;
Table of Contents

;
Table of Contents

;
Table of Contents

```
        .NOBIN
.TITLE  "KS10 MICROCODE V123, 1 MARCH, 1982 -- SEAN KEENAN, DON DOSSA"

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;                                                                ;;;
;;;                                                                ;;;
;;;     COPYRIGHT (C) 1976,1977,1978,1979,1980,1981,1982           ;;;
;;;       DIGITAL EQUIPMENT CORP., MAYNARD, MASS.                  ;;;
;;;                                                                ;;;
;;;     THIS SOFTWARE IS FURNISHED UNDER A LICENSE  FOR  USE       ;;;
;;;     ONLY  ON  A SINGLE COMPUTER SYSTEM AND MAY BE COPIED       ;;;
;;;     ONLY WITH  THE  INCLUSION  OF  THE  ABOVE  COPYRIGHT       ;;;
;;;     NOTICE.  THIS SOFTWARE, OR ANY OTHER COPIES THEREOF,       ;;;
;;;     MAY NOT BE PROVIDED OR OTHERWISE MADE  AVAILABLE  TO       ;;;
;;;     ANY  OTHER  PERSON EXCEPT FOR USE ON SUCH SYSTEM AND       ;;;
;;;     TO ONE WHO AGREES TO THESE LICENSE TERMS.  TITLE  TO       ;;;
;;;     AND  OWNERSHIP  OF  THE  SOFTWARE SHALL AT ALL TIMES       ;;;
;;;     REMAIN IN DEC.                                             ;;;
;;;                                                                ;;;
;;;     THE INFORMATION  IN  THIS  DOCUMENT  IS  SUBJECT  TO       ;;;
;;;     CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS       ;;;
;;;     A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.             ;;;
;;;                                                                ;;;
;;;     DEC  ASSUMES  NO  RESPONSIBILITY  FOR  THE  USE   OR       ;;;
;;;     RELIABILITY  OF  ITS  SOFTWARE IN EQUIPMENT WHICH IS       ;;;
;;;     NOT SUPPLIED BY DEC.                                       ;;;
;;;                                                                ;;;
;;;     DESIGNED AND WRITTEN BY:                                   ;;;
;;;             DONALD A. LEWINE                                   ;;;
;;;             DIGITAL EQUIPMENT CORP.                            ;;;
;;;             MARLBORO, MASS.                                    ;;;
;;;             MR1-2/E47  X6430                                   ;;;
;;;                                                                ;;;
;;;     MAINTAINED AND ENHANCED BY:                                ;;;
;;;             DONALD D. DOSSA                                    ;;;
;;;             DIGITAL EQUIPMENT CORP.                            ;;;
;;;             MARLBORO, MASS.                                    ;;;
;;;             MR1-2/E18  DTN 231-4138                            ;;;
;;;                                                                ;;;
;;;             SEAN KEENAN                                        ;;;
;;;             DIGITAL EQUIPMENT CORP.                            ;;;
;;;             MARLBORO, MASS.                                    ;;;
;;;             MR1-2/E18  DTN 231-4463                            ;;;
;;;                                                                ;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

```
;  51     .TOC    "REVISION HISTORY"
;  52
;  53     ;REV    WHY
;  54     ;1      START KS10 MICROCODE BASED ON SM10 MICROCODE VERSION 510
;  55     ;2      UPDATE TO KS10 VERSION 512
;  56     ;3      FIX SOME DEFAULTS
;  57     ;4      CHANGE HARDWARE TO MATCH ECO #215
;  58     ;5      START TO UPDATE IO MICROCODE
;  59     ;6      MORE WORK ON IO
;  60     ;7      MAKE INTERRUPT THE 8080 BE A PULSE.
;  61     ;10     ADD NEW RDIO AND WRIO
;  62     ;11     FIX PROBLEMS IN MUUO CODE & CORRECT T-FIELDS
;  63     ;12     FIX PROBLEMS IN DDIV
;  64     ;13     FIX UP PROBLEMS IN PI
;  65     ;14     TURN ON WRITE FOR FL-EXIT
;  66     ;15     FIX UP MAP INSTRUCTION
;  67     ;16     MORE WORK ON KI-STYLE MAP
;  68     ;17     INVERT HOLD RIGHT AND HOLD LEFT BITS
;  69     ;20     FIXUP WRIO & RDIO EFFECTIVE ADDRESS CALC.
;  70     ;21     FIX EDIT 15
;  71     ;22     HAVE LSH USE FAST SHIFT HARDWARE
;  72     ;23     FIX T-FIELD VALUES FOR PRODUCTION HARDWARE
;  73     ;24     REMOVE WRITE TEST FROM IO READS & WRITES
;  74     ;25     REWRITE MUL & MULI TO BE FASTER AND SMALLER. ALSO MAKE ADJBP
;  75     ;       USE NEW MULSUB
;  76     ;26     MAKE BYTES USE FAST SHIFT ECO.
;  77     ;27     MAKE SURE VMA FETCH IS CORRECT
;  78     ;30     MORE OF 25 (FORGOT FMP)
;  79     ;31     FIX SOME PROBLEMS WITH TRAPS
;  80     ;32     SPEED UP EFFECTIVE ADDRESS CALCULATION
;  81     ;33     MORE OF 32
;  82     ;34     SPEED UP ASH & ROT
;  83     ;35     FIX UP RDTIM SO THAT TIME DOES NOT GO BACKWARDS
;  84     ;36     MORE OF 35
;  85     ;37     FIX UP PROBLEMS WITH INTERRUPTS AND DOUBLE F.P.
;  86     ;40     IMPROVE LISTING FORMAT
;  87     ;41     SPEEDUP KL-MODE PAGE REFILL
;  88     ;42     FIX UP DDIV
;  89     ;43     STILL MORE DDIV STUFF
;  90     ;44     CORRECT PROBLEMS IN D.P. PARITY STUFF
;  91     ;45     CORRECT THE BLT CLEAR-CORE CASE TO INTERRUPT CORRECTLY
;  92     ;46     MORE OF 45
;  93     ;47     DO NOT ALLOW SOFTWARE INTERRUPTS IF THE PI LEVEL IS NOT
;  94     ;       ACTIVE.
;  95     ;50     MAKE FDV WORK THE SAME AS THE KL10
;  96     ;51     FIX INTERRUPT IN CVTBDX. MAKE ABORT WORK LIKE SPEC.
;  97     ;52     FIX BUG IN HALT LOOP
;  98     ;53     FIX IOEA TO WORK IF NO @ OR INDEXING
;  99     ;54     EDIT 47 BROKE JEN
; 100     ;55     FIX FLAGS IN MULTIPLY. ALSO CODE BUMS
; 101     ;56     MORE CODE BUMS
; 102     ;57     CORRECT OVERFLOW TRAPS WHICH DO MUUOS TO NOT STORE
; 103     ;       THE TRAP FLAGS.
; 104
```

```
; 105    ;60     CORRECT TRAPS SO THAT DSKEA RUNS RIGHT
; 106    ;61     MORE OF 60. NOTE: MICROCODE REQUIRES ECO #299!!
; 107    ;62     ONE MORE TRY AT EDIT 60.
; 108    ;63     CORRECT TOPS-10 STYLE PAGING SO THAT A WRITE VIOLATION SETS
; 109    ;       BIT 2 IN THE PAGE FAIL WORD (ACCESS ALLOWED).
; 110    ;64     EDIT 63 BROKE HARD PAGE FAILS. (NXM, BAD DATA, AND IO NXM)
; 111    ;65     INTERRUPTS OUT OF MOVSRJ INSTRUCTIONS DO STRANGE THINGS.
; 112    ;66     IO NXM PAGE FAIL FOR MISSING UBA GIVES PC+1 IN PAGE FAIL BLOCK.
; 113    ;67     ON A BAD DATA ERROR, STORE THE BAD WORD IN AC BLOCK 7 WORD 0 AND
; 114    ;       1
; 115    ;70     FIX A BUG WHICH CAUSED INTERRUPTS OUT OF CVTBDT TO GENERATE A BAD
; 116    ;       ANSWER.
; 117    ;71     CLEANUP SOME THINGS TO MAKE LIFE EASIER FOR FIELD SERVICE
; 118    ;72     LOOK FOR 1-MS TRAP ON @ PAGE POINTERS AND ABORT REFILL IF
; 119    ;       SET.
; 120    ;73     CORRECT EDIT 72.
; 121    ;74     EDIT 67 GENERATES A DATA PATH PARITY ERROR BECAUSE OF THE BAD
; 122    ;       DATA. CORRECT TO NOT CHECK PARITY.
; 123    ;       ALSO CHANGE POP TO TIE UP BUS LESS.
; 124    ;75     EDIT 60 BROKE TRAPS. MISSING =0 AT TRAP:.
; 125    ;76     CORRECT BUG IN DFAD AND DFSB
; 126    ;77     FIX PROBLEM SEEN IN SOME (ALL BUT ENGINEERING?) MACHINES CAUSED
; 127    ;       BY EDIT 76
; 128    ;100    CHANGE DFAD/DFSB TO HAVE 2 MORE GUARD BITS. THIS SHOULD PRODUCE
; 129    ;       KL10 ANSWERS FOR ALL NORMALIZED INPUTS
; 130    ;       ALSO FIX A BUG IN CVTBDX PAGE FAIL LOGIC.
; 131    ;101    DFDV OF 0.0 / -0.5 HANGS THE MACHINE
; 132    ;102    FIX CHOPPED FLOATING POINT INSTRUCTIONS
; 133    ;103    CORRECT DFDV ROUNDING BUG.
; 134    ;104    CORRECT PROBLEMS IN DFMP
; 135    ;105    RDTIME SOMETIMES GIVES WRONG ANSWER. CARRY BETWEEN
; 136    ;       WORDS GETS LOST SOMETIME.
; 137    ;106    MOVEM (ALSO, SETZM, SETOM, ETC.) SOMETIMES DOES NOT GENERATE
; 138    ;       A WRITE-TRAP IN 100% OF THE CASES THAT IT SHOULD.
; 139    ;107    PXCT 14, DOES NOT GET THE INDEX REGISTER IN THE PREVIOUS
; 140    ;       CONTEXT ALL THE TIME.
; 141    ;110    FIX TYPO IN EDIT 103
; 142    ;111    63. BIT BYTES DO NOT WORK CORRECTLY. DSKDA FAILS BECAUSE OF THIS
; 143    ;       PROBLEM.
; 144    ;******* VERSION 111 WENT OUT WITH SYSTEM REV 2 *******
; 145
; 146    ;112    FIX COMMENT IN TEST INSTRUCTIONS
; 147    ;113    CORRECT IOEA TO COMPUTE CORRECT ADDRESS IF JUST LOCAL INDEXING
; 148    ;       IS USED.
; 149    ;114    CORRECT INTERRUPT BUG IN DMUL
; 150    ;115    CORRECT COMMENTS HALT STATUS BLOCK
; 151    ;116    CORRECT PROBLEM WHERE CST MODIFIED BIT GETS SET BY MISTAKE.
; 152    ;117    RDINT INSTRUCTION DOES NOT WORK AT ALL. IT STORES RANDOM TRASH
; 153    ;       IN THE WRONG PLACE. NEED TO LOAD BR NOT AR.
; 154    ;120    FLOATING POINT OPERATIONS SOMETIMES GET THE WRONG RESULT WITH
; 155    ;       INPUTS OF UNNORMALIZED ZEROS. THIS SHOULD NEVER HAPPEN WITH
; 156    ;       FORTRAN OR ANY OTHER DEC LANGUAGE.
; 157    ;121    PREVENT KEEP-ALIVE CRASHES FROM OCCURRING BECAUSE THE MOVSRJ
; 158    ;       INSTRUCTION CAN LOCK OUT THE 1MS TIMER INTERRUPTS FROM BEING
; 159    ;       HANDLED. THIS CAUSES THE OPERATING SYSTEM TO LOSE TRACK OF THE
; 160    ;       PASSAGE OF TIME.
```

```
; 161    ;122    DFAD FOLLOWED BY A FSC OF -5 CAUSES THE FSC TO GET WRONG
; 162    ;       ANSWER. HAD TO CLEAR FLAG WORD AT EXIT OF DFAD TO FIX PROBLEM
; 163    ;123    MORE CODE FOR EDIT 121. ADDED ANOTHER DISPATCH ADDRESS FOR
; 164    ;       PAGE FAIL CODE AT PFD:.
; 165
; 166
```

```
; 167    .TOC    "HOW TO READ THE MICROCODE"
; 168
; 169    ;
; 170    ;
; 171    ;               1.0 FIELD DEFINITIONS
; 172    ;
; 173    ;               THESE OCCUR AT THE BEGINNING OF THE LISTING, IN THE SOURCE FILE KS10.MIC
; 174    ;               (CONTROL AND DISPATCH RAM DEFINITIONS).  THEY HAVE THE FORM:
; 175    ;
; 176    ;                       SYMBOL/=<L:R>M,J
; 177    ;
; 178    ;               THE PARAMETER (J) IS MEANINGFUL WHEN "D" IS SPECIFIED AS THE DEFAULT MECHANISM,
; 179    ;               AND IN THAT CASE, GIVES THE DEFAULT VALUE OF THE FIELD IN OCTAL.  WHEN "F" IS
; 180    ;               SPECIFIED AS THE DEFAULT MECHANISM, (J) IS THE NAME OF A FIELD WHICH CONTAINS
; 181    ;               THE DEFAULT VALUE FOR THIS FIELD.
; 182    ;
; 183    ;               THE PARAMETER (L) GIVES THE BIT POSITION OF THE LEFTMOST BIT IN THE FIELD.  THE
; 184    ;               SAME METHOD IS USED AS FOR (R) BELOW.
; 185    ;
; 186    ;               THE PARAMETER (R) GIVES THE FIELD POSITION IN DECIMAL AS THE BIT NUMBER OF THE
; 187    ;               RIGHTMOST BIT OF THE FIELD.  BITS ARE NUMBERED FROM 0 ON THE LEFT.  NOTE THAT
; 188    ;               THE POSITION OF BITS IN THE MICROWORD SHOWN IN THE LISTING BEARS NO RELATION TO
; 189    ;               THE ORDERING OF BITS IN THE HARDWARE MICROWORD, WHERE FIELDS ARE OFTEN BROKEN UP
; 190    ;               AND SCATTERED.
; 191    ;
; 192    ;               THE PARAMETER (M) IS OPTIONAL, AND SELECTS A DEFAULT MECHANISM FOR THE FIELD.
; 193    ;               THE LEGAL VALUES OF THIS PARAMETER ARE THE CHARACTERS "D", "F", "T", "P", OR
; 194    ;               "+".
; 195    ;
; 196    ;                       "D" MEANS (J) IS THE DEFAULT VALUE OF THE FIELD IF NO EXPLICIT VALUE IS
; 197    ;                       SPECIFIED.
; 198    ;
; 199    ;                       "F" IS USED TO CAUSE THIS FIELD TO DEFAULT TO SOME OTHER FIELD.
; 200    ;
; 201    ;                       "T" IS USED ON THE TIME FIELD TO SPECIFY THAT THE VALUE OF THE FIELD
; 202    ;                       DEPENDS ON THE TIME PARAMETERS SELECTED FOR OTHER FIELDS.  "T" IS NOT
; 203    ;                       USED IN THE KS10.
; 204    ;
; 205    ;                       "P" IS USED ON THE PARITY FIELD TO SPECIFY THAT THE VALUE OF THE FIELD
; 206    ;                       SHOULD DEFAULT SUCH THAT PARITY OF THE ENTIRE WORD IS ODD.  "P" IS NOT
; 207    ;                       USED ON THE KS10.
; 208    ;
; 209    ;                       "+" IS USED ON THE JUMP ADDRESS FIELD TO SPECIFY THAT THE DEFAULT JUMP
; 210    ;                       ADDRESS IS THE ADDRESS OF THE NEXT INSTRUCTION ASSEMBLED (NOT, IN
; 211    ;                       GENERAL, THE CURRENT LOCATION +1).
; 212    ;
; 213    ;               IN GENERAL, A FIELD CORRESPONDS TO THE SET OF BITS WHICH PROVIDE SELECT INPUTS
; 214    ;               FOR MIXERS OR DECODERS, OR CONTROLS FOR ALU'S.
; 215    ;
; 216    ;
; 217
```

```
; 218    ;
; 219    ;
; 220    ;
; 221    ;
; 222    ;
; 223    ;
; 224    ;
; 225    ;
; 226    ;
; 227    ;
; 228    ;
; 229    ;
; 230    ;
; 231    ;
; 232    ;
; 233    ;
; 234    ;
; 235    ;
; 236    ;
; 237    ;
; 238    ;
; 239    ;
; 240    ;
; 241    ;
; 242    ;
; 243    ;
; 244    ;
; 245    ;
; 246    ;
; 247    ;
; 248    ;
; 249    ;
; 250    ;
; 251    ;
; 252    ;
; 253    ;
; 254    ;
; 255    ;
; 256    ;
; 257    ;
; 258    ;
; 259    ;
; 260    ;
; 261    ;
; 262    ;
; 263    ;
; 264    ;
; 265    ;
; 266    ;
; 267
```

2.0 VALUE DEFINITIONS

FOLLOWING A FIELD DEFINITION, SYMBOLS MAY BE CREATED IN THAT FIELD TO CORRESPOND TO VALUES OF THE FIELD. THE FORM IS:

```
SYMBOL=N
```

"N" IS, IN OCTAL, THE VALUE OF SYMBOL WHEN USED IN THE FIELD.

3.0 LABEL DEFINITIONS

A MICRO INSTRUCTION MAY BE LABELLED BY A SYMBOL FOLLOWED BY COLON PRECEDING THE MICROINSTRUCTION DEFINITION. THE ADDRESS OF THE MICROINSTRUCTION BECOMES THE VALUE OF THE SYMBOL IN THE FIELD NAMED "J". EXAMPLE:

```
FOO:  J/FOO
```

THIS IS A MICROINSTRUCTION WHOSE "J" FIELD (JUMP ADDRESS) CONTAINS THE VALUE "FOO". IT ALSO DEFINES THE SYMBOL "FOO" TO BE THE ADDRESS OF ITSELF. THEREFORE, IF EXECUTED BY THE MICROPROCESSOR, IT WOULD LOOP ON ITSELF.

4.0 COMMENTS

A SEMICOLON ANYWHERE ON A LINE CAUSES THE REST OF THE LINE TO BE IGNORED BY THE ASSEMBLER. THIS TEXT IS AN EXAMPLE OF COMMENTS.

5.0 MICROINSTRUCTION DEFINITION

A WORD OF MICROCODE IS DEFINED BY SPECIFYING A FIELD NAME, FOLLOWED BY SLASH (/), FOLLOWED BY A VALUE. THE VALUE MAY BE A SYMBOL DEFINED FOR THAT FIELD, AN OCTAL DIGIT STRING, OR A DECIMAL DIGIT STRING (DISTINGUISHED BY THE FACT THAT IT CONTAINS "8" AND/OR "9" AND/OR IS TERMINATED BY A PERIOD). SEVERAL FIELDS MAY BE SPECIFIED IN ONE MICROINSTRUCTION BY SEPARATING FIELD/VALUE SPECIFICATIONS WITH COMMAS. EXAMPLE:

```
AD/ZERO,RAMADR/AC*#,ACALU/AC+N,ACN/1,DBUS/DP
```

6.0 CONTINUATION

THE DEFINITION OF A MICROINSTRUCTION MAY CONTINUED ONTO TWO OR MORE LINES BY BREAKING IT AFTER ANY COMMA. IN OTHER WORDS, IF THE LAST NON-BLANK, NON-COMMENT CHARACTER ON A LINE IS A COMMA, THE INSTRUCTION SPECIFICATION IS CONTINUED ON

```
; 268    ;                THE FOLLOWING LINE.  EXAMPLE:
; 269    ;                        READ [AR],              ;LOOK AT EFFECTIVE ADDRESS
; 270    ;                        SKIP DP18,              ;SEE IF RIGHT OR LEFT SHIFT
; 271    ;                        SC_SHIFT-1,              ;PUT NUMBER OF PLACE TO
; 272    ;                                                ;SHIFT IN SC
; 273    ;                        LOAD FE,                ; AND IN FE
; 274    ;                        INST DISP               ;GO DO THE SHIFT
; 275    ;
; 276    ;
; 277    ;
; 278    ;                7.0 MACROS
; 279    ;
; 280    ;                A MACRO IS A SYMBOL WHOSE VALUE IS ONE OR MORE FIELD/VALUE SPECIFICATIONS AND/OR
; 281    ;                MACROS.   A  MACRO  DEFINITION IS A LINE CONTAINING THE MACRO NAME FOLLOWED BY A
; 282    ;                QUOTED STRING WHICH IS THE VALUE OF THE MACRO.  EXAMPLE:
; 283    ;
; 284    ;                        LOAD VMA "MEM/1, LDVMA/1
; 285    ;
; 286    ;                THE APPEARANCE OF A MACRO IN A MICROINSTRUCTION DEFINITION IS EQUIVALENT TO  THE
; 287    ;                APPEARANCE OF ITS VALUE.
; 288    ;
; 289    ;                MACRO MAY HAVE PARAMETERS ENCLOSED IN [].  FOR EXAMPLE,
; 290    ;
; 291    ;                        []_[] "AD/A,A/@2,DEST/AD,B/@1"
; 292    ;
; 293    ;                THE @1 GETS REPLACED BY WHAT IS WRITTEN IN  THE  FIRST  SET  OF  []  AND  @2  IS
; 294    ;                REPLACED BY WHAT IS WRITTEN IN THE SECOND SET OF [].  THUS
; 295    ;
; 296    ;                        [AR]_[ARX]
; 297    ;
; 298    ;                HAS THE SAME EFFECT AS SAYING
; 299    ;
; 300    ;                        AD/A,A/ARX,DEST/AD,B/AR
; 301    ;
; 302    ;
; 303    ;                        SEE DESCRIPTION OF RULES FOR MACRO NAMES.
; 304    ;
; 305    ;                8.0 PSEUDO OPS
; 306    ;
; 307    ;                        THE MICRO ASSEMBLER HAS 13 PSEUDO-OPERATORS:
; 308    ;
; 309    ;                        .DCODE AND .UCODE SELECT THE RAM INTO WHICH SUBSEQUENT MICROCODE WILL BE
; 310    ;                        LOADED,  AND  THEREFORE  THE  FIELD  DEFINITIONS  AND  MACROS  WHICH ARE
; 311    ;                        MEANINGFUL IN SUBSEQUENT MICROCODE
; 312    ;                        .TITLE DEFINES A STRING OF TEXT TO APPEAR IN THE PAGE HEADER, AND
; 313    ;                        .TOC DEFINES AN ENTRY FOR THE TABLE OF CONTENTS AT THE BEGINNING.
; 314    ;                        .SET DEFINES THE VALUE OF A CONDITIONAL ASSEMBLY PARAMETER,
; 315    ;                        .CHANGE REDEFINES A CONDITIONAL ASSEMBLY PARAMETER,
; 316    ;                        .DEFAULT ASSIGNS A VALUE TO AN UNDEFINED PARAMETER.
; 317    ;                        .IF ENABLES ASSEMBLY IF THE VALUE OF THE PARAMETER IS NOT ZERO,
; 318    ;                        .IFNOT ENABLES ASSEMBLY IF THE PARAMETER VALUE IS ZERO, AND
; 319    ;                        .ENDIF RE-ENABLES ASSEMBLY IF SUPPRESSED BY THE PARAMETER NAMED.
; 320    ;                        .NOBIN TURNS OFF THE BINARY A GETS RID OF THE SPACE USED TO LIST IT,
; 321    ;                        .BIN TURN BINARY BACK ON AGAIN.
; 322    ;                        .WIDTH CONTROLS THE NUMBER OF BITS IN THE CRAM
; 323
```

9.0 LOCATION CONTROL

A MICROINSTRUCTION "LABELLED" WITH A NUMBER IS ASSIGNED TO THAT ADDRESS.

THE CHARACTER "=" AT THE BEGINNING OF A LINE, FOLLOWED BY A STRING OF 0'S, 1'S, AND/OR *'S, SPECIFIES A CONSTRAINT ON THE ADDRESS OF FOLLOWING MICROINSTRUCTIONS. THE NUMBER OF CHARACTERS IN THE CONSTRAINT STRING (EXCLUDING THE "=") IS THE NUMBER OF LOW-ORDER BITS CONSTRAINED IN THE ADDRESS. THE MICROASSEMBLER ATTEMPTS TO FIND AN UNUSED LOCATION WHOSE ADDRESS HAS 0 BITS IN THE POSITIONS CORRESPONDING TO 0'S IN THE CONSTRAINT STRING AND 1 BITS WHERE THE CONSTRAINT HAS 1'S. ASTERISKS DENOTE "DON'T CARE" BIT POSITIONS.

IF THERE ARE ANY 0'S IN THE CONSTRAINT STRING, THE CONSTRAINT IMPLIES A BLOCK OF <2**N> MICROWORDS, WHERE N IS THE NUMBER OF 0'S IN THE STRING. ALL LOCATIONS IN THE BLOCK WILL HAVE 1'S IN THE ADDRESS BITS CORRESPONDING TO 1'S IN THE STRING, AND BIT POSITIONS DENOTED BY *'S WILL BE THE SAME IN ALL LOCATIONS OF THE BLOCK.

IN SUCH A CONSTRAINT BLOCK, THE DEFAULT ADDRESS PROGRESSION IS COUNTING IN THE "0" POSITIONS OF THE CONSTRAINT STRING, BUT A NEW CONSTRAINT STRING OCCURING WITHIN A BLOCK MAY FORCE SKIPPING OVER SOME LOCATIONS OF THE BLOCK. WITHIN A BLOCK, A NEW CONSTRAINT STRING DOES NOT CHANGE THE PATTERN OF DEFAULT ADDRESS PROGRESSION, IT MERELY ADVANCES THE LOCATION COUNTER OVER THOSE LOCATIONS. THE MICROASSEMBLER WILL LATER FILL THEM IN.

A NULL CONSTRAINT STRING ("=" FOLLOWED BY ANYTHING BUT "0", "1", OR "*") SERVES TO TERMINATE A CONSTRAINT BLOCK. EXAMPLES:

```
=0
```

THIS SPECIFIES THAT THE LOW-ORDER ADDRESS BIT MUST BE ZERO-- THE MICROASSEMBLER FINDS AN EVEN-ODD PAIR OF LOCATIONS, AND PUTS THE NEXT TWO MICROINSTRUCTIONS INTO THEM.

```
=11
```

THIS SPECIFIES THAT THE TWO LOW-ORDER BITS OF THE ADDRESS MUST BOTH BE ONES. SINCE THERE ARE NO 0'S IN THIS CONSTRAINT, THE ASSEMBLER FINDS ONLY ONE LOCATION MEETING THE CONSTRAINT.

```
=0*****
```

THIS SPECIFIES AN ADDRESS IN WHICH THE "40" BIT IS ZERO. DUE TO THE IMPLEMENTATION OF THIS FEATURE IN THE ASSEMBLER, THE DEFAULT ADDRESS PROGRESSION APPLIES ONLY TO THE LOW-ORDER 5 BITS, SO THIS CONSTRAINT FINDS ONE WORD IN WHICH THE "40" BIT IS ZERO, AND DOES NOT ATTEMPT TO FIND ONE IN WHICH THAT BIT IS A ONE. THIS LIMITATION HAS BEEN CHANGED WITH NEWER ASSEMBLER VERSIONS. HOWEVER NONE OF THE LOCATIONS IN THE MICROCODE REQUIRE ANYTHING BUT THE CONSTRAINT MENTIONED ABOVE.

```
; 372    .TOC    "CONDITIONAL ASSEMBLY DEFINITIONS"
; 373
; 374    .DEFAULT/SIM=0          ;0=RUN ON REAL HARDWARE
; 375                            ;1=RUN UNDER SIMULATOR
; 376
; 377    .DEFAULT/FULL=1         ;0=INCLUDE ONLY BASIC INSTRUCTIONS
; 378                            ;1=INCLUDE FULL INSTRUCTION SET
; 379
; 380    .WIDTH/108              ;ONLY FIELDS BETWEEN BITS 0 AND 107 EVER
; 381                            ; GET LOADED INTO THE CRAM. OTHER FIELDS
; 382                            ; ARE USED FOR DEFAULTING PROCESS.
; 383
; 384    .REGION/0,1377/2000,3777/1400,1777
; 385                            ;TRY AND KEEP STUFF OUT OF DROM SPACE
; 386
```

```
; 387    .TOC    "2901 REGISTER USAGE"
; 388
; 389    ;       !=========================================================================!
; 390    ;0:     !                    MAG (ONES IN BITS 1-36, REST ZERO)                   !
; 391    ;       !-------------------------------------------------------------------------!
; 392    ;1:     !                 PC (ADDRESS OF CURRENT INSTRUCTION + 1)                 !
; 393    ;       !-------------------------------------------------------------------------!
; 394    ;2:     !                       HR (CURRENT INSTRUCTION)                          !
; 395    ;       !-------------------------------------------------------------------------!
; 396    ;3:     !                    AR (TEMP -- MEM OP AT INST START)                    !
; 397    ;       !-------------------------------------------------------------------------!
; 398    ;4:     !                ARX (TEMP -- LOW ORDER HALF OF DOUBLE PREC)              !
; 399    ;       !-------------------------------------------------------------------------!
; 400    ;5:     !                                BR (TEMP)                                !
; 401    ;       !-------------------------------------------------------------------------!
; 402    ;6:     !           BRX (TEMP -- LOW ORDER HALF OF DOUBLE PREC BR!BRX)            !
; 403    ;       !-------------------------------------------------------------------------!
; 404    ;7:     !                          ONE (THE CONSTANT 1)                           !
; 405    ;       !-------------------------------------------------------------------------!
; 406    ;10:    !                        EBR (EXEC BASE REGISTER)                         !
; 407    ;       !-------------------------------------------------------------------------!
; 408    ;11:    !                        UBR (USER BASE REGISTER)                         !
; 409    ;       !-------------------------------------------------------------------------!
; 410    ;12:    !           MASK (ONES IN BITS 0-35, ZERO IN -1, -2, 36 AND 37)           !
; 411    ;       !-------------------------------------------------------------------------!
; 412    ;13:    !         FLG (FLAG BITS)          !          PAGE FAIL CODE              !
; 413    ;       !-------------------------------------------------------------------------!
; 414    ;14:    !                  PI (PI SYSTEM STATUS REGISTER [RDPI])                  !
; 415    ;       !-------------------------------------------------------------------------!
; 416    ;15:    !                       XWD1 (1 IN EACH HALF WORD)                        !
; 417    ;       !-------------------------------------------------------------------------!
; 418    ;16:    !                              TO (TEMP)                                  !
; 419    ;       !-------------------------------------------------------------------------!
; 420    ;17:    !                              T1 (TEMP)                                  !
; 421    ;       !=========================================================================!
; 422
```

```
; 423           .TOC    "MICROCODE FIELDS -- LISTING FORMAT"
; 424
; 425   ;                                                                ; 3633  1561:
; 426   ;                                                                ; 3634  SUB:    [AR]_AC-[AR],
; 427   ;                                                                ; 3635          AD FLAGS, 3T,
; 428   ;       U 1561, 1500,2551,0303,0274,4463,7701,4200,0001,0001     ; 3636          EXIT
; 429   ;         [--]  [--] !!!! [][] !!![-][][-][]! !!!     [----]
; 430   ;           !     !    !!!! ! !  !!!!  ! !  ! ! !!!       !
; 431   ;           !     !    !!!! ! !  !!!!  ! !  ! ! !!!       +---- # (MAGIC NUMBER)
; 432   ;           !     !    !!!! ! !  !!!!  ! !  ! ! !!!
; 433   ;           !     !    !!!! ! !  !!!!  ! !  ! ! !!+------------ MULTI PREC, MULTI SHIFT, CALL (4S, 2S, 1S)
; 434   ;           !     !    !!!! ! !  !!!!  ! !  ! ! !!
; 435   ;           !     !    !!!! ! !  !!!!  ! !  ! ! !+-------------- FM WRITE, MEM, DIVIDE (4S, 2S, 1S)
; 436   ;           !     !    !!!! ! !  !!!!  ! !  ! ! !
; 437   ;           !     !    !!!! ! !  !!!!  ! !  ! ! +--------------- CRY38, LOAD SC, LOAD FE (4S, 2S, 1S)
; 438   ;           !     !    !!!! ! !  !!!!  ! !  ! !
; 439   ;           !     !    !!!! ! !  !!!!  ! !  ! +----------------- T
; 440   ;           !     !    !!!! ! !  !!!!  ! !  !
; 441   ;           !     !    !!!! ! !  !!!!  ! !  +------------------- SKIP
; 442   ;           !     !    !!!! ! !  !!!!  ! !
; 443   ;           !     !    !!!! ! !  !!!!  ! +--------------------- DISP
; 444   ;           !     !    !!!! ! !  !!!!  !
; 445   ;           !     !    !!!! ! !  !!!!  +----------------------- SPEC
; 446   ;           !     !    !!!! ! !  !!!!
; 447   ;           !     !    !!!! ! !  !!!+------------------------- CLOCKS & PARITY (CLKR, GENR, CHKR, CLKL, GENL, CHKL)
; 448   ;           !     !    !!!! ! !  !!!
; 449   ;           !     !    !!!! ! !  !!+------------------------- DBM
; 450   ;           !     !    !!!! ! !  !!
; 451   ;           !     !    !!!! ! !  !+------------------------- DBUS
; 452   ;           !     !    !!!! ! !  !
; 453   ;           !     !    !!!! ! !  +------------------------- RAM ADDRESS
; 454   ;           !     !    !!!! ! !
; 455   ;           !     !    !!!! ! +----------------------------- B
; 456   ;           !     !    !!!! !
; 457   ;           !     !    !!!! +------------------------------- A
; 458   ;           !     !    !!!!
; 459   ;           !     !    !!!+--------------------------------- DEST
; 460   ;           !     !    !!!
; 461   ;           !     !    !!+----------------------------------- RSRC
; 462   ;           !     !    !!
; 463   ;           !     !    !+-----------------------------------  LSRC   ]
; 464   ;           !     !    !                                      ]  - AD
; 465   ;           !     !   +------------------------------------- ALU    ]
; 466   ;           !     !
; 467   ;           !     +--------------------------------------------- J
; 468   ;           !
; 469   ;           LOCATION OF THIS MICRO WORD
; 470
```

```
; 471   .TOC    "MICROCODE FIELDS -- DATAPATH CHIP"
; 472
; 473   J/=<0:11>+                ;CRA1
; 474                             ;NEXT MICROCODE ADDRESS
; 475
; 476   ;ALU FUNCTIONS
; 477
; 478   ;NOTE: THE AD FIELD IS A 2 DIGIT FIELD. THE LEFT DIGIT IS THE
; 479   ; 2901 ALU FUNCTION. THE RIGHT DIGIT IS THE 2901 SRC CODE FOR
; 480   ; THE LEFT HALF. NORMALY THE RIGHT HALF SRC CODE IS THE SAME AS
; 481   ; THE LEFT HALF.
; 482   AD/=<12:17>D,44        ;DPE1 & DPE2
; 483           A+Q=00
; 484           A+B=01
; 485           0+Q=02
; 486           0+B=03
; 487           0+A=04
; 488           D+A=05
; 489           D+Q=06
; 490           0+D=07
; 491           Q-A-.25=10
; 492           B-A-.25=11
; 493           Q-.25=12
; 494           B-.25=13
; 495           A-.25=14
; 496           A-D-.25=15
; 497           Q-D-.25=16
; 498           -D-.25=17
; 499           A-Q-.25=20
; 500           A-B-.25=21
; 501           -Q-.25=22
; 502           -B-.25=23
; 503           -A-.25=24
; 504           D-A-.25=25
; 505           D-Q-.25=26
; 506           D-.25=27
; 507           A.OR.Q=30
; 508           A.OR.B=31
; 509           Q=32
; 510           B=33
; 511           A=34
; 512           D.OR.A=35
; 513           D.OR.Q=36
; 514           D=37
; 515           A.AND.Q=40
; 516           A.AND.B=41
; 517
```

```
; 518    ;MORE ALU FUNCTIONS
; 519
; 520            ZERO=42
; 521    ;       ZERO=43
; 522    ;       ZERO=44
; 523            D.AND.A=45
; 524            D.AND.Q=46
; 525    ;       ZERO=47
; 526            .NOT.A.AND.Q=50
; 527            .NOT.A.AND.B=51
; 528    ;       Q=52
; 529    ;       B=53
; 530    ;       A=54
; 531            .NOT.D.AND.A=55
; 532            .NOT.D.AND.Q=56
; 533    ;       ZERO=57
; 534            A.XOR.Q=60
; 535            A.XOR.B=61
; 536    ;       Q=62
; 537    ;       B=63
; 538    ;       A=64
; 539            D.XOR.A=65
; 540            D.XOR.Q=66
; 541    ;       D=67
; 542            A.EQV.Q=70
; 543            A.EQV.B=71
; 544            .NOT.Q=72
; 545            .NOT.B=73
; 546            .NOT.A=74
; 547            D.EQV.A=75
; 548            D.EQV.Q=76
; 549            .NOT.D=77
; 550
; 551    ;THIS FIELD IS THE RIGHTMOST 3 BITS OF THE
; 552    ; AD FIELD. IT IS USED ONLY TO DEFAULT THE RSRC
; 553    ; FIELD.
; 554    LSRC/=<15:17>          ;DPE1
; 555
; 556    ;THIS IS THE SOURCE FOR THE RIGHT HALF OF THE
; 557    ; DATA PATH. IT LETS US MAKE THE RIGHT AND LEFT
; 558    ; HALF WORDS DO SLIGHTLY DIFFERENT THINGS.
; 559    RSRC/=<18:20>F,LSRC      ;DPE2
; 560            AQ=0             ;A  Q
; 561            AB=1             ;A  B
; 562            0Q=2             ;0  Q
; 563            0B=3             ;0  B
; 564            0A=4             ;0  A
; 565            DA=5             ;D  A
; 566            DQ=6             ;D  Q
; 567            D0=7             ;D  0
; 568
```

