TITLE: Protection & Relocation for the PDP-11/40

PDP-11/40 Technical Memo #29

AUTHOR: Ad van de Goor
Robert Gray
K.C. Huang

DATE: March 8,1971

REVISION: C OBSOLETE: Memo #29 Feb 24, 1971

INDEX KEY: Relocate/Protect
Segmentation

DISTRIBUTION: PDP-11/40 Group
Van Diehl
Ken Stapleford
Hank Spencer
John Hittel
Gordon Bell: CMU
David Parnas: CMU
Nick Pappas

ABSTRACT

The memory mapping or relocate protect scheme proposed for the 11/40 is essentially a segmentation scheme. It is designed in such a way that it is upwards compatible with the 11/60 scheme and does not presume or dictate a particular use.

The scheme provides for a physical address space of 2↑18 bytes and a maximum active virtual address space of 2↑16 bytes. The total virtual address space (i.e. the length of the segment table) is determined under software control.

The active virtual address space can be divided into 8 segments. The size of each segment can vary from 1 to 16 pages (1 page = 256 words). Protection and relocation are provided on the segment level.

An implementation of the relocate/protect option is proposed. It will fit on two QUAD boards. 16 ACTIVE SEGMENT registers and 4 STATUS registers will be provided. A hardware aid is proposed to help recover from NON-RESIDENT faults. Appendices contain suggested recovery routines.

PREFACE

This document is intended to provide a detailed description of the relocate/protect scheme being designed for the PDP-11/40. It further is being used as a "working" set of engineering specifications. As such it will ultimately become the text for the "Engineering Specifications" and the "Maintenance Manual".

New in Revision C are:

1. Segment Length Field and Segment Address Field descriptions.
2. Section 1.2 "Segment Fault Action
3. Use of SSR0 bit 7 to enable/disable Memory Management Trapping.
4. Storing of Trap Vectors in SSR2
5. Section 7.2 Address bits 16 and 17
6. Sections 9.14 and 9.15 Console communication
7. Section 7.3 Address Assignments

1.0 INTRODUCTION

This memo is intended to be a preliminary description. Many details have yet to be worked out. It is our intention to revise this document from time to time with the additional detail.

Because this is a living document, it is extremely important that we know about errors in writing, detail, or most importantly, in plan. We expect there will be questions and contentions raised by this technical description., We remain always ready to listen and try to understand questions you may raise. If we are going a wrong direction, NOW is the time to change! we earnestly request your criticisms and suggestion, all deserve an honest reply.

1.0 BASIC SOLUTION

The addresses generated by the PDP-11/20 are 16-bit byte addresses. On the 11/40 with segmentation these addresses are considered Virtual Address "VA's". A VA is considered to be a two dimensional address as shown in Figure 1. It consists of:

1. The Active Segment Field "ASF". This 3-bit field determines which of 8 Active Segment Registers "ASR's" has to be used to form the Physical Address "PA".

2. The Displacement Field "DF". This is a 13-bit field which contains an address relative to the beginning of a segment. This allows for segment sizes of 2**13=8k bytes.

The formation of a physical address "PA" is shown in Figure 2. The Active Segment Field "ASF" of the Virtual Address "VA" is used to address one of the eight Active Segment Registers "ASR's". The Segment Address Field "SAF" of the addressed ASR is used together with the Displacement Field "DF" to form the PA.

The ASR's can be loaded with Segment Descriptor words "SDW's" under program control.

1.1 THE SEGMENT DESCRIPTOR WORD "SDW"

A SDW is a 16-bit word containing information relevant to a particular segment. A SDW consists of 3 fields, see Figure 3, which are described below.

1. The Access Control Field "ACF". This 3-bit field contains the access rights, called KEYS, of a process with respect to a particular segment. The following keys have been assigned already,

   a. Non Resident "NR" (key=0). Any access to such a segment will cause an abort. The reasons why a segment is

non-resident can be many fold, e.g. not swapped in yet, segment does not exist, etc.

b. Resident Read Only and Trap "RROT" (key=4). This is essentially a read only segment, a trap upon read could be desired to gather statistics about the use of the segment, etc.

c. Resident Read Only "RRO" (key=5). An attempt to write in an RRO segment will cause an abort.

d. Resident Read Write and Trap "RRWT" (key=1). In this segment essentially Read and Write operations are allowed, the trap upon a read or a write access could be desired for statistics gathering (see b above).

e. Resident Read Write and Trap when Write "RRWTW" (key=2). This is a segment where essentially read and write operations are allowed. When a write operation is done, a trap will occur. This could be used to indicate that no valid backup copy exists any more.

f. Resident Read Write and Written into "RRWW" (key=3). This is a segment in which read and write operations are allowed. The "written into" could indicate that no backup copy is available.

