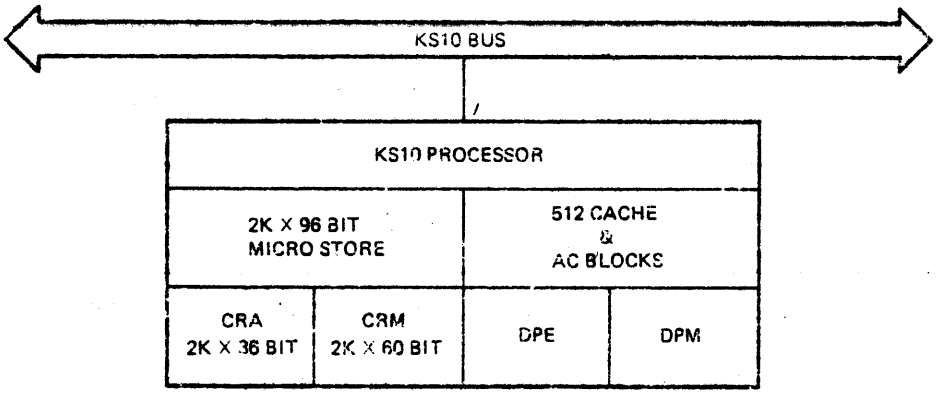
TOPS-10
ON THE
DECSYSTEM-2020



Software Specialists' Training Program

digital

digital

INTEROFFICE MEMORANDUM

TO: DISTRIBUTION

DATE: 21 MAR 79
FROM: Clem Daems CD:9-107
DEPT: Software Course Development
EXT: 231-5013
LOC/MAIL STOP: MR1-1/A86

SUBJ: TOPS-10 ON THE DECSYSTEM-2020

Enclosed is your copy of the TOPS-10 on the DECSYSTEM-2020 document developed by John Wyndham while he was a developer for Software Course Development.

This is an information document for TOPS-10 Software Specialists. It presents the 6.03A LIR implementation of the TOPS-10 monitor for the DECSYSTEM-2020. 2020 hardware, including the KS10 processor is also presented, as well as the modifications to TOPS-10 to accomodate hardware differences between the 2020 and DECSystem-10s.

The enclosed document is the Training Program for Software Specialists.

ccr

TOPS-10 ON THE DECSYSTEM-2020

Course Code: EH-J210A-98

This course reflects:

OPERATING SYSTEM: TOPS-10 6.03A LIR

HARDWARE: DECSYSTEM-2020

Educational Services
Digital Equipment Corporation
Marlboro, Masachusetts

TOPS-10 ON THE DECSYSTEM-2020

INTRODUCTION

This document, designed for the TOPS-10 Software Specialist, presents the 6.03A LIR implementation of the TOPS-10 monitor for the DECSYSTEM-2020. It describes the 2020 hardware, including KS10 processor, and the modifications to TOPS-10 to accomodate hardware differences between the 2020 and DECsystem-10s, such as the 1090 (KL10 processor).

This page is for notes.

TABLE OF CONTENTS

Page

Chapter 3 KS1Ø I/O Instructions

Chapter 4 TOPS-1Ø Changes for KS1Ø

Chapter 5 Other Software Changes

Page

Chapter 6 Installation Procedure

Chapter 7 Operator Procedures

Chapter 8 System Administrator Procedures

Page

FIGURES

Page

TABLES

Chapter 1

TOPS-10/DECSYSTEM-2020 FEATURES

PRODUCT OVERVIEW

The modification of the TOPS-10 operating system to run on DECSYSTEM-2020 hardware (KS10 processor) was not an original design goal; however, a significant need developed for this, both in the Digital customer base and in-house user groups. The main motivation for the modification was a low cost system that could reduce capital expenditure for Digital equipment in the coming years. Implementing TOPS-10 on the 2020 creates a low end system for TOPS-10 customers. There is also considerable interest in a small local system, which can be part of a network that includes larger, remote systems performing 'heavy duty' processing.

TOPS-10 on the 2020 combines two existing products - the well-established TOPS-10 Monitor and the DECSYSTEM-2020 hardware. The 2020 hardware is relatively new; it was originally presented as a low end DECSYSTEM-20 running the TOPS-20 monitor. The TOPS-10 monitor and associated software have now been modified to support 2020 specific hardware such as the KS10 processor, TU45 tape drives and the RH11 controller. MONGEN has also been simplified to facilitate installation and acceptance procedures.

The first release of TOPS-10 on the 2020 will be the 6.03A monitor (as modified for the 2020 hardware), followed some months later by the 7.01 monitor. The first release is termed the 6.03A LIR for the 2020. LIR (Limited Interim Release) is a Digital term for the distribution of a modified subset of a complete software product.

The target expectation is that TOPS-10 on the 2020 can match the performance of a KA10. Plans for communications are not finalized; product management has agreed, as a contingency, that product sales should be restricted to OEM and current customers at first.

SUMMARY OF FEATURES

From the viewpoint of the TOPS-10 timesharing user, there is little difference between the DECSYSTEM-2020 running TOPS-10 and the DECsystem-10. The TOPS-10 monitor modified to run on the 2020 hardware retains most of its familiar functions. The system supports up to 32 users (i.e., terminal lines) in a multiprogramming, timesharing and batch environment. The virtual memory capabilities of the TOPS-10 monitor are also retained.

The languages and related software to be supported include the following:

ALGOL
APL
BACKUP
BASIC
BLISS
COBOL - ANSI Standard 68, 74
CPL (A Subset of PL/1)
DBMS
FORTRAN
IQL (Interactive Query Language)
LINK
MACRO
MAKLIB
SORT

Most other DECsystem-10 software is also supported.

From the hardware standpoint, the DECSYSTEM-2020 offers the smallest and least expensive mainframe on the market. The KS10 processor, console microprocessor (which replaces the front-end subsystem found on the DECsystem-1090), main memory, and internal channels for mass storage and other devices, are all housed in a single cabinet of dimensions 5' x 2 1/2' x 2' (see Figure 1-1).

FRONT VIEW

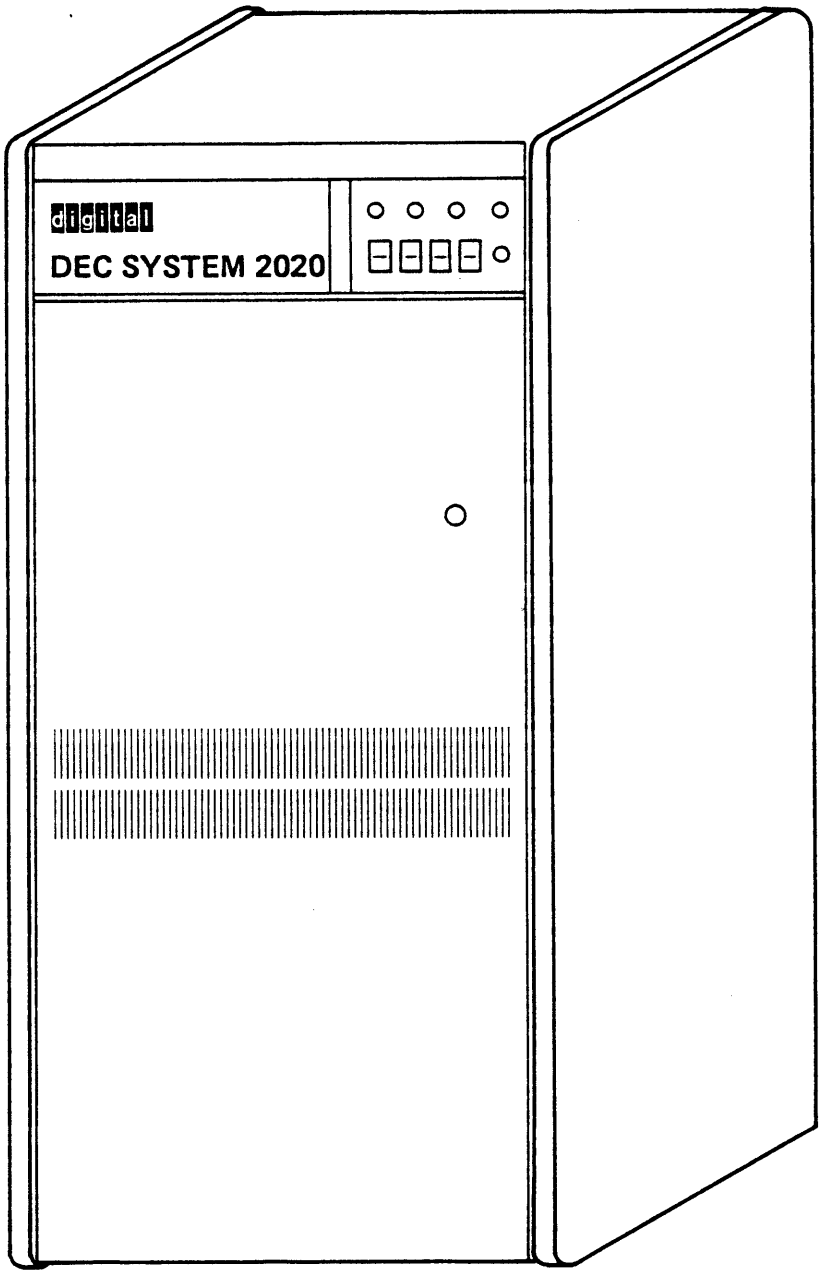


Figure 1-1.
2020 Cabinet

This significant reduction in the mainframe's size was accomplished by replacing ECL logic (Emitter-Coupled Logic) with Schottky TTL logic (Transistor-Transistor Logic) for the KS1Ø processor, and implementing MOS (Metal-Oxide-Silicon) memory. Besides leading to a size and price reduction, these changes produce the additional advantage of a very low power requirement - 14ØØ watts, or roughly the power consumed by a toaster. This means that a traditional computer room is not required, and the DECSYSTEM-2Ø2Ø can easily be incorporated into a standard office environment.

The DECSYSTEM-2Ø2Ø hardware configuration for use with the TOPS-1Ø monitor is as follows:

Processor:	KS1Ø	
Memory:	MS1Ø	(MOS memory, 192K-512K)
Controllers:	RH11	(two controllers, one for disk, one for tape)
Disks:	RPØ6, RMØ3	(1 - 8 drives)
Tape Drives:	TU45	(1 - 4 drives)
UNIBUS Adaptor:	UBA	(two, one for each UNIBUS)
Unit Record Equipment:	LP2Ø	Controller with Ø - 1 LPØ5/LP14 printers
	CD2Ø	Controller with Ø - 1 CR1ØE/F card readers
Communications:	DZ11	(8-32 asynchronous lines)
	KCM11/DUP11	(Ø-2 synchronous lines)

Unlike the traditional DECsystem-1Ø, the DECSYSTEM-2Ø2Ø hardware does not support card punches, DECtapes or paper tape reader/punches. The standard product requires magnetic tape for installation and Field Service reasons; however, the system will work without magnetic tape. A KLINIK line is available.

With a few exceptions (most notably the I/O instructions), the KS10 processor has the same instruction set as the KL10-D and KL10-E processors. The KS10 features 396 microprogrammed instructions; each instruction consists of a series of microinstructions that are implemented by the microcode, loaded into a 2K word (96 bits/word) random access memory at system startup. The microcode is the same as that used by TOPS-20 on the DECSYSTEM-2020. The basic microinstruction cycle time is 300 nanoseconds. As for the KL10 processor, there are eight sets of ACs of 16 words each.

The KS10 implements a 512 word cache memory, with an access time of 300 nanoseconds. With a cache hit rate of 80%, the effective memory access time for the KS10 is 420 nanoseconds. Main memory is MOS memory, and comes in 64K units up to a maximum of 512K. Main memory access time is 1.05 microseconds.

The console and remote diagnostic link (KLINIK) are handled by a microprocessor, which is loaded and started when the power is turned on. The microprocessor program is then used to load a bootstrap program into main memory to boot the TOPS-10 monitor.

OVERVIEW OF SYSTEM ARCHITECTURE

The DECSYSTEM-2020 has two types of bus, a KS10 bus that transfers 36-bit words, and an 18-bit UNIBUS that interfaces with an RH11 controller and other peripheral controllers. The KS10 processor, main memory and 8080 microprocessor all attach to the KS10 bus. The interface between a UNIBUS and the KS10 bus is a UNIBUS Adaptor.

There are two UNIBUS Adaptors, one which interfaces with the UNIBUS connected to the RH11 controller for disk drives, and the other which interfaces with the UNIBUS for tape drives and other peripherals. The UNIBUS Adaptors are numbered 1 and 3 (UBA2 is not used at present). The basic system architecture is shown in Figure 1-2. All hardware depicted in Figure 1-2, except for the peripherals themselves, is contained in the single corporate high boy cabinet.

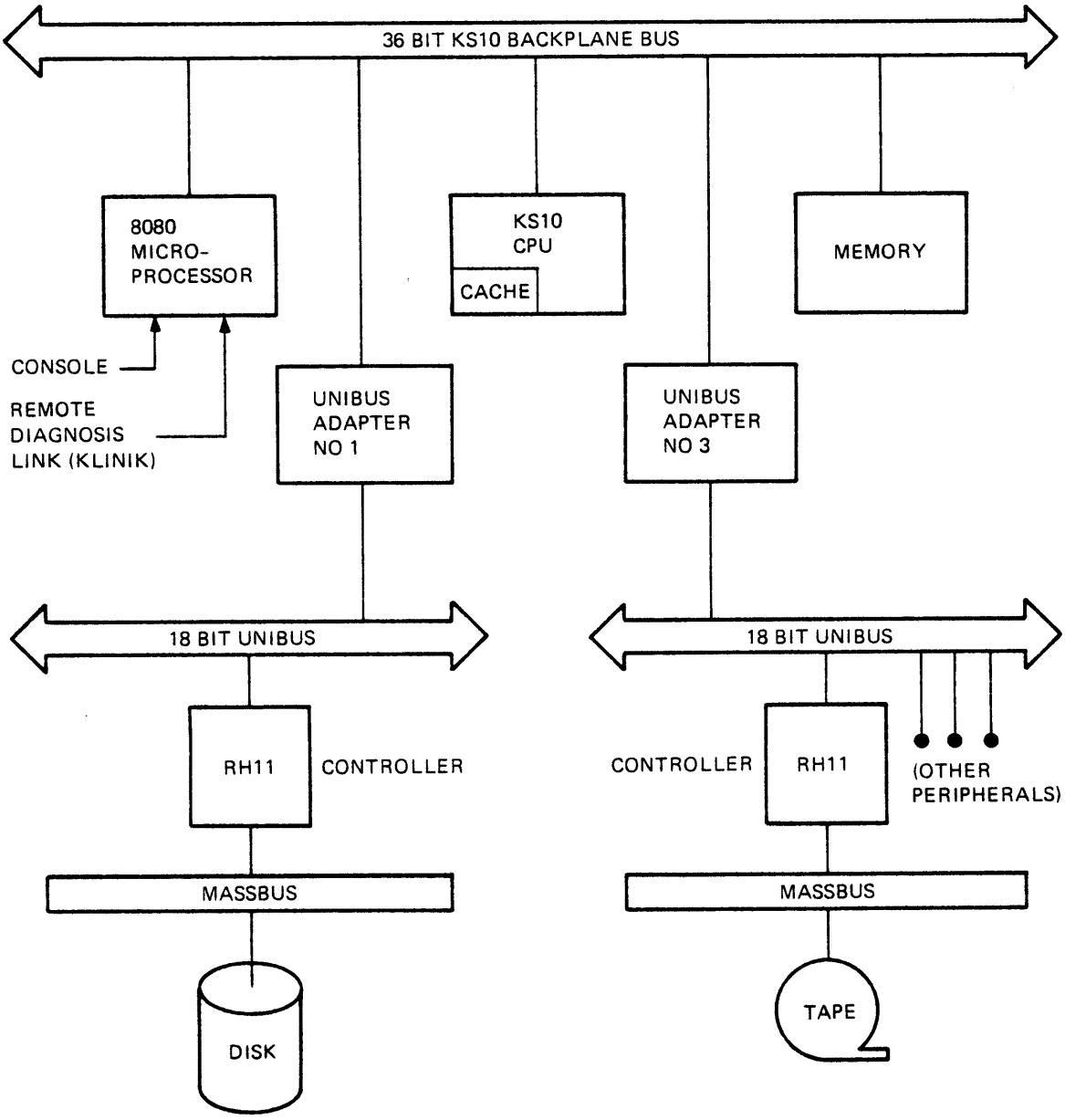


Figure 1-2.
2020 System Architecture

Chapter 2

DECSYSTEM-2020/DECSYSTEM-1090/1091 HARDWARE DIFFERENCES

KS10/KL10 COMPARISON

Since 1975, the KL10 processor has been the basis for the DECsystem-10 and DECSYSTEM-20 large-scale computer families. The KL10 exists in several versions; two versions are the KL10-D, which has been optimized for the DECsystem-1090 (high end of the DECsystem-10 line), and the KL10-E, which can support either TOPS-10 with the DECsystem-1091, or TOPS-20 for the DECSYSTEM-20 family. Both of these model KL10s support single section (section 0) or extended addressing. The main difference between them is that the KL10-D interfaces with memory (external) via a DMA (Direct Memory Access), and requires a DIA (DECsystem IBUS Adaptor) for interface with certain I/O devices, whereas the KL10-E supports internal memory and uses a DIA only as an option. In addition, the KL10-D and KL10-E are both model B processors, supporting extended addresses with up to 32 sections of 256K each. Previous models of the KL10 (KL10-A, KL10-B and KL10-C, all model A processors) supported single section (section 0) addressing only.

The KL10 processor comprises four major subsystems:

1. The Execution Box (E-Box), which executes instructions and controls other subsystems. The E-Box contains the arithmetic logic, fast memory (AC blocks) and microcontroller, which performs the instruction using microcode in a 2K RAM (Random Access Memory).

2. The main memory subsystem, consisting of a memory control unit (M-Box), associated busses, and internal core and/or MOS memory or external core memory.

3. The front-end diagnostic and console PDP-11 subsystem, which also supports unit-record equipment connected to the UNIBUS.

4. An I/O subsystem, which includes integrated channel/controllers for disk and magtape, an I/O bus (on some models) for other peripherals, and PDP-11 based communications subsystems.

Figures 2-1 and 2-2 illustrate fully configured KL10-D and KL10-E central processing units. Cache, memory and channel control are in the M-Box. There can be up to four DTEs (Digital Ten-to-Eleven) to accommodate the console and communication front-end subsystem.

The KS10 processor, introduced in 1978, provides a low-end configuration for both DECsystem-10 and DECSYSTEM-20 families. TOPS-20 on the DECSYSTEM-2020 was presented in the Software Specialists' DECSYSTEM-2020 Training Course. The present document is a parallel treatment of the DECSYSTEM-2020 for the TOPS-10 monitor.

The KS10 processor comprises four major subsystems:

1. KS10 CPU and memory. The KS10 CPU consists of only four boards, uses Schottky TTL logic as opposed to KL10 ECL logic, and includes a 512 word cache memory and 2K RAM for the instruction microcode. The memory is internal MOS memory.

2. Console and Diagnostic Microprocessor. This is an 8080 microprocessor that handles a CTY and a KLINIK line.

3. KS10 Bus and UNIBUS Adaptors. The KS10 bus carries 36-bit words and other control bits. The UNIBUS Adaptors interface both PDP-11 type controllers and UNIBUS to the KS10 bus.

4. Peripheral Controllers. These include PDP-11 type controllers (RH11) for disk and magtape, LP20 and CD20 controllers for line printer and card reader, and DZ11 controllers for terminals.

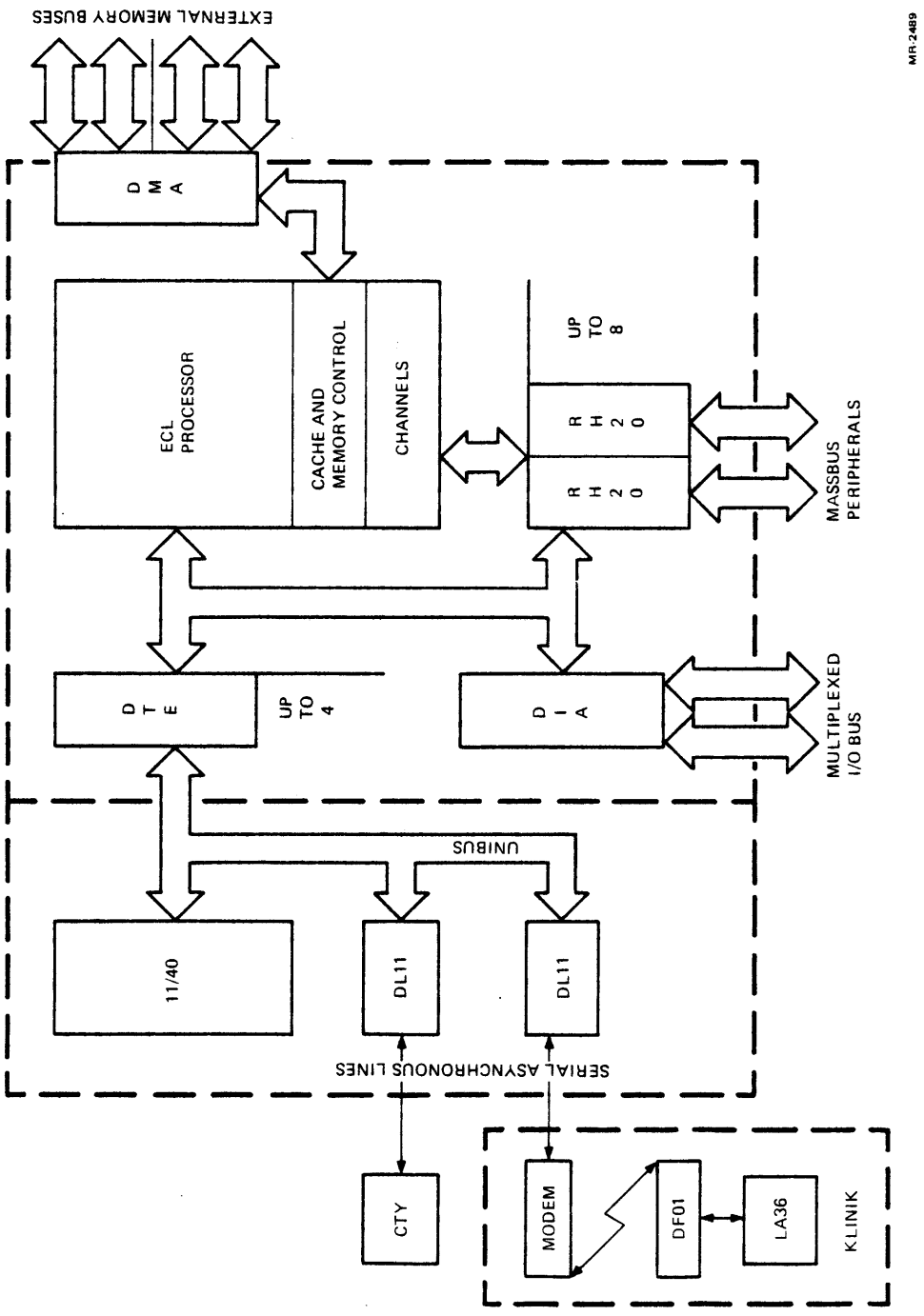


Figure 2-1. Fully Configured KL10-D Central Processing Unit

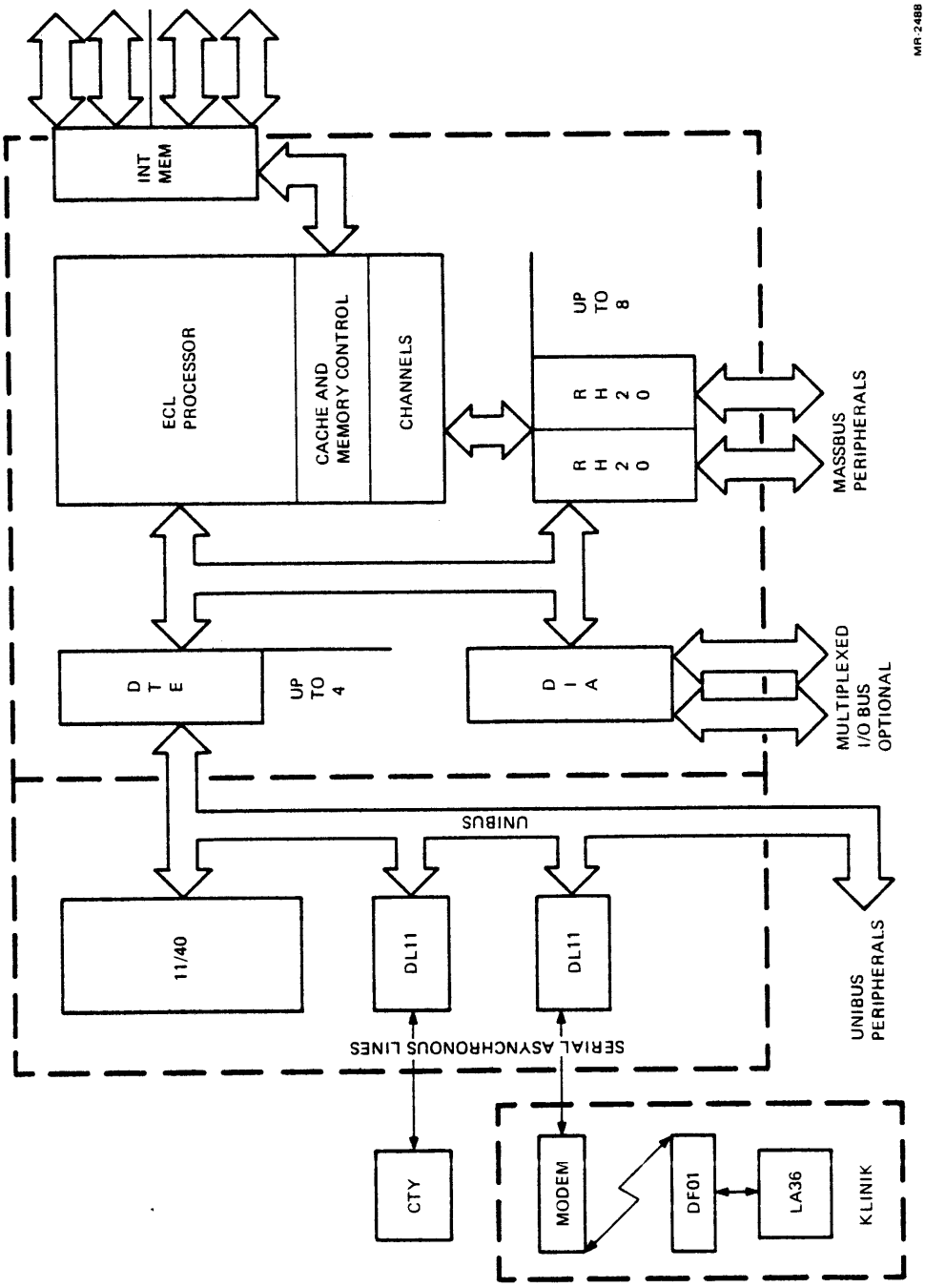


Figure 2-2.
Fully Configured KL10-E Central Processing Unit

The KS10 basic configuration illustrated in Figure 1-1 may be compared with the KL10 configurations in Figures 2-1 and 2-2. In the KS10 configuration, the PDP-11 front end for CTY and KLINIK is replaced with the 8080 microprocessor. There is no M-Box, and cache memory is now part of the KS10 CPU. I/O devices no longer interface via a PDP-11 front end or DIA, but connect with the KS10 bus via controller, UNIBUS and UNIBUS Adaptors. Instead of RH20 controllers, the KS10 has RH11 controllers for disk and magtape. The RH11 controllers allow use of the RM03 disk drive in addition to the RP06 drive. The tape drive model is TU45, standard with the DECSYSTEM-20.