```
; 569    ;DESTINATION CONTROL
; 570    ;SEE DPE1 AND DPE2 (2'S WEIGHT IS INVERTED ON DPE5)
; 571    DEST/=<21:23>D,3        ;DPE1 & DPE2
; 572            A=0               ;A REG IS CHIP OUTPUT, AD IS WRITTEN
; 573                              ; INTO REG FILE
; 574            AD=1              ;REG FILE GETS AD
; 575            Q_AD=2            ;REG FILE IS NOT LOADED
; 576            PASS=3            ;AD OUTPUT IS CHIP OUTPUT
; 577                              ; Q AND REG FILE LEFT ALONE
; 578            Q_Q*2=4           ;ALSO REG FILE GETS AD*2
; 579            AD*2=5            ;AND Q IS LEFT ALONE
; 580            Q_Q*.5=6          ;ALSO REG FILE GETS AD*.5
; 581            AD*.5=7           ;AND Q IS LEFT ALONE
; 582
; 583    ;       <24:25>           ;UNUSED
; 584
; 585    A/=<26:29>                ;DPE1 & DPE2
; 586            MAG=0
; 587            PC=1
; 588            HR=2
; 589            AR=3
; 590            ARX=4
; 591            BR=5
; 592            BRX=6
; 593            ONE=7
; 594            EBR=10
; 595            UBR=11
; 596            MASK=12
; 597            FLG=13
; 598            PI=14
; 599            XWD1=15
; 600            TO=16
; 601            T1=17
; 602
; 603    ;       <30:31>           ;UNUSED
; 604
; 605    B/=<32:35>D,0             ;DPE1 & DPE2
; 606            MAG=0
; 607            PC=1
; 608            HR=2
; 609            AR=3
; 610            ARX=4
; 611            BR=5
; 612            BRX=6
; 613            ONE=7
; 614            EBR=10
; 615            UBR=11
; 616            MASK=12
; 617            FLG=13
; 618            PI=14
; 619            XWD1=15
; 620            TO=16
; 621            T1=17
; 622
```

```
; 623    .TOC    "MICROCODE FIELDS -- RAM FILE ADDRESS AND D-BUS"
; 624
; 625    RAMADR/=<36:38>D,4      ;DPE6
; 626            AC#=0           ;AC NUMBER
; 627            AC*#=1          ;AC .FN. #
; 628            XR#=2           ;INDEX REGISTER
; 629            VMA=4           ;VIRTUAL MEMORY REFERENCE
; 630            RAM=6           ;VMA SUPPLIES 10-BIT RAM ADDRESS
; 631            #=7             ;ABSOLUTE RAM FILE REFERENCE
; 632
; 633    ;       <39:39>
; 634
; 635    ;LEFT HALF ON DPE3 AND RIGHT HALF ON DPE4
; 636    DBUS/=<40:41>D,1        ;DPE3 & DPE4
; 637            PC FLAGS=0      ;PC FLAGS IN LEFT HALF
; 638            PI NEW=0        ;NEW PI LEVEL IN BITS 19-21
; 639    ;       VMA=0           ;VMA IN BITS 27-35
; 640            DP=1            ;DATA PATH
; 641            RAM=2           ;CACHE, AC'S AND WORKSPACE
; 642            DBM=3           ;DBM MIXER
; 643
; 644    ;LEFT HALF ON DPM1 AND RIGHT HALF ON DPM2
; 645    DBM/=<42:44>D,7         ;DPM1 & DPM2
; 646            SCAD DIAG=0     ;(LH) SCAD DIAGNOSTIC
; 647            PF DISP=0       ;PAGE FAIL DISP IN BITS 18-21
; 648            APR FLAGS=0     ;APR FLAGS IN BITS 22-35
; 649            BYTES=1         ;5 COPIES OF SCAD 1-7
; 650            EXP=2           ;LH=EXPONENT, RH=TIME FRACTION
; 651            DP=3            ;DATA PATH
; 652            DP SWAP=4       ;DATA PATH SWAPPED
; 653            VMA=5           ;VMA FLAGS,,VMA
; 654            MEM=6           ;MEMORY BUFFER
; 655            #=7             ;NUMBER FIELD IN BOTH HALVES
; 656
```

```
; 657   .TOC    "MICROCODE FIELDS -- PARITY GENERATION & HALF WORD CONTROL"
; 658
; 659   AD PARITY OK/=<108>D,0  ;**NOT STORED IN CRAM**
; 660                           ;THIS BIT IS A 1 IF THE ALU IS DOING
; 661                                   ; SOMETHING WHICH DOES NOT INVALIDATE
; 662                                   ; PARITY. IT DOES NOT APPEAR IN THE
; 663                                   ; REAL MACHINE. WE JUST USE IT TO SET
; 664                                   ; THE DEFAULT FOR GENR & GENL
; 665
; 666   CLKL/=<45:45>D,1        ;DPE5
; 667                           ;CLOCK THE LEFT HALF OF THE MACHINE
; 668   GENL/=<46:46>F,AD PARITY OK ;DPE4 FROM CRM2 PARITY EN LEFT H
; 669                           ;STORE PARITY FOR 2901 LEFT
; 670   CHKL/=<47:47>           ;DPE4 FROM CRM2 PARITY CHK LEFT H
; 671                           ;CHECK LEFT HALF DBUS PARITY
; 672
; 673   CLKR/=<48:48>D,1        ;DPE5
; 674                           ;CLOCK THE RIGHT HALF OF THE MACHINE
; 675   GENR/=<49:49>F,AD PARITY OK ;DPE4 FROM CRM2 PARITY EN RIGHT H
; 676                           ;STORE PARITY FOR 2901 RIGHT
; 677   CHKR/=<50:50>           ;DPE4 FROM CRM2 PARITY CHK RIGHT H
; 678                           ;CHECK RIGHT HALF DBUS PARITY
; 679
```

```
; 680    .TOC    "MICROCODE FIELDS -- SPEC"
; 681
; 682
; 683    ;
; 684    ;THE FOLLOWING SPECIAL FUNCTION ARE DECODED ON DPE1, DPE5, AND DPMA:
; 685    ;       !=============================================================================!
; 686    ;       !S!     EFFECT      !    CRA6 SPEC    !    CRA6 SPEC    !    CRA6 SPEC    !
; 687    ;       !P!    ON SHIFT     !      EN 40      !      EN 20      !      EN 10      !
; 688    ;       !E!      PATHS      !  E102 ON DPE5   !  E101 ON DPE5   !  E410 ON DPMA   !
; 689    ;       !C!   (SEE DPE1)    !                 !  E411 ON DPMA   !  E113 ON CRA2   !
; 690    ;       !=============================================================================!
; 691    ;       !0!      NORMAL     !   CRY 18 INH.   !    PREVIOUS     !        #        !
; 692    ;       !-----------------------------------------------------------------------------!
; 693    ;       !1!       ZERO      !     IR LOAD     !     XR LOAD     !   CLR 1 MSEC   !
; 694    ;       !-----------------------------------------------------------------------------!
; 695    ;       !2!       ONES      !     <SPARE>     !     <SPARE>     !  CLR IO LATCH  !
; 696    ;       !-----------------------------------------------------------------------------!
; 697    ;       !3!       ROT       !     PI LOAD     !    APR FLAGS    !  CLR IO BUSY   !
; 698    ;       !-----------------------------------------------------------------------------!
; 699    ;       !4!       ASHC      !    ASH TEST     !    SET SWEEP    !   PAGE WRITE   !
; 700    ;       !-----------------------------------------------------------------------------!
; 701    ;       !5!       LSHC      !    EXP TEST     !     APR EN      !     NICOND     !
; 702    ;       !-----------------------------------------------------------------------------!
; 703    ;       !6!       DIV       !    PC FLAGS     !    PXCT OFF     !     PXCT EN    !
; 704    ;       !-----------------------------------------------------------------------------!
; 705    ;       !7!       ROTC      !  AC BLOCKS EN   !     MEM CLR     !    MEM WAIT    !
; 706    ;       !=============================================================================!
; 707    ; THE DPM BOARD USES THE SPEC FIELD TO CONTROL THE
; 708    ;  DBM MIXER, AS FOLLOWS:
; 709    ;
; 710    ;       !=====================================!
; 711    ;       !  S  !                              !
; 712    ;       !  P  !        ACTION WHEN DBM       !
; 713    ;       !  E  !          SELECTS DP          !
; 714    ;       !  C  ! GET DP BITS  !  GET SCAD 1-7  !
; 715    ;       !=====================================!
; 716    ;       !  0  !      ALL      !      NONE      !
; 717    ;       !-------------------------------------!
; 718    ;       !  1  !     7-35      !      0-6       !
; 719    ;       !-------------------------------------!
; 720    ;       !  2  !0-6 AND 14-35 !      7-13      !
; 721    ;       !-------------------------------------!
; 722    ;       !  3  !0-13 AND 21-35!     14-20      !
; 723    ;       !-------------------------------------!
; 724    ;       !  4  !0-20 AND 28-35!     21-27      !
; 725    ;       !-------------------------------------!
; 726    ;       !  5  ! 0-27 AND 35  !     28-34      !
; 727    ;       !-------------------------------------!
; 728    ;       !  6  !         SAME AS ZERO          !
; 729    ;       !-------------------------------------!
; 730    ;       !  7  !         SAME AS ZERO          !
; 731    ;       !=====================================!
; 732
```

```
; 733   ;THE SPEC FIELD IS DEFINED AS A 6-BIT FIELD. THE TOP 3 BITS
; 734   ; ARE SPEC SEL A, SPEC SEL B, AND SPEC SEL C. THE LOW 3 BITS ARE
; 735   ; THE SELECT CODE.
; 736
; 737   SPEC/=<51:56>D,O          ;DPE1 & DPE5 & DPM1 & DPMA
; 738           #=10              ;DECODE # BITS
; 739           CLRCLK=11         ;CLEAR 1MS NICOND FLAG
; 740           CLR IO LATCH=12   ;CLEAR IO LATCH
; 741           CLR IO BUSY=13    ;CLEAR IO BUSY
; 742           LDPAGE=14         ;WRITE PAGE TABLE
; 743           NICOND=15         ;DOING NICOND DISPATCH
; 744           LDPXCT=16         ;LOAD PXCT FLAGS
; 745           WAIT=17           ;MEM WAIT
; 746           PREV=20           ;FORCE PREVIOUS CONTEXT
; 747           LOADXR=21         ;LOAD XR #, USES PXCT FIELD TO SELECT
; 748                             ; CORRECT AC BLOCK
; 749           APR FLAGS=23      ;LOAD APR FLAGS
; 750           CLRCSH=24         ;CLEAR CACHE
; 751           APR EN=25         ;SET APR ENABLES
; 752           MEMCLR=27         ;CLEAR PAGE FAULT CONDITION
; 753           SWEEP=34          ;SET SWEEP
; 754           PXCT OFF=36       ;TURN OFF THE EFFECT OF PXCT
; 755           INHCRY18=40       ;INHIBIT CARRY INTO LEFT HALF
; 756           LOADIR=41         ;LOAD THE IR
; 757           LDPI=43           ;LOAD PI SYSTEM
; 758           ASHOV=44          ;TEST RESULT OF ASH
; 759           EXPTST=45         ;TEST RESULT OF FLOATING POINT
; 760           FLAGS=46          ;CHANGE PC FLAGS
; 761           LDACBLK=47        ;LOAD AC BLOCK NUMBERS
; 762           LDINST=61         ;LOAD INSTRUCTION
; 763
; 764   ;THE SPEC FIELD IS REDEFINED WHEN USED FOR BYTE MODE STUFF
; 765   BYTE/=<54:56>             ;DPM1 (SPEC SEL)
; 766           BYTE1=1
; 767           BYTE2=2
; 768           BYTE3=3
; 769           BYTE4=4
; 770           BYTE5=5
; 771
; 772   ;THE SPEC FIELD IS REDEFINED WHEN USED TO CONTROL SHIFT PATHS
; 773   SHSTYLE/=<54:56>          ;DPE1 (SPEC SEL)
; 774           NORM=0            ;2 40-BIT REGISTERS
; 775           ZERO=1            ;SHIFT ZERO INTO 36 BITS (ASH TOP 2901)
; 776           ONES=2            ;SHIFT IN ONES
; 777           ROT=3             ;ROTATE
; 778           ASHC=4            ;ASHC
; 779           LSHC=5            ;LSHC
; 780           DIV=6             ;SPECIAL DIVIDE
; 781           ROTC=7            ;ROTATE DOUBLE
; 782
```

```
; 783   .TOC     "MICROCODE FIELDS -- DISPATCH"
; 784   ;         !=========================================================!
; 785   ;         ! D !       CRA1       !       CRA1       !       DPEA       !
; 786   ;         ! I !       DISP       !       DISP       !       DISP       !
; 787   ;         ! S !         10       !         20       !         40       !
; 788   ;         ! P !                  !                  !                  !
; 789   ;         !=========================================================!
; 790   ;         ! 0 !    DIAG ADR      !    DIAG ADR      !         0        !
; 791   ;         !---------------------------------------------------------!
; 792   ;         ! 1 !     RETURN       !     RETURN       !    DP 18-21      !
; 793   ;         !---------------------------------------------------------!
; 794   ;         ! 2 !    MULTIPLY      !         J        !         J        !
; 795   ;         !---------------------------------------------------------!
; 796   ;         ! 3 !   PAGE FAIL      !      AREAD       !      AREAD       !
; 797   ;         !---------------------------------------------------------!
; 798   ;         ! 4 !     NICOND       !   NOT USABLE     !       NORM       !
; 799   ;         !---------------------------------------------------------!
; 800   ;         ! 5 !      BYTE        !   NOT USABLE     !    DP 32-35      !
; 801   ;         !---------------------------------------------------------!
; 802   ;         ! 6 !    EA MODE       !   NOT USABLE     !      DROM A      !
; 803   ;         !---------------------------------------------------------!
; 804   ;         ! 7 !      SCAD        !   NOT USABLE     !      DROM B      !
; 805   ;         !=========================================================!
; 806   ;NOTE:   DISP EN 40 & DISP EN 10 ONLY CONTROL THE LOW 4 BITS OF THE
; 807   ;         JUMP ADDRESS. DISP EN 20 ONLY CONTROLS THE HI 7 BITS. TO DO
; 808   ;         SOMETHING TO ALL 11 BITS BOTH 20 & 40 OR 20 & 10 MUST BE ENABLED.
; 809
; 810   DISP/=<57:62>D,70        ;CRA1 & DPEA
; 811           CONSOLE=00       ;CONSOLE DISPATCH
; 812           DROM=12          ;DROM
; 813           AREAD=13         ;AREAD
; 814           DP LEFT=31       ;DP 18-21
; 815           NORM=34          ;NORMALIZE
; 816           DP=35            ;DP 32-35
; 817           ADISP=36         ;DROM A FIELD
; 818           BDISP=37         ;DROM B FIELD
; 819           RETURN=41        ;RETURN
; 820           MUL=62           ;MULTIPLY
; 821           PAGE FAIL=63     ;PAGE FAIL
; 822           NICOND=64        ;NEXT INSTRUCTION DISPATCH
; 823           BYTE=65          ;BYTE SIZE AND POSITION
; 824           EAMODE=66        ;EFFECTIVE ADDRESS MODE
; 825           SCADO=67         ;J!2 IF SCAD BIT 0 = 1
; 826
```

```
; 827   .TOC    "MICROCODE FIELDS -- SKIP"
; 828   ;       !=========================================================!
; 829   ;       ! S !      CRA2       !      DPEA       !      DPEA       !
; 830   ;       ! K !      SKIP       !      SKIP       !      SKIP       !
; 831   ;       ! I !       10        !       20        !       40        !
; 832   ;       ! P !                 !                 !                 !
; 833   ;       !=========================================================!
; 834   ;       ! 0 !       0         !       0         !       0         !
; 835   ;       !---------------------------------------------------------!
; 836   ;       ! 1 !   TRAP CYCLE    !     CRY 02      !    CARRY OUT    !
; 837   ;       !---------------------------------------------------------!
; 838   ;       ! 2 !      AD=0       !    ADL SIGN     !      ADL=0      !
; 839   ;       !---------------------------------------------------------!
; 840   ;       ! 3 !    SC SIGN      !    ADR SIGN     !      ADR=0      !
; 841   ;       !---------------------------------------------------------!
; 842   ;       ! 4 !    EXECUTE      !    USER IOT     !      -USER      !
; 843   ;       !---------------------------------------------------------!
; 844   ;       ! 5 !  -BUS IO BUSY   !   JFCL SKIP     !    FPD FLAG     !
; 845   ;       !---------------------------------------------------------!
; 846   ;       ! 6 !   -CONTINUE     !     CRY 01      !  AC # IS ZERO   !
; 847   ;       !---------------------------------------------------------!
; 848   ;       ! 7 !    -1 MSEC      !      TXXX       !  INTERRUPT REQ  !
; 849   ;       !=========================================================!
; 850
; 851   SKIP/=<63:68>D,70        ;CRA2 & DPEA
; 852           IOLGL=04         ;(.NOT.USER)!(USER IOT)!(CONSOLE EXECUTE MODE)
; 853           LLE=12           ;AD LEFT .LE. 0
; 854           CRY0=31          ;AD CRY -2
; 855           ADLEQ0=32        ;ADDER LEFT = 0
; 856           ADREQ0=33        ;ADDER RIGHT = 0
; 857           KERNEL=34        ;.NOT. USER
; 858           FPD=35           ;FIRST PART DONE
; 859           AC0=36           ;AC NUMBER IS ZERO
; 860           INT=37           ;INTERRUPT REQUEST
; 861           LE=42            ;(AD SIGN)!(AD.EQ.0)
; 862           CRY2=51          ;AD CRY 02
; 863           DP0=52           ;AD SIGN
; 864           DP18=53          ;AD BIT 18
; 865           IOT=54           ;USER IOT
; 866           JFCL=55          ;JFCL SKIP
; 867           CRY1=56          ;AD CRY 1
; 868           TXXX=57          ;TEST INSTRUCTION SHOULD SKIP
; 869           TRAP CYCLE=61    ;THIS INSTRUCTION IS THE RESULT OF A
; 870                            ; TRAP 1, 2, OR 3
; 871           ADEQ0=62         ;AD.EQ.0
; 872           SC=63            ;SC SIGN BIT
; 873           EXECUTE=64       ;CONSOLE EXECUTE MODE
; 874           -IO BUSY=65      ;.NOT. I/O LATCH
; 875           -CONTINUE=66     ;.NOT. CONTINUE
; 876           -1 MS=67         ;.NOT. 1 MS. TIMER
; 877
```

