KT8-A memory management control user's guide

EK-KT08A-UG-001

digital equipment corporation • maynard, massachusetts

1st Edition, July 1978

The material in this manual is for informational purposes and is subject to change without notice.

Digital Equipment Corporation assumes no responsibility for any errors which may appear in this manual.

Printed in U.S.A.

This document was set on DIGITAL's DECset-8000 computerized typesetting system.

CONTENTS

FIGURES

TABLES

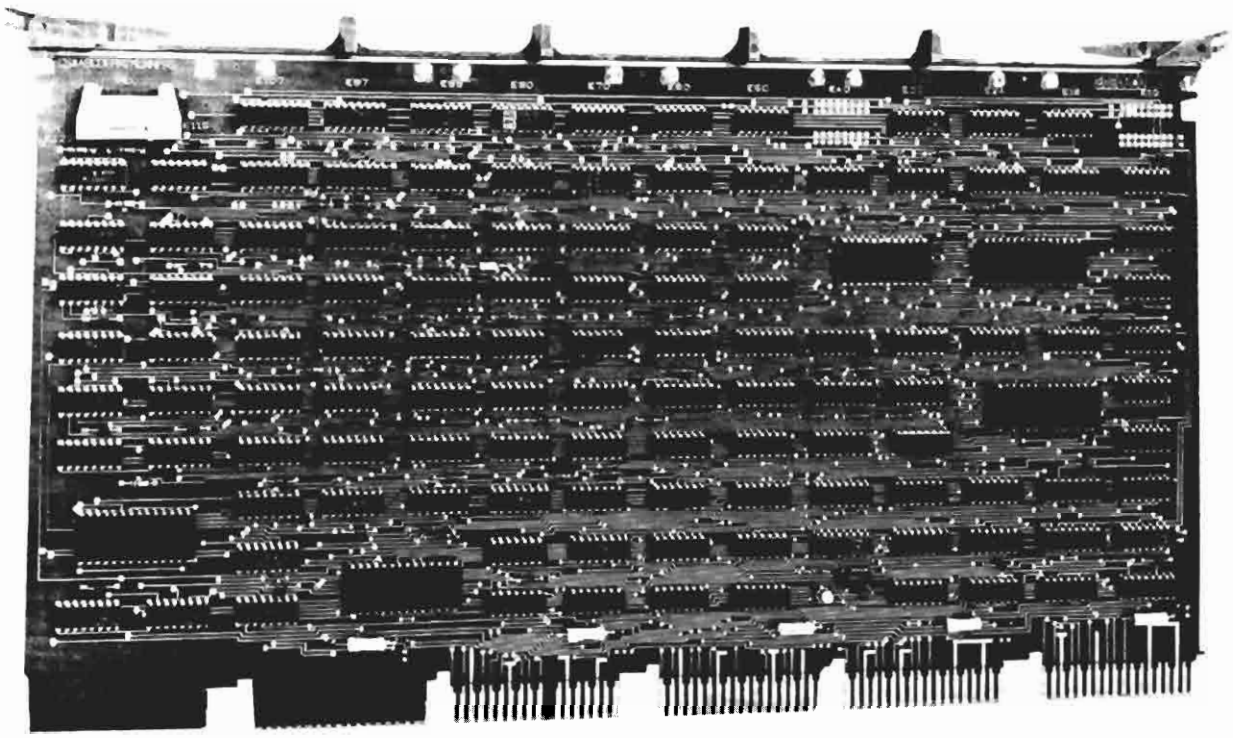
CHAPTER 1
INTRODUCTION

1.1 SCOPE OF MANUAL

The KT8-A is an Omnibus option that provides memory extension (up to 128K words) and memory management capability for PDP-8/A systems. This manual describes the KT8-A, tells how to install it in a system, and gives detailed operating and programming instructions. Paragraph 1.4 lists additional documents helpful to the KT8-A user.

1.2 GENERAL DESCRIPTION

The logic and circuit components that comprise the KT8-A are mounted on a single hex-size printed circuit board (M8416 module). The module can be inserted in the Omnibus of all PDP-8/A computers, except the 8A100. When the KT8-A is installed in a two-box system (any combination of H9300 and BA8-C boxes) containing memory in both boxes, a terminator module (M9020) and an interconnecting cable must be used to achieve bank select signal continuity between boxes. The M8416 is pictured in Figure 1-1, while the M9020 and the cable that connects the two modules are shown in Figure 1-2.



9102-2-A0187

Figure 1-1 KT8-A (M8416) Module

9102-7-A0188

Figure 1-2 M9020 Terminator and Cable

The basic function of the KT8-A is to furnish PDP-8/A memory extension, i.e., to supply the five highest-order address bits for PDP-8/A memory. Memory addresses are generated by the CPU and by direct memory access (DMA) peripherals. However, the CPU can provide only a 12-bit address (bits MA0–MA11) and DMA devices are limited to 15-bit addresses (bits MA0–MA11 and EMA0–EMA2); consequently, to achieve the full potential of 128K word memory address space, the KT8-A adds five bits to a CPU-generated address and two bits to a DMA-generated address.

Earlier PDP-8 memory extension controls, the KM8-E and the KM8-A, are capable of supplying only 15 bits of memory address (up to 32K words). However, programs that were written for these options are compatible with the KT8-A. Furthermore, the KM8-AC can be used with the KT8-A to provide bootstrap and/or power fail/auto-restart capabilities (the memory extension and time-share feature of the KM8-AC must be disabled in this application).

In addition to the memory extension function, the KT8-A performs memory management, which provides two important advantages for an operating system. First, memory management allows an operating system to monitor the I/O operations and memory access of a program that is under the system's control. Second, memory management enables an operating system to execute a program originally constructed for exclusive use of the computer.

Unlike memory management carried out by the KM8-E, wherein the operating system must perform certain calculations each time the running program tries to access a different 4K memory field, the KT8-A is set up by its operating system to perform these calculations automatically. Thus, the KT8-A requires a less complex operating system than the KM8-E and results in faster execution of the running program.

1.3 KT8-A SPECIFICATIONS

Table 1-1 lists the significant specifications of KT8-A systems.

Table 1-1 KT8-A Specifications

Item	Specification
CPU Memory Address Space	Programmable – 32K (2^{15}), 64K (2^{16}), or 128K (2^{17}) words
DMA Memory Address Space	Programmable – 32K or 128K words
Addressing Space Accessible without KT8-A Intervention	4K Segments (2^{12})
Memory Compatibility	Total memory less than or equal to 32K – Any PDP-8/A memory type can be used
	Total memory greater than 32K – Only MM8-AB, MS8-CA, MS8-CB memories can be used.
Enclosure Compatibility	KT8-A can be used in all BA8-C and H9300 boxes†
Option Compatibility	All PDP-8/A options are compatible, except KM8-AA and KM8-AB
Software Compatibility	All programs written for PDP-8/A and PDP-8/E are compatible, except instruction 6200 (LXM)
Device Codes	Permanent: 20–27
	Program Enabled: 17 and 30–37
Mode Control	Foreground/background 32K/64K/128K KM8-A/KM8-E or Extended Instruction Set Maintenance mode
Background Program Options (Programmable)	Dynamic Memory Relocation (4K increments) Memory Protection (4K boundaries) I/O Instruction Trap
Power Consumption	KT8-AA: +5 V @3.7 A KT8-AB: +5 V @5.7 A KT8-EX: +20 V @0.1 A
Environmental Requirements	
Operating Temperature* Operating Humidity	5° C–50° C (41° F–122° F) 10%–95%, with a maximum wet bulb temperature of 32° C (90° F) and a minimum dew point of 2° C (36° F)

* The maximum allowable operating temperature is based on operation at sea level, i.e., at 760 mm Hg (29.92 inches Hg); maximum allowable operating temperature will be reduced by a factor of 1.8° C/1000 m (1.0° F/1000 ft) for operation at higher altitude sites.

† H9194 backplanes in the H9300 box must have a sticker (P/N 3615653-00) reading "KT8-A COMPATIBLE" affixed to the component side of the backplane; if the sticker is not present, the backplane must be modified according to DEC ECO H9194-00003.

1.4 RELATED DOCUMENTS

The following documents contain information of interest to the KT8-A user.

Document Title	Document Number	Remarks
PDP-8/A Miniprocessor User's Manual	EK-8A002-MM-002	Available in hard copy.
PDP-8/A Minicomputer Handbook	EB-06219-76	Available in hard copy.
PDP-8 Family Configuration Guide	EK-0PDP8-SP-001	Available in hard copy.
KT8-A Technical Manual	EK-KT08A-TM-001	In Microfiche library; available in hard copy.
MS8-C MOS Memory Operation and Maintenance Manual	EK-MS8C-TM001	In Microfiche library; available in hard copy.

Hard copy documents can be ordered from:

Digital Equipment Corporation
444 Whitney St.
Northboro, MA 01532

ATTN: Communication Services (NR2/M15)
Customer Services Section

For information concerning microfiche library, contact:

Digital Equipment Corporation
132 Parker St.
Maynard, MA 01754

ATTN: Micropublishing Group
PK3-2/T12

1.5 SOFTWARE

1.5.1 Diagnostic

The KT8-A uses the following diagnostic software:

Program Title	Program Number
KT8-A Memory Management Diagnostic	MAINDEC-08-DJKTA-A
Extended Address Test	MAINDEC-08-DHKMC-C
Extended Memory Data and Checkerboard Test	MAINDEC-08-DHKMA-D

DEC/X8 Monitor, Version 2, and all support modules appropriate to Version 2.

1.5.2 System

OS/8, Version 3D/128K, support will include greater-than-32K functions of GET, SAVE, ODT, and RUN. Verson 2 of MACREL/LINKER will support greater-than-32K assemblies and linkages. RTS8, Version 3, is capable of using all available memory and supports up to 32K OS/8 background task. Existing OS/8 and assembly language programs can work with the KT8-A option installed, but will not fully utilize the KT8-A features. Programs can be written under PAL8 to make use of KT8-A extended functions. Early versions of OS/8 utilities do not support extended memory.

CHAPTER 2
INSTALLATION

2.1 GENERAL

There are three types of KT8 options. The KT8-AA consists of the M8416 module alone; this type is found in a system configuration that is arranged and shipped by DIGITAL. The KT8-AB consists of an M8416 module and a KM8-AC option (memory extension and timeshare, power fail/auto-restart, and bootstrap); this type is intended to be an addition to an existing PDP-8/A system. The KT8-EX consists of an M9020 terminator module and a cable that connects the M9020 to the KT8-A; this type is required in a 2-box system when memory is to occupy space in both boxes.

The KT8 can be installed in all PDP-8/A computers, although it cannot be used in systems that include a PDP-8/E or PDP-8/M. Thus, there are a number of possible 1-box and 2-box configurations involving the KT8. Table 2-1 summarizes the possible 1-box configurations, while Figure 2-1 illustrates the possible 2-box configurations. Note that an 8A600 computer can be expanded only with a BA8-C box, while the 8A620 or 8A625 computers can be expanded with both a BA8-C box and an H9300 box.