THE KS10 CPU

Overview

Let us now consider the KS10 Central Processing Unit (CPU) in more detail, confining our attention to the box labeled KS10 CPU in Figure 1-2. The KS10 CPU consists of the following four modules (electronic circuit boards):

1. CRA - Control RAM (Random Access Memory) module. This board contains the logic that determines the location from which the next microword will be taken, microcode loading hardware, and 2K x 36 bits of microcode.

2. CRM - Control RAM Memory module. This board contains 2Kx60 bits of microcode.

3. DPE - Data Path Executive module. This board contains the data path, registers, cache, PI system and Dispatch ROM (Read Only Memory).

4. DPM - Data Path Memory module. This board contains the KS10 bus interface, processor status flags, paging hardware, cache directory, and the shift counter.

The KS10 CPU is illustrated in Figure 2-3.

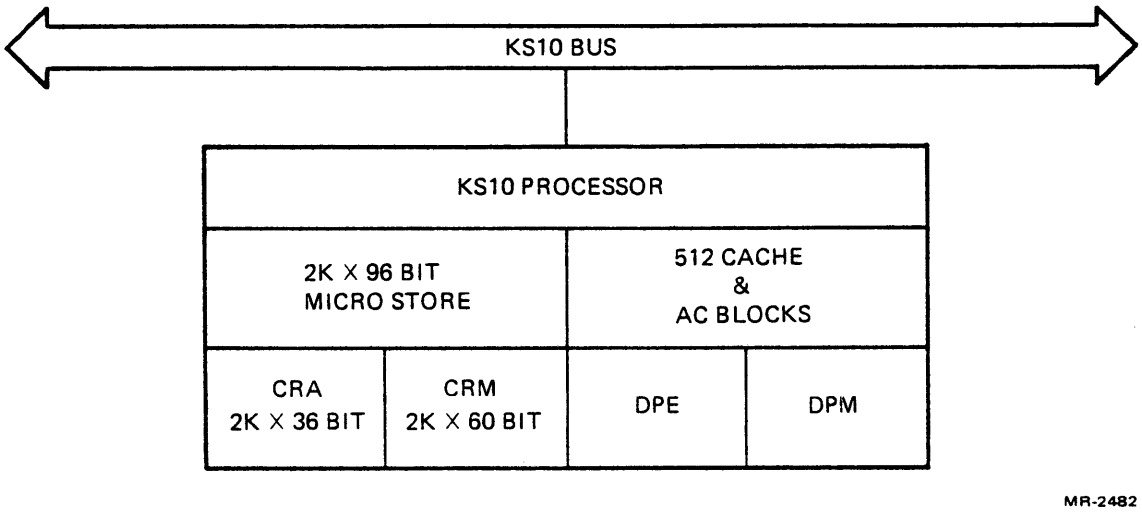


Figure 2-3.
KS10 Central Processing Unit

Schottky TTL Logic

The KS1Ø processor uses standard Schottky TTL logic circuits, which feature relatively low power consumption and moderate speed (as compared with the ECL logic used in the KL1Ø processor).

Microcode

Program execution by the KS1Ø CPU depends on both the processor hardware and the microcode, which is loaded into the CRA and CRM modules at system startup. There is a total of 2K x 96 bits for the microcode, 2K x 6Ø bits being located on the CRM board and 2K x 36 bits on the CRA board.

The microcode handles the execution of individual program instructions and procedures of a more general nature, such as the handling of priority interrupts and page failures. Two quantities associated with the microcode are the microinstruction word itself, and a dispatch word that

supplies data for individual program instruction execution. Note that the same microcode is used for both TOPS-10 and TOPS-20 on the DECSYSTEM-2020. This allows the CPU to execute either TOPS-10 or TOPS-20 paging.

The microcode, as shipped, contains a default CPU serial number that can be changed at installation time, using the SMFILE program (see Chapter 6). The serial number is in location 1700 of CRAM.

Cache Memory

The 512 words of KS10 cache memory occupy the top half of a 1K RAM file on the DPE board. Associated with cache memory is a cache directory, located on the DPM board. The cache directory serves as a pointer to addresses in cache memory. As in the KL10, the KS10 cache memory is used to decrease program execution time, by reading words from main memory into cache where they can be accessed more quickly. Note that in the KL10, cache memory consists of 4K words located in the M-Box. For the KS10, there is no four-way interleaving of main memory, so a memory fetch consists of a single word only.

Another difference between the implementation of cache memory on the KL10 and KS10, is that KL10 cache memory is write-back, while KS10 cache memory is write-through. In the KL10, if the E-Box instructs the M-Box to write a given memory location, the location is modified only in cache. The corresponding physical location is updated later, when the monitor instructs the M-Box to sweep cache, or when a portion of the cache must be emptied to make room for other data. The unloading of a location in cache is carried out in two steps: (1) validation, or writing a modified cache entry to main memory, and (2) invalidation, or simply emptying the cache location by clearing appropriate bits. In the KS10, whenever a word is modified in the cache, it's also written back immediately to main memory.

General Registers

The KS10 has eight sets of 16 word accumulator blocks, which are part of the 1K RAM file on the DPE board. Since the cache occupies 512 words of this RAM file, this leaves 384 spare words in the RAM file that are available as a scratch pad for the microcode. The eight sets of registers can be used as accumulators, index registers and the first sixteen locations in main memory. Access time to the general registers is 300 nanoseconds.

The KS10 Instruction Set

The KS10 instruction set is the same as that for the KL10-D and KL10-E (Section 0 addressing only), with the following exceptions:

1. The single-precision (without rounding), floating point instructions that facilitate software double-precision operations are not supported on the KS10 and trap as illegal instructions. These are:

   a. UFA (Unnormalized Floating Add)

   b. DFN (Double Floating Negate)

   c. FADL (Floating Add Long)

   d. FSBL (Floating Subtract Long)

   e. FMPL (Floating Multiply Long)

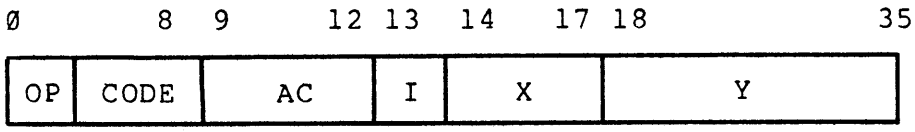
   f. FDVL (Floating Divide Long)

   All the above instructions are considered obsolete.

2. The KS10 checks certain fields in the EXTEND instruction that must be zero (MBZ fields). These fields are not checked by the KL10 processor. For example, the AC field (bits 9-12) in the extended opcode word (location E0) must be zero, and the KS10 checks this. If this field is not zero, an illegal instruction trap results. This allows the

features of these instructions to be expanded without breaking current programs.

3. All KL10 I/O instructions are replaced with a new I/O instruction set for the KS10. The KL10 I/O instruction set uses a word format which differs from the format used for other instructions. For the KS10, the word format for all instructions (including I/O instructions) is the same, as shown below in Figure 2-4. The KL10 I/O instruction format is also shown for comparison.



KS10 Instruction Word Format for all Instructions Including I/O

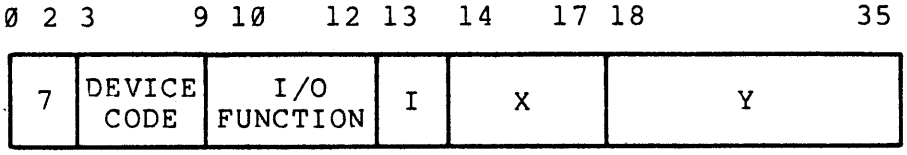


Figure 2-4.
KL10 I/O Instruction Word Format

For KS10 I/O instructions, the opcode is in the 700-777 range, and the AC field can be used to select different instructions or to contain data. The I, X and Y fields have their usual meaning of indirect bit, index register and address (offset), respectively. The KS10 I/O instructions are discussed in more detail in Chapter 3.

Extended Addressing

The KS10 supports only single section addressing (section 0). However, the KL10 model B processor instructions XJRSTF, XJEN, XPCW and SFM are supported. In this respect, the KS10 can be considered the same as a KL10 model B processor with only one section.

Processor Modes

In contrast to the KL10, the KS10 has only two processor modes, user mode and exec mode. In exec mode all implemented instructions are legal. The TOPS-10 monitor operates in exec mode, allowing it to control all system resources and the state of the processor. KS10 exec mode is equivalent to KL10 kernel mode; there is no supervisor mode on the KS10.

In user mode, certain instructions cause a trap to the monitor. User I/O is accomplished through calls to the monitor, because KS10 I/O instructions are illegal in user mode. Whenever the monitor allows a user program to run, the processor is placed in user mode. While in user mode, certain physical events such as priority interrupts and traps place the processor in exec mode.

Interrupt Handling

KS10 priority interrupt handling has the features listed below:

1. Only the JSR or XPCW instruction is allowed as an interrupt instruction, i.e., as the first instruction executed as a result of an interrupt. Any other instruction halts the processor.

2. Non-vectored interrupts (APR, software requested interrupts) trap to EPT + 40 + 2*N, where EPT is the Executive Process Table (the UPT and EPT for the KS10 are similar to the UPT and EPT for the KL10 - see Figure 2-5).

3. For devices, the only implemented interrupt function is the vectored interrupt. The KS10 references an EPT location determined by the UBA number (EPT + 100 + UBA #). It then uses this word as an executive virtual address of a table and executes the instruction at TABLE + VECTOR/4.

4. The KS10 implements two levels of PIA (Priority Interrupt Channel Number) for I/O (UNIBUS) devices; one PIA can have a higher priority than the other. The PI level (1-7) assigned to UNIBUS devices interrupting on BR levels 7 and 6 (7 is the highest PDP-11 bus request level) is set by loading a high level PIA (bits 30-32 of UBA status register). The PI level (1-7) for devices interrupting on BR levels 5 and 4 is set by loading a low level PIA (bits 33-35 of UBA status register).

Table 2-1 lists the hard wired interrupt vectors and BR (Bus Request) levels for the various KS10 I/O (UNIBUS) devices in a fully configured system. It also indicates which PIA (high or low level) is associated with each device.

Table 2-1. I/O (UNIBUS) Device Vectors and BR Levels

Device	UBA #	PIA	Interrupt Vector	BR
RH11 #1(RP06)	1	HI	254	6
RH11 #3(TU45)	3	HI	224	6
LP20 #1	3	LO	750	4
DZ11 #1	3	LO	340	5
DZ11 #2	3	LO	350	5
DZ11 #3	3	LO	360	5
DZ11 #4	3	LO	370	5
KMC11 #1	3	LO	540	5
DUP11 #1	3	LO	570	5
DUP11 #2	3	LO	600	5

Paging Differences

The KS10 microcode is able to handle both TOPS-10 and TOPS-20 paging. To select TOPS-10 paging, the I/O instruction WREBR (opcode 701) is used with bit 21 set to zero.

Following all page failures (other than access not allowed), the processor causes a page-fail trap and stores a page-fail word in location 500 (octal) of the UPT. Bits 2-5 of this word contain a page-fail code if bit 1 = 1. The page-fail codes for the KS10 are:

20 An I/O instruction selected a device or register that does not exist. The address in the page fail word (bits 14-35) is the I/O address that caused the error.

If the address causing the page fail was an I/O address, bit 10 is set in the page-fail word. If the address causing the page fail is for an I/O byte, bit 13 is set in the page-fail word.

25 Page Table Parity Error.

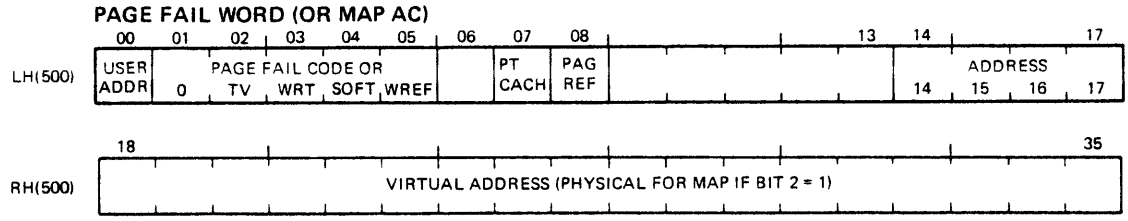
36 Hard memory error - ECC (Error Correction Code) was not able to correct the data.

37 NXM (Non-existent Memory).

The complete bit format for the page-fail word in shown in Figure 2-5.

KS10 UPT/EPT

User Process Table and Executive Process Table configurations for the KS10 are shown in Figure 2-6.



BIT(S)	FUNCTION
0	USER ADDRESS
2-5 (BIT 1 = 0)	
2	TRANSLATION VALID
3	WRITABLE (KL PAGING MODE = 0) WRITTEN (KL PAGING MODE = 1)
4	SOFTWARE (KL PAGING MODE = 0) WRITABLE (KL PAGING MODE = 1)
5	WRITE REFERENCE
2-5 (BIT 1 = 1)	PAGE FAIL CODE
	20 AN I/O INSTRUCTION SELECTED A NONEXISTENT DEVICE OR REGISTER. (BITS 14-35 = I/O ADDRESS)
	25 PAGE TABLE PARITY ERROR
	36 HARD MEMORY ERROR
	37 NXM
7	PAGE TABLE CACHE
8	PAGED REFERENCE
18-35	VIRTUAL ADDRESS (PHYSICAL FOR MAP IF BIT 2 = 1)

MR-0259

Figure 2-5.
Page Fail Word

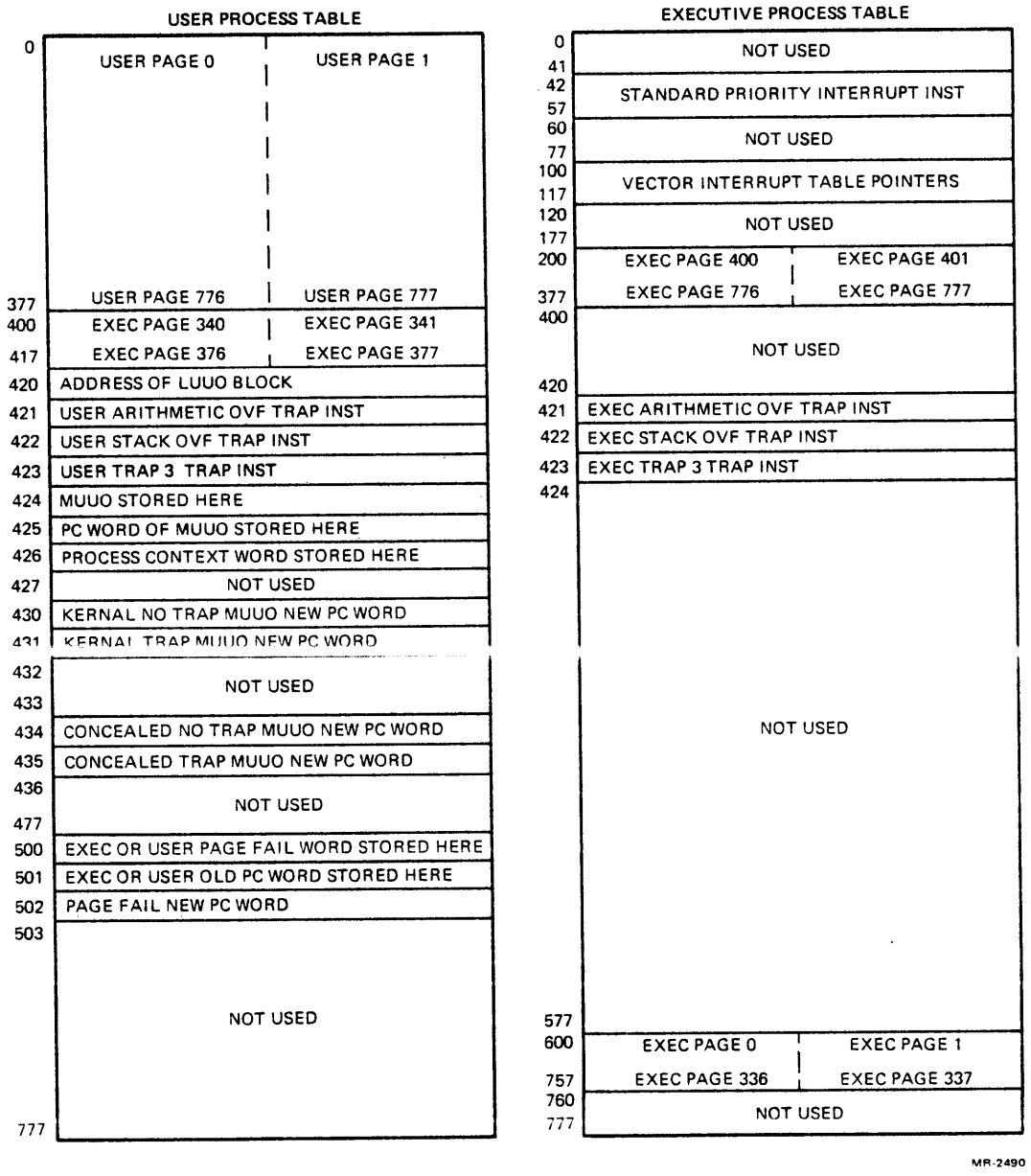


Figure 2-6.
KS10 EPT/UPT (TOPS-10 Paging)

KS10 Error Recovery

When the processor halts, it stores a halt status code in bits 24-35 of physical memory location zero (not AC 0) and the PC in physical memory location one (not AC 1). Codes in the range 0-77 indicate "normal" halts, codes in the range 100-777 indicate software failures, and codes of 1000 or greater indicate microcode or software failures. Halt status codes are:

0 Microcode just started.

1 HALT instruction executed.

2 Console program halted CPU.

100 I/O page failure. While starting an interrupt, there was a page fault. Either a memory error (bad data or NXM), or pager trap (page not in core, not writable, etc.) occurred while executing the interrupt instruction.

101 Illegal interrupt instruction. Interrupt instruction must be either XPCW or JSR.

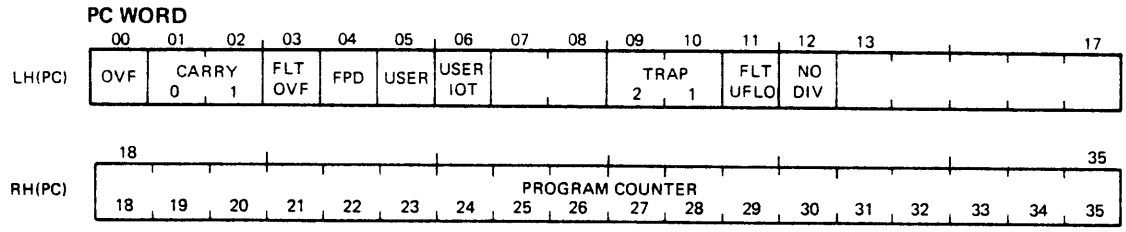
102 Pointer to UNIBUS vector is zero.

1000 Illegal microcode dispatch. The dispatch ROM contains an unused "B-write" dispatch code.

1005 Startup check (run by microcode on every startup) failed.

The PC word is saved in physical memory location one. Several instructions (e.g., JSR) also save the PC and various processor flags in a memory location or an AC. The bit format for the PC word is shown in Figure 2-7.

When a halt occurs, the microprocessor prints the halt status code and the right half of the PC word on the CTY (see Chapter 7).



BIT(S)	FUNCTION
0	OVERFLOW
1	CARRY 0
2	CARRY 1
3	FLOATING OVERFLOW
4	FIRST PART DONE
5	USER MODE
6	USER IOT (ALSO PCU)
9	TRAP 2
10	TRAP 1
11	FLOATING UNDERFLOW
12	NO DIVIDE
18-35	PROGRAM COUNTER

MR-0258

Figure 2-7.
PC Word

In addition to saving the halt status code and PC word in memory, the processor optionally stores a halt status block of 18 words at a location selected by the programmer. This block consists of the 16 internal CPU registers and the Virtual Memory Address (VMA) with flags. The WRHSB instruction allows the program to load a value for the halt status block address.

If the halt status block address is positive, a processor halt has the halt status block stored in KS10 memory starting at the specified address. If the halt status block address is negative, the halt status block is not stored. Initially, when the microcode is started, the halt status block address is set to a value of (+) 376000 and the halt status block is stored. However, the TOPS-10 monitor changes the address at which the block is stored to location 424 of the EPT.

Figure 2-8 shows the information stored in each location of the halt status block. These locations can be examined using the console microprocessor command language (see Chapter 7).

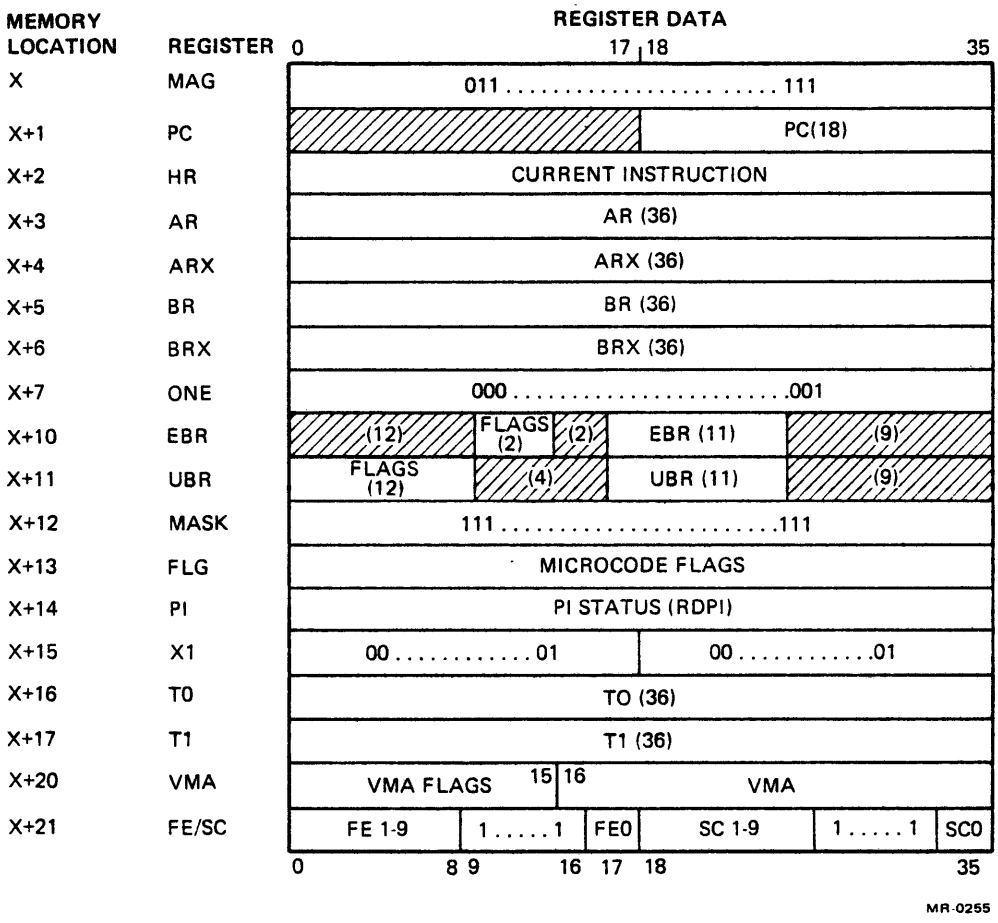


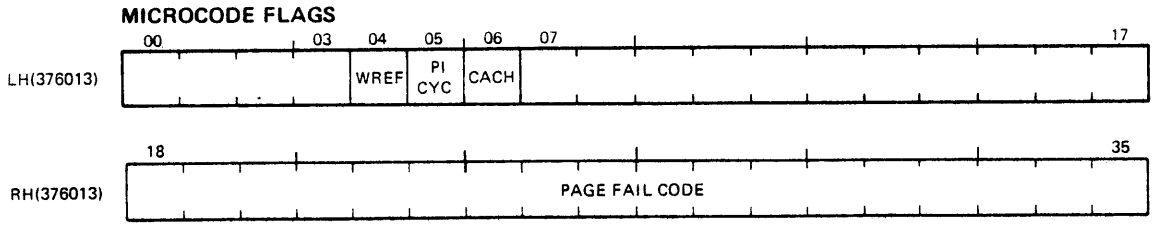
Figure 2-8.
Halt Status Block

The first 16 memory locations of the halt status block hold the register data that was read from the 16-word RAMs associated with the microprocessor circuits. Status information includes the PC (also stored in physical memory location one), the current instruction, Exec Base Register (EBR), User Base Register (UBR), microcode flags and PI

system status. (PI system status is the same as that read by the RDPI instruction). Another processor status word, the Virtual Memory Address (VMA) plus flags, is stored in the next to last location of the halt status block. Bit definitions for the microcode flag and VMA words are given below.