2. SEGMENT LENGTH FIELD (SLF)

This four bit field specifies the number of 256 word pages in the segment. From 1 to 16 pages may be specified in this field.

Note that "0" in the SLF specifies 1 page and 15 in the SLF specifies 16 pages.

The four bits of the SLF are compared with the four high order address bits in the DISPLACEMENT FIELD from the processor to detect SEGMENT LENGTH errors. A SEGMENT LENGTH error exists if the SLF is smaller than the four high order bits of the DISPLACEMENT FIELD.

3. SEGMENT ADDRESS FIELD (SAF)

This field of 9 bits in combination with the 13 bit DISPLACEMENT FIELD from the processor form the 18 bit physical address. This process is shown schematically in Figure 3.

Since the 11/40 segmentation scheme allows segments of different sizes (256 word to 4096 words in 256 word increments), a method must be provided for creating segment boundries at 256 word intervals rather than at 4K intervals

if memory is to be fully used. This is implemented by adding the 4 high order address bits in the DISPLACEMENT FIELD into the SAF as shown in Figure 3. This technique allows a VIRTUAL address Ø of a 256 to 4K word segment to be physically located at any 256 word boundry in the memory system. This allows segments of varying lengths to be placed in physical core with no "gaps" between them.

1.2 SEGMENTATION FAULT ACTION

If the processor sends an address that is: non-resident, exceeds segment length or violates "read only" restrictions, the operation is aborted before the memory operation occurs. No memory reference occurs and an ABORT signal is sent to the processor. No further memory references can occur until the processor acknowledges the ABORT signal with a segmentation fault acknowledge (SEGACK) signal. This process is handled automatically in the 11/40 hardware.

If a segment access is made that requires a memory management trap, the processor is notified by a signal (MEM MGMT FLAG). The processor acknowledges this flag only at the end of an instruction. The acknowledgement is signal SEG ACK. See section 6.Ø for a full discussion of this operation.

The MEM MGMT FLAG has priority over "T bit" traps.

2.Ø DEGREE OF HARDWARE AIDS ON SEGMENTATION FAULTS

When a segment fault occurs, if it is a "non-resident" fault, the system must bring the missing segment into core then restart the task. Since such "non-resident" faults can occur within instructions, some means of either restarting the instruction in the middle or backing the instruction up and restarting at the beginning must be provided.

Restarting the instruction in the middle is rejected as there are a number of internal inaccessible registers whose value can be neither saved nor restored for "middle of instruction" restarts.

An investigation of backing up the effects of a partially executed instruction shows that only auto-increment and auto-decrement operations effect the registers and that to restart, it is sufficient to reverse the effects of any auto decrement/incrementing done by the partially completed instruction before restarting at the "instruction address." Further it has been determined that a maximum of two registers are changed during a given instruction.

The next decision must determine the degree of help provided by the segmentation hardware in correcting register values. Clearly a range of help is possible-from fully automatic correction to merely indicating how far the instruction got in SOURCE/DESTINATION calculations.

Previous experience with the KT11 (11/20 Paging option) [which merely provided the EXEC with a bit indicating whether the SRC/DST had been completed.] indicated additional help was needed.

A fully automatic scheme was rejected because it was regarded as "overkill" and the intimate connection to the processor ROM that would have been required was not thought desirable.

The proposed scheme is implemented in Segment Status Register "SSR #1". It allows the EXEC to correct all registers modified in less than 40 memory references. It's hardware implementation is straight forward and does not control the processor ROM. Appendix A is a sample recovery routine. Basically it provides the EXEC with the register number changed and a description of how much and in what direction it was changed.

3.0 SEGMENT STATUS REGISTER #0 (Segmentation status and error indicators)

SSR0 will contain error flags and the "virtual segment number" causing the error as well as other status flags. The register will be organized as in Figure 4.