```
; 878    .TOC    "MICROCODE FIELDS -- TIME CONTROL"
; 879
; 880    DT/=<109:111>D,O        ;**NOT STORED IN CRAM**
; 881                            ;DEFAULT TIME FIELD (USED IN MACROS)
; 882                            ; CAN BE OVERRIDDEN IN MACRO CALL
; 883            2T=0
; 884            3T=1
; 885            4T=2
; 886            5T=3
; 887
; 888
; 889    T/=<69:71>F,DT          ;CSL5 (E601)
; 890                            ;CLOCK TICKS MINUS TWO REQUIRED TO
; 891                            ; DO A MICRO INSTRUCTION
; 892            2T=0            ;TWO TICKS
; 893            3T=1            ;THREE TICKS
; 894            4T=2            ;FOUR TICKS
; 895            5T=3            ;FIVE TICKS
; 896
; 897
```

```
; 898   .TOC    "MICROCODE FIELDS -- RANDOM CONTROL BITS"
; 899
; 900   CRY38/=<72>              ;DPE5
; 901                            ;INJECT A CARRY INTO THE 2901 ADDER
; 902   LOADSC/=<73>             ;DPM4
; 903                            ;LOAD THE STEP COUNTER FROM THE SCAD
; 904   LOADFE/=<74>             ;DPM4
; 905                            ;LOAD THE FE REGISTER FROM THE SCAD
; 906   FMWRITE/=<75>            ;DPE5 (E302)
; 907                            ;WRITE THE RAM FILE.
; 908   MEM/=<76>                ;DPM5 (E612) & DPE5 (E205)
; 909                            ;START (OR COMPLETE) A MEMORY OR I/O CYCLE UNDER
; 910                            ; CONTROL OF THE NUMBER FIELD.
; 911   DIVIDE/=<77>             ;DPE5
; 912                            ;THIS MICROINSTRUCTION IS DOING A DIVIDE
; 913   MULTI PREC/=<78>         ;DPE5
; 914                            ;MULTIPRECISION STEP IN DIVIDE, DFAD, DFSB
; 915   MULTI SHIFT/=<79>        ;CSL5 (HAS NOTHING TO DO WITH DPE5 MULTI SHIFT)
; 916                            ;FAST SHIFT
; 917   CALL/=<80>               ;CRA2 (STACK IS ON CRA3)
; 918                            ;THIS IS A CALL
; 919
```

```
; 920   .TOC    "MICROCODE FIELDS -- NUMBER FIELD"
; 921
; 922   ;HERE IS THE GENERAL FIELD
; 923   #/=<90:107>              ;MANY PLACES
; 924
; 925   ;# REDEFINED WHEN USED AS SCAD CONTROL:
; 926   SCAD/=<90:92>             ;DPM3
; 927           A*2=0
; 928           A.OR.B=1
; 929           A-B-1=2
; 930           A-B=3
; 931           A+B=4
; 932           A.AND.B=5
; 933           A-1=6
; 934           A=7
; 935   SCADA/=<93:95>            ;DPM3
; 936           SC=0
; 937           S#=1
; 938           PTR44=2 ;44 AND BIT 6 (SEE DPM3)
; 939           BYTE1=3
; 940           BYTE2=4
; 941           BYTE3=5
; 942           BYTE4=6
; 943           BYTE5=7
; 944   SCADB/=<96:97>            ;DPM3
; 945           FE=0
; 946           EXP=1
; 947           SHIFT=2
; 948           SIZE=3
; 949   S#/=<98:107>              ;DPM3
; 950
; 951   ;# REDEFINED WHEN USED AS STATE REGISTER CONTROL:
; 952   STATE/=<90:107>           ;NOT USED BY HARDWARE
; 953           SIMPLE=0          ;SIMPLE INSTRUCTIONS
; 954           BLT=1             ;BLT IN PROGRESS
; 955           MAP=400002        ;MAP IN PROGRESS
; 956           SRC=3             ;MOVE STRING SOURCE IN PROGRESS
; 957           DST=4             ;MOVE STRING FILL IN PROGRESS
; 958           SRC+DST=5         ;MOVE STRING DEST IN PROGRESS
; 959           DSTF=6            ;FILLING DEST
; 960           CVTDB=7           ;CONVERT DEC TO BIN
; 961           COMP-DST=10       ;COMPARE DEST
; 962           EDIT-SRC=11       ;EDIT SOURCE
; 963           EDIT-DST=12       ;EDIT DEST
; 964           EDIT-S+D=13       ;BOTH SRC AND DST POINTERS
; 965
```

```
; 966    ;# REDEFINED WHEN USED AS WORSPACE ADDRESS
; 967
; 968    WORK/=<98:107>          ;DPE6
; 969            BADWO=160       ;AC BLK 7 WORD 0 (BAD DATA FROM MEMORY)
; 970            BADW1=161       ;AC BLK 7 WORD 1 (BAD DATA FROM MEMORY)
; 971            MUL=200         ;TEMP FOR MULTIPLY
; 972            DIV=201         ;TEMP FOR DIVIDE
; 973            SV.VMA=210      ;SAVE VMA
; 974            SV.AR=211       ;SAVE AR
; 975            SV.ARX=212      ;SAVE ARX
; 976            SV.BR=213       ;SAVE BR
; 977            SV.BRX=214      ;SAVE BRX
; 978            SBR=215         ;SPT BASE REGISTER
; 979            CBR=216         ;CST BASE ADDRESS
; 980            CSTM=217        ;CST MASK
; 981            PUR=220         ;PROCESS USE REGISTER
; 982            ADJP=221        ;"P" FOR ADJBP
; 983            ADJS=222        ;"S" FOR ADJBP
; 984            ADJPTR=223      ;BYTE POINTER FOR ADJBP
; 985            ADJQ1=224       ;TEMP FOR ADJBP
; 986            ADJR2=225       ;TEMP FOR ADJBP
; 987            ADJBPW=226      ;(BYTES/WORD) FOR ADJBP
; 988            HSBADR=227      ;ADDRESS OF HALT STATUS BLOCK
; 989            APR=230         ;APR ENABLES
; 990    ;THE FOLLOWING WORDS ARE USED BY EXTEND INSTRUCTION
; 991            EO=240          ;ORIGINAL EFFECTIVE ADDRESS
; 992            E1=241          ;EFFECTIVE ADDRESS OF WORD AT EO
; 993            SLEN=242        ;SOURCE LENGTH
; 994            MSK=243         ;BYTE MASK
; 995            FILL=244        ;FILL BYTE
; 996            CMS=245         ;SRC BYTE IN STRING COMPARE
; 997            FSIG=246        ;PLACE TO SAVE ARX WHILE STORING
; 998                            ; THE FLOAT CHAR
; 999            BDH=247         ;BINARY BEING CONVERTED TO
; 1000           BDL=250         ; DECIMAL
; 1001
; 1002   ;TIMER STUFF
; 1003           TIMEO=300       ;HIGH ORDER 36 BITS OF TIME
; 1004           TIME1=301       ;LOW ORDER 36 BITS OF TIME
; 1005           PERIOD=302      ;INTERRUPT PERIOD
; 1006           TTG=303         ;TIME TO GO TO NEXT INTERRUPT
; 1007
```

```
; 1008  ;DDIV STUFF
; 1009          AC0=314
; 1010          AC1=315
; 1011          AC2=316
; 1012          AC3=317
; 1013          DDIV SGN=320
; 1014          DVSOR H=321
; 1015          DVSOR L=322
; 1016  ;POWERS OF TEN
; 1017          DECLO=344       ;LOW WORD
; 1018          DECHI=373       ;HIGH WORD
; 1019
; 1020          YSAVE=422       ;Y OF LAST INDIRECT POINTER
; 1021          PTA.E=423       ;ADDRESS OF EXEC PAGE MAP (NOT PROCESS TABLE)
; 1022          PTA.U=424       ;ADDRESS OF USER PAGE MAP
; 1023          TRAPPC=425      ;SAVED PC FROM TRAP CYCLE
; 1024          SV.AR1=426      ;ANOTHER PLACE TO SAVE AR
; 1025
```

```
; 1026  ;# REDEFINED WHEN USED AS PC FLAG CONTROL (ALL ON DPE9)
; 1027
; 1028  SETOV/=<90>             ;DPE9
; 1029                          ;SET ARITHMETIC OVERFLOW
; 1030  SETFOV/=<91>            ;SET FLOATING OVERFLOW
; 1031  SETNDV/=<92>            ;SET NO DIVIDE
; 1032
; 1033  ;-------------------------------------------------------------------
; 1034
; 1035  CLRFPD/=<93>            ;CLEAR FIRST PART DONE
; 1036  SETFPD/=<94>            ;SET FIRST PART DONE
; 1037  HOLD USER/=<95>         ;WHEN THIS BIT IS SET IT:
; 1038                          ; 1. PREVENTS SETTING USER IOT IN USER MODE
; 1039                          ; 2. PREVENTS CLEARING USER IN USER MODE
; 1040
; 1041  ;-------------------------------------------------------------------
; 1042
; 1043  ;       <96>            ;SPARE
; 1044  TRAP2/=<97>             ;SET TRAP 2
; 1045  TRAP1/=<98>             ;SET TRAP 1
; 1046
; 1047  ;-------------------------------------------------------------------
; 1048
; 1049  LD PCU/=<99>            ;LOAD PCU FROM USER
; 1050  ;       <100>           ;SPARE
; 1051  ;       <101>           ;SPARE
; 1052
; 1053  ;-------------------------------------------------------------------
; 1054
; 1055  ;       <102>           ;SPARE
; 1056  ;       <103>           ;SPARE
; 1057  JFCLFLG/=<104>          ;DO A JFCL INSTRUCTION
; 1058
; 1059  ;-------------------------------------------------------------------
; 1060
; 1061  LD FLAGS/=<105>         ;LOAD FLAGS FROM DP
; 1062  ;       <106>
; 1063  ADFLGS/=<107>           ;UPDATE CARRY FLAGS
; 1064
```

```
; 1065  ;# REDEFINED WHEN USED AS MEMORY CYCLE CONTROL
; 1066
; 1067  FORCE USER/=<90>          ;FORCE USER MODE REFERENCE
; 1068  FORCE EXEC/=<91>          ;FORCE EXEC MODE REFERENCE
; 1069                           ; (DOES NOT WORK UNDER PXCT)
; 1070  FETCH/=<92>               ;THIS IS AN INSTRUCTION FETCH
; 1071
; 1072  ;-----------------------------------------------------------------------
; 1073
; 1074  READ CYCLE/=<93>          ;SELECT A READ CYCLE
; 1075  WRITE TEST/=<94>          ;PAGE FAILE IF NOT WRITTEN
; 1076  WRITE CYCLE/=<95>         ;SELECT A MEMORY WRITE CYCLE
; 1077
; 1078  ;-----------------------------------------------------------------------
; 1079
; 1080  ;       <96>              ;SPARE BIT
; 1081  DONT CACHE/=<97>          ;DO NOT LOOK IN CACHE
; 1082  PHYSICAL/=<98>            ;DO NOT INVOKE PAGING HARDWARE
; 1083
; 1084  ;-----------------------------------------------------------------------
; 1085
; 1086  PXCT/=<99:101>            ;WHICH PXCT BITS TO LOOK AT
; 1087          CURRENT=0
; 1088          E1=1
; 1089          D1=3
; 1090          BIS-SRC-EA=4
; 1091          E2=5
; 1092          BIS-DST-EA=6
; 1093          D2=7
; 1094
; 1095  ;-----------------------------------------------------------------------
; 1096
; 1097  AREAD/=<102>               ;LET DROM SELECT SYSLE TYPE AND VMA LOAD
; 1098  DP FUNC/=<103>             ;IGNORE # BITS 0-11 AND USE DP 0-13 INSTEAD
; 1099                           ; DP9 MEANS "FORCE PREVIOUS"
; 1100  LDVMA/=<104>               ;LOAD THE VMA
; 1101
; 1102  ;-----------------------------------------------------------------------
; 1103
; 1104  EXT ADR/=<105>            ;PUT VMA BITS 14-17 ONTO BUS
; 1105  WAIT/=<106>               ;START A MEMORY OR I/O CYCLE
; 1106  BWRITE/=<107>             ;START A MEMORY CYCLE IF DROM ASKS FOR IT
; 1107
```

```
; 1108  ;THESE BITS ARE USED ONLY TO SETUP DP FOR A DP FUNCTION
; 1109
; 1110  ;       <99>            ;PREVIOUS
; 1111  IO CYCLE/=<100>         ;THIS IS AN I/O CYCLE
; 1112  WRU CYCLE/=<101>        ;WHO ARE YOU CYCLE
; 1113
; 1114  ;-------------------------------------------------------------------
; 1115
; 1116  VECTOR CYCLE/=<102>     ;READ INTERRUPT VECTOR
; 1117  IO BYTE/=<103>          ;BYTE CYCLE
; 1118  ;       <104>
; 1119
```

```
; 1120  ;# REDEFINED WHEN USED AS PI RIGHT BITS
; 1121  PI.ZER/=<90:92>          ;ZEROS
; 1122  PI.IP1/=<93>             ;PI 1 IN PROG
; 1123  PI.IP2/=<94>
; 1124  PI.IP3/=<95>
; 1125  PI.IP4/=<96>
; 1126  PI.IP5/=<97>
; 1127  PI.IP6/=<98>
; 1128  PI.IP7/=<99>
; 1129  PI.ON/=<100>             ;SYSTEM IS ON
; 1130  PI.CO1/=<101>            ;CHAN 1 IS ON
; 1131  PI.CO2/=<102>
; 1132  I.CO3/=<103>
; 1133  I.CO4/=<104>
; 1134  I.CO5/=<105>
; 1135  I.CO6/=<106>
; 1136  I.CO7/=<107>
; 1137
; 1138  ;# REDEFINED WHEN USED AS WRPI DATA
; 1139  PI.MBZ/=<90:93>          ;MUST BE ZERO
; 1140  PI.DIR/=<94>             ;DROP INTERRUPT REQUESTS
; 1141  PI.CLR/=<95>             ;CLEAR SYSTEM
; 1142  PI.REQ/=<96>             ;REQUEST INTERRUPT
; 1143  PI.TCN/=<97>             ;TURN CHANNEL ON
; 1144  PI.TCF/=<98>             ;TURN CHANNEL OFF
; 1145  PI.TSF/=<99>             ;TURN SYSTEM OFF
; 1146  PI.TSN/=<100>            ;TURN SYSTEM ON
; 1147  PI.SC1/=<101>            ;SELECT CHANNEL 1
; 1148  PI.SC2/=<102>
; 1149  PI.SC3/=<103>
; 1150  PI.SC4/=<104>
; 1151  PI.SC5/=<105>
; 1152  PI.SC6/=<106>
; 1153  PI.SC7/=<107>
; 1154
```

```
;# REDEFINED WHEN USED AS AC CONTROL

;THIS FIELD CONTROLS THE INPUT TO A 74LS181 ON DPE6. THE NUMBER
; FIELD HAS THIS FORMAT IN <98:107>:
;
;       !-----!-----!-----!-----!-----!-----!-----!-----!-----!-----!
;       !CARRY! S8  !  S4 ! S2  !  S1 ! MODE! B8  ! B4  !  B2 ! B1  !
;       !  IN !       FUNCTION       !     !      DATA INPUTS      !
;       !-----!----------------------!-----!----------------------!
;

ACALU/=<98:103>
        B=25
        AC+N=62
ACN/=<104:107>
                        ;AC NAMES FOR STRING INSTRUCTIONS
        SRCLEN=0        ;SOURCE LENGTH
        SRCP=1          ;SOURCE POINTER
        DLEN=3          ;DEST LENGTH
        DSTP=4          ;DEST POINTER
        MARK=3          ;POINTER TO MARK
        BINO=3          ;HIGH WORD OF BINARY
        BIN1=4          ;LOW WORD OF BINARY

;# FIELD REDEFINED WHEN USE AS APRID DATA
MICROCODE OPTIONS/=<90:98>
        OPT=0
MICROCODE VERSION/=<99:107>
        UCV=123
```

```
; 1187  ;# FIELD REDEFINED WHEN USED AS A HALT CODE
; 1188
; 1189  HALT/=<90:107>
; 1190                          ;CODES 0 TO 77 ARE "NORMAL" HALTS
; 1191          POWER=0         ;POWER UP
; 1192          HALT=1          ;HALT INSTRUCTION
; 1193          CSL=2           ;CONSOLE HALT
; 1194                          ;CODES 100 TO 777 ARE SOFTWARE ERRORS
; 1195          IOPF=100        ;I/O PAGE FAIL
; 1196          ILLII=101       ;ILLEGAL INTERRUPT INSTRUCTION
; 1197          ILLINT=102      ;BAD POINTER TO UNIBUS INTERRUPT VECTOR
; 1198                          ;CODES 1000 TO 1777 ARE HARDWARE ERRORS
; 1199          BW14=1000       ;ILLEGAL BWRITE FUNCTION (BAD DROM)
; 1200          NICOND 5=1004   ;ILLEGAL NICOND DISPATCH
; 1201          MULERR=1005     ;VALUE COMPUTED FOR 10**21 WAS WRONG
;;1202  .IFNOT/FULL
;;1203          PAGEF=1777      ;PAGE FAIL IN SMALL MICROCODE
; 1204  .ENDIF/FULL
; 1205
; 1206
; 1207
; 1208  ;# FIELD REDEFINED WHEN USED AS FLG BITS
; 1209
; 1210  FLG.W/=<94>             ;W BIT FROM PAGE MAP
; 1211  FLG.PI/=<95>            ;PI CYCLE
; 1212  FLG.C/=<96>             ;CACHE BIT FROM PAGE MAP
; 1213  FLG.SN/=<97>            ;SPECIAL NEGATE IN FDV & DFDV
; 1214
; 1215  ;RIGHT HALF OF FLG USED TO RECOVER FROM PAGE FAILS
; 1216
```

