NAVAL AIR DEVELOPMENT CENTER
WARMINSTER, PENNSYLVANIA 18974

2071
7 Mar 1974

TECHNICAL MEMORANDUM

Subj: Addressing Structure of the All Applications Digital Computer Data Processing Element

Ref: (a) NAVAIRSYSCOM ltr 360F:FRR of 15 Feb 1974
(b) Naval Research Laboratory Memorandum 5403:159:HC:1a of 16 Oct 1973

Introduction

Reference (a) requested NAVAIRDEVCEN to review and comment on reference (b). Accordingly, this technical memorandum has been prepared in accordance with the provisions of reference (a).

Machine Organization

The DPE of the AADC (All Applications Digital Computer) contains a PMU (Program Management Unit), an AP (Arithmetic Processor), a Channel and a TM (Task Memory). The TM has 4K 36-bit words divided into sixteen 256 word pages. Program pages are stored in TM during execution. In addition, TM can be used to store data temporarily. The PMU performs the control functions of the DPE, and the AP performs the arithmetic and logical computations. The channel provides the interface with the main data buses.

The PMU is separated from the AP by a queue (APQ) to provide for parallel operation. The APQ holds up to sixteen 56 bit items. The PMU fetches an instruction from TM and an operand from one of the memories described below, and places the operation code, operand, and necessary control and sequencing information into the APQ. The AP removes and executes these instructions.

The BORAM (Block Organized Random Access Memory) is read-only and stores procedures and constants for programs in blocks of 256 words each. Program pages are transferred to TM on demand via the main data bus at a rate of 150 nanoseconds per word.

The RAMM (Random Access Main Memory), which is divided into 256-word pages, stores data for the system. The data in a RAMM page is organized by software into one or more data segments. A data segment is a block of contiguous data words which does not cross a page boundary. It is defined by a starting address and there is no record of its length. For example, a data segment may consist of a single precision word, five contiguous ten-word arrays, or an entire page of data. Data in RAMM can either be directly accessed by the DPE or transferred in pages into TM at a 150 nanosecond per word rate.

A block diagram of an AADC SP (Simplex Processor) configuration appears in figure 1.

Modes of Operation

The DPE can operate in either executive or problem mode. Any PMU in a particular AADC configuration, including the DPE PMU, can be preset to act as the executive. A Transfer to Executive command issued by a user program or an I/O interrupt will cause the executive program in the designated PMU to become active. Normal (problem) mode will be reentered upon completion of the executive task. The identity of the executive PMU can be changed operationally only by an instruction issued by the present executive, informing another PMU that it (the second PMU) is now the executive. Instructions dealing with data flow and control between different AADC subsystems within a particular configuration are illegal if fetched from local memory in problem mode. They will, however, be executed in executive mode.

The AADC supports both "virtual" and absolute addressing. The desired mode of addressing is specified by the Set Task Parameters instruction sent to the DPE by the executive. Absolute addressing is, in general, no faster than virtual addressing, and should not be used under normal conditions.

In both virtual and absolute addressing modes, the program counter contains a virtual address. In virtual mode, data is either fetched directly from an external RAMM or transferred in pages to TM. When a data page is overwritten in TM, the contents of the overwritten page are not written back into RAMM automatically. Any altered information not programmatically restored in RAMM will be lost. In absolute addressing mode, data addresses are interpreted as physical addresses in TM.

Page Replacement Policy

Pages being transferred from RAMM or BORAM replace pages in TM according to one of seven page replacement algorithms specified by the Set Task Parameters instruction. Six of these algorithms are automatic (Most Recently Used, Random, Random by Pairs, Reverse, First In/First Out, First In/First Out by Pairs). The other algorithm is "programmer specifiable." The user indicates where in TM he wants to store pages by placing the TM page number in the TM field of the kernel word associated with the particular RAMM or BORAM page (see sections 6-8). The "programmer specifiable" algorithm provides the programmer with the capability to generate his own procedure overlay, which may be more efficient than any of the standard algorithms for some types of structured programs.

A pair of registers bound the TM area which is subject to automatic page replacement. The executive loads the LBR (Lower Bounds Register) and UBR (Upper Bounds Register) with TM page numbers specified by the Set Task Parameters instruction. Up to fifteen of the sixteen TM pages may be reserved in this manner. The number of pages reserved is application dependent, but up to four kernel pages can be resident in TM at one time, and up to three pages are required for array data, and additional pages may be needed for trap routines and local memory stacks.

Instruction Word Format

The DPE instruction word consists of 36 bits. In all instructions, bits 0-7 and bits 32-35 have the same meaning. Bits 8-31 have several formats. The "standard" instruction format is shown in figure 2. The operation code also determines whether the operation is to be performed on a 16 bit half word operand or a 32 bit full word operand.

The DPE instruction set is divided into AP and non-AP instructions. Bits 0-7 designate the operation code of either class of instructions.

In AP instructions, bit 8 is the precision bit (PR), and bits 9-11 specify the PC (parenthetical code). The precision bit controls the precision of the result of each AP operation. Zero indicates single precision, (32 bits) one indicates double precision (64 bits). The parenthetical code controls the order of instruction execution.

In instructions which involve the PMU only, bits 8-11 and the operation code specify one of 16 scratchpad registers (16 bits each). The operation code of the instruction whether the low scratchpad register set (0-15), the high scratchpad register set (16-31), or a 32 bit register pair (0 and 16, 1 and 17, etc) should be addressed.

Bit 12 is the indirect addressing bit (I). When the bit is set, additional operand fetches are performed. Bits 13-15 specify the indexing field. If no indexing is to be performed, a value of zero should appear in the field. Otherwise, one of the scratchpad registers 1-7 is used to index the instruction. Indexing is performed before indirect addressing. The indirect addressing and indexing sequence is more fully described in section 11.

Bits 16-31 are the operand address field which may address a word of procedure or a data operand. A DPE data operand may be single precision, double precision, a complex quantity, or an array preceded by a dimension word. The 3 bit tag identifies the type. Operands to be processed by the PMU (exclusive of the AP) do not use the data tag. These operands may be either half word or full word.