Bits 15-12 are the error flags. They may be considered to be in a "priority queue" in that "flags to the right" are ignored! That is a "non resident" fault service routine would ignore segment length, access, and memory management flags. A "sgment length" service routine would ignore access and memory management faults, etc. Note that the word format is convenient for: "ROTATE" and "BRANCH" breakout sequences.

Bits 11-8 are presently spares. They may be assigned uses in a "Debugging Option" allowing "hardware breakpoints."

Bit 7 enables MEMORY MANAGEMENT trapping. If bit 7 is 0, Segment Status Register 3 will keep track of which segment references requested memory management traps, but the "trap signal (MGMT TRAP FLAG) is not sent to the processor. When bit 7 is made 1, the next time a segment whose ACF calls for a MEM MGMT trap is

referenced, MGMT TRAP FLAG is sent to the processor.

Bit 6 specifies a MAINTENANCE mode in which only the DESTINATION fetch/store is relocated and protected. It is expected to be useful for diagnostic program development.

Bit 5 indicates that the instruction has completed. It will be set when non-instruction (traps) memory references are made. This provides the error handling routine a way of finding that the last instruction will not have to be repeated on restart.

Bits 4-1 give the virtual segment number of the reference causing a fault. Bit 4 on a "1" indicates USER segments. Note that this field is positioned for convenient relative addressing on the segment number.

Bit Ø this bit controls whether virtual addresses are operated upon by the segmentation hardware. This bit will be cleared by the processor signal INIT.

SSRØ bits Ø,5,6,12-15 can be written into as a word. Other bits (1-4) will not contain valid information after writing into SSRØ until the next Trap occurs.

3.5 DESTINATION ONLY SEGMENTATION

Experience with debugging KT11 diagnostics has shown that a DESTINATION MODE ONLY relocation & protection is desirable. It is proposed that SSR1 bit 6 be used for that purpose, and that segmentation be controlled by the following Boolean equation:

segmentation=SSR1<Ø>(1)+SSR1<6>(1)*DST

The amount of logic required to implement destination only segmentation is estimated to be one 14 pin chip.

4.Ø SEGMENT STATUS REGISTER #1

This register keeps track of any AUTO INCREMENTING/DECREMENTING of the general registers, i.e. PUSHES and POPS. The register is cleared at the beginning of each instruction fetch. Whenever a general register is either pushed or popped, the register number and the 2's complement number of bytes the register got modified are written into SSR1. The low order byte is written first. See

Figure 4.

When the instruction is completed, there is no need to restart the instruction even if there may be segment faults before next instruction fetch. The register SSR1 is cleared and any stack modification from that time till next instruction fetch is recorded.

Register numbers will be recorded "MOD 8". It will be up to the recovery service routine to determine which set of registers was modified by the state of the processor Status Word at the time of trapping.

The existence of SSR1 will speed up the recovery from a non-fatal segment fault by 2-3 times (a saving 100-200 uS) and requires only 1/5 to 1/10 of core (75 words) of the case where the only information available to the EXEC are the SRC/DST bits. Although non-resident segment faults are not expected to occur at a rate more than one per two milli-seconds, thus making the 100-200 uS saving in overhead perhaps seem insignificant; the implementation of SSR1 can probably be justified in terms of the amount of core saved (500 words) and the small amount of hardware that has to be added. It now seems that the 4*256 ROM is required in any case to eliminate some of the logic needed to decode the relevant CP states. The register itself requires nothing extra and the control logic is estimated to be five 14-16 pin chips. A read only memory (ROM) will be used to detect processor states during which AUTO INCREMENTING or DECREMENTING occur. The ROM will "track" the processor Control Memory ROM. That is, the address being sent to the processor ROM will be bussed to the segmentation modules, where it will be used to select the same numbered location in a ROM located in the segmentation option. One bit of the ROM will indicate that an "AUTO" change is occuring. Another bit will indicate the direction (INC/DEC) of the change.

SSR1 is READ ONLY. It cannot be written into.

5.0 STATUS REGISTER #2

SSR2 will contain the 16 bit virtual instruction address. It will be loaded at the beginning of each instruction fetch. It will be loaded with the Trap Vector (TV) address at the beginning of an interrupt or a "T bit" trap.

SSR2 is READ ONLY. It cannot be written into.

