PDP-11/45
memory management
reference manual

KT11-C

PDP-11/45
memory management
reference manual

digital equipment corporation · maynard, massachusetts

1st Printing, August 1972
2nd Printing (Rev), October 1973

Copyright © 1972, 1973 by Digital Equipment Corporation

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.

The following are trademarks of Digital Equipment Corporation, Maynard, Massachusetts:

DEC	PDP
FLIP CHIP	FOCAL
DIGITAL	COMPUTER LAB
UNIBUS	

CONTENTS

ILLUSTRATIONS

TABLES

INTRODUCTION

The KT11-C Memory Management Unit is a hardware option designed for use with the PDP-11/45 Programmed Data Processor. This manual:

- Provides an understanding of the KT11-C in a PDP-11/45 system.
- Explains the KT11-C hardware and how it can be used to develop the memory management module of a software operating system.
- Describes the KT11-C logic in sufficient detail to enable maintenance personnel to perform on-site troubleshooting and repair.

The KT11-C interacts with the KB11-A Central Processor Unit and operating system software to achieve PDP-11/45 system memory management objectives. For this reason, a description of memory management system objectives and programming information is included in this manual.

Chapter 1 introduces the purpose and features of the memory management unit.

Chapter 2 describes the implementation of the features from a programming level. It also describes the internal registers and their application, hints, and exceptions of interest to programmers.

Chapter 3 provides a detailed description of the logic. The content and organization of this chapter are based on the block schematics contained in Volume 2 of this manual.

Chapter 4 references the installation and maintenance procedures provided in the *PDP-11/45 System Maintenance Manual* (DEC-11-H45A-D). There are no specific KT11-C installation or maintenance procedures in this manual.

Appendix A is a glossary of terms.

Appendix B is a bibliography of references on operating systems memory management.

The Index is an alphabetical list of subjects with the page number where the subject appears.

Detailed descriptions of the processor, console, Unibus, Fastbus, and memory logic that interface with the memory management unit are provided in the following related documents.

PDP-11/45 System Maintenance Manual	DEC-11-H45B-D
KB11-A Central Processor Unit Maintenance Manual	DEC-11-HKBB-D
MS11 Semiconductor Memory Systems Maintenance Manual	DEC-11-HMSB-D
PDP-11/45 Processor Handbook	112.01071.1876
PDP-11 Unibus Interface Manual (2nd Edition)	DEC-11-HIAB-D

CHAPTER 1
GENERAL DESCRIPTION

This chapter describes the features of the KT11-C in "systems" terms and also includes a specification summary.

1.1 PURPOSE OF OPTION

The KT11-C Memory Management Unit intercepts addresses generated by the processor (before they reach memory), processes the addresses received, and then transmits the processed addresses to memory. Address processing is the main function of the memory management option. This processing or modification of addresses is called *relocation.* Processing is termed relocation because it consists of adding a fixed constant to every processor address. The location of the KT11-C option in the PDP-11/45 system is shown in Figure 1-1.

The terms and definitions contained in Appendix A and referred to in this manual are consistent with industry-accepted definitions.

1.2 PREREQUISITE

The KT11-C Memory Management Unit is required on all systems with more than 28K of main memory (bipolar, MOS, and core). The option should also be considered for systems with real-time and timesharing applications as well as any system that runs user programs under a control or monitor program. For a narrative description of the PDP-11/45 system's ability to support timesharing and real-time operating systems, refer to Paragraph 4.3 in the *KB11-A Central Processor Maintenance Manual.*

1.3 FEATURES

The KT11-C features outlined below are described in detail in this manual. The KT11-C option:

- expands the basic 28K-word memory capability to 124K words.
- provides dynamic read-only and execute-only memory protection.
- provides up to 16 relocatable, variable length pages per processor mode.
- ensures protection of operating system and user programs by implementing separate address spaces for the Kernel, Supervisor, and User modes.
- provides additional advanced memory management capabilities.

1.3.1 Memory Protection

The memory management unit enables the user to protect one section of memory from access or destruction by programs located in another section. The KT11-C divides the memory into sections called *pages.* Each individual page has a protection or access key associated with it that restricts access to the page. With the memory management unit, a page can be keyed *non-resident* (memory neither readable nor writable), *read-only* (no write

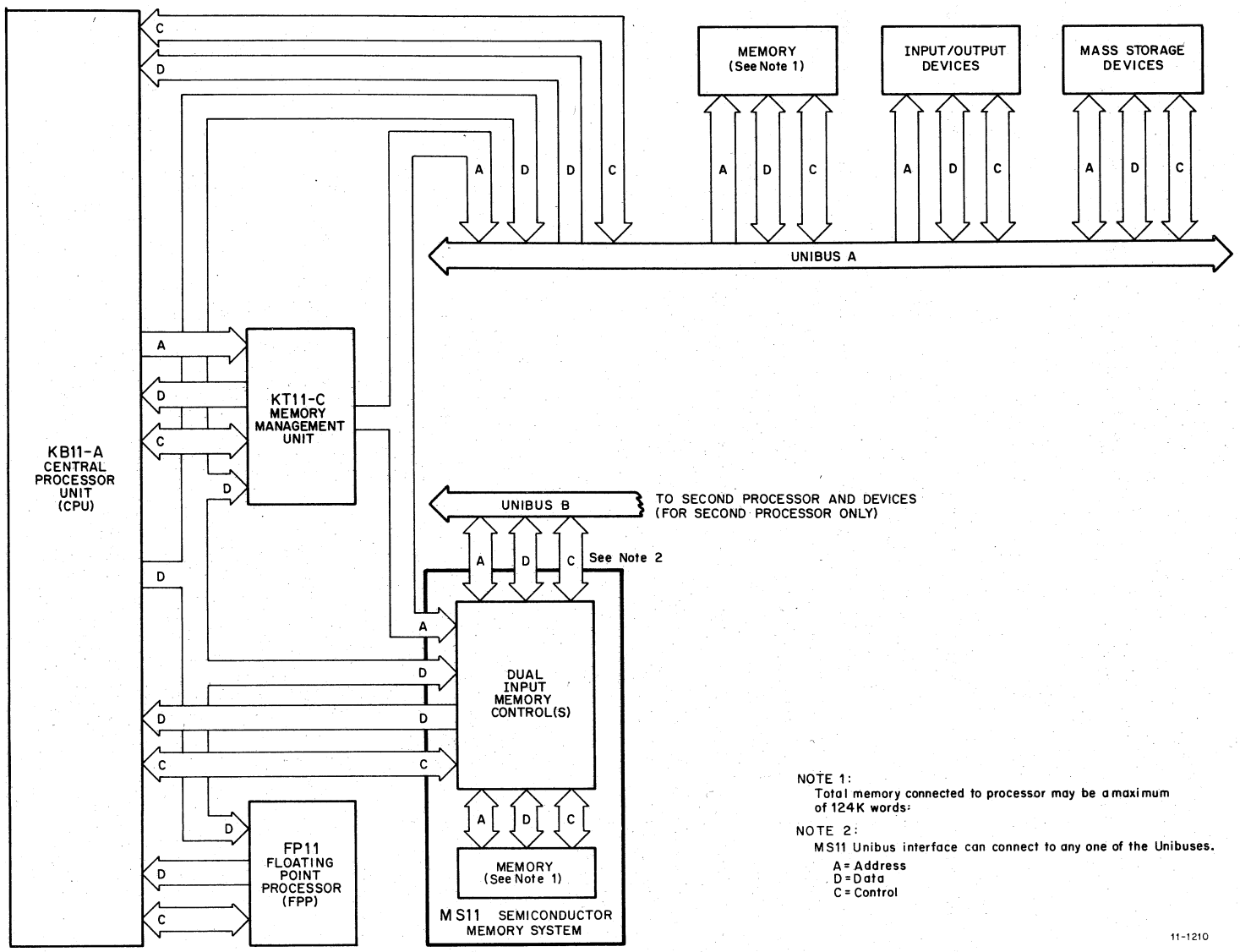

Figure 1-1 System Block Diagram

or store operations to memory), or *execute-only* (only instructions, immediate operands, absolute addresses, and index words can be accessed from memory). These three types of protection in association with other features of the KT11-C enable the user to develop an ultrareliable computer operating system. With the non-resident key, memory not specifically assigned to a program can be made unavailable to it. As a result, program errors are unable to execute unwanted material left over from some other process, and they cannot modify any other programs. The read-only key protects data bases and pure code sections from malicious or accidental destruction while allowing them to be accessed. With User mode programs, the execute-only feature allows proprietary codes to be executed but not copied.

1.3.2 Relocation/Virtual Memory

Often it is desirable to load a program into one set of locations in memory and then execute it as if it were located in another set of locations, e.g., when several user programs are simultaneously stored in memory. When any one program is running, it must be accessed by the processor as if it were located in the set of addresses beginning at 0. This process is called *relocation.* When the processor accesses program location 0, an address base (base address) is added to the address; thus, the relocated 0 location of the program is accessed. This same base address is added to all references while the program is running. A different base address is used for each of the other programs in memory.

Processor-generated addresses differ from those that address memory; thus the processor addresses are sometimes termed *virtual addresses,* and the memory addresses termed *physical addresses.* The memory management option specifies relocation on a page basis, which allows a large program to be loaded into discontiguous pages in memory. This ability eliminates the need to shuffle programs to accomodate a new one. It also minimizes unusable memory fragments, allowing more users to be loaded in a specific memory size.

In timesharing systems with swapping, relocation eliminates the need to "relink" a program when it is swapped into a different memory location.

1.3.3 Memory Expansion

The relocated address is an 18-bit address that can access 128K words of address space, enabling the memory management option to expand the accessable address space of a program from 32K to 128K. Expanding the address space permits larger programs to be handled and allows several programs to occupy the memory at once. In addition, the KT11-C option provides for expansion into multiprocessor systems where typically the total memory exceeds 28K words.

1.3.4 Variable Size Pages

A program and its data may occupy as many as 16 pages in the memory. The size of each page varies and each page can be any multiple of 32 words up to 4096 words in length. These features enable small areas in memory to be protected, i.e., stacks, buffers, etc., and also enable the last page of a program exceeding 4K words to be of adequate length to protect and relocate the remainder of the program. As a result, the page fragmentation problem inherent with fixed length pages is eliminated. With the relocation mechanism, the base address of each page can be any multiple of 32 words in the 128K physical address space, thus ensuring compacted code. Finally, the variable page size enables pages to be dynamically changed at run time.

1.3.5 Page State Information

The memory management unit provides two bits of active page state information: an "accessed" bit and a "written into" bit. These bits are read by the operating system and indicate whether the page has been accessed

and, if so, whether it was written into. The accessed bit is used with operating system programs to determine which page should be overlaid with the new program page in systems that swap programs back and forth from a disk. The written into bit is used to determine whether the page to be overlaid must be swapped back to the disk or whether it is identical to a copy already there.

1.3.6 Instruction/Data Spaces

The memory management unit can relocate data and instruction references with separate base address values; thus, it is possible to have a user program of 64K words consisting of 32K of pure procedure and 32K of data. Moreover, a convenient means of building reentrant shared programs is provided (these programs keep a separate data area for each user). The ability of the KT11-C option to relocate information with separate base address values enables shared compilers, assemblers, editors, and supervisors to be developed, in addition to providing an "execute-only" form of protection. With this form of protection, alterable data is automatically separated from reentrant code, and it is impossible to read any information relocated by an instruction base address as data.

1.3.7 Kernel/Supervisor/User Spaces

The KT11-C provides three separate sets of pages (spaces) for use in the processor's Kernel, Supervisor, and User modes. These sets of pages increase system protection by physically isolating User programs from service Supervisor programs and the basic Kernel program. The service programs (compilers, editors, file system, assemblers, etc.) are also separated from the Kernel program (exception handling, I/O, memory management, etc.). Separate relocation register sets greatly reduced the time necessary to switch context between modes. The 3-space (page) construction also aids the user in designing an operating system that has clearly defined communications, is modular, and is easily debugged and maintained. During development cycles, these features result in time and cost savings; in the final system design, they result in an efficient and reliable system.

1.3.8 Program vs Stack Pages

PDP-11 stacks expand by pushing words into lower addresses and thus grow downward; procedure sections increase by growing upward into higher addresses. All memory pages can be expanded by adding lower addresses (stack) or higher addresses (procedure, data). As a result, both stack and program pages are easily dynamically expanded.

1.3.9 Fault Recovery

Four status registers record all information necessary to recover from a page fault. This information comprises the page number that faulted, the type of violation that caused the fault (exceeded length, read-only violation, etc.), and all information needed to easily restart the faulting instruction once the offending address has been made resident in memory.

