RSX-20F System Specifications

Order Number: AA-H213A-TK

November 1978

These specifications describe RSX-20F, the operating system that runs on the PDP-11/40 front-end processor of KL10-based computers. RSX-20F loads the KL10 microcode, configures main and cache memory, loads the boot program, and performs diagnostics. For systems that are running TOPS-20, RSX-20F also provides device handling for unit record equipment.

		1080/90	1091	2040/50	2060
OPERATING SYSTEM:	TOPS-10	6.03A	6.03A		
	TOPS-20			3A	3A
SOFTWARE:	RSX-20F (TOPS-10)	VA10-20	VE12-16		
	RSX-20F (TOPS-20)			VB12-32	VB12-34

To order additional copies of this document, contact the Software Distribution Center, Digital Equipment Corporation, Maynard, Massachusetts 01754

digital equipment corporation · maynard, massachusetts

First Printing, November 1978

The information in this document is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation. Digital Equipment Corporation assumes no responsibility for any errors that may appear in this document.

The software described in this document is furnished under a license and may only be used or copied in accordance with the terms of such license.

No responsibility is assumed for the use or reliability of software on equipment that is not supplied by DIGITAL or its affiliated companies.

The postage-prepaid READER'S COMMENTS form on the last page of this document requests the user's critical evaluation to assist us in preparing future documentation.

CONTENTS

Page

CONTENTS (CONT.)

FIGURES

TABLES

PREFACE

This document is a preliminary collection of information about the RSX-20F front-end operating system. It represents an interim step in the process of publishing an RSX-20F System Reference Manual.

The audience for this document is presumed to be software support specialists, field service personnel, and systems programmers with some knowledge of PDP-11s and the RSX-11M operating system.

Chapter 1 contains introductory material on the ancestry of RSX-20F, the front-end system tasks, and the front-end file system.

Chapter 2 contains information on the structure of RSX-20F, the management and scheduling of system tasks, and the monitoring of system events via system traps.

Chapter 3 contains information on the RSX-20F utility programs.

Chapter 4 contains information on error detection, error recovery, and error reporting.

Chapter 5 contains information on the DTE20 hardware operations such as Deposit/Examine of KL10 memory, TO10/TO11 transfer, doorbell function, and diagnostic operation.

Appendix A contains a listing and short description of the front-end tasks.

Appendix B contains a procedure for transferring files between TOPS-10 and RSX-20F.

Appendix C contains a list of the RSX-20F stop codes.

RSX-20F is a modification of RSX-11M, a PDP-11 operating system. Readers who are not familiar with the PDP-11 hardware and software will find the following three handbooks quite useful. The material in them is both tutorial and reference.

PDP-11 Processor Handbook	EB 05138 76
PDP-11 Software Handbook	EB 09798 78
PDP-11 Peripherals Handbook	EB 05961 76

This document does not include information about the KLINIT task used by RSX-20F to start the KL10 processor. This document also does not discuss KLINIK, the sub-mode of console terminal operation that provides remote diagnostic access via a communications line. Information about KLINIT and KLINIK can be found in the following manuals:

DECsystem-10 Operator's Guide (KL Series) Updates	AA-5104A-TB AD-5104A-T1,T2,T3
DECSYSTEM-20 Operator's Guide Update	AA-4176C-TM AD-4176C-T1

CHAPTER 1

INTRODUCTION

RSX-11M and RSX-11D, two PDP-11 operating systems, provided the base upon which RSX-20F was built. These operating systems were chosen for their ability to satisfy the needs of the KL10 front end, which are:

1. Be small and efficient.

2. Handle special cases such as booting the KL10 or diagnosing errors on the KL10.

3. Handle the unit record devices of TOPS-20 and TOPS-10/1091.

The purpose of the KL10 front end is to take some of the load off the KL10; specifically, the KL10 handles booting, configuring and loading the KL, and driving the unit-record and terminal hardware.

This chapter presents important concepts of PDP-11 software, explains the needs of RSX-20F, and describes how RSX-11M and RSX-11D were modified to produce RSX-20F.

1.1 THE PDP-11

The PDP-11 has several unique features that make it an easy machine to program and use.

1.1.1 The UNIBUS

The UNIBUS is a 56-line bus used to send addresses, data, and control information to the system components and peripherals. The method of communication is the same for every device on the UNIBUS, including memory and the central processor. Each device, including memory locations, processor registers, and peripheral-device registers, is assigned an address on the UNIBUS. Thus, peripheral-device registers can be manipulated as easily as main memory by the central processor. The UNIBUS is both bidirectional and asynchronous; this allows devices of varying speeds to be connected to it.

1.1.2 The I/O Page

The UNIBUS permits a new concept of memory. All the device registers are assigned an address in an area at the high end of memory. This area is called the I/O page.

1.1.3 Vector Interrupts

Each device on the UNIBUS also has two words assigned to it in low memory to handle interrupts for that device. The first word is the address of the interrupt routine. The second word contains the processor status word to be installed when control is transferred to the interrupt routine. This method allows the user to control interrupt handling as easily as depositing into memory.

1.1.4 Priorities

Each device has a priority level that it can interrupt on. In the processor status word, the priority level field (bits 5-7) can be set to a value of 0 through 7. Only devices with a priority higher than the priority in the status word can interrupt. The user, therefore, can control interrupts by depositing into memory.

1.1.5 Traps

PDP-1ls are set up to handle many types of traps, either synchronous or asynchronous. Synchronous traps occur immediately upon the issuance of an illegal instruction or general trap. They are dispatched via the vectors in low memory and provide user flexibility. Asynchronous traps occur independently of user instructions, usually as the result of I/O completion.

1.1.6 BR/NPR Requests

Transfer of data can be done in two ways via the UNIBUS: BR or NPR. BR stands for Bus Request and is the normal method used. The device wanting to use the UNIBUS must first request the use of the bus from the bus master. NPR (Non-Processor Request) parallels Direct Memory Access (DMA) on the KL10. An NPR request steals UNIBUS cycles without directly gaining control of the bus. It is much faster than sending data via the BR mechanism.

1.1.7 General Registers

The central processor has eight registers for general use. The registers can be used as accumulators, index registers, or stack pointers. Register 6 is used as the stack pointer (SP) and Register 7 is used as the machine's program counter (PC).

1.1.8 Stacks

The PDP-11 is a stack-oriented machine. It contains built-in addressing modes designed to manipulate stacks.

1.1.9 Instruction Set

The PDP-11 instruction set operates on single or double byte operands. Addressing on the PDP-11 is by eight-bit bytes with a word size being sixteen bits. Addressing includes a variety of addressing modes which when combined with the instruction set allow the programmer great flexibility in programming.

1.2 RSX-11M OPERATING SYSTEM

RSX-11M is a PDP-11 operating system. It controls I/O, schedules tasks to be run, and provides common subroutines. The resident operating system is referred to as the Monitor or Executive.

1.2.1 Directives

A directive is a request to the Executive to perform a function. Directives can perform I/O functions, obtain task and system information, suspend and resume task execution, and cause a task to exit. They are equivalent to JSYSs in TOPS-20 and UUOs in TOPS-10.

1.2.2 Device Drivers

A device driver is a program that controls physical hardware activities on a peripheral device. The device driver is generally the device-dependent interface between a device and the common, device-independent I/O code in an operating system.

1.2.3 Significant Events

A significant event is an event or condition which indicates a change in the system status of an event-driven system. A significant event is declared, for example, when an I/O operation completes. A declaration of a significant event indicates that the Executive should review the eligibility of task execution, because the event might have unblocked the execution of a higher priority task. The following are considered to be significant events:

- queueing of I/O requests
- completion of I/O requests
- requesting a task
- scheduling a task
- waking up a task
- exiting a task

There are 64 significant event flags and most of them are directly related to servicing directives. These flags can also be used by tasks to communicate with other tasks.

1.2.4 Mapped and Unmapped Systems

A mapped system uses hardware memory management to relocate virtual memory addresses. This provides greater flexibility in contructing programs. An unmapped system, however, must be assembled with the correct physical addresses because it has no hardware to relocate virtual addresses into physical addresses. RSX-20F is an unmapped system.

1.3 TASKS

A task is the fundamental executable unit in PDP-11 operating systems. However, this does not mean that all tasks are self-sufficient. Some tasks must call other tasks to complete their function. Others are considered subroutines to be called by still other tasks. Some tasks are self-sufficient and can be thought of in much the same way as a program in TOPS-20 or TOPS-10.

Tasks can reside in one of two places: the resident Executive partition or the general (GEN) partition. A partition is an area of memory reserved for the execution of tasks. In the simplest case, a task uses all of the partition. If the task is smaller than the partition, the unused space is unavailable to other tasks. If a task is larger than the partition, it must be written to use overlays. Overlays are sections of code that are brought into memory as needed and are written over existing code that is no longer required.

Most of the tasks that run in the RSX-20F Executive (or EXEC for short) partition handle specific system functions and devices. They are always core resident and are not swapped out. This is important because system functions and devices demand instant service and should not have to wait for code to be read in from a peripheral device. These tasks are usually called by other tasks.

A situation can arise where a task is larger than the partition it must fit in. This can be handled by the use of overlays. When a section of code is needed that is not in core, it is brought into memory and replaces (overlays) existing code.

There are two types of tasks: privileged and non-privileged. A privileged task can access its own partition, the Executive partition, and the I/O page. A non-privileged task can only access its own partition and shared regions.

When a task has been compiled, it is still not ready to be loaded and executed. It must be put through the Task Builder. A compiler produces an output file called an object module. The Task Builder accepts object modules as input, links them together, resolves references to global symbols and library files, and produces an output file called a task image. In the task image file, all relocatable expressions and external references have been converted to absolute addresses. The task image file can then be loaded into a partition and executed. The Task Builder can also produce a memory map file. A memory map describes the allocation of storage, itemizes the separate modules that comprise the task, and lists all global-symbol values.

1.4 FILES-11 SYSTEM

All RSX-based operating systems have a standard file system called FILES-11. Users who access files in an RSX-20F FILES-11 system use a syntax that is similar to TOPS-20 and TOPS-10. The file specification for FILES-11 is:

```
dev:[g,m]filename.ext;version
```

where:

dev: is the name of a physical or logical device on which the desired file is located. The device name consists of 2 ASCII characters followed by an optional one-digit unit number and a colon.

[g,m] is the group number and member number associated with the User File Directory (UFD). These numbers are octal and are in the range of 1 to 777. This section of the file specification is also referred to as the User Identification Code (UIC).

filename is the name of the file and can be from 1 to 9 alphanumeric characters.

ext is the extension of the file and can be from 1 to 3 alphanumeric characters or null.

version is the version number of the file and can range from 1 to 77777. If no version number is specified, the number defaults to the most recent version on a read operation and the next version number on a write operation.

By comparison, the TOPS-20 file specification format is:

```
dev:<directory>filename.type.gen
```

The TOPS-10 file specification format is:

```
dev:filename.ext[p,pn]
```

The quantity [g,m] is the directory number and corresponds to the directory name in TOPS-20 and the project-programmer number in TOPS-10. The User Identification Code (UIC) relates to directory numbers in the same manner that users relate to directories under TOPS-20 and project-programmer numbers under TOPS-10. Here are some examples of valid RSX-20F FILES-11 file specifications:

```
DB0[5,5]KLINIK.TSK

DX1:[5,5]MIDNIT.1
```

The word volume is used frequently when talking about FILES-11 devices. A volume is a logical file structure which includes one or more devices of the same type. A FILES-11 volume can be compared to a file structure under TOPS-10 and TOPS-20.

When FILES-11 devices are used by a task, each device is assigned a number called a Logical Unit Number (LUN). LUNs are associated with a physical device during a task's I/O operations. The Executive can also assign LUNs for its own use.

1.5 RSX-20F REQUIREMENTS

The PDP-11/40 fulfills the normal functions of a "front-end" computer. It acts as a peripheral handler and data concentrator/router in its relation with the KL10. The devices that it handles are the slower, unit record devices (TTY, CDR, and LPT). This allows the KL10 to concentrate on computing and not have to service interrupts from the slower devices.

The front end can also be used for other special functions. It can perform all the following steps necessary to get the KL10 up and running:

load the microcode

configure memory

configure cache

load a bootstrap program

It can also perform diagnostics on the KL10 when hardware problems develop.