6.0 MEMORY MANAGEMENT AND SSR #3

Three of the ACF keys cause a trap under the condition that the segment was referenced. This will be used for swapping control and other memory management functions. Since these are not "error" faults, there is no reason to "abort" the operation when a MEMORY MANAGEMENT trap occurs. It is sufficient to merely "note" that the trap occured. A convenient "unit sampling period" is the "instruction," During a single instruction several (5 max in an indirect BINARY instruction) different segments may cause STATISTICAL traps. Each of these must be retained until the trap is serviced. In addition since the move to user and move from user instructions operate in both USER and EXEC virtual address space, some combination of 16 different segments can be at fault. A MEMORY MANAGEMENT trap causes a bit in the SSR3 to be set. EXEC Segment 0 will set bit 0, USER segment 7 will set bit 15 of the register. See Figure 4.

Once a bit is set in SSR3 signal MEMORY MANAGEMENT TRAP will send MEM MGMT TRAP to the processor. This will cause a trap at the end of the current instruction. The request will be signified by bit #12 in SSR0 becoming a "1". During the service routine the bits in the SSR3 must be reset. (Note additional bits may be set during the service routine.) At the end of the service routine, SSR0 bit 12 must be cleared. This "rearms" the MEMORY MANGEMENT error logic and enables The "T" bits to be set again. An "ABORT" error occuring on a fetch or the location being a register that is internal to the segmentation option will prevent a Memory Management bit from being set, even if the access key calls for a MEM. MGMT. TRAP. SSR3 can be written into as a word.

Note that if bit 7 in SSR0 is "0" [ENABLE MEMORY MANAGEMENT TRAPS], no flag is sent to the processor when the MEM MGMT access keys are detected. The proper bit in SSR3 is set however. This allows "periodic" EXEC memory use checking at EXEC discression rather than checking forced by hardware.

7.0 DETAILED SYSTEM SPECIFICATIONS

7.1 CLEARING SEGMENT STATUS REGISTERS FOLLOWING TRAP

At the end of the segmentation fault service routine certain bits in SSR0 must be cleared to "rearm" the segmentation trap logic.

Bits 12-15 and 1-5 must be cleared to resume segmentation error checking. On the next memory reference following clearing these bits the other SSR's will continue monitoring the computer operation. SSR2 will be loaded with the next instruction address. SSR1 will get register information. If a new

statistical trap were detected, SSR3 will be loaded.

7.2 ADDRESS BITS 16 AND 17

a. SEGMENTED-bits 16 and 17 are specified by the contents of the Active Segment Register. (Max Memory=128K)

b. UNSEGMENTED-bits 16 and 17 are equal to 0 unless 13,14,15 are 1, then bits 16 and 17 are automatically made 1. (Max Memory = 32k)

c. CONSOLE ACCESS-when console ADDRESS SELECT switch is in "PHYSICAL" position, bits 16 and 17 are controlled explicitly by switches 16 and 17 respectively. (examines and deposits only.)

7.3 REGISTER ADDRESS ASSIGNMENTS

777630	SSR0
777632	SSR1
777634	SSR2
777636	SSR3
777640	USER ASR 0
777642	USER ASR 1
777644	USER ASR 2
777646	USER ASR 3
777650	USER ASR 4
777652	USER ASR 5
777654	USER ASR 6
777656	USER ASR 7
777660	EXEC ASR 0
777662	EXEC ASR 1
777664	EXEC ASR 2
777666	EXEC ASR 3
777670	EXEC ASR 4
777672	EXEC ASR 5
777674	EXEC ASR 6
777676	EXEC ASR 7

8.0 FLOATING POINT PROCESSOR

Several considerations regarding the interaction with the Floating Point Unit (FPU) remain unresolved.

Floating point instructions cause general registers to be auto inc/dec by 4 or 8 bytes. These values will be stored in SSR1.

9,0 INTERFACE SIGNAL SPECIFICATIONS

Introduction

The internal registers of the segmentation option will interface with the processor through the "fast bus". In addition a number of other signals will be passed between the two units. It is intended that agreements on the characteristics of these interfacing signals will be catalogued in this section.

9,1 PROCESSOR/SEGMENTATION INTERFACE

9,1,1 Fast Bus Interface

1. VIRTUAL ADDRESS LINES - [VA0 through VA15](16 lines)

   16 lines sending the virtual address from the processor or floating point processor to the segmentation option. A true signal will be ground. These signals will begin and end with the leading edge of "T1".

2. BUS START - BUST (1 line)

   A pulse indicating an address is on the VA's. A true condition will be ground. This signal will begin at the leading edge of "T2".