Under normal operating conditions, the DPE is in "virtual" addressing mode and the operand address field specifies a VA ("virtual" address). When the DPE is in absolute addressing mode and the operand is a data element, bits 16-31 contain an AA (absolute address) of the data operand in TM. The 16-bit virtual address is used as two 8 bit bytes. The first byte addresses a KW (kernel word) which contains addressing, trap, residency, and protection information on the procedure page or data segment containing the operand. The kernel, which plays a major role in the addressing scheme, is discussed further in section 6-8. The second byte specifies a displacement (D) from the beginning of the procedure page or data segment.

Bit 32 is the addressable/literal bit (A). This bit is examined when bit 12 is a ONE as an indicator of register indirection (see figure 6A). The bit is examined in the addressable/literal context when bit 12 is 0. When bit 12 is 0 and bit 32 is ONE, bits 16-31 of the instruction are the address of the 32 bit operand to be fetched. When bit 32 is zero, bits 16-31 are used as a data operand. In full word instructions a 32 bit positive integer operand is formed by catenating 16 zero's with bits 16-31 of the instruction word. Bits 16-31 are the least significant bits of the operand so formed. In half-word instructions, bits 16-31 are used as a signed integer operand. Certain instructions are designated as non-immediate. For these instructions, bit 32 has no effect on the operand fetch cycle. Bits 16-31 are always treated as an address. The Non Immediate instructions are:

1. All Transfer Instructions (Excluding Skip Instructions)
2. All Store Instructions
3. Load Multiple
4. Execute
5. Return Stack to P
6. Return Stack to P and Proceed
7. Single Word I/O
8. Double Word I/O
9. Set Bit N
10. Reset Bit N

Single Word I/O
Double Word I/O

Bit 33 is the AP/Not AP bit (U). When the bit is zero, the instruction is not for the AP. When the bit is one, the instruction is for the AP. In the latter case, the operation code, the operand specified by the address information, and other control and sequencing information is placed on the APQ.

Bit 34 is the instruction trace bit (T). When the bit is one, the PMU will trap to a procedure at a fixed location reserved for instruction trace after the instruction is executed.

Bit 35 is the odd parity check bit (P).

Kernel

Associated with each task is a procedure kernel page and a data kernel page. They are usually the same page. However, when total storage requirements for a single task exceed 65K words, up to four pages can be reserved as a kernel area.

The kernel contains the addressing and control information necessary to execute the task and remains resident in TM throughout the execution of the task. The kernel may only be written into by the executive. NO

There is a word in the kernel associated with each procedure page or data segment. Since the maximum kernel size is four 256 word pages, a task may have a maximum of 1024 procedure pages or data segments. A procedure page can be stored anywhere in either BORAM, RAMM, or TM. A data segment, whose size ranges from one to 256 words, may be stored anywhere in RAMM as long as it is contained in a single page. Tables may extend across page boundaries; however, the first word of the table should be in the first word of a page.

Kernel Word Format

7. The 36 bit kernel word has the format illustrated in figure 3.

Bits 0-7 are used when bit 10 = 1. The field specifies the TM page containing the procedure or data page associated with that kernel word. This field is 8 bits to allow for future expansion of TM.

Bit 8 is the kernel load bit (KL). This bit, when set, allows the load of a data kernel page through this kernel word.

Bit 9 is the kernel trace bit (T). When the bit is one, the PMU will trap to a procedure at a fixed location reserved for kernel trace after the current instruction has completed execution.

Bit 10 is the residency bit (R) which is used when bit 11 = 1. When the bit is zero, the page is not resident in TM. When the bit is one, the page is the TM page identified by bits 0-7 of the kernel word.

Bit 11 is the page/word bit (PW). When the bit is zero, the data is "word oriented" and the data operand is accessed directly from RAMM. When the bit is one, the instruction or data desired is "page bound." The entire page must be transferred by BORAM or RAMM to TM before operations utilizing the data operand within the page are performed.

Bits 12-31 specify the RAMM or BORAM address of the desired data segment or procedure page. Bits 12-19 indicate the system resource number of the BORAM memory unit, and bits 20-31 contain the physical page number of the desired page within the identified BORAM. Up to

4096 pages per BORAM may be addressed. When the memory unit is a RAMM, bits 12-31 contain the address of the first word of the desired data segment within the RAMM. Up to 4096 words or 16 pages may be addressed per RAMM system resource. A physically contiguous RAMM may contain up to 65K words. In this case it would respond to 16 system resource names, or alternatively, bits 12-15 would be the RAMM system resource name and bits 16-31 would specify the memory location in the RAMM. A 32K word RAMM would correspondingly respond to 8 system resource names and bits 17-31 would specify the internal address.

Bit 32 is the read protect bit (RP). When this bit is set, the user program is not permitted to read information in the indicated procedure page or data segment. If this bit is set in the kernel word while a read is attempted, a trap occurs.

Bit 33 is the write protect bit (WP). When this bit is set, the user program is not permitted to write into the indicated procedure page or data segment. If this bit is set in the kernel word while a write or store is attempted, a trap occurs.

Bit 34 is the command protect bit (CP). In order for a page of instructions to be executed, the command protect bit of the kernel word must be zero. Otherwise, a trap occurs and the execution is aborted.

Bit 35 is the odd parity check bit (P).

The example illustrated in figure 4 demonstrates kernel, RAMM and BORAM organization for a user program. The program has 1050 instruction words, comprising five pages, and 542 data words. The data consists of five arrays, four single precision numbers and two double precision numbers. One particular grouping of the data into segments is displayed in the example. Numerous other data segment configurations are possible. The restrictions in grouping data into segments are that a data segment may not exceed 256 words and that no data operand (e.g., array) may be divided among two or more segments. The restriction in loading data segments into RAMM is that a data segment must be contained within a single RAMM page.

Addressing Registers

The registers described below are used in the translation of a "virtual" address to a physical address.

The PC (Program Counter) is a 16 bit register which contains the "virtual" address of the current instruction. It is divided into two 8 bit bytes.

Bits 0-7 address a KW (Kernel Word) for the procedure page containing the instruction. Bits 8-15 specify a displacement (D) from the beginning of the procedure page to obtain the instruction word.

The PPR (Present Page Register) is an 8 bit register which addresses the TM page containing the current instruction.

The KPR (Kernel Page Register) is a 2 bit register which addresses the TM page containing the current kernel page. There are two KPR's; one for procedure addressing and one for data addressing. Their contents may be identical.