1.3.10 Statistical Traps

Three protection keys will cause a trap, i.e., an automatic transfer of program control to location 250 at end of current instruction. The trap feature is useful for gathering frequency-of-page-use statistics. One protection key traps on a read access to a read-only page; a second key traps on either a read or a write access from a read/write page; and the third key traps only on a write to a read/write page.

1.4 KT11-C MEMORY MANAGEMENT UNIT SPECIFICATIONS

Table 1-1 is a summary of specifications and technical characteristics of the KT11-C.

Table 1-1
Abridged Specifications Summary

Characteristic	Specification or Description
Memory Expansion	Expands PDP-11/45 memory address capability up to 124K words.
Interface	Address line outputs compatible with PDP-11 Unibus and PDP-11/45 Fastbus.
Timing	Timing derived from basic KB11-A processor TIG module.
Delay	Adds 90 ns to every memory reference when enabled.
Modes of Operation	Implements the KB11-A processor Kernel, Supervisor, and User modes.
Available Pages	Provides sixteen 4K-word pages for each mode. (Eight for I space and eight for D space.)
Page Length	A page can vary in length from one 32-word block up to 128 32-word blocks. Maximum page length is therefore 4096 words.
Program Capacity	Eight 4096-word pages will accommodate 32K-word programs. Use of I and D space can provide 64K-word capacity (32K words of program and 32K words of data).
Page Fault Recovery	Contains status registers that allow full recovery from page faults.
Physical Description	Option consists of two standard hex modules (15 X 8.5 in.) that mount in PDP-11/45 CPU backplane assembly.
SAP Module M8107	Located in slot 14. Replaces SJB Module M8116, when installed.
SSR Module M8108	Located in slot 13.
Power Requirements	Provided by PDP-11/45 power system.
SAP Module M8107	3A, +5.0 Vdc
SSR Module M8108	3.3A, +5.0 Vdc
Environmental	Refer to overall PDP-11/45 specifications listed in *PDP-11/45 System Maintenance Manual*, DEC-11-H45A-D.

CHAPTER 2
OPERATION AND PROGRAMMING

This chapter describes the relocation, protection, and abort mechanisms of the KT11-C, all registers available to the programmer, and operating hints and procedures.

2.1 BASIC KT11-C MECHANISMS

2.1.1 Address Relocation Mechanism

The current processor mode (Kernel, Supervisor, or User) bits (bits 14 and 15 in the processor status word) select one of three sets of 16 registers to serve as the page base on the current access. The logic, in combination with the PDP-11/45 processor address, selects either the subset of eight instruction base registers or the subset of eight data base registers within the set. All instructions, index words, absolute addresses, and immediate operands use the instruction base registers. All other references use the data base registers. Bits 13, 14, and 15 of the processor address are then employed as a 3-bit encoded index into the eight register subsets previously selected. This encoded index selects a specific base (relocation) register, and thus for each memory reference, one of 48 base registers is selected to perform the address relocation.

The content of the selected base register is an initial or base value in the physical address space. The base value is always a multiple of 32 words; as a result, the lowest 6 bits of the base address (bit 0 specifies byte address on the PDP-11) are always 0. In actuality, only the upper 12 bits of the base address are stored in the base registers because the lower 6 bits are by implication 0.

To form the final physical address, a displacement from the base specified by the lowest 13 bits of the processor address (the upper 3 bits have been stripped off as an index into the base registers) is added to the base value contained in the base register. Note that this mapping technique allows pages of variable length (32 to 4096 words) to be packed efficiently in physical memory. Also note that the use of pages with less than 4096 words will result in discontiguous virtual address spaces because in the virtual address space a page begins on a 4096-word boundary.

As a specific example consider program A, starting address 0, is relocated by the constant 6400, which provides an address of 6400 (Figure 2-1). If the next processor virtual address is 2, the relocation constant will then cause physical address 6402_8, which is the second item of program A to be accessed. When program B is running, the relocation constant is changed to 1000_8. Then program B virtual addresses starting at 0 are relocated to access physical addresses starting at 100000_8. Using the active page address registers to provide relocation eliminates the need to "relink" a program each time it is loaded into a different physical memory location. To the processor, the program always appears to start at the same address. Note that the base address is stored in a Page Address Register (PAR).

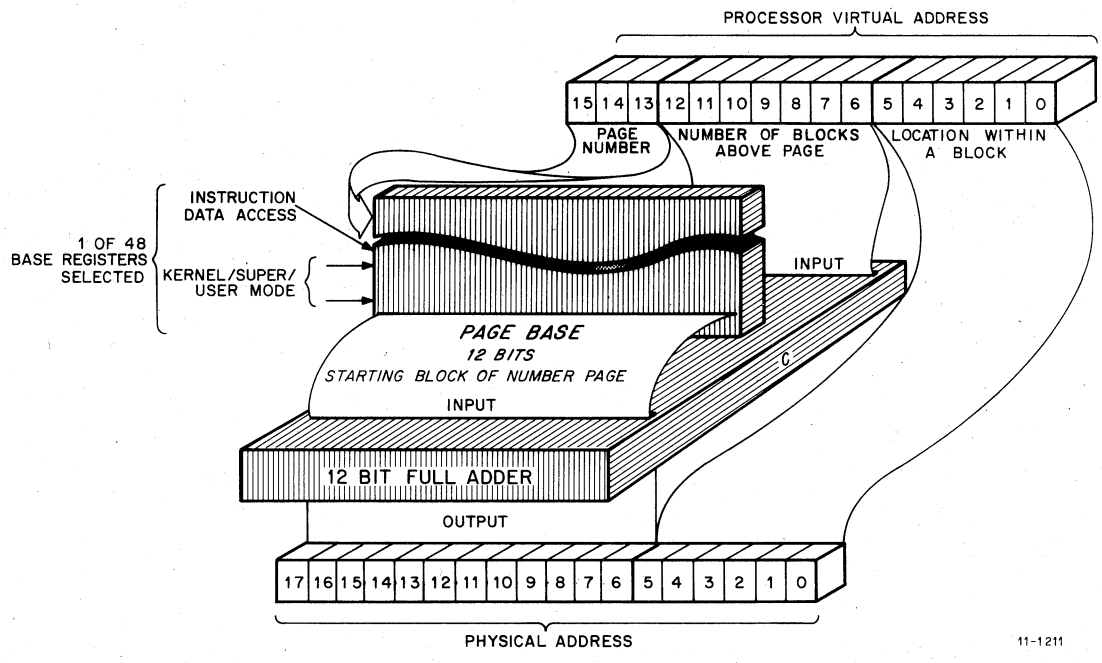

Figure 2-1 Basic KT11-C Relocation Mechanism

The relocation example shown in Figure 2-2 illustrates several points about memory relocation. These points are:

a. Although the program appears to be in contiguous address space to the processor, the 32K-word virtual address space is actually relocated to several separate areas of physical memory. As long as the total available physical memory space is adequate, a program can be loaded. The physical memory space need not be contiguous.

b. Pages may be relocated to higher or lower physical addresses, with respect to their virtual address ranges. In Figure 2-2, page 1 is relocated to a higher range of physical addresses, page 4 is relocated to a lower range, and page 3 is not relocated at all (even though its relocation constant is non-zero).

c. All of the pages in the example start on 32-word boundaries.

d. Each page is relocated independently. There is no reason two or more pages could not be relocated to the same physical memory space. Using more than one page address register in the set to access the same space is one way of providing different memory access rights to the same data, depending on which part of a program was referencing that data. Further information on memory protection is provided in Paragraph 2.1.2. In Figure 2-2, note that the same relocation constant is assigned to Pages 4 and 6. As a result, virtual addresses within both address ranges access the same physical addresses in memory, using separate page address registers.

When the KT11-C Memory Management Unit option is added to the PDP-11/45 system, the 16-bit KB11-A address output is no longer interpreted as the direct physical address of a device or a memory location. Instead, it is considered a 16-bit virtual address that contains information to be used by the KT11-C to construct an 18-bit physical address. Figure 2-3 shows how the 18-bit physical address is constructed. Virtual address bits ⟨15:13⟩ are interpreted as an active page field to select one of eight page registers in a set. Virtual address bits ⟨12:06⟩ provide the block number (0 to 177_8) within the page; VA ⟨05:00⟩ indicate the displacement within each 32-word block. The Page Address Register (PAR) contains a Page Address Field (PAF) that is written into the PAR under program control when the program page is defined. Consider the PAF as the base address of the page.

The block number, VA ⟨12:06⟩, is added to the base address PAF ⟨11:00⟩ to provide the 12 most significant bits of the physical address. This address and virtual address bits VA ⟨05:00⟩ (unchanged by relocation) form the 18-bit physical address.

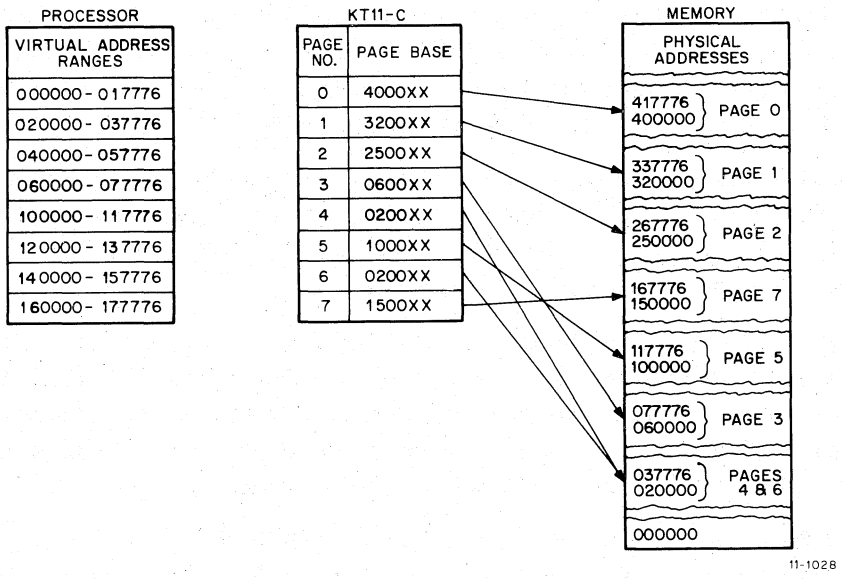

Figure 2-2 Relocation of a 32K-Word Program into 124K-Word Physical Memory

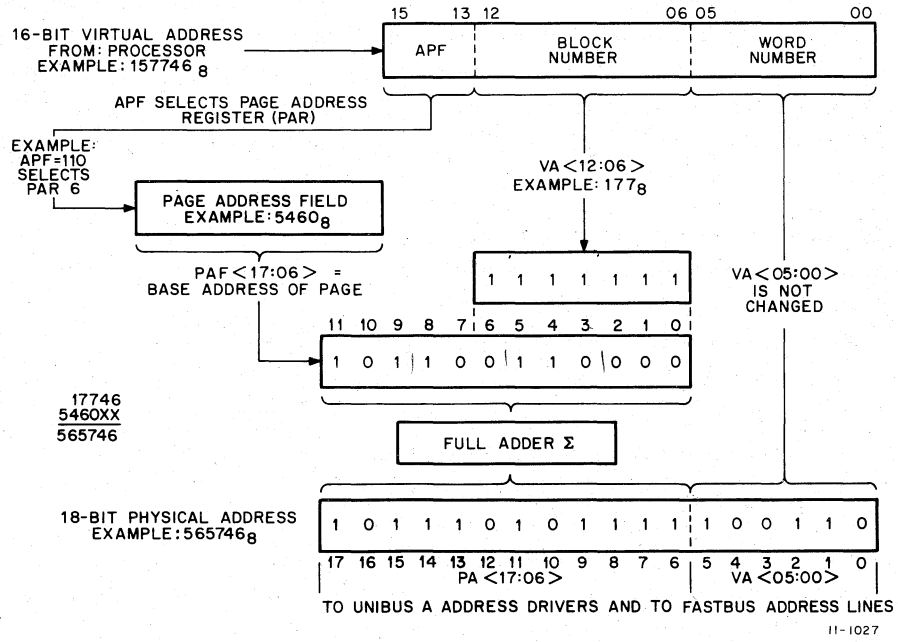

Figure 2-3 Construction of an 18-Bit Physical Address

2.1.2 Memory Protection Mechanisms

Three address protection mechanisms in the KT11-C provide non-resident and read-only protection, execute-only protection, and internal protection for the three processor mode address spaces.

2.1.2.1 Non-Resident/Read-Only Protection – A Page Descriptor Register (PDR) is selected in the same manner as a Page Address Register (PAR). After the selection occurs, three bits from the PDR are decoded as an access key. If the access rights designated by the key are inconsistent with the current memory reference, the memory reference is not completed and an abort to Kernel D space 250 occurs.

When the access key is set to 0, the page is defined as non-resident, and an immediate abort prevents any attempt by a program to access a non-resident page. Using this feature to provide memory protection, only those pages associated with the current program are set to legal access keys. The access control keys of all other program pages are set to 0, which prevents illegal memory references.