3. FAST (1 line)

   A signal coming from "memory" that can respond in 150ns from BUS START at processor (250ns in segmented mode). The segmentation will "pass" FAST from the semicondctor memory and generate FAST when one of the 20 internal registers of the segmentation option is addressed by the processor. A true condition will be ground. The pulse will have a nominal width of n.s.

4. CONTROL (C1) (1 line)

   This will be used to differentiate READ and WRIIE memory cycles. [No byte operations will be allowed in registers internal to this option.

5. INTERNAL BUS DATA (16 lines)

   Data to be read from the 4 status registers in the segmentation option will be placed on these lines. True value is ground.

6. BBR DATA (16 lines)

   These are used by the processor to transfer data to one of the 20 internal registers. True value is ground. Signals

are made valid at leading edge of "T1".

7. BUS END - BEND (1 line)

Within 125ns (225ns with segmentation) after BUST, This tells to stop processing the address. No flag/error bits should be set when BEND is decoded as it indicated an aborted fetch. True value is ground.

8. PHYSICAL ADDRESS OUT (18 lines)

The physical address generated by segmentation option.

9. BUST OUT

A delayed (100 ns) and enabled version of Bus Start received from the processor. This signal is generated only if no error occured on the access attempt.

9.1.2 PROCESSOR SIGNALS NEEDED BY SEGMENTATION

1. ROM ADDRESS (8 lines)

The 8 bit ROM ADDRESS will be needed by segmentation to decode "pushes",+"pops" to registers for SSR 2. It will also be used to decode DESTINATION FETCH, INSTRUCTION DONE. All signals will be true on a +3V logic level.

2. REGISTER NUMBER (3 lines)

The 3 bit register number of the register being pushed or popped.

3. CONSTANT (4 lines)

A 4 bit positive number that is to be added or subtracted from the register.

4. USER/EXEC Address (1 line)

Not Status Register bit 15, but a signal that indicates whether the address on the fast bus is a USER or EXEC address. Required because of INTERmode instructions!

5. CLOCK - (several lines)

To be used for various timing and syncronizing functions within the segmentation option.

6. LOAD IR - signal indicating this is an INSTRUCTION FEICH.

7. SEG ACK (1 line)

A pulse from the processor that lowers the ABORT and MEM MGMT trap flags. Once ABORT or MEM MGMT is true no further memory references can occur until SEG ACK is received.

8. INIT (1 line)

A power clear signal that occurs during powerup and when the console START switch is depressed.

9.1.3 SIGNALS NEEDED BY THE PROCESSOR

1. ABORT (1 line)

Made true when segmentation fault occurs. Will be cleared by SEG ACK.

2. MEM MGMT TRAP (1 line)

A level will be used for MEM MGMT traps. It will be cleared by SEG ACK.

3. SEGMENTED/UNSEGMENTED

TRUE LEVEL SIGNAL WHEN SEGMENTATION OPTION IS IN PLACE AND BIT 0 OF SSR 1 is a 1.

9.1.4 CONSOLE SIGNALS REQUIRED BY SEGMENTATION

1. EX ADDRESS 16 and 17 (2 lines)

Those are the output of console swithces 16 and 17.

2. CONSOLE INHIBIT SEGMENTATION (1 line)

Tells the segmentation option to use the values of EX ADDRESS 16 and 17 to form the high order physical address bits.

3. VIRTUAL (1 line)

When True sends the VIRTUAL address to the console ADDRESS lamps. When false sends the physical address.

9.1.5 SEGMENTATION SIGNALS REQUIRED BY CONSOLE

1. CONSOLE ADDRESS (18 lines)

Output of a multiplexer which provides the console with either the PHYSICAL or VIRTUAL address.