1.6 COMPARISON OF RSX-20F AND RSX-11M

RSX-11M is geared toward quick response to real-time events and also has multiprogramming capabilities. The real-time response allows any attached devices to be serviced quickly. The multiprogramming ability allows the development and use of utility programs which can perform special tasks. These were the reasons that RSX-11M was chosen as the basis for RSX-20F.

There is only one partition in which utility programs may be run in RSX-20F. That partition is the GEN partition. Only one utility task may run at any one time in the GEN partition and that task runs until completion. Some tasks use overlays but using overlays is controlled by the task and not by the Executive.

The significant event scheme of RSX-11M was kept in order to handle changes in system states and to provide directives with information. The directives that were kept provide I/O service, task information and task control. The scheduling algorithm used to decide which task runs next is round robin within priority value.

Specific programs are brought in to do special tasks. Some of the RSX-20F utility programs are MOUNT and DISMOUNT to control access to FILES-11 devices, PIP to transfer files from one FILES-11 device to another, UFD to create User File Directories on FILES-11 devices, and PARSER to provide communication and diagnostic functions.

The biggest change had to do with programming the DTE-20 interface between the PDP-11 and the KL10. A device driver was added. Since the DTE-20 is used by several devices, a queue mechanism had to be set up to handle all the requests. As a consequence, the queued protocol task was added to handle the communication between TOPS-20 and the device drivers in the front end.

The dual-ported RP04/06 is accessible by both the KL10 and the PDP-11, independently. However, RSX-20F does not have access to the entire dual-ported disk. RSX-20F is limited to 950 pages and logical block number 400 is the home block for the FILES-11 system. TOPS-20 views the front-end file system as one big file, <ROOT-DIRECTORY>FRONT-END-FILE-SYSTEM.BIN. TOPS-10 also views the front-end file system as one big file, SYS:FEFILE.SAV.

System access to front-end files is usually done with file IDs. The front-end file system contains relatively few files and this access method can find those files quickly. The directory structure is kept for those few situations when users must interact with a FILES-11 area on floppy disk, DECtape, or dual-ported RP04/06. There exist several UFDs but only one UIC. No protection checking is enforced with the file systems.

Real PDP-11 formatted disks have 16-bit words, and disk addressing and accessing is consonant with this scheme. However, disks supported by TOPS-10 and TOPS-20 must be formatted in 18-bit words to make them compatible with the 36-bit word size expected by the KL10 processor. Therefore, the RSX-20F disk driver is a modified RSX-11M routine. Each PDP-11 word of data in the FILES-11 area is written right-justified in the 18-bit space available. The two left-hand (high-order) bits are ignored by RSX-20F's disk driver.

CHAPTER 2

RSX-20F INTERNALS

Before examining the internals of RSX-20F, let us recall the functions of an operating system. These functions are:

- To provide service to the I/O devices in the form of device drivers
- To control the scheduling of the device drivers via some monitor call and queue mechanism
- To control the scheduling of tasks
- To provide common routines that any program can use

2.1 RSX-20F EXECUTIVE

RSX-20F is not a paging or swapping system. All of the RSX-20F Executive is in memory all the time. Also, it is always in the same location in memory. Therefore, it is possible to know exactly where every module of the Executive is located. This differs from TOPS-20 and TOPS-10 where only part of the monitor is in memory at any one time. Furthermore, each time a module is brought into memory, it may occupy different areas of memory. Overlays are not handled by the Executive, but by tasks. The only tasks that use overlays are tasks such as the PARSER or KLINIT that are too large to fit into the GEN partition.

The components of the RSX-20F Executive are shown in Figure 2-1.

```
+-------------------------------------------------------+
!                   LC - Lower Core                     !
! Contains all vectors to handle interrupts and traps   !
!-------------------------------------------------------!
!                  SCH - The Scheduler                  !
!   Handles trap instructions and scheduling of tasks   !
!-------------------------------------------------------!
!           BOOT - The Boot Protocol Handler            !
!   Handles communications with the KL10, when RSX-20F  !
!                 is booting the KL10                   !
!-------------------------------------------------------!
!                    PF - Power Fail                    !
!      Contains code to handle power-fail conditions    !
!-------------------------------------------------------!
!             DMDTE - DTE Directive Service             !
!    Handles all directives concerned with the DTE      !
!-------------------------------------------------------!
!             DMASS - Assign LUN Directive              !
!    Assigns system Logical Unit Numbers (LUNs)         !
!                      to devices                       !
!-------------------------------------------------------!
!           DMGLI - LUN Information Directive           !
!              Gives information about the              !
!     Logical Unit Numbers that have been assigned      !
!-------------------------------------------------------!
!              DMGTP - Get Time Parameters              !
!              Gets information about time              !
!-------------------------------------------------------!
!          DMSED - Significant Event Directive          !
!          Handles the setting and clearing of          !
!               significant event flags                 !
+-------------------------------------------------------+
!              DMMKT - Mark Time Directive              !
!   Contains code to mark time, or keep a program in    !
!    a wait condition until a significant event occurs  !
!-------------------------------------------------------!
!          DMCMT - Cancel Mark Time Directive           !
!    Contains code to cancel a mark time condition.     !
!-------------------------------------------------------!
!         DMSUS - Suspend and Resume Directives         !
!    Suspends or resumes execution of issuing task      !
!-------------------------------------------------------!
!                DMEXT - Exit directive                 !
!          Terminates execution of issuing task         !
!-------------------------------------------------------!
!                 DMQIO - QIO directive                 !
!        Places an I/O request for a device into        !
!               the queue for that device               !
!-------------------------------------------------------!
!          DMSAR - Send and Receive Directives          !
!       Sends data to and receive data from a task      !
!-------------------------------------------------------!
!          DMSDV - Specify SST Table Directive          !
!     Records synchronous system trap entry points.     !
!             (For debugging purposes only)             !
!-------------------------------------------------------!
```

Figure 2-1 RSX-20F Executive

```
!-------------------------------------------------------!
!         DMAST - Specify AST Service Directive         !
!    Records the service routine to be executed on a    !
!                power fail for a device                !
!-------------------------------------------------------!
!             DMREQ - Run a Task Directive              !
!            Makes a task active and runnable           !
+-------------------------------------------------------+
!         DMGPP - Get Task Parameters Directive         !
!        Gets information about a task and puts it      !
!        into a 16-word block for a task to read        !
!-------------------------------------------------------!
!      DMGMP - Get Partition Parameters Directive       !
!     Gets information about a partition and puts it    !
!        into a 16-word block for a task to read        !
!-------------------------------------------------------!
!         RUN - Clock Tick Recognition Service          !
!         Checks time dependent flags at each           !
!                       clock interrupt                 !
!-------------------------------------------------------!
!                        QPRDTE                         !
!       DTE-20 device driver and queued protocol        !
!-------------------------------------------------------!
!                        TTYDRR                         !
!                Terminal device driver                 !
!-------------------------------------------------------!
!                         SCOMM                         !
!              RSX-20F Executive Data Base              !
!-------------------------------------------------------!
!                         ARITH                         !
!          Miscellaneous arithmetic functions           !
!                (Multiply, divide, etc)                !
!-------------------------------------------------------!
!                         DBDRV                         !
!            Dual-ported disk device driver             !
!-------------------------------------------------------!
!           DTDRV                       DXDRV           !
! DECtape device driver (or) Floppy disk device driver  !
!         (TOPS-10)                   (TOPS-20)         !
+-------------------------------------------------------+
!                         FEDRV                         !
!               Pseudo FE: device driver                !
!-------------------------------------------------------!
!                         LPDRV                         !
!              Line-printer device driver               !
!-------------------------------------------------------!
!                         CRDRV                         !
!               Card-reader device driver               !
!-------------------------------------------------------!
!                        INSTAL                         !
!          Task that installs a task into the           !
!                     GEN partition                     !
+-------------------------------------------------------+
```

Figure 2-1 RSX-20F Executive (Cont.)

The bulk of the RSX-20F Executive is taken up by the directive service routines and the device drivers. The scheduler is small and not as involved as the scheduler in TOPS-20 or TOPS-10 because there are fewer tasks to schedule and they run quickly. Scheduling is discussed in Section 2.2. A representation of the entire memory is shown in Figure 2-2. The RSX-20F Executive takes up memory locations 000000 to 070000. The area labeled .FREPL is a free pool of space for general use by the Executive. The TTY thread lists, task information, and LPT thread lists are stored in the free pool. The other area, .BGBUF, is a big buffer used to store LPT RAM data or task information when a task is being installed. The GEN partition is where the RSX-20F utility programs are executed. It is sometimes referred to as the "user" area. The F11TPD partition is a system partition and usually hosts the FILES-11 Ancillary Control Processor (F11ACP). Other tasks that also use this partition are SETSPD, KLRING, KLDISC, and MIDNIT. The I/O page (also referred to as the external page) resides in upper memory and contains the input and output device registers.

With the aid of the Task Builder map for RSX-20F and the PDP-11 Peripherals Handbook it is possible to determine what is in any of the locations in memory and what the contents represent. This can be useful when using the switch registers to look at a crashed system. Not only are the locations of the hardware registers known but also many key software locations can be examined. In addition, with the aid of a map, data in the task that is in the GEN partition can be examined if necessary.

```
+------------------------------+ 177776
!                              !
!                              !
!                              !
!                              !
!                              !
!          I/O PAGE            !
!                              !
!                              !
!                              !
!                              !
!                              !
!------------------------------! 160000
!                              !
!           F11TPD             !
!                              !
!------------------------------! 145400
!                              !
!                              !
!                              !
!                              !
!                              !
!        GEN PARTITION         !
!                              !
!                              !
!                              !
!                              !
!                              !
!------------------------------! 100000
!           .BGBUF             !
!------------------------------! 076000
!           .FREPL             !
!------------------------------! 070000
!                              !
!                              !
!                              !
!                              !
!                              !
!         RSX-20F EXEC         !
!                              !
!                              !
!                              !
!                              !
!                              !
+------------------------------+ 000000
```

Figure 2-2 RSX-20F Memory Layout

2.2 TASKS AND SCHEDULING

The tasks that run in the front end are either part of the RSX-20F Executive or are utility programs. Executive tasks are resident in memory while the utilities are brought in from auxiliaries storage as needed. The following parts of the Executive are considered tasks and must be scheduled:

DTEDRV	DTE device driver
FEDRV	FE device driver
DBDRV	RP04 device driver
DXDRV	Floppy disk device driver
DTDRV	DECtape driver
TTYDRV	Terminal device driver
LPTDRV	LPT device driver
CDRDRV	Card-reader device driver
F11ACP	FILES-11 Control processor
QPRDTE	Queued Protocol
INSTAL	Installs task into GEN partition
NULL	Null task

Notice that it is a task (INSTAL) that installs the task in the GEN partition. The Executive has a system partition for its own use. The F11ACP task is usually kept here. F11ACP stands for FILES-11 Ancillary Control Processor. An Ancillary Control Processor (ACP) is a privileged task that extends the function of the Executive. F11ACP receives and processes file-related I/O requests on behalf of the Executive.

RSX-20F keeps several lists about tasks so that it knows what the tasks are doing. The System Task Directory (STD) is a list of all tasks installed into the system. Each task on the list has a 15-word block (referred to as an STD node) that contains the information shown in Figure 2-3.

```
+--------------------------------------------------------+
!                      Task Name                         !  0 S.TN
!                (6 char in RADIX-50)                    !
!--------------------------------------------------------!
!           Default Task Partition Address               !  4 S.TD
!--------------------------------------------------------!
!                     Flags Word                         !  6 S.FW
!--------------------------------------------------------!
!  System Disk Indicator   !     Default Priority        ! 10 S.DI/S.DP
!--------------------------------------------------------!
!       1/64th of Base Address of Load Image             ! 12 S.BA
!--------------------------------------------------------!
!                 Size of Load Image                     ! 14 S.LZ
!--------------------------------------------------------!
!                  Maximum Task Size                     ! 16 S.TZ
!--------------------------------------------------------!
!                  Initial PC of Task                    ! 20 S.PC
!--------------------------------------------------------!
!             Initial Stack Pointer of Task              ! 22 S.SP
!--------------------------------------------------------!
!        Send and Request Queue Forward Pointer          ! 24 S.RF
!--------------------------------------------------------!
!        Send and Request Queue Backward Pointer         ! 26 S.RB
!--------------------------------------------------------!
!               SST Vector Table Address                 ! 30 S.SS
!--------------------------------------------------------!
!            Load Image First Block Number               ! 32 S.DL
!                      (32 bits)                         !
+--------------------------------------------------------+
```

```
Task Flags (Bytes 6 and 7)

----------

SF.TA==000001   Bit  0 - Set when task is active
SF.FX==000002   Bit  1 - Set when task is fixed in memory
SF.EX==000004   Bit  2 - Set when task to be removed on exit
SF.IR==040000   Bit 14 - Set when install is requested
SF.ST==100000   Bit 15 - Set when task is system task
```

Figure 2-3 System Task Directory (STD) Node

The 15-word blocks in the STD are pointed to by entries in the table .STDTB. This table has an entry for every installed task in the system.

