the relative humidity levels at 40 percent or greater, and by using special antistatic carpeting. If carpeting exists in the area in which the equipment will be installed, special antistatic pads are recommended and can be placed close to the equipment where the operators will be located.

▪ Power

The microsystems will operate with the following ac line voltages and frequency variations.

	120 V ac	**240 V ac**
Voltage	88–128 V RMS	176–256 V RMS
Frequency	47–63 Hz	47–63 Hz

The ac power cord that supplies power to the system unit and terminals should be a separate line dedicated only to the system and terminals.

The appropriate country power cord is shipped with each microsystem. If terminals are ordered, the appropriate country power cords are shipped with them as well.

ac Power Quality

The quality and reliability of the ac power can vary depending on the location. Some sites may require that the user provide power conditioning equipment to ensure that the proper power tolerances and filtering are maintained.

ac System Power

The ac power source should be adequate to supply the original system and allow for system expansion. A dedicated 15-ampere branch circuit from the power distribution panel is recommended for each system. This circuit must meet all national and local standards that apply to it. It must provide a good system ground, be stable, and free from electrical noise.

NOTE

Do not connect other equipment, such as air conditioners, office copiers, or coffee pots, on the same circuit with the microsystem.

Table C-3 lists the maximum ac current requirements, wattage, and heat dissipation for the microsystem units and console terminals. When other devices, such as printing terminals, are used, the additional ac current requirements must be included in the total.

Table C-3 ▪ Microsystem Power

Device	**ac Current**		**Watts**	**Heat Dissipation**
	120 V	**240 V**		**BTU/k (g-cal/s)**
MicroVAX I Unit	4.4 A	2.2 A	345	1177.4 (82.3)
MicroPDP-11 Unit	4.4 A	2.2 A	345	1177.4 (82.3)
VT100 Terminal	1.9 A	1.5 A	150	511.9 (35.8)
VT220 Terminal	.48 A	.24 A	60	204.7 (14.3)

▪ Storage

Adequate storage facilities should be provided for the equipment, software, paper supplies, and accessories.

Systems and Peripherals

If it becomes necessary to store the microsystem units, add-on storage devices, or terminals for extended periods, the equipment should be enclosed in plastic covering and placed in the original containers. It is recommended that the system be stored in a safe area not subjected to rapid or extreme temperature changes or high humidity levels.

Supplies

The system supplies, including flexible diskettes, printer paper, and ribbons, should be stored in a cabinet or containers that will protect them from damage or contaminants. The storage facilities should be located close to the system for ease of accessibility.

Installation Considerations

Before an option can be added to a microsystem, the following questions should be answered:

- Does the system have sufficient dc power to support the additional option?
- Does the Q22 bus have adequate ac loads available for the module?
- Is space available in the backplane for the module installation?
- Is space available on the I/O distribution panel for the connector panel insert?
- Have the modules been properly configured for installation?

Disk-drive Installation

The RQDX1 disk controller module for the MicroVAX I and MicroPDP-11 controls and communicates with up to four logical disk-drive units. The RX50 is considered to be two units because it contains two separate diskette drives in one mechanical assembly. The RD51 and the RD52 disk drives are each considered to be a single unit. Detailed information on the removal and installation procedures is contained in the *MicroVAX I Owner's Manual* and the *MicroPDP-11 Owner's Manual*. Currently, the only allowable integrated disk-drive configuration in the MicroVAX I and MicroPDP-11 systems is one fixed-disk drive (either an RD51 or RD52) and one RX50.

The RLV12 disk controller module controls and communicates with up to four RL02 disk-drive units. Two RL02 disk drives can be packaged with the MicroPDP-11 system unit in an H9642 cabinet, and two additional RL02 disk drives can be installed in an expansion RL02 cabinet.

The RQDX1 and RLV12 controller modules are preset during manufacturing and require no adjustments or wiring before installation. The position of the modules in their respective backplanes is dependent on the other option modules that are installed.

Backplane Assembly Layout

The H9278-A backplane assembly of the MicroVAX I and MicroPDP-11 contains eight slots. Figure C-1 shows the backplane layout. The systems and options modules can be quad height or dual height. A quad-height module occupies all four rows—A, B, C, and D—of a backplane slot. A dual-height module occupies only two rows—A and B or C and D. Some of the backplane slots are dedicated to particular systems modules. Options modules are installed in the remaining slots according to their priority. Table C-4 lists the slots and rows and defines the module locations.

Slots 1 through 3 of the MicroVAX I backplane are dedicated to the CPU (occupies two slots) and main memory (one slot). Rows A and B and rows C and D of slots 4 through 8 provide the Q22-bus signals to all of the modules.

Slots 1 and 2 of the MicroPDP-11 backplane are dedicated to the CPU and main memory (one slot each). Rows A and B of all the backplane slots provide the Q22-bus signals to each of the modules. Rows C and D of slots 1 through 3 provide an interconnection between each other. This interconnection is referred to as the CD bus. The C and D slots of the five remaining slots (4 through 8) also provide Q22-bus signals as in those supplied by rows A and B.

MicroVAX

	ROWS				
SLOTS	A	B	C	D	MODULE SIZE
1	M7136 MEMORY CONTROLLER MODULE				2 quad-height KD32-A
2	M7135 DATA PATH MODULE				
3	Q-CD Slot		Q-CD Slot		1 quad-height or 2 dual-height
4	Q22 Bus		Q22 Bus		
5	Q22 Bus		Q22 Bus		
6	Q22 Bus		Q22 Bus		
7	Q22 Bus		Q22 Bus		
8	Q22 Bus		Q22 Bus		

*viewed from module insertion side

MicroPDP-11

	ROWS				
SLOTS	A	B	C	D	
1	KDJ11-B	or	KDF11-B		MODULE SIZE 1 quad-height
2	Q-CD Slot		Q-CD Slot		1 quad-height or 2 dual-height
3	Q-CD Slot		Q-CD Slot		
4	Q22 Bus		Q22 Bus		
5	Q22 Bus		Q22 Bus		
6	Q22 Bus		Q22 Bus		
7	Q22 Bus		Q22 Bus		
8	Q22 Bus		Q22 Bus		

*viewed from module insertion side

Figure C-1 ▪ Backplane-slot Assignments

Table C-4 ▪ Backplane-module Assignments

Slot	Row	MicroVAX I	MicroPDP-11/73	MicroPDP-11/23
1	ABCD	CPU	CPU	CPU
2	ABCD	CPU	Memory	Memory
3	ABCD	Memory	Additional memory or communications option	Additional memory or communications option
4	ABCD	Additional memory or communications option	Two dual-height options or one quad-height option. RQDX1 if it is the last-used slot.	Two dual-height options or one quad-height option. RQDX1 if it is the last-used slot.
5	ABCD	Two dual-height options or one quad-height option. RQDX1 if it is the last-used slot.	See Slot 4	See Slot 4
6	ABCD	See Slot 5	See Slot 4	See Slot 4
7	ABCD	See Slot 5	See Slot 4	See Slot 4
8	ABCD	See Slot 5	See Slot 4	See Slot 4

Power and Loading Requirements

The System Configuration Worksheet, Figure C-2, can be used to determine the power consumption and bus load requirements of the options to be installed. The worksheet includes +5 V dc current, + 12 V dc current, and the ac bus loads.

To calculate the power and loading, enter the current values of the options that have previously been included in the system and the current values of the options to be added as shown. These are entered in the option column.

The current (amperes) column lists the total current available from the +5 V dc and the + 12 V dc outputs of the power supply in the "available" columns. The *PDP-11 System and Options Catalog* lists the specifications of all the system modules, options modules, and storage devices. Enter the current requirements in the "used" column of Figure C-2 for each option included or to be installed and subtract that value from the previous "available" column. Enter the new value in the "available" column next to the value entered into the "used" column. When all the values have been entered on the worksheet, perform the total wattage calculation shown on the bottom of Figure C-2.