Microcode Flags — In the event of a page failure, three flags and a page-fail code are stored as part of the halt status block in offset 13 (X + 13). The page-fail code, which is the contents of the microword's magic number field, gives the operation associated with the failure. Status word bit format and page-fail code definitions are given in Figure 2-9.

VMA — The Virtual Memory Address (VMA) and VMA flags are stored in location X + 20 of the halt status block. Bit format and definitions are given in Figure 2-10.

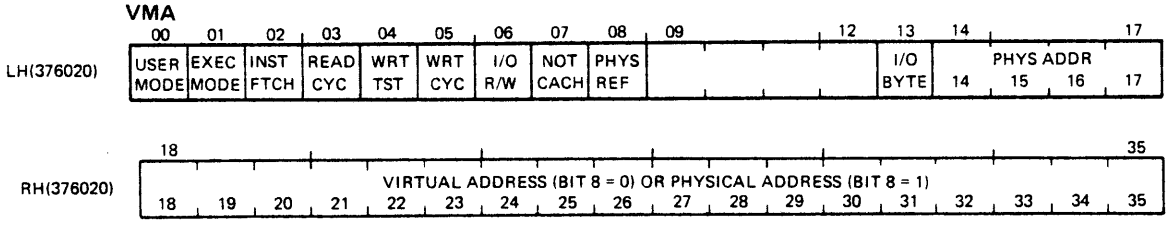


BIT(S)	FUNCTION
4	WRITE REFERENCE BIT FROM PAGE MAP
5	PI CYCLE
6	LOOK IN CACHE BIT FROM PAGE MAP
18-35	PAGE FAIL CODE

000000	SIMPLE INSTRUCTIONS
000001	BLT IN PROGRESS
400002	MAP IN PROGRESS
000003	MOVE STRING SOURCE IN PROGRESS
000004	MOVE STRING FILL IN PROGRESS
000005	MOVE STRING DESTINATION IN PROGRESS
000006	FILLING DESTINATION
000007	EDIT SOURCE
000010	EDIT DESTINATION
000011	CONVERTING DECIMAL TO BINARY
000012	COMPARING DESTINATION

MR-0256

Figure 2-9.
Microcode Flags



BIT(S)	FUNCTION
0	USER MODE
1	EXEC MODE
2	INSTRUCTION FETCH
3	READ CYCLE
4	WRITE TEST
5	WRITE CYCLE
6	I/O READ OR WRITE
7	DO NOT LOOK IN CACHE
8	PHYSICAL REFERENCE
13	I/O BYTE INSTRUCTION
14-17	BITS 14-17 OF PHYSICAL ADDRESS (OR 0s)
18-35	BITS 18-35 OF VIRTUAL ADDRESS (BIT 8 = 0) OR PHYSICAL ADDRESS (BIT 8 = 1)

MR-0257

Figure 2-10.
VMA

KS/KL MAJOR DIFFERENCES

KS1Ø	KL1Ø
Cheaper, smaller in size than KL	Larger, more expensive than KS
Uses low power SCHOTTKY TTL logic	Uses high power ECL logic
Up to 512K internal MOS memory	Up to 4196K internal and external memory
8Ø8Ø microprocessor used for console functions, invisible to monitor	PDP-11 based front end used for console functions
All I/O done via UNIBUS Adaptors and UNIBUSes	All I/O done via PDP-11 console front end and DTE, or via a DIA
Single port MOS memory, only one-way interleaving possible	Up to four-way interleaving possible for memory
512-word cache memory in CPU	4K word cache memory in M-Box
36-bit KS1Ø backplane bus connecting major components	Connections via various busses, UNIBUS, SBUS, CBUS DMA, DIA, DTE
Two processor modes	Four processor modes
396 instructions	398 instructions
Uses RMØ3, RPØ6 disk drives	Uses RPØ4, RPØ6 disk drives
Checks parity on data paths, on KS1Ø bus, in micro-store	Checks E-Bus parity for some devices

Summary of KS10/KL10 Differences Continued

KS10	KL10
Address break feature not implemented in hardware	Address break feature implemented in hardware
No multiprocessing	Multiprocessing available
No meter board, perf. meters	Meter board, perf. meters
No elaborate accounting	E-Box/M-Box accounting

KS10 RELATED HARDWARE

KS10 Backplane

The KS10 backplane bus (see Figure 1-2), serves as a data and control path between the KS10 CPU, microprocessor, MOS memory and the two UNIBUS Adaptors that interface with the peripheral devices. The bus can transfer 36-bit data words, and has additional lines for parity checking and control. Each device checks for correct parity (even) when it receives information over the bus; if bad parity is detected, the CPU clock is stopped.

In order to transfer information over the bus, a device must first gain control by requesting and being granted the bus. For each device, there is a request line and a grant line. The bus arbitrator (located on the console module) monitors requests, resolves priority and (when the bus is free) grants the bus to the highest priority device, by asserting the device grant line.

Console Subsystem

The console subsystem handles user and maintenance mode commands given at a local operator console or CTY. There is also a remote line for diagnosis (KLINIK line). The CTY is an LA36 terminal.

Figure 2-11 illustrates the components of the console subsystem. The subsystem is housed on a single circuit board, the Console (CSL) board, and consists of:

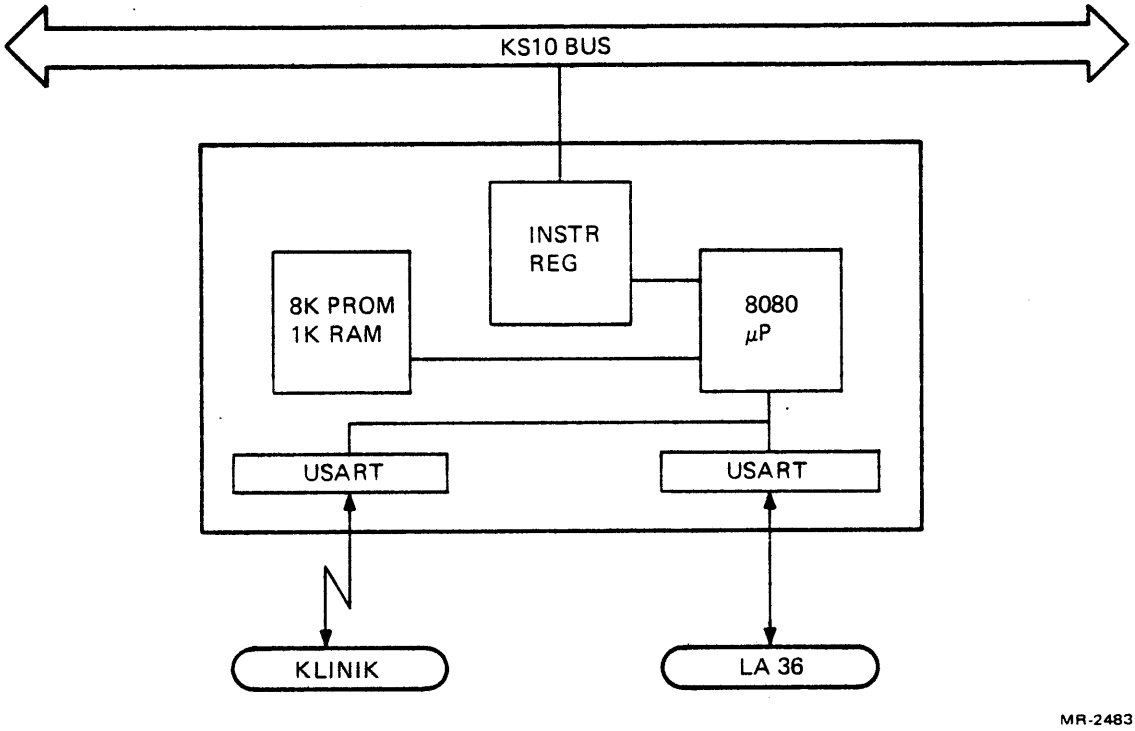


Figure 2-11.
Console Subsystem

1. An 8080 microprocessor. This is an 8-bit central processing unit fabricated on a single LSI chip.

2. An 8K PROM (Programmable-Read-Only Memory) and a 1K RAM (Random Access Memory).

3. Two USARTs (Universal Synchronous/Asynchronous Receiver/Transmitter), one for console operations and one for the remote diagnostic link.

4. The 8080 Instruction Register.

The 8080 microprocessor executes a program (resident in the PROM) which starts automatically when the system is powered ON. This program, or console program, is not destroyed by powering down the system. The console program implements commands typed at the CTY or remote diagnosis terminal. Two USARTs effect the interface between the CTY/diagnostic link and the microprocessor. A USART allows 8-bit characters to be passed between a communications line and the microprocessor. The console board also contains the main clock source for the KS10 processor, which arbitrates access to the backplane bus and the 8080 based console hardware.

The console subsystem uses the console program to load the KS10 microcode from disk at system startup. It also boots the system by loading a bootstrap program into main memory. The console program automatically attempts to boot the system every 30 seconds. The system can also be booted by commands given at the CTY.

Besides loading the KS10 microcode, the console program can also be used to verify the microcode, to read and write main memory and to stop, start and single step the clock. The KL10 console's full capability is not available to the operator, since the KS10 console cannot read the PI status, and the VMA, PC or AC registers. The reason for this is that many of the registers are internal to the DPE board.

For diagnostic purposes, the PC can be read as follows: when a halt instruction is executed, or the KS10 CPU is requested to stop by the 8080 microprocessor, the microcode deposits the PC and halt status block in main memory. These values in memory can then be read via the CTY. The contents of the AC's at the time of a crash can be obtained from the crash dump file, CRASH.EXE. The console command language used for diagnosis and control of the DECSYSTEM-2020 is discussed in Chapter 7.

Memory Subsystem

The memory subsystem consists of memory controller and up to 512K of MOS (Metal-Oxide-Silicon) memory. It interfaces directly with the KS1Ø bus, as shown in Figure 2-12.

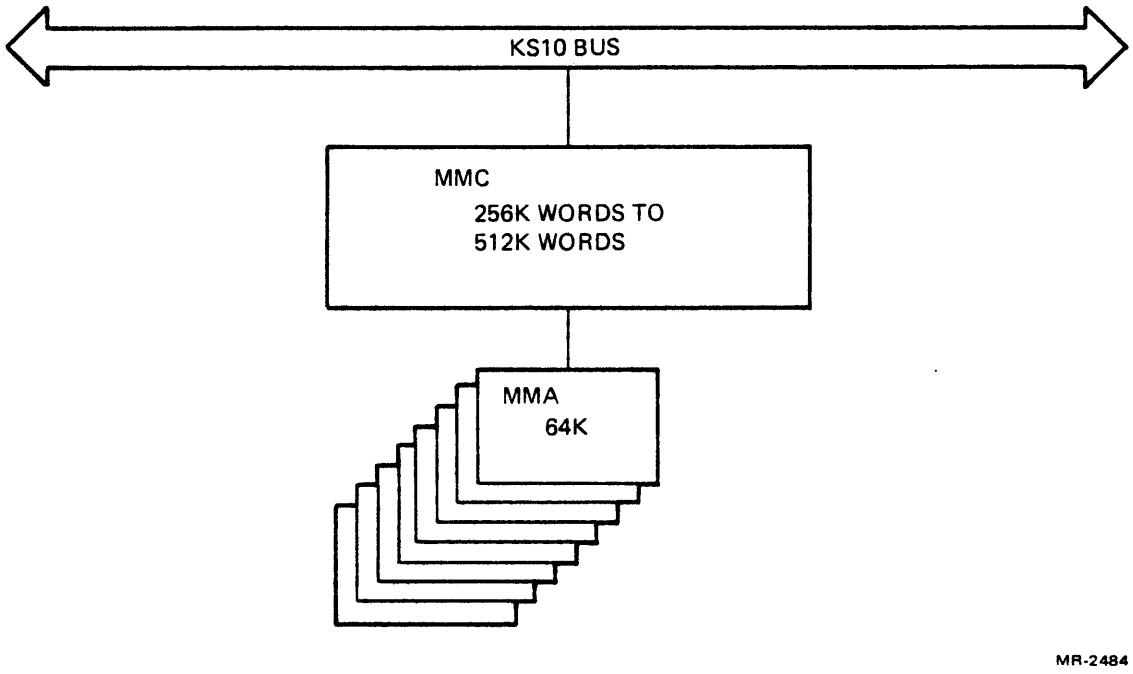


Figure 2-12.
Memory Subsystem

The memory controller (MMC board) connects to the backplane bus and drives up to eight memory array boards of 64K words each. Only one memory controller can be connected to the KS1Ø bus; it responds to memory addresses below 2ØØØØØ octal.

MOS memory is said to be "volatile"; this means that a loss of power results in a loss of data, i.e., in the loss of memory contents. MOS memory is refreshed (data is re-established) every 2 msec. The refresh cycle is controlled by the memory controller, MMC.

MOS memory is composed of silicon chips containing thousands of semiconductors that have two states: conducting a current and not conducting a current. Semiconductor memory offers increased speed, reduced size and a truly nondestructive read. Unlike core memory, there is no need to write the contents of memory back after it has been read. Hence, there is no need for memory interleaving to speed up memory access by allowing memory reads in one memory module during write-back in another module.

MOS memory is built in 16K chips, with error correction logic (ECC) that allows hardware to correct all single-bit errors, and to detect all multiple bit errors. A failing module is expected to be replaced when hard errors occur, while modules with correctable errors need not be replaced. Memory that experiences hard errors can be set off-line, using the TOPS-10 SET MEMORY OFF-LINE command.

The memory cycle time is 1.05 microseconds. There is a single port into memory, so only one word can be read or written at a time.

UNIBUS Adaptor and UNIBUS

The UNIBUS Adaptor (UBA) is a KS10 I/O controller that allows UNIBUS peripheral devices to be connected to the KS10 system. It connects between the KS10 bus and a UNIBUS, as shown in Figure 2-13. Up to three UBAs may be installed in the KS10, although two UBAs are standard in the 2020 end-user configuration. One UBA (and UNIBUS) is reserved for disk. Another UBA (and UNIBUS) is used for all other devices (i.e., for magtape, line printer, card reader, and synchronous and asynchronous communications lines). Each UBA consists of a single module mounted in the KS10 cabinet.

The UNIBUSes in the KS10 are the same as those in PDP-11 systems. Since the data lines on a PDP-11 bus are 16 bits wide, it was necessary to convert two parity indicator lines to data lines, so that 18 bits are available for data transfers. Thus, a half word at a time can be transferred on the UNIBUS. There are in fact several transfer modes, which are discussed in Chapter 3.

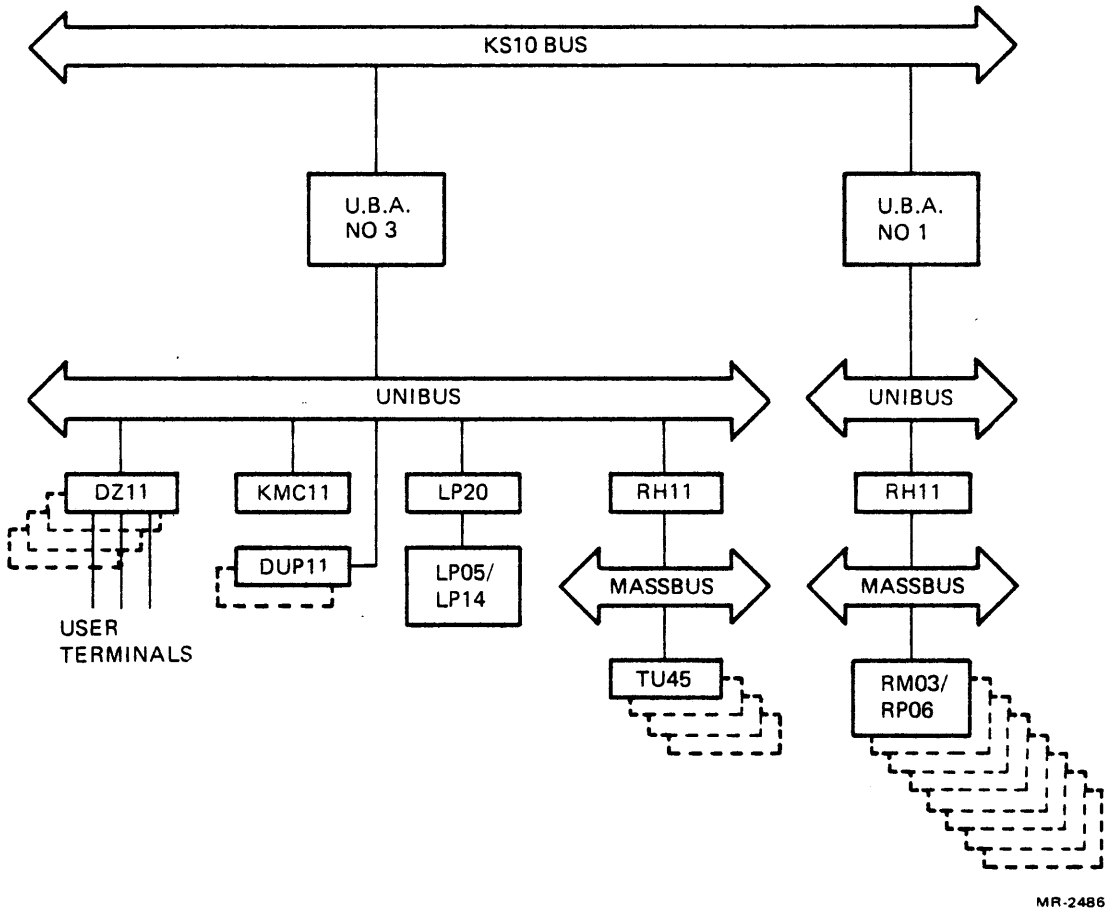


Figure 2-13.
UNIBUS Adaptors and UNIBUSes

RH11 Controller

The RH11 controller is able to control, via a MASSBUS, a number of newer, high-speed mass storage devices such as the RMØ3 and RPØ6 disk systems, and TU45 tape drives. The RH11 controller serves as an interface between UNIBUS and MASSBUS, where MASSBUS is the name of the I/O bus for the above-mentioned devices.

The RH11 does not accept a channel command list. A disk transfer must be contiguous both on the disk and in the UNIBUS address space (there are no scatter reads and writes). This means that TOPS-1Ø buffered I/O can only operate at the rate of one block per transfer, which is equivalent to a transfer rate of one block per disk revolution (6Ø blocks/second) or approximately one-twentieth the rated speed of an RPØ6. This fact may degrade considerably the performance of jobs that do a significant amount of sequential I/O. However, nonbuffered mode I/O, (i.e., dump mode) exercises the disk to full extent.

Disks are not supported on the same UNIBUS with other devices, because of possible overruns on the UNIBUS from simultaneous transfers on different devices.

RM03/RP06 Disk Drives

The RMØ3 disk drive is a PDP-11 type device that may be used with a KS1Ø system by virtue of the RH11 controller. The RPØ6 disk drive is a standard disk drive for DECsystem-1Ø computers. The table below compares some of the characteristics of the RMØ3 and RPØ6 disk drives/packs.

Table 2-2. RM03/RP06 Comparison

Characteristic	RM03	RP06
Average Access Time	38.3 ms	36.3 ms
Average Seek Time	30 ms	28 ms
Formatted Capacity	122,250 blocks	309,700 blocks
Words per Sector	128	128
Sectors per Track	30	20
Tracks per Surface	815	815
Recording Surfaces	5	19
Max. Data Transfer Rate	250K words/sec	166K words/sec

Note that RM03 and RP06 drives may be mixed on the same RH11 controller, but may not be mixed within the same logical disk structure. Up to eight drives are allowed. Both types of drive transfer 18 bits of data over the UNIBUS and 36 bits of data over the backplane bus. In these Non-Processor Request (NPR) transfers, the KS10 initiates the transfer via the UNIBUS adaptor; then, the transfer proceeds, using the same control and data paths used by the KS10 processor - but not under control of the processor.

TU45 Tape Drives

The TU45 tape drive is the standard drive on the DECSYSTEM-20 running TOPS-20 or TOPS-10. It is also used on the DECsystem-1091, which has the same hardware as a DECSYSTEM-2050. The TU45 tape drive interfaces with the KS10 via a TM02 or TM03 tape controller, which can control from one to four drives. Early machines will ship with the TM02 controller. All tape functions currently supported for TOPS-20 on the 2020 will also be supported for TOPS-10.

The TU45 tape drive must be loaded manually. It has the following characteristics:

1. Tape speed of 78 inches/sec.
2. Density of 800/1600 bpi.
3. 9-Track format.
4. Maximum data transfer rate of 60K 18-bit words/second.
5. Use of 2400 ft., 1/2 inch industry standard reels.
6. Data transfers between magtape and other devices are 18-bit transfers over the UNIBUS and 36-bit transfers over the backplane bus. In these Non-Processor Request (NPR) transfers, the KS10 initiates the transfer via the UNIBUS Adaptor; then the transfer proceeds using the same control and data paths used by the KS10 processor - but not under control of the processor. There may be up to four tape drives on the system. Tape transfers longer than 32K bytes are not possible, because they require too much UBA address space.

CR10E/F Card Readers

The card reader models available for use with TOPS-10 on the 2020 are the CR10E (floor model, fast) and the CR10F (table top model, slow). These are the same model card readers currently available for use with a KL10 running TOPS-10.

The controller for the card reader is a CD20 that interfaces with a UNIBUS. The card reader performs the same functions as a card reader on the KL10. Speeds of the card readers are 1200 cards per minute (CR10E) and 300 cards per minute (CR10F). Card capacities are 2200 (CR10E) and 550 (CR10F) cards in the hopper or stacker. There may be a maximum of one card reader on the system.

LP05/LP14 Line Printers

The printer models for TOPS-10 on the 2020 are the small model LP05 and LP14. These printers are currently available on other DECsystem-10s and also on the DECSYSTEM-20 as models LP20A (upper case), LP20B (upper/lower case) and LP14. All models are similar in appearance and controls, but the LP14 is capable of greater speeds. All models have a programmable Vertical Format Unit (VFU) and both LP05 and LP14 exist as upper case only, and upper/lower case models. The following table gives the specific characteristics of the LP05 and LP14 models.

Table 2-3. Line Printer Characteristics

Characteristic	LP05	LP14
Printable Characters	64 or 96	64 or 96
Characters per line	132	132
Print Rate - 64 char. drum - 96 char. drum	300 LPM 240 LPM	900 LPM 600 LPM
Paper Slew Speed - 6 LPI - 8 LPI	20 IPS	30 IPS 22.5 IPS
Carriage Control (VFU)	Programmable	Programmable

LPM = lines per minute IPS = inches per second

The DECSYSTEM-2020 also supports the LP07 line printer. The line printer (a maximum of one is allowed) connects with the UNIBUS via an LP20 controller, which includes a programmable translation RAM for translating between ASCII and printable characters. However, only the standard translation RAM is supported.

Asynchronous Communications

The DZ11 8-line controller is used to interface asynchronous lines with the UNIBUS. There can be up to four DZ11 controllers; hence, a system can have from 8 to 32 lines. A wide range of baud rates is allowed (from 50 up to 9600 baud), but split speeds (different rates of receive and transmit) are not possible due to the DZ11 hardware. Character lengths range from 5 to 8 bits, with odd or even parity and 1, 1.5 or 2 stop bits. Terminals operate in full duplex mode. Data transfers over the UNIBUS and KS10 bus are 8 bit (register I/O) transfers. For remote terminals, there is carrier, ring, date, terminal ready and break MODEM control.

Synchronous Communications

The synchronous communications interface has the following features:

1. DUP11 single-line controller (one per line).
2. Bit rate of 2000-19200 BPS.
3. KMC11 NPR microprocessor (one per 1-2 lines).
4. Two lines/system maximum.
5. 8- or 16- bit (NPR) data transfers over UNIBUS or the KS10 backplane bus.

The DUP11 is a BR (Bus Request) device; if used alone, it interrupts the KS10 for each character. The KMC11 is a UNIBUS compatible general purpose microprocessor, which is loaded at system startup by the monitor. It can be used to reduce the load on the KS10, by putting characters from the DUP11 into a silo in main memory and outputting characters from "chunks" in main memory to the DUP11. In this way, the DUP11 is made to appear as an NPR device. Alternatively, the KMC11 can be used to implement part of the line protocol.

Physical Description

The DECSYSTEM-2020 hardware (apart from the peripheral devices themselves) is housed in a single cabinet of dimensions 60 inches x 27 inches x 30 inches, as shown in Figure 1-1. Specifically, this cabinet contains:

1. The KS10 CPU and KS10 Bus.
2. 192K to 512K MOS memory.
3. The console subsystem, including the 8080 microprocessor, PROM and two USARTS.
4. Two UNIBUS Adaptors and two UNIBUSes.
5. Two RH11 controllers, one for disk and one for tape.
6. DZ11 asynchronous communications controllers (one to four) and KMC11/DUP11 synchronous communication interface (two DUP11's maximum).
7. LP20 and CD20 controllers for a line printer and card reader respectively.

At the top right front of the cabinet is the operator switch panel. The switches on this panel are used to turn on power, boot the system, reset the system and lock the other switches so the system cannot be crashed accidentally. There are also several indicator lights and a three position, key-operated switch for the remote diagnosis link. The operator switch panel is discussed in detail in Chapter 7.

Unsupported Hardware

TOPS-10 on the DECSYSTEM-2020 does not support DECtapes, card punches or paper tape reader/punches. The following is a list of features (supported on the KL10) that are not supported on the KS10, since they are not implemented in the hardware.

Features Not Supported On KS10 Hardware

1. Address break.
2. Public mode. This implies no proprietary execute only segments that share an address space with a user provided segment.
3. Existing programs that use USER IOT privileges to access external devices. This implies a likelihood of breaking existing real time programs. (The IO BUS interface is replaced by a UNIBUS interface. There are different instructions to control the CPU).
4. Real time (while real time is essentially customer provided, there is no testing to see that real time devices work on the UNIBUS).
5. Multiprocessors - single-port memory only.
6. User accounting that excludes interrupt overhead.
7. E-Box/M-Box accounting as it exists on the KL10.
8. Performance meter functions as on the KL10.
9. Power fail/auto restart as on the KL10. TOPS-10/KS10 executes an auto reload after power fail, since that is the best recovery possible without battery backup for MOS memory. However, this feature is supported if a power supply with battery backup is purchased.
10. Tapes and disks on the same UNIBUS, since UNIBUS bandwidth can be exceeded.