RSX-20F keeps another list of those tasks wanting to run. This list is called the Active Task List (ATL). The ATL is a doubly linked list of nodes (entries) for active tasks that have memory allocated for their execution. The list is in priority order. Tasks with an entry in the ATL are either in memory or are being loaded into memory. A node, in RSX-20F, is a block of data, in this case data about a task. "Doubly linked" means that each node is linked to both the previous node and the following node. The ATL nodes have the format shown in Figure 2-4.

```
+--------------------------------------------------------+
!                    Forward Pointer                     !  0
!--------------------------------------------------------!
!                   Backward Pointer                     !  2
!--------------------------------------------------------!
! SP of running task when this task is not current task !  4 A.SP
!--------------------------------------------------------!
!         The task's run partition (TPD address)         !  6 A.PD
!--------------------------------------------------------!
!          (null)           !   Task's Run Priority      ! 10     /A.RP
!--------------------------------------------------------!
!          1/64th of real address of load image          ! 12 A.HA
!--------------------------------------------------------!
!       Task Flags byte     !         Task Status        ! 14 A.FB/A.TS
!--------------------------------------------------------!
!        System Task Directory (STD) entry address       ! 16 A.TD
!--------------------------------------------------------!
!              Task's Significant Event Flags            ! 20 A.EF
!                        (32 bits)                       !
!--------------------------------------------------------!
!                 Task's Event Flags Masks               ! 24 A.FM
!                        (64 bits)                       !
!                                                        !
!--------------------------------------------------------!
!                Power Fail AST Trap Address             ! 34 A.PF
+--------------------------------------------------------+
```

Status Bits (Byte 14)

TS.LRQ==02	Bit 1 - Task load request queued
TS.TKN==04	Bit 2 - Task waiting for termination notice
TS.LRF==06	Task load request failed
TS.RUN==10	Task is running
TS.SUS==12	Task is suspended
TS.WF0==14	Task is waiting for an event 1-14
TS.WF1==16	Task is waiting for an event 17-32
TS.WF2==20	Task is waiting for an event 33-48
TS.WF3==22	Task is waiting for an event 49-64
TS.WF4==24	Task is waiting for an event flag 1-64
TS.EXT==26	Task exited

Flag Bits (Byte 15)

AF.PP==200	Bit 7 set when task is primary protocol task

Figure 2-4 Active Task List (ATL) Node

When you are looking at dumps of RSX-20F, the location .CRTSK points to the ATL node of the current task.

Installing a task into the GEN partition consists of reading it into memory from the system file area, putting the task into the STD and ATL and setting the appropriate flags. The STD and ATL entries are located in the Executive Data Base.

One of the tasks in the Active Task List is the Null Task. The Null Task is the task that runs when no other task wants to run (a very quiet system) or no other task can run (tasks are blocked waiting for pending I/O).

Scheduling for all tasks is by a priority system. When a task is installed it gives a priority which is reflected in its position in the ATL. The task with the highest priority goes first in the list, the next highest goes second, and so on. Scheduling occurs in two cases. One is when a task has declared itself waiting for some significant event to occur. The other is when a directive service routine exits. There are two entry points to the ATL scan routine. Control is passed to ASXE1 when a significant event is declared. The ATL is scanned from the beginning to the end to find the first runnable job. Control is passed to ASXE2 when a directive service routine exits. In this case, one of three things can happen:

1. Control can be returned to the task that issued the directive,

2. The ATL can be scanned for the next runnable task beginning with the task that issued the directive down the ATL through the lower priority tasks, or

3. The ATL can be scanned from the beginning.

2.3 SYSTEM TRAPS

A trap is a CPU-initiated interrupt which is automatically generated when a predetermined condition is detected. Two vector locations in low memory are dedicated for each trap type. The vector locations contain the PC and PS for the trap service routine. When the trap occurs, the current PC and PS are put on the stack and the contents of two vector locations are loaded as the new PC and PS. Traps can occur as the result of the following conditions:

Location	Trap
004	CPU errors
010	Illegal and reserved instructions
014	BPT
020	IOT
024	Power Fail
030	EMT
034	TRAP

Traps can be one of two types; synchronous or asynchronous. An asynchronous trap occurs as the result of an external event such as the completion of an I/O request. In this case, the task will be doing other work or waiting for the I/O to be done. A synchronous trap occurs immediately upon the issue of the instruction that causes the trap.

The PDP-11 instruction set contains several instructions that cause traps. The EMT instruction, generally reserved for system software, causes a trap to an emulator routine. This is used by RSX-20F to perform directives. Whenever a directive must be performed, the necessary information is loaded into the registers and an EMT is issued. The EMT instruction traps to a routine that will decide which directive is to be performed. A TRAP instruction is like the EMT instruction except that it is used by user tasks. The only difference between TRAP and EMT is a different vector location. IOT is used by RSX-20F for error reporting. When RSX-20F detects an error that is considered serious enough to crash the system, an IOT instruction is issued.

Power fail conditions cause an automatic trap independent of the software operations mentioned above.

There are two places in RSX-20F where traps are handled. The following events cause a trap to location COMTRP:

- IOT instruction
- TRAP instruction
- BPT (break point trap)
- Trap to 10 (illegal instruction)
- Trap to 4 (device or memory timeout)
- Illegal Interrupts
- Parity Error

COMTRP has to sort out the type of error it gets. If possible, only the offending task will be terminated. If a conclusion is reached that this is an error that is serious enough to crash the system, the COMTRP routine will issue a .CRASH macro itself. This causes control to come right back to COMTRP. COMTRP sees that it was an IOT instruction that has occurred and dispatches to IOTTRP. The COMTRP routine performs the following functions:

1. Tries to restore the user task that had the problem,
2. Issues an IOT error instruction, or
3. Dispatches to the IOT handling routine.

The IOTTRP routine performs the following functions:

1. Sets up the emergency stack pointer
2. Sets up the emergency message pointer
3. Saves the registers
4. Saves the crash code
5. Saves the parity-error data
6. Prints the 11-Halt message on the CTY (and KLINIK)
7. Requests the KL10 to reload the PDP-11 via the DTE20
8. Rings the KL10 Doorbell
9. Loops through the previous step until the PDP-11 is reloaded

The routine to handle EMT instructions is comparable in a way to JSYSs under TOPS-20. Since the PDP-11 is a smaller system, it cannot have one instruction for every directive it wants to run. Therefore, what is hardware on the KL10 is handled by a combination of hardware and software on the PDP-11. The KL10 handles the instruction by dispatching to the right routine. The PDP-11 issues the trap and then the software checks the stored argument to decide which routine is to be dispatched.

2.4 RSX-20F CRASH CODES

Upon encountering a serious error, the front end will stop and wait for the KL10 to reload it. Before it dies, however, it prints the following message on the CTY:

```
11-HALT
<code>
```

where <code> is a 3-character error code indicating why it crashed. Refer to Appendix C for a list of RSX-20F stop codes.

Whenever RSX-20F discovers a condition that it considers serious enough to crash, it executes an IOT instruction. The 3-character crash code following the IOT is picked up by the crash routine.

```
IOT
ASCIZ /3-character crash code/
```

For a disk-based PDP-11 operating system, the IOT instruction is used for error reporting. It first executes a trap vector at location 20. From there it dispatches to CDMTRP, then to IOTTRP which will halt the PDP-11 trying to save what information in the registers that it can.

The IOT routine stores the crash code and parity-error registers in locations 0 to 3 of PDP-11 memory. This information is readily available in a dump listing. For example, an IOT instruction followed by ASCIZ /DTB/ would result in the following:

```
+----------------------+
!     T    !     D     ! 0
!----------------------!
! Par.Error!     B     ! 2
+----------------------+
```

When looking at the source listings of RSX-20F, you will notice that a macro is used instead of the IOT instruction. The macro is .CRASH and expands to the IOT plus ASCIZ crash code as stated above.

CHAPTER 3

RSX-20F UTILITIES

RSX-20F is designed to run with as little human intervention as possible. The main function of RSX-20F is data concentration and routing. However, occasions arise when some flexibility is needed such as when the front end's file system must be used or a failing KL10 must be diagnosed. A set of utility programs gives the operator, system programmer, or field service engineer the capabilities required.

Since RSX-20F was derived from RSX-11M, RSX-11M utilities were adapted to run on RSX-20F. PIP, COPY, MOUNT, and DMOUNT were brought over in this way. UFD, the utility to create User File Directories, was taken out of other programs and made into a separate entity. The PARSER was designed with RSX-20F in mind. The PARSER is the most flexible and useful of all the programs. It controls the loading into memory of all the other utilities, examines the state of the KL10, and provides the communications between operator and system.

3.1 PARSER

The PARSER runs as a task under the RSX-20F Executive. Its primary function is to receive ASCII command strings, usually from the console terminal, and perform console functions on the KL10 or PDP-11 computer. To run the PARSER, type a control backslash:

```
CTRL/\
```

The PARSER will respond with one of three prompts:

PAR> indicates that the PARSER is ready to accept commands and the KL10 is running.

PAR% indicates that the PARSER is ready to accept commands and the KL10 microcode is in the HALT loop.

PAR# indicates that the PARSER is ready to accept commands and the KL10 is not running: the KL clock is stopped.

3.1.1 Command Format

Commands to the PARSER are typed one or more to a line in response to a PAR>, PAR% or PAR# prompt. The following rules apply when typing commands:

1. You can type multiple commands on a line by separating them with a semicolon (;). For example:

   ```
   PAR>EXAMINE PC;EXAMINE 20;SHUTDOWN<CR>
   ```

2. You can continue a command on the next line by ending the first line with a dash (-). Continuation lines will prompt with another dash. For example:

   ```
   PAR>START MICRO-<CR>
   -CODE<CR>
   ```

3. You can include comments on command lines or by themselves. Comments are indicated by an exclamation mark. These may be used either for documentation or for communication between an operator on site and the remote diagnostician when the KLINIK line is in use. For example:

   ```
   PAR>!IS THE CPU STILL SMOKING?<CR>
   ```

   or:

   ```
   PAR>REPEAT 5;EXAMINE PC!SEE IF THE CPU IS IN A LOOP<CR>
   ```

4. You can suppress typeout with CONTROL-O (^O).

5. You can abbreviate keywords in commands to their shortest unique value. For example:

   ```
   PAR>H!HALT THE KL10 CPU<CR>
   ```

   or:

   ```
   PAR%RE<CR>
   PAR -- [PARSER] AMB - AMBIGUOUS KEYWORD "RE"
   ```

   In the latter example, the PARSER found two commands that started with RE: REPEAT and RESET.

6. The radix of integers defaults to octal if an address or 36-bit value is expected; otherwise, the default is decimal.

7. An underscore (_) between two numbers causes the left-hand number to be binary shifted to the left, the number of positions indicated by the right-hand number. For example:

   ```
   PAR>EXAMINE 2_3
   ```

   results in:

   ```
   20/ 000000 000033
   ```

8. A unary minus (-) preceding a number causes the negative of the number to be used. For example:

   ```
   PAR>DEPOSIT TEN 30:-1
   30/ 000000 000000
   PAR>EXAMINE TEN 30
   30/ 777777 777777
   ```

9. Numbers may be added and subtracted using the plus (+) and minus (-) arithmetic operators, respectively. For example:

```
PAR>EXAMINE 123654+33
123707/ 440010 003201
PAR>START TEN 407-6
```

10. A single quote (') adds the value of the PDP-11 relocation counter to a number and a double quote (") subtracts the value of the relocation counter. For example, assume that the relocation counter contains 101204:

```
PAR>EXAMINE ELEVEN 34'
101240\ 000000
PAR>EXAMINE ELEVEN 101240"
34\ 003004
```

3.1.2 Console Modes

The PARSER has three modes of operation. The mode can be altered by the CLEAR CONSOLE and SET CONSOLE commands. Console modes are the following:

MAINTENANCE MODE

Allows you to use all of the PARSER commands.