Instruction Fetch

The DPE is assigned a new task by means of the "Initiate New Task" instruction sent to it by the executive. The instruction sets up the KPR's and provides the BORAM address of the first kernel page. If the page is not resident, the BORAM address is used to obtain it. The instruction also indicates if the kernel page is already resident. The first word of the kernel page is associated with the first procedure page of the task. This procedure page is transferred from BORAM to TM if the kernel word associated with the procedure page is marked as non-resident. The first word of this page is the first instruction to be executed. Prior to execution, the PC addresses the first kernel word (word 0) with a zero displacement and the PPR addresses the TM page containing the first procedure page. The loading of subsequent data kernel pages into TM may be performed any time during program execution.

or an "internal" [illegible]

The PC contains the "virtual" address of the instruction to be executed. To obtain the TM absolute address, the kernel word pointed to by bits 0-7 of the PC is examined. If the procedure page containing the instruction is not resident, the page is transferred from the BORAM to TM, the residency bit of the corresponding kernel word is set, and bits 4-11 of the kernel word, which contains the TM page number of the procedure page, are moved into the PPR. The 16 bit TM address is the concatenation of the 8 bit PPR and the 8 bit displacement field of the PC. Formation of the TM absolute address is displayed in figure 5.

The PC is incremented by one to anticipate the next instruction. The addressing cycle in figure 5 is not performed when the kernel word field (bits 0-7) of the PC is unchanged, because the next instruction is in the same procedure page as the previous instruction. When bits 0-7 of the PC change (page fault) a new page will be brought into TM automatically via the full virtual addressing cycle. All procedure pages to be executed are referenced in the kernel page. If the same page is executed by two different tasks, it will be referenced in both kernel pages. If task storage requirements, or the possibility of an interrupt requiring immediate response require the residence of more than one kernel page in TM, the other kernel pages can be reached by means of the "Transfer and Stack Kernel N" instruction (N = 0, 1, 2, 3). The KPR's are updated to the new kernel page number while the subroutine is being executed, and the old contents of the KPR's will be restored when the subroutine is completed.

Operand Fetch (See figure 6, 6A)

Bits 12-15 of the instruction word are used to modify the operand address. If the index field (bits 13-15) is zero, no indexing is performed. An index field of one through seven determines the desired register in the low scratchpad register set. The 16 bit contents of the specified index register is added to bits 16-31 of the instruction word, losing a carry to bit 15. Indirect addressing is performed after indexing. The process of indirect addressing entails replacing bits 12-32 of the original instruction word with a new set of bits. Bit 12 of the original instruction, when a one, specifies that indirect addressing is to be performed. When bit 12 is a 0, no indirect addressing is performed.

When bit 12 is a one, bit 32 specifies one of two indirect modes: Memory or Register Indirect Addressing. In memory indirect addressing (bit 32 is a ONE), Bits 16-31 of the instruction are used to address local TM (normally a virtual address). Bits 12-32 of the referenced word replace bits 12-32 of the original instruction and the fetch cycle is reentered. In register indirect addressing (bit 32 set to 0) the contents of the instruction word are rearranged as follows:

Bits 28-31 replace bits 12-15. (New address modification field).
Bit 17 replaces bit 32. (New indirect tag)
In addition, bit 16, if a one, indicates a replacement operation.
Bits 23-27 are saved for use with indexing which follows.

After this rearrangement, the contents of the scratchpad register (SP) specified by bits 23-27 replace bits 16-31 of the original instruction. The process of indexing is again performed and indirect addressing may be repeated. If the replacement operation was specified by the previous indirection (bit 16 a ONE), bits 16-31 of the modified instruction word, (after indexing) are written back into the register specified by bits 23-27 of the preceeding register indirect cycle, which were saved as explained above. The indexing and indirect addressing cycles are continued until bit 12 is ZERO. At this point the operand fetch cycle is entered. After all indexing and indirect addressing is performed, the addressable/literal bit (bit 32) of the modified instruction word is examined to determine if the operand field (bits 16-31) should be treated as a 16 bit literal operand or as the address of an operand.

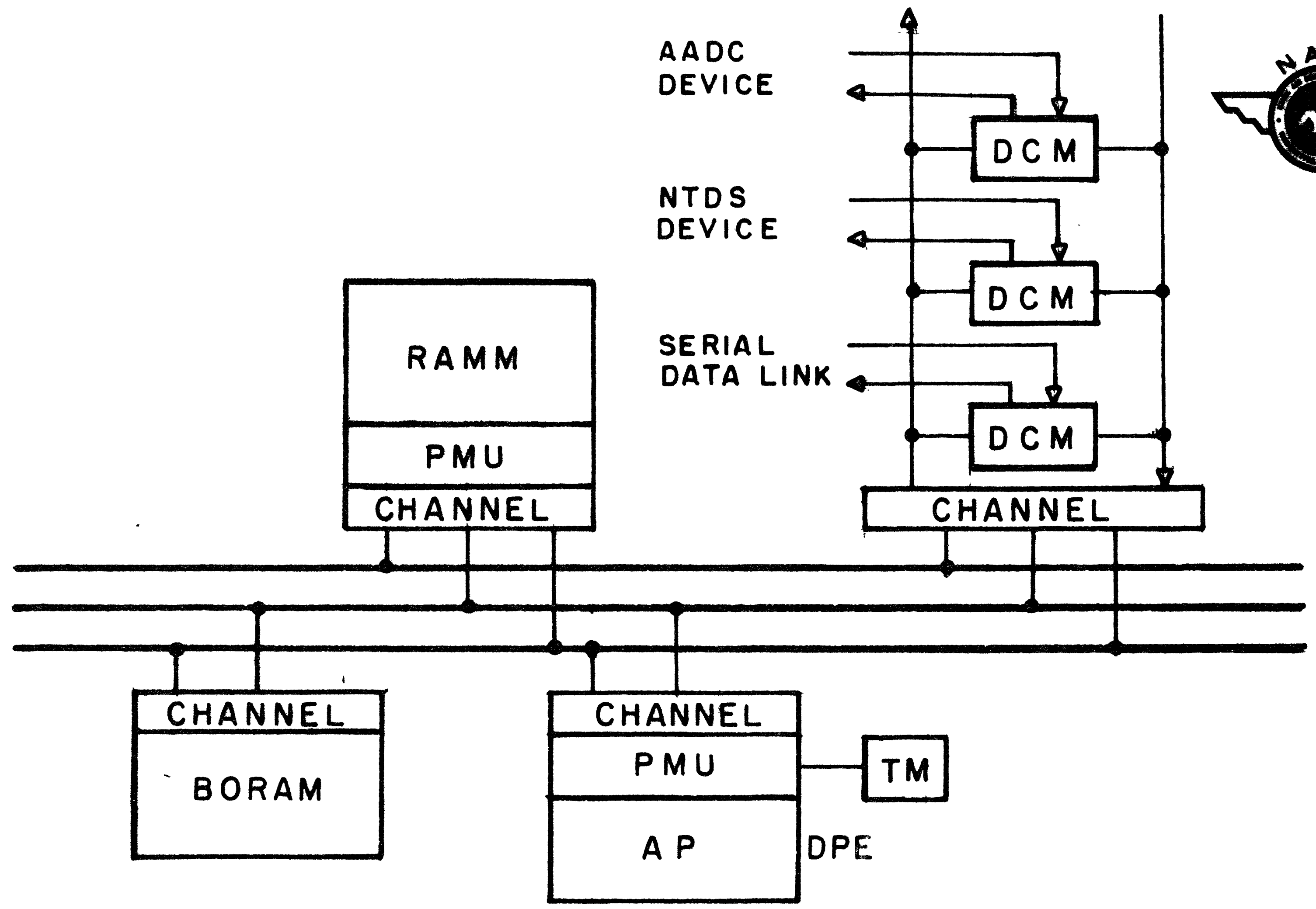
When the addressable bit is set or indirect addressing is specified, an operand must be fetched from TM or RAMM. When the operand field is interpreted as a physical address (bits 16-31), the kernel is not used.