The power supply on the MicroVAX I and MicroPDP-11 systems provides a maximum of 230 watts—up to 36 amperes at +5 V (180 watts) and up to 7 amperes at + 12 V (84 watts). This total is 264 watts and exceeds the 230-watt maximum allowable total of the power supply. Therefore, the total wattage for the +5 V dc and + 12 V dc must be calculated separately to prevent exceeding the total power-supply voltage. The RX50, RD51, and RD52 disk drives also require power from the power supply and must be included in the power calculations.

The ac bus loading is calculated in a similar manner to the dc bus loading. Each bus load is entered into the "used" column and subtracted from the previous available value. When the value is less than 1, the total bus loading has been exceeded.

For complete power and loading requirements for each of the microsystems and options, refer to the *PDP-11 Systems and Options Catalog.*

slot		OPTION	CURRENT (amps)				ac BUS LOADS		PANEL INSERTS			
			+5V		+12V				2 x 3		1 x 4	
			used	available	used	available	used	available	used	available	used	available
		Maximum	–	36.0	–	7.0	–	29.0	–	4	–	2 or 5
1	two quads = KD32-A											
2		KD32-A	14.0	22.0	0.5	6.5	2.0	27.0	1	3	–	2 or 5
3	reserved											
4	1 quad or 2 dual	MSV11-PL	3.6	18.4			2.0	25.0				
5	1 quad or 2 dual	DZV11	1.2	17.2	.4	6.1	4.0	21.0	1	2		
6	1 quad or 2 dual	RQDX1	6.4	10.8	.1	6.0	2.5	18.5				
7	1 quad or 2 dual											
8	1 quad or 2 dual											
	diskette drive unit 1	RX50	.85	9.95	1.8	4.2						
	disk drive unit 2	RD51	1.0	8.95	1.6	2.6						

Total of this column x 12 = 52.8 watts at + 12V

Total of this column x 5 = 135.25 watts at + 5V

188.0 Total watts (must be 230 or less)

Figure C-2 ▪ Sample Configuration Worksheet

Module Space Requirements

After the power and ac load values have been calculated, the location of the module in the backplane must be determined. The backplane slot where the module will be installed is determined by the number of existing modules installed and by the interrupt priority level of the additional modules. Table C-4 lists the priority levels of each option. Modules with the highest priority should be closest to the CPU module(s). Figure C-3 shows several different module configurations that can be implemented in the backplanes. The following summary provides some specific guidelines for the module locations:

- The MicroVAX I CPU modules always occupy slots 1 and 2. The MicroPDP-11 CPU module occupies slot 1 only.
- The memory module(s) is always installed starting with the next slot after the CPU module(s).
- The DZV11 module installs in the slot following the last memory module.
- The DEQNA module installs in the slot following the last memory module.
- If both the DZV11 and the DEQNA modules are present, the DEQNA is installed in the next highest slot following the last memory module and the DZV11 follows in the next slot after the DEQNA.
- The DLVJ1 module installs in the slot following the DEQNA or DZV11 module.
- The RQDX1 or RLV12 always install in the next highest quad slot following the last option installed. If this last option has an open dual slot, a G7272 grant-continuity module must be mounted in row A or C of that slot.
- If only one dual-height module is installed in a slot between two quad-height modules, the G7272 must be installed in row A or row C of the same slot.
- If only one dual-height module is installed in slot 8, it must be installed in rows A and B. If only one dual-height module is installed in slot 7, and slot 7 is the last-used slot, the module must be installed in rows C and D.

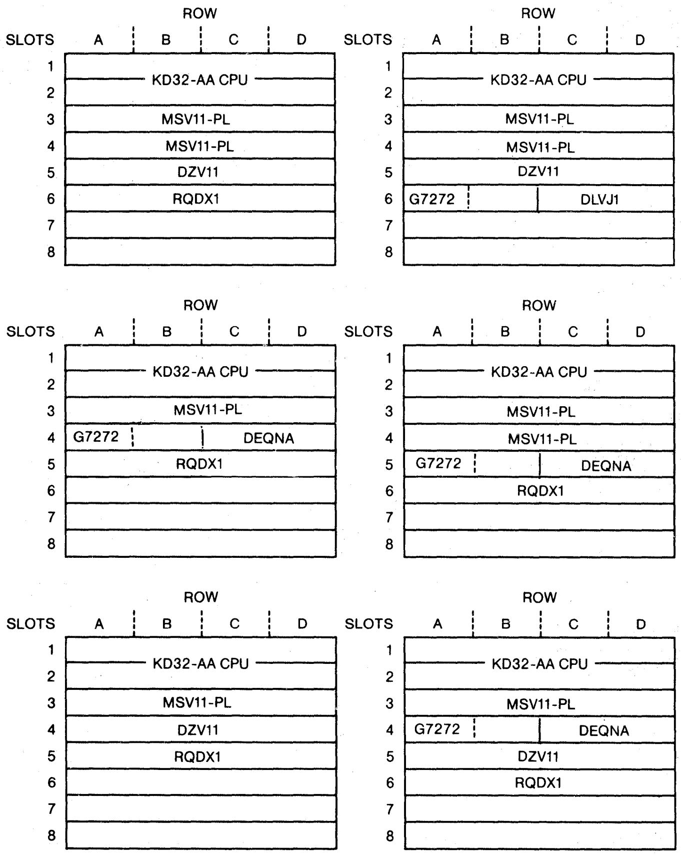


Figure C-3 ▪ Sample MicroVAX-module Assignments

I/O Distribution Panel Inserts

Each option that requires an external connection to a device includes a panel insert that mounts on the I/O distribution panel. All models are shown in Figure C-4. The MicroVAX I and MicroPDP-11 I/O distribution panel inserts are 1 × 4 inches and 2 × 3 inches. Blank panel covers are mounted where no panel insert is installed. The panel covers can be removed and the I/O distribution panel inserts installed.

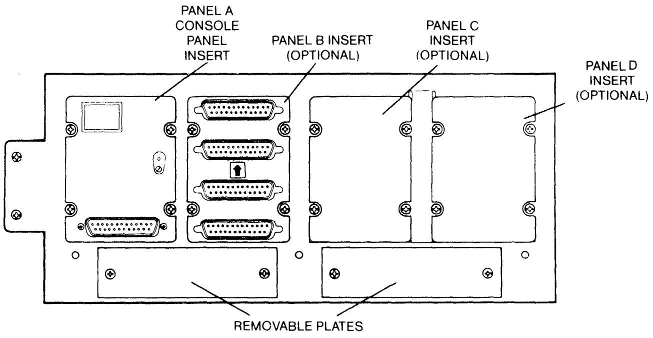

PANEL A CONSOLE PANEL INSERT
PANEL B INSERT (OPTIONAL)
PANEL C INSERT (OPTIONAL)
PANEL D INSERT (OPTIONAL)
REMOVABLE PLATES

Figure C-4 ▪ I/O Distribution Panel Inserts

▪ Additional Documentation

MicroVAX I Owner's Manual	*EK-KD32A-UG*
MicroPDP-11 Owner's Manual	*EK-OLCP5-OM*
PDP-11 Systems and Options Catalog	*ED-26051-41*
Terminal & Printers Handbook	*EB-23909-54*

Appendix D • Customer Services

Like all of Digital's products, the microsystems and their system software have been designed for reliability and manufactured to strict quality control standards that ensure that each unit meets its design goals. Digital's customer services organization is ready to follow up with quality support if it is required. Digital is the complete service vendor and has the products and tools to back its commitment to customer satisfaction.