PROGRAMMER MODE

Allows you to use all of the PARSER commands except diagnostic functions.

OPERATOR MODE

Allows you to use only those commands that will not crash the TOPS-10 or TOPS-20 monitor.

USER MODE

Exits the PARSER and connects the CTY to the TOPS-10 or TOPS-20 operating system. (SET CONSOLE USER and QUIT are equivalent.)

When you first bring up the front-end system and enter the PARSER (CTRL/\), you will be in PROGRAMMER mode.

3.1.3 Operator Mode Commands

The following commands make up the subset of PARSER commands that are available to you when in OPERATOR console mode.

ABORT

The ABORT command stops the KL10 by trying to force it into the HALT loop. If this fails after a reasonable number of EBOX clock ticks, the command tries to START MICROCODE, which implies a master reset of the KL10 processor. This is a way to get the KL10 into a known state when a previous state left it in a hung condition.

CLEAR

The CLEAR command accepts the following arguments:

CONSOLE

Forces the PARSER into OPERATOR console mode.

INCREMENT

Sets the KL10 increment factor for examines to zero.

KLINIK

Closes the KLINIK access window and terminates the KLINIK link. Refer to the section on KLINIK in the appropriate Operator's Guide.

MEMORY

Sets the default memory for examines to be the KL10. CLEAR MEMORY is equivalent to SET MEMORY TEN.

NOT

Used with CLEAR, it is equivalent to SET.

REPEAT

Sets the repeat counter to zero. All subsequent command lines will be executed once.

DISCONNECT

Disconnects the KLINIK link by running the KLDISC task. Refer to the section on KLINIK in the appropriate Operator's Guide.

EXAMINE

The EXAMINE command accepts the following arguments:

PC

Displays the current contents of the KL10 program counter.

ELEVEN addr

Displays the current contents of the specified PDP-11 address.

KL

Performs the EXAMINE FLAGS, EXAMINE PC, EXAMINE PI and EXAMINE VMA commands. The individual commands are documented in Section 3.1.4.

TEN addr

Displays the current contents of the specified KL10 address. If no address is specified NEXT is assumed.

DECREMENT

Takes the last address referenced, decrements it by the increment factor in effect and performs the examine. For the KL10, the increment can be changed (see SET INCREMENT). The increment for the PDP-11 is fixed at 2.

INCREMENT

Takes the last address referenced, increments it by the increment factor in effect, and performs an examine. For the KL10, the increment can be changed (see SET INCREMENT). The increment for the PDP-11 is fixed at 2.

NEXT

Takes the last address referenced, adds 1 for a KL10 (2 for a PDP-11) and performs the examine.

PREVIOUS

Takes the last address referenced, subtracts 1 for a KL10 (2 for a PDP-11), and performs the examine.

THIS

Performs the examine with the last address referenced.

JUMP addr

Starts the KL10 at the specified address and exits from the PARSER. The CTY is then connected to the TOPS-10 or TOPS-20 operating system.

MCR taskname

Loads and starts the specified task file.

QUIT

Exits the PARSER. The CTY is then connected to the TOPS-10 or TOPS-20 operating system.

REPEAT n

Causes subsequent commands on the command line to be repeated n times. n must be a positive, decimal, nonzero integer.

RUN taskname

Loads and starts the specified task file. This is an alias for the MCR command.

SET

The SET command accepts the following arguments:

CONSOLE mode

The SET CONSOLE command sets the front-end console command mode. The available modes are:

USER

Exits the PARSER and connects the CTY to the TOPS-10 or TOPS-20 operating system.

OPERATOR

Allows only those commands in the OPERATOR subset.

PROGRAMMER

Allows only those commands in the PROGRAMMER subset.

MAINTENANCE

Allows all commands.

INCREMENT n

Sets the KL10 increment factor for examines to n. n must be an integer.

KLINIK

Enables the KLINIK link. This command invokes a dialogue which requests additional information. Refer to the section on KLINIK in the appropriate Operator's Guide.

MEMORY system

The SET MEMORY command accepts the following arguments:

ELEVEN

Sets the PDP-11 as the default memory for examines.

TEN

Sets the KL10 as the default memory for examines.

NOT

Used with SET, it is equivalent to CLEAR.

REPEAT n

Sets the repeat counter to the value n. n is a decimal, positive, nonzero number and must be specified. All subsequent command lines will be repeated that number of times.

SHUTDOWN

This command DEPOSITs a minus one into the KL10 EXEC virtual location 30 (octal). It is used to gracefully bring down the KL10 timesharing system. It will cause the PARSER to EXIT if the DEPOSIT was successful.

WHAT

The WHAT command accepts the following arguments:

CONSOLE

Displays the current console mode: OPERATOR, PROGRAMMER or MAINTENANCE.

DATE

Displays the day, date, and time currently stored in the front-end Executive.

INCREMENT

Displays the KL10 increment for examines.

KLINIK

Displays the current status of the KLINIK link. Refer to the section on KLINIK in the appropriate Operator's Guide.

MEMORY

Displays the default memory for deposits and examines (TEN or ELEVEN).

REPEAT

Displays the current value of the REPEAT counter set by the last SET REPEAT command.

VERSION

Displays the versions of the current front-end Executive and PARSER.

3.1.4 Programmer Mode Commands

The following commands make up the subset of PARSER commands that are available to you when in PROGRAMMER console mode.

ABORT

The ABORT command stops the KL10 by trying to force it into the HALT loop. If this fails after a reasonable number of EBOX clock ticks, the command tries to START MICROCODE, which implies a master reset of the KL10 processor. This is a way to get the KL10 into a known state when a previous state left it in a hung condition.

CLEAR

The CLEAR command accepts the following arguments:

CONSOLE

Forces the PARSER into OPERATOR console mode.

DATE

This is an invalid command.

INCREMENT

Sets the KL10 increment factor for examines to zero.

KLINIK

Closes the KLINIK access window and terminates the KLINIK link. Refer to the section on KLINIK in the appropriate Operator's Guide.

MEMORY

Sets the default memory for examines to be the KL10. CLEAR MEMORY is equivalent to SET MEMORY TEN.

NOT

Used with CLEAR, it is equivalent to SET.

OFFSET

Sets the value of the PDP-11 relocation counter to zero. (See Section 3.1.1, Command Format.)

RELOAD

Disables the automatic reloading of the KL10 following an error condition.

REPEAT

Sets the repeat counter to zero. All subsequent command lines will be executed once.

TRACKS

Stops displaying all KL10 operations on the CTY.

CONTINUE

Takes the KL10 out of the HALT loop, causing it to execute the instruction pointed to by the PC.

DEPOSIT

The DEPOSIT command accepts the following arguments:

AR:newdata

Sets the contents of the arithmetic register to newdata.

ELEVEN addr:newdata

Displays the contents of the PDP-11 address, addr; then replaces the contents with newdata.

TEN addr:newdata

Displays the contents of the KL10 address, addr; then replaces the contents with newdata.

DECREMENT:newdata

Takes the last address referenced, decrements it by the increment factor in effect, displays the contents of that location, and then replaces those contents with newdata.

INCREMENT:newdata

Takes the last address referenced, increments it by the increment factor in effect, displays the contents of that location, and then replaces those contents with newdata.

NEXT:newdata

Takes the last address referenced, adds 1 for a KL10 (2 for a PDP-11), displays the contents of that location, and then replaces those contents with newdata.

PREVIOUS:newdata

Takes the last address referenced, subtracts 1 for a KL10 (2 for a PDP-11), displays the contents of that location, and then replaces those contents with newdata.

THIS:newdata

Displays the contents of the last address referenced and then replaces those contents with newdata.

DISCONNECT

Disconnects the KLINIK link by running the KLDISC task. Refer to the section on KLINIK in the appropriate Operator's Guide.

EXAMINE

The EXAMINE command accepts the following arguments:

AB

Displays the contents of the address break register.

AD

Displays the contents of the arithmetic adder.

ADX

Displays the contents of the arithmetic adder extension.

AR

Displays the current contents of the arithmetic register.

ARX

Displays the contents of the arithmetic register extension.

BR

Displays the contents of the buffer register.

BRX

Displays the contents of the buffer register extension.

CRADDR

Displays the contents of the CRAM address register.

CRLOC

Displays the contents of the CRAM location register.

DRADDR

Displays the contents of the DRAM address register.

DTE-20

Displays the contents of the three diagnostic registers for the console DTE and its status register.

EBUS

Displays the contents of the EBUS.

ELEVEN addr

Displays the contents of the specified PDP-11 address.

FE

Displays the contents of the floating exponent register.

FLAGS

Displays the state of the flag bits (0-12) in the left half of the PC: OVF, CY0, CY1, FOV, BIS, USR, UIO, LIP, AFI, AT1, AT0, FUF, and NDV.

FM

Displays the contents of the fast memory register.

MQ

Displays the contents of the multiplier quotient register.

PI

Displays the current state of the priority interrupt system.

REGISTERS

Displays the contents of the following registers: AD, ADX, AR, ARX, BR, BRX, EBUS, FM, MQ and PC.

SBR

Displays the contents of the subroutine return register.

SC

Displays the contents of the shift count register.

TEN addr

Displays the contents of the specified KL10 address.

VMA

Displays the contents of the virtual memory address register.

VMAH

Displays the contents of the virtual memory address held register.

HALT

The HALT command takes no arguments. It tries to put the KL10 into the HALT loop by clearing RUN and waiting. If the KL10 refuses to go into the HALT loop, the front end tries to force it in by using BURST mode. If this does not work, an error message is typed.

INITIALIZE

Reinitializes the PARSER.

JUMP addr

Starts the KL10 at the specified address and exits from the PARSER. The CTY is then connected to the TOPS-10 or TOPS-20 operating system.

MCR taskname

Loads and starts the specified task file.

QUIT

Exits the PARSER. The CTY is then connected to the TOPS-10 or TOPS-20 operating system.

REPEAT n

Causes subsequent commands on the command line to be repeated n times. n must be a positive, decimal, nonzero integer.

RESET

The RESET command accepts the following arguments:

<CR>

Causes a master reset of the KL10 keeping the clock enables and parity stops set before the reset.

All

Executes a RESET DTE, RESET APR, RESET PAG, and RESET PI. The PARSER does not allow this command to be performed while the KL10 is running.

APR

Executes a CONO APR, 267760 to clear the APR. It cannot be done while the KL10 is running.

DTE-20

Resets the DTE-20 by depositing a 1 in bit 6 of diagnostic word 2 in the DTE. Then bit 0 in diagnostic word is set to 1 indicating word-mode transfers.

ERROR

Resets the error flags by executing a CONO APR,27760.

INITIALIZE

Does a master reset of the KL10; sets up the normal clock enables and parity stops.

IO

Performs an I/O reset of the KL10 processor by executing a CONO APR,200000.

PAG

Resets the KL10 PAGing box by executing a CONO PAG,0 and DATAO PAG,100. RESET PAG can be executed while the KL10 is running.

PI

Resets the KL10 Programmable Interrupt System with a CONO PI,10000. RESET PI can be executed while the KL10 is running.

RUN taskname

Loads and starts the specified task file. This command is an alias for the MCR command.

SET

The SET command accepts the following arguments:

CONSOLE mode

The SET CONSOLE command sets the front-end console command mode. The available modes are:

USER

Exits the PARSER and connects the CTY to the TOPS-10 or TOPS-20 operating system.

OPERATOR

Allows only those commands in the OPERATOR subset.

PROGRAMMER

Allows only those commands in the PROGRAMMER subset.

MAINTENANCE

Allows all commands.

DATE

Sets the date and time to be used by the RSX-20F Executive. SET DATE is invalid if RSX-20F thinks that it already has a valid date (Validity flag is on).