The access control key for a page can be set to 2, allowing read (fetch) memory references to the page but immediately aborting any attempt to write into the page. This read-only type of memory protection can be afforded to pages that contain common data, subroutines, or shared algorithms. This type of memory protection makes certain that access rights to a given information module are user-dependent, i.e., the access right to a given information module may be varied for different users by altering the access control key.

A Page Address Register in each of the sets (Kernel, User, and Supervisor modes) may be set up to reference the same physical page in memory and each may be keyed for different access rights. For example, the User access control key might be 2 (read-only access), the Supervisor access control key might be 0 (non-resident), and the Kernel access control key might be 6 (allowing complete read/write access).

2.1.2.2 Execute-Only Protection – The execute-only type of memory protection is part of an overall ability to use reentrant software, which prevents excessive use of memory space when a program is provided for several users. Such programs can be written in two parts. One part contains pure code that is not modified during execution and can be used to simultaneously service any number of users. For example, the pure code portion of FORTRAN can service multiple FORTRAN users. A separate second part of the program belongs strictly to each user and consists of the code and data that is developed during the compiling process. This portion is stored in a separate page of memory. Figure 2-4 shows a comparison of memory space for non-reentrant and reentrant systems.

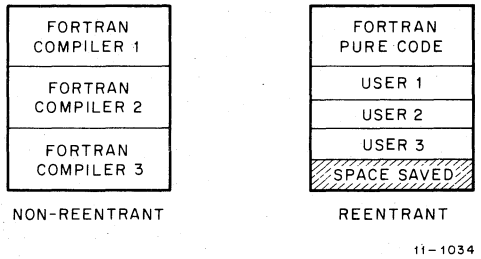

Figure 2-4 Comparison of Non-Reentrant and Reentrant Program Space Requirements

The KT11-C hardware can differentiate between pure code memory references (which are instructions, immediate operands, index words, and absolute addresses) and all other memory references. When this feature is enabled under program control, the KT11-C will take the "pure code" portion of such a program from an I (instruction) space page. All other data will be put into an associated D (data) space page. Any illegal attempt by an unauthorized user to read from the execute-only I space will be relocated to a separate D space page. Because the I and D space pages have separate PDRs, access control can be keyed differently for each page. For example, the D space access control key could be set to 0 to cause a non-resident abort, or, as an alternative, the base address of the D space page could be set up to map over some other part of a user's program. In any case, the D space PDR must not be set so that a user can write into the pure code I space. The difference between the read-only protection described in Paragraph 2.1.2.1 and execute-only

protection is that the contents of execute-only pages cannot be read as data. This feature is particularly useful in protecting proprietary programs from some users. Paragraph 2.4.6 describes I and D space more fully.

In the special case when in User mode and the previous mode specified by the processor status is also User mode, the action of the MFPI instruction is modified so that the read is via the D relocation registers. In this manner, the integrity of the execute-only mode is reserved.

2.1.2.3 Multiple Address Space Protection – The three completely separate PAR/PDR sets provided by the KT11-C for each mode of processor operation (Kernel, Supervisor, and User) give the timesharing system another type of memory protection. The mode of operation is specified by the Processor Status Word current mode field.

The active page register sets are enabled as follows:

PS ⟨15:14⟩	PAR/PDR Set Enabled
00	Kernel mode
01	Supervisor mode
10	Illegal (all references aborted)
11	User mode

Thus, a User mode program is relocated by its own PAR/PDR set, as are Kernel and Supervisor programs. It is therefore impossible for a program running in one mode to accidentally reference space allocated to another mode when the active page registers are set correctly by a monitor program. For example, a user cannot transfer to Supervisor or Kernel space. The Supervisor mode address space may be reserved by the system to accommodate resident compilers, utility programs, and other shared resource programs. The Kernel mode address space may be reserved for resident system monitor functions, such as the basic Input/Output Control (IOC) routines, memory management trap handlers, abort handlers, and timesharing scheduling module. By dividing the types of timesharing system programs functionally between the Kernel, Supervisor, and User modes, a minimum amount of space control housekeeping is required as the timeshared operating system sequences from one user program to the next. For example, only the User PAR/PDR set needs to be updated as each new user program is serviced. The three PAR/PDR sets implemented in the KT11-C Memory Management Unit option are shown in Figure 2-5.

2.1.3 Memory Management Statistics Mechanism

A timeshared system swaps programs or parts of programs in and out of memory using secondary storage facilities such as disk or drum systems. In a swapping environment, the operating system must provide the software routines that decide what programs should be swapped and when and how these programs can be swapped between memory and secondary storage. The operating system routines can be simple or complex depending on system requirements, e.g., the amount of overhead time that can be tolerated. The operating system may also have to decide which active page is least likely to be required in the immediate future and may therefore be swapped out to make memory space available for a new program. To make such a memory management decision, the operating system requires statistics on the use of active pages. Some indication of whether a program has been modified during its residence in memory is also desirable. If it has been modified, the modified program must be swapped (re-written) into secondary storage. If no modification has been made and the program can always be re-called from secondary storage, the space it occupies in memory can be overlayed without swapping delay. The KT11-C logic provides the kind of information required by an operating system to gather memory management statistics on the use of active pages. The availability of this information in the hardware reduces the overhead time of any routine, simple or complex, in the efficient management of memory. In the

KT11-C, the Page Descriptor Register associated with each active page includes a W (written into) and an A (attention) bit. When any active page is written into, the W bit is set by the logic; therefore, by testing the W bit the memory management software routine can decide whether a page can be overlayed or if it needs to be swapped out.

The A bit has several uses. To use this feature the system programmer first enables the memory management trap logic. He can then set the access control keys of the active pages of interest for special trap conditions. Access control keys are provided to cause:

a. Memory management trap on read (including instruction fetch)

b. Memory management trap on write

c. Memory management trap on read or write.

Then, the A bit for the active page is set when the page is accessed as specified and a resultant memory management trap occurs. The vector at trap location 250 Kernel address space causes the operating system routine to service the memory management trap. The routine can test the A bit to accumulate statistics on the use of that page. When a swapping decision is required of the operating system, these statistics can be examined to determine the more active pages (which might therefore be retained in memory).

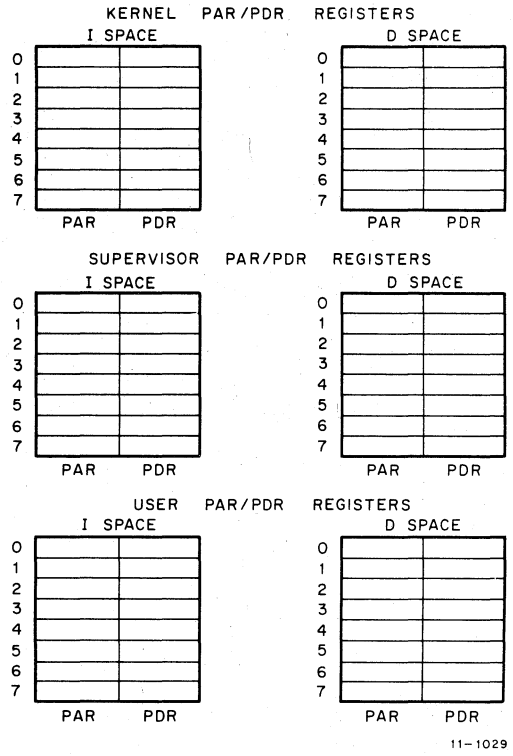

Figure 2-5 KT11-C Memory Management Unit PAR/PDR Registers

2.1.4 Trap/Abort Mechanism

Memory references that violate the protection keys set in the KT11-C cause an interrupt in the processor. The interrupt process is described in Paragraph 5.3 of the *PDP-11/45 Handbook.* In KT11-C aborts and traps, the new PC for the abort/trap service routine is taken from 250 in Kernel virtual D space, while the new PS is taken from 252. No other interrupts use 250 as a trap vector. Note that both abort and statistical traps are taken to 250.

Non-resident, read-only, and page length violations cause an interrupt to the processor before the reference is made. Memory management traps occur only at the end of complete instructions. Paragraphs 3.11.1 and 3.11.2 describe the trap/abort logic and include two system diagrams that indicate how the interrupt is implemented in the logic. The order of interrupt service is listed in Appendix C of the *PDP-11/45 Processor Handbook.*

2.2 PAR/PDR REGISTERS

The contents of the Page Address Register (PAR) and the Page Descriptor Register (PDR) describe a memory page.

The KT11-C provides three sets of 16 PAR/PDR pairs. As indicated in Figure 2-5, one PAR/PDR set is used to reference memory while the processor is in Kernel mode, another in Supervisor mode, and the third in User mode. Each set is subdivided into two groups: one for reference to instruction (I) space and one for reference to data (D) space. Figure 2-5 shows the organization of the three sets. Each pair consists of a Page Address

Register (PAR) and a Page Descriptor Register (PDR). These registers are always used as a pair and contain all the information required to locate and describe the current active pages for each mode of operation.

The current mode bits (bits 14 and 15) of the Processor Status Word determine which set will be referenced for each memory access. A program operating in one mode cannot use the PAR/PDR sets of the other two modes to access memory. Thus, the three sets are a key feature in providing a fully protected environment for a time-shared multiprogramming system. The virtual address value determines which page within a set is to be used. This correspondence is listed below:

Page No.	Virtual Addresses
0	000000 – 017777
1	020000 – 037777
2	040000 – 057777
3	060000 – 077777
4	100000 – 117777
5	120000 – 137777
6	140000 – 157777
7	160000 – 177777

Finally the use of the reference (part of an instruction or data access) determines whether the I space PAR/PDR set or the D space PAR/PDR set is used. Each PAR and PDR of every set is assigned to a specific processor I/O address. Table 2-1 is a complete list of address assignments.

NOTE
Unibus devices cannot access PARs or PDRs.

In a fully-protected multiprogramming environment, only a program operating in Kernel mode would be allowed to access the PAR and PDR locations for the purpose of mapping a user's programs. However, there are no restraints imposed by the KT11-C logic that will prevent Supervisor or User mode programs from accessing these registers. PAR and PDR are not cleared by INIT. The option of implementing such a feature in the operating system and thus explicitly protecting these locations from user's programs is available to the system software designer.

2.2.1 Page Address Registers (PAR)

The Page Address Register (PAR) shown in Figure 2-6 contains the base address of the page in the form of a 12-bit Page Address Field (PAF). Bits 15–12 of the PAR are not implemented in the hardware.

The PAR can also be thought of as a relocation register containing a relocation constant or as a base register containing a base address. Either interpretation indicates the basic function of the PAR in the relocation scheme.

Note that the PAF is interpreted in address calculations as a multiplier of 32, i.e., when used as a base register, the lowest 6 bits are assumed to be 0. The bit stored in bit 0 of the PAF becomes bit 6 of the page base address, bit 1 of the PAF, bit 7 of the page base address, etc. Thus, bit 11 of the PAF becomes bit 17 of the page base address.

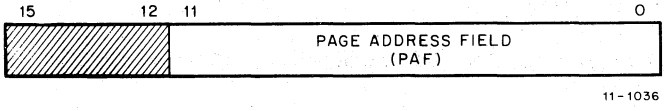

Figure 2-6 Page Address Register (PAR) Format

Table 2-1
PAR/PDR Address Assignments

Kernel					
I Space			D Space		
No.	PAR	PDR	No.	PAR	PDR
0	772340	772300	0	772360	772320
1	772342	772302	1	772362	772322
2	772344	772304	2	772364	772324
3	772346	772306	3	772366	772326
4	772350	772310	4	772370	772330
5	772352	772312	5	772372	772332
6	772354	772314	6	772374	772334
7	772356	772316	7	772376	772336
Supervisor					
I Space			D Space		
No.	PAR	PDR	No.	PAR	PDR
0	772240	772200	0	772260	772220
1	772242	772202	1	772262	772222
2	772244	772204	2	772264	772224
3	772246	772206	3	772266	772226
4	772250	772210	4	772270	772230
5	772252	772212	5	772272	772232
6	772254	772214	6	772274	772234
7	772256	772216	7	772276	772236
User					
I Space			D Space		
No.	PAR	PDR	No.	PAR	PDR
0	777640	777600	0	777660	777620
1	777642	777602	1	777662	777622
2	777644	777604	2	777664	777624
3	777646	777606	3	777666	777626
4	777650	777610	4	777670	777630
5	777652	777612	5	777672	777632
6	777654	777614	6	777674	777634
7	777656	777616	7	777676	777636

2.2.2 Page Descriptor Registers (PDR)

The Page Descriptor Register (PDR) contains page expansion, direction, page length, and access control (Figure 2-7).