11. DECtape.

12. Dual-ported disks.

13. Split speed terminals. The hardware (DZ11) does not support this feature (the input and output speed of a terminal must be the same).

14. KA style double precision, floating point instructions.

15. Tape transfers longer than 32K bytes, since too much UNIBUS Adaptor address space is required.

16. Dial-out (the ability to dial a phone number).

Chapter 3

KS10 I/O INSTRUCTIONS

INTRODUCTION TO KS10 I/O

The KS10 I/O instructions are valid only in executive mode; they perform input/output operations for internal (PI, clocks, etc.) and external (disk, printer, etc.) devices. The KS10 I/O instruction word format is different from the KL10 I/O instruction word format; the KS10 I/O format is the same as the normal instruction word format, shown in Chapter 2 (Figure 2-4).

The KS10 I/O instructions for external devices differ from the KL10 I/O instructions in mnemonic and action. Since the DECSYSTEM-2020 consists of a KS10 processor that uses 36-bit words, and PDP-11 peripherals that address 8-bit bytes (PDP-11 word size is 16 bits), certain incompatibilities must be resolved. Some are resolved by means of the UNIBUS Adaptors (UBAs), others are resolved individually for each device.

As an example, let us look briefly at the WRIO instruction (a more detailed treatment is given in later sections of this chapter). The WRIO instruction is a KS10 instruction used to set up the contents of device registers and to initiate data transfers between devices.

The instruction format is:

```
WRIO AC, ADDRESS
```

The address is interpreted as a device register address, and data in the AC is deposited in the device register. A bus timeout occurs if an attempt is made to address a location without an associated device. The address must not only specify an 18-bit device register address, but also the number of the UBA (at present 1 or 3) to which the device is connected via a UNIBUS. Therefore, an 18-bit address is not sufficient.

Up to 20 bits are needed for the effective address, (two bits to specify the UBA number and 18 for the device register). This is accomplished by using indirect or indexed addressing. For example, to specify device register 776734 on UBA1 as the address in a WRIO instruction, a memory location containing 1,,776734 is set up; this location is then specified using indirection. Only one level of indirection is allowed:

```
WRIO AC, @LOC

LOC/ 1,,776734
```

Another way to do this is using an index register:

```
WRIO AC, (AC3)

AC3/ 1,,776734
```

Device registers are set up in this manner with the WRIO instruction. Most devices have registers that get the word or byte count, the address to transfer to, the address to transfer from and a status register that specifies (among other things) the direction of the transfer. When the status register is set up, the I/O is started.

The KS10 I/O instructions use new mnemonics for internal devices, but the instructions are otherwise the same as standard KL10 internal I/O instructions. For example, the instructions:

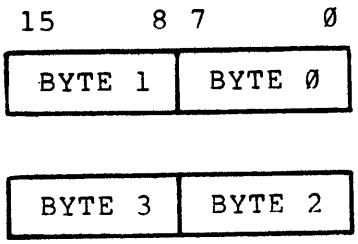
```
CONO PI, PIOFF          ; KL10
WRPI PIOFF              ; KS10
```

both decode to the same instruction and perform equivalent functions. Note that the new mnemonic does not use an AC field for the internal I/O instruction. The KS10 I/O instructions are discussed individually later in this chapter.

UBA MAPPING

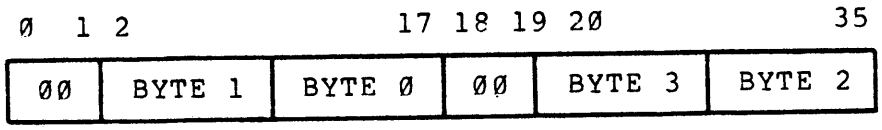
KS10/PDP-11 Address Differences

The KS1Ø addresses 36-bit words, the PDP-11 addresses 16-bit words and PDP-11 peripherals address 8-bit bytes. In a KS1Ø word, the bits are numbered Ø through 35 left to right. In a PDP-11 word, bits are numbered Ø through 15 right to left, and a word consists of two bytes (numbered byte Ø and byte 1 right to left). For several consecutive PDP-11 words, the bytes are numbered as shown below:



PDP-11 Word

The PDP-11 thus stores bytes in reverse order from the KS1Ø. Also, a KS1Ø word can accommodate four PDP-11 8-bit bytes. Devices such as the line printer, which transfers data one byte at a time, must shuffle bytes in a 36-bit KS1Ø word prior to the data transfer. The line printer, for example, shuffles bytes to the following format:



Bytes in the left half word are transferred first in the order indicated (BYTE Ø, then BYTE 1).

PDP-11 peripherals can request PDP-11 addresses between Ø and 377777 as memory locations. Since a KS1Ø word can contain four PDP-11 bytes, one KS1Ø page (512 words) can contain 4 x 512 = 2Ø48 decimal or 4ØØØ octal PDP-11 bytes. Hence, one KS1Ø page can contain 4ØØØ octal PDP-11 addresses.

On the PDP-11, all addresses (including memory) are called UNIBUS addresses. Those addresses between 760000-777777 are reserved for use as device registers. Any reference to an address in this range gets a bus time-out if no device is present. If the device is present, it responds by depositing data into its register, or by returning data from its register.

Data Transfer Modes

The UNIBUS Adaptor (UBA) performs an important part of the necessary address translation between KS10 I/O instructions and the PDP-11 peripherals on the system. On the standard system there are two UBAs, numbers 1 and 3 (UBA 2 is not currently used). A UNIBUS Adaptor makes use of two extra lines on the UNIBUS for data transfer; therefore, 18 bits of data can pass simultaneously over the UNIBUS.

A UNIBUS Adaptor contains the necessary elements to provide byte, half word or full word (36-bit) transfers to the KS10 and main memory, and to provide 18 or 16-bit transfers to the UNIBUS. All 16-bit transfers are enabled/disabled by the setting of a flag bit in the UBA; the device type determines the setting. Specifically, the UBA contains:

1. A buffer where it can assemble a full 36-bit word from two 18-bit half words.

2. A 64 x 16-bit paging RAM (otherwise known as 64 mapping registers), used to translate a PDP-11 address to a corresponding KS10 physical memory location.

In order to transfer 36-bit words between the KS10 and peripherals, 18-bit transfers on the UNIBUS are enabled. Then, a 36-bit word transfer can be accomplished in either one of two sub-modes: read-pause-write, or 36-bit mode.

The 36-bit transfer mode requires one memory cycle (rather than two) for each word transferred. In 36-bit mode, the PDP-11 word count must be even (an odd word count would hang the UBA). Only one device on each UBA can do 36-bit transfers; this is because the UBA has only one buffer to hold the left 18 bits, while the right 18 bits come across the UNIBUS. On the 2020 system, the disk is the only device using the 36-bit transfer mode. Listed below is an explanation of each mode for a read and a write operation.

1. 36-bit transfer on a write operation:

   The UBA picks up the 36-bit word, sends the left 18 bits and holds the right 18 bits in a register, until the device requests the next 18 bits.

2. Read-pause-write transfer on a write operation:

   The UBA picks up the left 18 bits of the word from KS10 memory and sends them to the device. Then it picks up the right 18 bits of the word from KS10 memory and sends them to the device.

3. 36-bit transfer on a read operation:

   The UBA transfers 18 bits from the device and holds them in a register. When the next 18 bits are transferred from the device, they are concatenated with the first 18 bits and written to KS10 memory.

4. Read-pause-write transfer on a read operation:

   The UBA transfers 18 bits from the device and writes them to KS10 memory in the left half of the word; garbage is written into the right half of the word. When the next 18 bits are transferred from the device, the first 18 bits are read from KS10 memory. Then the two half words are concatenated and written to KS10 memory.

KS10/PDP-11 Address Translation

The paging RAM in a UBA translates addresses between the KS10 and PDP-11. The paging RAM consists of 64 words (mapping registers) of 16 bits each. Of the 16 bits in a mapping register, five are used for control bits and eleven to specify a physical page in KS10 main memory. In other words, each mapping register can contain a KS10 physical page number, which is used to map a PDP-11 physical byte address to a KS10 physical memory address.

When the monitor wishes to perform an I/O operation to or from a page of KS10 memory, the KS10 page number must first be stored in a UNIBUS mapping register. The mapping registers are set up using the WRIO instructions. If the effective address of a WRIO instruction is between 1,,763000 - 1,,763077, the contents of the AC are stored in a mapping register in UBA 1. If the effective address is between 3,,76300 - 3,,763077, the contents of the AC are stored in a mapping register in UBA 3. Figure 3-1 shows examples of the contents of mapping registers in UBA 1.

UBA ADDRESS	RAM CONTENTS (KS10 PAGE NO.)	PDP-11 BYTE ADDRESS RANGE COVERED
763000	406	0 - 3777
763001	32	4000 - 7777
763002	305	10000 - 13777
.	.	.
.	.	.
.	.	.
763076	742	370000 - 373777
763077	5	374000 - 377777
763100	STATUS REGISTER	-

Figure 3-1
Example of Mapping Register Contents in UBA 1

The first mapping register, 1,,763000, maps PDP-11 addresses 0-3777; the second mapping register, 1,,763001, maps PDP-11 addresses 4000-7777 and so on. Mapping register 1,,763002 contains the KS10 page number 305. Hence, KS10

page number 305 maps PDP-11 addresses between 10000-13777. The PDP-11 address to transfer to or from is also set up using a WRIO instruction. The UNIBUS Adaptor then uses the mapping registers to translate the PDP-11 address to a KS10 page number.

Figure 3-2 depicts the address translation process in more detail. At the right of the diagram, an 18-bit address is transferred over the UNIBUS. The 18 bits are numbered PDP-11 style 0 to 17, where bit 0 is the rightmost bit in the 18-bit half word. Bit 0 is the "byte bit". If set to 0, this bit indicates the rightmost byte in a KS10 18-bit half word; if set to 1, it indicates the leftmost byte in a KS10 half word. Bit 1 is the "word bit". If set to 0, this bit indicates the leftmost half word in a KS10 full word; if set to 1, it indicates the rightmost half word in a KS10 full word.

Bits 2 to 10 of the 18-bit half word (nine bits in all) directly map the KS10 word location within a KS10 page. These bits are translated to become bits 35 to 27 of the KS10 address. Remember that bits 2 to 10 in the 18-bit half word are numbered right to left (PDP-11 style), while bits 27 to 35 in the KS10 address are numbered left to right.

Next, bits 11 through 16 of the 18-bit half word give the offset, which determines the mapping register to use in the UBA. For example, if we have:

Bit	16	15	14	13	12	11
	0	0	0	0	1	0

this indicates an offset of 2, or mapping register 763002 in the appropriate UBA. Note that these bits are numbered from right to left, PDP-11 style. The offset determines the mapping register. The contents of the mapping register give the KS10 page number. The KS10 page number is placed in bits 16 through 26 of the KS10 address to give a complete address.

As an example of this process, consider some of the setup instructions required for a disk transfer. Figure 3-3 shows some of the registers set up using the WRIO instruction. In the UBA, a mapping register must be set up with the KS10 page number for the transfer to or from the

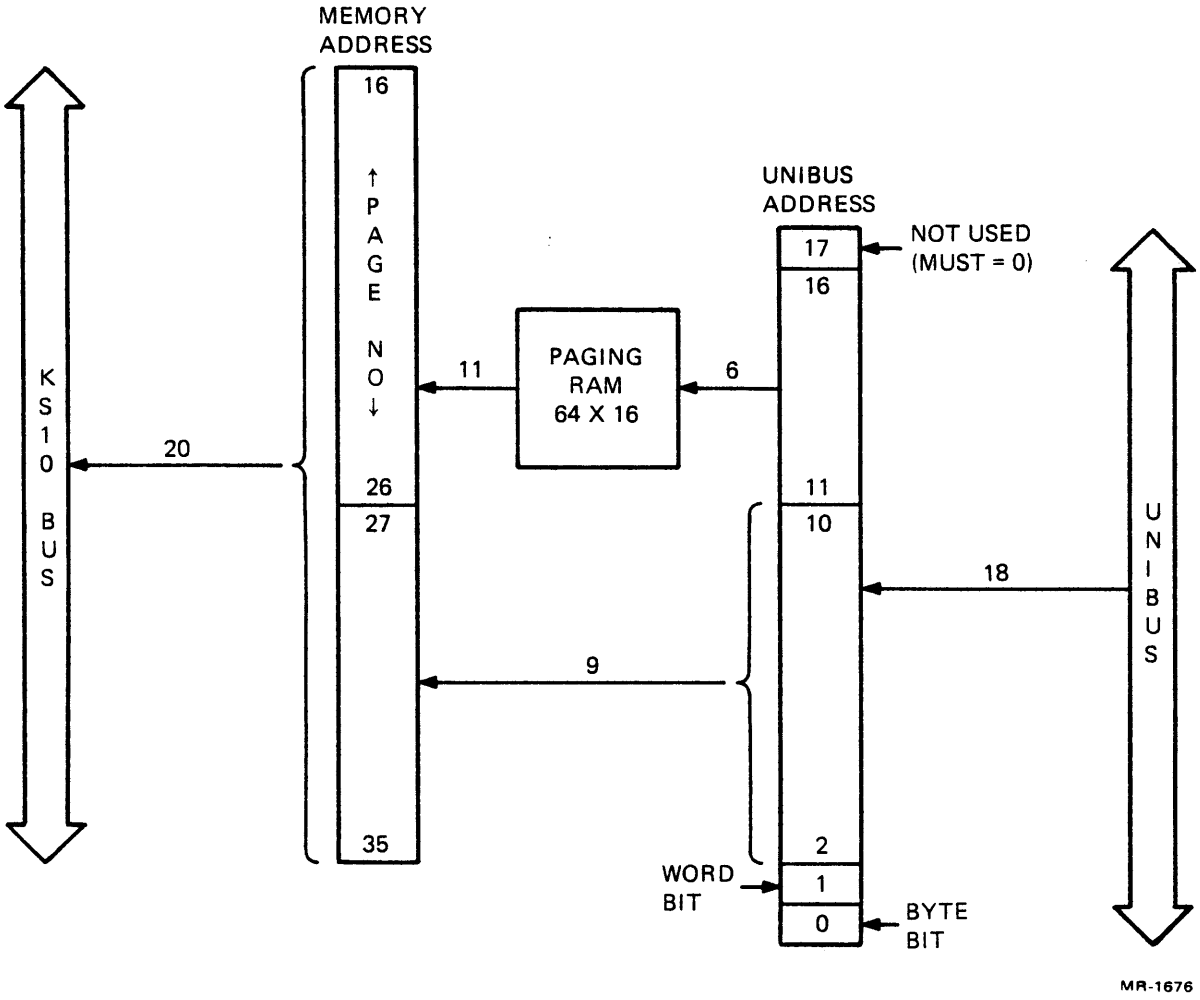


Figure 3-2.
KS10 Address Translation

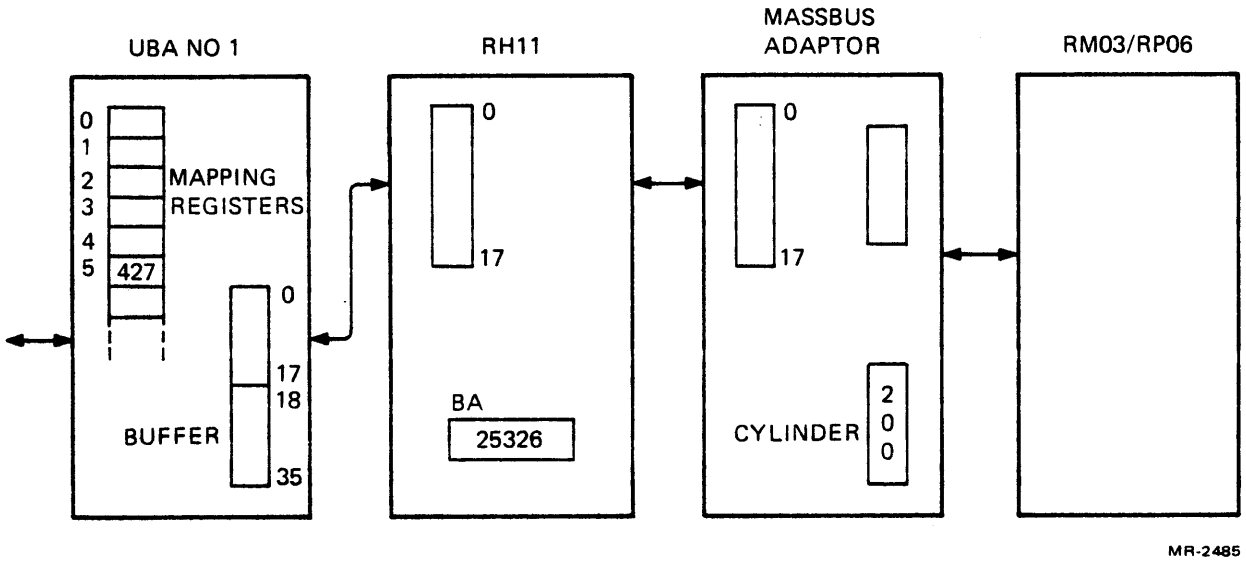


Figure 3-3.
Disk Transfer

disk. In the RH11 disk controller, the BA register must be set up with the PDP-11 address. In the MASSBUS Adaptor, a register must be set up with a cylinder number. These are just some of the registers whose contents must be established before the transfer can take place.

Suppose that the KS10 page number for the transfer is 427, and the word location in the page at which the transfer will begin is 265. In addition, we want to transfer to or from the right half word of word location 265 in page 427.

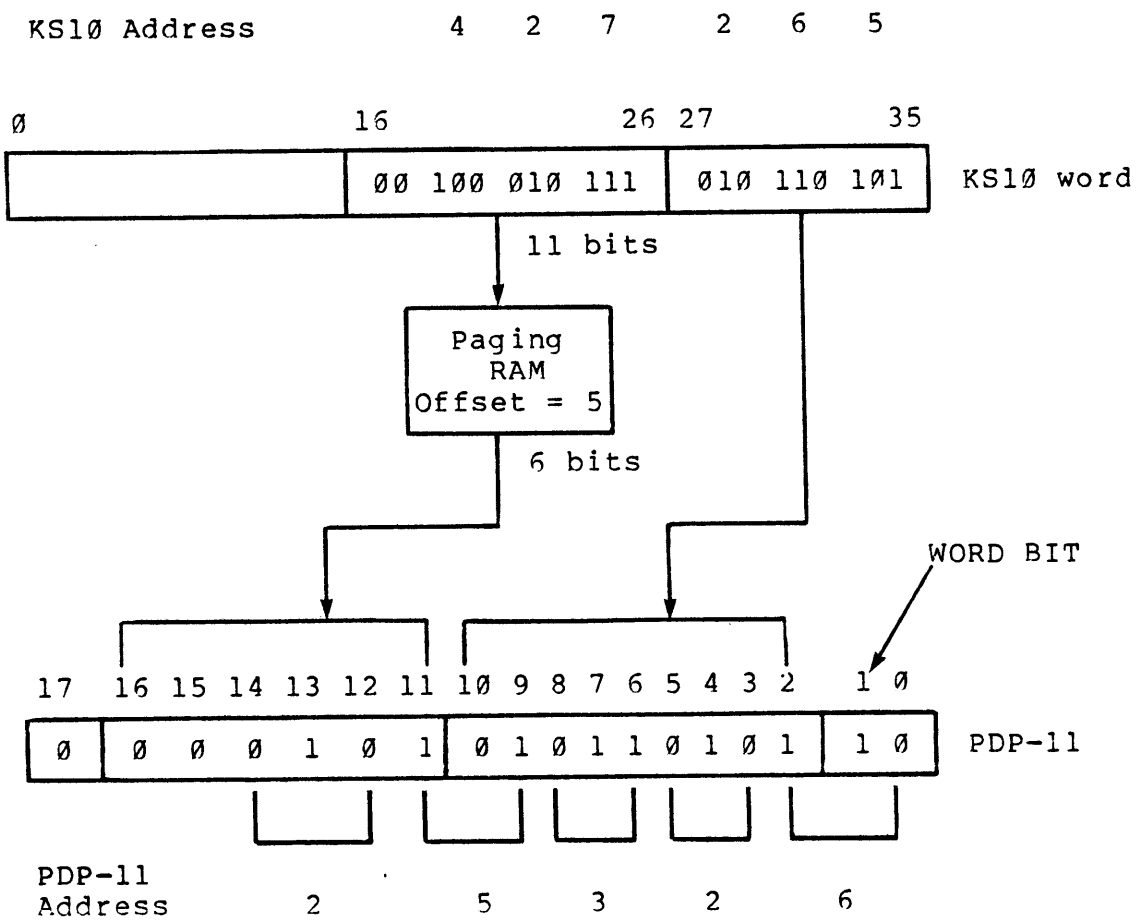


Figure 3-4
KS10 Address Translation Example

If 427 is placed in mapping register 1,,763005, for example, bits 11-16 (PDP-11 style numbering) of the PDP-11 address must contain a 5. Bits 2-10 will contain a 265. Bit 1 must be set to 1 to indicate the right half word.

Therefore, the PDP-11 address to set up in the RH11 BA register is 25326. This address is used with the mapping register contents to specify the KS10 page and word location for the transfer. Finally, the cylinder number (for example) is set up in the MASSBUS Adaptor.

The WRIO instructions that perform the above setup operations are:

(1) WRIO 17, @1000

```
Where 1000/1,,776734      ;776734 = cylinder register
      17/200
```

(2) WRIO 17, @1000

```
Where 1000/1,,776704      ;776704 = buffer address
      17/25326                    register in RH11
```

(3) WRIO 17, @1000

```
Where 1000/1,,763005      ;763005 = mapping register
      17/427
```

Note that the direction of transfer has not yet been specified. The above setup instructions are the same whether the transfer is to or from the disk.

If bytes are transferred, the byte in the KS10 word to begin with is specified by setting the word and byte bits in the PDP-11 address.

KS10 I/O INSTRUCTION SET

Opcode and AC Assignments

This section describes the operation of I/O instructions on the KS10. All I/O instructions are ordinary instructions, with the same format as normal instructions (opcode, AC, and effective address). The opcodes are in the range 700 through 777.

Any operation code in the range 700 through 777, AC number, field or bit not described in this section should be considered reserved to DEC. The microcode attempts to generate an illegal instruction trap whenever a reserved action is attempted.

Opcodes 700 through 737 are for exec mode only (I/O legal) instructions. Opcodes 740 through 777 are for exec and user instructions. There are currently no opcodes assigned in this range.

Table 3-1.
OPCODE Assignment Map

	0	1	2	3	4	5	6	7
700	APR0	APR1	APR2	-	UMOVE	UMOVEM	-	-
710	TIOE	TION	RDIO	WRIO	BSIO	BCIO	-	-
720	TIOEB	TIONB	RDIOB	WRIOB	BSIOB	BCIOB	-	-
730	-	-	-	-	-	-	-	-
740	-	-	-	-	-	-	-	-
750	-	-	-	-	-	-	-	-
760	-	-	-	-	-	-	-	-
770	-	-	-	-	-	-	-	-

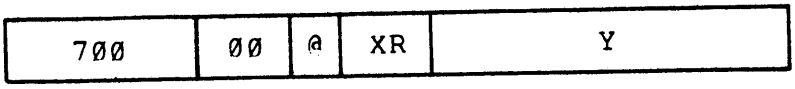
Table 3-2.
AC Field Assignments

AC	700	701	702
00	APRID	-	RDSPB
01	-	RDUBR	RDCSB
02	-	CLRPT	RDPUR
03	-	WRUBR	RDCSTM
04	WRAPR	WREBR	RDTIM
05	RDAPR	RDEBR	RDINT
06	-	-	RDHSB
07	-	-	-
10	-	-	WRSPB
11	-	-	WRCSB
12	-	-	WRPUR
13	-	-	WRCSTM
14	WRPI	-	WRTIM
15	RDPI	-	WRINT
16	-	-	WRHSB
17	-	-	-

Internal I/O Instructions

These instructions control the CPU. They do not transfer data between the outside world and the CPU or memory.

APRID



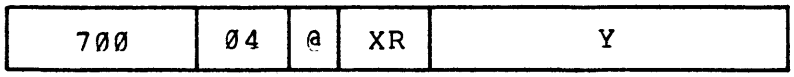
This instruction returns the microcode version number and the CPU serial number. The word is stored at the effective address, E, in the following format:

0-8	Reserved for microcode options
9-17	Microcode version number
18-20	Hardware options (at present these are all zero)
21-35	Processor serial number

NOTE

The processor serial number comes from the magic number field of CRAM location 1700. This number is not wired into the machine. The microcode, as supplied, contains a default serial number which can be changed during installation using the SMFILE program.

WRAPR



This immediate mode instruction decodes its effective address to control the processor. The effective address bits are used as follows:

19 I/O reset. When this bit is set, all internal devices are reset. In addition, an INPUT/OUTPUT initialization is generated.

20 Enable conditions selected by bits 24 through 34 to cause interrupts.

21 Disable interrupts for conditions selected by bits 24 through 31.

22 Clear flags indicated by bits 24 through 31.

23 Set flags indicated by bits 24 through 31.

NOTE

The action of the processor is not defined when both bits 20 and 21 or 22 and 23 are set in the same instruction.

25 Interrupt the 8080 (this bit causes an interrupt in the 8080 if bit 23 is also set).

26 Power fail

27 NXM (Non-existent Memory)

NOTE

This flag is set whenever any memory request generates an NXM condition. If the CPU generated the error, a page fail also occurs. If a UNIBUS request generated the error the NXM flag is set in the device requesting the NPR cycle.

28 Hard memory error (cannot be corrected by ECC)

NOTE

This flag set whenever any memory request generates a hard memory error condition. If the CPU generated the error, a page fail also occurs. If a UNIBUS request generated the error, the NXM flag is set in the device requesting the NPR cycle.