INCREMENT n

Sets the KL10 increment factor for examines to n. n must be an integer.

KLINIK

Enables the KLINIK link. This command invokes a dialogue which requests additional information. Refer to the section on KLINIK in the appropriate Operator's Guide.

MEMORY system

The SET MEMORY command accepts the following arguments:

ELEVEN

Sets the PDP-11 as the default memory for examines.

TEN

Sets the KL10 as the default memory for examines.

NOT

Used with SET, it is equivalent to CLEAR.

OFFSET nnn

Sets the PDP-11 relocation counter to the value nnn.

RELOAD

Enables automatic reload of the KL10 by the PDP-11 front end.

REPEAT n

Sets the repeat counter to the value n. n is a decimal, positive, nonzero number and must be specified. All subsequent command lines will be repeated that number of times.

TRACKS

Causes RSX-20F to type out, on the console terminal, all operations performed on the KL10 and their results.

SHUTDOWN

This command DEPOSITs a minus one into the KL10 EXEC, virtual location 30 (octal). It is used to gracefully bring down the KL10 timesharing system. It causes the PARSER to EXIT if the DEPOSIT is successful.

START

The START command accepts two arguments:

MICROCODE addr

Performs a MASTER RESET on the KL10 and starts the MICROCODE at the specified address. If addr is omitted, the default address is zero. Starting the MICROCODE at an address other than zero is not recommended. This command is the best way to get the KL10 to clear itself, and leave itself in a useful state.

TEN addr

Starts the KL10 at the specified address. Control then returns to the PARSER. addr is required and cannot be zero.

WHAT

The WHAT command accepts the following arguments:

CLOCK

Displays the current state of the KL10's internal clocks.

CONSOLE

Displays the current console mode: OPERATOR, PROGRAMMER or MAINTENANCE.

DATE

Displays the day, date, and time currently stored in the front-end Executive.

INCREMENT

Displays the KL10 increment for examines.

KLINIK

Displays the current status of the KLINIK link. Refer to the section on KLINIK in the appropriate Operator's Guide.

MEMORY

Displays the default memory for deposits and examines (TEN or ELEVEN).

OFFSET

Displays the value in the PDP-11 relocation counter.

PARITY-STOP

Displays the status of the parity stop settings.

RELOAD

Displays the automatic reload status as set by the last SET or CLEAR RELOAD command.

REPEAT

Displays the current value of the REPEAT counter set by the last SET REPEAT command.

TRACKS

Displays the KL10 tracking status as set by the last SET or CLEAR TRACKS command.

VERSION

Displays the versions of the current front-end Executive and PARSER.

XCT

This command takes a 36-bit numerical expression as an argument and executes it as a KL10 instruction. Note that executing an instruction with an opcode (bits 0 through 8) of zero is not allowed. If attempted, you will receive an ILLEGAL KL OPCODE error message.

ZERO loaddr>hiaddr

Zeros the specified area of KL10 memory. ZERO accepts as an argument the boundary addresses of the area to be zeroed: loaddr and hiaddr.

3.2 COPY

The COPY program copies the contents of one floppy disk to another floppy disk or one DECtape to another DECtape. This procedure can be used to make backup copies of the installation floppies or DECtapes.

To copy a floppy disk, type the following:

```
CTRL/\ (control backslash)
PAR>MCR COP
COP>DX0:=DX1:
```

This will copy the floppy in DX1: onto the floppy in DX0:, and then verify the operation.

To copy a DECtape, type the following:

```
CTRL/\ (control backslash)
PAR>MCR COP
COP>DT0:=DT1:
```

This will copy the DECtape on drive 1 onto the DECtape on drive 0 and then verify the operation.

The following switches are available:

/BL:n,m

Copy, starting at (extended) block n,m, until the last block on the device. If COPY is interrupted by a CTRL/C, it will type out the last block copied.

/CP Copy (default action).

/HE Help, types a list of switches.

/RD Read Device, check for errors.

/VF Verify the Copy operation. (default, when a copy is in effect)

NOTE

> To skip the verification procedure after a Copy operation, include this switch with a negation factor:/-VF or /NOVF.

/ZE Write zeros onto a device (deleting all files).

Examples:

To zero a floppy, deleting all files:

```
COP>DX0:/ZE
```

To check for errors on a device:

```
COP>DX0:/RD
```

3.3 PIP

The Peripheral Interchange Program (PIP) is an RSX-20F file utility program that transfers data files from one standard FILES-11 device to another. Depending on how you enter a command line, PIP either copies FILES-11 files or performs control functions. To access PIP, enter:

```
PAR>MCR PIP
```

PIP will prompt with:

```
PIP>
```

The general file specification format is:

```
dev:[ufd]filename.filetype;version
```

The format of the PIP command line is:

```
PIP>outfile=infile[,infile2,...][/switch[/subswitch,...]]
```

where:

outfile

specifies the output file. If the output filename, filetype and version are either null or *.*;*, the input filename, filetype and version are preserved (see /NV and /SU subswitches). If either filename, filetype, or version is entered, wildcards may not be used in the file specification.

infile [,infile2...]

specifies the input file or files. If the filename filetype, and version fields are all null, *.*;* is the default.

/switch

specifies one of the switches that can be entered.

/subswitch

specifies one or more of the subswitches that can be entered with its master switch entered just before it.

PIP provides several switches and subswitches for file control processing. These switches and subswitches perform such diverse tasks as deleting files, displaying the contents of the directory, and specifying file protection values. PIP switches may be specified on a command line with no file specifiers (that is, they may be entered by themselves). Only one switch at a time may be specified on the command line; however, more than one subswitch may be specified.

Table 3-1
PIP Switches and Subswitches

Switch	Subswitch	Function
/AP		Append files to the end of an existing file.
/BR		List a directory file in brief format (an alternate mode for the /LI switch).
/DE		Delete one or more files.
/DF		Change PIP's default device or UFD.
/DI		Same as /LI.
/EN		Enter a synonym for a UIC in a directory file.
	/NV	Force the version number of a file to one greater than the latest version.
/FI		Access a file by its file identification number.
/FR		Display the amount of free space on the specified volume.
/FU		List a directory file in full format (an alternative for the /LI switch).
/ID		Identify the version of PIP being used.
/LI		List a directory file.

Table 3-1 (Cont.)
PIP Switches and Subswitches

Switch	Subswitch	Function
/ME		Concatenate two or more files into one file.
	/BL:n[.]	Allocate contiguous blocks for the output file. If the dot is used, the number is considered decimal.
	/CO	Specify that the output be contiguous.
	/NV	Force the version number of a file to the latest version plus one.
	/SU	Supersede the filename, filetype, and version of a file.
/PR		Change the protection status.
	/GR[:RWED]	Set read/write protection at the group level.
	/OW[:RWED]	Set read/write protection at the owner level.
	/SY[:RWED]	Set read/write protection at the system level.
	/WO[:RWED]	Set read/write protection at the world level.
/PU[:n]		Delete obsolete version(s) of a file. If the current version of a file is m, PIP will delete all versions equal to or less than the value m-n. If :n is omitted, all but the current version are deleted.
/RE		Change the name of a file.
/TB		List the totals line for a directory (an alternative for the /LI switch).
/UN		Unlock a file. A file is locked when a task has the file open and then does not properly close it. Usually this occurs when the task or the system crashes. Files cannot be used until unlocked.
/UP		Overwrite an existing file.

Examples:

To copy a file from one place to another:

```
PIP>DB2:A.B[34,45]=DB2:FILE.EXT[100,1]
```

To get a directory of files in a UFD:

```
PIP>/LI
```

To merge two files into one file:

```
PIP>DB2:A.B=DB2:A.C,DB2:A.D/ME
```

To protect a file:

```
PIP>DB2:FILE.EXT/PR/GR:W
```

To type a file at a terminal:

```
PIP>TT:=HELP.TXT
```

To append files to an existing file:

```
PIP>JOHN.DAT/AP=GAIL.DAT,JIM.DAT
```

To delete version 4 of a file:

```
PIP>WITHROW.LOG;4/DE
```

To change the default device or UFD:

```
PIP>DB2:[123,456]/DF
```

Purge all old versions of a file:

```
PIP>MULTI.PLE/PU
```

3.4 MOUNT AND DMOUNT

The MOUNT and DMOUNT utilities are versions of the same programs used in RSX-11M. Their function is to give information to RSX-20F which will either make available a device to another user utility or take away a device. Most devices that PIP uses must be MOUNTed before using them. RSX-20F will then allocate or deallocate buffer space and LUN information as required.

There are no switches available to either of the MOUNT or DMOUNT utilities. The only devices available to be mounted are:

```
DB0: disk drive 0 (RP04/06)
.
.
DBn: disk drive n (RP04/06)

DX0: floppy drive  0
DX1: floppy drive  1
DT0: DECtape drive 0
DT1: DECtape drive 1

FE: pseudo device FE: to talk to KL10
```

The only devices that can be MOUNTed or DMOUNTed are those that have an Ancilliary Control Processor. The only devices that have an ACP are the FILES-11 devices: rigid disk, floppy disk, DECtape, and the FE: device.

Example:

```
PAR>MCR MOU
MOU>FE:
Mount completed
MOU>DB2:
Mount completed
MOU>^Z (to exit)

*** Perform desired function here ***

PAR>MCR DMO
DMO> FE:
Dismount Complete
DMO> DB2:
Dismount Complete
DMO>^Z
```

3.5 UFD

The utilities that RSX-20F uses are not able to create User File Directories on FILES-11 devices. That function has been left to another utility called UFD. UFD takes a command line and checks to see that the device on which the UFD is to be created exists. If so, the UFD is created with the specified attributes. Actual I/O is then done using directives.

Input to UFD consists of devices and user file directories followed by appropriate switches. Several UFD specifications can be entered on one line. For example:

```
PAR>MCR UFD
UFD>DB2:[10,10],DX0:[100,123]
```

There are two possible switches that can follow a directory, /ALL or /PRO. Both take arguments. The /ALL switch allocates space for the UFD and takes as an argument the number of blocks to allocate. For example:

```
UFD>DB1:[10,11]/ALL:4
```

The /PRO switch defines the protection for the UFD being created. The protection is specified by letters that denote the allowable means of access.

```
R - Read Access
W - Write Access
E - Extend Access
D - Delete Access
```

Several letters can be given at one time when defining access protection. For example:

```
UFD>DX1:[100,100]/PRO:RE
```

allows read and extend access.

CHAPTER 4

ERROR HANDLING

4.1 THE KEEP ALIVE COUNT

The KL10 and the PDP-11 watch each other to make sure that the other does not crash. This is done by means of the "Keep Alive Count." The Keep Alive Count for each processor is a word in the Communications Region of KL10 memory. Both processors have clock interrupts regularly. During the servicing of those clock interrupts, the processors increment their own Keep Alive Counts and check the other processor's Keep Alive Count. If the count has not changed after a certain number of interrupts, then that processor is assumed to be hung or crashed and must be reloaded. If the KL10 has gone down, RSX-20F will request the TKTN task to run. TKTN shuts down the protocol and schedules the KLERR task.

The console front end must go through the DTE-20 in order to update and examine the Keep Alive Counts because they are both located in the Communications Region of KL10 memory. The KL10's Keep Alive Count is kept in CMKAC, the sixth word in the KL10's area of the communications region. The PDP-11's Keep Alive Count is stored in the sixth word of the PDP-11's area of the communications region.

4.2 KLERR TASK

When the PDP-11 notices that the KL10 has not responded via the Keep Alive Count mechanism, a flag is set in RSX-20F's Exec. The TKTN task notices this flag and schedules the KLERR task to run. The decision to run KLERR is usually automatic and not controlled by the operator. However, the operator can force KLERR to run by using the PARSER command:

```
     PAR>MCR KLE
```

KLERR has several important functions. First it forces the KL10 into a known state by putting the microcode into the halt loop. KLERR then tries to read as much information as it can from registers in the KL10. It does this by performing function reads over the diagnostic portion of the EBUS. The information that is retrieved is stored in the KLERRO.SNP file in the front end's file system. When KLERR has completed its function, TKTN schedules the KLINIT task to restart the KL10.