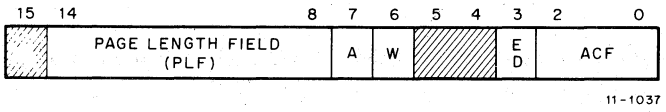

Figure 2-7 Page Descriptor Register (PDR) Format

2.2.2.1 Access Control Field (ACF) – The ACF is a 3-bit field (occupying bits 2–0 of the PDR) that describes the access rights to this particular page. The access codes or "keys" specify the manner in which a page may be accessed and whether or not a given access should result in a trap or an abort of the current operation. A memory reference that causes an abort is not completed and is terminated immediately. Hence an aborted "read" reference does not obtain any data from the location, and an aborted "write" reference does not change the data in the location. A memory reference that causes a trap is completed. When a memory reference causes a trap, the trap does not occur until the entire instruction has been completed. Aborts are caused by attempts to access non-resident pages, page length errors, or access violations, such as attempting to write into a read-only page. Traps are used as an aid in gathering memory management information.

In the context of access control, the term *write* is used to indicate the action of any instruction that modifies the contents of any addressable word. A write is synonymous with what is usually termed a *store* or *modify* in many computer systems. Table 2-2 lists the ACF keys and their functions. The ACF is written into the PDR under program control.

Table 2-2
Access Control Field Keys

ACF	Key	Description	Function
000	0	Non-resident (NR)	Abort any attempt to access this non-resident page.
001	1	Read only and trap (RROT)	Trap to location 250 in Kernel D space after read. Abort any attempt to write into this page.
010	2	Resident read only (RRO)	Abort any attempt to write into this page.
011	3	Illegal	Unused. Reserved for future use. Abort any access attempt.
100	4	Resident read/write and trap (RRWT)	Memory management trap to location 250 in Kernel D space upon completion of a read or write to this page.
101	5	Resident read/write and trap when write (RRWTW)	Allows read/write of page and traps to location 250 in Kernel D space upon completion of a write.
110	6	Resident read/write (RRW)	Read or write allowed. No trap or abort occurs.
111	7	Illegal	Unused. Reserved for future use. Abort any access attempts.

2.2.2.2 Expansion Direction (ED) – The ED bit located in PDR bit position 03 indicates the authorized direction in which the page expands. A logic 0 in this bit (ED = 0) indicates that the page expands upward from relative zero (page base address). A logic 1 in this bit (ED = 1) indicates that the page expands downward toward relative zero (page base address). The ED bit is written into the PDR under program control. When expansion is upward (ED = 0), the page length is increased by adding blocks with higher relative addresses. Upward

expansion is usually specified for program or data pages to add more program or table space. An example of page expansion upward is shown in Figure 2-8.

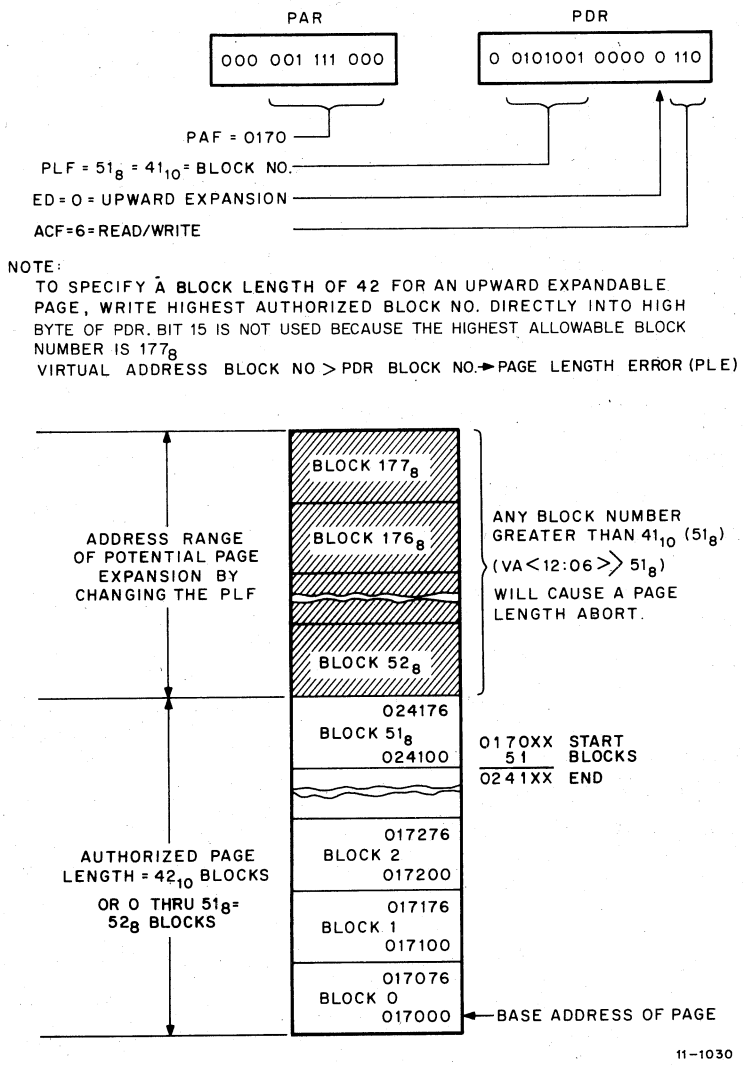

Figure 2-8 Example of an Upward Expandable Page (From the Base)

When expansion is downward (ED = 1), the page length is increased by adding blocks with lower relative addresses. Downward expansion is specified for stack pages so that more stack space can be added. An example of page expansion is shown in Figure 2-9. Paragraphs 2.5.3 and 2.5.4 provide a description of the interaction of the ED bit with the Page Length Field (PLF).

2.2.2.3 **Written Into (W)** – The W bit located in PDR bit position 06 indicates whether the page has been written into since it was loaded into memory. W = 1 is affirmative. Note that the W bit is set independent of the ACF key and the memory management enable bit (bit 9) in SR0. The W bit is automatically cleared when either the PAR or PDR of a page is written into. It can only be set by KT11-C control logic.

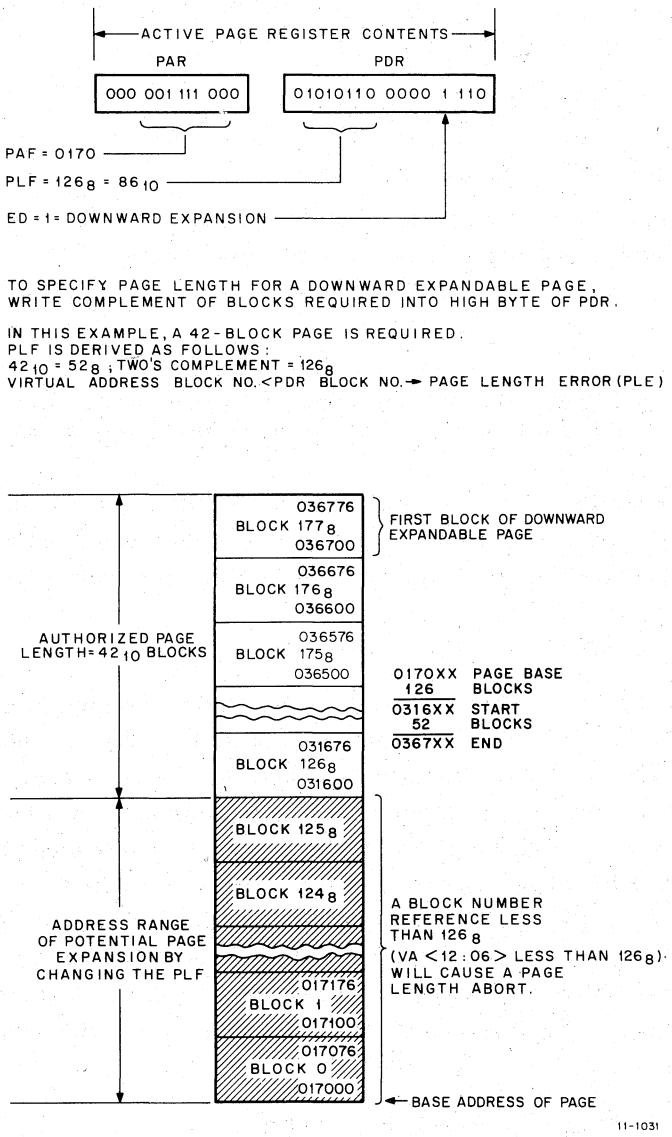

Figure 2-9 Example of a Downward Expandable Page
(From Top of Page Toward Base)

In disk swapping and memory overlay applications, the W bit can be used to determine the pages in memory that have been modified by a user. Those that have been written into must be saved in their current form; those that have not been written into (W = 0), need not be saved and can be overlayed with new pages if necessary.

NOTE

The W bit cannot be set by a memory access of a KT11-C internal register or a memory access that causes an abort.

2.2.2.4 **Attention (A)** – The A bit located in PDR bit position 07 indicates whether any memory page accesses caused memory management trap conditions to be true. A = 1 is affirmative. Trap conditions are specified by the ACF. The following conditions will set the A bit.

1. ACF = 001 and read reference.
2. ACF = 100.
3. ACF = 101 and write reference.

Note that the A bit is set independent of the memory management enable bit (bit 9) in SR0. The A bit is used in the process of gathering memory management statistics for the purpose of optimizing memory use. The A bit is automatically cleared when the PAR or PDR of the page is written into. It can only be set by the KT11-C control logic.

NOTE
The A bit cannot be set by a memory access of a KT11-C internal register or a memory access that causes an abort.

2.2.2.5 **Page Length Field (PLF)** – The 7-bit PLF is located in PDR bits ⟨14:08⟩. It specifies the authorized length of the page in 32-word blocks. The PLF holds block numbers from 0 to 177_8, thus allowing any page length from 1 to 128_{10} blocks. The PLF is written in the PDR under program control.

PLF for an Upward Expandable Page

When the page expands upward, the PLF must be set to one less than the intended number of blocks authorized for that page. For example, if 42_{10} blocks are authorized, the PLF is set to 51_8 (41_{10}) (Figure 2-8). The KT11-C hardware compares the virtual address block number, VA ⟨12:06⟩, with the PLF to determine if the virtual address is within the authorized page length. When the virtual address block number is less than or equal to the PLF, the virtual address is within the authorized page length. If the virtual address block number is greater than the PLF, a page length fault (address too high) is detected by the hardware and an abort occurs. In cases where the authorized page length is less than 4096 words, the virtual address space legal to the program is noncontiguous. This is because the three most significant bits of the virtual address are used to select the PAR/PDR set, and the unauthorized virtual addresses within the page space are "lost". The operating system abort recovery routines might include some form of dynamic memory allocation. Such routines would have to perform the following:

1. Determine the cause of the abort (page length fault).
2. Check the expansion direction bit (upward).
3. Allocate an additional block or more of memory space by incrementing the PLF in the case of an upward expandable page.
4. Correct the general registers modified by the partially completed operation.
5. Return control to the user program and thus allow the now legal virtual address memory reference to be completed.

Note that Paragraph 2.4.10 contains such a recovery routine.

PLF for a Downward Expandable Page

The downward expansion capability for a page is intended specifically for those pages that are to be used as stacks. In the PDP-11, a stack starts at the highest location reserved for it and expands downward toward the lowest address as items are added to the stack. (See Chapter 5 of the *PDP-11/45 Processor Handbook* for

complete details.) When the page is to be downward expandable, the PLF must be set to authorize a page length, in blocks, that starts at the highest address of the page, i.e., always block 177_8. Figure 2-9 shows an example of a downward expandable page. A page length of 42_{10} blocks is arbitrarily chosen so that the example can be compared with the upward expandable example shown in Figure 2-8.

NOTE

The same PAF is used in both examples to emphasize that the PAF, as the base address, always determines the lowest address of the page, whether it is upward or downward expandable.

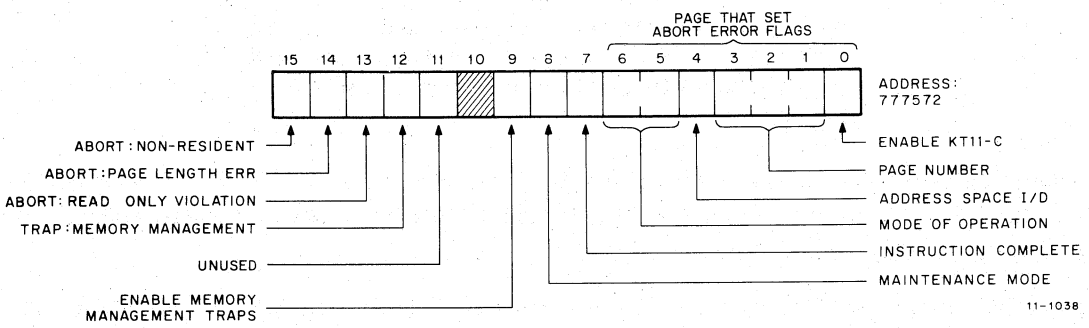

Figure 2-10 Format of Status Register 0 (SR0)

To calculate the value to be placed in the PLF for a downward expandable page proceed as follows.