Table 2-1 KT8-A Single-Box Configurations

PDP-8/A Computer	CPU Type	Chassis Type
8A205 8A400 8A405	KK8-A	H9300 (12-slot)
8A600	KK8-F	
8A420 8A425	KK8-A	BA8-C (20-slot)
8A620 8A625	KK8-F	

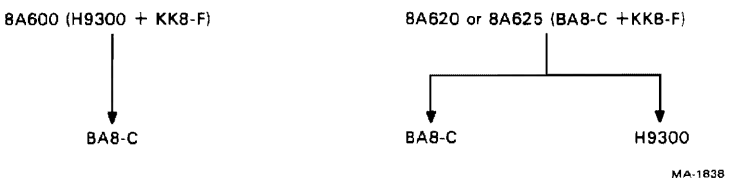


Figure 2-1 KT8-A Double-Box Configurations

2.2 SINGLE-BOX CONFIGURATIONS

In a single-box configuration the KT8-A can be inserted in any Omnibus slot that has an 'E' connector. Practically, the module should be placed in slot 4, 8, or 11, depending on the box type and the CPU type. For example, Figure 2-2 lists the Omnibus slots and their contents for a repesentative 8A600 system containing a KT8-A. The KT8-A can be inserted in any of slots 4 through 8. However, memory should be placed as far away from the CPU as possible (refer to the *PDP-8 Family Configuration Guide*); hence, the most practicable location for the KT8-A is in slot 8. (DMA modules must always be placed between memory and the CPU; in this context the KT8-A is not considered to be a memory module; hence, DMA modules can be placed between the KT8-A and memory.) Figure 2-3 represents a different configuration. Here, the KT8-A occupies slot 4 again so that there will be maximum separation of CPU and memory.

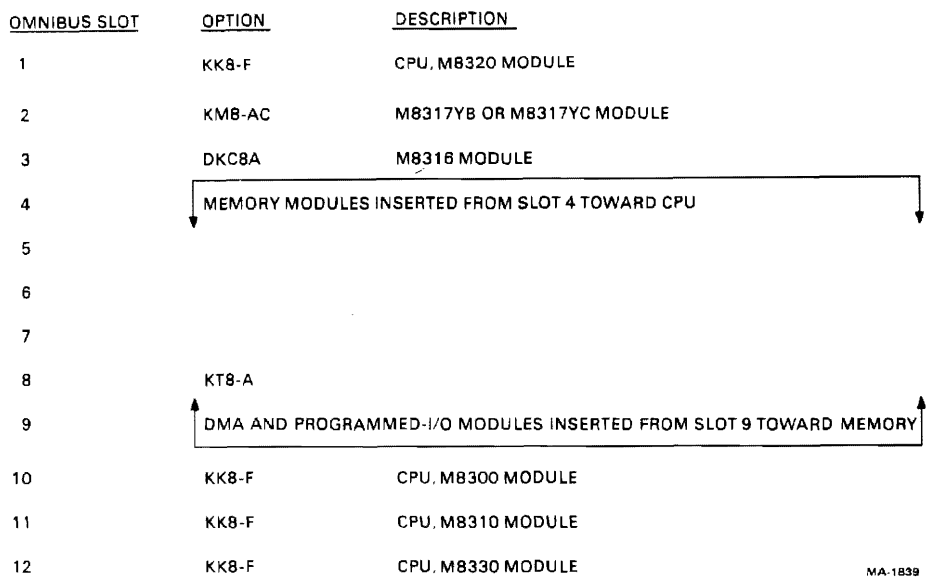


Figure 2-2 Representative KT8-A System Configuration, 8A600 Computer

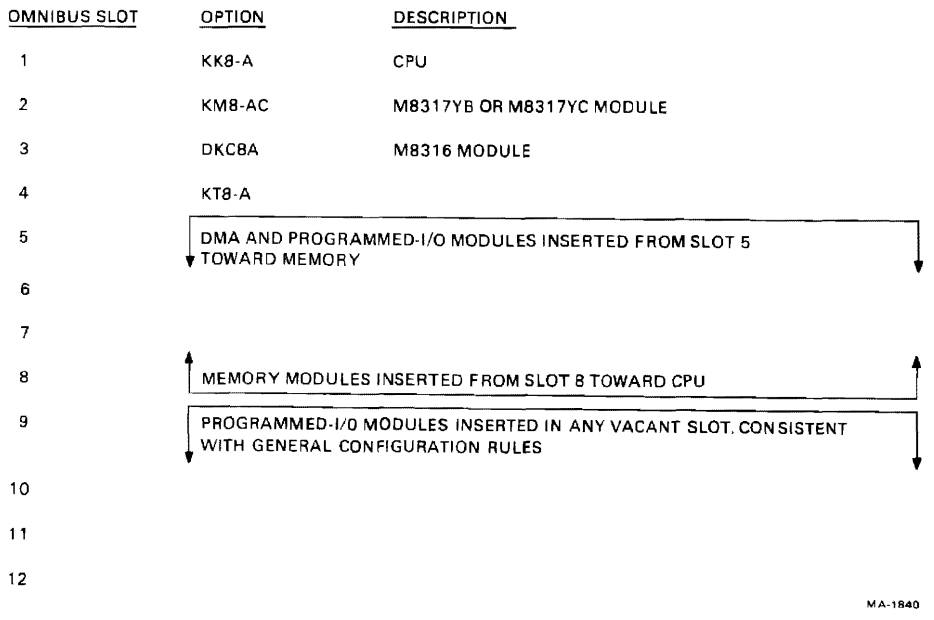


Figure 2-3 Representative KT8-A System Configuration, 8A400 Computer

2.3 DOUBLE-BOX CONFIGURATIONS

A 2-box system can be comprised of two BA8-C boxes or one BA8-C and one H9300 box (an H9300 box cannot be used to expand another H9300 box). In either combination the KK8-F CPU must be used.

Figure 2-4 shows a representative 8A600 system expanded with a BA8-C box. Memory is contained in both boxes; hence, the M9020 terminator must be inserted in an 'E' connector of the BA8-C so that bank select signals generated by the KT8-A can be supplied to all memories (a quad-size programmed-I/O module could be inserted in the same Omnibus slot occupied by the M9020). The M9020 (and the KT8-A in the 8A600) can be inserted in any slot having an E connector; however, its placement in slot 11, assuming slots 4 through 10 are occupied by memory, means that memory will be as far away from the CPU as is possible.

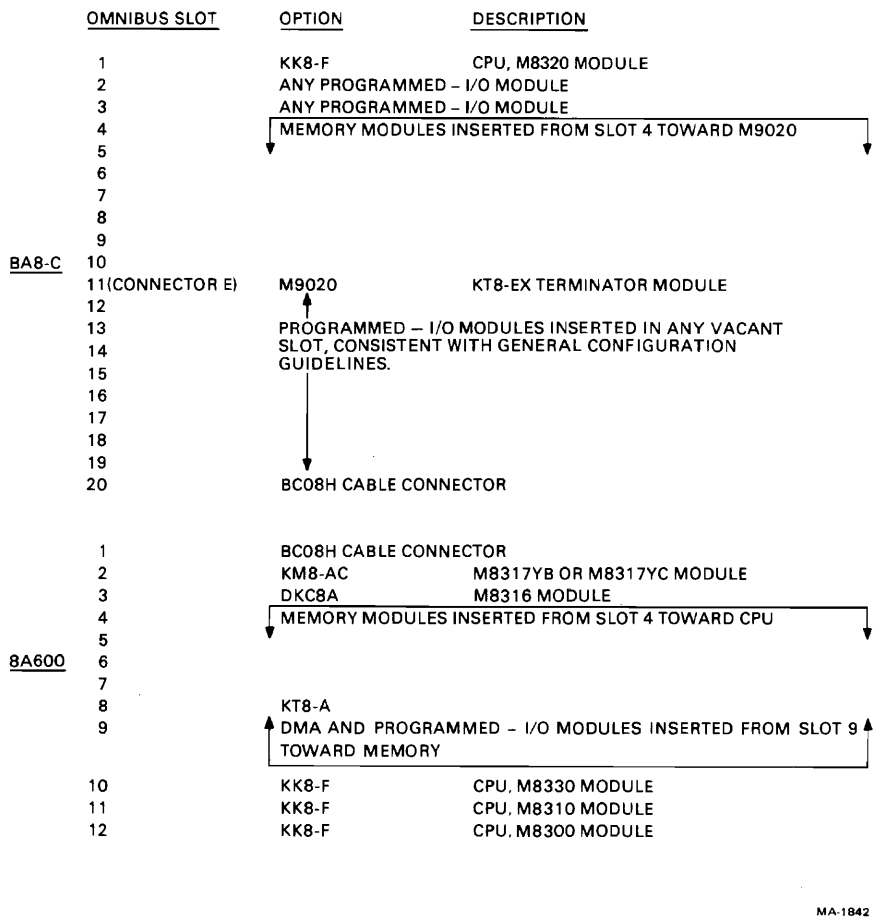


Figure 2-4 Representative KT8-A System Configuration, 8A600 Computer with BA8-C Expander

If memory were to be contained in only one box, it would be installed in the expander. In this case, the KT8-A would be moved to slot 11 of the expander (if memory occupied slots 4 through 10), or to slot 9 (if memory occupied only slots 4 through 8). This latter situation is illustrated in Figure 2-5.

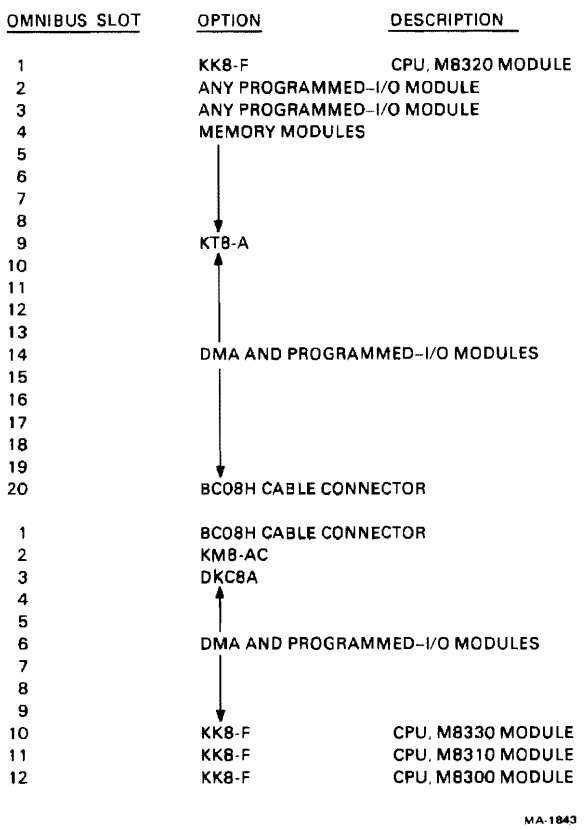


Figure 2-5 Representative KT8-A System Configuration, Single-Box Memory

2.4 MEMORY MODULE INSTALLATION

The KT8 can be used with any combination of MM8-AB memory (16K core), MS8-CA memory (16K MOS), and MS8-CB memory (32K MOS). Each memory module is assigned a field of bus addresses before being inserted in the Omnibus. The field assignment is made by switches on both the MS8-CA and MS8-CB modules. Tables 2-2 and 2-3 show how the switches are set for these MOS memories (refer to the *MS8-C MOS Memory Operation and Maintenance Manual*, EK-MS8C-TM-001, for more information).