4.3 KLERRO.SNP FILE

KLERRO.SNP is the output file from the KLERR task. It is written in the front end's file area on the dual-ported RP04/06. Note that when KLERRO.SNP is in this area it is in FILES-11 format and TOPS-20 can not access the data. The KLXFER task is required to transfer the data into the TOPS-20 file area.

The bulk of the information in KLERRO.SNP comes from storing the contents of the diagnostic DTE-20 registers after execution of function reads. Every possible function read is performed (100 to 177) octal). In addition, version number, checksum, and the time of day are stored. Refer to Figure 4-1 for the format of the KLERRO.SNP file.

```
+--------------------------------------------------------+
!              KLERR file type (version #)                !     0
!--------------------------------------------------------!
!                Record length in words                   !     2
!--------------------------------------------------------!
!                      Time of day                        !     4
!--------------------------------------------------------!
!                   KL10 Serial Number                    !    24
!--------------------------------------------------------!
!                 Front End Serial Number                 !    26
!--------------------------------------------------------!
!                Microcode Version Number                 !    30
!--------------------------------------------------------!
!              KL10 Monitor Version Number                !    32
!--------------------------------------------------------!
!                  RSX20F Version Number                  !    42
!--------------------------------------------------------!
!                       Error Code                        !    46
!--------------------------------------------------------!
!                  DTE Diagnostic Word 1                  !    50
!--------------------------------------------------------!
!                  DTE Diagnostic Word 2                  !    52
!--------------------------------------------------------!
!                     DTE Status Word                     !    54
!--------------------------------------------------------!
!                  DTE Diagnostic Word 3                  !    56
!--------------------------------------------------------!
!                                                         !    60
!     This block contains the information obtained        !
! by executing every possible diagnostic function read.   !
!    The function reads vary from 100 to 177 (octal)      !
!    After each function read, the three diagnostic       !
!               registers are stored here.                !
!--------------------------------------------------------!
!               KLERR RAD50 Error (if any)                !   660
!--------------------------------------------------------!
!                                                         !   662
!              Reserved for future expansion              !
!--------------------------------------------------------!
!                 Checksum for above words                !   676
+--------------------------------------------------------+
```

Figure 4-1 KLERRO.SNP File Contents

4.4 KLERR MESSAGES

KLERR is designed to run automatically without operator intervention. However, it does output some messages on the console and these can be useful in diagnosing a system problem. The following is a list of possible error messages and their causes.

Argument Out of Range

The number that was to be loaded into the burst count register was greater than the maximum number allowable.

KL10 CRAM Address Error

The CRAM address that was to be read is not a valid CRAM address.

Can't Clear KL10 Clock

The attempt to clear the KL10 clock has failed.

Can't Clear KL10 Run Flop

An attempt to clear the Run Flop has failed.

KL10 Clock Error Stop

A check of diagnostic register 1 reveals that there is an error in the KL10 Clock.

Can't Find KL10 Halt Loop

The microcode would not go into its halt loop even when told to do so.

Can't Sync KL10 Clock

The function execute to synchronize the KL10 clock has failed.

Can't Set KL10 Run Flop

KLERR was unable to set the Run flop.

DTE-20 Not Privileged, No KL10 Operations are Legal

This DTE is not the privileged DTE that is allowed complete access to the KL10. The privilege switch on the outside of the cabinet of the DECSYSTEM-20 should be checked.

DTE-20 Status Failure

An attempt to read or write the DTE status register failed.

Run and Halt Loop Both On

The KL10 thinks that the microcode is currently in the halt loop and also running normally, both at the same time.

EBOX Clock Timeout

The EBOX clock timed out during an attempt to simulate the EBOX clock via MBOX clocks.

EBUS Parity Error

Parity errors have been detected on the EBUS.

Function Read Failed

A Function Read operation failed.

Function Write Failed

A Function Write operation failed.

Function Execute Failed

A Function Execute operation failed.

Illegal Function Code

The code in the argument of the Function Read, Function Write, or Function Execute command does not match existing values.

Internal Programming Error

There is a problem in the system software.

Examine Deposit Mode Illegal

The arguments for a Deposit or Examine of the KL10 were not set up correctly.

Odd Function Code

The number of the Function Read, Function Write, or Function Execute does not match existing values.

Unmatched Error Code

The error code reported does not match any on the list of known errors.

KL in Halt Loop

The PDP-11 put the KL into the halt loop.

KL not in Halt Loop

The PDP-11 could not put the KL10 into the halt loop.

Version V02-02 Running

Banner when KLERR starts to run.

Directory File Not Found

Could not find the directory file into which the error file would be placed.

KLERRO.SNP File Creation Failed

Creation of the error file failed.

Unable to Enter KLERRO.SNP into Directory

There was an error when the file name was inserted into the directory file.

Unable to Extend KLERRO.SNP file

KLERR was denied access when trying to append to KLERRO.SNP.

UNABLE TO WRITE KLERRO.SNP file

There was an error when writing the KLERRO.SNP file.

4.5 KLXFER

KLXFER is not run immediately after KLERR or KLINIT. At that stage the KL10 is not up and running. The front end must wait until TOPS-20 has been loaded and is running. KLXFER runs after the SETSPD task in the front end runs. SETSPD runs when the KL10 has just started running and wants information about line speeds from the front end. This SETSPD is not to be confused with the SETSPD that runs on TOPS-20. The last thing that SETSPD does is to make a request for KLXFER to run. KLXFER then transfers the information to TOPS-20 or TOPS-10 via the DTE-20. TOPS-20 or TOPS-10 will then append the information to ERROR.SYS, the master error file. KLERRO.SNP is then deleted from the front-end file area.

CHAPTER 5

DTE HARDWARE OPERATION

There are four basic DTE20 hardware operations:

- Deposit/Examine
- TO11 Transfer/TO10 Transfer
- Doorbell Function
- Diagnostic Operations

5.1 DEPOSIT AND EXAMINE

Each of the four DTE20 operations has a preliminary phase where control information and/or data is loaded into the DTE registers. This is always done before the interface begins any operation. For the Deposit operation, the following information is always loaded into the indicated registers in the RAM by the PDP-11 processor:

Register	Word Loaded
1	Data Word 3
2	Data Word 2
3	Data Word 1
4	Address Word 1
5	Address Word 2

The deposit will load the contents of the three data words into the KL10 address specified by the address words. Bit 12 Address Word 1 specifies whether the operation is to be Examine or Deposit. If bit 12 is set (=1), the operation is a Deposit: if bit 12 is clear (=0), the operation is an Examine. For an Examine, the data words do not have to have any data loaded into them. The Examine will copy the contents of the specified KL10 memory address into the three data words. For a privileged front end, the protection bit (bit 11 of Address Word 1) can be set on by the software to perform an unprotected Deposit. For unprotected Deposits, the address space field (bits 15-13) specifies the type of address. Currently, three types of address space may specified:

Bits 15-13	Space
0	Executive Process Table
1	Executive Virtual Address Space
4	Physical Address Space

When Address Word 2 is loaded, the operation begins.

5.2 TRANSFER OPERATIONS

The TO10 and TO11 transfer operations pass variable length data between the two processors. The sender of the data must specify the address of the source string. The KL10 controls the address either to or from the KL10 via byte pointers in the Executive Process Table (EPT). The PDP-11 controls the address to or from the PDP-11 via two locations in the DTE (one word for each direction of transfer). It is the responsibility of the receiver to control scatter writes. The PDP-11 specifies the transfer rate (via the delay count) and the type of transfer. Bit 13 in the TO11 Byte Count word controls whether the DTE is in byte mode or word mode (1=byte mode, 0=word mode). Byte mode transfers 8-bit bytes while word mode moves 16-bit bytes.

When transferring string data from the KL10 to the PDP-11, the following DTE registers must be loaded by the PDP-11:

Register	Word Loaded
0	Delay Count
7	TO11 Byte Count
11	TO11 PDP-11 Address

The TO11 Byte Count register holds a negative number equal to the number of bytes to be transfered. As each byte is transferred, this register is incrementd by one. When the byte count reaches zero, the transfer is over. To allow for scatter reads, there is a special provision in the byte count word. This allows the receiver of data (and only the receiver) the option of being interrupted before the transfer is complete. At this point, another transfer can be started without reloading all of the parameters just by changing the address. The transfer in progress continues from the new address. On termination of the transfer, the PDP-11 is interrupted. The KL10 can be interrupted also if desired.

The TO11 Delay Count word is used to force the DTE to pause before starting each transfer, thus eliminating bursts of interrupts. The register is loaded with some negative number that is subsequently incremented by the DTE clock. The transfer can not start until Bit 13 equals zero. During transfers. TO11DT is used as a temporary buffer for the data until it is sent to PDP-11 memory via the NPR facility. The tO10 transfer is very similar. The PDP-11 loads the following registers:

Register	Word Loaded
0	TO10 Delay Count
10	TO10 PDP-11 Address

The KL10 loads the following register:

6	TO10 Byte Count

The TO10 operation uses the NPR operation to fetch data and then the KL10 is interrupted to give the character. TO10DT is used in the same way as TO11DT, as a temporary buffer used in conjunction with the delay count. The KL10 is interrupted when the transfer is finished.

5.3 DOORBELL FUNCTION

The doorbell function allows the KL10 to interrupt each PDP-11 connected to it by a DTE20 and vice versa. The doorbell consists of a programmable interrupt and a status bit. In order for the PDP-11 to interrupt the KL10, the PDP-11 sets the request-10 interrupt flip-flop (bit 8 in the DTE status word). When this bit is set, the DTE20 generates an interrupt in the KL10 with a status bit set in the CONI word (bit 26) indicating that the PDP-11 CPU has programmed an interrupt of the KL10.

This procedure works in a reversed but identical manner for the KL10 interrupting the PDP-11. The KL10 sets the request-11 interrupt by doing a CONO to the DTE20. The PDP-11 discovers the cause for the interrupt by looking at TO11DB (bit 11 in the DTE status word). Communications is then performed by loading one or more words in the communication region of KL memory. The Deposit and Examine features are used by the PDP-11 to gain access to these words.

5.4 PROTOCOLS

The KL10/PDP-11 DTE20 protocol is a tightly coupled communications protocol designed for use between the processors in a KL10-based system and their PDP-11 front-end computers. There are two "sub-protocols" within this protocol: Secondary Protocol and Primary Protocol.

5.4.1 Secondary Protocol

Secondary Protocol uses only the Examine/Deposit feature of the DTE20. Secondary Protocol is only used in special situations such as booting the KL10 or in emergencies where normal communications with the KL10 are not available. Only the privileged PDP-11 can run Secondary Protocol because it requires privileged Deposits and Examines. The code to support Secondary Protocol is in module BOOT of RSX-20F. Secondary Protocol is not a real protocol because the two processors do not cooperate. The PDP-11 decides what it wants to do and does all the work.

5.4.2 Primary Protocol

Primary Protocol is the main protocol used for communications and uses the Doorbell, Examine/Deposit and Byte Transfer features of the DTE20. The switch to Primary Protocol is made just before the [PS MOUNTED] message appears at the CTY.

Primary Protocol is a queued protocol because the DTE20 is used by several tasks. Multiple tasks cannot be allowed to use the DTE20 whenever they desire or confusion would result. Therefore, tasks that want to use the DTE must line up in a queue and wait their turn.

5.5 QUEUED PROTOCOL

The queued protocol driver is responsible for many things: controlling the exchange of data between the KL10 and the PDP-11 via the DTE, scheduling the transmission of information packets sent across the DTE, and interfacing between the KL10 and the PDP-11 device drivers that must communicate with it (terminals, line printers and card readers). It is the queued protocol driver that places output data in the thread packets for terminals and line printers. It is the queued protocol driver that takes data from card readers and terminals, bundles them into packets and sends them off to the KL10. When device status information is needed, it is the queued protocol driver that gathers the information for those devices that must report to the KL10.

The functions that the queued protocol driver can perform are:

1. To-11 initial message -- first message sent after KL10 enters Primary Protocol.

2. From-11 CTY DLS alias -- the CTY is accessible to the KL10 through the CTY device code or through the console front end. This function informs the KL10 to which DLS (data line scanner) line the CTY is "connected."

3. To-11 string data -- general data transferring mechanism of the protocol.

4. From-11 string data -- see (3) above.

5. To-11 character data -- allows the protocol to handle data transfers for several lines with a single function. Cuts down overhead by reducing number of messages.