```
; 1217  .TOC    "DISPATCH ROM DEFINITIONS"
; 1218
; 1219  ;ALL ON DPEA
; 1220
; 1221          .DCODE
; 1222  A/=<2:5>                ;OPERAND FETCH MODE
; 1223          READ=0          ;READ
; 1224          WRITE=1         ;WRITE
; 1225          DREAD=2         ;DOUBLE READ
; 1226          DBLAC=3         ;DOUBLE AC
; 1227          SHIFT=4         ;SIMPLE SHIFT
; 1228          DSHIFT=5        ;DOUBLE SHIFT
; 1229          FPI=6           ;FLOATING POINT IMMEDIATE
; 1230          FP=7            ;FLOATING POINT
; 1231          RD-PF=10        ;READ, THEN START PREFETCH
; 1232          DFP=11          ;DOUBLE FLOATING POINT
; 1233          IOT=12          ;CHECK FOR IO LEGAL THEN SAME AS I
; 1234
; 1235  B/=<8:11>                ;STORE RESULTS AS
; 1236          SELF=4          ;SELF
; 1237          DBLAC=5         ;DOUBLE AC
; 1238          DBLB=6          ;DOUBLE BOTH
; 1239          AC=15           ;AC
; 1240          MEM=16          ;MEMORY
; 1241          BOTH=17         ;BOTH
; 1242
; 1243  ;B-FIELD WHEN USED IN FLOATING POINT OPERATIONS
; 1244  ROUND/=<8>               ;ROUND THE RESULT
; 1245  MODE/=<9>                ;SEPARATE ADD/SUB & MUL/DIV ETC.
; 1246  FL-B/=<10:11>            ;STORE RESULTS AS
; 1247          AC=1            ;AC
; 1248          MEM=2           ;MEMORY
; 1249          BOTH=3          ;BOTH
; 1250
; 1251  J/=<12:23>               ;DISPATCH ADDRESS (MUST BE 1400 TO 1777)
; 1252
; 1253  ACDISP/=<24>             ;DISPATCH ON AC FIELD
; 1254  I/=<25>                  ;IMMEDIATE DISPATCH. DISP/AREAD DOES A DISP/DROM
; 1255                           ; IF THIS BIT IS SET.
; 1256  READ/=<26>               ;START A READ AT AREAD
; 1257  TEST/=<27>               ;START A WRITE TEST  AT AREAD
; 1258  COND FUNC/=<28>          ;START A MEMORY CYCLE ON BWRITE
; 1259  VMA/=<29>D,1             ;LOAD THE VMA ON AREAD
; 1260  WRITE/=<30>              ;START A WRITE ON AREAD
; 1261          .UCODE
; 1262
```

```
; 1263  .TOC    "HOW TO READ MACROS"
; 1264  ;
; 1265  ;               1.0  REGISTER TRANSFER MACROS
; 1266  ;
; 1267  ;               MOST MACROS USED IN THE KS10 ARE USED TO OPERATE ON DATA IN  (OR  FROM/TO)  2901
; 1268  ;               REGISTERS.   THE  NAMES  OF  THE  2901  REGISTERS  ARE  MACRO PARAMETERS AND ARE
; 1269  ;               ENCLOSED IN [].  A TYPICAL MACRO IS:
; 1270  ;
; 1271  ;                       [AR]_[AR]+[BR]
; 1272  ;
; 1273  ;               THE SYMBOL _ IS PRONOUNCED "GETS".  THE ABOVE MACRO WOULD BE READ "THE  AR  GETS
; 1274  ;               THE AR PLUS THE BR".
; 1275  ;
; 1276  ;               IF A MACRO DOES NOT HAVE A _ IN IT, THERE IS NO RESULT STORED.  THUS,  [AR]-[BR]
; 1277  ;               JUST COMPARES THE AR AND THE BR AND ALLOWS FOR SKIPS ON THE VARIOUS ALU BITS.
; 1278  ;
; 1279  ;
; 1280  ;
; 1281  ;               1.1  SPECIAL SYMBOLS
; 1282  ;
; 1283  ;               THERE ARE A BUNCH OF SYMBOLS USED IN THE MACROS WHICH ARE  NOT  2901  REGISTERS.
; 1284  ;               THEY ARE DEFINED HERE:
; 1285  ;
; 1286  ;                    1.  AC -- THE AC SELECTED BY THE CURRENT INSTRUCTION.  SEE DPEA
; 1287  ;
; 1288  ;                    2.  AC[] -- AC+N.  AC[1] IS AC+1, AC[2] IS AC+2, ETC.
; 1289  ;
; 1290  ;                    3.  APR -- THE APR FLAGS FROM DPMA
; 1291  ;
; 1292  ;                    4.  EA -- THE EFFECTIVE ADDRESS.  THAT IS, 0  IN  THE  LEFT  HALF  AND  THE
; 1293  ;                        CONTENTS OF THE HR IN THE RIGHT HALF.
; 1294  ;
; 1295  ;                    5.  EXP -- THE F.P.  EXPONENT  FROM  THE  SCAD.   [AR]_EXP  WILL  TAKE  THE
; 1296  ;                        EXPONENT OUT OF THE FE AND PUT IT BACK INTO THE NUMBER IN THE AR.
; 1297  ;
; 1298  ;                    6.  FE -- THE FE REGISTER
; 1299  ;
; 1300  ;                    7.  FLAGS -- THE PC FLAGS (FROM DPE9) IN THE LEFT HALF.
; 1301  ;
; 1302  ;                    8.  Q -- THE Q REGISTER
; 1303  ;
; 1304  ;                    9.  RAM -- THE RAM FILE, RAM ADDRESS IS IN THE VMA.
; 1305  ;
; 1306  ;                   10.  P -- THE P FIELD OF THE BYTE POINTER.  SAME IDEA AS EXP.
; 1307  ;
; 1308  ;                   11.  TIME -- THE 1MS.  TIMER
; 1309  ;
; 1310  ;                   12.  VMA -- THE VMA.  WHEN READ IT INCLUDES THE VMA FLAGS
; 1311  ;
; 1312  ;                   13.  XR -- INDEX REGISTER
; 1313
```

14. XWD -- HALF WORD. USED TO GENERATE CONSTANTS. FOR EXAMPLE, [AR]_O XWD [4O] WOULD LOAD THE CONSTANT 4O (OCTAL) INTO THE AR.

15. +SIGN AND -SIGN -- SIGN BITS USED TO SIGN SMEAR F.P. NUMBERS. FOR EXAMPLE, [AR]_+SIGN WOULD CLEAR AR BITS O TO 8.

16. WORK[] -- LOCATIONS IN THE WORKSPACE USED AS SCRATCH SPACE. FOR EXAMPLE, [AR]_WORK[CSTM] WOULD LOAD THE AR WITH THE CST MASK FROM THE RAM. CSTM IS A SYMBOL DEFINED IN THE WORK FIELD.

1.2 LONG

LONG IS USED ON SHIFT OPERATIONS TO INDICATE THAT THE Q REGISTER IS ALSO SHIFTED. THIS SAYS NOTHING ABOUT HOW THE SHIFT PATHS ARE CONNECTED UP.

2.O MEMORY MACROS

MEMORY IS INDICATED BY THE SYMBOL "MEM". WHEN WE ARE WAITING FOR DATA FROM MEMORY THE "MEM READ" MACRO IS USED. WHEN WE ARE SENDING DATA TO MEMORY, THE "MEM WRITE" MACRO IS USED. EXAMPLE,

```
        MEM READ,               ;WAIT FOR MEMORY
        [AR]_MEM                ;LOAD DATA INTO AR
```

VMA_ IS USED THE LOAD THE VMA. THUS, VMA_[PC] LOADS THE VMA FROM THE PC.

3.O TIME CONTROL

THERE ARE 2 SETS OF MACROS USED FOR TIME CONTROL. THE FIRST, SELECTS THE RAM ADDRESS TO SPEED UP THE NEXT INSTRUCTION. THESE MACROS ARE AC, AC[], XR, VMA, WORK[]. THE SECOND, SETS THE TIME FIELD. THESE ARE 2T, 3T, 4T, AND 5T TO SELECT 2, 3, 4, OR 5 TICKS.

4.O SCAD MACROS

THE SCAD MACROS LOOK LIKE THE 2901 MACROS EXECPT NO [] ARE REQUIRED. THERE ARE ONLY A FEW SYMBOLS USED.

1. FE -- THE FE REGISTER

2. SC -- THE SC REGISTER

```
;                    3.  EXP -- THE EXPONENT FROM A F.P. NUMBER.  FOR EXAMPLE FE_EXP LOADS  THE
;                        FE FROM DP BITS 1-8.
;
;                    4.  SHIFT -- THE SHIFT COUNT FROM SHIFT INSTRUCTIONS.  THAT IS DP  BITS  18
;                        AND 28-35.
;
;                    5.  S# -- THE SMALL NUMBER.  THE 10 BIT MAGIC NUMBER  INPUT  TO  THE  SCADA
;                        MIXER.
;
;
;
;
;               5.0  CONTROL MACROS
;
;               ALL CONTROL MACROS LOOK LIKE ENGLISH COMMANDS.  SOME EXAMPLES,
;                       HOLD LEFT               ;DO NOT CLOCK LEFT HALF OF DP
;                       SET APR ENABLES         ;LOAD APR ENABLES FROM DP
;                       SET NO DIVIDE           ;SET NO DIVIDE PC FLAG
;
;
;
;               6.0  SKIPS
;
;               ALL SKIPS CAUSE THE NEXT MICRO INSTRUCTION TO COME  FROM  THE  ODD  WORD  OF  AN
;               EVEN/ODD PAIR.  THE MACROS HAVE THE FORMAT OF SKIP COND.  THEY SKIP IF CONDITION
;               IS TRUE.  SOME EXAMPLES,
;                       SKIP AD.EQ.0            ;SKIP IF ADDER OUTPUT IS ZERO
;                       SKIP IRPT               ;SKIP IF INTERRUPT IS PENDING
;
;
;
;               7.0  DISPATCH MACROS
;
;               DISPATCH MACROS CAUSE THE MACHINE TO GO TO ONE OF MANY PLACES.  IN  MOST  CASES
;               THEY HAVE THE WORD "DISP" IN THE NAME OF THE MACRO.  FOR EXAMPLE, MUL DISP, BYTE
;               DISP.
;
;
;
;               8.0  SUPER MACROS
;
;               THERE ARE PLACES WHERE ONE MICRO INSTRUCTION  IS  USED  IN  MANY  PLACES.  FOR
;               EXAMPLE,  MANY  PLACES  DETECT ILLEGAL OPERATIONS AND WANT TO GENERATE A TRAP TO
;               THE MONITOR.  WE COULD WRITE
;                       J/UUO
;               BUT THIS WASTES A MICRO STEP DOING A USELESS JUMP.  INSTEAD WE WRITE,
;                       UUO
;               THIS MACRO IS THE FIRST STEP  OF  THE  UUO  ROUTINE  AND  JUMPS  TO  THE  SECOND
;               INSTRUCTION.  WE  WRITE THE EXPANSION OF THE UUO MACRO AS THE FIRST INSTRUCTION
;               OF THE UUO ROUTINE SO THAT THE READER CAN SEE WHAT IT DOES.  SOME  EXAMPLES  OF
;
```

```
; 1414  ;
; 1415  ;
; 1416  ;
; 1417  ;
; 1418  ;
; 1419  ;
; 1420  ;
; 1421
```

```
SUPER MACROS ARE:
        PAGE FAIL TRAP          ;GENERATE A PAGE FAIL TRAP
        DONE                    ;THIS INSTRUCTION IS NOW COMPLETE
                                ; USED WITH A SKIP OR DISP WHERE
                                ; SOME PATHS ARE NOP'S
        HALT []                 ;JUMP TO HALT LOOP. ARGUMENT IS A
                                ; CODE
```

```
; 1422  .TOC    "MACROS -- DATA PATH CHIP -- GENERAL"
; 1423
; 1424  .NOT.[]          "AD/.NOT.A,A/@1"
; 1425  []+[]            "AD/A+B,A/@1,B/@2"
; 1426  []-[]            "AD/A-B-.25,A/@1,B/@2,ADD .25"
; 1427  []-#             "AD/A-D-.25,DBUS/DBM,DBM/#,A/@1,ADD .25"
; 1428  [].AND.#         "AD/D.AND.A,DBUS/DBM,DBM/#,A/@1"
; 1429  [].AND.Q         "AD/A.AND.Q,A/@1,DEST/PASS"
; 1430  [].AND.[]        "AD/A.AND.B,A/@2,B/@1,DEST/PASS"
; 1431  [].AND.NOT.[]    "AD/.NOT.A.AND.B,A/@2,B/@1,DEST/PASS"
; 1432  [].OR.[]         "AD/A.OR.B,A/@2,B/@1,DEST/PASS"
; 1433  [].XOR.#         "AD/D.XOR.A,DBUS/DBM,DBM/#,A/@1"
; 1434  [].XOR.[]        "AD/A.XOR.B,A/@2,B/@1,DEST/PASS"
; 1435  []_#-[]          "AD/D-A-.25,DEST/AD,A/@2,B/@1,DBUS/DBM,DBM/#,ADD .25"
; 1436  []_#             "AD/D,DBUS/DBM,DBM/#,DEST/AD,B/@1"
; 1437  []_-1            "AD/-A-.25,A/ONE,DEST/AD,B/@1,ADD .25"
; 1438  []_-2            "AD/-A-.25,DEST/AD*2,A/ONE,B/@1,ADD .25"
; 1439  []_-Q            "AD/-Q-.25,DEST/AD,B/@1,ADD .25"
; 1440  []_-Q*2          "AD/-Q-.25,DEST/AD*2,B/@1,ADD .25"
; 1441  []_-Q*.5         "AD/-Q-.25,DEST/AD*.5,B/@1,ADD .25"
; 1442  []_-[]           "AD/-A-.25,A/@2,DEST/AD,B/@1,ADD .25"
; 1443  []_-[]-.25       "AD/-A-.25,A/@2,DEST/AD,B/@1"
; 1444  []_-[]*2         "AD/-A-.25,A/@2,DEST/AD*2,B/@1,ADD .25"
; 1445  []_.NOT.AC       "AD/.NOT.D,DBUS/RAM,RAMADR/AC#,DEST/AD,B/@1"
; 1446  []_.NOT.AC[]     "AD/.NOT.D,DBUS/RAM,RAMADR/AC*#,ACALU/AC+N,ACN/@2,DEST/AD,B/@1,DT/3T"
; 1447  []_.NOT.Q        "AD/.NOT.Q,DEST/AD,B/@1"
; 1448  []_.NOT.[]       "AD/.NOT.A,A/@2,DEST/AD,B/@1"
; 1449  []_0             "AD/ZERO,DEST/AD,B/@1"
; 1450  []_0*.5 LONG     "AD/ZERO,DEST/Q_Q*.5,B/@1"
; 1451  []_0 XWD []      "AD/47,DEST/AD,B/@1,DBM/#,DBUS/DBM,#/@2,RSRC/DA,A/MASK"
; 1452  []_AC            "AD/D,DBUS/RAM,RAMADR/AC#,DEST/AD,B/@1,AD PARITY"
; 1453  []_-AC           "AD/-D-.25,DBUS/RAM,RAMADR/AC#,DEST/AD,B/@1,ADD .25"
; 1454  []_-AC[]         "AD/-D-.25,DBUS/RAM,RAMADR/AC*#,ACALU/AC+N,ACN/@2,DEST/AD,B/@1,ADD .25,DT/3T"
; 1455  []_AC*.5         "AD/D,DBUS/RAM,RAMADR/AC#,DEST/AD*.5,B/@1,DT/3T"
; 1456  []_AC*.5 LONG    "AD/D,DBUS/RAM,RAMADR/AC#,DEST/Q_Q*.5,B/@1,DT/3T"
; 1457  []_AC*2          "AD/D,DBUS/RAM,RAMADR/AC#,DEST/AD*2,B/@1,DT/3T"
; 1458  []_AC+1          "AD/D+A,DBUS/RAM,RAMADR/AC#,A/ONE,DEST/AD,B/@1"
; 1459  []_AC+1000001    "AD/D+A,DBUS/RAM,RAMADR/AC#,A/XWD1,DEST/AD,B/@1"
; 1460  []_AC+[]         "AD/D+A,A/@2,DBUS/RAM,RAMADR/AC#,DEST/AD,B/@1,DT/3T"
; 1461  []_AC-1          "AD/D-A-.25,DBUS/RAM,RAMADR/AC#,A/ONE,DEST/AD,B/@1,ADD .25"
; 1462  []_AC-[]         "AD/D-A-.25,A/@2,DBUS/RAM,RAMADR/AC#,DEST/AD,B/@1,ADD .25"
; 1463  []_AC-[]-.25     "AD/D-A-.25,A/@2,DBUS/RAM,RAMADR/AC#,DEST/AD,B/@1"
; 1464  []_AC[]-[]       "AD/D-A-.25,A/@3,DBUS/RAM,RAMADR/AC*#,ACALU/AC+N,ACN/@2,DEST/AD,B/@1,ADD .25,DT/3T"
; 1465  []_AC[]-1        "AD/D-A-.25,A/ONE,DBUS/RAM,RAMADR/AC*#,ACALU/AC+N,ACN/@2,DEST/AD,B/@1,ADD .25,DT/3T"
; 1466  []_AC[].AND.[]   "AD/D.AND.A,A/@3,DBUS/RAM,RAMADR/AC*#,ACALU/AC+N,ACN/@2,DEST/AD,B/@1,DT/3T"
; 1467  []_AC.AND.MASK   "AD/D.AND.A,A/MASK,DBUS/RAM,RAMADR/AC#,DEST/AD,B/@1,AD PARITY"
; 1468  []_AC[]          "AD/D,DBUS/RAM,RAMADR/AC*#,ACALU/AC+N,ACN/@2,DEST/AD,B/@1,AD PARITY,DT/3T"
; 1469  []_AC[]*2        "AD/D,DBUS/RAM,RAMADR/AC*#,ACALU/AC+N,ACN/@2,DEST/AD*2,B/@1,AD PARITY,DT/3T"
; 1470  []_AC[]*.5       "AD/D,DBUS/RAM,RAMADR/AC*#,ACALU/AC+N,ACN/@2,DEST/AD*.5,B/@1,AD PARITY,DT/3T"
; 1471  []_APR           "AD/D,DBUS/DBM,DBM/APR FLAGS,DEST/AD,B/@1,DT/3T"
; 1472  []_CURRENT AC [] "AD/D,DBUS/RAM,RAMADR/#,ACALU/B,ACN/@2,DEST/AD,B/@1,AD PARITY,DT/3T"
; 1473  []_EA FROM []    "AD/57,RSRC/OA,A/@2,DEST/AD,B/@1"
; 1474
```