The field assignments for the MM8-AB memory require modification to the module itself. A wire must be added between two connector fingers to make a bank assignment and a jumper must be connected between two terminal posts to make a field assignment (the terminal post locations are shown in the PDP8/A Miniprocessor User's Manual). Table 2-4 indicates the wire and jumpers that must be added to achieve the desired field assignment (refer to DEC ECO MM8-AB, Number 7 for complete instructions concerning modification).

Table 2-2 MS8-CA Switch Settings for Field Assignments

Assigned Bank and Field		
Bank	Field	Switch Off (All Others On)
0	0–3 (0–16K)	S1-1
	4–7 (16–32K)	S1-2
1	0–3 (32–48K)	S1-3
	4–7 (48–64K)	S1-4
2	0–3 (64–80K)	S1-5
	4–7 (80–96K)	S1-6
3	0–3 (96–112K)	S1-7
	4–7 (112–128K)	S1-8

Table 2-3 MS8-CB Switch Settings for Field Assignments

Assigned Bank and Field		
Bank	Field	Switch Off (All Others On)
0	0–7 (0–32K)	S1-1, S1-2
1	0–7 (32–64K)	S1-3, S1-4
2	0–7 (64–96K)	S1-5, S1-6
3	0–7 (96–128K)	S1-7, S1-8

Table 2-4 MM8-AB Modifications for Field Assignments

Assigned Bank and Field			
Bank	Field	Connect These Fingers With a Wire*	Connect a Jumper at These Terminal Post Locations (Other Locations Blank)
0	0–3 (0–16K)	AB1 – EB2	1-2, 1-3
	4–7 (16–32K)	AB1 – EB2	1-2, 2-4
1	0–3 (32–48K)	AB1 – ED2	3-4, 1-3
	4–7 (48–64K)	AB1 – ED2	3-4, 2-4
2	0–3 (64–80K)	AB1 – EL2	3-4, 1-3
	4–7 (80–96K)	AB1 – EL2	3-4, 2-4
3	0–3 (96–112K)	AB1 – ER2	3-4, 1-3
	4–7 (112–128K)	AB1 – ER2	3-4 2-4

* There is a feed-through connector on the printed circuit board just above pin AB1. One end of the wire can be soldered into this feed-through; the other end of the wire must be soldered to the pin itself.

2.5 KM8-AC MODULE INSTALLATION

In cases where the KT8 is being added to an existing system, the KM8-AC option (memory extension and timeshare, power fail/auto-restart, and bootstrap) may be present. The KM8-AC can continue to be used with the KT8 to provide power fail/auto-restart and/or bootstrap capability. However, the memory extension and timeshare portion of the KM8-AC option must be disabled. This is done by arranging jumpers W1 through W4 on the KM8-AC module (M8317-YB or M8317-YC) as indicated in Table 2-5 (the *PDP-8/A Miniprocessor User's Manual* shows the location of the jumpers).

Table 2-5 KM8-AC Jumper Configuration for Disabling Memory Extension and Time-Share

Jumper Location	*Jumper
W1	OUT
W2	IN
W3	IN
W4	IN

* When the KM8-AC is used with the KT8-A, this jumper arrangement *must* be employed.

2.6 VERIFICATION

When the various modules have been configured properly and installed in the backplane, verify system operation by running the KT8-related software diagnostics.

1. Load and run the KT8-A Memory Management Diagnostic as described in the diagnostic description for five minutes; there must be no errors.

2. Load and run the Extended Address Test for one pass; there must be no errors.

3. Load and run the Extended Memory Data and Checkerboard Test for one pass; there must be no errors.

If a problem occurs during verification, contact the DIGITAL Field Service organization.

CHAPTER 3
OPERATION AND PROGRAMMING

3.1 OPERATION

3.1.1 Power-up Conditions

When the KT8-A is turned on, these registers and flip-flops are cleared:

Instruction Field Register
Instruction Bank Register
Data Field Register
Data Bank Register
Instruction Field Buffer
Instruction Bank Buffer
Interrupt Inhibit flip-flop
User Mode flip-flop
User Interrupt flip-flop
User Size Register
Relocation Register
Extended Mode Register
Last Break Register

The contents of these elements are undefined:

Save Register
Break Map

3.1.2 PDP-8/A Programmer's Console Functions

Both the instruction field (identified by the instruction bank register and the instruction field register) and the data field (identified by the data bank register and the data field register) can be set from the programmer's console. To do so, enter the four octal digits on the console key pad (see Figure 3-1). As each octal digit is entered, it appears in the DISP readout on the console (refer to the PDP-8/A Miniprocessor User's Manual for console operating instructions). When all four digits are entered, press the LXA button. This loads the instruction field and data field, clears the user mode buffer, the user mode flip-flop, the relocation register, the user size register, and the extended mode register, and puts the KT8-A in the KM8-A/KM8-E compatible mode.

The contents of the four registers can be checked by pressing the BUS button and then the DISP button. The console BUS indicator lights and the contents of the PDP-8/A DATA bus are displayed in the DISP readout. The readout is interpreted as illustrated in Figure 3-2.

When the console INIT button is pressed (or the CAF instruction is issued), the user interrupt flag, the last break register, the maintenance register, the interrupt inhibit flip-flop, and the fatal flip-flop are cleared.

When the console LXA button is pressed, the user size register, the relocation register, the extended mode register, the user mode flip-flop, and the interrupt inhibit flip-flop are cleared.

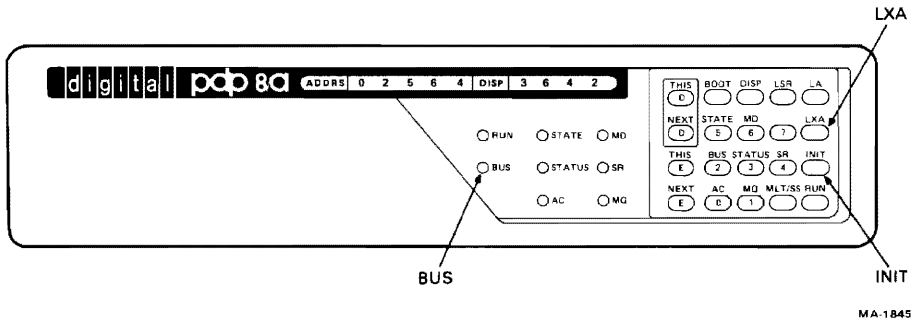


Figure 3-1 KT8-A Related Programmer's Console Functions

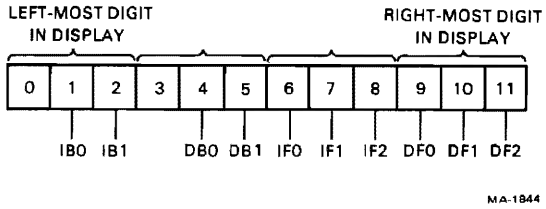


Figure 3-2 Bus Display Bit Assignment

3.2 PROGRAMMING

3.2.1 KT8-A Instructions

The KT8-A logic includes registers and control flip-flops that are described in Table 3-1. Figures 3-3, 3-4, and 3-5; and Paragraph 3.2.2., illustrate how these registers and flip-flops are used.

There are two sets of instructions for the KT8-A. One set, that of KM8-A/KM8-E compatible instructions, is appropriate when KM8-A/KM8-E software is used with the KT8-A. The other set, that of KT8-A expanded instructions, is used with software designed expressly for use with the KT8-A. The LXM instruction (load extended mode register), in a category of its own, determines which set is to be performed; a description of this instruction follows. Tables 3-2 and 3-3 list and describe the two sets selectable with the LXM instruction.

LXM,6200,Load eXtended Mode register – The AC register is defined as shown below and is cleared upon the execution of this instruction.

AC Bit	**Contents**
0	*EMM* – Enable Memory Management instructions. When this bit = 1, the entire KT8-A instruction set is active. When this bit = 0, only the KM8-A/KM8-E instructions are performed.
1, 2	*EN5, EN9* – ENable MD bits 5 and 9 of the CIF and CDF instructions such that they define bank bit 0 and bank bit 1, respectively. These control bits are independent of the EMM bit. Thus, memory access can be extended with the EN5 and EN9 instructions without enabling the other KT8-A instructions.

Table 3-1 KT8-A Registers

Register	Description
Instruction Field Register (IFR)	5-bit register linked with the CPU's program counter (PC). Identifies the 4K-unit of memory from which instructions are being executed. Loaded from the instruction field buffer or from the programmer's console; cleared at power-up or upon interrupt.
Instruction Field Buffer (IFB)	Programmable 5-bit register whose contents are transferred to the IFR upon execution of a JMP or JMS instruction. Loaded from the programmer's console; cleared at power-up or upon interrupt.
Data Field Register (DFR)	Programmable 5-bit register whose contents become the most significant bits of the operand address for indirect AND, TAD, ISZ, and DCA instructions. Loaded from the programmer's console; cleared at power-up or upon interrupt.
Extended Mode Register (EMR)	Programmable 7-bit register that is loaded with parameters that define KT8-A operation, in particular, that group of instructions that are to be decoded by the KT8-A. Cleared at power-up and by the LXA key.
User Mode Buffer (UMB)	Programmable 1-bit register whose contents are transferred to the User Mode register upon execution of a JMP or JMS instruction. Cleared at power-up, upon interrupt and when the IFR and DFR are loaded from the programmer's console.
User Mode (UM) Flip-flop	Serves as a switch for multi-level software, i.e., when equal to 0, the foreground mode is selected, when equal to 1 the background mode is selected. The UM flip-flop is loaded from the user mode buffer upon execution of JMP and JMS instructions; the flip-flop is cleared at power-up, upon interrupt, and when the IFR and DFR are loaded from the programmer's console.
User Interrupt Flip-flop	In the background mode, this flip-flop is set when an illegal instruction is detected. Tested and cleared under program control or the INIT key.
User Size Register (USR)	Programmable 6-bit register that defines the number of contiguous memory fields to which a background program has access. In background mode the value of a field change is compared to the USR; if the USR is greater than the value specified by the change field instruction, normal operation continues; if not, the instruction is inhibited, an interrupt is generated, and the user interrupt flip-flop is set. A zero in the USR causes all change field instructions issued in background mode to result in an interrupt. The USR is cleared at power-up and by the LXA key.
Save Field Register (SFR)	11-bit register that the IFR, the DFR, and the UM flip-flop are transferred to upon interrupt. The SFR can be transferred to the AC for data retention, or its contents can be restored to the IFR, the DFR, and the UM flip-flop (see RMF instruction).
Relocation Register	Programmable 5-bit register. Used in the background mode to change the memory area in which a program operates. The field value specified in instructions that change the instruction field or data field are added to the contents of the relocation register. The result is loaded into the IFB or DFR. Cleared at power-up and by the LXA key.