6. From-11 character data -- see (5) above.

7. To-11 request device status -- requests the -11 to return the status of a device.

8. From-11 request device status -- requests the KL10 to return the status of a device.

9. To-11 device status -- response to (8) above.

10. From-11 device status -- response to (7) above.

11. To-11 request date/time -- normally sent at KL10 startup to get current date and time from the -11.

12. From-11 request date/time -- normally sent at -11 startup to get the current date and time from the KL10.

13. To-11 here is date/time -- response to (12) above.

14. From-11 here is date/time -- response to (11) above.

15. To-11 flush output for line - provides the CTRL/O function.

16. To-11 send all -- type out a specified string on all TTY-type devices connected to the -11.

17. From-11 dataset connected -- sent to the KL10 after a dataset line rings and is answered by the -11.

18. To-11 hang up dataset -- causes the -11 to hang up the specified dataset line.

19. From-11 dataset hung up -- confirmation of (18) above.

20. To-11 acknowledge -- sent when the KL10 has finished processing a data transfer for a device. This makes the entire buffer space for that device available to the -11 for further transfers.

21. From-11 acknowledge -- sent when the -11 has finished processing a data transfer for a device. This makes the entire buffer space for that device available to the KL10 for further transfers.

22. To-11 XOFF line -- used by the KL10 to produce the effect of CTRL/S.

23. TO-11 XON line -- allows the KL10 to resume TTY output suspended by function (22). Produces the effect of CTRL/Q.

24. To-11 here is line speed -- causes the -11 to set the line speed to specified values.

25. From-11 here is line speed -- informs the KL10 of a line's status. Sent after the KL10 has been reloaded and re-enters Primary Protocol.

26. To-11 here are line buffer allocations -- specifies the maximum amount of data that a device can accept between acks.

27. From-11 here are line buffer allocations -- specifies the maximum amount of data that a device can accept between acks.

28. From-11 here is reboot information -- provides the KL10 with the settings of the PDP-11 switch register.

29. To-11 ack all devices -- used by the KL10 to start the from-11 data again after the KL10 has temporarily left Primary Protocol. This can happen if the KL10 goes into EDDT or does a parity sweep.

30. From-11 ack all devices -- used by the -11 to start the to-11 data again.

31. To-11 turn device on/off line -- tells the -11 to turn a specified device on- or off-line.

When two processors communicate, they require an area that both can access to exchange information. This area is in KL memory because the KL10 can not access PDP-11 memory via the DTE. The area where this common data is stored is called the Communications Region.

The first part of the region is a header area. Following this, each processor that is connected to the KL10 has its own communications area. A minimal configuration has two communications areas, one for the KL10 and one for the PDP-11 front end. If there is another PDP-11 for data communications, there will be three communications areas. Each communications area has one section of data about itself and one or more sections for the other processors it is connected to.

The KL10 and the -11 also use the Executive Process Table (EPT) to communicate. The EPT occupies a page in KL memory in which several words are reserved for DTE communications. The location of the EPT is always known because a hardware register points to it.

The EPT stores the KL10 addresses for byte-transfer operations and tells the PDP-11 where it may Examine and Deposit into KL memory. The relocation and protection words are set once by the KL10. The PDP-11 reads those words, and stores them so it will not have read them again. All PDP-11 Examines and Deposits of KL memory are checked by the hardware using the relocation and protection words. There is also a command word so the KL10 can give RSX-20F information that it can use to decide what to do as in the case of rebooting.

5.6 DIRECT AND INDIRECT TRANSFERS

There are two separate packet-transfer operations that are used by the KL10 and the PDP-11 to exchange messages across the DTE20. One is called direct transfer and the other is called indirect transfer. The difference between the two is largely a matter of the size of the packet of information to be transmitted. Direct transfers are used to send relatively small, fixed-length packets where the data is included in the packet. Indirect transfers are used to send longer, variable-length packets where the data is separate from the packet. Both operations use Deposit/Examines and byte transfers.

5.6.1 Direct Packets

Direct packets are identified by a 0 in bit 15 of the second word. The packet shown in Figure 5-1 is the largest size direct packet that can be sent across the DTE. Direct packets can be as short as 5 words and as long as 10 words.

```
+--------------------------------------+
!      Count of bytes in packet        !    0
!--------------------------------------!
!0!           Function code            !    2
!--------------------------------------!
!             Device code              !    4
!--------------------------------------!
!             Spare (null)             !    6
!--------------------------------------!
!                 Data                 !   10
!--------------------------------------!
!                 Data                 !   12
!--------------------------------------!
!                 Data                 !   14
!--------------------------------------!
!                 Data                 !   16
!--------------------------------------!
!                 Data                 !   20
!--------------------------------------!
!                 Data                 !   22
+--------------------------------------+
```

Figure 5-1 Direct Packet

5.6.2 Indirect Packets

Indirect packets can handle more information than direct packets and can be written in noncontiguous areas of memory. The data that is sent is not a part of, nor necessarily contiguous to, the packet itself. The indirect packet, shown in Figure 5-2, is identified by a 1 in bit 15 of the second word. The size is 12 bytes and last two words provide the size and location of the data to be transmitted across the DTE.

```
+---------------------------------------+
!       Count of bytes in packet        !     0
!              (12 bytes)               !
!---------------------------------------!
!1!          Function code              !     2
!---------------------------------------!
!            Device code                !     4
!---------------------------------------!
!            spare (null)               !     6
!---------------------------------------!
!       Count of bytes in data          !    10
+---------------------------------------+
!          Pointer to data    *         !    12
+-----------------------------!---------+
                              !
                              +-------------+
                                            !
                                            !
+---------------------------------------+<---+
!                                       !
!                                       !
!                                       !
!                                       !
!                 Data                  !
!                                       !
!                                       !
!                                       !
!                                       !
!                                       !
+---------------------------------------+
```

Figure 5-2 Indirect Packet and Data

5.7 DATA STRUCTURE OF PACKETS

The information that is sent from the PDP-11 to the KL10 via the DTE is stored in dynamically allocated buffers. The buffers are located in the free pool area of -11 memory. Each packet to be sent is stored in a buffer area called a node. Each node has two words of overhead referred to as the node header. One word contains a pointer to the next node in the list; the other word contains the size of the buffer. The remainder of the information is the actual packet that will be sent.

The To-10 queue is maintained by the use of pointers. The location TO10Q points to the first node in the queue. The To-10 queue operates in a first-in/first-out manner. Location TO10Q+2 contains a pointer to the last node in the collection of nodes. This pointer is necessary when adding nodes to the list.

The To-11 queue handles the data received from the KL10. It uses space from the free pool area of -11 memory to form buffers for nodes, much as the To-10 queue. It uses the nodes in the same way as the To-10 queue except that a count of nodes replaces the pointer to the last node.

APPENDIX A

FRONT END TASKS

The tasks that are listed here are those tasks that exist separate from the RSX-20F Exec. They reside in the front end's file area from which they can be loaded into core and executed in either the GEN user partition or the F11TPD system partition.

F11ACP.TSK FILES-11 Ancillary Control Processor
An Ancillary Control Processor (ACP) is an extension of the Monitor. F11ACP handles the front-end disk files. It performs file access, management, and control functions. F11ACP runs in the F11TPD partition.

PARSER.TSK The Command Parser
PARSER is the primary means of access to the front-end programs. It also controls the KLINIK link and provides KL10 diagnostic tools. PARSER runs in the GEN partition.

KLE.TSK KLERR
KLERR processes KL10 errors using the diagnostic DTE functions. It takes a "snapshot" of KL10 error conditions and stores it as the file KLERRO.SNP in the front-end file system. It then calls KLINIT to restart the system. KLERR runs in the GEN partition.

KLI.TSK KLINIT
KLINIT initializes the KL10 processor by loading the microcode, configuring memory, configuring cache, and then loading and starting the KL10 bootstrap program. KLINIT runs in the GEN partition.

KLXFER.TSK KLXFER
KLXFER transfers KLERRO.SNP (the "snapshot" taken by KLERR) across the DTE to the KL10 where it is placed in the ERROR.SYS file. KLXFER runs in the GEN partition.

MOU.TSK Mount a Device
MOUNT makes a device known to F11ACP so that it can be accessed by a given user. MOUNT runs in the GEN partition.

PIP.TSK Peripheral Interchange Program
PIP performs general file transfers and some maintenance functions between FILES-11 devices and other peripherals. PIP runs in the GEN partition.

FRONT END TASKS

TKTN.TSK

Task Termination Program
TKTN outputs task termination notification and provides for the orderly termination of front-end tasks. It also acts as an interface between KLINIT and KLERR. TKTN runs in the GEN partition.

COP.TSK

Copy from device to device
COPY is a device copy utility that also allows verification of the physical state of the device. COPY supports both floppy disks and DECtapes. COPY runs in the GEN partition.

RED.TSK

Redirect the system device
REDIRECT moves the front end's system device from one FILES-11 device to another and informs the system of its new location. REDIRECT runs in the GEN partition.

INI.TSK

Initialize volumes
INI initializes FILES-11 devices to be recognizable FILES-11 volumes and sets up Master Directory space, index, home block, etc. INI runs in the GEN partition.

UFD.TSK

User File Directory
UFD creates User File Directories on FILES-11 volumes. User File Directories are used to store file identifiers. UFD runs in the GEN partition.

T20ACP.TSK

TOPS-20 Ancillary Control Processor
T20ACP is the file handler for files to be transferred to and from the KL10's disk file area. It interacts with the TOPS-10 and TOPS-20 pseudo device "FE:". T20ACP provides access to the TOPS-10 and TOPS-20 disk file areas in terms compatible with FILES-11 operations. T20ACP runs in the GEN partition.

SAV.TSK

Save system image
SAV creates a task-image file of the current RSX-20F monitor and saves it in the FILES-11 area. SAV runs in the GEN partition.

DMO.TSK

Dismount a Device
DMOUNT declares a device off-line to F11ACP and therefore inaccessible to a user. DMOUNT runs in the GEN partition.

SETSPD.TSK

Set Line Speeds
SETSPD sets the line speed table in the KL10 after a restart. It also sets the time in the KL10 processor. SETSPD.TSK is a front-end task and is not to be confused with the TOPS-20 program, SETSPD.EXE. SETSPD runs in the F11TPD partition.

KLRING.TSK

KLINIK Request
KLRING checks the KLINIK time window and password whenever the KLINIK line rings. If the time and security checks are verified, KLINIK is enabled. KLRING runs in the F11TPD partition.

KLDISC.TSK

KLINIK Disconnect
KLDISC performs system functions associated with disconnecting the KLINIK line. KLDISC also logs significant KLINIK events across the DTE into the KL10 ERROR.SYS file. KLDISC runs in the F11TPD partition.

FRONT END TASKS

MIDNIT.TSK

Update the clock
Each time the clock passes midnight, MIDNIT updates the time and date on the -11. Then, if the KL10 is running, MIDNIT obtains the KL10's time and date and resets its own to match. MIDNIT runs in the F11TPD partition.

The following two tasks are KL10 programs that are distributed as part of the front-end software and reside in the front-end file system. They are read into the -11 by the KLINIT task and deposited over the DTE into KL10 memory where they perform their specific functions.

BOOT.EXB

Bootstrap Program
BOOT loads the KL10 monitor system image file into memory from rigid disk (BOOT.EXB is an executable binary file).

MTBOOT.EXB

Magtape Boot Program
MTBOOT loads the KL10 monitor system image file into memory from a magnetic tape (MTBOOT.EXB is an executable binary file).

APPENDIX B

RSXT10 AND FE UNDER TOPS-10

RSXT10 is a TOPS-10 program used to make TOPS-10 files compatible with the FILES-11 file system on the RSX-20F front-end. FE (under TOPS-10) and PIP (under RSX-20F) are the programs that are used to effect file transfers bidirectionally between TOPS-10 and RSX-20F. Both of these programs execute in a normal timesharing environment.

B.1 RUNNING RSXT10

RSXT10 is available to all users on the ersatz device named SYS:; it does not normally require any special privileges to execute. RSXT10 has an associated HELP file and DOC file.