Field Service

Digital's Field Service is committed to customer satisfaction through quality delivery of a complete range of service products. The service organization complements Digital's hardware offerings and makes a significant contribution to Digital's position as an industry leader.

The Digital Field Service organization includes over 16,000 service engineers who are backed by more than 2000 administration and support personnel. Backing each service engineer are resources and support programs that help each engineer to meet customer needs more effectively. Some of these resources and programs are a computerized logistics network, formalized training programs, an automated call-handling system, remote diagnosis, remote support, the site-management guide, and an action planning and problem escalation program.

Onsite Services

Onsite contract services are available for all microsystems, subject to minimum hardware configurations. These services provide corrective maintenance, preventive maintenance, and all applicable engineering changes to ensure that the microsystems and their options are operational and kept completely up to date. In addition to priority service, contractual maintenance allows Digital's customers to budget for their annual maintenance needs. The monthly contract charge covers all travel, labor, and materials. Users have a choice of tailored service agreements. In addition to basic coverage, extended hours are available to customers with critical applications that require special attention.

- DECSERVICE
  The DECservice agreement is Digital's most comprehensive onsite service product. It provides committed response time including a 4-hour service response if your system is located within 100 miles of a Digital service location. DECservice also provides continuous repairs until the problem is solved, a program of preventive maintenance, installation of the latest engineering changes, and automatic escalation for complex problems. DECservice also offers the customer a choice of coverage hours—up to 24 hours, seven days per week.

- BASIC SERVICE
  The Basic Service agreement is the best alternative for customers whose requirements do not demand a fixed response time to calls for remedial maintenance, or for continuous work to resolve system-down situations outside coverage hours. Basic Service offers economical, yet full-service coverage. Calls for service receive priority status, second only to DECservice calls. It also provides preventive maintenance, installation of the latest engineering changes, and automatic escalation of complex problems. The hours of coverage are during first-shift business hours.

- PER CALL SERVICE
  Per Call Service is a noncontractual, time and materials service. It is available on an onsite and offsite basis. Onsite service is available Monday through Friday during standard business hours, from 8:00 a.m. to 5:00 p.m. Offsite Per Call Service is available through mail-in board replacement and carry-in system repairs.

- SHARED MAINTENANCE SERVICE
  Shared Maintenance Service is a product that combines onsite and offsite services. It is offered to qualified customers who perform their own preventive and remedial maintenance provided that they meet certain prerequisites. The onsite support provided by Digital is similar to a DECservice agreement except that customers, after paying a fixed monthly fee (a percentage of the DECservice maintenance charge), pay only for labor (at the local Per Call rate) and materials as they are required. Shared Maintenance Service features include onsite repairs, committed response, branch telephone support (technical), emergency access to branch logistics, extended coverage (optional), and remote diagnosis (optional where available).

- DECOMPATIBLE SERVICE
  DECompatible Service is a product that provides standard onsite services to selected non-Digital hardware products that are attached to Digital systems. DECompatible Service is provided under DECservice, Basic Service, or Carry-In agreements, depending on the particular device. The level of service and response time under this program is the same quality service as that available for Digital's hardware products.

- RECOVER-ALL SERVICE
  Recover-All Service provides full product repair and/or replacement to Digital's hardware products that have been damaged due to accidents or incidents not covered under the other service agreements. Recover-All service expands the customer's service agreement to cover fire, water damage, natural disasters, power failure, sprinkler leakage, and theft. Recover-All also provides reimbursement for the cost of movement of equipment to a safe place, returning equipment to the site when safe conditions have been restored, removal of damaged equipment, transportation and installation of replacement equipment, replacement of fire protection chemicals, restoration of damaged Digital system software and customer data from backup disks and tapes, and data processing at a temporary location.

Offsite Services
Customers who do not require onsite services can take advantage of the Digital's offsite services that include Digital's Servicenters and DECmailer.

- SERVICENTERS
  The Digital Servicenter is a carry-in repair center for Digital's terminals and microsystems. The Servicenter offers low-cost repairs at over 160 convenient locations. At the Servicenter, the same quality service is provided that is given to offsite service calls. The Servicenter guarantees 2-day turnaround time. The customer may select from a variety of service offerings—contract, per call, or parts exchange. All Servicenter service and parts come with a 90-day warranty.

- DECMAILER
  DECmailer is a return-to-factory replacement service for customers who maintain their equipment at the module or subassembly level. It provides 5-day turnaround, free return shipping, 90-day warranty on service and parts, 24-hour emergency service, monthly billing, and quarterly activity reports.

▪ Software Services

Digital's Software Services are available to support customers during any phase of their system analysis, software development, or implementation efforts. These services start with the personal attention of a Digital software specialist and continue as long as the customer owns the system.

A Digital software specialist often works with a Digital sales representative to evaluate a prospective user's needs prior to purchase, in order to recommend hardware/software solutions appropriate to the customer's requirements. A full range of services is available to assist customers throughout the planning, implementation, and production phases of their systems.

Software Product Services

For most applications, resources are available to install software and provide support to assure that the purchased software products perform according to Digital's commitments. Software support is assured through a variety of *Software Product Services*, which offer customers the opportunity to keep their software up to date and running smoothly through the life of the software.

Software Product Services provide informational, preventive, and remedial service to help customers after the software is installed. These services provide updates to the latest software products, responses to inquiries on software, advice on software usage, and technical publications that contain programming notes and documentation corrections.

▪ DECSUPPORT SERVICE
DECsupport is the most comprehensive software product service available. DECsupport in^cludes all of the elements of Basic Service, plus onsite assistance and software support for critical situations.

▪ BASIC SERVICE
Basic Service is appropriate for users who do not require onsite support, and includes all of the elements of Self-Maintenance Service, telephone-support service, plus online support via the Digital Software Information Network (DSIN) for usage and remedial software questions. DSIN enables customers to receive software information and solutions to software problems by computer access to a Digital Customer Support Center. Currently, DSIN is available only in the U.S.A. and Canada.

▪ SELF-MAINTENANCE SERVICE
Self-maintenance Service enables users to maintain their own system software and provides tools that include media and documentation updates, formal software problem-reporting mechanisms, and newsletters and dispatches containing information about new software developments and enhancements.

Startup Service Packages

Software Product Services offers three comprehensive Startup Service Packages for support of the operating system. Each package provides media and documentation and one year of service for the operating system and for qualified layered products on that system. Each service package also allows customers to take advantage of several levels of training developed by Digital's Software Services and Educational Services organizations. Training materials and information are available upon purchase of the package.

- STARTUP SERVICE PACKAGE—LEVEL I

  Level I is appropriate for a technical staff requiring minimal training and having the skill to install and support the new system, using the Basic Service's telephone-support service to maintain the software at its most current level. Level I provides:

  - Basic Service
  - Media and documentation for the operating system and the most dependent software purchased concurrently
  - Training

- STARTUP SERVICE PACKAGE—LEVEL II

  Level II is appropriate for a technical staff that can support the new system after Digital has trained the staff, installed the product, and oriented the staff concerning system operation. Level II provides:

  - Basic Service
  - Media and documentation for the operating system and the most dependent software purchased concurrently
  - Installation and orientation
  - Training

- STARTUP SERVICE PACKAGE—LEVEL III

  Level III includes onsite software support when needed for critical situations and for more comprehensive training. Level III provides:

  - DECsupport Service
  - Media and documentation for the operating system and the most dependent software purchased concurrently
  - Installation and orientation
  - Training

Supplementary Service Options
Software Services provides supplementary options for microsystems already under a service agreement.