```
; 1475  []_EA              "AD/57,RSRC/OA,A/HR,DEST/AD,B/@1"
; 1476  []_EXP             "AD/D,DBUS/DBM,DBM/EXP,A/@1,B/@1,DEST/A,SCAD/A+B,SCADA/S#,S#/O,SCADB/FE,HOLD RIGHT,EXP TEST"
; 1477  []_FE              "AD/D,DEST/AD*.5,B/@1,DBUS/DBM,DBM/DP,SCAD/A+B,SCADA/S#,S#/O,SCADB/FE,BYTE/BYTE5"
; 1478  []_FLAGS           "AD/D.AND.A,DBUS/PC FLAGS,A/MASK,DEST/AD,B/@1,RSRC/OQ"
; 1479  []_P               "AD/D,DEST/A,A/@1,B/@1,DBUS/DBM,DBM/DP,BYTE/BYTE1,SCAD/A+B,SCADA/S#,S#/O,SCADB/FE"
; 1480  []_PC WITH FLAGS "AD/D,DBUS/PC FLAGS,RSRC/OA,A/PC,DEST/AD,B/@1"
; 1481  []_Q               "AD/Q,DEST/AD,B/@1"
; 1482  []_Q*.5            "AD/Q,DEST/AD*.5,B/@1"
; 1483  []_Q*2             "AD/Q,DEST/AD*2,B/@1"
; 1484  []_Q*2 LONG        "AD/Q,DEST/Q_Q*2,B/@1"
; 1485  []_Q+1             "AD/A+Q,A/ONE,DEST/AD,B/@1"
; 1486  []_RAM             "AD/D,DBUS/RAM,RAMADR/RAM,DEST/AD,B/@1,AD PARITY"
; 1487  []_TIME            "AD/44,RSRC/DA,A/MASK,DBUS/DBM,DBM/EXP,DEST/AD,B/@1"
; 1488  []_VMA             "AD/D,DEST/AD,B/@1,DBUS/DBM,DBM/VMA"
; 1489  []_XR              "AD/D,DBUS/RAM,RAMADR/XR#,DEST/AD,B/@1"
; 1490  []_[]              "AD/A,A/@2,DEST/AD,B/@1"
; 1491  []_[] SWAP         "AD/D,DBUS/DBM,DBM/DP SWAP,DEST/A,A/@2,B/@1"
; 1492  []_[] XWD O        "AD/45,DEST/AD,B/@1,DBM/#,DBUS/DBM,#/@2,RSRC/DO,A/MASK"
; 1493  []_[]*.5           "AD/A,A/@2,DEST/AD*.5,B/@1"
; 1494  []_[]*.5 LONG      "AD/A,A/@2,DEST/Q_Q*.5,B/@1"
; 1495  []_[]*2            "AD/A,A/@2,DEST/AD*2,B/@1"
; 1496  []_[]*2 LONG       "AD/A,A/@2,DEST/Q_Q*2,B/@1"
; 1497  []_[]*4            "AD/A+B,A/@2,B/@1,DEST/AD*2"
; 1498  []_[]+#            "AD/D+A,DBUS/DBM,DBM/#,A/@2,DEST/AD,B/@1"
; 1499  []_[]+.25          "AD/O+A,A/@2,DEST/AD,B/@1, ADD .25"
; 1500  []_[]+O            "AD/O+A,A/@2,DEST/AD,B/@1"
; 1501  []_[]+1            "AD/A+B,A/ONE,B/@1,B/@2,DEST/AD"
; 1502  []_[]+1000001      "AD/D+A,A/@2,DBUS/DBM,DBM/#,#/1,DEST/AD,B/@1"
; 1503  []_[]+AC           "AD/D+A,A/@2,DBUS/RAM,RAMADR/AC#,DEST/AD,B/@1"
; 1504  []_[]+AC[]         "AD/D+A,A/@2,DBUS/RAM,RAMADR/AC*#,ACALU/AC+N,ACN/@3,DEST/AD,B/@1,DT/3T"
; 1505  []_[]+Q            "AD/A+Q,A/@2,DEST/AD,B/@1"
; 1506  []_[]+RAM          "AD/D+A,A/@2,DBUS/RAM,RAMADR/RAM,DEST/AD,B/@1"
; 1507  []_[]+XR           "AD/D+A,DBUS/RAM,RAMADR/XR#,A/@2,DEST/AD,B/@1,HOLD LEFT"
; 1508  []_[]+[]           "AD/A+B,A/@3,B/@1,B/@2,DEST/AD"
; 1509  []_[]+[]+.25       "AD/A+B,A/@3,B/@1,B/@2,DEST/AD, ADD .25"
; 1510  []_[]-#            "AD/A-D-.25,DBUS/DBM,DBM/#,A/@2,DEST/AD,B/@1, ADD .25"
; 1511  []_[]-1            "AD/B-A-.25,B/@1,A/ONE,DEST/AD,ADD .25"
; 1512  []_[]-1000001      "AD/A-D-.25,A/@2,DBUS/DBM,DBM/#,#/1,DEST/AD,B/@1,ADD .25"
; 1513  []_[]-AC           "AD/A-D-.25,A/@2,DBUS/RAM,RAMADR/AC#,DEST/AD,B/@1,ADD .25"
; 1514  []_[]-RAM          "AD/A-D-.25,A/@2,DBUS/RAM,RAMADR/RAM,DEST/AD,B/@1,ADD .25"
; 1515  []_[]-[]           "AD/B-A-.25,B/@1,B/@2,A/@3,DEST/AD,ADD .25"
; 1516  []_[]-[] REV       "AD/A-B-.25,B/@1,B/@3,A/@2,DEST/AD,ADD .25"
; 1517  []_[].AND.#        "AD/D.AND.A,DBUS/DBM,DBM/#,DEST/AD,A/@2,B/@1"
; 1518  []_[].AND.# CLR LH "AD/ZERO,RSRC/DA,DBUS/DBM,DBM/#,DEST/AD,A/@2,B/@1"
; 1519  []_[].AND.# CLR RH "AD/D.AND.A,RSRC/OQ,DBUS/DBM,DBM/#,DEST/AD,A/@2,B/@1"
; 1520  []_(AC[].AND.[])*.5 "AD/D.AND.A,DEST/AD*.5,A/@3,B/@1,RAMADR/AC*#,DBUS/RAM,ACALU/AC+N,ACN/@2"
; 1521  []_(Q+1)*.5        "AD/A+Q,A/ONE,DEST/AD*.5,B/@1"
; 1522  []_(#-[])*2        "AD/D-A-.25,DEST/AD*2,A/@2,B/@1,DBUS/DBM,DBM/#,ADD .25"
; 1523  []_(-[])*.5        "AD/-A-.25,A/@2,DEST/AD*.5,B/@1,ADD .25"
; 1524  []_(-[]-.25)*.5 LONG "AD/-A-.25,A/@2,DEST/Q_Q*.5,B/@1"
; 1525  []_(-[]-.25)*2 LONG "AD/-A-.25,A/@2,DEST/Q_Q*2,B/@1"
; 1526
```

```
; 1527  []_([].AND.#)*.5 "AD/D.AND.A,DBUS/DBM,DBM/#,DEST/AD*.5,A/@2,B/@1"
; 1528  []_([].AND.#)*2 "AD/D.AND.A,DBUS/DBM,DBM/#,DEST/AD*2,A/@2,B/@1"
; 1529  []_([].AND.NOT.#)*.5 "AD/.NOT.D.AND.A,DBUS/DBM,DBM/#,DEST/AD*.5,A/@2,B/@1"
; 1530  []_([].AND.NOT.#)*2     "AD/.NOT.D.AND.A,DBUS/DBM,DBM/#,DEST/AD*2,A/@2,B/@1"
; 1531  []_([].AND.[])*.5 "AD/A.AND.B,DEST/AD*.5,A/@3,B/@1,B/@2"
; 1532  []_([].AND.[])*2 "AD/A.AND.B,DEST/AD*2,A/@3,B/@1,B/@2"
; 1533  []_([].OR.#)*.5 "AD/D.OR.A,DBUS/DBM,DBM/#,DEST/AD*.5,A/@2,B/@1"
; 1534  []_([].OR.#)*2  "AD/D.OR.A,DBUS/DBM,DBM/#,DEST/AD*2,A/@2,B/@1"
; 1535  []_([]+#)*2     "AD/D+A,DBUS/DBM,DBM/#,DEST/AD*2,A/@2,B/@1"
; 1536  []_([]+1)*2     "AD/A+B,A/ONE,B/@1,B/@2,DEST/AD*2"
; 1537  []_([]+[])*.5 LONG      "AD/A+B,A/@3,B/@1,B/@2,DEST/Q_Q*.5"
; 1538  []_([]+[])*2 LONG       "AD/A+B,A/@3,B/@1,B/@2,DEST/Q_Q*2"
; 1539  []_([]-[])*.5 LONG      "AD/B-A-.25,A/@3,B/@1,B/@2,DEST/Q_Q*.5, ADD .25"
; 1540  []_([]-[])*2 LONG       "AD/B-A-.25,A/@3,B/@1,B/@2,DEST/Q_Q*2, ADD .25"
; 1541  []_([]+[]+.25)*.5 LONG  "AD/A+B,A/@3,B/@1,B/@2,DEST/Q_Q*.5, ADD .25"
; 1542  []_[].AND.AC    "AD/D.AND.A,DBUS/RAM,RAMADR/AC#,A/@2,DEST/AD,B/@1"
; 1543  []_[].AND.NOT.# "AD/.NOT.D.AND.A,DBUS/DBM,DBM/#,A/@2,DEST/AD,B/@1"
; 1544  []_[].AND.NOT.[] "AD/.NOT.A.AND.B,DEST/AD,B/@1,B/@2,A/@3"
; 1545  []_[].AND.NOT.AC "AD/.NOT.D.AND.A,DBUS/RAM,RAMADR/AC#,A/@2,DEST/AD,B/@1"
; 1546  []_[].AND.Q     "AD/A.AND.Q,A/@2,DEST/AD,B/@1"
; 1547  []_[].AND.[]    "AD/A.AND.B,A/@3,B/@1,B/@2,DEST/AD"
; 1548  []_[].EQV.AC    "AD/D.EQV.A,DBUS/RAM,RAMADR/AC#,A/@2,DEST/AD,B/@1"
; 1549  []_[].EQV.Q     "AD/A.EQV.Q,A/@2,DEST/AD,B/@1"
; 1550  []_[].OR.#      "AD/D.OR.A,DBUS/DBM,DBM/#,A/@2,DEST/AD,B/@1"
; 1551  []_[].OR.AC     "AD/D.OR.A,DBUS/RAM,RAMADR/AC#,A/@2,DEST/AD,B/@1"
; 1552  []_[].OR.FLAGS  "AD/D.OR.A,DBUS/PC FLAGS,RSRC/OA,A/@1,DEST/AD,B/@1"
; 1553  []_[].OR.[]     "AD/A.OR.B,A/@3,B/@2,B/@1,DEST/AD"
; 1554  []_[].XOR.#     "AD/D.XOR.A,DBUS/DBM,DBM/#,DEST/AD,A/@2,B/@1"
; 1555  []_[].XOR.AC    "AD/D.XOR.A,DBUS/RAM,RAMADR/AC#,A/@1,DEST/AD,B/@2"
; 1556  []_[].XOR.[]    "AD/A.XOR.B,A/@3,B/@1,B/@2,DEST/AD"
; 1557
; 1558  [] LEFT_O       "AD/57,RSRC/OB,DEST/AD,B/@1"
; 1559  [] RIGHT_O      "AD/53,RSRC/DO,DEST/AD,B/@1"
; 1560  [] LEFT_-1      "AD/54,RSRC/OB,DEST/AD,A/MASK,B/@1"
; 1561  [] RIGHT_-1     "AD/53,RSRC/OA,DEST/AD,A/MASK,B/@1"
; 1562
; 1563
; 1564  []_+SIGN        "[@1]_[@1].AND.#, #/777, HOLD RIGHT"
; 1565  []_-SIGN        "[@1]_[@1].OR.#, #/777000, HOLD RIGHT"
; 1566  ;THE FOLLOWING 2 MACROS ARE USED IN DOUBLE FLOATING STUFF
; 1567  ; THEY ASSUME THAT THE OPERAND HAS BEEN SHIFTED RIGHT 1 PLACE.
; 1568  ; THEY SHIFT 1 MORE PLACE
; 1569  []_+SIGN*.5     "AD/.NOT.D.AND.A,A/@1,B/@1,DEST/AD*.5,DBUS/DBM,DBM/#,#/777400,RSRC/OA"
; 1570  []_-SIGN*.5     "AD/D.OR.A,A/@1,B/@1,DEST/AD*.5,DBUS/DBM,DBM/#,#/777400,RSRC/OA"
; 1571
```

```
; 1572  .TOC    "MACROS -- DATA PATH CHIP -- Q"
; 1573
; 1574  Q-[]              "AD/Q-A-.25,A/@1,ADD .25"
; 1575  Q.AND.NOT.[]      "AD/.NOT.A.AND.Q,A/@1,DEST/PASS"
; 1576  Q_[]              "AD/A,DEST/Q_AD,A/@1"
; 1577  Q_[]-[]           "AD/A-B-.25,A/@1,B/@2,DEST/Q_AD,ADD .25"
; 1578  Q_[]+[]           "AD/A+B,A/@1,B/@2,DEST/Q_AD"
; 1579  Q_[].AND.[]       "AD/A.AND.B,A/@1,B/@2,DEST/Q_AD"
; 1580  Q_.NOT.AC[]       "AD/.NOT.D,DBUS/RAM,RAMADR/AC*#,ACALU/AC+N,ACN/@1,DEST/Q_AD,DT/3T"
; 1581  Q_-[]             "AD/-A-.25,DEST/Q_AD,A/@1, ADD .25"
; 1582  Q_-1              "Q_-[ONE]"
; 1583  Q_-AC[] "AD/-D-.25,DBUS/RAM,RAMADR/AC*#,ACALU/AC+N,ACN/@1,DEST/Q_AD,ADD .25,DT/3T"
; 1584  Q_-Q              "AD/-Q-.25,ADD .25,DEST/Q_AD"
; 1585  Q_AC              "AD/D,DBUS/RAM,RAMADR/AC#,DEST/Q_AD,CHK PARITY"
; 1586  Q_AC[]            "AD/D,DBUS/RAM,RAMADR/AC*#,ACALU/AC+N,ACN/@1,DEST/Q_AD,CHK PARITY,DT/3T"
; 1587  Q_AC[].AND.MASK   "AD/D.AND.A,A/MASK,DBUS/RAM,RAMADR/AC*#,ACALU/AC+N,ACN/@1,DEST/Q_AD,CHK PARITY,DT/3T"
; 1588  Q_AC[].AND.[]     "AD/D.AND.A,A/@2,DBUS/RAM,RAMADR/AC*#,ACALU/AC+N,ACN/@1,DEST/Q_AD,CHK PARITY,DT/3T"
; 1589  Q_.NOT.Q                "AD/.NOT.Q,DEST/Q_AD"
; 1590  Q_#               "AD/D,DBUS/DBM,DBM/#,DEST/Q_AD"
; 1591  Q_O               "AD/ZERO,DEST/Q_AD"
; 1592  Q_O XWD []        "AD/47,DEST/Q_AD,DBM/#,DBUS/DBM,#/@1,RSRC/DA,A/MASK"
; 1593  Q_Q+.25           "AD/O+Q,DEST/Q_AD,ADD .25"
; 1594  Q_Q+1             "AD/A+Q,A/ONE,DEST/Q_AD"
; 1595  Q_Q-1             "AD/Q-A-.25,A/ONE,DEST/Q_AD, ADD .25"
; 1596  Q_Q+AC            "AD/D+Q,DBUS/RAM,RAMADR/AC#,DEST/Q_AD"
; 1597  Q_Q*.5            "[MAG]_[MASK]*.5 LONG, SHSTYLE/NORM"
; 1598  Q_Q*2             "[MASK]_[MAG]*2 LONG, SHSTYLE/NORM"
; 1599  Q_Q.OR.#          "AD/D.OR.Q,DBUS/DBM,DBM/#,DEST/Q_AD"
; 1600  Q_Q.AND.#         "AD/D.AND.Q,DBUS/DBM,DBM/#,DEST/Q_AD"
; 1601  Q_Q.AND.[]        "AD/A.AND.Q,A/@1,DEST/Q_AD"
; 1602  Q_Q.AND.NOT.[]    "AD/.NOT.A.AND.Q,A/@1,DEST/Q_AD"
; 1603  Q_Q+[]            "AD/A+Q,A/@1,DEST/Q_AD"
; 1604  Q_[].AND.Q        "AD/A.AND.Q,A/@1,DEST/Q_AD"
; 1605  Q_[].OR.Q         "AD/A.OR.Q,A/@1,DEST/Q_AD"
; 1606
```

```
; 1607  .TOC    "MACROS -- DATA PATH CHIP -- MISC."
; 1608
; 1609  CLEAR []O       "AD/D.AND.A,A/@1,DBUS/DBM,DBM/#,#/377777,DEST/AD,B/@1,HOLD RIGHT"
; 1610  CLEAR ARXO      "CLEAR [ARX]O"
; 1611
; 1612  ;CYCLE CHIP REGISTERS THRU AD SO WE CAN TEST BITS
; 1613  READ XR         "AD/D,DBUS/RAM,RAMADR/XR#"
; 1614  READ []         "AD/B,B/@1"
; 1615  READ Q          "AD/Q"
; 1616
; 1617  ;TEST BITS IN REGISTERS (SKIP IF ZERO)
; 1618  TR []           "AD/D.AND.A,DBUS/DBM,DBM/#,A/@1,SKIP ADR.EQ.O,DT/3T"
; 1619  TL []           "AD/D.AND.A,DBUS/DBM,DBM/#,A/@1,SKIP ADL.EQ.O,DT/3T"
; 1620
; 1621
; 1622  ;CAUSE BITS -2 AND -1 TO MATCH BIT O.
; 1623  FIX [] SIGN     "AD/D,DEST/A,A/@1,B/@1,DBUS/DP,HOLD RIGHT"
; 1624
; 1625  ;GENERATE A MASK IN Q AND ZERO A 2901 REGISTER
; 1626  GEN MSK []      "AD/ZERO,DEST/Q_Q*2,B/@1,ONES"
; 1627
```