Table 3-1 KT8-A Registers (Cont)

Register	Description
Maintenance Register	10-bit register that monitors the bank select and EMA lines. Loaded during each break cycle, whenever a data field access is made, and in maintenance mode during the execute cycle of a JMS instruction and the fetch cycle following a JMP instruction. Read by an I/O instruction and cleared by the CAF instruction.
Interrupt Inhibit Flip-flop	Set when IFB is changed or user mode buffer is set. When the interrupt inhibit flip-flop is set, the interrupt latency time is increased by an amount equal to the time between an instruction that changes the IFB or sets the UM flip-flop and the next JMP or JMS. When the interrupt inhibit flip-flop is set in the background mode, any illegal instruction issued causes an unrecoverable interrupt to occur immediately. Cleared upon execution of a JMP or JMS instruction.
Fatal Flip-flop	When set, this flip-flop indicates that an illegal instruction was issued in the background mode while the interrupt inhibit flip-flop was set. The flip-flop is accessible to the program and is displayed by the programmer's console in bit 3 of the status word. Cleared when the user interrupt flip-flop is 0.
Last Break Register (LBR)	4-bit register that is loaded with the priority number of the last device to have requested a break. The LBR is loaded with 17 at power-up and when a CAF instruction is issued. Each time a break occurs, a number between 0 and 14_8 is loaded.
Break Map	Programmable 13 X 2-bit register file that provides two additional address bits (bank bits) for the 15-bit address generated by a DMA device. The break map is activated by the EBM bit of the EMM word; when the map is not enabled, all data breaks are to bank 0.

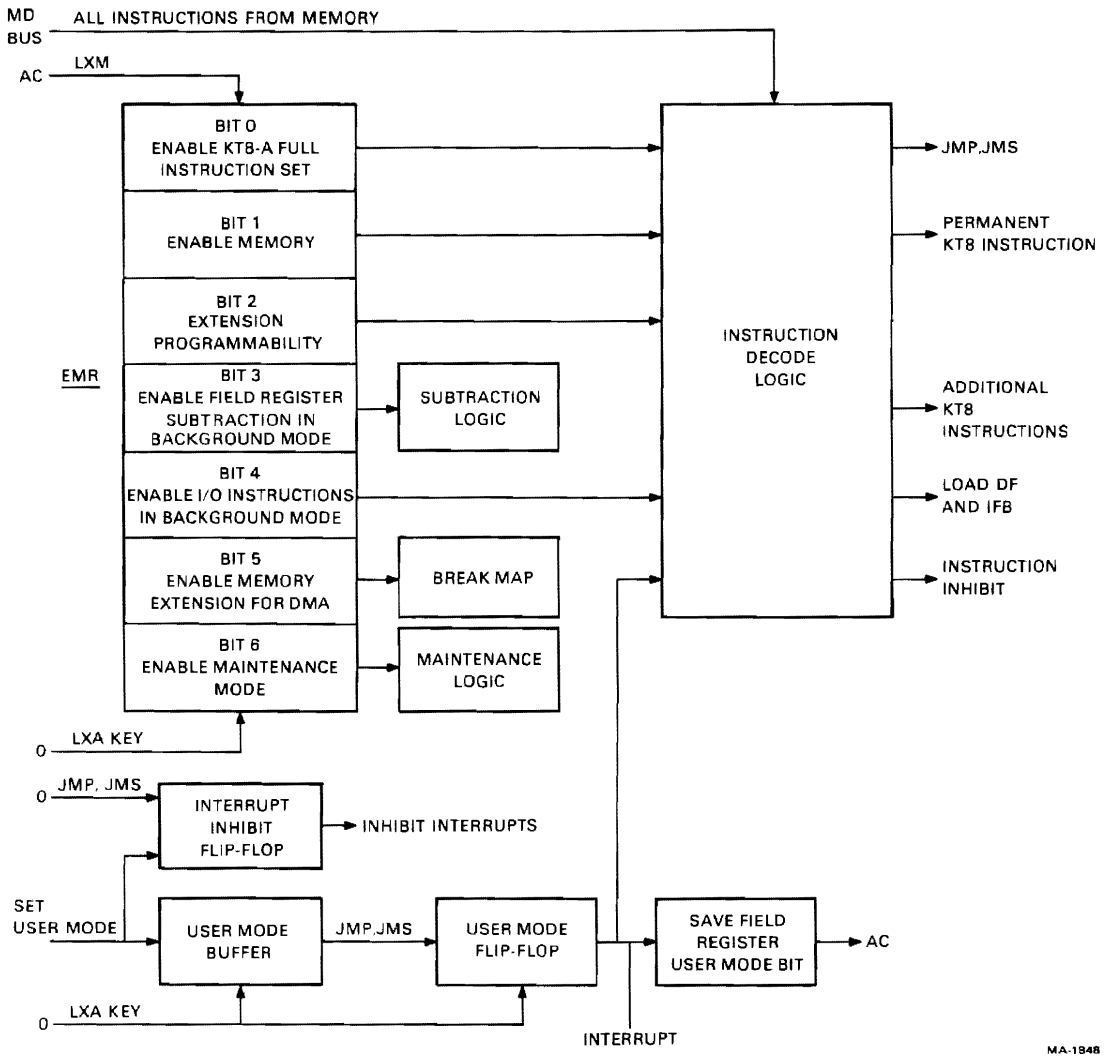


Figure 3-3 Mode Control Functional Diagram

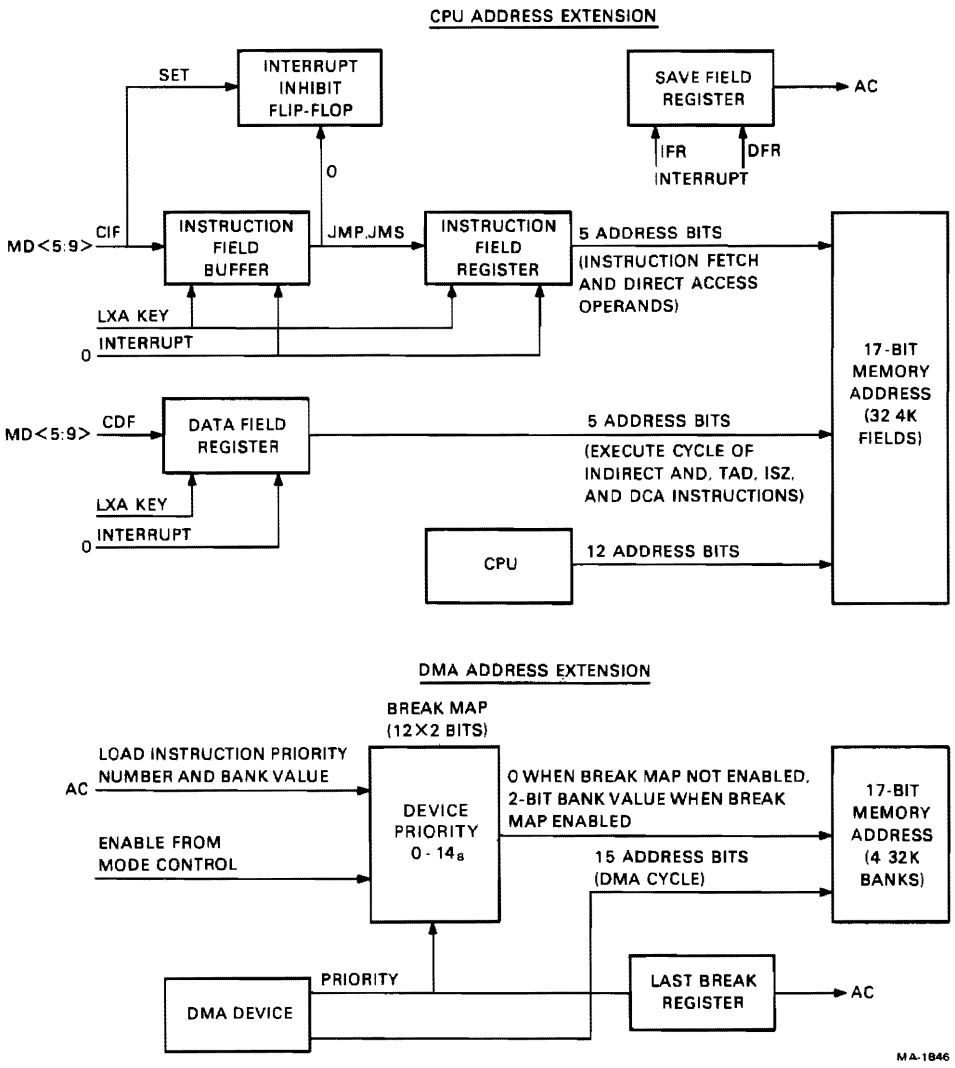


Figure 3-4 Memory Extension Functional Diagram

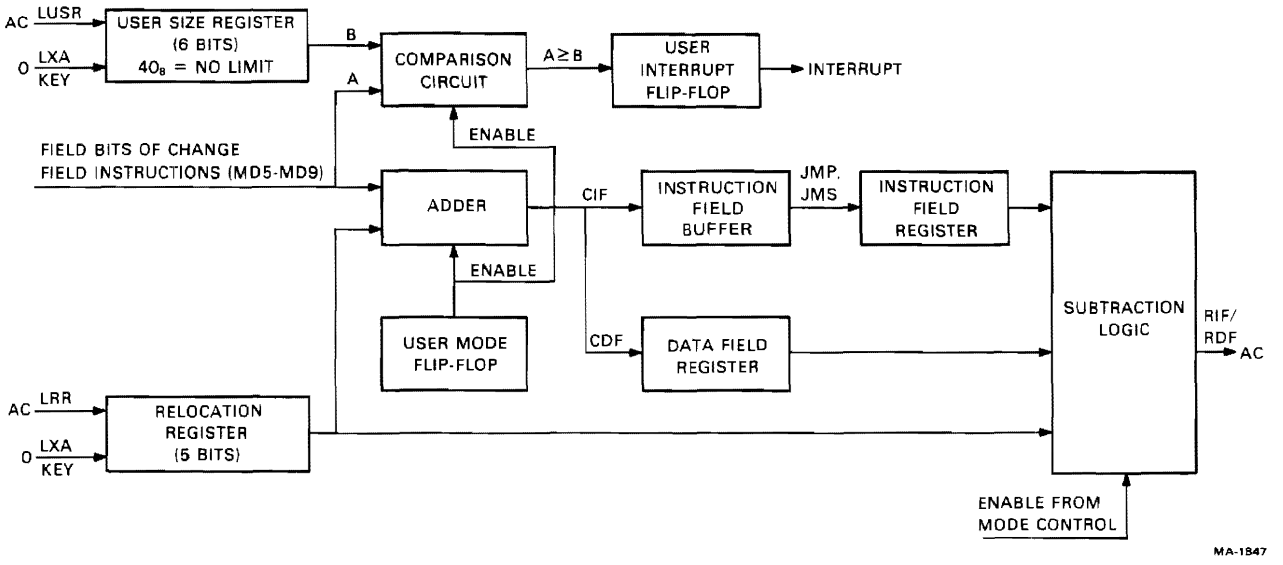


Figure 3-5 Memory Management Functional Diagram

Table 3-2 KM8-A/KM8-E Compatible Instructions

Instruction		
Octal	**Mnemonic**	**Description**
62N1	CDF	Change Data Field as defined by MD bits 6–8 (N) of the instruction. MD6=DF0, MD7=DF1, MD8=DF2
62N2	CIF	Change Instruction Field buffer as defined by MD bits 6–8 (N) of the instruction and set the interrupt inhibit flip-flop. MD6=IF0, MD7=IF1, MD8=IF2
62N3	CDF, CIF	Combination of the CDF and CIF instructions
6004	GTF	GeT Flags – Loads the contents of the save field register and CPU status bits into the AC register as shown below
6005	RTF	ResTore Flags – Loads the link, user mode buffer, instruction field buffer, and data field register from the AC register as shown below. RTF also turns on the interrupt system (ION) and sets the interrupt inhibit flip-flop until the next JMP or JMS