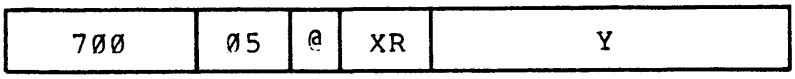
29 Soft memory error (correct data was placed on the bus)

30 Interval timer

31 Interrupt from 8080

33-35 PIA (Priority Interrupt Channel Number)

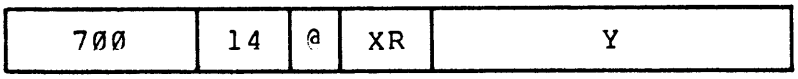
RDAPR



This instruction stores the APR status in the word addressed by E, the effective address. The status is as follows:

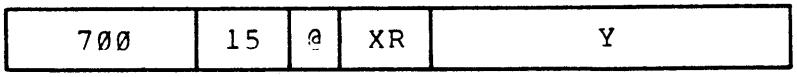
08	Power fail enabled
09	NXM (Non-existent Memory) error enabled
10	Hard memory error interrupt enabled
11	Soft memory error interrupt enabled
12	Interval timer enabled
13	From 8080 interrupt enabled
26	Power fail error
27	NXM (Non-existent Memory) error
28	Hard memory error
29	Soft memory error
30	Interval timer interrupt
31	Interrupt from 8080
32	Interrupt requested
33-35	PIA (Priority Interrupt Channel Number)

WRPI



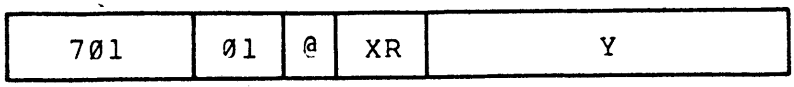
This instruction is identical to the CONO PI, instruction in the KL10 processor, except bits 18-20 (write even parity) are ignored.

RDPI



This instruction is identical to CONI PI, on the KL10.

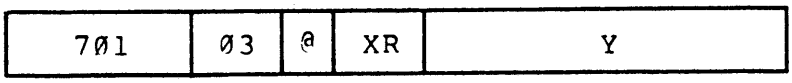
WRUBR



This instruction loads the user base register (UBR) with the word at E. The format of this word is:

0	Load AC block numbers
2	Load UBR
6-8	Current AC block
9-11	Previous AC block
25-35	Physical page number of UPT

RDUBR



This instruction reads back the user base register and returns a word in exactly the same format as used by WRUBR. In order to allow this word to be used directly in a WRUBR instruction, bits 0 and 2 are set to 1 in the result.

CLRPT

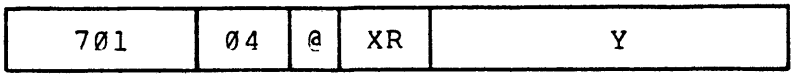
701	02	@	XR	Y

This instruction clears the hardware page table so that the next reference to the word at E will cause a refill cycle.

NOTE

There is only one entry in the page table for any virtual page. Clearing the mapping information for a page clears both the exec and user mapping.

WREBR



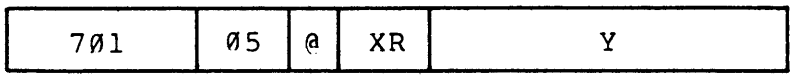
This immediate instruction looks at its effective address bits as follows:

21 TOPS-2Ø style paging

22 Trap and paging enable

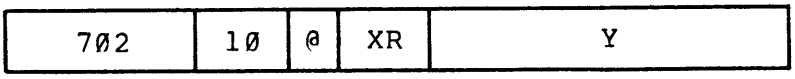
25-25 Physical page number of EPT.

RDEBR



This instruction returns the value given to WREBR and stores in the word addressed by E.

WRSPB



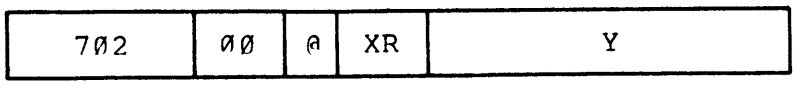
WRITE SPT BASE REGISTER

Load the word at E into the SPT base register.

BASE REGISTER FORMAT

All base registers are loaded with a physical word address. All high order bits must be zero. The address need not be on a page boundary and may be anyplace in physical memory. There is no range check on SPT or CST offsets. The monitor is assumed to always put correct data into page tables.

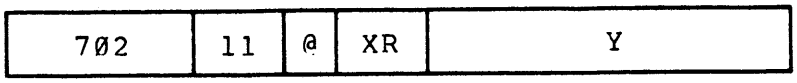
RDSPB



READ SPT BASE REGISTER

Store the SPT base register at E.

WRCSB



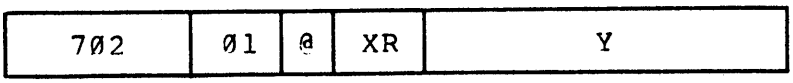
WRITE CORE STATUS TABLE BASE REGISTER

BASE REGISTER FORMAT

All base registers are loaded with a physical word address. All high order bits must be zero. The address need not be on a page boundary and may be anyplace in physical memory. There is no range check on SPT or CST offsets. The monitor is assumed to always put correct data into page tables.

Load the CST base register with the word at E.

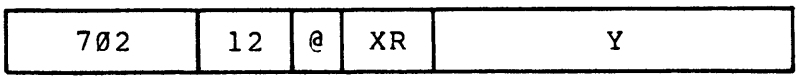
RDCSB



READ CORE STATUS TABLE BASE REGISTER

Store the CST base register at E.

WRPUR



WRITE PROCESS USE REGISTER

Load the process use register from E. The process use register contains the AGER (age register) in the left few bits. The bits containing the AGER are cleared by ANDing the CST entry with the CST MASK, then the entire PUR is ORed with the CST entry.

RDPUR

READ PROCESS USE REGISTER

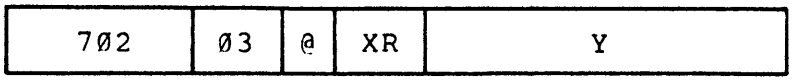
Store the PUR at E.

WRCSTM

WRITE CST MASK REGISTER

Load the CST mask register from E. The CST mask register should contain a 0 for every bit in the AGER and a 1 in all other bit positions.

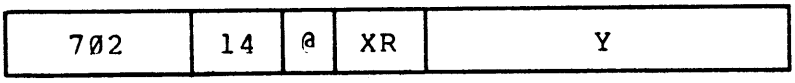
RDCSTM



READ CST MASK REGISTER

Store the CST mask register at E.

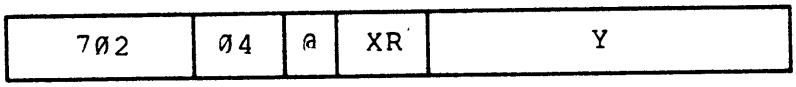
WRTIM



WRITE TIME BASE

Load the double word at E and E+1 into the time base. The time base counts up at 4.1 MHZ.

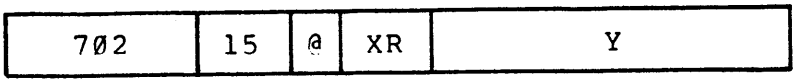
RDTIM



READ TIME BASE

Store the time base at E and E+1.

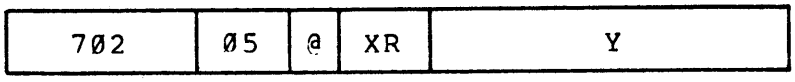
WRINT



WRITE INTERVAL TIMER

Load the interval timer period register with the word at E. The units are the same as the time base.

RDINT



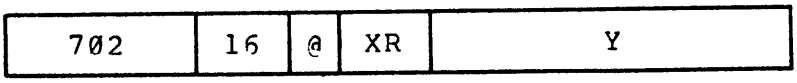
READ THE INTERVAL REGISTER

Store the current value of the interval register at E.

NOTE

This instruction returns the inverval loaded by WRINT. The value returned does not change with time or anything else except a WRINT instruction.

WRHSB



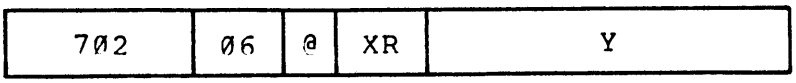
WRITE HALT STATUS BLOCK ADDRESS

Use the word at E as the address of halt status block. If this word is negative or zero, no halt status is stored. If the word is positive, internal processor status is stored.

The status consists of the 16 general purpose registers, the VMA, the FE and the SC. The programmer should allocate 32 words to allow this data area to expand in the future.

When the microcode starts, it sets the halt status block address to 376000. This can be changed by patching the magic number in CRAM location 3.

RDHSB



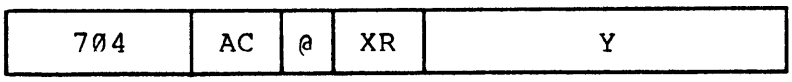
READ THE HALT STATUS BLOCK ADDRESS

Store the current value of the halt status block address at E.

PXCT EXTENSIONS

The following two instructions have been added to save time and space in the monitor.

UMOVE

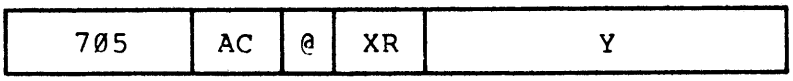


MOVE FROM PREVIOUS CONTEXT

Perform the same function as:

```
PXCT 4,[MOVE AC,E]
```

UMOVEM



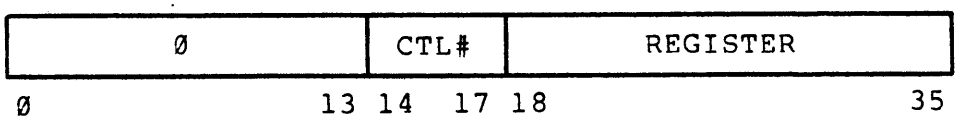
MOVE TO PREVIOUS CONTEXT

Perform the same function as:

```
PXCT 4,[MOVEM AC,E]
```

External I/O Instructions

The external I/O instructions on the KS10 allow a program to read, write, modify and test registers in external devices. They do this by generating a 36-bit extended address, which is used as an I/O address and has the following format:



where CTL# is the controller number, and register is a register address within the controller. Note that an extended address (greater than 18 bits) is required to address controllers other than zero. The effective address calculation for external I/O instructions (see figure 3-5) is as follows:

1. If there is no indexing or indirection, Y is used as the effective address.

2. If there is indirection (or indirection and indexing), Y (or Y indexed by bits 18-35 of the index register) is used as an address for an indirect word fetch and the contents of the indirect word (bits 14-35) are used as the effective address.

3. If there is indexing (and no indirection) and the left half of the index register is less than or equal to zero, Y indexed by bits 18-35 of the index register is used as the effective address.

4. If there is indexing (and no indirection) and the left half of the index register is positive, Y indexed by bits 6-35 of the index register is used as the effective address.

The operations described above are summarized in table 3-3. The result of the effective address calculation, which can be either an 18-bit or an extended I/O address is also indicated.

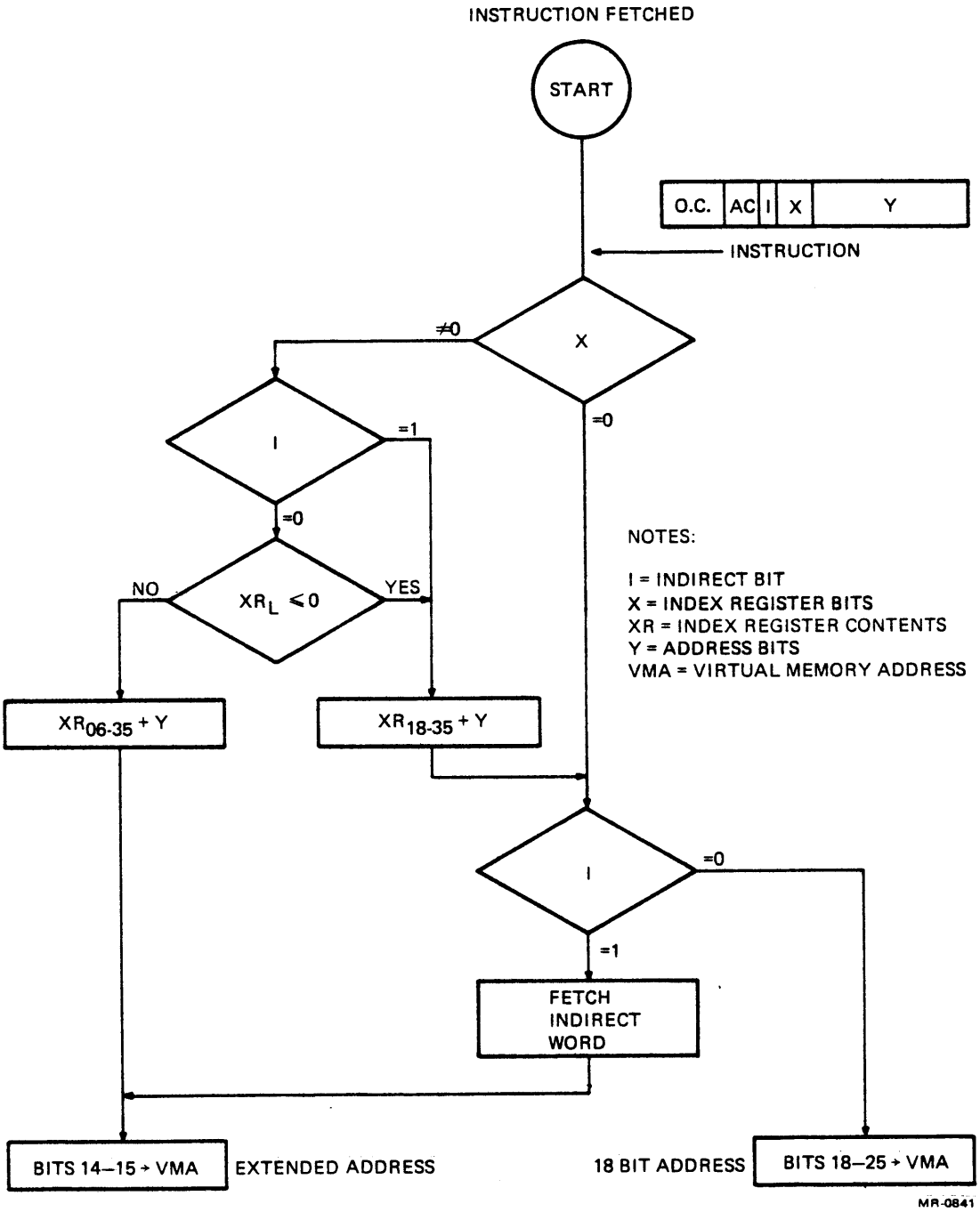


Figure 3-5.
Effective Address Calculation for External I/O Instructions

Table 3-3.
Summary of Effective Address Calculation
For External I/O Instructions

Indirection	Indexing	XR L	Effective Address	Comments
NO	NO	-	Y	18-bit Addr.
YES	NO	-	(Y)	Ext. Address
YES	YES	-	(Y + XR18-35)	Ext. Address
NO	YES	0	Y + XR18-35	18-bit Addr.
NO	YES	0	Y + XR06-35	Ext. Address

Controllers other than zero require an extended address and instructions using indexing or indirection. In the following examples, the RDIO instruction is used to read the status register (address = 763100) in UBA #1 (controller number = 1) into an AC.

Example 1 (using indexing):

```
RDIO AC,763100(X)        where the contents of index
                         register X = 100000
```

The index register contents (the controller number) is added to Y (the register address), giving the extended I/O address 1736100.

Example 2 (using indirection):

```
RDIO AC,@100             where 100 contains 1763100
```

An indirect word fetch of location 100 is made, and the contents are used to generate the extended I/O address 1763100.

The external I/O instructions come in two types, normal and byte. The normal type transfer 36 bits of data and use the full contents of AC. The byte type are for use only with UNIBUS devices. They transfer only 8 bits of data and use only AC bits 28-35.

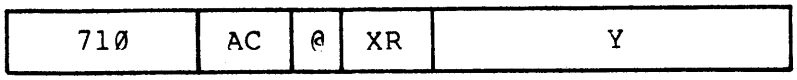
The specific I/O addresses for a fully configured KS10 are listed in Table 3-4. An asterisk (*) indicates a base address.

Table 3-4.
External I/O Addresses

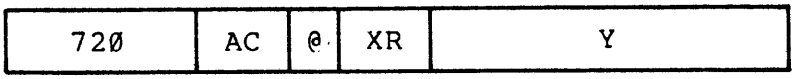
Register(s)	KS10 Bus Device	UNIBUS Device	CTL #	Register Address
Memory Status	Memory Cont.		0	100000
Console Instruction	Console		0	200000
UBA Paging RAM	UBA1		1	763000-77
UBA Status	UBA1		1	763100
UBA Maintenance	UBA1		1	763101
		UNIBUS 1		
	UBA1	RH11 # 1	1	776700*
UBA Paging RAM	UBA3		3	763000-77
UBA Status	UBA3		3	763100
UBA Maintenance	UBA3		3	763101
		UNIBUS 3		
	UBA3	RH11 # 3	3	772440*
	UBA3	LP20 # 1	3	775400*
	UBA3	DZ11 # 1	3	760010*
	UBA3	DZ11 # 2	3	760020*
	UBA3	DZ11 # 3	3	760030*
	UBA3	DZ11 # 4	3	760040*
	UBA3	KMC11 # 1	3	760540*
	UBA3	DUP11 # 1	3	760300*
	UBA3	DUP11 # 2	3	760310*

TIOE and TIOEB

TIOE



TIOEB



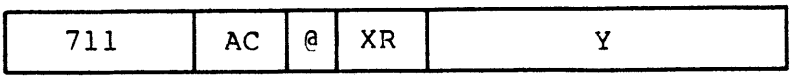
TEST INPUT/OUTPUT, no modification, skip equal

The effective address of this instruction is used as an INPUT/OUTPUT address. For TIOE, a word is fetched from the I/O register and "ANDed" with the contents of AC. For TIOEB, an 8-bit byte is fetched from the I/O register and "ANDed" with the contents of AC bits 28 through 35.

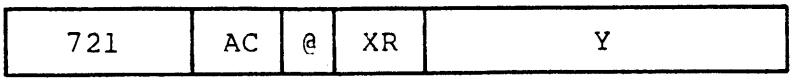
The instruction skips if the result of the "AND" is zero. The contents of the AC are not modified.

TION and TIONB

TION



TIONB

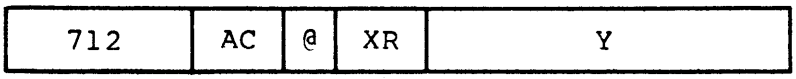


TEST INPUT/OUTPUT, no modification, skip not equal

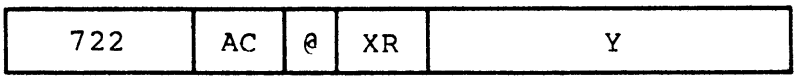
This instruction is the same as TIOE, except the sense of the skip is inverted.

RDIO and RDIOB

RDIO



RDIOB

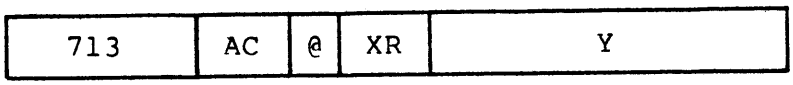


READ INPUT/OUTPUT

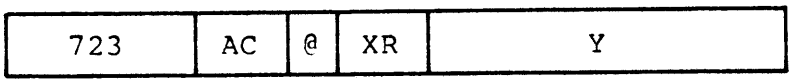
This instruction uses the effective address to address an I/O register. One word (or byte) is fetched from the I/O register and stored right-justified in AC.

WRIO and WRIOB

WRIO



WRIOB

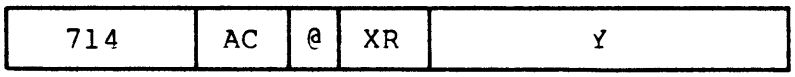


WRITE INPUT/OUTPUT

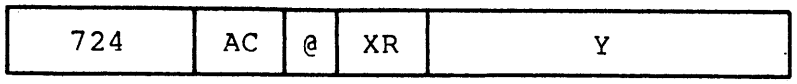
The effective address is used as an INPUT/OUTPUT address, and one word (or byte) is taken from AC and sent to the I/O register.

BSIO and BSIOB

BSIO



BSIOB



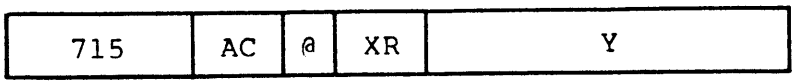
BIT SET TO INPUT/OUTPUT

The effective address is used to select an INPUT/OUTPUT address. One word (or byte) is fetched from the I/O register, "ORed" with the contents of AC, and written back to the I/O register. This operation does not use a read-modify-write cycle on the UNIBUS.

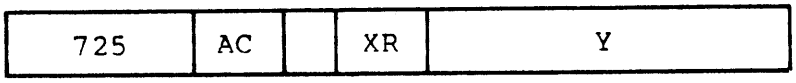
The contents of the AC are not changed by this operation.

BCIO and BCIOB

BCIO



BCIOB



BIT CLEAR INPUT/OUTPUT

The effective address is used to select an INPUT/OUTPUT register. One word (or byte) is fetched from the I/O register, "ANDed" with the complement of the contents of AC, and written back to the I/O register. This operation does not use a read-modify-write cycle on the UNIBUS.

The contents of the AC are not changed by this operation.

The net effect of this operation is to clear selected bits in INPUT/OUTPUT device registers.

DEVICE DRIVER REQUIREMENTS

CTY

The 8080 processor handles the I/O for the CTY. Low level support code and interrupt level code are in module SCNSER of the TOPS-10 monitor.

The 8080 processor handles CTY I/O one character at a time. When the 8080 reads a character from the CTY, it deposits that character into KS10 physical memory location 32 and interrupts the KS10. The 8080 causes an APR interrupt with a status bit when it is indicated that the interrupt was from the 8080. The 8080 does not check to see if the KS10 has removed the previous character. If the character has not been removed, it is overwritten by the new character.

When the KS10 wants a character typed on the CTY, it deposits the character in KS10 physical memory location 33 and interrupts the 8080. The 8080 sends the character to the CTY, deposits a zero in location 33, then interrupts the KS10. This can be considered the "done" interrupt. TOPS-10 checks to ensure that location 33 has been set to zero (indicating that the 8080 has picked up the last character) before it deposits the next character to be typed; therefore, no characters are missed on output.

At present, the 8080 interrupts the KS10 only when it has deposited a character read from the CTY, or when it has picked up a character from location 33 to be written to the CTY. The KLINIK line also uses this interrupt feature.

Terminals

Asynchronous terminals are supported in the monitor module DZINT. This module supports the DZ11; it is assembled with modules such as SCNSER.

For terminals where one 8-bit byte is transferred at a time, there is no problem with byte reversal, etc. The interrupt level code places each byte in the TTY character buffer associated with a line number, and calls SCNSER for each byte for an associated line. SCNSER transfers individual bytes to their lines. There is one interrupt per character. Running several terminals at 9600 baud may degrade throughput as compared with lower speeds.

NOTE

The DZ11 does not allow split speeds for input and output.

Line Printers

The line printer device driver is module LP2SER. For the line printer, the bytes are first shuffled to the following format, where each 8-bit byte contains one ASCII character:

00	BYTE 1	BYTE 0	00	BYTE 3	BYTE 2

Bits 0 and 1 and bits 18 and 19 are not used in this format; two 8-bit bytes are right-justified in each half of the word. Note that one 36-bit word (composed of five 7-bit bytes) is reformatted to five 8-bit bytes, four in the first word and one in the second word.

The line printer does not use the 18-bit transfer mode. The data transfer takes place one byte at a time, and the bytes are reordered to appear on the line printer as BYTE 0, BYTE 1, BYTE 2, BYTE 3, etc. The line printer driver sets up a byte count, and bytes are transferred individually over the UNIBUS to the line printer. When the byte count is exhausted, the KS10 gets a done interrupt.

Magtape

The module TRHKON, which supports the RH11, contains device-specific code for magtape. This module has been added for TOPS-10 on the 2020.

The UNIBUS uses two extra lines for data transfer, making it possible to transfer 18 bits at a time. This 18-bit mode is used for magtape and disk. For magtape, the left 18 bits of the word are transferred first, then the right 18 bits. If it is a read operation, 18 bits are read into the left half of the word. When the next 18 bits are received by the UBA, the word must be read from memory, concatenated, then written back to memory. This is the read-pause-write mode discussed earlier.

Disk

The module RHXKON contains device-specific code for the disk, which is the only device on the system using the 36-bit transfer mode. The disk RH11 also uses its UNIBUS in "hog" mode. The RH11 with disk units has no other devices on it. Therefore, the RH11 can keep the UNIBUS until a whole transfer request is satisfied, i.e., it does not have to give up the UNIBUS after each 18 bits is transferred. This is necessary to provide service to the RM03, and is the reason why this UBA can have only one RH11 (and no other devices) on it.

The RM03 device has 150 sectors per cylinder; consequently for the RM03, the disk allocation logic must handle this increase in sectors (RP06 drives have 20 sectors/track). Also, since 150 is not evenly divisible by four (the page size is four sectors), two sectors on each track are unused in the swapping area.

A system may have both RM03 and RP06 disks. RP04's are not supported.

General Steps for Device I/O

1. If writing in byte mode, shuffle the bytes into the format required for the peripherals. (See above.)

2. Associate the KS10 page addresses with some 4000-byte set of PDP-11 addresses. TOPS-10 allocates mapping registers to devices at system generation time. A line printer or a card reader is allocated two mapping registers (to handle transfers that cross a page boundary) in UBA 3. Magtape is allocated 16 mapping registers. Since each KS10 page contains 2K PDP-11 bytes, the largest single transfer possible for magtape is 16 x 2K = 32K bytes. All 64 mapping registers in UBA 1 are allocated for the disk; hence, the largest single disk transfer possible is 64 x 1/2K = 32K words. The teletype service does not need a mapping register, because each character is passed in the AC. When the KMC11 is implemented, it requires two mapping registers.