Take the number of 32-word blocks to be authorized and create the negative of it in 2's complement notation. The 2's complement is formed by negating each binary bit, then adding 1 to the result.

Example:

52_8 = 0101010
bit negation = 1010101
add 1
126_8 = 1010110

The same dynamic memory allocation routine can be used to recover from a page length abort. After determining that the page was downward expandable, the PLF would be decremented to allow downward expansion.

2.3 MEMORY MANAGEMENT STATUS REGISTERS

Aborts and traps generated by the KT11-C logic are vectored through Kernel D space address location 250. Status Registers SR0, SR1, and SR2 can be referenced by fault recovery routines to differentiate between an abort and a trap, determine why the abort or trap occurred, perform the service routines required to recover from the abort or handle the trap, and allow for program restart. The following paragraphs describe the formats of each status register.

2.3.1 Status Register 0 (SR0)

SR0 contains abort error flags, memory management enable and trap flag bits, plus other essential information required by an operating system to recover from an abort or to service a memory management trap. The physical address of SR0 is 777572. The SR0 format is shown in Figure 2-10.

Bits 15–11 are the abort and trap flags, and can be considered to be in a "priority queue" in that "flags to the right" are less significant and should be ignored. For example, a "non-resident" abort service routine would ignore page length, access control, and memory management flags. A "page length" abort service routine would ignore access control and memory management faults.

NOTE

Bits 15, 14, or 13, when set (abort conditions) cause the KT11-C logic to freeze the contents of SR0 bits 1–7 and status registers SR1 and SR2 to facilitate recovery from the abort.

The error flag bits 15–12 are enabled when an address is being relocated by the KT11-C. This implies that either SR0, bit 0 is equal to 1 (KT11-C operating) or that SR0, bit 8 is equal to 1 and the memory reference is the final one of a destination calculation (maintenance/destination mode).

Note that SR0 bits 0, 8, and 9 can be set under program control to provide meaningful memory management control information. However, information written into all other bits is not meaningful. Only that information automatically written into these remaining bits as a result of hardware actions is useful as a monitor of the status of the memory management unit. Setting bits 15–11 under program control will not cause aborts or traps to occur. Bits 15–11 must be reset to 0 after an abort or trap has occurred in order to resume monitoring memory management. Setting bits 15, 14, or 13 will, however, freeze the contents of SR0 bits ⟨1:7⟩, Status Register 1 (SR1) and Status Register 2 (SR2).

2.3.1.1 **Abort-Nonresident** – Bit 15 is the "Abort-Nonresident" bit. It is set by attempting to access a page with an access control field (ACF) key equal to 0, 3, or 7.

2.3.1.2 **Abort-Page Length** – Bit 14 is the "Abort Page Length" bit. It is set by attempting to access a location in a page with a virtual address block number (Virtual Address bits ⟨12:06⟩) that is outside the area authorized by the Page Length Field (PLF) of the PDR for that page. Bits 14 and 15 of SR0 may be set simultaneously by the same access attempt.

2.3.1.3 **Abort-Read Only** – Bit 13 is the "Abort-Read Only" bit. It is set by attempting to write in a page with an access key of 1 or 2.

2.3.1.4 **Trap-Memory Management** – Bit 12 is the "Trap-Memory Management" bit. It is set by a read operation that references a page with an ACF key of 1 or 4 and by a write operation with an ACF key of 4 or 5. The functions of these ACF keys are listed in Table 2-2. Note that the "Enable Memory Management" bit (SR0, bit 9) must be a 1 for the "Trap-Memory Management" bit to be set.

2.3.1.5 **Bit 11** – Bit 11 is a spare flag reserved for future use.

2.3.1.6 Enable Memory Management – Bit 9 is the "Enable Memory Management" bit. It can be set or cleared by doing a direct write into SR0. If bit 9 is cleared, no memory management traps can occur. The A and W bits will, however, continue to log potential memory management traps. When bit 9 is set to 1, the next memory management trap condition will cause a trap, vectored through Kernel D virtual address 250.

NOTE

If the instruction that clears bit 9 (to disable memory management) causes a potential memory management trap in the course of any of its memory references prior to the one actually changing SR0, the trap will occur at the end of the instruction.

2.3.1.7 Maintenance/Destination Mode – Bit 8 specifies maintenance use of the memory management unit. It is used for KT11-C diagnostic purposes. For the instructions used in the initial diagnostic program, bit 8 is set so that only the final destination reference is relocated. This bit must not be used for other purposes.

2.3.1.8 Instruction Completed – When an abort has occured, bit 7 indicates that the current instruction has been completed and that the current memory references are caused by an interrupt or by a trap vector or stack reference of a "trap" instruction (EMT, TRAP, BPT, or IOT).

2.3.1.9 Abort Recovery Information – When an abort occurs, bits ⟨1:6⟩ of SR0 contain the information necessary to determine which of the 48 pages caused the abort. Specifically, bits ⟨1:6⟩ contain the processor mode, whether I or D space, and which 4K page within mode-space.

Mode of Operation

Bits 5 and 6 indicate the CPU mode (User/Supervisor/Kernel) associated with the page causing the abort or trap (Kernel = 00, Supervisor = 01, User = 11). These bits are controlled by the KT11-C logic that decodes current previous mode bits of the PSW.

Address Space I/D

Bit 4 indicates the type of space (I or D) being accessed (0 = I Space, 1 = D Space). It is controlled by the KT11-C logic that selects I/D space.

Page Number

Bits 3–1 contain the page number of a reference. Pages, like blocks, are numbered from 0 upwards. The page number and address space bits are used by the error recovery routine to identify the page being accessed if an abort occurs.

2.3.1.10 Enable KT11-C – Bit 0 is the "Enable KT11-C" bit. When it is set to 1, all addresses are relocated and protected by the memory management unit. When bit 0 is set to 0, the memory management unit is disabled and addresses are neither relocated nor protected. Note that with this bit 0, all programs that run on the PDP-11/20 will run on the PDP-11/45.

2.3.2 Status Register 1 (SR1)

SR1 records any autoincrement/autodecrement of the general purpose registers, excluding implicit changes to the PC. The physical address of SR1 is 777574. SR1 is cleared at the beginning of each instruction fetch and interrupt. When a general register is autoincremented or autodecremented, the register number and the amount

(in 2's complement notation) by which the register was modified is written into SR1 (Figure 2-11). The information contained in SR1 is necessary to accomplish an effective recovery from an abort. The low order byte is written first, and it describes the first general register that was changed. If a second general register is changed by the instruction, that change information is written into the high order byte of SR1. It is not possible for a PDP-11 instruction to autoincrement/autodecrement more than two general registers per instruction before an "abort-causing" reference. Because only three bits are provided for the general register number, it is up to the software to determine which set of registers (User/Supervisor/Kernel-General Set 0/General Set 1) was modified by determining the CPU and Register modes as contained in the Processor Status Word (PSW) at the time of the abort. SR1 is read-only; a write attempt will not modify its contents.

2.3.3 Status Register 2 (SR2)

SR2 is loaded with the 16-bit Virtual Address (VA) at the beginning of each instruction fetch, or with the address Trap Vector at the beginning of an interrupt; "T Bit" trap, Parity, Odd Address, and Timeout traps are also loaded with the trap vector. The physical address of SR2 is 777576. SR2 is read-only; a write attempt will not modify its contents. SR2 is the Virtual Address Program Counter (Figure 2-12).

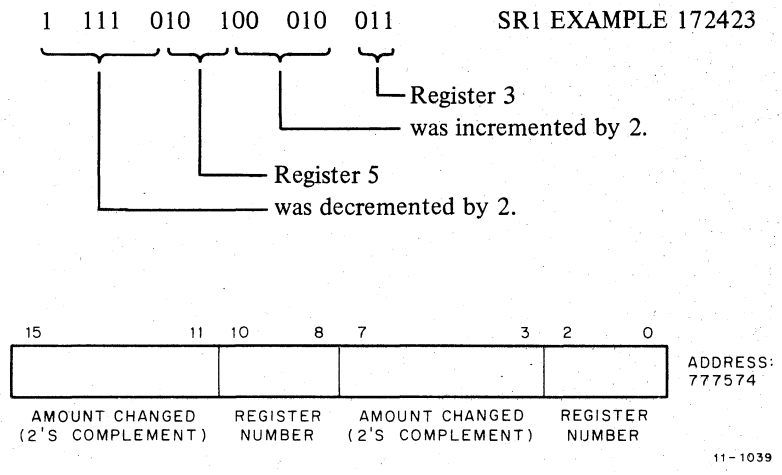

Figure 2-11 Format of Status Register 1 (SR1)

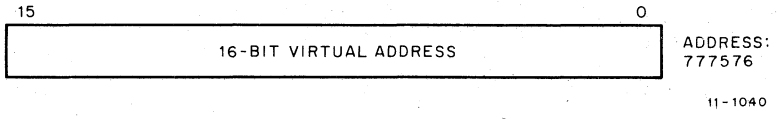

Figure 2-12 Format of Status Register 2 (SR2)

2.3.4 Status Register 3 (SR3)

SR3 is a 4-bit register that stores I/D space control information. The physical address of SR3 is 772516. Data can be written into or read from SR3 under program control. The SR3 format is shown in Figure 2-13. When the ENABLE D SPACE bit for a specific mode (Kernel, Supervisor, or User) is cleared, only the I space PAR/PDR set will be used when operating in that mode. If an ENABLE D SPACE bit is set, then the D space PAR/PDR set for that mode is enabled. Then, if a memory reference is not an instruction fetch, immediate operand, index word, or an absolute address, the D space PAR/PDR set will be used to relocate that memory reference.

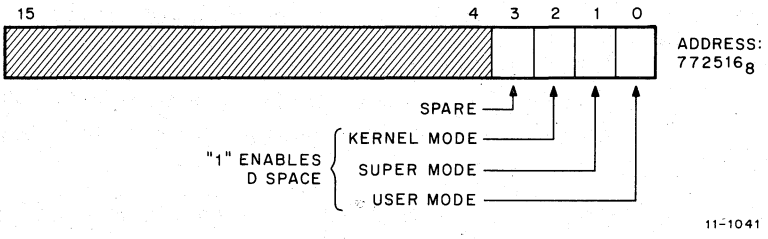

Figure 2-13 Format of Status Register 3 (SR3)

2.4 KT11-C OPERATION

This section contains operational information, including techniques, hints, and cautions.

2.4.1 Console Operations

When the KT11-C option is implemented and enabled in the PDP-11/45 system, console operations are effected as described in the following paragraphs.

2.4.1.1 Single Step Mode – To single step through a program, do not use the console START switch. To do so will disable the KT11-C by clearing SR0. As an alternative, load the PC (R7) with the starting address and press the CONT switch.

2.4.1.2 Address Display – When the KT11-C is enabled, the console address display is determined by the Address Display Select switch position, as indicated in Table 2-3.

2.4.1.3 Stepping Over 32K-Word Boundaries – On Examine Next and Deposit Next operations a carry is not propagated from bit 15 to bit 16.

2.4.2 Physical Address Determination

A 16-bit virtual address can specify up to 32K words in the range from 0 to 177776_8 (word boundaries are even octal numbers). The three most significant virtual address bits designate the PAR/PDR set to be referenced during page address relocation. Table 2-4 lists the virtual address ranges that specify each of the PAR/PDR sets.

To calculate the physical address, disregard the three most significant VA bits and add the remainder to the PAR contents, left-shifted six places.

Example:

VA = 167456 =		xxx0	111	100	101	110
+(PAR) = 3456	= 011	100	101	110		
PA = 355256 =	011	101	101	010	101	110

Where x indicates these bits are not used in the calculation.

Table 2-3
Address Display Select Switch

Switch Position	Interpretation and Use
PROG PHY	Program Physical Address: Physical (relocated) addresses accessed by the program. This position is used for single stepping through the program being relocated. It will always display the actual memory address of the location being accessed.
CONS PHY	Console Physical Operations: Used for examines or deposits to a physical address whether the KT11-C is enabled or not. It will display the address loaded from the console switch register.
USER, SUPER, or KERNEL, I or D	Virtual Address: Used for examines or deposits to the virtual address when the current physical location may be unknown. These positions always display the address generated by the KB11-A processor.

NOTE
A detailed description of the source of individual address bits is provided in Paragraph 3.5.1.