AC Bit	**Contents**
0	Link
1	Greater Than flag
2	Interrupt bus
3	Fatal flip-flop
4	Interrupt on
5	Save user mode
6	Save field bit 0 (IF bit 0)
7	SF1 (IF1)
8	SF2 (IF2)
9	SF3 (DF0)
10	SF4 (DF1)
11	SF5 (DF2)

Table 3-2 KM8-A/KM8-E Compatible Instructions (Cont)

Instruction		
Octal	**Mnemonic**	**Description**
		AC Bit — **Contents** 0 — Link 1 — Greater Than flag 2 3 4 5 — User mode buffer 6 — Save field bit 0 (IF bit 0) 7 — SF1 (IF1) 8 — SF2 (IF2) 9 — SF3 (DF0) 10 — SF4 (DF1) 11 — SF5 (DF2)
6204	CINT	Clear user INTerrupt flag
6214	RDF	Read Data Field – Inclusive-OR of the data field register with AC bits 6, 7, and 8
6224	RIF	Read Instruction Field – Inclusive-OR of the instruction field register with AC bits 6, 7, and 8
6234	RIB	Read Interrupt Buffer – Inclusive-OR of the save field register (not including save user mode) with AC bits 6 through 11, as shown below **AC Bit** — **Contents** 6 — SF0 (IF0) 7 — SF1 (IF1) 8 — SF2 (IF2) 9 — SF3 (DF0) 10 — SF4 (DF1) 11 — SF5 (DF2)
6244	RMF	Restore Memory Field – Transfers the contents of the save field register into the user mode buffer, instruction field buffer, and data field buffer; also sets the interrupt inhibit flip-flop
6254	SINT	Skip if user INTerrupt flag is set
6264	CUF	Clear User mode flip-flop and buffer
6274	SUF	Set User mode buffer and set interrupt inhibit flip-flop

Table 3-3 KT8-A Expanded Instructions

Instruction		Description
Octal	**Mnemonic**	
6 01X NNN Y01	CDF	Change data field as defined by MD bits 5 through 9 and bits EN5 and EN9 of the extended mode word; the order of significance of the resultant field is MD5, 9, 6, 7, 8 **MD Bit** — **Contents** 0 — 1 1 — 1 2 — 0 3 — 0 4 — 1 5 — B0 6 — F0 7 — F1 8 — F2 9 — B1 10 — 0 11 — 1
6 01X NNN Y10	CIF	Change instruction field buffer as defined by MD bits 5 through 9 and bits EN5 and EN9 of the extended mode word; also set the interrupt inhibit flip-flop **MD Bit** — **Contents** 0 — 1 1 — 1 2 — 0 3 — 0 4 — 1 5 — B0 6 — F0 7 — F1 8 — F2 9 — B1 10 — 1 11 — 0
6 01X NNN Y11	CDF, CIF	Combination of the CDF and CIF instructions
6004	GTF	GeT Flags – Loads the contents of the save field register and CPU status bits into the AC register as shown below **AC Bit** — **Contents** 0 — Link 1 — Greater Than flag 2 — Interrupt bus 3 — Fatal flip-flop 4 — Interrupt on 5 — Save user mode 6 — Save field bit 0 (IF bit 0) 7 — SF1 (IF1) 8 — SF2 (IF2) 9 — SF3 (DF0) 10 — SF4 (DF1) 11 — SF5 (DF2)
6005	RTF	ResTore Flags – Loads the link, user mode buffer, instruction field buffer, and data field register from the AC register as shown below. RTF also turns on the interrupt system (ION) and sets the interrupt inhibit flip-flop until the next JMP or JMS. (Bank bits are unchanged)

Table 3-3 KT8-A Expanded Instructions (Cont)

Instruction		
Octal	**Mnemonic**	**Description**
		AC Bit **Contents** 0 Link 1 Greater Than flag 2 3 4 5 User Mode buffer 6 Save field bit 0 (IF bit 0) 7 SF1 (IF1) 8 SF2 (IF2) 9 SF3 (DF0) 10 SF4 (DF1) 11 SF5 (DF2)
6170	LBM	Load Break Map – The map location (device priority) is defined by an octal number in AC bits 6 through 9 and the break bank is defined in bits 10 and 11. LBM also clears the AC
6171	RBM	Read Break Map – The map location to be read is defined by an octal number in AC bits 6 through 9 and the contents of that location are inclusively-ORed with AC bits 10 and 11; at the conclusion of this instruction the AC contains the device number (map location) and bank with AC bits 0 through 5 unaffected.
6172	RLB	Read Last Break – Each time a break occurs, the priority of the device requesting the break is loaded into the last break register. The priority is an octal number from 0 to 14. The CAF instruction loads the last break register with 17 to indicate that a break has not occurred. The RLB instruction loads the content of the last break register into AC bits 6 through 9 with all other AC bits cleared.
6173	RMR	Read Maintenance Register – The contents of the maintenance register are loaded into AC bits 1 through 11, as shown below **AC Bit** **Contents** 0 0 1 Remote Bank Select Line BS0 2 Remote Bank Select Line BS1 3 Remote Bank Select Line BS2 4 Remote Bank Select Line BS3 5 Local Bank Select Line BS0 6 Local Bank Select Line BS1 7 Local Bank Select Line BS2 8 Local Bank Select Line BS3 9 EMA0 10 EMA1 11 EMA2
6174	MBC	Maintenance Break Cycle – When this instruction is executed the AC represents the priority bus and a break cycle is simulated. If the AC = 0000, device 13 is specified. If more than one bit is set, the lowest numbered bit wins (0 is the highest priority). The memory cycle following the MBC instruction is a break from memory at location X00000, where X = the bank specified by the break map if the enable break map bit of the extended mode register = 1. If EBM = 0, X = 0. MBC also clears the AC and will not be inhibited if issued in background mode if the EMM bit is set.

Table 3-3 KT8-A Expanded Instructions (Cont)

Instruction		
Octal	**Mnemonic**	**Description**
6175	RACA	Rearrange the AC from form A to form B
6176	RACB	Rearrange the AC from form B to form C
6177	RACC	Rearrange the AC from form C to form A

These three instructions manipulate the AC bits, allowing easier calculations of field values. When these instructions are issued in background mode with the EMM bit of the extended mode register = 1, they are not inhibited. The contents of the AC register before and after each instruction are indicated below. Unused bits are always cleared; given the word ABCDE, A is the most significant bit and E the least significant bit of an octal number.

	Content		
AC Bit	**Form A**	**Form B**	**Form C**
0			
1			A
2			B
3		A	
4		B	
5	A		
6	C		C
7	D		D
8	E		E
9	B	C	
10		D	
11		E	

6204	CINT	Clear user INTerrupt flip-flop and the fatal flip-flop
6210	GTS	GeT Save register – The four save bank bits, six save field bits, the save user mode bit and the link are loaded into the AC register as shown below

AC Bit	**Contents**
0	Link
1	SB0 (IB0)
2	SB1 (IB1)
3	SB2 (DB0)
4	SB3 (DB1)
5	SUF
6	SF0 (IF0)
7	SF1 (IF1)
8	SF2 (IF2)
9	SF3 (DF0)
10	SF4 (DF1)
11	SF5 (DF2)

6214	RDF	Read Data Field – In expanded mode, the RDF instruction operates in one of three ways depending on the state of the user mode flip-flop, the ERVF bit of the extended mode word, and the relocation register. When the user mode flip-flop = 0, the 5-bit field is inclusively-ORed with the AC register as shown:

AC Bit	**Contents**
5	B0
6	F0
7	F1
8	F2
9	B1

Table 3-3 KT8-A Expanded Instructions (Cont)

Instruction		Description
Octal	Mnemonic	
		When the user mode flip-flop = 1 and ERVF = 0, the instruction is inhibited, the user interrupt flag is set, and an interrupt is generated. When the user mode flip-flop = 1 and ERVF = 1, the virtual field is inclusively ORed with AC bits 5 through 9 (the virtual field is the real field, i.e., B0, B1, F0, F1, and F2, minus the contents of the relocation register).
6220	RTS	ResTore Save registers – The instruction field buffer, data field register, and user mode buffer are loaded from the AC register as shown in the GTS instruction. This instruction also sets the interrupt inhibit flip-flop and clears the AC upon execution. Note that this instruction does not restore the link.
6224	RIF	Read Instruction Field – The RIF instruction inclusively-ORs the instruction field register with AC bits 5 through 9, with the same conditions as the RDF instruction.
6230	RXM	Read eXtended Mode register – The AC is loaded with the contents of the extended mode register as shown below. Unused AC bits are cleared. **AC Bit** **Contents** 0 EMM 1 EN5 2 EN9 3 ERVF 4 DIOI 5 EMB 6 MAINT
6234	RIB	Read Interrupt Buffer – Inclusively-ORs the save field register with the AC as shown below. **AC Bit** **Contents** 0 1 SB0 (IB0) 2 SB1 (IB1) 3 SB2 (DB0) 4 SB3 (DB1) 5 6 SF0 (IF0) 7 SF1 (IF1) 8 SF2 (IF2) 9 SF3 (DF0) 10 SF4 (DF1) 11 SF5 (DF2)
6240	LRR	Load Relocation Register – Load the relocation register from the AC as shown below. The AC is cleared upon execution of this instruction. **AC Bit** **Contents** 7 RB0 8 RB1 9 RF0 10 RF1 11 RF2
6244	RMF	Restore Memory Field – This instruction operates on the entire save register in all cases. The 5-bit IFB, the 5-bit DF, and the UMB are restored, and the interrupt inhibit flip-flop is set.