   Although the UBA mapping registers are allocated at system generation time, they must be set up at transfer time with the KS10 physical page (see MAPIO in KSSER).

   All 16-bit, 18-bit, and 36-bit transfer modes are set with flag bits in the mapping registers. The AC for the WRIO instruction to set up a mapping register contains the transfer mode flag settings and the KS10 physical page number. The modes are controlled with the following flag values:

   a. 16-bit mode UNBD18 on

   b. 18-bit mode UNBD18 off

   c. 36-bit mode UNB36B on

3. Set up the device registers for the device, specifying the PDP-11 address, the byte or word count, the status, etc. The PDP-11 address (specifying where to transfer to or from) determines which mapping register the UBA should use for translation to a KS10 page number; the mapping register contains the associated KS10 page number.

4. Set the status register, indicating direction of transfer. This starts I/O.

5. If reading, shuffle the bytes from PDP-11 format when the interrupt for transfer done comes in.

KS10 I/O INSTRUCTION EXAMPLES

```
Example 1:  J/RPACB
            DOAS == 16
            MOVE    P1, RHXBAS##(J)
            RDIO    T2, DOAS(P1)

            P1/     1,,776700
            T2/     CONTENTS OF REGISTER 776716
                    ON UBA #1 (THE DISK ATTENTION
                    SUMMARY REGISTER)


Example 2:  P3/CHN1CB
            RDIO    T1,@CHNUBA(P3)

            CHNUBA(P3)/   3,,763100
            T1/     CONTENTS OF UBA #3 STATUS REGISTER


Example 3:  DOOF == 32                  ;OFFSET
            MOVE P1, RHXBAS##(J)        ;SET UP INDEX REGISTER
                                        ;TO RH11 BASE ADDRESS
            SETZ T4,
            WRIO T4, DOOF(P1)           ;CLEAR OFFSET
                                        ;REGISTER
```

Chapter 4

TOPS-10 CHANGES FOR KS10

OVERVIEW

The major changes to the TOPS-1Ø monitor required by the DECSYSTEM-2Ø2Ø hardware are in the I/O routines. In the DECSYSTEM-2Ø2Ø, there is really no such thing as a "channel" per se; the concept of a channel is replaced by a combination of the UNIBUS Adaptors and the respective device controllers (RH11, LP2Ø, CD2Ø).

To adapt the TOPS-1Ø monitor to the 2Ø2Ø hardware, the feature test switch, FTKS1Ø, was added throughout the sources. All device I/O drivers (device-dependent code) were redone and a new processor module, KSSER (cf. KLSER), was added. Because of the hardware differences between the KS1Ø and KL1Ø, the bootstrap programs were reworked. Monitor-related software was also modified (eg. LPTSPL, SYSERR etc.), and new modules created for the microprocessor file system. FGEN and MONGEN were modified, so that the MONGEN dialog for generating a monitor is now simpler. The monitor modules and other modified software, together with new software, are shown in Table 4-1.

TOPS-10 MONITOR CALL CHANGES

Because of the KS1Ø and KL1Ø hardware differences, certain monitor calls do not exist for TOPS-1Ø on the 2Ø2Ø. For example, the KS1Ø has no address break facility; therefore, the SETUUO call with function .STABK is not implemented. All DECtape related UUO's do not exist for TOPS-1Ø on the 2Ø2Ø. Since there is no E-Box/M-Box style accounting or performance meters for the KS1Ø, all UUO's related to these hardware features do not exist. For example, the PERF. UUO does not exist for TOPS-1Ø on the 2Ø2Ø.

Another monitor call implemented for TOPS-10 on the KL10, but not on the KS10, is the DIAG. UUO. The DIAG. UUO, CALLI 163, allows a privileged program to set up a channel command list. For the RH11 controller, channel command lists do not exist; hence the DIAG. UUO is meaningless for the KS10.

Table 4-1.

TOPS-10/KS10 Affected Modules

MONITOR - EXISTING

CLOCK1	DATMAN	ONCMOD	SWPSER
COMCON	ERRCON	PSISER	SYSCHK
COMDEV	FILIO	PTYSER	SYSINI
COMMOD	FILUUO	S	TAPUUO
COMMON	KILOCK	SCHED1	UUOCON
COMNET	MSGSER	SCNSER	VMSER
CORE1	ONCE	SEGCON	

MONITOR - NEW

CD2SER	RHXKON
DZINT	TRHKON
KSSER	KDPINT (7.01)
LP2SER	

OTHERS - EXISTING

BOOTM	DDT	SPRINT
BOOTS	FGEN	SYSERR
WTBOOT	LPTSPL	UUOSYM
DAEMON	MONGEN	

OTHERS - NEW

SMFILE	KS10.ULD

TOPS-10 AFFECTED MODULES

Modules with Minor Changes

The table below gives those TOPS-10 monitor modules that are changed in minor ways for TOPS-10 on the 2020.

Table 4-2.

TOPS-10 Monitor Modules - Minor Changes

CLOCK1	FILIO	PSISER	SWPSER
COMCON	FILUUO	PTYSER	SYSCHK
COMNET	KILOCK	SCHED1	SYSINI
CORE1	MSGSER	SCNSER	TAPUUO
DATMAN	ONCE	SEGCON	UUOCON
ERRCON			

In several modules (for example, CORE1, DATMAN and PTYSER), the only changes are the addition of the feature test switch, FTKS10, to a few conditional assembly pseudo-ops. For example, below is a portion of a file compare for the standard 6.03 version of PTYSER and the changed version.

```
**************
1)3     IFN  FTKI10!FTKL10!FTKS10,<
1)              PUSHJ   P,RLCPTR        ;RELOCATE IT
****
2)3     IFN  FTKI10!FTKL10,<
2)              PUSHJ   P,RLCPTR        ;RELOCATE IT
```

Modifications to ONCE set up the PI assignments for UBAs 1 and 3. These assignments are checked once per second in the KSSEC routine in KSSER.

Modules with Major Changes

VMSER

This module was reworked because of the UNIBUS Adaptors on the 2020. A major routine starting at symbolic location THIS:: was added. This routine sets up the UNIBUS Adaptor mapping registers for a swap or page operation.

ONCMOD

The main addition to the ONCMOD module is a subroutine at symbolic location CHKCYL for the RM03 SWAP SAT (Storage Allocation Table.)

COMDEV

This module contains additions to the magtape Controller Data Block (KDB) and Unit Data Block (UDB). It also contains new LPT and CDR Device Data Blocks (DDB's) for the line printer and card reader.

COMMOD

This module contains additions as follows:

1. Additions to the Channel Data Block

   -CHANNEL TYPES 4,5,6 (RH11, LP20, CD20 respectively)

   -UBA VARIABLES

CHNUBA	UNIBUS Adaptor Status Register Address in format XWD UBA#,,UNBSTW where UNBSTW = 763100.
CHNIMR	Address of initial mapping register for this device on this "channel".
CHNMRC	Number of mapping registers available to this device on the channel.

CHNIEA	Initial eleven style address for the current transfer.
CHNBTC	Eleven style byte count for the current transfer.
CHNNXF	No transfer flag - used by FILSER to read a block without data transfer, to determine if there are errors.
CHNDBF	Dummy one-word buffer for bus address inhibit usage in RHXKON (read a sector, but no data transfer).
CHNRRV	Read reverse flag - used by MAPIO to set read reverse bits in the UBA.

2. Additions to disk KDB (controller data block).

S

Module S contains bit definitions and macros. The additions to module S.MAC for the KS10 processor and 2020 hardware include the following:

1. Hardware bit definitions for the KS10 processor.

2. Hardware bit definitions for the KS10 UBA and devices; e.g., UNBSTW = 763100, which is the basic UNIBUS status register address (minus the unit number). Also, control bits for the mapping registers (e.g., UNB36B for 36-bit transfers).

3. Definitions (OPDEFs) for the KS10 I/O instruction set. For example:

```
OPDDEF RDPI [700640,,0]   ;read the PI system
OPDDEF WRIO [713000,,0]   ;write I/O
```

4. Definitions (OPDEFs) for KL10 instructions that work on the KS10 (e.g., ADJSP - adjust stack pointer).

5. Macros to change the AC blocks.

COMMON

The most extensive changes and additions were made to this module. These changes include:

1. Code to define the KS10 Executive Process Table (EPT). Locations beginning at .EPVIT (EPT+101) point to vector interrupt tables, VECTB1 and VECTB3.

2. Code to generate a table for dispatching vectored interrupts. Dispatch addresses are initially set to zero, but the appropriate locations are set up by the device-dependent initialization routines.

3. Changes to the CPU data block.

4. Changes to the PI (priority interrupt) macros.

5. A macro to allocate UBA mapping register assignments.

6. A routine to define the KS10 APR interrupt handler.

7. Routines to initialize the KS10 time base. These routines perform the same functions as do the DK10 routines for the KA10 and KI10.

New Modules

New modules written for TOPS-10 on the 2020 consist of device drivers (device specific code) for the card reader, line printer, asynchronous and synchronous communications, disk, magtape and the KS processor itself. The module for the KMC11 (synchronous communication) is called KDPINT, but will not be available until the 7.01 version of the TOPS-10 monitor.

CD2SER

CD2SER is the card reader driver. The card reader driver implements support for a CD20 controller on the UNIBUS with the same functions as a CD20 controller on the KL10.

LP2SER

LP2SER is the line printer driver. It implements support for the LP20 controller on the UNIBUS, and involves taking 7-bit characters from the user's buffer and writing 8-bit characters to the LP20.

Unlike previous line printer controllers for the DECsystem-10, the LP20 controller contains a programmable translation RAM that translates computer characters to printing characters. LP2SER contains code to load the standard translation RAM; this is the only RAM supported (unlike the LP200 printer system for DECSYSTEM-20, which uses the MAKRAM program for user-created translation RAMs). The RAM is loaded (if not already loaded) automatically whenever output to the line printer is attempted (either via LPTSPL and a PRINT command, or via a COPY command).

THE LP05 or LP14 line printer models available for the 2020 have programmable Vertical Format Units (VFUs) rather than carriage control tapes. LP2SER contains code to load a standard VFU file. Like the RAM, the VFU is loaded (if not already loaded) automatically whenever output to the line printer is attempted. The user can also specify nonstandard VFU files to be loaded by LPTSPL, by means of entries in LPFORM.INI.

DZINT

DZINT implements support for the DZ11, including interrupt routines for local asynchronous lines. The implementation is similar to that for an existing DC10 line (i.e., the TTY's are not network terminals). This is an interim support for terminals. The long term support is similar to that which exists on the DN87s. Both interim and long term implementation are part of a larger project to support full ANF or DECnet communications on the KS10.

KDPINT

This module will be available with the 7.01 TOPS-10 monitor. It will support the KMC11 and DUP11's for synchronous communications lines.

RHXKON

The disk driver RHXKON is a reworking of RPXKON. It implements support of the RH11 controller for RP06 and RM03 disk drives.

Unlike the RH20 controller used with the KL10, the RH11 does not accept a channel command list. The channel command list allows "scatter reads and writes", i.e., a single disk transfer can specify non-contiguous areas of physical core. For the RH11, a disk transfer must be contiguous both on the disk and in the UNIBUS address space. Because of this, TOPS-10 buffered I/O can only operate at the rate of one block per transfer (equivalent to a transfer rate of one block per disk revolution, or 60 blocks/sec). This restriction may severely degrade jobs that perform a significant amount of I/O. However, dump mode I/O fully utilizes the disk.

TRHKON

The magtape driver TRHKON is a reworking of TM2KON. It implements support of the RH11 controller for TU45 tape drives. All functions currently supported for the TU45 on the DECSYSTEM-20, including tape read backwards, have been implemented.

KSSER

The KSSER module implements basic support for the KS10 CPU. It is the counterpart of KLSER for the KL10. The code in KSSER supports all APR functions (e.g., parity error, nonexistent memory, traps, clock handling) and mapping, to make memory addressable to the correct routines (particularly remapping I/O words). Multiple bit errors are treated just like hard parity errors in the KL10.

Chapter 5

OTHER SOFTWARE CHANGES

CHANGES TO EXISTING SOFTWARE

Existing system programs that are modified for TOPS-10 on the 2020 are listed in Table 4-1, Chapter 4.

Bootstrap Programs

The bootstrap programs BOOTS (boot a monitor from disk), BOOTM (boot a monitor from magtape) and WTBOOT (write BOOTS on blocks 4 through 7 of all disk packs in the system) are modified to take account of the new processor (KS10) and hardware (RH11 etc.) Using WTBOOT as a generic source program, WSBOOT was created to perform the same function for the KS10 that WLBOOT performs for KL10 systems.

Some of the changes and additions to BOOTS for the KS10 are:

1. Conditionals that test for KS10 processor (FTKS10) and RH11 controller (FTRH11).
2. OPDEFs for each KS10 I/O instruction.
3. Symbols that define UBA and RH11 registers.
4. Physical core definitions for KS10 CTY and KLINIK service.
5. Initialization and read/write routines for RH11 with RP06 or RM03.
6. A faster routine for zeroing core on the KS10.

DAEMON

DAEMON may be thought of as a swappable piece of the monitor. It performs functions such as writing entries in the error file (ERROR.SYS) and accounting files (FACT.SYS). The modifications to DAEMON for the KS10 support the new

hardware for system error entries. This includes a routine to determine if a crash is a KS10 system dump and, if so, to make a halt status block entry. The following error entries can be written:

RH11 Massbus Peripheral Errors
RP06
TU45
RM03
KS10 Processor Errors

Memory uses ECC (Error Correction Code) to correct single bit errors, but no error entry is stored.

DDT

DDT, used for the on-line checking and testing of MACRO-10 and FORTRAN programs, now checks for a KS10, and handles the KS10 CTY (EDDT only).

FGEN

The file FGEN.HLP contains all the feature test switches and their default settings. When generating F.MAC during the MONGEN dialog, MONGEN looks for and reads FGEN.HLP. FGEN.HLP includes the KS10 feature test set.

LPTSPL

LPTSPL includes DEVOPs for loading the RAM and VFU in an LP20 controller and LP05/LP14 printer, respectively.

MONGEN

MONGEN produces a simplified dialog for the DECSYSTEM-2020 and allows for the specification of KS10 devices and controllers.

SPRINT

SPRINT allows for the different image input for the CD20 controller.

SYSERR

SYSERR reports errors for KS10 specific devices, such as the RM03 disk drive and TU45 tape drive. Release 12A of SYSERR only supports brief entry reports with the 6.03A LIR; full support will be in release 13 of SYSERR, for the 7.01 monitor.

UUOSYM

UUOSYM includes new symbols such as extended I/O error codes and symbols for new devices such as RM03 and TU45.

NEW SOFTWARE

Two new software modules are provided for the 2020 in general. These are the microcode file for the KS10, and a program called SMFILE, which is used to create a file system for the microprocessor on a disk structure.

Microcode File - KS10.ULD

The microcode file is loaded into the KS10 CRA and CRM boards by the 8080 microprocessor. The microcode implements each KS10 instruction as a series of microinstructions. The file KS10.ULD is the same microcode file as that used for TOPS-20 on the DECSYSTEM-2020.

SMFILE.EXE

The SMFILE program is used to create the microprocessor file system on disk. It is also used to create a microcode file and a read-in magtape bootstrap file, which can be copied to a tape ("system" magtape); this is used for magtape booting of the monitor.

The microprocessor file system consists of a file called KS10FE.BIN. This file is created by SMFILE in a special directory, [6,2020] on a disk file structure (usually DSKB:). Three words in the disk structure home block point to this file (see Chapter 8).

The microprocessor file includes the following (note, however, that some of the diagnostic files are not normally included in the installed microprocessor file system):

1. The KS10 microcode.

2. A system preboot program. This is a page of code that the 8080 microprocessor reads into main memory. The preboot program then reads in the regular bootstrap program.

3. A diagnostic preboot program used for diagnostic purposes.

4. Bootcheck programs for the microcode and diagnostic code; these are used for diagnostic purposes.

5. The monitor bootstrap program.

The first page in the microprocessor file is used as an 8080 directory. It contains both the physical disk addresses and lengths of the associated files in the microprocessor file system.

The microprocessor file and the necessary pointers in the home block are set up at system installation time by running SMFILE and giving suitable commands. A system pack with a microprocessor file system can also be created at a later time, during system operation. The SMFILE commands are discussed in subsequent chapters.

UNSUPPORTED SOFTWARE

The Multiprogramming Batch (MPB) system is not supported for TOPS-10 on the 2020. Only GALAXY will be available. There is no front end on the 2020, so IBM links by way of DN6X software are not supported.

Chapter 6

INSTALLATION PROCEDURE

INSTALLATION TAPES

Three tapes are needed to install TOPS-10 on the DECSYSTEM-2020. These tapes comprise the regular TOPS-10 monitor and CUSP tapes, together with a 2020 installation tape that contains KS10 specific software. The tapes required for installation are:

1. The 2020 installation tape. This tape contains the files necessary to boot a TOPS-10 monitor on a "cold" machine. The file layout is as follows:

   a. KS10 microcode file in 8080 read-in format.

   b. BOOTM program in 8080 read-in format.

   c. BACKUP program in TOPS-10 saved format (.EXE).

   d. A BACKUP saveset containing a TOPS-10 monitor, SYSTEM.EXE, saved as DSKB:SYSTEM.EXE [1,4] at 1600 bpi. This is a minimal monitor for the KS10, used to bring up the system for installation.

   e. BACKUP saveset containing only modified 6.03 monitor modules, KS10 library file (TOPS10.REL), and the files needed to build a microprocessor file system. Also included are the files necessary to install GALAXY. A complete list of the files contained in the BACKUP saveset is given on the next page. This saveset is saved at 1600 bpi in interchange mode.

2. The 6.03A monitor tape. This is a standard TOPS-10 tape containing source versions of all monitor modules and the command files needed to build a standard TOPS-10 monitor. It also contains a number of executable files. For TOPS-10 on the

2020, many of these modules are superseded by KS10 specific modules contained in the BACKUP saveset on the 2020 installation tape. The monitor tape is recorded at 1600 bpi in interchange mode.

1600 bpi

KS10.RAM
BOOTM.RDI
BACKUP.EXE

BACKUP Saveset, 1600 bpi

DSKB: SYSTEM.EXE [1,4]

BACKUP Saveset, 1600 bpi, Interchange Mode

2020	CTL	CONKS	CMD	KSSER	MAC	S	MAC
2020	LOG	CORE1	MAC	LNKKS	CMD	SCHED1	MAC
BOOTM	EXE	DAEMON	EXE	LP2SER	MAC	SCNSER	MAC
BOOTM	MAC	DAEMON	MAC	LP64	RAM	SEGCON	MAC
BOOTM	RDI	DAEMON	REL	LP96	RAM	SMFILE	EXE
BOOTM	REL	DATMAN	MAC	LPTSPL	EXE	SMFILE	HLP
BOOTS	MAC	DDT	MAC	LPTSPL	LIR	SMFILE	TXT
BOOTS	REL	DDT	VMX	MAKPFH	EXE	SPRINT	EXE
BT192K	EXE	DLINK	EXE	MAKPFH	MAC	SPRINT	LIR
BT192K	REL	DZINT	MAC	MONGEN	EXE	SWPSER	MAC
BT256K	EXE	ERRCON	MAC	MONGEN	MAC	SYSCHK	MAC
BT256K	REL	FGEN	HLP	MONGEN	REL	SYSINI	MAC
BWR6KS	MEM	FILIO	MAC	MSGSER	MAC	TAPUUO	MAC
BWR6KS	RNO	FILUUO	MAC	NORMAL	VFU	TOPS10	REL
CLOCK1	MAC	GET	EXE	ONCE	MAC	TRHKON	MAC
CMBKS	CCL	GET	MAC	ONCMOD	MAC	UUOCON	MAC
COMCON	MAC	GET3	RND	PFH	VMX	VMSER	MAC
COMDEV	MAC	KILOCK	MAC	PSISER	MAC	WSBOOT	EXE
COMMOD	MAC	KS10	RAM	PTYSER	MAC	WSBOOT	REL
COMMON	MAC	KS10	ULD	RHXKON	MAC	WTBOOT	MAC

Figure 6-1.

Contents of the TOPS-10/2020 Installation Tape

3. The CUSP tape. This is a standard TOPS-10 tape, containing both source and executable versions of Commonly Used System Programs such as PIP, TECO, SOS etc. The executable versions of these programs

are usually placed in SYS: (DSKB: [1,4]) and protected <155> so that system users can access them. The CUSP tape is recorded at 16ØØ bpi in interchange mode.

In addition to the above three tapes, tapes containing unbundled software, such as DBMS-1Ø, may also be needed.

BOOTING FROM MAGTAPE

The installation tape described in the preceding section contains the files (in proper format) needed to boot a system from magtape. Before an attempt is made to boot the system and create a system pack, one or more formatted disk packs (RPØ6 or RMØ3) must be available. At present, packs are formatted by Field Service personnel under stand-alone conditions.

The procedure for booting the system from magtape is as follows:

1. Place the formatted disk pack (recommend using only a single pack for the system structure) on the drive (place the prospective system pack on drive Ø), set the controller select switch to A, and place on-line.

2. Place the installation tape on MTAØ: and set on-line.

3. Press the POWER ON switch at the 2Ø2Ø switch panel (see Figure 7-1 in Chapter 7), so the power on light comes on. When power is switched on, the microprocessor program is loaded and started. Typing a CRTL/C at the CTY returns the KS1Ø> prompt from the microprocessor program.

4. Type MT and a carriage return at the console to have the microprocessor initiate the magtape bootstrap procedure.

```
KS10> MT
KS10> USR MOD

BOOTM V5(22)

BTM>
```

5. Following the BTM> prompt, type /TM02 (works for either TM02 or TM03 controller). This loads the monitor from the installation tape; the ONCE-Only dialog is invoked, and you are asked to enter the date, time and startup option.

```
BTM> /TM02

SYSCHK (N,Y): Y

MEMORY MAP =
FROM       TO         SIZE/K
00000000   01777777   512

Installation Monitor 11-10-78
WHY RELOAD: NEW
DATE:  14-NOV-78
TIME:  1030

STARTUP OPTION:
```

If a tape drive other than unit 0 is used, the default unit can be changed with the MS microprocessor command (see Chapter 7). BOOTM can also be directed to another tape unit by typing /TM02:n following the BTM> prompt, where n is the tape unit (slave) number. For further information about BOOTM, see the 2020 Installation Guide, Appendix E.

CHOOSING THE STARTUP OPTION

The DESTROY option is recommended for a new installation's initial system startup (see following section). For installations who wish to choose their own disk structure parameters or create private structures, the LONG option is recommended (see section on ONCE-Only Dialog).

DESTROY Option

The version of the TOPS-10 monitor, SYSTEM.EXE, supplied on the installation tape, was generated using the value DSKB for the SIXBIT symbol M.SR00 (see the Installation Guide, Appendix A, for the SIXBIT default symbol values in the MONGEN dialog for HDWCNF). Selecting the DESTROY option automatically destroys all existing disk file structures, then restructures and refreshes the disk units as directed by MONGEN. Any disk units are automatically restructured as DSKB:, using default parameters. This provides a simple means of creating a DSKB: system structure that involves a minimum of dialog. It is recommended that only one pack be mounted, so a one-pack DSKB: is created.

The DESTROY option invokes the type of dialog shown in the following example:

```
STARTUP OPTION: DESTROY
%WARNING - ALL STRS WILL BE REFRESHED.
PROCEED? (Y OR <CR>) Y

%RPA1 IS OFF-LINE

%RPA2 IS OFF-LINE

HOME BLOCKS WRITTEN ON ALL UNITS
START SYSTEM? (Y OR <CR>) Y
TO AUTOMATICALLY LOG-IN UNDER [1,2] TYPE "LOGIN"
Installation Monitor 10:30:13

.LOGIN
```

ONCE Dialog

The ONCE-Only dialog is essentially the same as for TOPS-10 on the DECsystem-10. If you begin the installation using blank, formatted packs and wish to choose parameters for DSKB: or other structures, you must answer LONG to the STARTUP OPTION question. In the ensuing LONG dialog, any drives off-line are noted and can be set on-line. In order to define a disk structure, typically DSKB:, you must answer Y to the question:

```
DO YOU WANT TO CHANGE ANY DISK PARAMETERS?  (CR IF NO)
```

The dialog then requests the name of a structure to be defined and the names of the physical units in the structure (e.g. RPA0, RPA1, etc.).

After defining the structure by giving values for the structure and physical unit parameters, you must define the active swapping list and system search list, write HOME blocks on all defined structures, then answer NO to the question "DO YOU WANT TO CHANGE ANY DISK PARAMETERS?" Next all created structures should be refreshed by answering ALL to the question "TYPE STRUCTURE NAME TO BE REFRESHED".

When the STARTUP OPTION: question is displayed again, type Q for QUICK. You may then automatically log in under [1,2] by typing LOGIN.

```
STARTUP OPTION: Q
TO AUTOMATICALLY LOG-IN UNDER [1,2] TYPE "LOGIN"
.LOGIN
```

RESTORING FILES

After loading the monitor, the installation tape should position itself at the BACKUP program in saved (.EXE) format. Then, to save the BACKUP program as an executable file on disk (DSKB: [1,2]) for later use, type at the console:

```
.GET MTA0: BACKUP
.SAVE BACKUP
.RUN BACKUP                    ;to run BACKUP
```

If you lose BACKUP during this procedure, you can get, save and start it again by rewinding the installation tape to BOT, then typing GET MTA0: BACKUP three times, followed by SAVE BACKUP and START.

The files to be restored are those on the monitor tape, the CUSP tape and the installation tape (2020 modified files). It is convenient to restore the 2020 modified files last, so as not to overwrite them with unmodified software.