When the operand field contains a "virtual" address (bits 16-31), it must be translated to a physical address. Bits 16-23 of the instruction word address a word in that kernel whose location in TM is pointed to by the KPR.

When the kernel word is marked as "page oriented" (bit 11 = 1), it may be associated with a BORAM procedure page, RAMM procedure page, or a RAMM data segment (see figure 7). When the memory unit is a BORAM, bits 20-31 of the kernel word reference a page in BORAM. If the kernel word associated with the page is non-resident, the page is transferred to TM. A TM address of an operand is obtained in the same way as the TM address of an instruction is obtained. When the system resource field identifies a RAMM, bits 20-31 of the kernel word reference a word in RAMM. If the page containing the referenced word is not in TM, the entire physical page is read from RAMM beginning with the addressed word and continuing consecutively until 256 words have been accessed. If the address does not point to the first word of the RAMM page, a wrapping around process occurs whereby the first physical word of the RAMM page is stored in TM after the last physical word of the same page. No page boundaries are crossed. Bits 0-7 of the kernel word are set to the TM page number. These 8 bits are concatenated to bits 24-31 of the instruction word to form the 16 bit TM address.

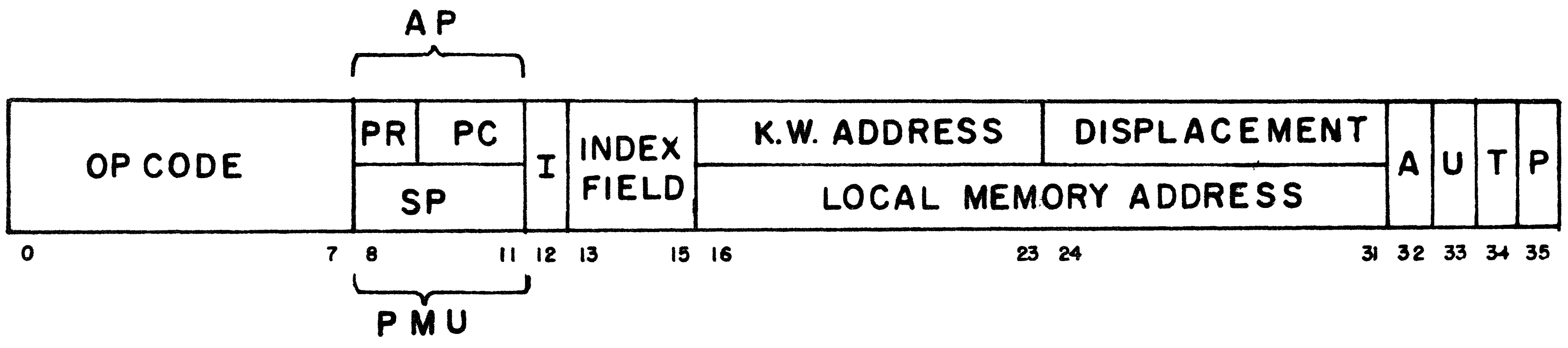
When the kernel word is marked as "word oriented" (bit 11 = 0), it must be associated with a RAMM data segment (see figure 8). Bits 24-31 of the instruction word are added to bits 24-31 of the kernel word, with no carries beyond bit 24, to obtain a new displacement. The carry is inhibited to keep the address in the same RAMM page. Bits 12-23 of the kernel word are concatenated to the new 8 bit displacement to locate the operand in the specified RAMM.

C. JOECKEL



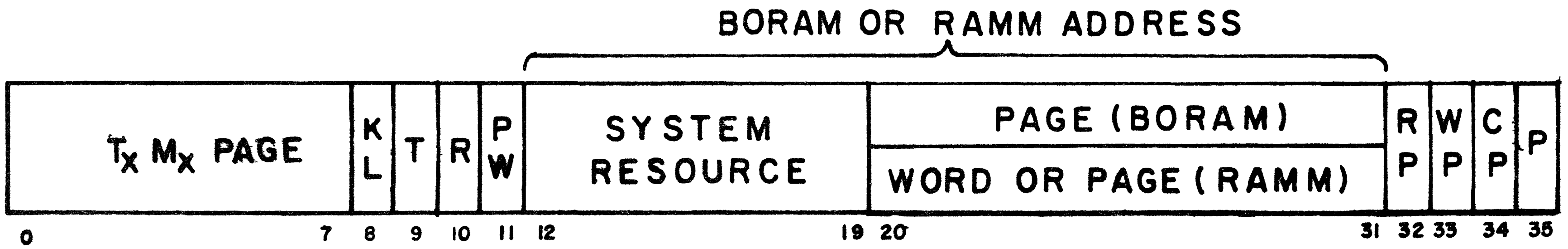
SIMPLEX PROCESSOR.

FIG. I



INSTRUCTION WORD FORMAT.