Table 2-4
Relating Virtual Address to PAR/PDR Set

Virtual Address Range	PAR/PDR Set
000000 – 17776	0
020000 – 37776	1
040000 – 57776	2
060000 – 77776	3
100000 – 117776	4
120000 – 137776	5
140000 – 157776	6
160000 – 177776	7

2.4.3 Protection Without Relocation

To obtain a one-to-one mapping of virtual address space to physical address space (no relocation), a value must be placed in each PAR to be accessed, because the three most significant bits of the virtual address are "stripped" to select the PAR/PDR set. Therefore, these three bits must be "mirrored" by the contents of each PAR, as indicated in the chart on the following page.

PAR	Contents
0	000000
1	000200
2	000400
3	000600
4	001000
5	001200
6	001400
7	001600

NOTE
If PAR 7 is to address the I/O page, its contents must be 007600.

2.4.4 Communication Between User/Supervisor/Kernel Address Space

A program in one address space can communicate with a program in another address space using three basic methods. The simplest method is to make a part of each program common by overlapping physical address space. A second method is to use the MTPD, MTPI, MFPD, and MFPI instructions. These instructions are defined in Chapter 4 of the *PDP-11/45 Processor Handbook*. A third method is to use the "return from interrupt" (RTI) instruction. These methods are described in the following paragraphs.

2.4.4.1 Overlapping Physical Addresses – Consider the problem of providing a buffer area to move data to and from I/O devices through the monitor/file system. Without overlapping, the data would be brought into a buffer area located in Kernel data space. It would then be moved into a separate buffer area in User space. However, using KT11-C memory mapping capabilities, a single buffer area can be located in physical memory with provisions for access by User, Supervisor, and Kernel mode programs. This method saves the additional memory space required for separate buffer areas and also the time required to move data from one buffer area to another.

2.4.4.2 MFPI and MTPD Instructions Use – In the following example, a word is popped off the Kernel stack and pushed onto the User stack. This method of stack-to-stack communication is typical of well-defined operating system communication. Assume that the processor is running in the Kernel mode and the previous mode was the User mode. The MFP and MTP instructions are double operand instructions and the missing operand is always the stack of the mode you are in.

```
                                                                    Internal Register
                                                                      Transfers
0065 SS     MFPI    R6          ;GET CURRENT VALUE OF USER STACK POINTER.   -(R6) ← (R17)
                                ;PUSH IT ONTO KERNEL STACK. NOTE            KER SP  USER SP
                                ;THAT SINCE ALL GENERAL REGISTERS ARE
                                ;IN BOTH I AND D SPACE, THIS INSTRUCTION
                                ;COULD HAVE BEEN "MFPD".
            MOV     (R6)+, R1   ;POP USER R6 VALUE OFF KERNEL STACK.        R1 ← + (R6)
                                ;PUT IT IN GENERAL REGISTER 1.                  KER SP
1066 DD     MTPD    -(R1)       ;POP ARGUMENT OFF KERNEL STACK. PUT         (R17) ← -(R1)
                                ;IT ON TOP OF USER STACK.                   USER SP
                                                                            (R1) ← +(R6)
                                                                                KER SP
        .
        ;ADDITIONAL WORDS MOVED HERE
        .
        .
            MOV     R1, -(R6)   ;PUT CORRECTED USER STACK POINTER ON        -(R6) ← R1
                                ;KERNEL STACK.                              KER SP
0066 DD     MTPI    R6          ;POP CORRECTED USER R6 VALUE OFF KERNEL     (R17) ← +(R6)
                                ;STACK AND PUT IN USER R6. (INSTRUCTION     USER SP  KER SP
                                ;COULD HAVE BEEN "MTPD".)
```

The use of MTPI and MFPI is restricted in the User mode. If the previous mode bits ⟨13:12⟩ of the PS word also specify User mode (11), then the following restriction is implemented to preserve the integrity of execute-only protection. The execution of a MFPI will be relocated by the equivalent D space PAR/PDR register set if "D" space for the User mode is enabled (SR3, bit 0 = 1). This prevents a user from using MFPI to read from an execute-only page in his address space.

2.4.4.3 Control Information – Control is passed inward from User and Supervisor modes to the Kernel mode by all traps and interrupts. All trap and interrupt vectors are located in Kernel virtual address space. Thus, all traps and interrupts pass through Kernel space to obtain their new PC and PS, and determine the new mode of processing. Control is passed outward from the Kernel mode to the User and Supervisor modes by the RTT and RTI instructions, which restore the old PC and PS by popping them from the stack.

2.4.5 Statistical Aids Use

Three ACF keys (1, 4, and 5) can cause traps at the end of the current instruction when the Enable Memory Management Traps bit is set in the SR0. These keys also cause the A bit in the associated PDR to be set, regardless of whether the Enable Memory Management Trap bit is set. A PDP-11 double operand instruction can access up to six different pages of memory during execution. If each of the six PDRs had ACF keys that specified memory management traps, then the A bit in each of the PDRs will have been set by the time the instruction is completed and the memory management trap service routine is entered. A statistic gathering service routine that collects "frequency of use" data can scan all PDRs and thus gather A bit information from all six PDRs that might have been set. The status registers are set only once. Therefore, if the statistical facility is to collect valid frequency of use on pages, a more elaborate approach is required of the software service routine.

One approach is to leave the Enable Memory Management Trap bit reset to 0, so that no traps occur, but set the ACF keys to specify the appropriate traps. This will cause the A bit to be set. Then, use a real-time clock to interrogate the A bits. When one is found set, a count location corresponding to that page can be incremented, and then that A bit must be cleared in the PDR.

One way to clear the A bit is to write into the PDR. For example, the instruction:

```
MOV     @#PDR,     @#PDR
```

will clear the A and W bits, but leave all other PDR bits unchanged. A less complex way is to set all read-only page ACF keys equal to 1 and all read/write page ACF keys equal to 4. At the end of the user time period, interrogate the A and W bits. Most state transitions with the ACF can be accomplished by incrementing the PDR. The ACF keys change as follows:

Original ACF Key	New ACF Key	Effect of ACF Key Change
1	2	No memory management trap on read.
4	5	No memory management trap on read.
5	6	No memory management trap on write.
4	6	No memory management trap on a read or write.

Thus, by incrementing the ACF key, characteristics of the original key that are no longer required are deleted by the new key. Other essentials are retained by the new key. For example, when the ACF key increments from 1 to 2, the KT11-C still aborts any attempt to write into that page.

2.4.6 I and D Space Use

NOTE

To use separate I and D space, a programmer must be able to write pure procedure code. The use of I and D space is an advanced technique that should not be attempted by novice programmers.

Eight I space PAR/PDR pairs accommodate up to 32K instruction words and eight D space PAR/PDR pairs accommodate up to 32K data words. By using the separate I and D space PAR/PDR pairs, a maximum 64K-word program capacity is possible. The following rules apply to any separate I and D space programs.

1. I space can contain only instructions, immediate operands (Mode 2, Register 7), absolute addresses (Mode 3, Register 7), and index words (Modes 6 and 7).
2. The stack page must be mapped into both I and D space if the Mark instruction is used (standard PDP-11/45 subroutine calling sequence), because it is executed off the stack.
3. I space only pages cannot contain subroutine parameters, which are data. Therefore, any page that contains standard PDP-11/20 calling sequences for example cannot be mapped into an I space page.
4. The trap catcher technique of putting .+2 in the TV followed by a halt must be mapped into both I space and D space.

The following chart shows the separation of I and D references for all address modes and all registers. Note that all registers (R0–R7) are in both spaces.

Mode	Register	Name		
000	X	Register	INSTRUCTION	I space
001	X	Register Deferred	INSTRUCTION	I space
			DATA	D space
010	0–6	Autoincrement	INSTRUCTION	I space
			DATA	D space
	7	Immediate	INSTRUCTION	I space
			IMMEDIATE DATA	I space
011	0–6	Autoincrement Deferred	INSTRUCTION	I space
			INDIRECT	D space
			DATA	D space
	7	Absolute	INSTRUCTION	I space
			ABSOLUTE ADDRESS	I space
			DATA	D space
100	0–6	Autodecrement	INSTRUCTION	I space
			DATA	D space
	7	DO NOT USE THIS CONSTRUCTION		

(continued on next page)

Mode	Register	Name		
101	0–6	Autodecrement Deferred	INSTRUCTION	I space
			INDIRECT	D space
			DATA	D space
	7	DO NOT USE THIS CONSTRUCTION		
110	X	Index	INSTRUCTION	I space
			INDEX	I space
			DATA	D space
111	X	Index Deferred	INSTRUCTION	I space
			INDEX	I space
			INDIRECT	D space
			DATA	D space

Note that when D space is not enabled for a mode by setting the proper bit in SR3, all memory references are relocated and protected by the I space set of PAR/PDR registers.

2.4.7 I/O Operations

2.4.7.1 Kernel Mode Protection – All I/O operations should be performed by programs running in Kernel space, forcing User and Supervisor programs to call upon a Kernel mode routine to perform their I/O operations. This precaution increases system reliability by preventing possible errors by User and Supervisor programs. This precaution is necessary because the physical addresses issued by NPR devices on the Unibus are not relocated by the KT11-C. Therefore, proper I/O transfers between NPR devices and relocated memory depends upon careful checking of I/O requests by the Kernel mode routines; proper I/O transfers provide relocated memory addresses and ensure that the referenced pages of memory are resident and authorized.

2.4.7.2 Avoiding I/O Lockout – If the Kernel mode is used to handle all I/O operations, the programmer must ensure that the Kernel space to which the I/O routines are assigned never becomes non-resident. If this occurs, it becomes impossible to access the PAR/PDR sets to re-establish residency and to turn the KT11-C off by accessing bit 0 of SR0.

One method of recovering from such a situation is to execute a RESET instruction, which will disable the KT11-C relocation logic and allow direct I/O access again.

2.4.8 Processor Status Word

In addition to the material below consult Paragraph 4.5 in the *KB11-A Maintenance Manual.*

2.4.8.1 Explicit References to PS

a. If the following three instructions cause a page fault on the "write" into the destination address, the PS condition codes may be modified.

```
MOV PS, (dst modes 3, 4, 5, 6, 7)
MFPI PS
MFPD PS
```

This may cause a different PS to be stored when the instruction is restarted.

(continued on next page)

b. The sequence

```
SPL
WAIT
```

should be used with care. Note that the processing of a KT11-C abort (page fault) may occur on the fetch of the WAIT instruction. Hence, priority levels must be carefully controlled or the interrupt that was expected to transfer control from the WAIT instruction may occur before the execution of the WAIT instruction.

2.4.8.2 **Implicit Modification of the PS** – When a KT11-C abort occurs, the PS condition code may have already been modified by the partially completed instruction. However, in all cases except those described in Paragraph 2.4.8.1, restarting the aborted instruction still causes the condition codes to be set to the expected value.

2.4.9 Non-Recoverable Aborts

Instructions that have autoincremented (popped) the Kernel stack pointer cannot be properly restarted, if they cause an abort. When an instruction that pops something off the Kernel stack causes an abort, it cannot be restarted because the abort mechanism pushes two items (old PS and PC) onto the Kernel stack. The first of these pushes wipes out the item that the abort-causing instruction previously tried to pop off the stack.

NOTE
The MTPI and MTPD instructions in particular are non-recoverable if done in Kernel mode to an illegal user area.

2.4.10 Page Fault Recovery

This paragraph presents an approach to recovering from either non-resident page faults or demands for a larger page. The description comprises a flow chart and sample program code with notation and includes only material that directly relates to using the KT11-C status registers properly. Emphasis is placed on proper procedure for restarting once the new page is defined and authorized.

The recovery routine takes one of two paths. The flow chart (Figure 2-14) shows that the correct path is determined by the Instruction Complete (IC) bit in status register SR0. If the IC bit is 1, then the "emulate aborted interrupt" path is taken. If the IC bit is 0, then the "general register backup/instruction restart" path is taken. Remember that the IC bit indicates if the aborted memory access was part of an instruction (IC = 0) or part of an interrupt or trap (IC = 1), as described in Paragraph 2.3.1.8.

2.4.10.1 **Definitions** – Table 2-5 lists the definitions of mnemonics used in Figure 2-14.