10.0 LOGIC REQUIREMENTS FOR SEGMENTATION

Our estimate is that the RELOCATION/PROTECTION can fit on two "QUAD" boards. The following chips are presently planned to be used:

DEC#	DEV	PINS	DESC	QUANTITY	COST	TOTAL
19-09930	7405	14	Hex INVT OPEN COL	3	.22	
19-10155	7408	14	Quad AND	2		
19-10091	7437	14	Quad NAND Buf	1		
19-09050	7475	14	Quad D FLIP FLOP	4	1.00	
19-09937	74153	16	Dual 4 to 1 Mux	8	.97	
19-09814	74154	16	4 to 16	1	1.91	
	74157	16	Quad 2 to 1 Mux	3		
	74174	16	Hex D Flip Flop	2		
	74175	16	Quad D Flip Flop	3		
	74187	16	ROM	1		
19-09056	74H00	14	4 2IN NAND	4	.24	
19-09931	74H04	14	6 INVT	8	.28	
19-09057	74H10	14	3 3IN NAND	2	.24	
19-09267	74H11	14	3 3IN AND	3	.24	
19-09058	74H21	14	2 4IN AND	1	.24	
19-09059	74H30	14	8 IN NAND	1	.24	
19-05586	74H40	14	2 4IN NAND BUF	6	.26	
19-09063	74H55	14	AND OR INV	1	.24	
19-09667	74H74	14	2 D FLIP FLOPS	1	.48	
	74s03	14	INVT OPER COL	1		
	74S04	14	INVT	8		
	74S15	14	3 3 IN NAND OPER COL	1		
	74S64	14	AND OR INVT	2		
	74S158	16	Quad 2 to 1 Mux	4		
	74S181	24	ALU	4		
	74182	16	LOOK AHEAD	1		
19-10087	4015	16	4 bit D with Set	5	.87	
	3101(Intel)	16	35 ns Max	4		

				Total 84		

APPENDIX A: EXAMPLE OF "NON-RESIDENT" RECOVERY PROGRAM

```
TRAPVECTOR:
        :EXEC,REGSET=0,NONINTERRUPTIBLE
HERE:   BIT     #B14,2(SP)      /WHICH REG WAS IN USE?
        BEQ     PUSHLDS         /WAS NORMAL
        BIS     #B14,STATUS     /WAS DEDICATED, ACTIVATE
                                /DECIDATED
PUSHLDS:PLODM   ,R6-R0          /PUT OLD REG ON STACK

        MOV     SSR2,R2
AUTOCK: BIT     #B7,6,5,4,3,R2  /TEST FOR A REGISTER
                                /AUTO INC/DEC
        BEQ     DONE            /NONE CHANGED
        MOVB    R2,R1           /WAS CHANGED, GET
                                /PARAMETERS
        MOVB    R1,R0           /SAVE ADDITIONAL COPY
        ASL     R1              /MOVE REG# TO WORD
                                /BOUNDRY
        BIC     #377741,R1      /CLEAR OUT OTHER BITS 8,
        ASH     R0,#3           /MOVE CONSTANT INTO
                                /PROPER POSITION
        ADD     R6,R1           /R1 GETS LOC IN STACK
                                /WITH REG. VALUE
        ADD     R0,(R1)         /CORRECT OLD REG VALUE
        CLRB    R2
        SWAB    R2              /SETUP FOR SECOND TS?
        BR      AUTOCK
ERRTYP: MOV     SSR0,R0
        BIT     #B15,R0
        BEQ     ,+2
        JSR     R7,NONRES       /NONRESIDENT ROUTINE
        BIT     #B14,R0
        BEQ     ,+2
        JSR     R7,PROTECT      /BOUNDRY ROUTINE
        JSR     R7,ACCESS       /WRITE ERROR ROUTINE

TBIT:   BIT     #B5,R0          /
        BEQ     INST            /INSTRUCTION COMPLETED
EMUL:   MOV     #12,R5          /TRAP DIDN'T COMPLETE,
        ADD     R6,R5           /EMULATE.
        MOV     'R5)+,R0        /OLD USER PC
        MOV     (R5)+,R1        /OLD USER PS
        MOV     @#16,-(R5)      /PUT TRAP VECTOR
        MOV     @#14,-(R5)      /MONITOR STACK
        MOV     #10,R5          /LOC OF USER R6
        ADD     R6,R5           /     "
        MTU     @-(R5)          /PUSH PC TO USER STACK
        MTU     @-(R5)          /PUSH PS TO USER STACK

INST:   MOV     #,R5            /INST TO COMPLETE
        ADD     R6,R5           /SET PROPER STARTING
        MOV     SSR2,(R5)       /ADDRESS IN STACK
```

```
RESTRT: POPM    R0-R6           /MACRO REG RESTORE
        BIS     #000001,SSR0    /REARM SEG CHECKING
        RTI
^C
```