Accordingly, after starting BACKUP, proceed as follows:

1. Unload the installation tape, load the monitor tape on MTA0: and restore the following files (Interchange, density 800) to DSKB: [1,4].

```
/TAPE MTA0:
/UNLOAD
/INTERCHANGE                ;load monitor tape
/RESTORE SYS: = *.EXE, *.HLP, *.UNV
```

2. Unload the monitor tape, load the CUSP tape on MTA0 and restore the following files (Interchange, density 800) to DSKB: [1,4].

```
/UNLOAD
/TAPE MTA0:                 ;load CUSP tape
/INTERCHANGE
/RESTORE SYS: = *.EXE, *.HLP, *.ERR,
                DDT.REL, *.UNV, *.SYS
```

3. Unload the CUSP tape, load the installation tape on MTA0: and restore files in the BACKUP saveset to DSKB: [10,7].

```
/UNLOAD
/TAPE MTA0:                 ;load installation
/INTERCHANGE                ;tape
/SKIP 4
/RESTORE DSKB: [10,7] = *.*
/UNLOAD
/EXIT
```

4. Copy the 2020 modified files to SYS:

```
.COPY SYS:/X = DSKB:DAEMON.EXE[10,7],DSKB:GET.EXE[10,7]
.COPY SYS:/X = DSKB:*.VMX[10,7]
```

The above steps do not restore all files (e.g., source files) on the monitor and CUSP tapes to disk, since many of these are superseded by sources on the installation tape. However, all files needed to install the system and generate a system-specific monitor are restored.

SMFILE - MICROPROCESSOR FILE SYSTEM

The microprocessor file system is built in the directory DSKB:[6,2020], using the SMFILE program. The first step is to create a [6,2020] UFD, using CREDIR. Input files required by SMFILE are KS10.ULD (the microcode file) and BTxxxK.EXE, an absolute (non-relocating) version of BOOTS; xxx is the size of the system main memory in units of K. The installation tape supplies KS10.ULD (the microcode), and BT192K.EXE and BT256K.EXE, versions of BOOTS for a 192K and 256K system respectively. These files will both be found in DSKB:[10,7] after following the files restoration procedure outlined above.

You may use the distributed versions (BT192K.EXE and BT256K.EXE), or create your own executable BOOTS as follows (example for BT256K.EXE):

1. Log in as [1,2].

2.
```
.R MACRO
*BT256K = TTY:, DSKB: BOOTS [10,7]
*FTKS10 == 1
*CORE == 1000000                    ;for 256K memory
*^Z                                 ;CTRL/Z
*^Z                                 ;CTRL/Z
```

3.
```
.R DLINK
*BT256K/SAVE = BT256K/GO
```

A file 2020.CTL is supplied on the installation tape, and is restored to DSKB:[10,7]. This file contains the commands needed to build a 192K and 256K BOOTS, and also the commands needed to build WSBOOT, BOOTM and a KS10 monitor. DLINK is an interim LINK for linking absolute BOOTS.

The input files KS10.ULD and BTxxxK.EXE need not be in directory [6,2020] when running SMFILE. However, SMFILE does create the microprocessor file in [6,2020] on a chosen structure. In the following example, the microprocessor file is created in DSKB:[6,2020].

```
.RUN SMFILE [10,7]
DECSYSTEM-2020 FE-FILE PROGRAM
VERSION 50.0, TOPS-10, KS10, CPU 4097
SMFILE> WRITE SETUP DSKB:          ;write bootstrap
                                   ;program on DSKB
SMFILE> WRITE RESET                ;initialize pointer
                                   ;words in three
                                   ;HOME block
SMFILE> READ KS10.ULD [10,7]       ;read system microcode

SMFILE> SERIAL nnnn                ;give KS10 serial number
                                   ;this is written on the
                                   ;back door, KS10 cabinet
SMFILE> WRITE CRAM                 ;write the microcode
                                   ;into the microprocessor
                                   ;file
SMFILE> WRITE BOOT BTxxx.EXE [10,7] ;write the monitor BOOT
                                   ;program into the
                                   ;microprocessor file
SMFILE> WRITE DONE                 ;to update HOME blocks

SMFILE> EXIT                       ;to exit from SMFILE.
```

The above SMFILE dialog creates a microprocessor file, KS10FE.BIN, in DSKB:[6,2020]. It also creates a file BTxxxK.RIM, which is the disk monitor bootstrap loaded by the microprocessor file preboot program. The microprocessor file contains the KS10 microcode, a monitor preboot (written automatically by SMFILE) and a pointer to the monitor BOOT program. At system startup, the microprocessor program accesses this file and loads the microcode and preboot. Then the preboot program loads the monitor bootstrap, which loads and starts the TOPS-10 monitor.

The serial number of the KS10 processor is written inside the back door of the KS10 cabinet. Giving the serial number as shown above overwrites a default serial number that exists in the microcode as shipped. The serial number can then be read using the APRID I/O instruction (see Chapter 3). Serial numbers for KS10 processors begin at number 4097.

WRITING BOOTS ON ALL DISK PACKS

The WSBOOT program is used to write the disk bootstrap program, BOOTS, on blocks 4 through 7 of all disk packs in the system. WSBOOT is created for the KS10 processor using WTBOOT, as follows (see 2020.CTL):

```
.R MACRO
*WSBOOT = TTY:, DSK:WTBOOT
*FTKS10 == 1
*STDWTB == 0
*^Z
*^Z

.R MACRO
*BOOTS = TTY:,DSK:BOOTS
*FTKS10 == 1
*FT22BI == 1
*^Z
*^Z

.R LINK
*WSBOOT, BOOTS/GO
.SAVE WSBOOT
```

WSBOOT.EXE is also supplied on the installation tape, and thus is found in directory DSKB:[10,7]. WSBOOT is run once when the system is first installed. If disks are reformatted, new disks introduced or a new version of BOOTS distributed, WSBOOT must be run again.

WSBOOT is used to write BOOTS on all (or selected) disk packs, so the TOPS-10 crash and reboot procedure can work. To run WSBOOT, proceed as follows (you are logged in as [1,2]):

```
.RUN WSBOOT[10,7]

TYPE "HELP" FOR HELP
SELECT UNITS? Y
UNIT RPA0 (DSKB0): RH

EXIT
```

In the above example, if N was typed in response to the SELECT UNITS question, BOOTS would be written on all disk packs. When the select option is answered (respond with Y), the program types each physical unit that has a disk pack loaded and spinning (on-line). You may then respond with one of the following:

RH Writes RH11/RP06/RM03 BOOTS on the pack mounted on the unit.

ZERO Prevents the monitor from reading BOOTS from the pack mounted on the unit.

SKIP Prevents WSBOOT from changing (writing on) the pack mounted on that unit.

<CR> Writes RH11/RP06/RM03 BOOTS (the default).

WSBOOTS writes BOOTS only on packs that are on-line and not write-locked.

MONGEN

The MONGEN program is modified to accept "2020" as a DECsystem type. The dialog is considerably simplified for the 2020, since many MONGEN questions (for example, those concerning channels when generating HDWCNF.MAC) do not appear.

The modified version of MONGEN is supplied on the installation tape and will be found in DSKB:[10,7] after the installation. The first step in generating a monitor for a specific 2020 system is to run MONGEN and create the files HDWCNF.MAC, TTYCNF.MAC, NETCNF.MAC and F.MAC. Note that there are at present no networks, so the answer to "Network Software" when generating NETCNF.MAC is N.

Creating a Standard Monitor

If you are using a standard combination of feature test switches and are not using any unbundled software, you may use the TOPS10.REL library file of bundled monitor modules provided on the installation tape. Several command files are provided on the installation tape to facilitate monitor generation.

To generate a standard monitor, proceed as follows:

```
.R SETSRC
*LIB: DSKB:[10,7]
*^C

.COMPIL @CONSKS
.R LINK
*@LNKKS
.COPY SYS:/X = SYSTEM.EXE
```

In the above example, the command files CONSKS.CMD and LNKKS.CMD contain all the commands necessary to compile the modules F, S, HDWCNF, TTYCNF, NETCNF, COMMON, COMDEV, and COMMOD, link these with TOPS10.REL and save the monitor as SYSTEM.EXE. The monitor can then be copied to SYS:.

Creating a Nonstandard Monitor

If you: (1) are using a nonstandard combination of feature test switches, (2) have SOUPed in your own changes, or (3) have unbundled software, you must assemble all monitor modules and create your own version of TOPS10.REL. Command files CONKS.CMD and CMBKS.CCL are supplied on the installation tape to facilitate this. The procedure for assembling and linking a nonstandard monitor is given in section 5.3.2 of the Installation Guide.

UETP

The UETP (User Environmental Test Package) is a collection of programs, data files and batch control files. These are designed to allow you to verify system integrity by exercising a variety of hardware and software elements installed in a DECsystem-10, DECSYSTEM-20 or DECSYSTEM-2020 running either TOPS-10 or TOPS-20. For the DECSYSTEM-2020, TOPS-10 UETP will not be available until the 7.01 release of the monitor.

The primary goal of UETP is to provide an automated test package for verifying system integrity. The UETP system acts as an interface between the user and the TOPS-10 batch system (GALAXY). The user can selectively access and run verification tests for specific devices (e.g. MTA0:) software (e.g. COBOL, FORTRAN) or other applications. Each test run under UETP generates a file containing the batch control log for the test. It also generates an Exception Log file, containing any error messages generated, and a Run Log file, containing milestone messages (beginning and ending of each UETP test, etc.).

UETP can be used to check hardware and software performance at installation time, or at any later time.

Chapter 7

OPERATOR PROCEDURES

OPERATOR CONSOLE AND KS10 SWITCH PANEL

Operator Console

The DECSYSTEM-2020 consists of two major components, the 8080 microprocessor and the KS10 processor. Whenever power is switched on by means of the power switch on the KS10 switch panel (see figure 7-1), a console program automatically runs in the microprocessor. The console terminal, or CTY, is connected directly to the microprocessor; commands typed at the CTY (or entered via the KLINIK link, which also connects directly with the microprocessor) are interpreted by the console program. The way in which commands are interpreted depends on the console mode.

The CTY operates in either of two modes:

1. Console mode

2. User mode

In console mode, commands typed at the CTY are interpreted by the console program as console commands. These commands are used for the boot procedure, diagnostics, error recovery and maintenance. The CTY is initialized to console mode at power up, and the prompt KS10> appears. Some examples of console commands are:

DS - used to select a disk drive for the bootstrap procedure
MS - used to select a tape drive for the bootstrap procedure
BT - used to load and start the disk boot program
MT - used to boot the system from magtape

These console commands and others are discussed in more detail in the following sections. When in console mode, you can switch the CTY to user mode by:

1. Booting the system with the BT or MT command

2. Continuing execution with the CO command

3. Starting the KS10 processor at address xxxxxx with the ST xxxxxx command

4. Typing a CTRL/Z

With the CTY in user mode, the operator can run TOPS-10 programs and communicate with timesharing users. In user mode, all characters input at the CTY are passed to and from the KS10, under control of the console program. All commands (except CTRL/backslash) are processed by the TOPS-10 operating system resident in KS10 main memory, so all features of TOPS-10 can be accessed. To return from user mode to console mode, a CTRL/backslash is typed at the CTY; however, this succeeds only if the LOCK switch on the KS10 switch panel is in the unlock position. Setting the LOCK switch to the lock position during timesharing prevents an inadvertent change to console mode.

When switching from user mode to console mode, the message ENABLED prints on the CTY. When switching from console to user mode, the message USR MOD prints on the CTY. Up to 80 characters can be typed on a single line in console mode. Also, more than one command can be typed on the same line, by separating each command with a comma. All numbers typed in console mode must be octal.

KS10 Switch and Indicator Panel

The KS10 Switch and Indicator Panel is shown in Figure 7-1. At the bottom of the panel are four rocker switches and a 3-position, key-operated switch for remote diagnosis. The functions and positions of each switch are as follows:

Switch	Function
BOOT	Used to boot the system from a default disk device or a selected disk device.

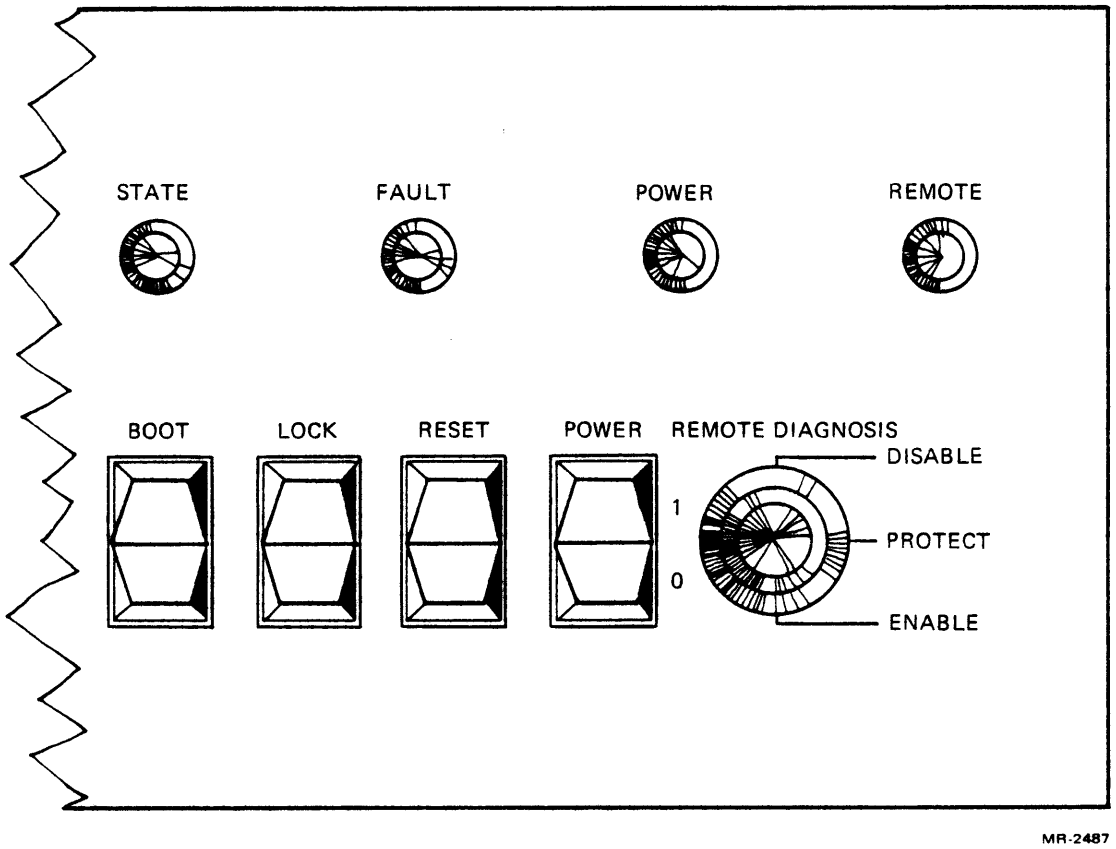


Figure 7-1.
KS10 Switch and Indicator Panel

LOCK	Disables the BOOT and RESET switches; also disallows CTRL/backslash at the CTY.
RESET	Reinitializes the system (equivalent to turning on the power), but does not initiate the boot procedure. An ECO scheduled for release in April, 1979, includes a new microcode, version CSL 4.2. With this version, pressing RESET causes an attempt to start the system at location 71, which is a power/fail restart.
POWER	The "1" position means ON, the "Ø" position means OFF. When switched to ON, the console program runs and the system attempts to load automatically from the default device after 3Ø seconds. To prevent the automatic load, type CTRL/C at the CTY.
REMOTE DIAGNOSIS	This is a key-operated, 3-position switch with the key removable in any position. DISABLE prohibits access to the system via the KLINIK link. PROTECT allows access via the KLINIK link, but only if a password entered from the link matches a password previously entered by the operator via a console command. ENABLE allows access to the system via the KLINIK link without password protection.

Above the panel switches are four indicator lights. These perform the following functions:

Light	Condition	Explanation
STATE	ON	The KS1Ø has successfully loaded the microcode and is up and running, but is not yet passing characters to TOPS-1Ø.

	OFF	The KS10 has stopped running the microde.
	BLINKING	The KS10 has successfully loaded the monitor, is in the timesharing state, and is exchanging dialogue with the console program in the microprocessor. However, during input and output to the CTY, the blinking is delayed.
FAULT	ON (during system loading)	The system is malfunctioning. This can mean that there is an error in device selection or that there is a KS10 fault. Make sure the disk or magnetic tape is mounted on drive 0. Try to reload. If the reload fails, call your Field Service representative.
	ON (during timesharing)	The KS10 has stopped. Try to reload. If the reload fails, call your Field Service representative.
	OFF	The KS10 is operating normally.
POWER	ON	The central processor is powered up.
	OFF	The central processor is powered down.
REMOTE	ON	The KLINIK link can be accessed with or without a password.
	OFF	The KLINIK link cannot be accessed.

LOADING THE SYSTEM

Automatic System Load and Reload

Thirty seconds after switching power on, the system attempts an automatic load of the TOPS-10 monitor from disk. An automatic reload is attempted after a TOPS-10 crash (STOP stopcode), or after the system is brought down (using the SH microprocessor command) without first giving the KSYS command to end timesharing. Note that a system shutdown should consist of a KSYS command, then control-backslash and the SH microprocessor command at the CTY. The default parameters used during an automatic load or reload are:

1. UBA1 - UNIBUS Adapter number 1
2. RHBASE 776700 - address of RH11 controller
3. Unit 0 - disk unit 0 (RPA0)
4. SYSTEM.EXE[1,4] - TOPS-10 monitor

In an automatic load, the console program goes to disk unit 0 to load the preboot and boot programs, using the microprocessor file system. The boot program then attempts to load the monitor, SYSTEM.EXE, from the [1,4] area. For an automatic load, the message:

```
BT AUTO
```

appears on the CTY. If the automatic load fails (e.g., there is no microprocessor file on disk unit 0), the microprocessor gives up.

For an automatic reload (after a crash or system shutdown without a KSYS), the word BOOTS is printed on the CTY and the system automatically reloads after 30 seconds. To stop an automatic load or reload, you must type a CTRL/C on the CTY within 30 seconds after switching on power, or after a crash.

When an automatic load succeeds, the monitor is loaded into main memory, the ONCE-Only dialog is invoked and the startup questions (WHY RELOAD, DATE, TIME) must be answered. The following messages appear on the console:

```
KS10  CSL.Vxx.x              ;microprocessor name
                             ;and version number
BT AUTO
BOOTS V23(110)               ;BOOTS version number

BTS>                         ;BOOTS prompt
```

For an automatic reload, the following appears on the console (the system retains date and time and can bring itself up without operator intervention):

```
BOOTS                         ;system down

SYSCHK (N,Y): N
TOPS-10 6.03  2020 System   06-15-78
WHY RELOAD: OTHER ;OPR DID NOT ANSWER
DATE: 11-10-1978
TIME: 1111

STARTUP OPTION:  QUICK   ;DEFAULT

%RPA2 IS OFF-LINE

%RPA3 IS OFF-LINE

TOPS-10 6.03  2020 System  11:11:15 CTY system 4112
.LOGIN 1,2
.R OPSER
[OPRPAF Processing auto command file]
```

Loading via the BOOT Switch

The BOOT switch is used to load the TOPS-10 monitor using default parameters. Make sure power is ON, the LOCK switch is OFF and the system pack is on unit 0. After the BOOT switch is pressed, the following messages are printed on the CTY:

```
KS10>BT SW
BOOTS V23(110)

BTS>                              ;load SYSTEM.EXE [1,4]

SYSCHK (N,Y): N
TOPS-10 6.03  2020 System   06-15-78
WHY RELOAD:
```

If you do not reply to the WHY RELOAD question within 15 seconds, the system prints OTHER ;OPR DID NOT ANSWER, the DATE: prompt, and waits.

Loading Using the BT Console Command

Default Device (Disk Unit 0)

The BT command can be typed following the KS10 prompt. The system attempts to load, using the default parameters and device, or the last specified device and parameters (see the following section).

For example, to load the system using the default parameters and device, type BT following the KS10 prompt, and a carriage return following the BTS> prompt.

```
KS10>BT
BOOTS V23(110)

BTS> <CR>                         ;type a carriage return to
                                  ;load the default monitor,
                                  ;SYSTEM.EXE[1,4]
```

To load a monitor with a name other than SYSTEM.EXE from the default device (disk unit 0), type the name of the monitor and the PPN following the BTS> prompt, then a carriage return.

Example: BTS>TSTMON.EXE[10,7] <CR>

The above command loads TSTMON.EXE from [10,7] on disk unit 0.

Non-Default Disk Device

To load a monitor from a non-default disk, you must tell the console program which unit to use. To do this, use the DS console command as follows:

```
KS10>DS                 ;to specify a non-default
                        ;disk unit
>>UBA?                  ;type 1 or <CR>

>>RHBASE?               ;type 776700 or <CR>

>>UNIT?3                ;type unit number, e.g., 3

KS10>BT                 ;type BT
BOOTS V23(110)

BTS>                    ;type <CR> for SYSTEM.EXE[1,4]
                        ;or type name of monitor.
```

Then answer the monitor startup questions: WHY RELOAD, DATE, and TIME.

Loading From Magtape - MT Command

The tape should contain the microcode, magtape bootstrap program (BOOTM) in tape read-in format, a copy of BACKUP.EXE and a BACKUP saveset containing the monitor. This is the same format as for the installation tape. The section "Creating a Monitor Backup Tape", in Chapter 8 shows how to create such a tape for your system.

The default tape unit is 0. If another tape unit is used, it must be specified using the MS command. In the following example, tape unit 2 is used.

```
KS10>MS                 ;to specify a non-default
                        ;tape unit
>>UBA?                  ;type 3 or <CR>

>>RHBASE?               ;type 772440 or <CR>
```

```
>>TCU?                    ;type tape controller number
                          ;(Ø or <CR>)

>>DEN?1600                ;type tape density

>>SLV?2                   ;unit number of drive,
                          ;eg., 2

KS10>MT                   ;to load from magtape
KS10>USR MOD
BOOTM V5(23)

BTM>/TM02                 ;to load SYSTEM.EXE[1,4]
```

To load a monitor of another name, type MONNAM.EXE[ppn]/TMØ2 following the BTM> prompt, where MONNAM is the monitor.

ENDING TIMESHARING

Assuming that OPSER is running from the CTY, the first step in ending timesharing is to give the :KSYS command with either hhmm or +hhmm as argument. When the KSYS time has expired, type CTRL/backslash at the CTY to put the CTY in console mode. The message ENABLED and the KS1Ø> prompt print on the CTY. Following the KS1Ø prompt, type SH to stop the KS1Ø processor. The message USR MOD followed by BOOTS is printed on the CTY.

Example

```
.ENABLED                  ;CTRL/\ was typed
KS10>SH                   ;halt the KS1Ø processor
KS10>USR MOD

BOOTS
```

CONSOLE COMMANDS AND ERROR MESSAGES

Console Commands

The following commands (all executed in console mode except for CTRL/backslash) are a subset of the microprocessor console program commands. They are the commands that an operator, system programmer or software specialist will probably use most often. For a complete list of console commands and their functions, refer to Specifications in the DECsystem-10 Software Notebooks (notebooks 11, 12).

Command	Function
BT (Boot)	Boots the system using the last disk drive selected. The microcode and the monitor bootstrap are loaded and started from unit 0 on UBA1 (default address) or from the drive selected with the last DS command. The KS10 starts at memory address 1000. The bootstrap program promts with BTS>.
MT	Boots the KS10 from magnetic tape. This command loads and starts the microcode and the monitor boot program from tape unit 0, from slave unit 0 on UBA3 (default address) or from the drive selected by the last MS command.
DS	Selects the disk for bootstrap. The console program asks for the UBA number (default = 1), the RH11 base address (default = 776700) and the disk drive unit number (default = 0) as follows: >>UBA? 1 >>RHBASE? 776700 >>UNIT? 0

The default value for the RH11 base address is currently the only value permitted. Also, the current value is retained if you press RETURN in response to any question.

MS

Selects the magnetic tape for bootstrap. The console program asks for the UBA number (default = 3), the RH11 base address (default = 772440), the tape controller unit number (default = 0), the tape density (default = 1600 bits/inch) and the slave (drive unit) number (default = 0) as follows:

```
>>UBA? 3
>>RHBASE? 772440
>>TCU? 0
>>DENS? 1600
>>SLV? 0
```

The default value for the RH11 base address is currently the only value permitted. Also, the current value is retained if you press RETURN in response to any question.

CO

Continues KS10 program execution. This command allows you to enter user mode.

HA

Halts the KS10 program. This command causes the microcode to enter the halt loop.

SH

Shutdown command. This command deposits non-zero data into KS10 memory location 30 to allow an orderly shutdown of the monitor.

SI

Single instruct. This command executes the next KS10 instruction.

EM xxxxxxx

Examines the contents of KS10 memory address xxxxxxx.

ST xxxxxxx	Starts the KS1Ø program at address xxxxxxx. This command allows you to enter user mode.
KL xx	KLINIK command. This command enables a remote link with system access to operate in user mode, but not in console mode when xx = Ø. However, if xx = 1, a remote link with system access can operate in either console mode or user mode. After you give a KL command, press RETURN to receive the current value.
ZM	Zeroes memory. This command deposits Øs into all KS1Ø memory locations.
PW xx	Sets the KLINIK password xx, where xx = maximum of six alphanumeric characters. To clear the password storage area, give a PW command and then press RETURN.
CTRL/backslash	Switches the CTY from user mode to console mode.
CTRL/O	Suppresses printing at the terminal.
CTRL/Q	Resumes printing at the terminal after a CTRL/S has stopped it.
CTRL/S	Stops printing at the terminal and hangs the microprocessor.
CTRL/U	Deletes the current line.
CTRL/Z	Switches the CTY from console mode to user mode.
DELETE KEY (or RUBOUT)	Deletes the last character.

Console Error Messages

Error messages are returned by the console program, some of which may occur when TOPS-10 is running. All error messages are preceded by a question mark and appear on the CTY. In most cases, consult with your Field Service representative if an error occurs.

Examples of console error messages are:

?MRE

Memory refresh error (occurs during timesharing).

?BFO

Buffer overflow. Occurs if more than 80 characters are typed on a single line while in console mode (occurs during timesharing).

?BT xxxyyy

Boot error where xxxyyy is an octal code representing the error (see DECSYSTEM-2020 Operator Guide, Appendix B).

?IL

Illegal command. The console program did not recognize the sequence of characters typed as a legal command. This error message is typed if a KLINIK user enters an incorrect password.

?PAR ERR

KS10 Bus parity error.

For a complete list of console error messages and their explanations, see the DECSYSTEM-2020 Operator Guide, Appendix B.