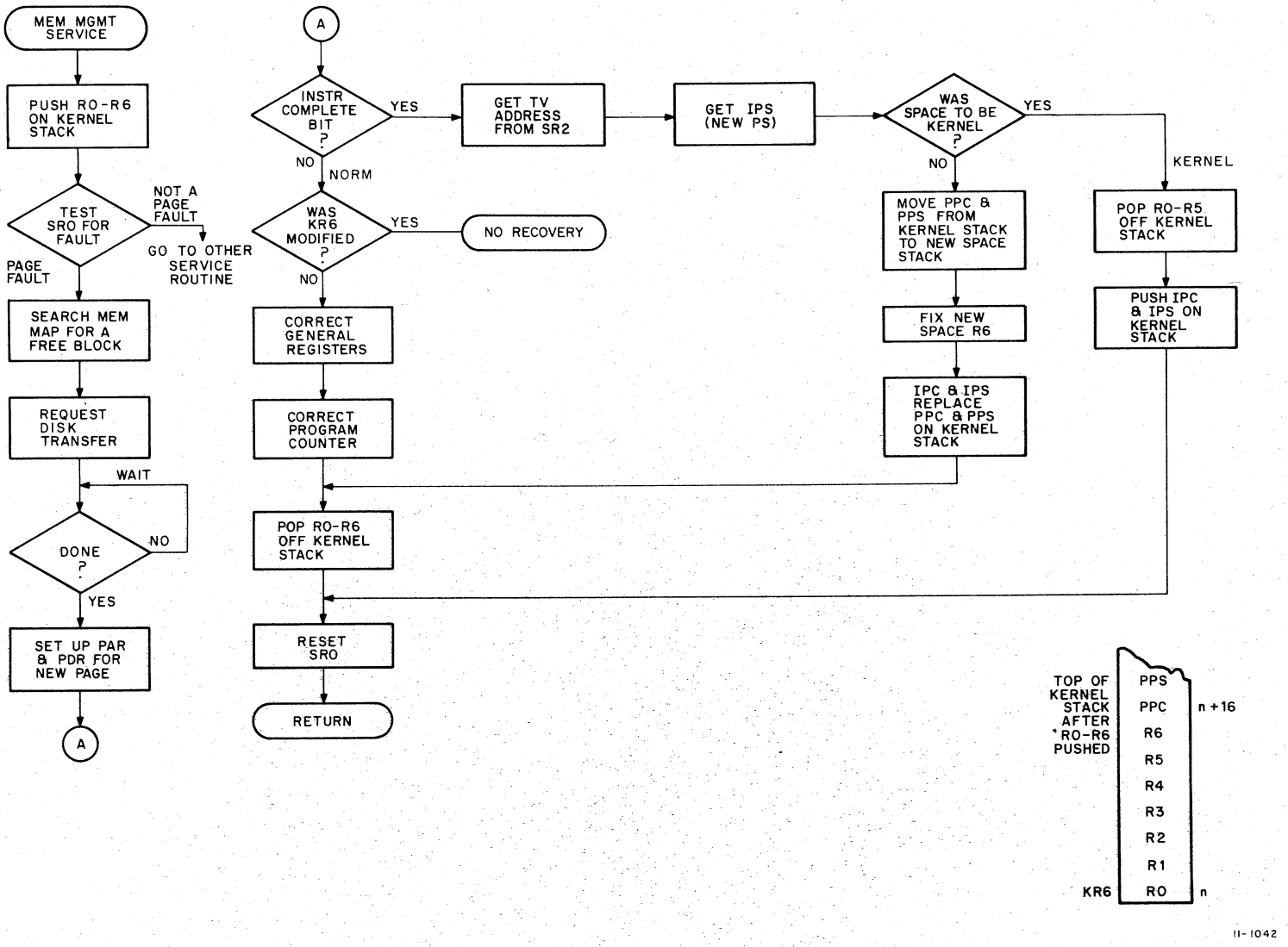

Figure 2-14 Flow Chart for a Page Fault Recovery Routine

Table 2-5
Mnemonic Definitions

Mnemonic	Definition
APC	Abort Program Counter. The value of the contents of location 250. This value normally is the starting address of the KT11-C Abort/Trap Service Routine. Location 250 is the address of the KT11-C Trap Vector.
APS	Abort Processor Status. The value of the contents of location 252. This value is normally the processor status during the KT11-C Abort/Trap Service Routine.
IPC	Interrupt Program Counter. The value contained in the interrupt trap vector location; the starting address of the interrupt service routine.
IPS	Interrupt Processor Status. The value contained in the interrupt trap vector location plus two; the processor status to be set upon servicing the interrupt.
IR6	Interrupt Register 6. Value of the stack pointer for the interrupt service routine; either the Kernel, Supervisor, or User R6 and is specified by bits ⟨15:14⟩ of the IPS.
KR6	Kernel Register 6. The value of the stack pointer when the processor is in Kernel mode.
PPC	Program Program Counter. The value placed on the Kernel stack when the KT11-C causes an abort. If the abort was caused by an interrupt, then the PPC is the location of the next instruction to be executed when control is returned to the program that was interrupted. If the abort was caused by an instruction reference, then the PPC is equal to two more than the value of the last I space reference. In the latter case, the PPC will not be used for any computation.
PPS	Program Processor Status. Value equal to the contents of the PS at the time of KT11-C abort that is placed on the Kernel stack.
PR6	Program Register 6. Value of the stack pointer for the "program". Note that PR6 may be KR6 if the program was in Kernel mode. If the program was in Supervisor mode, PR6 is the Supervisor R6; if the program was in User mode, PR6 is the User R6.

2.4.10.2 Program Example

```
;NOTE THAT REGISTER SET 1 IS ASSUMED TO BE IN USE FOR BOTH THE USER AND KERNEL
;MODES.

START:  MFPD    R6                  ;PUSH PR6 ONTO KERNEL STACK
        MOV     R5,-(R6)            ;PUSH R5-R0 ONTO KERNEL STACK
        MOV     R4,-(R6)
        MOV     R3,-(R6)
        MOV     R2,-(R6)
        MOV     R1,-(R6)
        MOV     R0,-(R6)

        BIT     #140000,@#177572    ;TEST FOR NON-RESIDENT OR PAGE
                                    ;LENGTH ABORT

        BEQ     +6
        JMP     ERROR               ;WAS SOMETHING OTHER THAN NON-RESIDENT
                                    ;OR PAGE LENGTH ABORT.
```

(continued on next page)

```
;"ERROR" ROUTINE IS NOT DESCRIBED IN THIS EXAMPLE. INSERT CODE HERE TO FIND A FREE
;MEMORY BLOCK AND PULL THE PAGE INTO MEMORY FROM DISK OR DRUM SYSTEM. ASSUME
;THE SUBROUTINE LEAVES THE PAR IN R1, THE PAR ADDRESS IN R2, THE PDR IN R3, AND THE
;PDR ADDRESS IN R4.

        MOV     R1, @R2         ;SET UP PAR OF NEW PAGE
        MOV     R3, @R4         ;SET UP PDR OF NEW PAGE

;THIS CODE TESTS FOR WHETHER THE ABORT WAS CAUSED BY AN INSTRUCTION ACCESS OR BY
;AN INTERRUPT ACCESS.

        TSTB    @#177572        ;TEST INSTR. COMP. BIT OF SR0.
        BPL     NORM            ;IF NOT SET, GO TO NORMAL RECOVERY
                                ;ROUTINE.

;THE FOLLOWING CODE OBTAINS LOCATION OF INTERRUPT TRAP VECTOR

        MOV     @#177576, R3    ;MOVE KT11-C SR2 (THE TRAP VECTOR ADDRESS)
                                ;TO R3
        MOV     2 (R3), R4      ;MOVE CONTENTS OF TV + 2 (THE TV PS) TO R4
        BIT     #140000, R4     ;TEST BITS 15:14 OF IPS FOR 0 (=KERNEL MODE)
        BEQ     KERNEL          ;IF KERNEL MODE, GO TO KERNEL ROUTINE.

;THE FOLLOWING ROUTINE OBTAINS THE STACK POINTER (IR6) OF THE SPACE THAT THE IPS IN
;THE TV CALLED FOR. IF KT11-C SR1 HAS LOGGED CHANGES TO THIS R6, THEN IT WILL BE
;CORRECTED. THE ROUTINE THEN PUTS THE PROGRAM PC AND PS (PPC AND PPS) CURRENTLY ON
;THE KERNEL STACK ONTO THE STACK THAT IS CALLED FOR BY THE IPS; THUS PERFORMING THE
;FIRST PART OF "EMULATING" THE INTERRUPT.

        MOV     @#177776, R0    ;PSW TO R0
        MOV     R0, R2          ;COPY PSW TO R2
        BIC     #030000, R2     ;CLEAR OUT PREVIOUS MODE BITS IN COPY OF
                                ;THE PSW.
        ASH     -2, R4          ;SHIFT THE NEW TV PS 2 PLACES RIGHT, PUTTING
                                ;THE TV PS CURRENT MODE BITS IN THE
                                ;PREVIOUS MODE POSITIONS.
        BIC     #147777, R4     ;MASK OUT ALL BUT PREVIOUS MODE BITS IN
                                ;SHIFTED COPY OF IPS.
        BIS     R4, R2          ;A NEW PS CREATED WITH CORRECT PREVIOUS
                                ;MODE IS NOW IN R2.
        MOV     R2, @#177776    ;CHANGE PS SO WE CAN GET TO R6
                                ;SPECIFIED BY IPS.
        MFPD    R6              ;GET PROPER R6
        MOV     (R6) +, R2      ;POP R6 OFF KERNEL STACK, PUT IT IN R2

;THE FOLLOWING CODE CORRECTS THE PROPER R6, IF NECESSARY, TO ACCOUNT FOR THE
;PUSHES OF THE PPC AND PPS ON THE STACK SPECIFIED BY THE IPS CURRENT MODE.

        TST     @#177574        ;CHECK SR1 FOR AUTODECREMENT OF R6 IF IR6
        BNE     ISOK            ;WAS ALREADY CHANGED. (ONLY POSSIBILITY
                                ;IS BY 4 BYTES.) IF SO, IT DOES NOT HAVE TO BE
                                ;CORRECTED.
        SUB     #4, R2          ;DO THE AUTODECREMENT TO COPY OF IR6 in R2
        MOV     R0, R1          ;COPY APS INTO R1 SO THAT WE CAN DETERMINE
                                ;WHETHER PR6 IS IDENTICAL TO R6.
        BIC     #147777, R1     ;LEAVE PMODE BITS OF APS (SPECIFYING MODE
                                ;OF PREVIOUS PROGRAM) IN R1.
        XOR     R1, R4          ;TV MODE BITS IN R4
```

(continued on next page)

```
        BEQ     SAME            ;GO TO "SAME" IF PR6 = IR6.
        MOV     R2,-(R6)        ;PUSH COPY ONTO KERNEL STACK
        MTPD    R6              ;PUT CORRECTED IR6 BACK IN REGISTER.
        BR      ISOK            ;DONE WITH THIS PHASE
SAME:   SUB     #4, 14 (R6)     ;THE PR6 ON THE KERNEL STACK IS THE ONE TO
                                ;BE CORRECTED.
```

;THIS CODE PUSHES THE PPC AND PPS, NOW ON THE KERNEL STACK, ONTO THE STACK SPECIFIED
;BY THE IPS (IR6)

```
ISOK:   MOV     16 (R6),-(R6)   ;PUSH PPC ONTO TOP OF KERNEL STACK
        MOV     22 (R6),-(R6)   ;PUSH PPS ONTO TOP OF KERNEL STACK
        MTPD    (R2)            ;PPS OFF KERNEL STACK
                                ;AND PUSHED ON I STACK.
        MTPD    -(R2)           ;PDC OFF KERNEL STACK
                                ;AND PUSHED ON I STACK.
        MOV     R0, #177776     ;RESTORE APS. GET CORRECT PMODE BITS BACK
```

;THE FOLLOWING CODE PUTS THE IPC AND IPS INTO THE KERNEL STACK IN THE LOCATIONS
;PREVIOUSLY OCCUPIED BY PPC AND PPS. NOTE THE PMODE BITS IN THE TV PS MUST BE PROPERLY
;SET FROM THE CMODE BITS OF THE PPS STILL ON THE KERNEL STACK.

```
        MOV     20 (R6), R0     ;PUT COPY OF PPS INTO R0
        ASH     -2, R0          ;RIGHT SHIFT PPS COPY PLACING CMODE BITS
                                ;INTO PMODE POSITION.
        BIC     #147777, R0     ;LEAVE ONLY PMODE BITS IN R0
        MOV     2 (R3), 20 (R6) ;PUT COPY OF IPS INTO KERNEL STACK WHERE
                                ;PPS USED TO BE.
        BIC     #30000, 20 (R6) ;CLEAR OUT PMODE BITS OF IPS.
        BIS     R0, 20 (R6)     ;SET IPS PMODE WITH VALUE FROM CMODE OF
                                ;PPS.
        MOV     (R3), 20 (R6)   ;PC IN TRAP VECTOR REPLACES PC ON KERNEL
                                ;STACK.
```

;THE FOLLOWING CODE RESTORES THE GENERAL REGISTERS

```
RESTUR: MOV     (R6) +, R0      ;RESTORE R0–R5 FROM THE KERNEL
        MOV     (R6) +, R1      ;STACK
        MOV     (R6) +, R2
        MOV     (R6) +, R3
        MOV     (R6) +, R4
        MOV     (R6) +, R5
        MTPI    R6              ;RESTORE R6 TO USER R6
```