- MEDIA UPDATE SERVICE
  The Media Update Service is a subscription service that provides Software Product Services customers with a means of obtaining additional copies of machine-readable media for most operating systems and dependent products. Customers may choose from a variety of distribution media. A prerequisite for the service is that customers have a DECsupport, Basic, or Self-maintenance agreement with Digital.

- DOCUMENTATION UPDATE SERVICE
  The Documentation Update Service supplies service customers with the documentation-only portion of a Software Product Services agreements. The documentation delivered with this service is the portion of the document that was changed or revised since the last release.

- SERVICE RIGHT-TO-COPY
  This option allows customers with a Software Product Services agreement to automatically copy the updates to the product under agreement onto a single, additional CPU.

- ADDITIONAL TELEPHONE SUPPORT CENTER CONTACT SERVICE
  This service allows customer who have a Basic or DECsupport agreement to add names to the list of people entitled to call the Digital Telephone Support Center.

- ADDITIONAL SOFTWARE DISPATCH SUBSCRIPTION SERVICE
  Customers who have a Software Product Services agreement can obtain an additional copy of the dispatches and technical newsletters supplied under the agreement.

- SOFTWARE REVISION RIGHT-TO-COPY
  The Software Revision Right-to-Copy option allows customers to copy a single update to the product onto a single, additional CPU.

Installation Service

The purchase of installation as a separate service is appropriate in those instances in which there is no need to purchase a Startup Service Package or a need to have add-on dependent products installed. Installation Service ensures that customers have received all of the proper distribution materials, and that the system generation process for the operating system and/or dependent software products is completed.

DECstart Service

DECstart consists of several levels of fixed-price consulting services with a proven combination of direct assistance, documentation review, discussion, and hands-on experience provided at the customer's site by a Digital Software Specialist. The DECstart services are conducted over a 90-day period to assure mastery of the system. Programmers and system managers are taken step by step through the techniques required to operate their system effectively.

To supplement the DECstart Service, additional services are priced on a time and materials basis. An estimate can be given for any consultation that a customer may be considering. In addition, a Digital Software Specialist can draw up a Customer Support Plan to help the user determine any further areas in which additional services might be beneficial.

▪ Educational Services

Educational Services offers a complete range of training programs and services to support the microsystems software and hardware.

Training facilities are located in Japan, Australia, Great Britain, Germany, France, the Netherlands, Sweden, Italy, Canada, and throughout the United States. Services are centered around fully equipped regional education centers. These centers provide a staff of educators dedicated to providing quality education and training, and a variety of instructional formats that supports the learning pace of individuals as well as classrooms of individuals learning together.

Digital's Educational Services publishes a *Digest* every three months that includes a description and 6-month schedule of all software courses and a 9-month schedule of all hardware courses. It also includes the ordering information for the self-paced instruction material. The *Digest* may be ordered by contacting your sales representative or your nearest Digital Training Center.

Catalog courses are regularly scheduled classes offered at training centers. They cover the student range from first-time user to those needing highly specialized training on the theory of operation. Most catalog courses include extensive hands-on laboratory time, and all incorporate the use of a wide range of student workbooks, reference manuals, and other instructional materials.

Specialized training is available for users with unique applications or training situations. This training is designed to give the student the maximum relevant material for specific applications, while minimizing extraneous information. The customized courses are tailored to the individual customer's schedule and typically are presented in a series. The customized courses can be modified from existing courses or can be entirely new programs based on mutually agreed-upon objectives.

Customers with a group of individuals to train may find it more economical to have Educational Services conduct courses at the users' home site. Onsite instruction of both catalog and custom courses eliminates travel and other expenses incurred by students attending classes at training centers. This method of instruction further enhances training by allowing Digital instructors to emphasize points of particular value to the students' applications and operations.

By taking advantage of the latest in text-based, computer-based, and audiovisual-based techniques, Educational Services has developed a series of courses that offers self-paced instruction (SPI). These courses are convenient, self-contained, and modular. SPI format allows students to progress at their own rate, to study when and where they wish, and to "play back" or reread modules for review. SPI course material is available in several forms—computer media (such as magtape and flexible diskette), videotape, videocassette, audio/filmstrip cassette, and text—all supported by student guides and workbooks. Computer-based instruction (CBI) refers to courses that students take at their own pace at their terminal. CBI course material is available on 1600-bits/inch tape and TU58 cassette tape, as well as on diskette for use online.

Training Centers

The Digital Training Centers located in the United States are listed as follows. For additional information on courses, training materials, or other training center locations, call the following customer support number in Massachusetts: (617) 276-4373.

- CALIFORNIA
  Los Angeles Training Center
  4311 Wilshire Boulevard, Suite 400
  Los Angeles, California 90010-3779
  Telephone: (213) 937-3870

  Santa Clara Training Center
  2525 Augustine Drive
  Santa Clara, California 95051-7576
  Telephone: (408) 748-4048

- COLORADO
  Denver Training Center
  8085 South Chester Street
  Englewood, Colorado 80112-1478
  Telephone: (303) 649-3000

- ILLINOIS
  Chicago Training Center
  5600 Apollo Drive
  Rolling Meadows, Illinois 60008-4063
  Telephone: (312) 640-5520

- MASSACHUSETTS
  Boston Training Center
  12 Crosby Drive
  Bedford, Massachusetts 01730
  Telephone: (617) 276-4111

- NEW JERSEY
  New Jersey Training Center
  20 Corporate Place South
  Piscataway, New Jersey 08854-4237
  Telephone: (201) 562-4000

- NEW YORK
  New York Training Center
  One Penn Plaza
  New York, New York 10119-0031
  Telephone: (212) 971-3349

- TEXAS
  Dallas Training Center
  12100 Ford Road, Suite 110
  Dallas, Texas 75234-7231
  Telephone: (214) 888-2500

- WASHINGTON, D.C.
  Washington, D.C. Training Center
  8100 Corporate Drive
  Landover, Maryland 20785-2231
  Telephone: (301) 577-4300

Appendix E • Documentation

The MicroVAX I and MicroPDP-11 systems and software are supported by a comprehensive set of documents dedicated to their operation, programming, and maintenance. These manuals are periodically updated to include new developments and equipment and can be ordered through Digital's Peripheral and Supplies Group. The following lists contain the titles and associated Digital order numbers of documents that apply to the MicroVAX I and MicroPDP-11 family.

For more information on ordering these documents, contact the Peripherals and Supplies Group at (800) 258-1710, or by writing to:

Digital Equipment Corporation
Peripheral and Supplies Group
Continental Boulevard
Merrimack, New Hampshire 03054

• MicroVAX I Hardware Manuals

MicroVAX I CPU Technical Description	*EK-KD32A-TD*
MicroVAX I Owner's Manual	*EK-KD32A-UG*
VAX Architecture Handbook	*EB-19580-20*
VAX Hardware Handbook	*EB-21710-20*
PDP-11 Microcomputer Interfaces Handbook	*EB-23144-18*

• MicroVAX I Software Manuals

VAXELN Technical Summary	*EJ-30083-41*
VAX/VMS Technical Summary (includes MicroVMS)	*EJ-26070-48*
VAX Software Source Book (Volumes 1 and 2)	*EB-26125-46*
VAX Software Handbook	*EB-21812-20*

▪ MicroPDP-11 Hardware Manuals

KDJ11-A CPU Module User's Guide	*EK-KDJ1A-UG*
KDF11-B CPU Module User's Guide	*EK-KDFEB-UG*
DCJ11 Microprocessor User's Guide	*EK-DCJ11-UG*
PDP-11 Architecture Handbook	*EB-23657-18*
PDP-11 Microcomputer Interfaces Handbook	*EB-23144-18*
MicroPDP-11 System Technical Manual	*EK-OLCP5-TM*
MicroPDP-11 System Owner's Manual	*EK-OLCP5-OM*
PDP-11/23-PLUS System Manual	*EK-1T23B-OP*