The console program reads and prints (at the CTY) the contents of certain 8080 registers in response to the EB (Examine Bus) command and a system parity error (?PAR ERR). The EB command prints registers 100-103, 300 and 301. Registers 100, 303 and 103 print when the system parity error is detected.

KLINIK

The KLINIK link allows a DIGITAL Field Service representative or software specialist to diagnose a problem in the system from a remote location. By using the KLINIK link, DIGITAL's Field Service product support group and software specialists can collect and analyze performance data without travelling to the installation.

The KLINIK link can be in one of four states, depending on the position of the REMOTE DIAGNOSIS switch on the front panel, the commands the operator types and the commands the KLINIK user types. The four states are:

1. State 1 - Unavailable

2. State 2 - Awaiting Password

3. State 3 - Timesharing User

4. State 4 - Remote CTY

The link must be in State 4 to use the KLINIK facility.

In State 1, the REMOTE DIAGNOSIS switch is in the DISABLE position. The KLINIK link is also in State 1 if the REMOTE DIAGNOSIS switch is in the PROTECT position, and the operator has not typed a password in memory with the PW command. In the latter version of State 1, the user at the remote site receives the message ?NA, (not available).

In State 2, the REMOTE DIAGNOSIS switch is in the PROTECT position, and the operator has typed a password with the PW command. The console program is waiting for the KLINIK user to type a password for verification. In this state, the KLINIK user receives the message PW: in response to the first character input on the remote terminal. If the KLINIK user types an incorrect password, the message ?IL followed by PW: is printed. The KLINIK user has three chances to type the correct password. After three failures, the console program hangs up the line. However, if the user enters the correct password, the message OK is printed.

In State 3, each character that the user types is sent directly to the running monitor, which causes the KLINIK link to behave as if it were a normal timesharing line. State 3 is helpful to Field Service personnel who wish to examine SYSERR files and crash dumps.

In State 4, the KLINIK link is considered a remote CTY. The KLINIK user can type any command on the remote CTY that the operator can type on the CTY. Input is taken from both the CTY and the remote CTY, as if it were typed from one terminal. For example, a SYSTAT can be done by typing S at the CTY, and Y followed by a carriage return at the remote CTY. Therefore, be careful not to type on the CTY while the KLINIK user is typing on the remote CTY.

To enter State 4, the KLINIK link must first be in State 3, and the KL command must be on (the operator has typed the KL 1 command). The KLINIK user types the correct password and then a CTRL/backslash. The KLINIK link is now in State 4.

Another way to enter State 4 is to turn the REMOTE DIAGNOSIS switch to the ENABLE position and turn the KL command on by typing KL 1. The KLINIK user needs to type only CTRL/backslash to enter State 4. No password is necessary. This method gives less security to the KLINIK link than the first method; however, it is an option for the time period when there is no operator coverage.

Table 7-1 presents the various states of the KLINIK link.

Table 7-1.
KLINIK States

Position of Switch	Operator Commands	KLINIK User	State
DISABLE	-	-	1
PROTECT	-	-	1
PROTECT	PW password	-	2
ENABLE	-	-	3
PROTECT	PW password KL 1	password CTRL/backslash	4
ENABLE	KL 1	CTRL/backslash	4

If you turn the REMOTE DIAGNOSIS switch to DISABLE while in any state, you switch the KLINIK link to State 1, unavailable.

You do not have to enter a password into memory with the PW command or turn the KL command on with KL 1 command every time it is necessary to enable the KLINIK link. They are retained in memory until one of the following occurs:

1. Power off/Power on
2. RESET switch is pressed
3. Type PW followed by a carriage return. This removes the password from memory. If this happens during States 2 or 4, the KLINIK link returns to State 1. Type KL Ø. Typing the KL Ø command changes the KLINIK link from State 4 to State 2.

The following commands to the console program allow you to enable or disable the KLINIK facility:

Command	Function
KL 1	Turns on the KL command so that State 3 can be entered.
KL Ø	Turns off the KL command so that State 3 is disabled.
KL	Gives the status of the KL command.
PW password	Allows you to enter a 1-6 character password into memory. The password remains in memory until you type PW followed by a carriage return, press power off/power on or press the RESET switch.

KS10 HALT STATUS CODES

When the KS1Ø enters a halt state, the microprocessor prints a halt status code on the console terminal. A halt status code appears in the form:

```
%HLTD/nnnnnn  PC/ØØØØØØ,,xxxxxx
```

where n = the halt status code (See Table 7-2.)

x = the program counter when the halt condition occurred. Note that the halt status code tells you why the KS1Ø processor decided to halt.

Table 7-2.
KS10 Halt Status Codes

Halt Code	Definition
000001	A halt instruction has been executed.
000002	The console program (microprocessor) halted the CPU. That is, an HA command was typed at the console.
000100	An I/O page failure occurred. Call your Field Service representative.
000101	An illegal interrupt instruction occurred. Call your Field Service representative.
000102	The pointer to the UNIBUS vector is zero. Call your Field Service representative.
001000 001004	An illegal microcode dispatch occurred. Call your Field Service representative.
001005	The microcode startup check failed. Call your Field Service representative.

Chapter 8

SYSTEM ADMINISTRATOR PROCEDURES

ACCOUNTING FILES

During installation, prototype versions of the accounting files ACCT.SYS, AUXACC.SYS and STRLST.SYS were transferred from the CUSP tape to SYS:. These basic accounting files contain entries for only a few PPN's such as [1,2] and [1,4]. Note that the [1,2] password as supplied by ACCT.SYS is FAILSA.

A continuing task of the system administrator is to update the accounting files (using the REACT program) so they reflect new assignments for project-programmer numbers, disk structure quotas, etc. The procedures for updating the accounting files for TOPS-1Ø on the 2Ø2Ø and TOPS-1Ø on the DECsystem-1Ø are identical.

SMFILE

The microprocessor file system is contained in the file KS1ØFE.BIN in [6,2Ø2Ø]. A microprocessor file system can be built on a disk structure using the SMFILE program, and the disk structure can then be used to boot the system.

The microprocessor file is pointed to by offset 1Ø1 of the home block; the length of the file is in offset 1Ø2 of the home block. Offset 1Ø3 of the home block also points to the microprocessor file, using a convenient format for the 8Ø8Ø processor called track/cylinder/sector format.

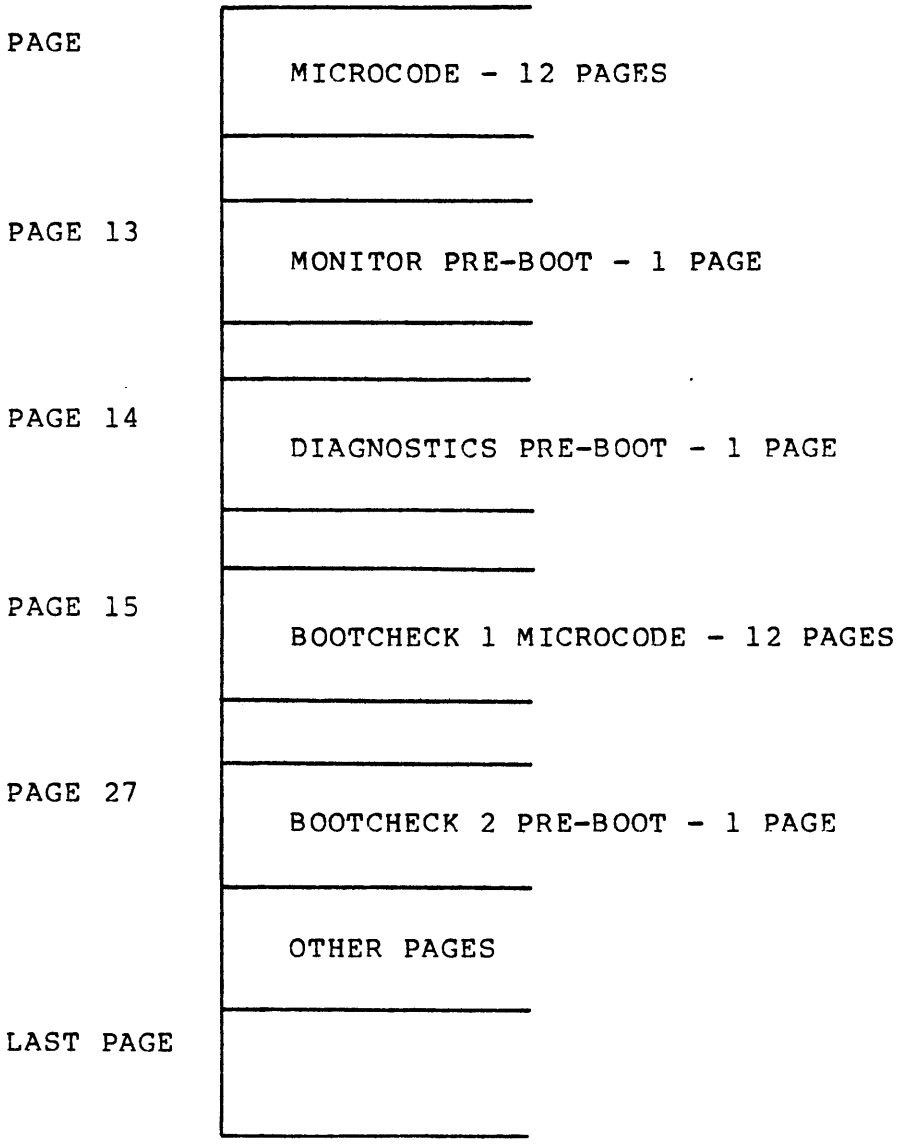
Page Ø of the microprocessor file functions as a directory and contains pointers to each file data area. File data in the microprocessor file consists of the microcode, preboot program, boot program and so on. Each file is pointed to by a two-word pair. The first word gives the file address in track/cylinder/sector format; the second word is the starting page number in the left half, and the length of the file in pages in the right half. The

format of the first page (page 0) is shown below.

WORD		
	0	POINTER TO FREE SPACE
	1	PAGE #,,LENGTH
	2	POINTER TO MICROCODE
	3	PAGE #,,LENGTH
	4	POINTER TO MONITOR PRE-BOOT
	5	PAGE #,,LENGTH
	6	POINTER TO DIAGNOSTIC PRE-BOOT
	7	PAGE #,,LENGTH
	10	POINTER TO BOOTCHECK 1 MICROCODE
	11	PAGE #,,LENGTH
	12	POINTER TO BOOTCHECK 2 PRE-BOOT
	13	PAGE #,,LENGTH
	14	POINTER TO MONITOR BOOT
	15	PAGE #,,LENGTH
	16	POINTER TO DIAGNOSTIC BOOT
	17	PAGE #,,LENGTH
	20	POINTER TO BOOTCHECK 2
	21	PAGE #,,LENGTH
	22	POINTER TO INDIRECT FILE 0
	23	PAGE #,,LENGTH
	24	POINTER TO INDIRECT FILE 1
	25	PAGE #,,LENGTH
		WORDS 26 THROUGH 775
	776	POINTER TO INDIRECT FILE 366(8)
	777	PAGE #,,LENGTH

Subsequent pages in the microprocessor file are as shown below. Free space starts at page 28, where BOOTS and other

files are assigned space as appropriate. Note that many of the files (e.g., diagnostic preboot, indirect files) are not present in the installed version of the microprocessor file, since special commands (normally used only by Field Service) to SMFILE are needed to put them there.



For information about SMFILE commands and the microprocessor file, obtain a copy of SMFILE.TXT from the [10,7] area.

CREATING A MONITOR BACKUP TAPE

After installing TOPS-10 on the 2020 and creating a system-specific monitor, you should create a bootstrap magtape for the system. This tape can be used to get the monitor running in the event of disk failure. It also saves the newly-created monitor in case the disk copy is accidently destroyed.

During normal system operation, you should periodically create a backup tape, so the system can be easily re-installed if disk failure occurs. A monitor backup tape basically contains the same files as the installation tape, except that these files are tailored to the specific system.

To create a monitor backup tape, log in as [1,2] and run SETSRC to allow access to files in the [10,7] area in DSKB:. Then, files such as the microcode, magtape bootstrap program, BACKUP program and the monitor are placed on the tape. The procedure is as follows:

```
.R SETSRC
*A DSKB:
*CP [10,7]              ;access files in DSKB: [10,7]
*^C
.
```

Load a scratch tape on MTA0:. Then type

```
.ASSIGN MTA0: BACKUP
.SET BLOCKSIZE MTA0: 512
.COPY MTA0: = KS10.RAM
.COPY MTA0: = BOOTM.RDI
.COPY MTA0: = SYS:BACKUP.EXE
.R BACKUP
/TAPE MTA0:
/SAVE DSKB:[1,4] = SYS:SYSTEM.EXE
/UNLOAD
/EXIT
.
```

The SAVE command to BACKUP saves the monitor as DSKB:SYSTEM.EXE[1,4], the default area for loading by BOOTM. As an option, an additional saveset can be placed on the tape containing all files in SYS:. This is not necessary, since these files are available from the regular system backup tapes.

KS10.RAM is the microcode file and BOOTM.RDI the BOOTM program in tape read-in format. Both these files are supplied on the installation tape and placed in [10,7]. They can also be created from the file KS10.ULD and BOOTM.MAC as follows:

```
.R MACRO
*BOOTM=TTY:,DSK:BOOTM          ;compile BOOTM.MAC
*MAGRIM==0
*^Z
*Z
.LOAD BOOTM
.SAVE BOOTM                    ;creates BOOTM.EXE
.RUN SMFILE
SMFILE>READ KS10.ULD
SMFILE>OUTPUT CRAM KS10.RAM    ;create KS10.RAM
SMFILE>MTBOOT BOOTM.EXE BOOTM.RDI
SMFILE>EXIT                    ;create BOOTM.RDI
.
```

Chapter 9

PERFORMANCE OF TOPS-10 ON THE 2020

AVAILABLE MEASUREMENTS - IOTEST

Adequate performance measurements for TOPS-10 on the 2020 do not exist at present. What does exist are some comparison measurements, using the IOTEST program for TOPS-10 on the 2020 and 1091, and TOPS-20 on the 2020/40/50.

The IOTEST program performs a test of disk I/O, which consists of three parts: writing a file, reading it sequentially and reading it randomly. The random reads are done by generating a uniform random number to use as a page number; a random read is done for each page in the file. A table gives the sizes of the files to use. The program starts with files of size 10 pages, then 50, 100, 250, 500, 1000, 2500, 5000, and 10000 pages. After processing a file of a specified size, the file is deleted and the next file selected. The elapsed and CPU times are reported for each phase of file processing. The results, given in the next section, show that it takes a lot more CPU and elapsed time to perform I/O on the 2020 than on other hardware.

IOTEST RESULTS

IOTEST results are shown in Figures 9-1, 9-2 and 9-3. Figure 9-1 gives a comparison for TOPS-10 and TOPS-20 running on 1091 (DECSYSTEM-2050) hardware. Figure 9-2 is a comparison of TOPS-20 running on 2020, 2040 and 2050 hardware. Figure 9-3 gives the results for TOPS-10 running on 2020 hardware. All results were obtained under stand-alone conditions.

For the results in Figure 9-1, the PMAP JSYS was used to write and read the file under TOPS-20, while dump mode I/O was used doing four blocks (one page) at a time under TOPS-10. All tests in Figure 9-1 were done stand-alone on a clean RP06 disk pack that was on a separate channel from the

system structure. Also, the backup register was not used for TOPS-20, because the proper ECO had not been installed.

	TOPS-10	TOPS-20
100 Pages:		
Write	24.16	8.44
Read	21.00	20.06
Random Read	18.67	17.95
500 Pages:		
Write	23.03	7.31
Read	21.17	20.18
Random Read	21.13	21.17
1000 Pages:		
Write	23.00	7.51
Read	21.18	20.16
Random Read	22.85	22.99
5000 Pages:		
Write	22.92	7.50
Read	21.12	20.60
Random Read	26.82	27.73
10,000 Pages:		
Write	22.86	7.49
Read	21.12	20.16
Random Read	53.97	44.12
50,000 Pages:		
Write	23.03	7.51
Read	21.24	20.17
Random Read	70.13	40.06

Figure 9-1.
DECsystem/1091 - DECSYSTEM-2050 IOTEST

NOTE: Numbers are given as milliseconds per page (4 blocks under TOPS-10). All times are elapsed times.

	2020	2040	2050
100 Pages:			
Write CPU	2.26	1.10	0.54
Write Elapsed	2.78	1.33	0.84
Pages/Second	35.90	75.20	119.00
Read CPU	1.46	0.72	0.39
Read Elapsed	3.02	2.08	2.01
Pages/Second	33.10	48.10	49.50
Random Read CPU	1.48	0.72	0.39
Random Read Elaped	3.31	2.42	1.79
Pages/Second	30.20	41.30	55.90
1000 Pages:			
Write CPU	21.63	10.56	5.56
Write Elapsed	25.03	12.11	7.51
Pages/Second	39.90	82.60	133.20
Read CPU	14.90	7.35	3.91
Read Elapsed	28.38	20.28	20.16
Pages/Second	35.20	49.30	49.60
Random Read CPU	15.89	8.60	4.18
Random Read Elapsed	40.48	29.64	22.99
Pages/Second	24.70	33.70	43.50

Figure 9-2a.
TOPS-20 on 2020,2040,2050 - IOTEST

	2020	2040	2050
10000 Pages:			
Write CPU	214.56	105.20	55.44
Write Elapsed	247.01	121.03	74.96
Pages/Second	40.50	82.60	133.40
Read CPU	149.94	74.22	39.49
Read Elapsed	292.97	203.28	201.65
Pages/Second	34.10	49.20	49.60
Random Read CPU	184.43	101.26	46.28
Random Read Elapsed	522.64	376.25	304.67
Pages/Second	19.10	26.60	32.80

Figure 9-2b.
TOPS-20 on 2020,2040,2050 - IOTEST

NOTE: Read and write times should be multiplied by 10 to give units of milliseconds per page.

10 Pages:		
Write file	CPU time: 187	Elapsed time: 450
Read file	CPU time: 128	Elapsed time: 267
Random read	CPU time: 140	Elapsed time: 300
10 Pages:		
Write file	CPU time: 140	Elapsed time: 417
Read file	CPU time: 107	Elapsed time: 267
Random read	CPU time: 127	Elapsed time: 316
10 Pages:		
Write file	CPU time: 179	Elapsed time: 433
Read dile	CPU time: 125	Elapsed time: 283
Random read	CPU time: 138	Elapsed time: 333
50 Pages:		
Write file	CPU time: 760	Elapsed time: 1683
Read file	CPU time: 584	Elapsed time: 1283
Random read	CPU time: 625	Elapsed time: 1583
50 Pages:		
Write file	CPU time: 672	Elapsed time: 1533
Read file	CPU time: 566	Elapsed time: 1317
Random read	CPU time: 689	Elapsed time: 1667
50 Pages:		
Write file	CPU time: 686	Elapsed time: 1567
Read file	CPU time: 565	Elapsed time: 1317
Random read	CPU time: 733	Elapsed time: 1733
100 Pages:		
Write file	CPU time: 1292	Elapsed time: 2917
Read file	CPU time: 1182	Elapsed time: 2667
Random read	CPU time: 1281	Elapsed time: 3517
250 Pages:		
Write file	CPU time: 3158	Elapsed time: 6967
Read file	CPU time: 2881	Elapsed time: 6500
Random read	CPU time: 2891	Elapsed time: 8700

Figure 9-3a.
TOPS-10 on 2020 - IOTEST

500 Pages:		
Write file	CPU time: 6255	Elapsed time: 13566
Read file	CPU time: 5819	Elapsed time: 13116
Random read	CPU time: 5622	Elapsed time: 19050
1000 Pages:		
Write file	CPU time: 12517	Elapsed time: 27083
Read file	CPU time: 11611	Elapsed time: 25800
Random read	CPU time: 10456	Elapsed time: 38950
2500 Pages:		
Write file	CPU time: 30940	Elapsed time: 66600
Read file	CPU time: 29069	Elapsed time: 64766
Random read	CPU time: 26564	Elapsed time: 103750
5000 Pages:		
Write file	CPU time: 61650	Elapsed time: 132850
Read file	CPU time: 58484	Elapsed time: 130017
Random read	CPU time: 51941	Elapsed time: 216367
10000 Pages:		
Write file	CPU time: 123126	Elapsed time: 264950
Read file	CPU time: 116790	Elapsed time: 259484
Random read	CPU time: 105130	Elapsed time: 461317
25000 Pages:		
Write file	CPU time: 309193	Elapsed time: 664683
Read file	CPU time: 293782	Elapsed time: 651634
Random read	CPU time: 353153	Elapsed time: 1845016
50000 Pages:		
Write file	CPU time: 619927	Elapsed time: 1331467
Read file	CPU time: 590369	Elapsed time: 1305483

Figure 9-3b.

TOPS-10 on 2020 - IOTEST

NOTE: Times are in milliseconds for the stated number of pages. Divide by the number of pages to obtain units of milliseconds per page.

Chapter 10
PRESALES INFORMATION

The DECSYSTEM-2020 is the least expensive mainframe computer system on the market today. It extends both the DECsystem-10 and DECSYSTEM-20 families of computers at the low-end. It is a general purpose computer system offering concurrent interactive timesharing and multi-stream batch. It has a minimum configuration price tag of $191,300, which includes 256K words of MOS memory, one RM03, one TU45, 16 asynchronous lines, an LA36 console and the TOPS-10 System Software package (including GALAXY).

The price tag (including tape drive) of a DECSYSTEM-2040 begins at $386,000, a DECSYSTEM-2060 at $511,500, and a minimum 1090 system package is priced at $701,800. The DECSYSTEM-2020 therefore provides considerable savings, while delivering the capabilities and features of the TOPS-10 (or TOPS-20) operating systems.

The DECSYSTEM-2020 provides an attractive opportunity for emerging markets, since the customer can later upgrade to a larger system while protecting the previous software investment. Traditional markets served by DEC's mainframes include education, government, engineering, scientific and commercial areas. The existing TOPS-10 customer base and OEMs should provide a volume market place. A key selling point is the upward compatibility of the 2020's software products. This establishes a growth pattern that allows OEMs a significant sales advantage over competition such as Hewlett-Packard. HP's OEMs have no upward growth plan with the HP3000.

The Educational Computer Group (ECG) marketing strategy will include installed base customers (DECsystem-10, KA, KI, KL) who find the 2020 system provides an entry solution for offsetting the current load. In colleges and universities, the 2020 is suited for the single purpose needs of a department, or small user community that requires mainframe versatility in software and applications. This also applies to the school district.

The primary market place for the Engineering Systems Group (ESG) will be engineering consultants, architectural and construction companies. These usually require a variety

of languages and applications, rather than strong performance in a particular area. The targets for the Federal Systems Group (FSG) include installed base customers who require incremental computing without the expense of an additional large system. Aerospace and subcontractors (OEM)) markets are dominated by TOPS-10. Hence, TOPS-10 on the 2020 should provide DECsystem-10 OEM users with powerful software and protect existing applications and languages developed over the years.

Market competition for the 2020 includes:

- IBM/3
- IBM 370/115
- HP3000/II
- Prime 400/500
- UNIVAC 90/30

Hewlett-Packard is the major competitor for the volume markets. Prime should also be considered, because of their "fine-tuned" COBOL and data base management capabilities. As a low-end mainframe, the DECSYSTEM-2020 offers more capability than IBM can offer at the price.

The DECSYSTEM-2020 should be offered when:

1. Price is critical.

2. Mainframe software capability is required.

3. A small computer system is required to fill the current need (entry level) or to provide incremental computing power for an installed base customer.

4. There is a requirement for COBOL, DBMS, IQL or APL.

5. Compatibility with existing or future TOPS-10/TOPS-20 systems is desired (for present or future growth assurance).

6. An installed PDP-11 account needs an upgrade path for more capabilities than VAX 11/780 can offer; or a replacement system is needed for IBM 1401 and low-end 360 system-type hardware.

Table 10-1 provides competitive data for the DEC 2020, IBM S/3, IBM 370/115, HP3000II, Prime 500 and UNIVAC 90/30.

Possible future improvements to the 2020 include:

1. Full ANF communications with TOPS-10 7.01.
2. IBM style remote station communication.
3. Multiple line printers (two LP20's).

Table 10-1.
2 3020 Competitive Data

	DEC 2020	IBM S/3	IBM 370/115	HP 3000II	PRIME 500	UNIVAC 90/30
CPU	-	-	-	-	-	-
Memory	256K wds	128KB	128KB	320KB	256KB	128KB
Disk	140 MB	100 MB	140 MB	100 MB	160 MB	120 MB
CR	300 CPM	200 CPM	200 CPM	600 CPM	300 CPM	160 CPM
Line Printer	300 LPM	600 LPM	600 LPM	600 LPM	600 LPM	500 LPM
Tape	Yes	No	Yes	Yes	Yes	Yes
SOFTWARE						
COBOL /SORT	$ 11,200	**R/$100- mo.	R/$141- mo.	B	Pkg. for	
OP.SYSTEM**	B	B	B	B	all 3 @	B
DBMS	$ 27,000	R/$115/mo	R/$115/mo	B	$25,000	
TOTAL PURCHASE	$252,170*	$252,845 plus $215/mo	$280,600 $256/mo	$233,625	$246,500	$237,328

* 2020 peripheral and software prices are significantly higher than the competitive systems above. Therefore, as system requirements increase, the 2020 becomes less and less price competitive.

** R = Rental, B = Bundled

Appendix A

NEW DOCUMENTATION

For the 6.03A LIR (Limited Interim Release) of TOPS-10 on the DECSYSTEM-2020, the following documentation is available:

1. DECSYSTEM-2020 TOPS-10 Monitor Installation Guide

2. DECSYSTEM-2020 TOPS-10 Operator's Guide. Note that a list (incomplete) of 8080 microprocessor console commands and and their functions is included in this document.

3. BWR6KS.MEM - This is the Beware File for TOPS-10 on the 2020, and is supplied on the Installation Tape in a BACKUP saveset.

4. SMFILE.TXT - This provides documentation for SMFILE, and is supplied on the Installation Tape. There is also a smaller file, SMFILE.HLP.

This page is for notes.

INDEX