Table 3-3 KT8-A Expanded Instructions (Cont)

Instruction		
Octal	**Mnemonic**	**Description**
6250	RRR	Read Relocation Register – The relocation register is loaded into AC bits 7 through 11 as shown in the LRR instruction. All unused bits are cleared.
6254	SINT	Skip if User INTerrupt flip-flop is set.
6260	LUSR	Load User Size Register – The user size register is loaded from AC bits 6 through 11 with an octal number that defines the number of fields accessible in background mode. If this number is 0, all CIF and CDF instructions are inhibited and the user interrupt flag is set. If the number is 40, or greater, no user size violations will occur. The user size register is cleared at power-up, thereby maintaining compatibility with KM8-E operation. The AC is cleared on execution of this instruction.
6264	CUF	Clear User mode Flip-flop.
6270	RUSR	Read User Size Register – The contents of the user size register are loaded into AC bits 7 through 11. All other AC bits are cleared.
6274	SUF	Set User mode buFfer and set interrupt inhibit flip-flop.
6101*	SBE	Skip if Battery Empty flag is set
6102*	SPL	Skip if AC Low flag is set
6103*	CAL	Clear the AC LOW interrupt

* These instructions are KM8-AC instructions; thus, they are used only with the KT8-AB option, and are listed only for reference.

3 *ERVF* – Enable the Reading of the Virtual Field in background mode. When ERVF = 0, the RIF and RDF instructions cause an interrupt trap. When ERVF = 1, the RIF and RDF instructions inclusively-OR the difference between the instruction/data field registers and the relocation register with the AC, removing the requirement for interrupt processing.

4 *DIOI* – Disable I/O Interrupts in background mode. In background mode, when DIOI = 0, all I/O instructions are inhibited and cause the illegal instruction flag to be set and an interrupt request to occur. When DIOI = 1, only illegal change field I/O instructions will cause a trap. Regardless of the state of this bit, HLT and OSR instructions will cause traps in background mode.

5 *EBM* – Enable Break Map. When EBM = 0, all DMA transfers are to bank 0. When EBM = 1, DMA transfers are to the bank defined by the break map for each device. The contents of the break map are not defined at power-up.

6 *MAINT* – MAINTenance mode. When MAINT = 1, a JMP or JMS instruction triggers a field change for one cycle. During this cycle the maintenance register is loaded with the state of the bank select and EMA lines, the user interrupt flag is set, and an interrupt occurs. This is done to allow complete testing of the KT8-A option, regardless of whether the full 128K words of memory are available. When MAINT = 0, the KT8-A operates normally.

3.2.2 Functional Description

The KT8-A performs three different functions: mode control, memory extension, and memory management. These functions are illustrated in Figures 3-3, 3-4, and 3-5, respectively. Mode control refers to programmable conditions that enable, disable, or alter the execution of certain instructions and DMA transfers. Mode control is established by the 7-bit extended mode register and an independent user mode flip-flop. Three bits of the extended mode register establish the programmability of the KT8-A, defining the I/O space that it will occupy (non-active KT8-A device codes may be used for other I/O devices). Table 3-4 shows the relationship between these bits and the KT8-A instruction codes.

Table 3-4 KT8-A Programmability Modes

Programmability Mode	Instruction Code
Permanent KT8-A instructions KM8-A compatible mode 15-bit memory address	6200, 62X1, 62X2, 62X3, 62X4
Additional KT8-A instructions	617X, 62X0
16th address bit programmable CPU access of 64K	62X5,62X6, 62X7
17th address bit programmable	63XN

Memory extension of CPU addresses is effected by three 5-bit registers in the KT8-A: the instruction field register defines the 4K segment of memory where the program is currently operating; the instruction field buffer register defines the destination field of any JMP or JMS instruction; and the data field register defines the field of any indirect operand address for AND, TAD, ISZ, or DCA instructions. The IFB and DF registers are loaded by I/O instructions. Bits 5 through 9 of the instruction contain the field number, bit 10 indicates the IFB, and bit 11 indicates the DF. The instruction field register is loaded from the IFB by any JMP or JMS instruction.

When an interrupt occurs, the instruction field register and data field register are loaded into a holding register (save register) and are then zeroed. Since the contents of the IFB are lost upon interrupts, the KT8-A, like the KM8-A or KM8-E, inhibits interrupts between the time that the IFB is loaded and the execution of a JMP or JMS instruction. Care must be taken to keep this period short since it directly adds to the interrupt latency time of the system.

When DMA memory extension is disabled, all DMA transfers are to the first 32K of memory. When it is enabled, the KT8-A uses a 13 X 2 bit programmable register file called the break map. By monitoring break priorities during DMA transfers, the KT8-A is able to distinguish one device from another. When the break map is programmed, the priority number is included with the number of the memory bank (32K-word segment) involved in the data transfer. The KT8-A provides additional hardware that gives a program the ability to identify a given device with its priority number. This is required since many DMA devices are set in the factory to a priority number relative to the other devices on the system.

As a program runs, instructions are executed sequentially as they occur. However, in memory management mode certain instructions are altered or inhibited. When an instruction to be inhibited is encountered in memory management mode, a flag is set in the KT8-A, an interrupt occurs, and the system returns to normal mode, where the operating system can examine the inhibited instruction and proceed accordingly. For example, consider that two programs are in memory, are runnable, and are controlled by an operating system. Program 1 is running when a HLT instruction is encountered. Since HLT is inhibited in memory management mode, an interrupt occurs, activating the operating system. Through an examination of flags (I/O skips), the operating system identifies the interrupt as being caused by an inhibited instruction. By examining the location of that instruction, the operating system detects the HLT instruction.

Rather than stopping operation, the operating system simply does not run program 1 any more and starts program 2 (program 1 can be restarted only through operator intervention).

In the KM8-A/KM8-E compatible mode, I/O, HLT, and OSR instructions are inhibited when memory management mode is enabled. Since the instructions to establish new values in the instruction field buffer and data field register are I/O instructions, the operating system is entered by means of an interrupt whenever these instructions occur. An adjustment is made by the operating system, the instruction issued, and the running program reentered. For example, consider that program 1 and program 2 are both 8K-word programs and were both written to run in fields 0 and 1. The operating system loaded program 1 into fields 3 and 4, and program 2 into fields 5 and 6. Program 2 is running when an instruction to change the data field register to field 1 occurs. The operating system identifies the instruction that caused the interrupt. It makes sure that program 2 is not trying to access more than the 8K allocated at load time, adds 5 to the 1 contained in the inhibited instruction, issues the instruction that makes the data field register equal to 6, and then continues from where the interrupt occurred.

With the KT8-A, memory management can be effected without operating system intervention. There are two programmable registers in the KT8-A that are loaded by the operating system before a program is started in memory management mode: the user size register and the relocation register. The user size register contains a value equal to the number of fields allocated to a running program. The relocation register is loaded with the number of the first field occupied by the running program. In KM8-A/KM8-E compatible mode, both of these registers are zero. Since the user size register says zero fields are allocated, the CDF1 instruction that loads the data field register in the preceding example is inhibited. If the user size register were loaded with a value of 2, and 5 were loaded into the relocation register, the CDF1 instruction would not be inhibited and the value 6 would be loaded into the data field register.

The switch from normal mode (foreground) to memory management mode (background) is effected by the user mode buffer and user mode flip-flop. The user mode buffer is programmable with I/O instructions. The user mode flip-flop is loaded with the state of the user mode buffer each time JMP or JMS instructions are executed. To start program 2 with KT8-A memory management enabled, the operating system does the following: Sets USR = 2, sets relocation = 5, sets IFB = 5, sets data field = 5, sets user mode buffer, and JMPs to first location of program 2.

When interrupts occur, the user mode flip-flop, the IF, and the DF are loaded into the save register and then zeroed, thus returning to normal mode. As with programming the instruction field buffer, care must be taken to minimize the time between setting the user mode buffer and executing the JMP instruction because interrupts are inhibited during this time.

As mentioned in the discussion of mode control, the extended mode register contains two memory management option bits. The ERVF bit allows the instructions that read the values of the IF and DF to occur in background mode and at the same time subtracts the value of the relocation register from the IF or DF value, thereby delivering to the AC a value consistent with the field that the running program thinks it is using. The DIOI allows all I/O instructions to be executed in background mode with the DIOI bit set; as in all other memory management modes, HLT and OSR instructions are always trapped.

If, while a program is running under memory management control, an inhibited instruction is encountered while interrupts are inhibited (after the IFB has been changed and before the JMP or JMS), the interrupt will occur anyway. This sequence is unsupportable under KT8-A memory management. Therefore, along with setting the illegal instruction flag, a status bit within the KT8-A (Fatal) is set.

3.2.3 Programming Examples

3.2.3.1 128K-Word Memory Implementation – The basic PDP-8 has 4K words of memory, uses indirect addressing, and does not require IOT instructions to access any part of memory. The KM8-E memory option increases the PDP-8 memory from 4K to 32K words. This option uses a data field and an instruction field. The instruction field indicates the 4K unit of memory from which instructions are executed. The data field indicates the 4K unit of memory in which indirect AND, TAD, DCA, and ISZ instructions are executed.

The KM8-E requires execution of IOT instructions to manipulate the data and instruction fields. CDF (62X1) changes the data field to the value of X, where X varies from 0 to 7. CIF (62X2) changes the instruction field buffer to the value of X to load the instruction field when the next JMP or JMS is executed. During the time between the CIF instruction and the following JMP or JMS, it is necessary for the hardware to inhibit interrupt so as to prevent loss of the next instruction field. Thus, with the KM8-E option, data can be fetched from another field by issuing a CDF instruction to that field and using indirect addressing. A program or subroutine in another field can be accessed by issuing a CIF instruction, followed by a JMP or JMS indirect to that field. The additional IOT instructions, RDF and RIF, allow the program to interrogate the instruction and data fields.

Implementation of the 128K-word memory by the KT8-A option is similar to the KM8-E 32K-word implementation. The KT8-A powers up in a state compatible with the KM8-E, i.e., 32K words are accessible. When the LXM instruction is issued to the KT8-A with AC bits 5 and 9 (corresponding to EN5 and EN9, respectively) equal to 1, greater-than 32K-word accessibility is enabled. Instruction and data fields have the same meaning as with the KM8-E, but they are up to five bits long, rather than three bits long as with the KM8-E.

The RDF and RIF instructions return five bits. The CDF and CIF instructions require an optional number of bits because of a conflict between the use of IOT bits to describe KT8-A memory space and to describe IOTs. In Figure 3-6 the letters A, B, C, D, and E represent the bits in the 5-bit field number, with the A bit being the most significant. The ones and zeroes represent the bits in a 62X1 CDF IOT. To reference over 64K words, the A bit would have to be set to 1. However, this would be indistinguishable from, for example, terminal IOTs in the range 63XX. When the LXM instruction establishes the KT8-A operation, the IOT5 enable option determines whether a 63XX is a terminal (or other device) IOT, or a CDF (or CIF) to a high field. The result is that from 32K to 64K words there is no ambiguity; but from 64K words and up, direct absolute addressing can be used only if there are no device IOTs in the 63XX range. The repositioning of the A, B, C, D, and E bits to A, C, D, E, B in the KT8-A is necessary to maintain compatibility with the existing KM8-E 32K-word hardware. A CDF70 used to access field 7 of a 32K machine would be 6271 for the KM8-E. For the KT8-A it is also 6271.