FIG. 2



KERNEL WORD FORMAT

FIG. 3

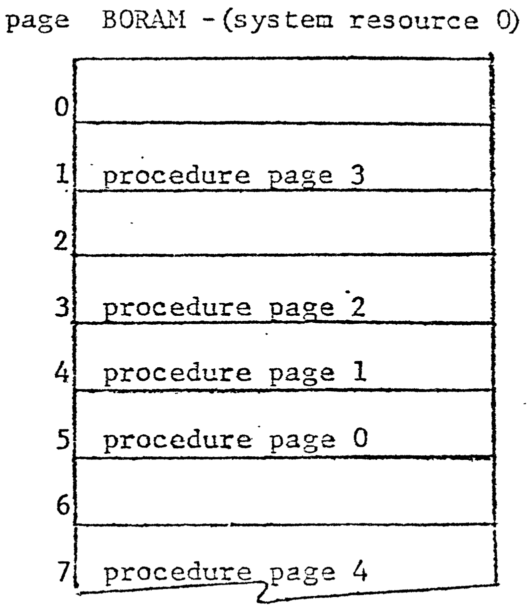
Fig. 4 User Program Example

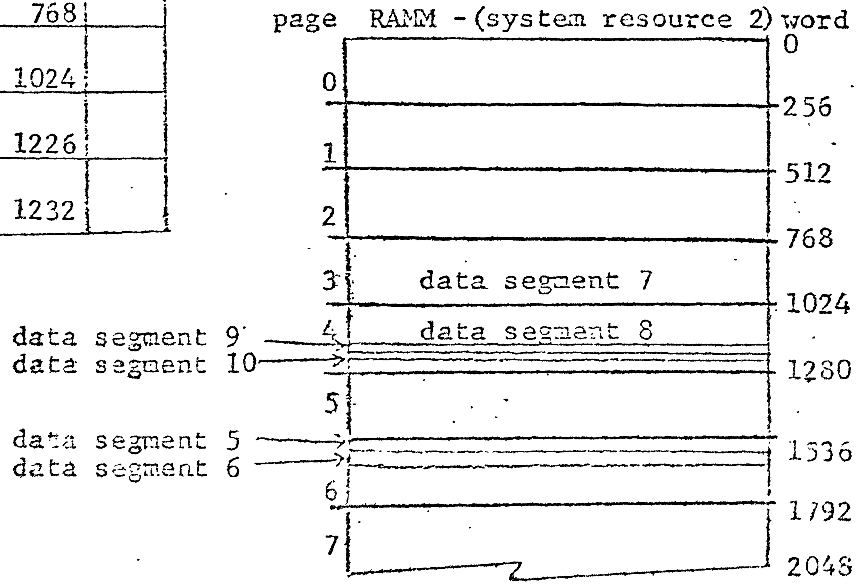
Procedure Page Number*	Number of Instruction Words
0	256
1	256
2	256
3	256
4	26

Data Segment Number*	Data Items	Number of Data Words
5	array A(10)	11**
6	array B(64)	65**
7	array C(255)	256**
8	array X(100), Y(100)	202**
9	single prec. I,J,K,L double prec. M	6
10	double prec. N	2

Kernel Page

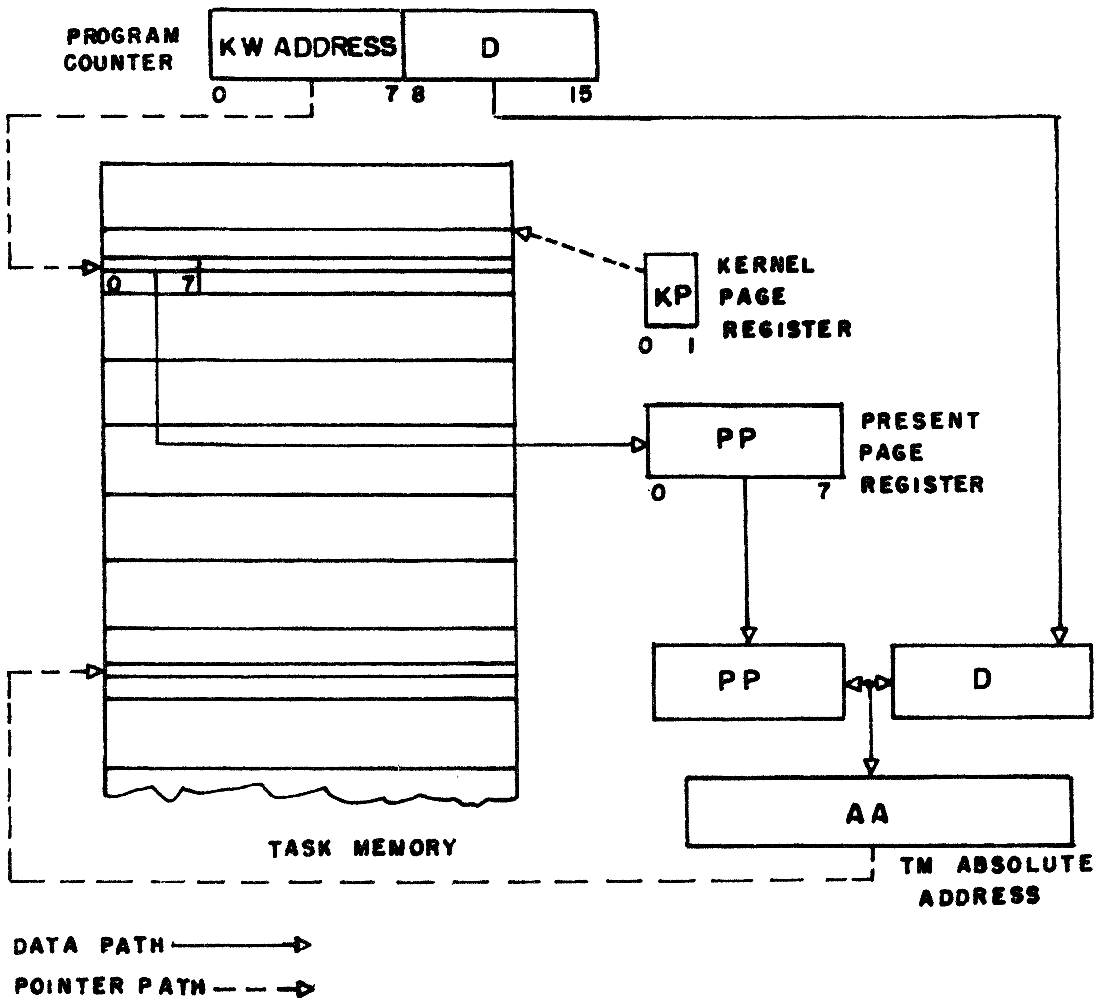
Word	0 – 12	12 – 19 system resource	20 – 31 page or word	31 – 35
0		0	5	
1		0	4	
2		0	3	
3		0	1	
4		0	7	
5		2	1536	
6		2	1547	
7		2	768	
8		2	1024	
9		2	1226	
10		2	1232	





* The procedure page and data segment numbers are the addresses of the kernel words associated with the page or segment.