```
; 1628  .TOC    "MACROS -- STORE IN AC"
; 1629
; 1630  FM WRITE         "FMWRITE/1"
; 1631
; 1632  AC[]_[] VIA AD   "AD/B,DEST/PASS,B/@2,RAMADR/AC*#,ACALU/AC+N,ACN/@1,DBUS/DP,FM WRITE,CHK PARITY"
; 1633  AC_[] VIA AD     "AD/B,DEST/PASS,B/@1,RAMADR/AC#,DBUS/DP,FM WRITE,CHK PARITY"
; 1634  AC[]_[]          "AD/A,DEST/A,B/@2,A/@2,RAMADR/AC*#,ACALU/AC+N,ACN/@1,DBUS/DP, FM WRITE"
; 1635  AC[]_[] TEST     "AD/D,DBUS/DP,DEST/A,B/@2,A/@2,RAMADR/AC*#,ACALU/AC+N,ACN/@1,DBUS/DP, FM WRITE"
; 1636  AC[]_[]+1        "AD/A+B,DEST/PASS,A/ONE,B/@2,RAMADR/AC*#,ACALU/AC+N,ACN/@1,DBUS/DP,FM WRITE"
; 1637  AC[]_[]*2        "AD/A+B,DEST/PASS,A/@2,B/@2,RAMADR/AC*#,ACALU/AC+N,ACN/@1,DBUS/DP,FM WRITE"
; 1638  AC_[]            "AD/A,DEST/A,B/@1,A/@1,RAMADR/AC#,DBUS/DP, FM WRITE"
; 1639  AC_[] TEST       "AD/D,DBUS/DP,DEST/A,B/@1,A/@1,RAMADR/AC#,DBUS/DP, FM WRITE"
; 1640  AC_[]+1          "AD/A+B,DEST/PASS,A/ONE,B/@1,RAMADR/AC#, FM WRITE"
; 1641  AC_[]+Q          "AD/A+Q,DEST/PASS,A/@1,B/@1,RAMADR/AC#, FM WRITE"
; 1642  AC[]_[]+Q        "AD/A+Q,DEST/PASS,A/@2,B/@2,RAMADR/AC*#,ACALU/AC+N,ACN/@1, FM WRITE"
; 1643  AC[]_[]-[]       "AD/A-B-.25,DEST/PASS,B/@3,A/@2,RAMADR/AC*#,ACALU/AC+N,ACN/@1,DBUS/DP,FM WRITE,ADD .25"
; 1644  AC[]_[]+[]       "AD/A+B,DEST/PASS,B/@3,A/@2,RAMADR/AC*#,ACALU/AC+N,ACN/@1,DBUS/DP,FM WRITE"
; 1645  AC_[]+[]         "AD/A+B,DEST/PASS,B/@2,A/@1,RAMADR/AC#,DBUS/DP,FM WRITE"
; 1646  AC[]_[].AND.[]   "AD/A.AND.B,DEST/PASS,B/@3,A/@2,RAMADR/AC*#,ACALU/AC+N,ACN/@1,DBUS/DP,FM WRITE"
; 1647  AC[]_Q.AND.[]    "AD/A.AND.Q,DEST/PASS,A/@2,RAMADR/AC*#,ACALU/AC+N,ACN/@1,DBUS/DP,FM WRITE"
; 1648  AC[]_[].EQV.Q    "AD/A.EQV.Q,DEST/PASS,A/@2,RAMADR/AC*#,ACALU/AC+N,ACN/@1,DBUS/DP,FM WRITE"
; 1649  AC[]_-[]         "AD/-B-.25,DEST/PASS,B/@2,RAMADR/AC*#,ACALU/AC+N,ACN/@1,DBUS/DP,FM WRITE,ADD .25"
; 1650  AC_-[]           "AD/-A-.25,DEST/PASS,A/@1,RAMADR/AC#,DBUS/DP, ADD .25,FM WRITE"
; 1651  AC_[].OR.[]      "AD/A.OR.B,A/@1,B/@2,RAMADR/AC#,DBUS/DP, FM WRITE"
; 1652  AC[]_.NOT.[]     "AD/.NOT.B,DEST/PASS,B/@2,RAMADR/AC*#,ACALU/AC+N,ACN/@1,DBUS/DP,FM WRITE"
; 1653  AC_.NOT.[]       "AD/.NOT.B,DEST/PASS,B/@1,RAMADR/AC#,DBUS/DP,FM WRITE"
; 1654  AC[]_-Q          "AD/-Q-.25,RAMADR/AC*#,ACALU/AC+N,ACN/@1,DBUS/DP,FM WRITE,ADD .25"
; 1655  AC_Q             "AD/Q,RAMADR/AC#,DBUS/DP, FM WRITE"
; 1656  AC[]_0           "AD/ZERO,RAMADR/AC*#,ACALU/AC+N,ACN/@1,DBUS/DP, FM WRITE"
; 1657  AC[]_1           "AD/B,DEST/PASS,B/ONE,RAMADR/AC*#,ACALU/AC+N,ACN/@1,DBUS/DP,FM WRITE"
; 1658  AC[]_Q           "AD/Q,RAMADR/AC*#,ACALU/AC+N,ACN/@1,DBUS/DP, FM WRITE"
; 1659
```

```
; 1660  .TOC    "MACROS -- MICROCODE WORK SPACE"
; 1661
; 1662
; 1663  WORK[]_Q          "AD/Q,DEST/PASS,RAMADR/#,WORK/@1,FM WRITE"
; 1664  Q_WORK[]          "AD/D,DEST/Q_AD,RAMADR/#,DBUS/RAM,WORK/@1,DT/3T"
; 1665  WORK[]_0          "AD/ZERO,DEST/PASS,RAMADR/#,WORK/@1,FM WRITE"
; 1666  WORK[]_1          "AD/B,DEST/PASS,RAMADR/#,WORK/@1,B/ONE,FM WRITE"
; 1667  WORK[]_[]         "AD/B,DEST/PASS,RAMADR/#,WORK/@1,B/@2,FM WRITE"
; 1668  WORK[]_[] CLR LH  "AD/47,RSRC/AB,DEST/PASS,RAMADR/#,WORK/@1,B/@2,A/MASK,FM WRITE"
; 1669  WORK[]_[]-1       "AD/A-B-.25,A/@2,B/ONE,DEST/PASS,RAMADR/#,WORK/@1,FM WRITE, ADD .25"
; 1670  WORK[]_.NOT.[]    "AD/.NOT.B,DEST/PASS,RAMADR/#,WORK/@1,B/@2,FM WRITE"
; 1671  WORK[]_[].AND.[]  "AD/A.AND.B,DEST/PASS,RAMADR/#,WORK/@1,A/@2,B/@3,FM WRITE"
; 1672  [].AND.NOT.WORK[] "AD/.NOT.D.AND.A,A/@1,DBUS/RAM,RAMADR/#,WORK/@2,DT/3T"
; 1673  [].AND.WORK[]     "AD/D.AND.A,A/@1,DBUS/RAM,RAMADR/#,WORK/@2,DT/3T"
; 1674  []_[]+WORK[]      "AD/D+A,A/@2,DEST/AD,B/@1,DBUS/RAM,RAMADR/#,WORK/@3,DT/3T"
; 1675  []_[].AND.WORK[]  "AD/D.AND.A,A/@2,DEST/AD,B/@1,DBUS/RAM,RAMADR/#,WORK/@3,DT/3T"
; 1676  []_[].AND.NOT.WORK[] "AD/.NOT.D.AND.A,A/@2,DEST/AD,B/@1,DBUS/RAM,RAMADR/#,WORK/@3,DT/3T"
; 1677  []_[].OR.WORK[]  "AD/D.OR.A,A/@2,DEST/AD,B/@1,DBUS/RAM,RAMADR/#,WORK/@3,DT/3T"
; 1678  []_WORK[]        "AD/D,DEST/AD,B/@1,DBUS/RAM,RAMADR/#,WORK/@2,DT/3T"
; 1679  []_.NOT.WORK[]   "AD/.NOT.D,DEST/AD,B/@1,DBUS/RAM,RAMADR/#,WORK/@2,DT/3T"
; 1680  []_-WORK[]       "AD/-D-.25,ADD .25,DEST/AD,B/@1,DBUS/RAM,RAMADR/#,WORK/@2,DT/3T"
; 1681  []_WORK[]+1      "AD/D+A,A/ONE,DEST/AD,B/@1,DBUS/RAM,RAMADR/#,WORK/@2,DT/3T"
; 1682  Q_Q-WORK[]       "AD/Q-D-.25,DEST/Q_AD,DBUS/RAM,RAMADR/#,WORK/@1,ADD .25,DT/3T"
; 1683  []_[]-WORK[]     "AD/A-D-.25,DEST/AD,A/@2,B/@1,DBUS/RAM,RAMADR/#,WORK/@3,ADD .25,DT/3T"
; 1684
; 1685  RAM_[]           "AD/B,DEST/PASS,RAMADR/RAM,B/@1,FM WRITE"
; 1686
```

```
; 1687  .TOC            "MACROS -- MEMORY CONTROL"
; 1688
; 1689  MEM CYCLE               "MEM/1"
; 1690
; 1691  ;THE FOLLOWING MACROS CONTROL MEMORY ADDRESS
; 1692  LOAD VMA                "MEM CYCLE,LDVMA/1"
; 1693  FORCE EXEC               "FORCE EXEC/1"
; 1694  VMA PHYSICAL            "PHYSICAL/1,FORCE EXEC/1,FORCE USER/0,EXT ADR/1,LOAD VMA"
; 1695  VMA PHYSICAL WRITE      "LOAD VMA,VMA PHYSICAL,WAIT/1,MEM/1,WRITE CYCLE/1,WRITE TEST/0"
; 1696  VMA PHYSICAL READ       "LOAD VMA,VMA PHYSICAL,WAIT/1,MEM/1,READ CYCLE/1,WRITE TEST/0"
; 1697  VMA EXTENDED            "EXT ADR/1"
; 1698
; 1699  PXCT EA                 "PXCT/E1"
; 1700  PXCT DATA               "PXCT/D1"
; 1701  PXCT BLT DEST           "PXCT/D1"
; 1702  PXCT BYTE PTR EA        "PXCT/E2"
; 1703  PXCT BYTE DATA          "PXCT/D2"
; 1704  PXCT STACK WORD         "PXCT/D2"
; 1705  PXCT BLT SRC            "PXCT/D2"
; 1706  PXCT EXTEND EA          "PXCT/E2"
; 1707
; 1708  ;THE FOLLOWING MACROS GET MEMORY CYCLES STARTED
; 1709  WRITE TEST              "WRITE TEST/1,WAIT/1"
; 1710  START READ              "MEM CYCLE,READ CYCLE/1,WAIT/1"
; 1711  START WRITE             "MEM CYCLE,WRITE TEST,WRITE CYCLE/1,WAIT/1"
; 1712  START NO TEST WRITE     "MEM CYCLE,WRITE CYCLE/1,WAIT/1"
; 1713  FETCH                   "START READ,FETCH/1,PXCT/CURRENT,WAIT/1"
; 1714
; 1715  ;THE FOLLOWING MACROS COMPLETE MEMORY CYCLES
; 1716  MEM WAIT                "MEM CYCLE,WAIT/1"
; 1717  MEM READ                "MEM WAIT,DBUS/DBM,DBM/MEM"
; 1718  MEM WRITE               "MEM WAIT,DT/3T"
; 1719  SPEC MEM READ           "SPEC/WAIT,DBUS/DBM,DBM/MEM"
; 1720  SPEC MEM WRITE          "SPEC/WAIT,DT/3T"
; 1721
; 1722
; 1723  ;THINGS WHICH WRITE MEMORY
; 1724  MEM_[]                  "AD/B,DEST/PASS,B/@1,DBUS/DP,RAMADR/VMA,CHK PARITY"
; 1725  MEM_Q                   "AD/Q,DBUS/DP,RAMADR/VMA"
; 1726
; 1727
; 1728  ;THINGS WHICH READ MEMORY
; 1729  []_IO DATA              "AD/D,DBUS/DBM,RAMADR/VMA,DEST/AD,B/@1"
; 1730  []_MEM                  "AD/D,DBUS/DBM,RAMADR/VMA,DEST/AD,B/@1,CHK PARITY"
; 1731  []_MEM THEN FETCH       "AD/D,DBUS/DBM,RAMADR/VMA,DEST/A,A/PC,B/@1,CHK PARITY, FETCH, LOAD VMA"
; 1732  []_MEM*.5               "AD/D,DBUS/DBM,RAMADR/VMA,DEST/AD*.5,B/@1,CHK PARITY"
; 1733  []_MEM.AND.MASK         "AD/D.AND.A,A/MASK,DBUS/DBM,RAMADR/VMA,DEST/AD,B/@1,CHK PARITY"
; 1734  []_(MEM.AND.[])*.5      "AD/D.AND.A,A/@2,DBUS/DBM,RAMADR/VMA,DEST/AD*.5,B/@1,CHK PARITY"
; 1735  Q_MEM                   "AD/D,DBUS/DBM,RAMADR/VMA,DEST/Q_AD,CHK PARITY"
; 1736
```

```
; 1737  .TOC    "MACROS -- VMA"
; 1738
; 1739  VMA_[]                    "AD/A,A/@1,DEST/PASS,LOAD VMA"
; 1740  VMA_[] WITH FLAGS         "AD/A,A/@1,DEST/PASS,LOAD VMA,WAIT/1, MEM/1, EXT ADR/1, DP FUNC/1, DT/3T"
; 1741  VMA_[].OR.[] WITH FLAGS  "AD/A.OR.B,A/@1,B/@2,DEST/PASS,LOAD VMA,WAIT/1, MEM/1, EXT ADR/1, DP FUNC/1, DT/3T"
; 1742  VMA_[]+1                  "AD/A+B,A/ONE,B/@1,DEST/AD,HOLD LEFT,LOAD VMA"
; 1743  VMA_[]-1                  "AD/B-A-.25,A/ONE,B/@1,ADD .25,HOLD LEFT,LOAD VMA"
; 1744  VMA_[]+XR                 "AD/D+A,DBUS/RAM,RAMADR/XR#,A/@1,LOAD VMA"
; 1745  VMA_[]+[]                 "AD/A+B,DEST/PASS,A/@1,B/@2,LOAD VMA"
; 1746
; 1747  NEXT [] PHYSICAL WRITE "AD/A+B,A/ONE,B/@1,DEST/AD,HOLD LEFT,LOAD VMA, VMA PHYSICAL, START WRITE"
; 1748
; 1749  ;MACROS TO LOAD A 2901 REGISTER WITH VMA FLAG BITS
; 1750  []_VMA FLAGS     "AD/45,DEST/AD,B/@1,DBM/#,DBUS/DBM,RSRC/DO,A/MASK"
; 1751  []_VMA IO READ   "[@1]_VMA FLAGS,READ CYCLE/1,IO CYCLE/1,WRITE TEST/0, PHYSICAL/1, FORCE EXEC/1, FORCE USER/0"
; 1752  []_VMA IO WRITE  "[@1]_VMA FLAGS,WRITE CYCLE/1,IO CYCLE/1,WRITE TEST/0, PHYSICAL/1, FORCE EXEC/1, FORCE USER/0"
; 1753
```

```
; 1754   .TOC     "MACROS -- TIME CONTROL"
; 1755
; 1756   AC                "RAMADR/AC#"
; 1757   AC[]              "RAMADR/AC*#,ACALU/AC+N,ACN/@1"
; 1758   XR                "RAMADR/XR#"
; 1759   VMA               "RAMADR/VMA"
; 1760   WORK[]            "RAMADR/#, WORK/@1"
; 1761
; 1762   2T                "T/2T"
; 1763   3T                "T/3T"
; 1764   4T                "T/4T"
; 1765   5T                "T/5T"
; 1766
```

```
; 1767  .TOC    "MACROS -- SCAD, SC, FE LOGIC"
; 1768
; 1769  LOAD SC          "LOADSC/1"
; 1770  LOAD FE          "LOADFE/1"
; 1771  STEP SC          "SCAD/A-1,SCADA/SC,LOAD SC,SKIP/SC"
; 1772  SHIFT            "SCAD/A+B,SCADA/S#,SCADB/FE,S#/1, LOAD FE, MULTI SHIFT/1"
; 1773
; 1774  SC_SC-1          "SCAD/A-1,SCADA/SC,LOAD SC"
; 1775  SC_SHIFT         "SCAD/A+B,SCADA/S#,S#/0,SCADB/SHIFT,LOAD SC"
; 1776  SC_SHIFT-1       "SCAD/A+B,SCADA/S#,S#/1777,SCADB/SHIFT,LOAD SC"
; 1777  SC_SHIFT-2       "SCAD/A+B,SCADA/S#,S#/1776,SCADB/SHIFT,LOAD SC"
; 1778  SC_-SHIFT        "SCAD/A-B,SCADA/S#,S#/0000,SCADB/SHIFT,LOAD SC"
; 1779  SC_-SHIFT-1      "SCAD/A-B,SCADA/S#,SCADB/SHIFT,S#/1777,LOAD SC"
; 1780  SC_-SHIFT-2      "SCAD/A-B,SCADA/S#,SCADB/SHIFT,S#/1776,LOAD SC"
; 1781  SC_SC-EXP        "SCAD/A-B,SCADA/SC,SCADB/EXP,LOAD SC"
; 1782  SC_SC-EXP-1      "SCAD/A-B-1,SCADA/SC,SCADB/EXP,LOAD SC"
; 1783  SC_SC-FE-1       "SCAD/A-B-1,SCADA/SC,SCADB/FE,LOAD SC"
; 1784  SC_SC-FE         "SCAD/A-B,SCADA/SC,SCADB/FE,LOAD SC"
; 1785  SC_EXP           "SCAD/A+B,SCADA/S#,S#/0,SCADB/EXP,LOAD SC"
; 1786  SC_S#-FE         "SCAD/A-B,SCADA/S#,SCADB/FE,LOAD SC"
; 1787  SC_FE+S#         "SCAD/A+B,SCADA/S#,SCADB/FE,LOAD SC"
; 1788  SC_FE            "SCAD/A.OR.B,SCADA/S#,S#/0,SCADB/FE,LOAD SC"
; 1789  SC_S#            "SCAD/A,SCADA/S#,LOAD SC"
; 1790
; 1791
; 1792  SC_36.           "SC_S#,S#/36."
; 1793  SC_35.           "SC_S#,S#/35."
; 1794  SC_34.           "SC_S#,S#/34."
; 1795  SC_28.           "SC_S#,S#/28."
; 1796  SC_27.           "SC_S#,S#/27."
; 1797  SC_26.           "SC_S#,S#/26."
; 1798  SC_24.           "SC_S#,S#/24."
; 1799  SC_22.           "SC_S#,S#/22."
; 1800  SC_20.           "SC_S#,S#/20."
; 1801  SC_19.           "SC_S#,S#/19."
; 1802  SC_14.           "SC_S#,S#/14."
; 1803  SC_11.           "SC_S#,S#/11."
; 1804  SC_9.            "SC_S#,S#/9."
; 1805  SC_8.            "SC_S#,S#/8."
; 1806  SC_7             "SC_S#,S#/7"
; 1807  SC_6             "SC_S#,S#/6"
; 1808  SC_5             "SC_S#,S#/5"
; 1809  SC_4             "SC_S#,S#/4"
; 1810  SC_3             "SC_S#,S#/3"
; 1811  SC_2             "SC_S#,S#/2"
; 1812  SC_1             "SC_S#,S#/1"
; 1813  SC_0             "SC_S#,S#/0."
; 1814  SC_-1            "SC_S#,S#/1777"
; 1815  SC_-2            "SC_S#,S#/1776"
; 1816
```