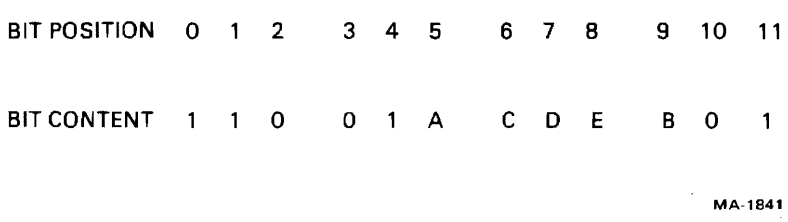


Figure 3-6 62X1 CDF IOT Instruction

Consider an existing single-user 32K-word program that is to be modified to be a KT8-A program and having no I/O being carried out above 32K and nothing being done in user mode. At start-up time the KT8-A would be enabled with the IOT9 bit set. At run time regular CDFs and CIFs would be issued to access up to 32K. CDFs and CIFs of the form 62X5 and 62X6 would be issued to access up to 64K. Nothing else would be required for the modification.

3.2.3.2 KT8-A As I/O Controller – The KT8-A functions in part as an I/O controller for DMA devices. The reason for this intrusion of a memory management device into I/O activities is that DMA addressing is limited to 32K words. The KT8-A is required for DMA transfers above 32K to provide the extra 2 bits to make up a 17-bit address. The KT8-A break map contains a list of 2-bit numbers that append DMA addresses encountered by the hardware. The list is ordered by bus priority for hardware convenience and speed. If a DMA break at bus priority 5 occurs, the hardware appends the fifth entry in the break map to the DMA break address.

Generally speaking, the bus priority of the DMA device is not known. The following programming example deals with this problem. When the bus priority has been obtained, that priority is used as an index to establish the break map bits for a given I/O transfer. If a transfer is to occur between 32K and 64K words, the map must contain 01 in the appropriate priority address. DMA I/O transfers always occur to absolute memory locations.

/DMA INITIALIZATION – CLEAR THE BREAK MAP
The break map (RAM) has been enabled but contains unknown data. Zero its entries so that DMA breaks occur to bank 0.

```
        TAD     (-15    /13 DECIMAL ENTRIES IN THE MAP
        DCA     COUNT   /CYCLE THROUGH ALL 13
CLEARL, TAD     COUNT   /BRING UP COUNT (-15 to -1)
        TAD     (15     /COUNT NOW TO 14 OCTAL
        CLL RTL         /POSITION COUNT FOR COMING IOT
        LBM             /BITS 10 AND 11 IN AC ARE ZERO
        ISZ     COUNT   /DONE?
        JMP     CLEARL  /NOT YET
```

/DMA INITIALIZATION – Each DMA handler must determine its bus priority on a one-time basis so that the handler may correctly alter the break map.

NOTE

Some devices can be made to cause a break in maintenance mode. With other devices the medium (DECtape, for example) must be mounted because an actual I/O transfer is necessary to cause a break.

Action: Issue a CAF instruction, which places 17_8 into the last break register (17_8 is not a legal bus priority). The handler must start up an I/O transfer to or from the device; if the break map has been cleared, that transfer will be within the first 32K (first bank) when an interrupt occurs.

```
RLB             /READ LAST BREAK REGISTER
TAD     (7704   /CHECK FOR ILLEGAL 17 (SHIFTED UP TWO)
SNA             /SKIP IF NOT
JMP     ERROR   /NO BREAK REGISTERED
TAD     (74     /RE-ESTABLISH ORIGINAL VALUE
DCA     DEVMAP  /THIS IS BUS PRIORITY FOR OUR DEVICE
```

Issues and Discussion: It is not necessary to place zeroes in the break map. The initial value could be any other value if it is more convenient to do transfers from bank 1, for example.

In the preceding programming example the handler initialization code assumed that it is not possible to receive some other device's break. Note that the RLB instruction reads the last break that occurred in the system, not the last for any particular device. If the initialization occurs in a dynamic environment, each break would have to be checked against all known break values. Only a new value would be valid for the device currently being initialized.

Use of Break Map for I/O Transfers: Assume that the handler has been called with the data field set to that field to which the transfer will take place.

```
RDF                         /OBTAIN BANK BITS IN BIT5 AND BIT9
RACA                        /REARRANGE BANK BITS TO BIT3 AND BIT4
RAR CLL                     /SHIFT BANK BITS TO BIT10 AND BIT11
BSW                         /KT MACHINES ARE ABLE TO BYTE SWAP
TAD         DEVMAP          /GET PRIORITY ADDRESS IN BREAK MAP
LBM                         /ESTABLISH BANK (FIELD BITS IN BIT0
                            /AND BIT1 ARE IRRELEVANT)
```

3.2.3.3 KT8-A in a System Environment – When an operating system is being created or modified to run with the KT8-A there are a number of important considerations. The operating system must handle interrupts and return correctly to the interrupted code. The system must handle context switching, during which a running job is suspended; enough context information must be saved to resume the suspended job at a later time. During the context switching another job must be started from its stored context information.

The operating system may deal with user mode jobs. A user mode job may not perform certain machine operations, such as HLT or I/O, which might disturb an important system job. RTS8, for example, runs OS8 as a user mode job. Sine OS8 in user mode can no longer perform I/O, RTS8 must handle I/O for OS8.

In the following discussion an operating system is invented, followed by a brief description of that system. Coding examples for system functions involving the KT8-A are given.

The system contains N jobs, numbered from 1 to N, where the lowest number is the highest priority. It is the responsibility of the system to run the highest priority runnable job. The location 'CURJOB' contains the number of the currently running job. A value of 0 in CURJOB means that the executive is between jobs, so that the state is not saved on a context switch.

The location 'NEWJOB' contains the number of the job to be run. Job status information does not include the EAE, MQ, step counter, and greater-than flag. The job status information is stored in field 0 in an array named 'STATE'. Each job has six status locations: LXM word, LRR word, LUSR word, PC, AC, and the RTS word.

The interrupt handling conventions assume that the system saves the AC, PC, Link, DF, and the IF for the interrupted job. Any other system resources (EAE, for instance) that are required by specific interrupt code must be saved and restored by that code.

KT8-A Start-up

Typical KT8-A start-up sequences are presented below. It is assumed that there is no 63XX IOT conflict, so that 128K words can be addressed.

```
32K

TAD         (4400
LXM                         /ENABLE EXTENDED MODE, BUT DO NOT
                            /NEED EN5, EN9, OR THE BREAK MAP

64K

TAD         (5500           /EXTENDED MODE, EN9, AND BREAK MAP
LXM
                            /MUST STILL INITIALIZE THE BREAK
                            /MAP AND THE DMA I/O HANDLERS
```

128K

```
        TAD     (7500       /EXTENDED MODE, EN9, EN5, AND
                            /BREAK MAP
        LXM
                            /MUST STILL INITIALIZE THE BREAK
                            /MAP AND THE DMA I/O HANDLERS
```

NOTE

In all cases ERVF is enabled for background (user mode) operation.

The use of the KT8-A with interrupt code is explained in the two following examples. All interrupts cause the PDP-8 to trap to location 0 of absolute field 0. The data and instruction fields are set to 0 and user mode is disabled.

Typical Interrupt Handling Code

NOTE

All code in the example is assumed to be in field zero.

```
0           HLT                     /INTERRUPT PC STORED HERE
1           JMP     INTRPT          /WE ARE NOW IN EXEC, NOT USER MODE
INTRPT,     DCA     SAVAC           /SAVE THE INTERRUPTED AC
            GTS                     /LINK, USER MODE, DF, IF FROM INTER-
                                    /RUPT
            DCA     INTFLG          /SAVE THESE FOR RETURN FROM INTER-
                                    /RUPT
```

Issue skip IOTs to determine which device has interrupted, clear the interrupting flag, do whatever work is necessary, and go to 'DISMIS' to dismiss the interrupt.

```
COMMON CODE TO DISMISS INTERRUPT
    DISMIS, AC4000                  /CLL CLA CML RAR; THE FOLLOWING RTS
            TAD     INTFLG          /DOES NOT RESTORE LINK, DO IT HERE
                                    /WITH TAD
            RTS                     /RESTORE DF; IF AND USER MODE TAKE
                                    /EFFECT ON NEXT JMP OR JMS; RTS
                                    /CLEARS AC
            TAD     SAVAC           /RESTORE INTERRUPTED AC
            ION                     /RTS INHIBITS INTERRUPTS UNTIL AFTER
                                    /JMP; TURN THE INTERRUPT SYSTEM ON
            JMP I   0               /RETURN TO INTERRUPTED TASK
```

Modified Interrupt Handling Code

This modified interrupt handling code allows faster processing for special case interrupts.

```
        0 HLT                       /INTERRUPT PC HERE
        1 JMP       CKSPCL          /GO CHECK SPECIAL CASE
```

NOTE

There is no reason that CKSPCL could not be at location 1, rather than jumping to it.

```
CKSPCL, CLSKP                       /SPECIAL CASE CLOCK; SKIP IF CLOCK
                                    /FLAG, CLEAR IT
        JMP         INTRPT          /NO, GO TO REGULAR CODE (OR ANOTHER
                                    /SPECIAL CASE)
        ISZ         CLKCNT          /COUNT OFF CLOCK TICKS UNTIL ACTION
                                    /NEEDED
        JMP         DMSPCL          /NOTHING NEEDED, GO TO SPECIAL DISMISS
        DCA         SAVAC           /FINALLY, WE HAVE TO SAVE AC
ACT ON CLOCK INTERRUPT, FOR EXAMPLE, INCREMENT SECONDS
ASSUME LINK NOT MODIFIED
        JMP         DMSPAC          /GO RESTORE AC, AND SPECIAL DISMISS
```

Special Dismiss Code

The RMF may be used when no modification of the IF or DF or Link has occurred in the interrupt handling code.

```
DMSPAC, TAD         SAVAC           /RESTORE AC
DMSPCL, RMF                         /INHIBIT INTERRUPTS, RESTORE KT8-A
                                    /TO PRE-INTERRUPT STATE
        ION                         /LET INTERRUPTS HAPPEN AFTER JMP I
        JMP I       0               /EXIT FROM INTERRUPT
```

3.2.3.4 Context Switching – Use of the KT8-A in context switching for saving current running jobs, new job start-up, and executive mode jobs is explained below.