RSXT10 is invoked like any other TOPS-10 CUSP:

```
.R RSXT10<CR>
```

RSXT10 responds with the prompt:

```
RSXFMT>
```

Files are converted using the following command format:

```
RSXFMT>CONVERT input-fileid output-fileid<CR>
```

The input-fileid and output-fileid are normal TOPS-10 file specifications. Note that the input and output file specifications are separated by a space.

After converting a file from one format to another, RSXT10 will type the following message on the terminal:

```
input-fileid [input mode] ==> output-fileid [output mode]
```

To obtain information about input modes, refer to RSXFMT.DOC, the documentation file.

To exit from RSXT10 type:

```
RSXFMT>EXIT<CR>
```

B.2 RUNNING FE

The TOPS-10 program FE must execute under a privileged [p,pn], and may run detached if this is desirable. Beyond this, FE requires access to the UIC/[p,pn] mapping file: SYS:FEUIC.TXT. Each invocation of FE will cause this file to be read.

FEUIC.TXT is an ASCII file which is created and maintained with standard TOPS-10 Text Editors (TECO, SOS). The format of the UIC to PPN mapping descriptor is:

```
     [uic]=str:[p,pn]
```

[uic] must already exist in the front-end file system. STR: must be a valid TOPS-10 structure name, and [p,pn] must be a valid TOPS-10 directory. The default for STR: is DSK:. FEUIC.TXT may contain as many UIC to [p,pn] mappings as required. Furthermore, these mappings may be internally documented by the insertion of comments which must begin with a semicolon or an exclamation mark.

FE may be invoked by either:

```
     .R FE<CR>
```

or

```
     .GET SYS:FE<CR>
     JOB SETUP
     .CSTART<CR>
     .DETACH<CR>
```

FE does not accept commands.

B.3 THE FE: DEVICE

The FE: pseudo-device exists under both TOPS-10 and RSX-20F. The correct interpretation of the FE: device under either system should be "the-other-file-system".

B.4 RSX-20F TASKS

In order to transfer files between the file systems, three RSX-20F tasks must be invoked and released; they are MOU (MOUNT), PIP (file transfer), and DMO (DMOUNT). These RSX-20F tasks are invoked via the PARSER which, in turn, is invoked by typing a control-backslash at the CTY. Type:

```
     CTRL/\
```

The system responds with the PARSER prompt:

```
     PAR>
```

The tasks themselves are invoked by the MCR command. For example:

```
     PAR>MCR MOU<CR>
```

invokes the MOU (MOUNT) task. All RSX-20F tasks prompt by typing their three-character task name and a right bracket. All RSX-20F tasks are released by typing a control-Z.

B.5 TRANSFERRING FILES

The following sequence of steps is used to transfer files both to and from the front-end file system:

1. Assign the TOPS-10 FE: device to your job.

2. Run (and detach) FE under TOPS-10.

3. Mount the RSX-20F FE: device.

4. Transfer the file(s) under RSX-20F PIP.

5. Dismount the RSX-20F FE: device.

6. Stop FE under TOPS-10.

7. Deassign the TOPS-10 FE: device from your job.

This basic sequence is used for all file transfers. If files were being transferred to the front end, those files would have to be converted by RSXT10 before they were transferred. If files were being transferred to KL10, those files would have to be converted after they had been transferred. In other words, conversion always takes place in TOPS-10.

To transfer files from TOPS-10 to RSX-20F, invoke the RSX-20F PIP task and type the following command string:

```
PIP>[uic]filename.ext/NV=FE:[uic]filename.ext<CR>
```

To transfer files from RSX-20F to TOPS-10, invoke the RSX-20F PIP task and type the following command string:

```
PIP>FE:[uic]filename.ext/NV=[uic]filename.ext<CR>
```

Refer to Section 3.3 for details on the RSX-20F utility, PIP.

APPENDIX C

RSX-20F STOP CODES

The following is a list of possible RSX-20F stop codes. Associated with each is the name of the module that issued the stop code, a short explanation of the error, and a possible cause of the error.

Code Module Meaning

BF1 QPRDTE BUFFER FAILURE 1

Attempt to obtain buffer space for the TO-11 protocol header failed.

Possible Cause:

Buffer pool space has become exhausted or highly fragmented. R1 contains the node (buffer) size requested. .FREPL points to the list of free nodes. Nodes are linked together in the forward direction through the first word of the node. The second word of each node contains the node size.

BO2 QPRDTE BUFFER OVERFLOW 2

The -11 was not able to obtain the buffer space required to receive an indirect data transfer from the -10.

Possible Cause:

Same as BF1 above.

B03 SCOMM BUFFER OVERFLOW 3

The -11 was not able to obtain the buffer space necessary for data it wanted to send to the -10.

Possible Cause:

Same as BF1 above.

CBR PF CROBAR ERROR

DTE-20 power has not returned after a power-fail restart. RSX-20F allows it 30 seconds to reappear.

Possible Cause:

Malfunctioning hardware in the KL10.

RSX-20F STOP CODES

Code	Module	Meaning
DTB	QPRDTE	TO-11 DTE TRANSFER FAILURE A To-11-done interrupt has occurred but the To-11 address in the DTE TO11AD register (register 22) did not have the expected value. Since TO11AD is incremented for each byte transferred, it should point to the first word following the buffer into which the To-11 data was written. Possible Cause: The -11 received the wrong byte count or, more likely, the DTE has a hardware malfunction. TO11BC contains the negative count of data that was actually transferred. TO11AS contains address of data node. R1 contains expected termination address and CR$DTB-2 contains the actual termination address for transfer.
DTD	COMTRP	UNIBUS TIMEOUT Reference to the DTE-20 caused a UNIBUS timeout. Possible Cause: Malfunction of the hardware in the KL10.
DTF	QPRDTE	TO-10 DTE TRANSFER FAILURE A To-10-done interrupt has occurred but the To-10 address in the DTE TO10AD register (register 20) did not have the expected value. Since TO10AD gets incremented for each byte transferred, it should point to the first word following the packet which was sent to the 10. Possible Cause: The -11 gave the -10 the wrong byte count or, more likely, the DTE has a hardware malfunction. TO10SZ contains the size of the transfer and TO10AS the start address. The expected termination address is in R4.
ETE	QPRDTE	TO -11 TRANSFER ERROR A DTE interrupt occurred with the TO11ER bit set in the DTE status register (register 34). Possible Cause: Hardware malfunction along the data path between the -10 and -11 (M-box, E-box, E-bus, DTE-20, through to 11-memory).

RSX-20F STOP CODES

Code	Module	Meaning
FTA	LC	FILES-11 TASK ABORTED A task occupying F11TPD partition has aborted and the task termination notification task (TKTN) cannot be started since it too runs in the F11TPD partition. Possible Cause: .TKTN itself aborted. R5 and .CRTSK point to the active task list (ATL) node of the aborted task.
IAS	SCH	UNKNOWN SIGNIFICANT EVENT An unused bit in .SERFG has been set. Possible Cause: PDP-11 hardware malfunction or corrupted software in PDP-11. .SERFG has the bit set.
ILF	QPRDTE	ILLEGAL PROTOCOL FUNCTION The function code in a To-11 protocol header specified a function which is outside the legal range or which is currently unimplemented. Possible Cause: -10 software is corrupted or hardware malfunction along data path between -10 and -11. R1 contains the function code times two. R4 contains the address of the protocol header.
ILQ	QPRDTE	ILLEGAL QUEUE COUNT The -10 and the -11 disagree on the number of direct transfers that have thus far taken place from the -10 to the -11. Possible Cause: The -11 is missing TO-11 doorbell interrupts or either the -10 or the -11 software is corrupted. STATI+0 to STATI+2 contain the -10's To-11 status word as read by RSX-20F at the last examine. STATI+4 is the count as the -10 expects it and TO10QC is the count as the -11 expects it.
LRF	SCH	LOAD REQUEST FAILURE An attempt to load a nonresident monitor routine into the F11TPD partition failed. Possible Cause: The FILES-11 system is incomplete or damaged.

Code	Module	Meaning
MPE	LC	MEMORY PARITY ERROR A memory parity error has occurred in the PDP-11 (trap to location 114). The memory status registers are stored starting at location PARSAVE. See the PDP-11 Processor Handbook for details.
PT1	QPRDTE	PROTOCOL BROKEN An illegal protocol device number was specified in To-11 request. It was found to be greater than the maximum allowed device number .DQPSZ (currently 10). Possible Cause: -10 software is corrupted or hardware malfunction along the data path between the -10 and -11. The device number from the protocol header is in TO11DV.
PT2	QPRDTE	PROTOCOL ERROR 2 An illegal protocol function was specified in a To-11 request. The function was found to be greater than the allowed maximum BC.FNM (currently 34). Possible Cause: Same as PT1 above. The function code from the protocol header is in TO11FN.
PT3	QPRDTE	PROTOCOL ERROR 3 The -11 has received a doorbell interrupt from the -10. The indirect bit in the -10's To-11 status word indicates that an indirect transfer is to be initiated. The function code, however, sent in the last protocol header, does not indicate that an indirect request is in progress (the most significant bit of the function code was not set). Possible Cause: Same as PT1 above. TO11FN contains the function code and STATI contains the To-11 protocol status word.
PT4	QPRDTE	PROTOCOL ERROR 4 The -10 wants to send a packet to the -11 and the packet size is greater than the maximum allowed size of 100. Possible Cause: Same as PT1 above. The size is in EQSZ.
RED	RED	REDIRECT ERROR A fatal error has occurred during an MCR REDIRECT command.

Code Module Meaning

RES LC RESERVED INSTRUCTION TRAP

This is the PDP-11 trap to location 10. An attempt was made to execute an illegal or reserved instruction. See the PDP-11 Processor Handbook for further details.

Possible Cause:

-11 software is corrupted or there is a PDP-11 hardware malfunction.

TBT LC T-BIT TRAP

This is the PDP-11 trap to location 14. It occurs when the BPT instruction (not used by RSX-20F) is executed or when the T-bit is set. See the PDP-11 Processor Handbook for further details.

Possible Cause:

Corrupted -11 software or PDP-11 hardware malfunction.

TET QPRDTE TO-10-TRANSFER ERROR

A DTE interrupt has occurred with either TO10ER (To-10 error) or MPE11 (-11 parity error) bit set in the DTE-20 status register (register 34).

Possible Cause:

DTE hardware error, PDP-11 memory parity error, or hardware malfunction along the data path between the -11 and -10.

T04 LC TRAP AT LOCATION 4

The PDP-11 traps to location 4 when making a word reference to an odd address or at the occurrence of a bus timeout. See the PDP-11 Processor Handbook for further details.

Possible Cause:

-11 software is corrupted, or a PDP-11 peripheral device is malfunctioning or has gone away.

UIE QPRDTE UNIMPLEMENTED PROTOCOL FUNCTION

The -10 uses bits 0-2 of its To-11 status word in the communications region to inform the front end of any disaster occurring in the KL. These bits are read by the front end on receipt of a To-11 doorbell. The currently implemented functions are KL-RELOAD REQUEST and KL POWER FAIL. Any other bits that are set cause this halt.

Possible Cause:

Corrupted -10 software, -10 hardware malfunction or any hardware malfunction along data path between -10 and -11.

INDEX

RSX-20F
System Specifications
AA-H213A-TK

READER'S COMMENTS

NOTE: This form is for document comments only. DIGITAL will use comments submitted on this form at the company's discretion. If you require a written reply and are eligible to receive one under Software Performance Report (SPR) service, submit your comments on an SPR form.

Did you find this manual understandable, usable, and well-organized? Please make suggestions for improvement.

Did you find errors in this manual? If so, specify the error and the page number.

Please indicate the type of reader that you most nearly represent.

- □ Assembly language programmer
- □ Higher-level language programmer
- □ Occasional programmer (experienced)
- □ User with little programming experience
- □ Student programmer
- □ Other (please specify)__________

Name__________ Date__________

Organization__________ Telephone__________

Street__________

City__________ State__________ Zip Code
or
Country__________

Please cut along this line.

-- Fold Here --

-- Do Not Tear - Fold Here and Staple --

FIRST CLASS
PERMIT NO. 152
MARLBOROUGH, MA
01752

BUSINESS REPLY MAIL
NO POSTAGE STAMP NECESSARY IF MAILED IN THE UNITED STATES

Postage will be paid by:

digital

Software Documentation
200 Forest Street MR1-2/E37
Marlborough, Massachusetts 01752