▪ MicroPDP-11 Software Manuals

PDP-11 Software Handbook	*EB-25398-41*
RSX-11 Handbook	*EB-25742-41*
RSTS/E Handbook	*EJ-23534-18*
ULTRIX Software Guidebook	*EJ-26153-20*
PDP-11 Software Source Book (Volumes 1 and 2)	*EB-26030-41*

▪ Miscellaneous

PDP-11 Systems and Options Catalog	*ED-26051-41*
Networks and Communications Buyer's Guide	*ED-26127-42*
Networks Handbook	*EB-26013-42*
Terminals & Printers Handbook	*EB-23909-54*

Appendix F • MicroPDP-11/SV Overview

The MicroPDP-11/SV is the new entry-level, low-cost member of the microsystems family. It is compatible with all of the MicroPDP-11 systems and with all MicroPDP-11 software. The MicroPDP-11/SV uses the same CPU as the MicroPDP-11/23 system—the KDF11-B. It can address up to 4 Mbytes of parity MOS memory, and comes packaged with 800 Kbytes of flexible-disk storage and either 11 Mbytes or 31 Mbytes of fixed-disk storage. The MicroPDP-11/SV accepts all of the same options as the MicroPDP-11 family.

The MicroPDP-11/SV can support up to four concurrent, active users depending on the application. It is an ideal system for businesses that need to start small while providing for expanded computing solutions as business requirements grow.

The MicroPDP-11/SV is available only from Digital's OEMs, Authorized Dealers, and Business Centers.

MicroPDP-11/SV Features

- Contains a Q22-bus interface that supports block-mode DMA and up to 4 Mbytes of physical memory
- Uses the KDF11-B CPU module and all of the features associated with this processor. See Chapter 2—System Hardware for a more detailed description of the KDF11-B central processor.
- Contains the same chassis as the other microsystems but uses a four-slot backplane. See Chapter 2—System Hardware for a detailed description of the microsystem chassis.
- Has 512 Kbytes of parity MOS memory
- Includes an integral RX50 800-Kbyte dual-diskette drive, and an integral RD51 11-Mbyte or RD52 31-Mbyte fixed-disk drive
- Has two asynchronous serial line units for optional video or printing terminals
- Available in floorstand or tabletop models
- Supports optional floating-point and commercial instruction sets
- Supports all of the MicroPDP-11 family operating systems

▪ MicroPDP-11/SV Configurations

The MicroPDP-11/SV can be installed as a floorstand or tabletop system. Both of these configurations come with the MicroPDP-11/23 CPU and chassis, along with an RX50 800-Kbyte dual-diskette drive and the choice of an RD51 11-Mbyte or RD52 31-Mbyte fixed-disk drive. For more information on the RX50, RD51, and RD52 storage devices, refer to Chapter 3—System Options.

Optional equipment for the MicroPDP-11/SV includes a floating-point processor, the Commercial Instruction Set (CIS), a 256-Kbyte, 512-Kbyte, or 1-Mbyte memory module, additional communications devices, video and printing terminals, a country kit, and an operating-system general license.

▪ MicroPDP-11/SV System Specifications

Because the MicroPDP-11/SV is based on the MicroPDP-11/23 system, most hardware specifications such as system dimensions and power requirements are the same for both systems. Refer to Appendix C—Site Preparation and Installation for these specifications.

▪ Additional Information

For more information on the MicroPDP-11/SV systems, contact your local Digital sales office for the address of the Digital OEM, Authorized Digital Dealer, or Digital Business Center closest to you.

Glossary

This glossary provides definitions and explanations of common terms and abbreviations used in this handbook.

absolute-indexed mode: An indexed-addressing mode in which the base-operand specifier is addressed in absolute mode.

absolute-mode: Autoincrement-deferred mode in which the program counter is used as the register and contains the address of the location containing the actual operand.

access mode: Any of the four processor access modes in which software executes. Processor access modes are, in order from most to least privileged and protected—kernel, executive (MicroVAX I only), supervisor, and user. When the processor is in kernel mode, the executing software has complete control of, and responsibility for, the system. In any other mode, the processor is inhibited from executing privileged instructions.

access type: The way in which the processor accesses instruction operands or a procedure accesses its arguments.

address: A number used by the operating system and user software to identify a storage location.

address-access type: The specified operand of an instruction is not directly accessed by the instruction. The address of the specified operand is the actual instruction operand. The context of the address calculation is given by the data type of the operand.

addressing mode: The way in which an operand is specified.

address space: The set of all possible addresses available to a process.

ANSI: American National Standards Institute.

application: A specific program or task to which a computer solution can be applied.

application program: A computer program designed to meet specific user needs (i.e., a program that controls inventory or monitors a manufacturing process).

architecture: Computer architecture refers to the design or organization of the central processing unit (CPU).

argument pointer: It contains the address of the base of the argument list for procedures initiated using the call instructions.

ASCII (American Standard Code for Information Interchange): A code that assigns a binary number to each alphanumeric character and several nonprinting characters used to control printers and communication devices. ASCII characters are seven or eight bits long and may have an additional parity bit for error detection.

asynchronous transmission: Transmission in which time intervals between transmitted characters may be of unequal length.

autodecrement-indexed mode: An indexed-addressing mode in which the base-operand specifier uses autodecrement-mode addressing.

autodecrement mode: The contents of the selected register are decremented, and the result is used as the address of the actual operand of the instruction. The contents of the register are decremented according to the data type context of the register.

autoincrement-deferred indexed mode: The specified register contains the address of a longword that contains the address of the actual operand. The contents of the register are incremented by four which are the number of bytes in a longword. If the program counter is used as the register, this mode is called absolute mode.

autoincrement-indexed mode: An indexed-addressing mode in which the base-operated specifier uses autoincrement-mode addressing.

autoincrement mode: The contents of the specified register are used as the address of the operand; then the contents of the register are incremented by the size of the operand.

automatic-dialing unit: A device capable of automatically generating dialing digits.

automatic-calling unit: A dialing device supplied by the communications common carriers that permits a business machine to dial calls automatically over the communications networks.

base-operand address: The address of the base of a table or array reference by indexed-mode addressing.

base-operand specifier: The register used to calculate the base-operand address of a table or array referenced by indexed-mode addressing.

base register: A general register that contains the address of the first entry in a list, table, array, or other data structure.

BASIC (Beginners All-purpose Symbolic Instruction Code): A widely used interactive programming language developed by Dartmouth College that is especially well suited to personal computers and beginning users.

batch processing: The technique of executing a set of computer programs without human interaction or direction during its execution.

baud: A unit of data transmitting/receiving speed, approximately equal to a single bit per second.

bidirectional printing: A printing terminal technique to increase printing throughput by printing every other line from right to left, thus saving the carriage return time.

binary digit (bit): In binary notation either of the characters 0 or 1. "Bit" is the commonly used abbreviation for binary digit.