Context Switching – Saving the Current Job

In this example an interrupt has caused the currently running job to be no longer runnable. Typically, the interrupt has taken a higher priority job out of a wait state. The context of the current job must be saved.

```
CONTEXT SWITCH CODE
INTERRUPTED JOB IS NO LONGER RUNNABLE (CODE STILL AT INTERRUPT LEVEL)
AC IN 'SAVAC', GTS WORD IN 'INTFLG'
CODE RUNNING IN FIELD 0
X10=10/USE AUTO-INCREMENT 10

THIS CODE DOES NOT SAVE EAE STATE
    TAD         CURJOB      /*TO ACCESS TABLE
    SNA                     /IS A JOB REALLY RUNNING?
    JMP         NOSAV       /BETWEEN JOBS NO SAVE
    CLL RAL
    TAD         CURJOB
```

```
        CLL RAL
        TAD     (STATE-1        /TABLE BASE ADDRESS
        DCA     X10
        RXM                     /EXTENDED MODE CONTROL WORD
        DCA I   X10             /NOT USUALLY PART OF JOB STATE
        RRR                     /RELOCATION REGISTER, PART OF
        DCA I   X10             /STATE IF MORE THAN ONE USER JOB
        RUSR                    /USER SIZE REGISTER
        DCA I   X10
        TAD     0               /INTERRUPTED PC
        DCA I   X10
        TAD     SAVAC           /INTERRUPTED AC
        DCA I   X10
        TAD     INTFLG          /DF, IF, LINK, USER MODE
        DCA I   X10             /FOR INTERRUPTED JOB
        DCA     CURJOB          /SIGNAL NO USER JOB RUNNING
NOSAV,  ION                     /WE ARE GOING TO START UP JOB
                                /WHOSE # IS IN NEW JOB
```

Context Switching – Starting the New Job

The following example shows the code required to start a new job. This code might be reached from several areas of the system, as well as from the previous example.

```
CONTEXT SWITCH CODE
INTERRUPTS ON, JOB TO START IN NEW JOB
THE 0 IN CURJOB SAYS DON'T SAVE PRESENT
JOB STATE; WE ARE 'BETWEEN' JOBS

        TAD     NEWJOB          /SET UP TO ACCESS STATE
        CLL RAL                 /OF NEW JOB
        TAD     NEWJOB          /*6 POINTS TO SAVED
        CLL RAL                 /STATE
        TAD     (STATE-1        /UPON AN INTERRUPT ENABLING
        DCA     X10             /STILL A HIGH PRIORITY JOB
        TAD I   X10             /THIS CODE IS RE-ENTERED
        LXM                     /EXTENDED MODE CONTROL WORD
        TAD I   X10             /RELOCATION REGISTER
        LRR
        TAD I   X10             /USER SIZE REGISTER
        LUSR
        TAD I   X10             /TASK'S PC
        DCA     TEMP            /NOT 0, AS INTERRUPTS STILL ON
        TAD I   X10             /TASK'S AC
        DCA     TEMP2           /PUT AWAY FOR NOW
        CIF     0               /HOLD OFF INTERRUPTS, NOW WE ARE
                                /COMMITTED TO
        TAD     NEWJOB          /START UP NEW JOB
        DCA     CURJOB
        AC4000                  /CLL CML CLA RAR
        TAD I   X10             /TAD SETS LINK
        RTS                     /SETS DF NOW (NO TAD I X10S CAN
                                /FOLLOW)
        TAD     TEMP2           /AC SET
        JMP I   TEMP            /GO TO TASK, IF AND USER MODE
                                /TAKE EFFECT NOW
```

Context Switching – Executive Mode Job

In the following example an executive mode job has called the system such that the calling job is no longer runnable. This is similar to the example in which the running job was saved because it was suspended by an interrupt. The programming difference is that there is no convenient word corresponding to the interrupt GTS available in the system call case.

```
AS A RESULT OF EXEC MODE MONITOR CALL,
'WAIT' FOR INSTANCE, PRESENT JOB IS NO LONGER
RUNNABLE. ASSUME DF POINTS TO CALLER'S FIELD
X11=11 /AUTO INCREMENT 11 USED (INTERRUPT LEVEL OWNS 10)

        TAD     CURJOB          /SET UP TO SAVE STATE
        CLL RAL
        TAD     CURJOB          /*6
        CLL RAL
        TAD     (STATE+2)       /FIRST 3 STATE TABLE ENTRIES UNCHANGED
        DCA     X11
        RDF                     /CALLER'S DATA FIELD
        RACA                    /DATA FIELD BITS IN RTS FORM
        DCA     TEMP            /SAVE DATA FIELD BITS
        TAD     TEMP
        RACB                    /INSTRUCTION FIELD BITS
        TAD     TEMP            /NOW HAVE RTS WORD
        DCA     TEMP            /LINK, USER MODE=0
        CIF CDF 0               /INHIBIT INTERRUPTS, OUR FIELD
        TAD     CALLEX          /CALL EXIT ADDRESS
        DCA I   X11             /FOR PC
        DCA I   X11             /AC=0
        TAD     TEMP            /RTS WORD HAS DF, IF
        DCA I   X11
        JMP     RESCAN          /ENABLE INTERRUPTS, GO SCAN FOR
                                /RUNNABLE JOB
```

3.2.3.5 User Mode Hidden State – The KT8-A user mode design places a non-obvious restriction on the user mode task. The 'hidden state' between the CIF and the following JMP or JMS was previously simulated in software so that the next instruction field would not be lost. This software simulation is no longer necessary with the KT8-A; however, the execution of an IOT (other than CDF, CIF, RDF, or RIF) in the 'hidden state' is prohibited. When the IOT traps out of the hidden state, the contents of the instruction field buffer are lost, so the user program cannot continue. (Delaying the trap does not help because the system cannot find the IOT.) If there are existing user mode programs that are to be converted for use with the KT8-A, any hidden state IOTs should be removed. When the user mode handling task receives an interrupt from such an IOT, the fatal bit is set, indicating that the interrupt cleared the instruction field buffer set by the CIF. The Fatal bit is loaded into AC bit 3 when the GTF instruction is issued.

3.2.3.6 Modified User Mode – With the KT8-A there are two ways to modify user mode so that it has some executive mode characteristics. The DIOI bit in the LXM control word is normally cleared. When this bit is set, user mode IOTs execute as if in executive mode and do not cause a user mode interrupt. HLTs and OSRs still cause a user mode interrupt. This feature is useful to a user mode program that must control an I/O device for which there is no system handler. The protection and mapping will still occur in the normal way. This mode of operation should be used with care because the user program can destroy the system.

The ERVF bit in the LXM control word is normally set when relocation is used. In this mode of operation RDF and RIF instructions return the virtual field value (real field minus relocation register). When the ERVF bit is cleared in user mode, the RDF and RIF instructions cause a user interrupt.

When the DIOI bit is set and the ERVF bit is cleared, user mode is then called 'mapped executive' mode. System I/O, including operation of the KT8-A, can be carried out in the normal manner. Note that HLTs and OSRs will interrupt. The consequence of this mode is that the relocation register can be used to reference data.

ASSUME STARTING WITH DATA AND INSTRUCTION FIELDS = 0

```
TAD      (HIFLD)     /VALUE OF SOME HIGH FIELD
LRR                  /RIGHT JUSTIFIED FOR RELOCATION
                     /REGISTER
TAD I    (LOCAT)     /FETCH SOME DATA
DCA      LOCAL       /TO LOCAL STORAGE
LRR                  /BRING DATA FIELD BACK
                     /ALSO INSTRUCTION FIELD BEFORE
                     /DOING A CIF
                     /USER SIZE REGISTER SHOULD BE 40
```

This seemingly awkward mechanism (mapped executive mode) is used when the user constructs a machine greater than 64K words and uses 63XX for I/O devices. The program must keep track of its field manipulations. Optimum results are obtained by running with IOT9 so that the window is of maximum size (up to 64K). Direct CIFs (with relocation register = 0) can be issued to reach code in other fields. The CDF could point to the field for the subroutine return.

NOTE
RDF and RIF instructions return real field values.

3.2.3.7 Programming Notes – KT8-A field bits come in a variety of configurations; for example:

```
NORMAL    0000000ABCDE     /Where ABCDE are the bits in a right-justified 5-bit
                           number

CDF       00000ACDEB00     /Bits in place for a CDF (the 6201 for the CDF not
                           shown)

DFORM     000AB0000CDE     /Bits in place for the data field in a GTS word

IFORM     0AB000CDE000     /Bits in place for the instruction field in a GTS word
```

The KT8-A hardware provides three instructions for bit transformations:

```
RACA    /(6175) converts CDF to DFORM
RACB    /(6176) converts DFORM to IFORM
RACC    /(6177) converts IFORM to CDF
```

In all cases, the other seven AC bits are cleared, and the link is not modified. The following are coding examples for some of the other possible transformations.

```
/
/  NORMAL TO CDF (AC TO AC)
/
        CLL RTR          /D, E00000000ABC
        RTR              /B, CDE00000000A
        BSW              /B, 00000ACDE000
        SZL              /SKIP IF NO B BIT
        TAD      (4      /B, 00000ACDEB00
```

```
/
/  CDF TO NORMAL (AC TO AC)
/
        BSW             /?, CDEB0000000A
        CLL RTL         /D, EB0000000A0C
        RTL             /B, 0000000A0CDE
        SZL             /SKIP IF NO B BIT
        TAD     (10     /B,0000000ABCDE


/
/  NORMAL TO IFORM (CORE TO AC)
/
        TAD     NORMAL  /?, 0000000ABCDE
        CLL RTR         /D, E00000000ABC
        BSW             /D, 000ABCDE00000
        TAD     NORMAL  /D, 000ABCABCDE
        RACB            /D, 0AB000CDE000    FOR DFORM, AND (607


/
/  INCREMENT A CDF TO THE NEXT FIELD (WRAPS 128K)

        TAD     THECDF
        RACA            /IN DFORM, OTHER BITS GONE
        TAD     (171    /INCREMENTS THE SCATTERED BITS
        RACB            /IFORM, CLEAR GARBAGE BITS
        RACC            /IN CDF FORM
        TAD     (CDF 0  /PUT BACK THE BITS FOR THE CDF
        DCA     THECDF  /PUT IT BACK
```

KT8-A MEMORY MANAGEMENT
CONTROL USER'S GUIDE
EK-KT08A-UG-001

Reader's Comments

Your comments and suggestions will help us in our continuous effort to improve the quality and usefulness of our publications.

What is your general reaction to this manual? In your judgment is it complete, accurate, well organized, well written, etc.? Is it easy to use? _____

_____

_____

_____

What features are most useful? _____

_____

_____

_____

What faults or errors have you found in the manual? _____

_____

_____

_____

Does this manual satisfy the need you think it was intended to satisfy? _____

Does it satisfy *your* needs? _____ Why? _____

_____

_____

_____

☐ Please send me the current copy of the *Technical Documentation Catalog*, which contains information on the remainder of DIGITAL's technical documentation.

Name _____ Street _____
Title _____ City _____
Company _____ State/Country _____
Department _____ Zip _____

Additional copies of this document are available from:

Digital Equipment Corporation
444 Whitney Street
Northboro, Ma 01532
Attention: Communications Services (NR2/M15)
Customer Services Section

Order No. EK-KT08A-UG-001