** The additional words are dimension words.



ADDRESSING CYCLE TO OBTAIN TM ABSOLUTE ADDRESS.

FIG. 5

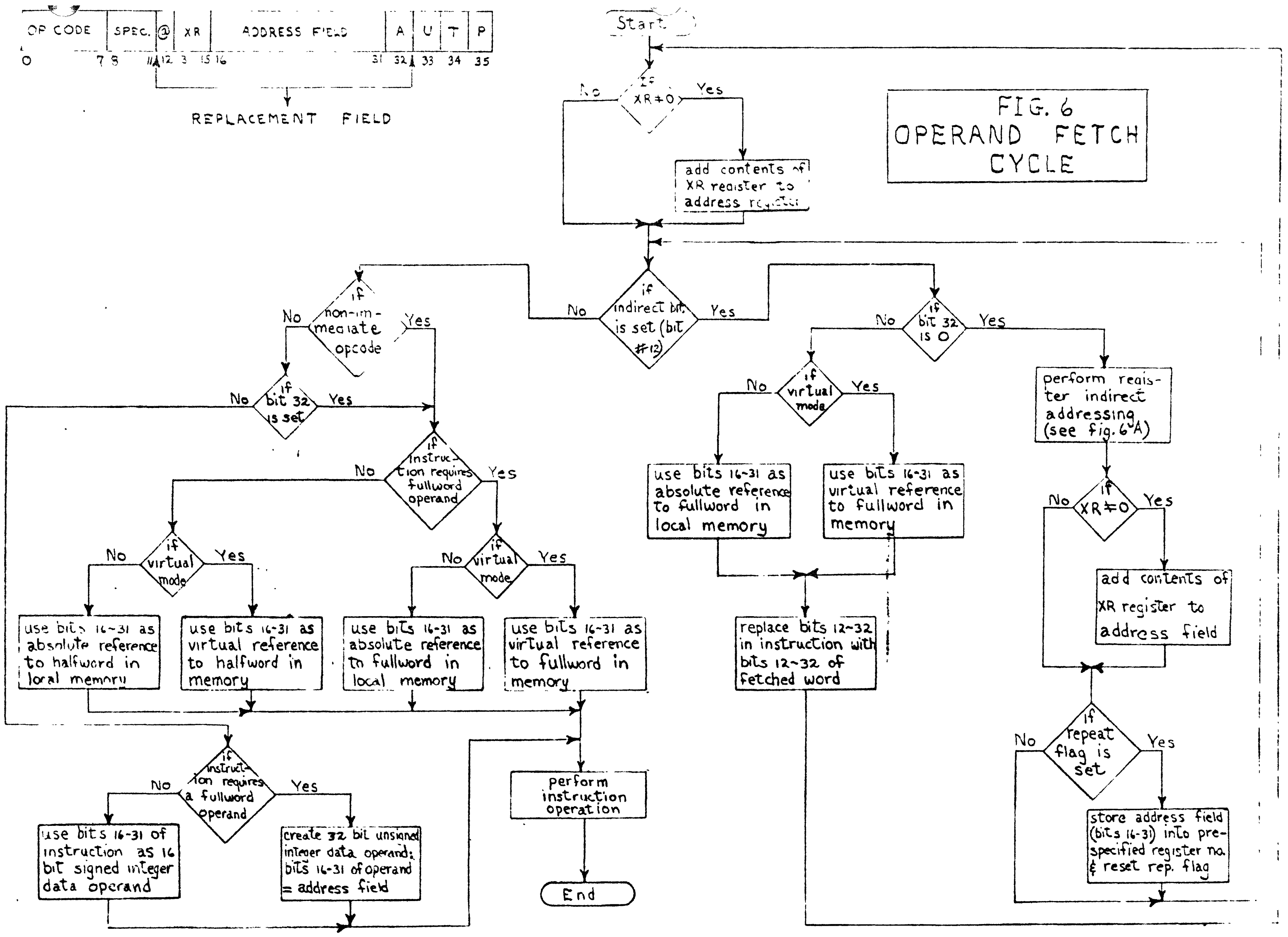
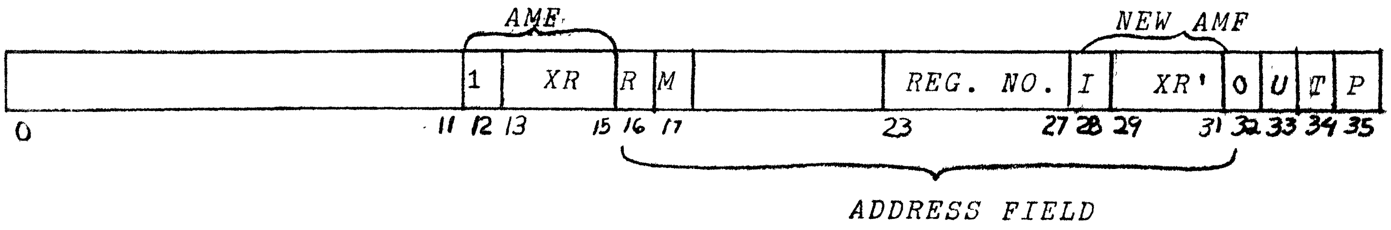
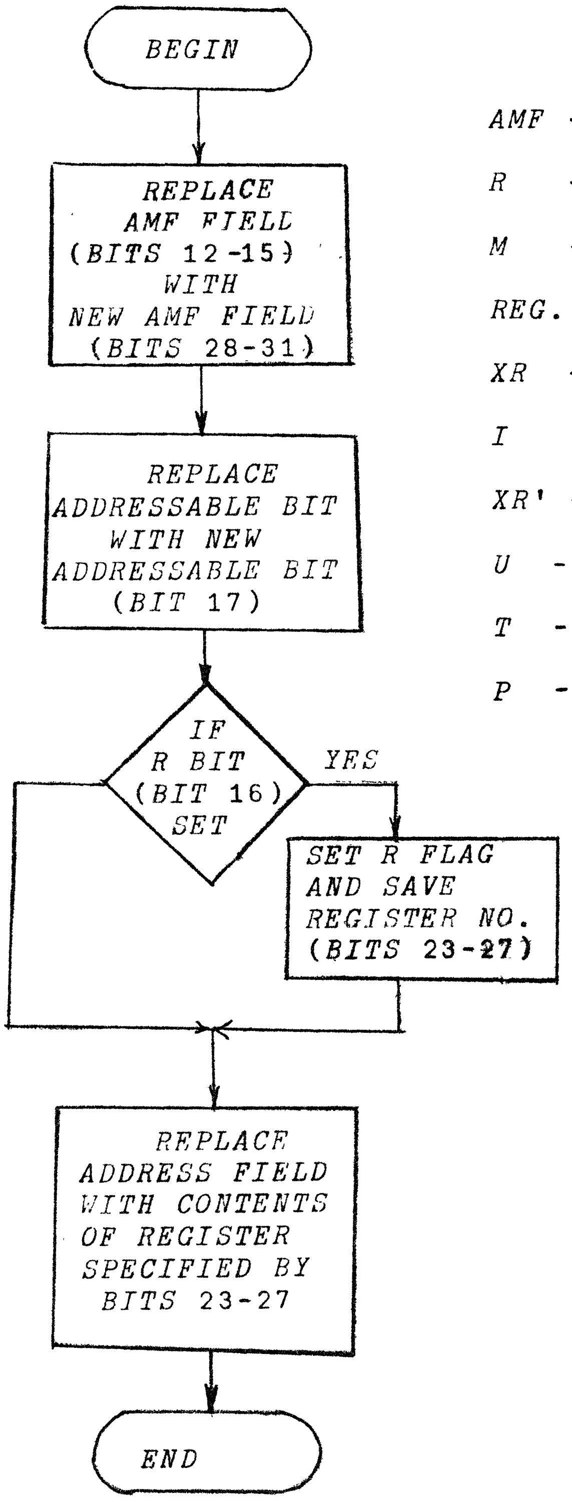