;RESET STATUS REGISTER SR0, RETURN

```
RETURN: MOV     #000001, @#177572   ;CLEAR SR0, SET KT11-C ON.
        RTI                         ;RETURN
```

;IF IT WAS KERNEL, ONLY NON-FATAL ERROR COULD BE TV NON-RESIDENT. THEREFORE, R6
;COULD NOT HAVE BEEN CHANGED BY THE INTERRUPT. THE FOLLOWING CODE POPS R0–R6 OFF
;KERNEL STACK AND PUTS IPC AND IPS ON STACK.

```
KERNEL: MOV     (R6) +, R0      ;POP R0–R6
        MOV     (R6) +, R1
        MOV     (R6) +, R2
        MOV     (R6) +, R3
        MOV     (R6) +, R4
        MOV     (R6) +, R5
```

(continued on next page)

```
        MOV     @#177576, (R6)      ;ITV WRITTEN OVER OLD KR6.
        MOV     @(R6), -(R6)        ;PUSH COPY OF CONTENTS OF ITV ON STACK
                                    ;(IPC).
        ADD     #2, 2 (R6)          ;ITV + 2
        MOV     @2 (R6), 2 (R6)     ;REPLACE ITV + 2 WITH IPS

;THE FOLLOWING CODE TESTS FOR AN AUTODECREMENT OR AUTOINCREMENT OF KERNEL R6.
;IF ONE IS DETECTED, CONTROL PASSES TO WAS6 WHICH CANNOT RESTART THE INSTRUCTION.

NORM:   MOV     @#177776, R3        ;GET PS TO R3
        BIT     #030000, R3         ;TEST FOR PREVIOUS KERNEL MODE
        BNE     CORECT              ;WAS NOT KERNEL
        MOV     @#177574, R4        ;PUT SR1 IN R4
        BIC     #174370, R4         ;LEAVE REG #S IN R4
        CMPB    #6, R4              ;TST FOR KR6, LOW BYTE
        BEQ     WAS6
        SWAB    R4
        CMPB    #6, R4              ;TST FOR KR6, HIGH BYTE
        BEQ     WAS6

;THE FOLLOWING CODE CORRECTS REGISTERS AUTOINCREMENTED OR AUTODECREMENTED
;DURING INSTRUCTION THAT CAUSED ABORT.  THIS CODE IS NOT USED WHEN AN INTERRUPT
;CAUSED THE ABORT.

CORECT: MOV     #177574, R2         ;ADDRESS OF SR1
        MOV     #2, R3

LOOP1:  MOVB    (R2) +, R0          ;PUT CONTENTS OF SR1 IN R0, STEP ADDRESS
        BEQ     DONE                ;NO REGISTER CHANGED
        MOV     R0, R1              ;COPY SR1 INTO R1
        BIC     #177770, R1         ;MASK ALL BUT REGISTER #
        ASL     R1                  ;MULTIPLY REGISTER # TO GET CORRECT INDEX
                                    ;INTO STACK WHERE REGISTER IS TEMPORARILY
                                    ;STORED.
        ASH     -3, R0              ;RIGHT JUSTIFY CORRECTION CONSTANT
        ADD     R6, R1              ;CREATE POINTER TO REG ON STACK
        ADD     R0, (R1)
        SOB     R3, LOOP1           ;BRANCH FOR SECOND CORRECTION

;CODE TO RESTORE PROGRAM COUNTER

DONE:   MOV     @#177576, 16 (R6)
        BR      RESTUR
```

2.4.11 Fatal System Errors

The PDP-11/45 hardware design employs many safeguards that facilitate its use in reliable real-time and time-shared operating systems. These safeguards either prevent careless programmers from causing fatal system errors or allow the operating system to recover from the program fault. However, certain restrictions apply to system error, or "exception" handling. All facilities for handling interrupts with priority greater than processor priority 7 must be resident, meaning that for each of the fault types below the trap vectors, theappropriate stack and the service routine must all be resident. Otherwise, a fatal system error loop may occur that can only be intercepted from the console.

Service Routine	Trap Vector
Old Address	4
Fatal Stack Violation (Red)	4
KT11-C Abort	250
Non-existent Memory Timeout	4
Parity Error	4
Warning Stack Violation (Yellow)	4
Power Fail	24
FP11 Exception Trap	224

This restriction is necessary because: if, during the initial processing of a KT11-C abort any of these interrupts causes a second KT11-C abort, recovery will not be possible because the KT11-C can log status information in SR0, SR1, and SR2 for a single abort only. Until the information in SR0, SR1, and SR2 has been captured to handle the first abort, a second KT11-C abort must not occur.

APPENDIX A
GLOSSARY

Term	Definition
Abort	A processor interrupt that can occur at any memory reference, not just between instructions. The KT11-C will cause an abort if an address attempts to access a non-resident address, to write into a read-only page, or violates the length of a page. When the abort occurs, the KT11-C logs the page number causing the abort, the virtual address of the instruction causing the abort, and any other information necessary to process the abort. A KT11-C abort takes a new PC from Kernel D virtual address 250 and a new PS from 252. The old PS and PC are pushed on the stack specified by the new PS bits ⟨14:15⟩.
Address Space	The set of addresses used during processing in a specific processor mode (Kernel, Supervisor, User). The Virtual Address Space is that set of addresses authorized by the KT11-C for a particular mode. The Physical Address Space for the mode is the virtual address space as mapped into physical addresses. In general, the Kernel, Supervisor, and User modes will operate out of the same virtual address space but out of different physical address spaces.
BEND	Acronym for Bus END – meaning that the bus cycle (read "memory" cycle) in process is terminated. A BEND condition occurs when an anticipatory fetch turns out to be wrong or when some error condition is detected on the reference. When the BEND condition is detected in the KT11-C logic, no abort or memory management trap can occur on the reference.
BUST	Acronym for BUs STart (read "memory cycle" start). The name of the processor state that constitutes the first half of a memory cycle. The second half of the memory cycle is designated PAUSE.
Internal Register	All Page Address Registers, Page Descriptor Registers, Status Registers 0, 1, 2, and 3 are considered to be "internal registers".
Mode	The PDP-11/45 has three modes of operation: Kernel, Supervisor, and User. The KT11-C provides a separate set of PAR/PDR registers for use in each of these modes. The mode is specified by bits ⟨15:14⟩ in the processor status (PS) word: 00 = Kernel, 01 = Supervisor, 11 = User.

Term	Definition
Operating System	A collection of programs that provides facilities and service to a user by allocating resources (CPU, memory, I/O) among several users. An operating system is identical to a "control" program, a "monitor" program, or "supervisor" program. The PDP-11/45 is designed to run the operating system in the Kernel and Supervisor modes.
PA	An abbreviation for "Physical Address".
Page	A collection of continuous memory addresses. The KT11-C divides the 32K word processor address space into eight 4K sections called pages. The lowest address in each page is a whole multiple of 4096. The length of the page is some whole multiple of 32 through 128. Thus, a page may vary in size from 32 to 4096 words, in 32 word increments, and begins on a 4096 word boundary.
PAR – Page Address Register	A register containing the "base address" or "relocation constant" associated with a page. The KT11-C has 48 PARs, 16 associated with each of the three processor modes (User, Supervisor, and Kernel).
Pause	The term used to describe the second half of a memory cycle. During the Pause cycle, KT11-C status registers are updated and the processor, if necessary, pauses to wait for completion of the current memory cycle.
PDR – Page Descriptor Register	A register containing information associated with a page. This includes the length of the page, the expansion direction, and the access key. The KT11-C has 48 PDRs, 16 associated with each of the three processor modes (User, Supervisor, and Kernel).
Physical Address	The address generated by the KT11-C that is used to select a specific memory location. When the KT11-C is operating, the "physical address" is identical to the "relocated address".
ROM – Read Only Memory	A hardware device that is able to store bit patterns that can be read by providing a specific address. In the KT11-C, four ROMs are used to control modifications to status registers, detect exceptions, and separate I space references from D space references.
VA	An abbreviation for "Virtual Address".
Virtual Address	The address generated by the PDP-11/45 processor. The term "virtual" is applied because the address will be relocated by the contents of the PAR. The memory location reached will be the relocated address. Hence, the processor "thinks" it is addressing one location but actually gets another. The first address is then really a dummy or "virtual" address.

APPENDIX B
REFERENCE LITERATURE

The following list of references covers some of the more general aspects of memory management tasks of interest to systems programmers. It is part of the recommended bibliography of the National Academy of Engineering for its course outline on "Operating System Principles".

The following abbreviations are used in the bibliography.

ACM	Association for Computing Machinery
IEEE	Institute for Electrical and Electronics Engineers
IEEETC	IEEE Transactions on Computers
CACM	Communications of the ACM
JACM	Journal of the ACM
CS	Computing Surveys (ACM)
FJCC	Fall Joint Computer Conference
SJCC	Spring Joint Computer Conference
2SOSP	Second Symposium on Operating Systems Principles (proceedings available from ACM, 1133 Avenue of Americas, New York, N.Y. 10036)

Abate, J., and Dubner, H. Optimizing the Performance of a Drum-Like Storage. IEEE Trans. C-18, 11 (Nov. 1969), 992–997).

Belady, L.A. A Study of Replacement Algorithms for Virtual Storage Computers. IBM Sys. J. 5, 2 (1966), 78–101.

Bensoussan, A., Clingen, C.T., and Daley, R.C. The Multics Virtual Memory. Proc. 2SOSP (Oct. 1969).

Denning, P.J. The Working Set Model for Program Behavior. Comm. ACM 11, 5 (May 1968), 323–333.

Denning, P.J. Thrashing: Its Causes and Prevention. AFIPS Conf. Proc. 33 (1968 FJCC), 915–922.

Denning, P.J. Virtual Memory. CS 2, 3 (Sept. 1970), 153–189.

Dennis, J.B. Segmentation and the Design of Multiprogrammed Computer Systems. JACM 12, 4 (Oct. 1965), 589–602.

Kilburn, T. et al. One-Level Storage System. IRE Trans. EC-11, 2 (Apr. 1962), 223–235.

Knuth, D.E. The Art of Computer Programming (Vol. 1). Addison-Wesley (1968), Ch. 2.

Mattson, R.L., Gecsei, J., Slutz, D.R., and Traiger, I.L. Evaluation Techniques for Storage Hierarchies. IBM Sys. J. 9, 2 (1970), 78–117.

Randell, B., and Keuhner, C.J. Dynamic Storage Allocation Systems. Comm. ACM 11, 5 (May 1968), 197–305.

Sayre, D. Is Automatic Folding of Programs Efficient Enough to Displace Manual? CACM 12, 12 (Dec. 1969), 656–660.

Wilkes, M.V. Slave Memories and Dynamic Storage Allocation. IEEE Trans. EC-14 (Apr. 1965), 270–271.

Wilkes, M.V. Time-Sharing Computer Systems. Am. Elsevier (1968).

INDEX

NOTE

In this index, "F" indicates a figure reference, "T" indicates a table. For information on subjects indexed to paragraphs, figures, and tables in "Chapter 3", refer to Chapter 3 of the KT11-C Memory Management Unit Maintenance Manual, DEC-11-HKTB-D. The hardware logic descriptions appearing in that manual are not reprinted in this reference manual.

NOTE: T = Tables and F = Figures.

READER'S COMMENTS

PDP-11/45 MEMORY MANAGEMENT REFERENCE MANUAL
DEC-11-HGKTC-B-D

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 do you find with the manual? _____

_____

_____

_____

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

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

_____

_____

_____

Would you please indicate any factual errors you have found. _____

_____

_____

_____

Please describe your position. _____

Name _____ Organization _____

Street _____ Department _____

City _____ State _____ Zip or Country _____

— — — — — — — — — — — — Fold Here — — — — — — — — — — —

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

FIRST CLASS
PERMIT NO. 33
MAYNARD, MASS.

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

Postage will be paid by:

Digital Equipment Corporation
Technical Documentation Department
146 Main Street
Maynard, Massachusetts 01754

READER'S COMMENTS

PDP-11/45 MEMORY MANAGEMENT REFERENCE MANUAL
DEC-11-HGKTC-B-D

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 do you find with the manual? _____

_____

_____

_____

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

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

_____

_____

_____

Would you please indicate any factual errors you have found. _____

_____

_____

_____

Please describe your position. _____

Name _____ Organization _____

Street _____ Department _____

City _____ State _____ Zip or Country _____

— — — — — — — — — — — — Fold Here — — — — — — — — — — —

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

FIRST CLASS
PERMIT NO. 33
MAYNARD, MASS.

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

Postage will be paid by:

Digital Equipment Corporation
Technical Documentation Department
146 Main Street
Maynard, Massachusetts 01754

DIGITAL EQUIPMENT CORPORATION
MAYNARD, MASSACHUSETTS 01754