BISYNC: IBM's 1968 Binary Synchronous Communications protocol (BSC).

bit: Abbreviation for binary digit.

bit complement (also called one's complement): The result of exchanging 0s and 1s in the binary representation of a number. The bit complement of the binary number 11011001 is 00100110. Bit complements are used in place of their corresponding binary numbers in some arithmetic computations in computers.

bit-map graphics: A technology that allows control of individual pixels on a display screen to produce graphic elements of superior resolution, permitting accurate reproduction of arcs, circles, sine waves, or other curved images that block-addressing technology cannot accurately display.

bit string: See *variable-length bit field.*

bit-transfer rate: The number of bits transferred per unit of time, usually expressed in bits per second.

block: A group of bits transmitted as a unit. A coding procedure is usually applied for synchronization or error-control purposes.

branch access type: An instruction attribute that indicates that the processor does not reference an operand address, but that the operand is a branch displacement. The size of the branch displacement is given by the data type of the operand.

buffer: A place where data can be stored temporarily. Terminals can store data in a buffer if data is received faster than it can be processed or displayed.

bus: A group of parallel electrical connections that carry signals between computer components or devices.

byte: Eight contiguous bits starting at an addressable byte boundary.

cache memory: A small, high-speed memory placed between slower main memory and the processor. A cache increases effective memory-transfer rates and processor speed. It contains copies of data recently used by the processor and fetches several bytes of data from memory in anticipation that the processor will access the next sequential series of bytes.

call frame: See *stack frame.*

call instructions: The processor instructions CALLG (call procedure with general argument list) and CALLS (call procedure with stack argument list) on the MicroVAX I.

call stack: The stack, and conventional stack structure, used during a procedure call. Each access mode of each process context has one call stack and interrupt-service context has one call stack.

carrier: A continuous frequency capable of being modulated or impressed with a signal.

CCITT (Commite Consultatif International de Telegraphie et Telephonie): An international consultative committee that sets international communications usage standards.

character: A symbol represented by an ASCII code. A single, printable letter (a through z), numeral (0 through 9), or symbol (%) (.) ($) (,) used to represent data.

character printer: A printer, similar to a typewriter, that prints one character at a time.

character string: A contiguous set of bytes. A character string is identified by two attributes—an address and a length. Its address is the address of the byte containing the first character of the string. Subsequent characters are stored in bytes of increasing addresses. The length is the number of characters in the string.

COBOL (Common Business-Oriented Language): A high-level programming language that is well suited to business applications involving complex data records and large amounts of printed output.

command: An instruction, typed by the user at a terminal or included in a command file, that requests the software monitoring a terminal or reading a command file to perform some well-defined activity.

command procedure: A file containing commands and data that the command interpreter can accept in lieu of the user's typing the commands individually on a terminal.

communications link: The physical connection, typically a phone line, between a terminal and a computer or another peripheral device.

compatibility: The ability of an instruction, source language, or peripheral device to be used on more than one computer.

compatibility mode: A mode of execution that enables the central processor to execute nonprivileged PDP-11 instructions.

computer network: An interconnection of computer systems, terminals, and communications facilities.

concentrator: A communications device that provides communications capability between many low-speed, asynchronous channels and one or more high-speed, synchronous channels.

condition: An exception condition detected and declared by software.

condition codes: Four bits in the processor status word that indicate the results of previously executed instructions.

condition handler: A process that requests the system to execute when an exception condition occurs.

console terminal: The terminal connected to the central processor used to control the computer system.

context: Also called process state. See *hardware context.*

context indexing: The ability to index through a data structure automatically because the size of the data type is known and is used to determine the offset factor.

context switching: Interrupting the activity in progress and switching to another activity. Context switching occurs as one process after another is scheduled for execution.

CPU (Central Processing Unit): Commonly called a computer. A set of electronic components that control the transfer of data and perform arithmetic and logic calculations.

CRC (Cyclic Redundancy Check): An error-detection scheme in which the check character is generated by taking the remainder after dividing all the serialized bits in a block of data by a predetermined binary number.

CRT terminal: Another name for a video terminal.

current-access mode: The processor access mode of the currently executing software. The current-mode field of the processor status longword (PSL) indicates the access mode of the currently executing software.

cursor: A distinctive mark on a video terminal screen, such as a flashing square or underline, that indicates where the next character will be displayed.

D_floating data: Eight contiguous bytes starting on an addressable-byte boundary, that are interpreted as containing a floating-point number.

daisywheel: A printhead that forms full characters rather than characters formed of dots. It is shaped like a wheel with many spokes, with a letter, numeral, or symbol at the end of each spoke.

data communications: The interchange of data messages from one point to another over communications channels.

data type: The way in which bits are grouped and interpreted. In reference to the processor instructions, the data type of an operand identifies the size of the operand and the significance of the bits in the operand.

DDCMP (Digital Data Communications Message Protocol): A uniform discipline for the transmission of data between stations in a point-to-point or multipoint-data communications system. The method of physical-data transfer used may be parrallel, serial synchronous, or serial asynchronous.

DECnet: Digital communications networks.

device interrupt: An interrupt received on processor-priority levels. Device interrupts can be requested only by devices, controllers, and memories.

device name: The field in a file specification that identifies the device unit in which a file is stored.

device register: A location in device controller logic used to request device functions such as I/O transfers and/or to report status.

device unit: One drive, and its contolling logic, of a mass storage device system. A mass-storage system can have several drives connected to it.

diagnostic: A program that tests logic and reports any faults it detects.

direct-mapping cache: A cache organization in which only one address comparison is needed to locate any data in the cache because any block of main-memory data can be placed in only one possible position in the cache.

diskette: A flexible, flat, circular plate permanently housed in a paper envelope with magnetic coating that stores data and software.

displacement mode: The specifier extension is a byte, word, or longword displacement. The displacement is sign extended to 32 bits and added to a base address obtained from the specified register. The result is the address of the actual operand.

displacement-deferred mode: The specifier extension is a byte, word, or longword displacement. The displacement is sign-extended to 32 bits and added to a base address obtained from the specified registers. The result is the address of a longword that contains the address of the actual operand.

displacement-deferred indexed mode: An indexed-addressing mode in which the base-operand specifier uses displacement-deferred mode addressing.

displacement-indexed mode: An indexed-addressing mode in which the base-operand specifier uses displacement-mode addressing.

distributed-data processing: A computing approach in which an organization uses computers in more than one location, rather than one large computer in a single location.

DMA (Direct Memory Access): A facility that permits I/O transfers directly into or out of memory without passing through the processor's general registers; performed either independently of the processor or on a cycle-stealing basis.

DNA (Digital Network Architecture): A hardware and software scheme for interconnecting Digital's computers in a network. It is composed of three elements—Data Access Protocol (DAP), Network Services Protocol (NSP), and Digital Data Communications Message Protocol (DDCMP). See also *DDCMP.*

dot-matrix printing: A printing technique that forms characters from a two-dimensional array of dots.

double-floating data: See *D_floating data.*

draft-quality printer: A printer that produces characters that are readable, but of less than typewriter quality.

drive: The electromechanical unit of a mass-storage device system on which a recording medium (disk cartridge, disk pack, or magnetic-tape reel) is mounted.

EBCDIC (Extended Binary Coded Decimal Interchange Code): An 8-bit character code used primarily in IBM equipment. The code provides for 256 different bit patterns.

editor: A program that interacts with the programmer to enter new programs into the computer and edit them as well as modify existing programs. Editors are language-independent and can edit anything in alphanumeric representation.

effective address: The address obtained after indirect- or direct-indexing modifications are calculated.

EIA (Electronic Industries Association): A standards organization specializing in the electrical and functional characteristics of interface equipment.

error: Any discrepancy between a computed, observed, or measured quantity and the true, specified, or theoretically correct value or condition.

event: A change in process status or an indication of the occurrence of some activity that concerns an individual process or cooperating processes. An incident reported to the scheduler that affects a process's ability to execute.

event flag: A bit in an event flag cluster that can be set or cleared to indicate the occurrence of the event associated with that flag. Even flags are used to synchronize activities in a process or among many processes.

exception: An event detected by the hardware (other than an interrupt or jump, branch, case, or call instruction) that changes the normal flow of instruction or set of instructions. There are three types of hardware exceptions—traps, faults, and aborts.

exception condition: A hardware- or software-detected event other than an interrupt or jump, branch, case, or call instruction that changes the normal flow of instruction execution.

exception enables: See *trap enables.*

exception vector: See *vector.*

executive mode: The second most privileged processor-access mode. The Record Management Services (RMS) and many of the operating system's programmed-service procedures execute in executive mode.