OP CODE
SPEC.
@
XR
ADDRESS FIELD
A
U
T
P
0
7 8
11 12 3 15 16
31 32 33 34 35
REPLACEMENT FIELD
Start
if XR≠0
No
Yes
add contents of XR register to address register
if indirect bit is set (bit #12)
No
Yes
if non-immediate opcode
No
Yes
if bit 32 is set
No
Yes
if instruction requires fullword operand
No
Yes
if virtual mode
No
Yes
use bits 16-31 as absolute reference to halfword in local memory
use bits 16-31 as virtual reference to halfword in memory
if virtual mode
No
Yes
use bits 16-31 as absolute reference to fullword in local memory
use bits 16-31 as virtual reference to fullword in memory
if instruction requires a fullword operand
No
Yes
use bits 16-31 of instruction as 16 bit signed integer data operand
create 32 bit unsigned integer data operand; bits 16-31 of operand = address field
perform instruction operation
End
if bit 32 is 0
No
Yes
if virtual mode
No
Yes
use bits 16-31 as absolute reference to fullword in local memory
use bits 16-31 as virtual reference to fullword in memory
replace bits 12-32 in instruction with bits 12-32 of fetched word
perform register indirect addressing (see fig. 6A)
if XR≠0
No
Yes
add contents of XR register to address field
if repeat flag is set
No
Yes
store address field (bits 16-31) into pre-specified register no. & reset rep. flag