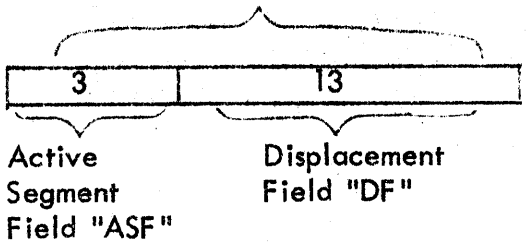


FIGURE 1. Interpretation of a Virtual Address

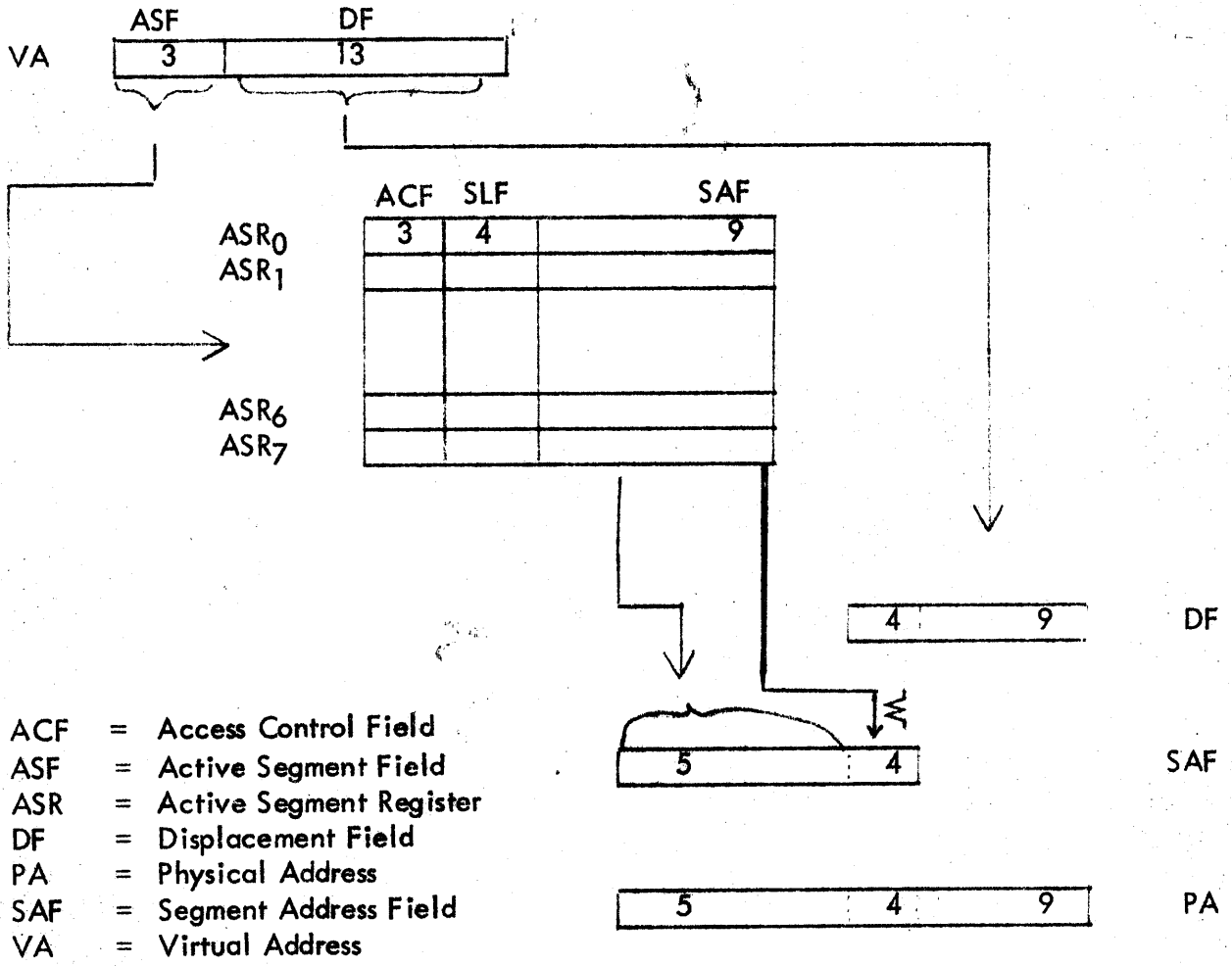
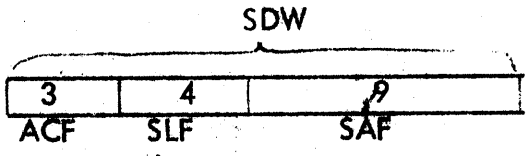