F_floating data: Four contiguous bytes starting on an addressable byte boundary. The bits are labeled from right to left 0 to 31. A two-word floating-point is identified by the address of the byte containing bit 0.

fanfold paper: A continuous sheet of paper whose pages are folded accordion-style and separated by perforations.

fault: A hardware-exception condition that occurs in the middle of an instruction and that leaves the registers and memory in a consistent state.

field: (1) See *variable-length bit field.* (2) A set of contiguous bytes in a logical record.

file: A collection of logically related records or data treated as a single item. A file is the means by which data is stored on a disk or diskette so it can be used at a later date.

floating (point) data: See *F_floating data.*

floppy disk: See *diskette.*

font: A complete set of letters, numerals, and symbols of the same typestyle of a given typeface.

formfeed: A device that automatically advances a roll of fanfold paper to the top of the next page or form when the printer has finished printing the previous form.

FORTRAN (Formula Translation): A widely used high-level programming language well suited to problems that can be expressed in terms of algebraic formulas. It is generally used in scientific applications.

frame pointer (FP): FP contains the base address of the most recent call frame on the stack.

full-duplex: Describes a communications channel on which simultaneous two-way communications are available.

function key: A key that causes a computer to perform a function such as clearing the screen or executing a program.

G_floating data: Eight contiguous bytes starting on an arbitrary-byte boundary. The bits are labeled from the right 0 through 63. A G_floating data is specified by its address A, the address of the byte containing bit 0.

general register: Any of the registers used as the primary operands of the native-mode instructions.

graphics: The use of lines and figures to display data in contrast with the use of printed characters.

half-duplex: A communications channel on which only one-way communications are permitted at a time. The line can be "turned around" to allow data to flow the other way. Some half-duplex links provide a special "reverse channel" in the direction opposite to the flow of data that permits transmission of control signals only.

hardcopy: Hardcopy refers to paper printout, as opposed to video displays that cannot be saved.

hardware context: The values contained in the following registers while a process is executing—the program counter, the processor status longword, the general registers, the processor registers that describe the process virtual-address space, the stack pointer for the current-access mode in which the processor is executing, plus the contents to be loaded in the stack pointer for every access mode other than the current-access mode.

Hertz (Hz): A unit of frequency equal to one cycle per second. Cycles are referred to as Hertz in honor of the physicist Heinrich Hertz.

host computer: A computer attached to a network that provides such services as computation, database access, or special programs or programming languages.

I/O (Input/Output): Pertaining to devices that accept data for transmission to a computer system (input) and that accept data from a computer system for transmission to a user or process (output).

image: Procedures and data that have been bound together by the linker. There are three types of images—executable, shareable, and system.

immediate mode: Autoincrement-mode addressing in which the program counter is used as the register.

indexed-addressing mode: Two registers are used to determine the actual instruction operand—an index register and a base-operand specifier. The contents of the index register are used as an index (offset) into a table or array. The base-operand specifier supplies the base address of the array (called the base-operand address or BOA).

index register: A register used to contain an address offset.

input stream: The source of commands and data. One of either the user's terminal, the batch stream, or an indirect-command file.

instruction: A command that tells the computer what operation to perform next.

instruction buffer: An 8-byte buffer in the processor used to contain bytes of the instruction currently being decoded and to prefetch instructions in the instruction stream. The control logic continuously fetches data from memory to keep the 8-byte buffer full.

integral modem: A modem built into a terminal rather than packaged separately.

integrated circuit (IC): A complete electrical circuit on a single chip.

interface: An electronic assembly that connects an external device, such as a printer, to a computer.

interleaving: Assigning consecutive physical memory addresses alternately between two memory controllers.

interrupt: An event other than an exception or branch, jump, case, or call instruction that changes the normal flow of instruction execution. Interrupts are generally external to the process executing when the interrupt occurs.

interrupt-service routine: The routing executed when a device interrupt occurs.

interrupt stack: The systemwide stack used when executing in interrupt-service context. At any time the processor is either in a process context executing in user, supervisor, executive, or kernel mode, or in systemwide interrupt-service context operation with kernel privileges, as indicated by the interrupt stack and current mode bits in the processor status longword. The interrupt stack is not context-switched.

interrupt-stack pointer: The stack pointer for the interrupt stack. Unlike the stack pointers for process-context stacks, the interrupt stack pointer is stored in an internal register.

interrupt vector: See *vector.*

kernel mode: The most privileged processor access mode. The operating system's most privileged services, such as I/O drivers and the pager, run in kernel mode.

letter-quality printer: A printer that produces printing comparable in quality to that achieved by a typewriter.

linefeed: The printer operation that advances the paper by one line.

literal mode: The instruction operand is a constant whose value is expressed in a 6-bit field of the instruction.

local: Handwired connection of a computer to another computer, terminal, or peripheral device, such as in a local area network.

longitudinal redundancy check (LRC): An error-checking technique based on an accumulated exclusive-OR of transmitted characters.

longword: Four contiguous bytes starting on an addressable byte boundary. Bits are numbered from right to left 0 through 31. The address of the longword is the address of the byte containing bit 0.

main memory: See *physical memory.*

mass-storage device: A device capable of reading and writing data on mass-storage media such as a diskpack or a magnetic-tape reel.

memory management: The system functions that include the hardware's page mapping and protection and the operating system's image activator and pager.

mnemonic: A short, easy-to-remember name or abbreviation.

modem (Modulator/Demodulator): A device that converts digital data from a terminal or CPU into analog signals for transmission over telephone lines and convert the receiver data back to digital format.

MOS (Metal-Oxide Semiconductor): The most common form of LSI technology.

multiplexer: A device for connecting a number of communications lines to a computer.

multiprogramming: A scheduling technique that allows more than one job to be in an executable state at any one time, so even with one CPU more than one program can appear to be running at a time because the CPU is giving small slices of its time to each executable program.

native mode: The processor's primary execution mode.

network: A group of computers that are connected to each other by communications lines to share information and resources.

node: An end point of any branch of a network, or a junction common to two or more branches of a network.

numeric string: A contiguous sequence of bytes representing up to 31 decimal digits (one per byte) and possibly a sign.

octaword: A set of 16 contiguous bytes starting at an arbitrary-byte boundary.

offset: A fixed displacement from the beginning of a data structure.

one's complement: See *bit complement.*

opcode: The pattern of bits within an instruction that specifies the operation to be performed.

operand specifier: The pattern of bits in an instruction that indicates the addressing mode, a register, and/or displacement, that taken together identify an instruction operand.

operand-specifier type: The access type and data type of an instruction operand(s).

operating system: A collection of computer programs that control the overall operation of a computer and perform such tasks as assigning places in memory to programs and data, processing interrupts, scheduling jobs, and controlling the overall input/output of the system.

packed decimal: A method of representing a decimal number by storing a pair of decimal digits in one byte.

packed-decimal string: A contiguous sequence of up to 16 bytes interpreted as a string of nibbles. Each nibble represents a digit, except the low-order nibble of the highest-addressed byte, which represents the sign.

packet switching: A data transmission process that utilizes addressed packets in which a channel is occupied only for the duration of transmission of the packet.

page: A set of 512 contiguous byte locations used as the unit of memory mapping and protection or the data between the beginning of file and a page marker, between two markers, or between a marker and the end of a file.