FIG. 6
OPERAND FETCH CYCLE

FIG 6 A

REGISTER INDIRECT ADDRESSING

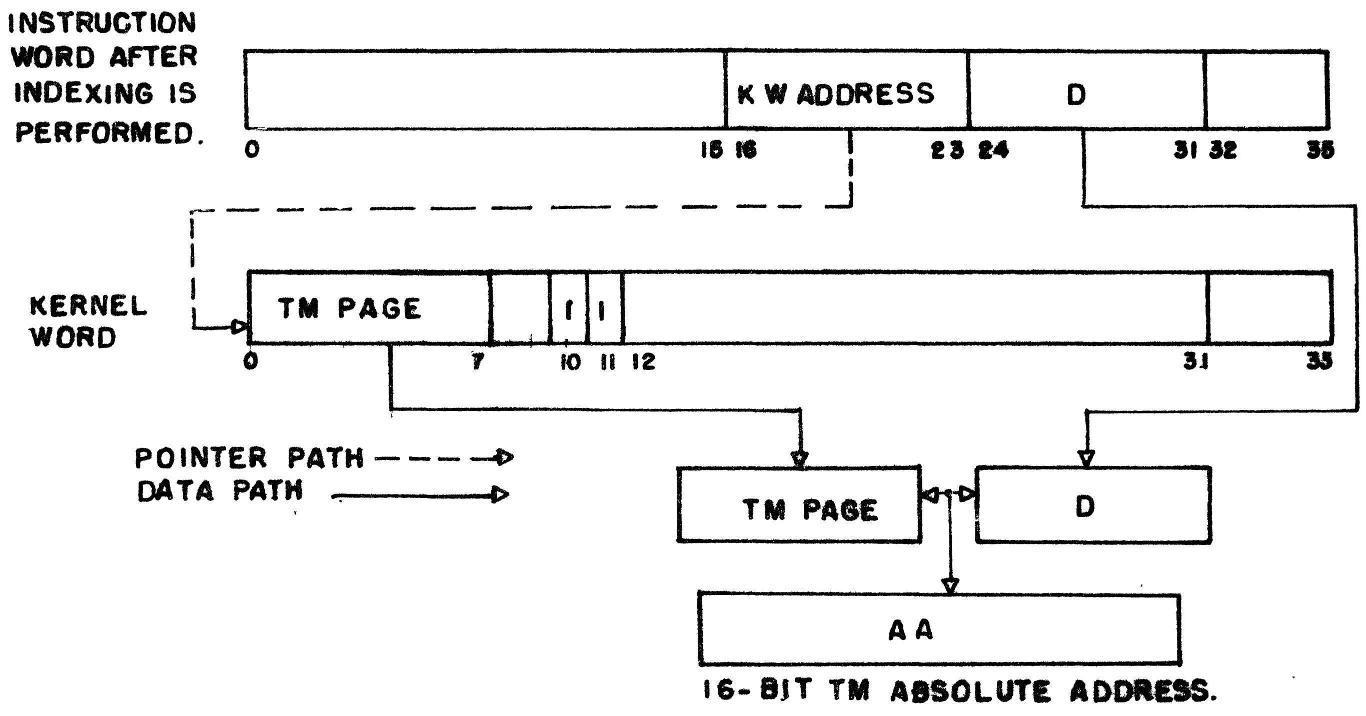




BEGIN
REPLACE AMF FIELD (BITS 12-15) WITH NEW AMF FIELD (BITS 28-31)
REPLACE ADDRESSABLE BIT WITH NEW ADDRESSABLE BIT (BIT 17)
IF R BIT (BIT 16) SET
YES
SET R FLAG AND SAVE REGISTER NO. (BITS 23-27)
REPLACE ADDRESS FIELD WITH CONTENTS OF REGISTER SPECIFIED BY BITS 23-27
END

LEGEND

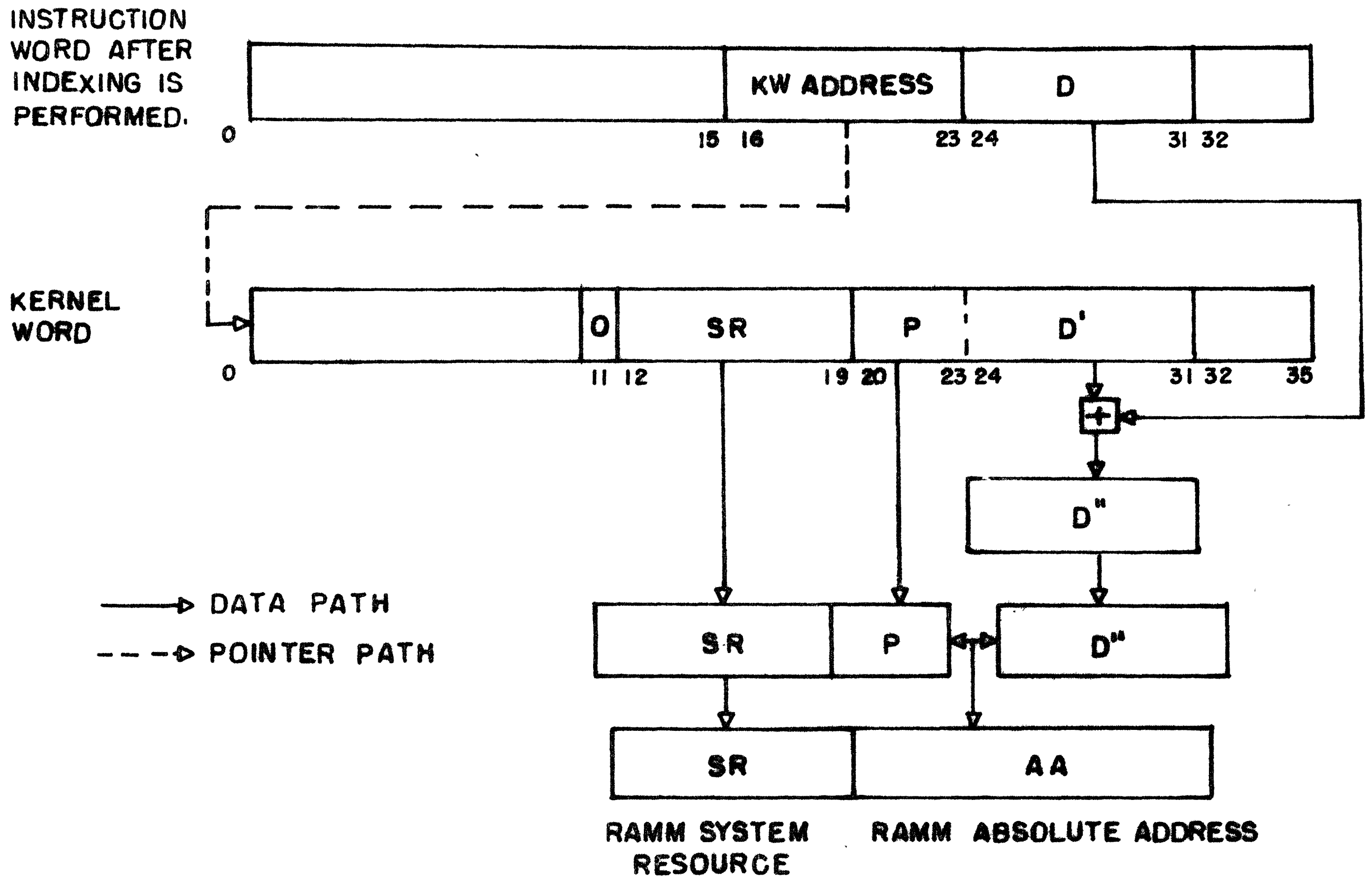
AMF	- ADDRESS MODIFICATION FIELD
R	- REPLACEMENT SPECIFICATION BIT
M	- ADDRESSABLE MODE BIT
REG. NO.	- REGISTER NUMBER (0-31)
XR	- INDEXING REGISTER NUMBER (0-7)
I	- INDIRECTION SPECIFICATION BIT
XR'	- NEW INDEX REGISTER NUMBER
U	- UNIT (AP/NON-AP) BIT
T	- TRACE BIT
P	- PARITY BIT



D= displacement of operand from beginning of TM page. (In the case of RAMM data, it is also the displacement from the beginning of the data segment, since the data segment was transferred from RAMM to the beginning of the TM page).

Figure 7

Virtual addressing of page oriented operands (Bit 11=1) which are resident (Bit 10=1)



D = DISPLACEMENT OF OPERAND FROM BEGINNING OF DATA SEGMENT.

D' = DISPLACEMENT OF OPERAND FROM BEGINNING OF RAMM PAGE.

D" = D + D' WITH NO CARRY.

P = RAMM PAGE.

FIG.8 VIRTUAL ADDRESSING OF OPERANDS, WHEN THE KERNEL WORD MARKS THE DATA SEGMENT AS WORD-ORIENTED. (BIT 11 = 0)