FIGURE 2. Formation of a Physical Address



ACF = Access Control Field
SAF = Segment Address Field
SDW = Segment Descriptor Word
SLF = Segment Length Field

FIGURE 3. Layout of a Segment

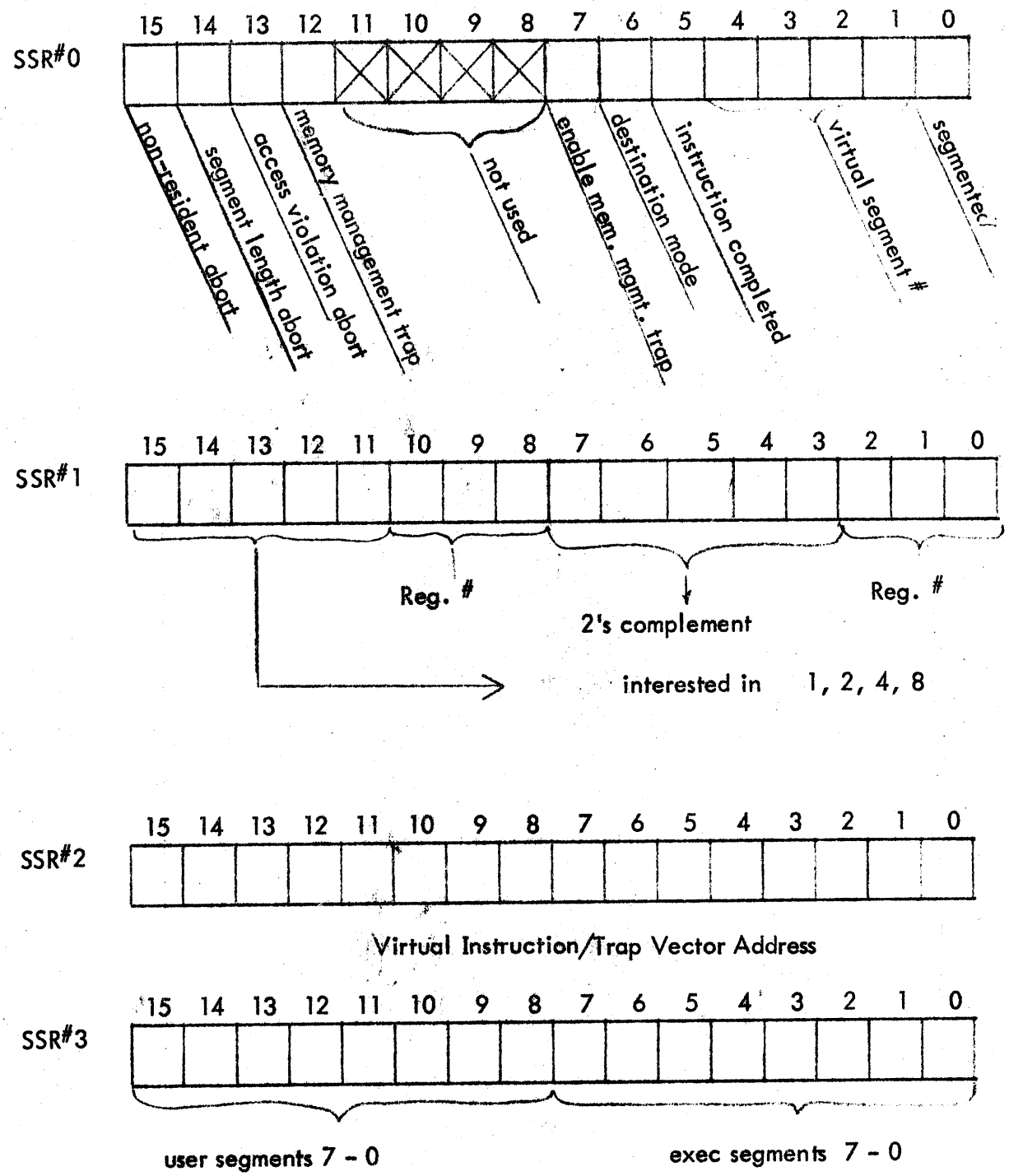


FIGURE 4. Segment Status Register Formats