paging: The action of bringing pages of an executing process into physical memory when referenced. When a process executes, all of its pages are said to reside in virtual memory.

parity: A common technique for error detection in data transmission. Parity-check bits are added to the data so that each group of data bits include an even number of "ones" for even parity and an odd number for odd parity.

peripheral: A device that is external to the CPU and main memory (i.e., printer, modem, or terminal), but connected to it by appropriate electrical connections.

physical address: The address used by hardware to identify a location in physical memory or on a directly addressable secondary-storage device such as a disk.

physical-address space: The set of all possible physical addresses that can be used to refer to locations in memory space or I/O space.

physical memory: The memory contained in the CPU memory modules.

pixels: Definable locations on a display screen that are used to form images on the screen. For graphics displays, screens with a large number of pixels generally provide higher resolution.

point-to-point connection: A network configuration in which a connection is established between two terminal installations.

polling: A technique for determining the order in which nodes take turns accessing the network. This is done so that access collision can be avoided.

port: A location on the CPU where physical connection is made between the central computer and a terminal, printer, modem, another computer, or a communications line.

position-dependent code: Code that can execute properly only in the locations in virtual-address space that are assigned to it by the linker.

position-independent code: Code that can execute properly without modification wherever it is located in virtual-address space.

printhead: The element in a printer that forms a printed character.

printout: An informal expression referring to almost anything printed by a peripheral device; any computer-generated hardcopy.

printwheel: See *daisywheel.*

privileged instructions: Any instruction intended for use by the operating system or privileged-system programs.

procedure: A routine entered via a call instruction. See also *command procedure.*

process: The basic entity scheduled by the system software that provides the context in which an image executes. A process consists of an address space and both hardware and software contexts.

process address space: See *process space.*

process context: The hardware and software contexts of a process.

process space: The lowest-addressed half of virtual-address space, where process instructions and data reside. Process space is divided into a program region and a control region.

processor: The functional part of the computer system that reads, interprets, and executes instructions. See also *central processing unit.*

processor register: A part of the processor used by the operating system software to control the execution states of the computer system.

processor status longword (PSL): A system-programmed processor register consisting of a word of privileged-processor status and the processor status word.

processor status word (PSW): The low-order word of the processor status longword.

program: A complete sequence of instructions and routines needed to solve a problem or to execute directions in a computer.

program counter (PC): At the beginning of an instruction's execution, the program counter normally contains the address of a location in memory from which the processor will fetch the next instruction it will execute.

program disk: A disk containing the instructions of a program.

programming language: The words mnemonics, and/or symbols, along with the specific rules allowed in constructing computer programs. Some examples are BASIC, FORTRAN, and COBOL.

protocol: A formal set of conventions governing the format and relative timing of message exchange between two communicating processes.

PSTN (Private Switched Telephone Network): Generic term for European telephone carriers.

quadword: Eight contiguous bytes (64 bits) starting at an addressable-byte boundary.

queue: (n.) A circular, doubly linked list. (v.) To make an entry in a list or table.

RAM (Random Access Memory): Memory that can both be read and written into (i.e., altered) during normal operation. RAM is the type of memory used in most computers to store the instructions of programs currently being run.

read-access type: An instruction- or procedure-operand attribute indicating that the specified operand is only read during instruction or procedure execution.

realtime: Refers to computer systems or programs that perform a computation during the actual time that a related physical process transpires.

ReGIS (Remote Graphics Instruction Set): Digital's graphics command interface to terminals for putting shapes on the terminal screen.

register: A storage location in hardware logic other than main memory. See also *general register; processor register; device register.*

register-deferred indexed mode: An indexed-addressing mode in which the base-operand specifier uses register-deferred mode addressing.

register-deferred mode: In register-deferred mode addressing, the contents of the specified register are used as the address of the actual instruction operand.

register mode: In register-mode addressing, the contents of the specified register are used as the actual instruction operand.

remote: Communications between computer and terminals via switched lines such as telephone lines.

reverse video: A feature on a display unit that produces the opposite combination of characters and background from that which is usually employed, that is, white characters on a black screen, if having black characters on a white screen is normal.

ROM (Read-only Memory): Memory containing fixed data or instructions that is permanently loaded during the manufacturing process.

scrolling: When a video terminal's screen is full, a new line of data can be displayed by adding it at the bottom of the screen and shifting all the previous lines upward, discarding the top line. When the upward movement is continuous rather than in line steps, it is called smooth scrolling.

serial transmission: A method of information transmission in which each bit of information is sent sequentially on a single path rather than simultaneously as in parallel transmission.

SNA (System Network Architecture): A network architecture of IBM.

software: A set of computer programs, procedures, rules and associated documentation concerned with the operation of network computers (i.e., compilers, monitors, editors, utility programs).

software interrupt: An interrupt generated on processor-priority levels that can be requested only by software.

stack: An area of memory set aside for temporary storage or for procedure- and interrupt-service linkages.

stack frame: A standard data structure built on the stack during a procedure call, starting from the location addressed by the frame pointer and going to lower addresses, and popped off during a return from procedure. Also called call frame.

stack pointer (SP): A general register that contains the address of the top (lowest address) of the processor-defined stack. Reference to stack pointer will access one of the five possible stack pointers—kernel, executive, supervisor, user, or interrupt—depending on the value in the current mode and interrupt stack bits in the processor longword.

status code: A longword value that indicates the success or failure of a specific function.

supervisor mode: The third most privileged processor access mode. The operating system's command interpreter runs in supervisor mode.

synchronous: A technique in which data bits are sent at precisely timed intervals. Synchronous channels are capable of higher data rates than asynchronous ones, often running at 56,000 bits per second.

system-address space: See *system space.*

system space: The higher-addressed half of virtual-address space.

system-virtual address: A virtual address identifying a location mapped by an address in system space.

system-virtual space: See *system space.*

terminal: The general name for those peripheral devices that have keyboards and video screens or printers.

timesharing: A mode of data processing that allows many terminal users to utilize a computer's resources to perform a variety of tasks simultaneously.

tool kit: The software and hardware components, including documentation, that are manufactured by Digital to help software developers create application programs that can be fully intergrated into computers.

tractorfeed: An attachment used to move paper through a printer. The roller that moves the paper has sprockets on each end that fit into the fanfold paper's matching pattern of holes.

translation buffer: An internal processor cache containing translations for recently used virtual addresses.

trap: An exception condition that occurs at the end of the instruction that caused the exception. The program counter saved on the stack is the address of the next instruction that would normally have been executed. All software can enable and disable some of the trap conditions with a single instruction.

trap enables: Bits in the processor status word that control the processor's action on certain arithmetic exceptions.

two's complement: A binary representation for integers in which a negative number is one greater than the bit complement of the positive number.

typeface: See *font.*

user mode: The privileges granted a user by the system manager.

variable-length bit field: A set of 0 to 32 contiguous bits located arbitrarily with respect to byte boundaries.

vector: An interrupt or exception vector is a storage location that contains the starting address of a procedure to be executed when a given interrupt or exception occurs. The system defines separate vectors for each interrupting device controller and for classes of exceptions.

virtual address: A 16-bit or 32-bit integer identifying a byte location in virtual-address space. The memory-management hardware translates a virtual address to a physical address. The term virtual address may also refer to the address used to identify a virtual block on a mass-storage device.

virtual-address space: The set of all possible virtual addresses that an image executing in the context of a process can use to identify the location of an instruction of data.

virtual memory: The set of storage locations in physical memory and on disk that are referred to by virtual addresses.

virtual-page number: The virtual address of a page of virtual memory.

word: Two contiguous bytes (16 bits) starting at an addressable-byte boundary.

Index