```
; 1817  FE_-FE          "SCAD/A-B,SCADA/S#,S#/0,SCADB/FE,LOAD FE"
; 1818  FE_-FE-1        "SCAD/A-B,SCADA/S#,S#/1777,SCADB/FE,LOAD FE"
; 1819  FE_FE-19        "SCAD/A+B,SCADB/FE,SCADA/S#,S#/1550,LOAD FE"
; 1820  FE_-FE+S#       "SCAD/A-B,SCADA/S#,SCADB/FE,LOAD FE"
; 1821  FE_FE+SC        "SCAD/A+B,SCADA/SC,SCADB/FE, LOAD FE"
; 1822  FE_FE.AND.S#    "SCAD/A.AND.B,SCADA/S#,SCADB/FE, LOAD FE"
; 1823  FE_P            "SCAD/A,SCADA/BYTE1, LOAD FE"
; 1824  FE_S            "SCAD/A+B, SCADA/S#, S#/0 ,SCADB/SIZE, LOAD FE"
; 1825  FE_S+2          "SCAD/A+B, SCADA/S#, S#/20, SCADB/SIZE, LOAD FE"
; 1826  FE_-S-20        "SCAD/A-B,SCADA/S#,S#/1760,SCADB/SIZE, LOAD FE"
; 1827  FE_-S-10        "SCAD/A-B,SCADA/S#,S#/1770,SCADB/SIZE, LOAD FE"
; 1828  FE_S#           "SCAD/A,SCADA/S#,LOAD FE"
; 1829  FE_S#-FE        "SCAD/A-B,SCADA/S#,SCADB/FE,LOAD FE"
; 1830  FE_-2           "FE_S#,S#/1776"
; 1831  FE_-12.         "FE_S#,S#/1764"
; 1832  FE_0            "FE_S#,S#/0"
; 1833  FE_-1           "FE_S#,S#/1777"
; 1834  FE_FE+1         "SCAD/A+B,SCADA/S#,SCADB/FE,S#/1,LOAD FE"
; 1835  FE_FE+2         "SCAD/A+B,SCADA/S#,SCADB/FE,S#/2,LOAD FE"
; 1836  FE_FE+10                "SCAD/A+B,SCADA/S#,SCADB/FE,S#/10,LOAD FE"
; 1837  FE_FE-1         "SCAD/A+B,SCADA/S#,SCADB/FE,S#/1777,LOAD FE"
; 1838  FE_FE+4         "SCAD/A+B,SCADA/S#,SCADB/FE,S#/4,LOAD FE"
; 1839  FE_EXP          "SCAD/A+B,SCADA/S#,S#/0,SCADB/EXP,LOAD FE"
; 1840  FE_SC+EXP       "SCAD/A+B,SCADA/SC,SCADB/EXP,LOAD FE"
; 1841  FE_SC-EXP       "SCAD/A-B,SCADA/SC,SCADB/EXP,LOAD FE"
; 1842  FE_FE+P         "SCAD/A+B,SCADA/BYTE1,SCADB/FE, LOAD FE"
; 1843  FE_FE-200       "SCAD/A+B,SCADA/S#,S#/1600,SCADB/FE,LOAD FE"
; 1844  FE_-FE+200      "SCAD/A-B,SCADA/S#,S#/200,SCADB/FE,LOAD FE"
; 1845  FE_FE+S#        "SCAD/A+B,SCADA/S#,SCADB/FE,LOAD FE"
; 1846
; 1847
; 1848  GEN 17-FE       "SCAD/A-B,SCADA/S#,S#/210,SCADB/FE"
; 1849
```

```
; 1850  .TOC    "MACROS -- DATA PATH FIELD CONTROL"
; 1851
; 1852  HOLD LEFT        "CLKL/0,GENL/0"
; 1853  ADL PARITY       "GENL/1"
; 1854  CHK PARITY L     "CHKL/1"
; 1855
; 1856  HOLD RIGHT       "CLKR/0,GENR/0"
; 1857  ADR PARITY       "GENR/1"
; 1858  CHK PARITY R     "CHKR/1"
; 1859
; 1860  AD PARITY        "AD PARITY OK/1"
; 1861  CHK PARITY       "CHKL/1,CHKR/1"
; 1862  BAD PARITY       "CHKL/0,CHKR/0"
; 1863
; 1864  INH CRY18        "SPEC/INHCRY18"
; 1865
```

```
; 1866  .TOC    "MACROS -- SHIFT PATH CONTROL"
; 1867
; 1868  ASH             "SHSTYLE/NORM"  ;ASH SHIFT
; 1869  LSH             "SHSTYLE/NORM"  ;LSH SHIFT (SAME HARDWARE AS ASH BUT
; 1870                                  ; BITS -2 AND -1 ARE PRESET TO ZERO)
; 1871  ROT             "SHSTYLE/ROT"
; 1872  LSHC            "SHSTYLE/LSHC"
; 1873  ASHC            "SHSTYLE/ASHC"
; 1874  ROTC            "SHSTYLE/ROTC"
; 1875  ONES            "SHSTYLE/ONES"  ;SHIFT IN 1 BITS
; 1876  DIV             "SHSTYLE/DIV"   ;SPECIAL PATH FOR DIVIDE (LIKE ROTC BUT
; 1877                                  ; COMPLEMENT BIT AS IT GOES AROUND)
; 1878
```

```
; 1879  .TOC    "MACROS -- SPECIAL FUNCTIONS"
; 1880
; 1881  LOAD IR          "SPEC/LOADIR"    ;LOAD INSTRUCTION REG FROM
; 1882                                   ; DBUS0-DBUS8, LOAD AC# FROM
; 1883                                   ; DBUS9-DBUS12
; 1884                                   ; UPDATE LAST-INST-PUBLIC PC FLAG
; 1885  LOAD INST        "SPEC/LDINST"
; 1886  LOAD INST EA     "SPEC/LOADXR,PXCT/CURRENT"
; 1887  LOAD BYTE EA     "SPEC/LOADXR,PXCT/E2"
; 1888  LOAD IND EA      "SPEC/LOADXR,PXCT/E1"
; 1889  LOAD SRC EA      "SPEC/LOADXR,PXCT/BIS-SRC-EA"
; 1890  LOAD DST EA      "SPEC/LOADXR,PXCT/BIS-DST-EA"
; 1891  ADD .25          "CRY38/1"        ;GENERATE CARRY IN TO BIT 37
; 1892  CALL []          "CALL/1,J/@1"    ;CALL A SUBROUTINE
; 1893  LOAD PXCT        "SPEC/LDPXCT"    ;LOAD PXCT FLAGS IF EXEC MODE
; 1894  TURN OFF PXCT    "SPEC/PXCT OFF"
; 1895  LOAD PAGE TABLE  "SPEC/LDPAGE"
; 1896  LOAD AC BLOCKS   "SPEC/LDACBLK"
; 1897  SWEEP            "SPEC/SWEEP,PHYSICAL/1"
; 1898  CLRCSH           "SPEC/CLRCSH,PHYSICAL/1"
; 1899  LOAD PI          "SPEC/LDPI"
; 1900  SET HALT         "SPEC/#,#/74"
; 1901  CLEAR CONTINUE   "SPEC/#,#/40"
; 1902  CLEAR EXECUTE    "SPEC/#,#/20"
; 1903  CLEAR RUN        "SPEC/#,#/10"
; 1904  UNHALT           "SPEC/#,#/62"
; 1905  SET APR ENABLES  "SPEC/APR EN"
; 1906  ABORT MEM CYCLE  "DBUS/DBM,RAMADR/VMA,DBM/MEM,AD/ZERO,SPEC/MEMCLR,LOAD VMA"
; 1907  CLR IO BUSY      "SPEC/CLR IO BUSY"
; 1908  CLR IO LATCH     "SPEC/CLR IO LATCH"
; 1909
```

```
; 1910  .TOC    "MACROS -- PC FLAGS"
; 1911
; 1912  CHANGE FLAGS     "SPEC/FLAGS"
; 1913
; 1914  SET AROV         "CHANGE FLAGS, HOLD USER/1, SETOV/1, TRAP1/1"
; 1915  SET FOV          "CHANGE FLAGS, HOLD USER/1, SETFOV/1, TRAP1/1"
; 1916  SET NO DIVIDE    "CHANGE FLAGS, HOLD USER/1, SETOV/1, SETNDV/1, TRAP1/1"
; 1917  SET FL NO DIVIDE "SET NO DIVIDE, SETFOV/1"
; 1918
; 1919  ASH AROV         "SPEC/ASHOV"
; 1920  SET FPD          "CHANGE FLAGS, HOLD USER/1, SETFPD/1"
; 1921  CLR FPD          "CHANGE FLAGS, HOLD USER/1, CLRFPD/1"
; 1922
; 1923  SET PDL OV       "CHANGE FLAGS, HOLD USER/1, TRAP2/1"
; 1924  SET TRAP1        "CHANGE FLAGS, HOLD USER/1, TRAP1/1"
; 1925
; 1926  LOAD PCU         "CHANGE FLAGS, LD PCU/1"
; 1927  UPDATE USER      "CHANGE FLAGS, HOLD USER/1"
; 1928  LEAVE USER       "CHANGE FLAGS, HOLD USER/0"
; 1929
; 1930  JFCL FLAGS       "CHANGE FLAGS, HOLD USER/1, JFCLFLG/1"
; 1931
; 1932  LOAD FLAGS       "CHANGE FLAGS, LD FLAGS/1"
; 1933  EXP TEST         "SPEC/EXPTST"
; 1934  AD FLAGS         "CHANGE FLAGS, ADFLGS/1, HOLD USER/1"
; 1935
; 1936  NO DIVIDE        "SET NO DIVIDE, J/NIDISP"
; 1937  FL NO DIVIDE     "SET FL NO DIVIDE, J/NIDISP"
; 1938
```

```
; 1939  .TOC    "MACROS -- PAGE FAIL FLAGS"
; 1940
; 1941  STATE_[]        "[FLG]_#,STATE/@1,HOLD LEFT"
; 1942  END STATE       "[FLG]_0, HOLD LEFT"
; 1943
; 1944  END BLT         "END STATE"
; 1945  END MAP         "END STATE"
; 1946
```

```
; 1947  .TOC    "MACROS -- SINGLE SKIPS"
; 1948                                  ;SKIPS IF:
; 1949  SKIP IF ACO     "SKIP/ACO"      ;THE AC NUMBER IS ZERO
; 1950  SKIP DPO        "SKIP/DPO"      ;DP BIT 0=1
; 1951  SKIP DP18       "SKIP/DP18"     ;DP BIT 18=1
; 1952  SKIP AD.EQ.0    "SKIP/ADEQO,DT/3T" ;ADDER OUTPUT IS ZERO
; 1953  SKIP AD.LE.0    "SKIP/LE,DT/3T" ;ADDER OUTPUT IS LESS THAN OR EQUAL
; 1954                                  ; TO ZERO.
; 1955  SKIP ADL.LE.0   "SKIP/LLE,DT/3T" ;ADDER LEFT IS LESS THAN OR EQUAL
; 1956                                  ; TO ZERO.
; 1957  SKIP FPD        "SKIP/FPD"      ;FIRST-PART-DONE PC FLAG IS SET
; 1958  SKIP KERNEL     "SKIP/KERNEL"   ;USER=0
; 1959  SKIP IO LEGAL   "SKIP/IOLGL"    ;USER=0 OR USER IOT=1
; 1960  SKIP CRY0       "SKIP/CRY0"     ;ADDER BIT CRY0=1 (NOT PC FLAG BIT)
; 1961  SKIP CRY1       "SKIP/CRY1"     ;ADDER BIT CRY1=1 (NOT PC FLAG BIT)
; 1962  SKIP CRY2       "SKIP/CRY2,DT/3T"       ;ADDER BIT CRY2=1
; 1963  SKIP JFCL       "SKIP/JFCL"     ;IF JFCL SHOULD JUMP
; 1964  SKIP ADL.EQ.0   "SKIP/ADLEQO"   ;ALU BITS -2 TO 17 = 0
; 1965  SKIP ADR.EQ.0   "SKIP/ADREQO"   ;ALU BITS 18-35 = 0
; 1966  SKIP IRPT       "SKIP/INT"      ;INTERRUPT IS PENDING
; 1967  SKIP -1MS       "SKIP/-1 MS"    ;DON'T SKIP IF 1MS TIMER HAS EXPIRED.
; 1968  SKIP AC REF     "SKIP/ACREF"    ;VMA IS 0-17
; 1969  SKIP EXECUTE    "SKIP/EXECUTE"  ;CONSOLE EXECUTE
; 1970  TXXX TEST       "SKIP/TXXX"     ;TEST INSTRUCTION SHOULD SKIP
; 1971
```

```
; 1972  .TOC             "MACROS -- SPECIAL DISPATCH MACROS"
; 1973
; 1974  NEXT INST        "DISP/NICOND,SPEC/NICOND,J/NICOND"
; 1975  NEXT INST FETCH  "DISP/NICOND,SPEC/NICOND,J/NICOND-FETCH"
; 1976  EA MODE DISP     "DISP/EAMODE,RAMADR/XR#"
; 1977  AREAD            "DISP/AREAD,WAIT/1,AREAD/1,MEM/1,J/O"
; 1978  B DISP           "DISP/BDISP"
; 1979  BWRITE DISP      "B DISP,MEM/1,BWRITE/1,WRITE CYCLE/1,J/BWRITE"
; 1980  INST DISP        "DISP/DROM,J/O"
; 1981  EXIT             "BWRITE DISP,SPEC/O, WRITE TEST/1""
; 1982  AD FLAGS EXIT    "BWRITE DISP, WRITE TEST/O, AD FLAGS"
; 1983  FL-EXIT          "WRITE CYCLE/1,WRITE TEST/1,MEM/1,BWRITE/1,B DISP,J/FL-BWRITE"
; 1984  TEST DISP        "B DISP,J/TEST-TABLE"
; 1985  SKIP-COMP DISP   "B DISP,J/SKIP-COMP-TABLE"
; 1986  JUMP DISP        "B DISP,J/JUMP-TABLE"
; 1987  DONE             "VMA_[PC],LOAD VMA, FETCH, NEXT INST FETCH"
; 1988  JUMPA            "[PC]_[AR],HOLD LEFT,LOAD VMA,FETCH,NEXT INST FETCH"
; 1989  UUO              "[HR]_[HR].AND.#,#/777740,HOLD RIGHT,J/UUOGO"
; 1990  LUUO             "[AR]_O XWD [40], J/LUUO1"
; 1991  PAGE FAIL TRAP   "TL [FLG], FLG.PI/1, J/PFT"
; 1992  TAKE INTERRUPT   "[FLG]_[FLG].OR.#,FLG.PI/1,HOLD RIGHT,J/PI"
; 1993  INTERRUPT TRAP   "WORK[SV.AR]_[AR], J/ITRAP"
; 1994  MUL DISP         "DISP/MUL"
; 1995  DIV DISP         "DISP/DIV"
; 1996  BYTE DISP        "DISP/BYTE, DT/3T"
; 1997  SCAD DISP        "DISP/SCADO"    ;SKIP (2'S WEIGHT) IS SCAD IS MINUS
; 1998  RETURN []        "DISP/RETURN,J/@1"
; 1999  PI DISP          "DISP/PI"
; 2000  NORM DISP        "DISP/NORM,DT/3T"
; 2001  DISMISS          "TR [PI], #/077400, CALL [JEN1],DT/3T"
; 2002  CALL LOAD PI     "[TO]_[PI] SWAP, CALL [LDPI2]"
; 2003  HALT []          "AD/47,DEST/AD,B/T1,DBM/#,DBUS/DBM,HALT/@1,RSRC/DA,A/MASK, J/HALTED"
; 2004  CLEANUP DISP     "READ [FLG], DBUS/DP, DISP/DP, 3T, J/CLEANUP"
; 2005
```

```
; 1947  .TOC    "MACROS -- SINGLE SKIPS"
; 1948
; 1949  SKIP IF ACO      "SKIP/ACO"       ;SKIPS IF:
; 1950  SKIP DPO         "SKIP/DPO"       ;THE AC NUMBER IS ZERO
; 1951  SKIP DP18        "SKIP/DP18"      ;DP BIT 0=1
; 1952  SKIP AD.EQ.O     "SKIP/ADEQO,DT/3T" ;DP BIT 18=1
; 1953  SKIP AD.LE.O     "SKIP/LE,DT/3T"  ;ADDER OUTPUT IS ZERO
; 1954                                    ;ADDER OUTPUT IS LESS THAN OR EQUAL
; 1955  SKIP ADL.LE.O    "SKIP/LLE,DT/3T" ; TO ZERO.
; 1956                                    ;ADDER LEFT IS LESS THAN OR EQUAL
; 1957  SKIP FPD         "SKIP/FPD"       ; TO ZERO.
; 1958  SKIP KERNEL      "SKIP/KERNEL"    ;FIRST-PART-DONE PC FLAG IS SET
; 1959  SKIP IO LEGAL    "SKIP/IOLGL"     ;USER=0
; 1960  SKIP CRY0        "SKIP/CRY0"      ;USER=0 OR USER IOT=1
; 1961  SKIP CRY1        "SKIP/CRY1"      ;ADDER BIT CRY0=1 (NOT PC FLAG BIT)
; 1962  SKIP CRY2        "SKIP/CRY2,DT/3T"       ;ADDER BIT CRY1=1 (NOT PC FLAG BIT)
; 1963  SKIP JFCL        "SKIP/JFCL"      ;ADDER BIT CRY2=1
; 1964  SKIP ADL.EQ.O    "SKIP/ADLEQO"    ;IF JFCL SHOULD JUMP
; 1965  SKIP ADR.EQ.O    "SKIP/ADREQO"    ;ALU BITS -2 TO 17 = 0
; 1966  SKIP IRPT        "SKIP/INT"       ;ALU BITS 18-35 = 0
; 1967  SKIP -1MS        "SKIP/-1 MS"     ;INTERRUPT IS PENDING
; 1968  SKIP AC REF      "SKIP/ACREF"     ;DON'T SKIP IF 1MS TIMER HAS EXPIRED.
; 1969  SKIP EXECUTE     "SKIP/EXECUTE"   ;VMA IS 0-17
; 1970  TXXX TEST        "SKIP/TXXX"      ;CONSOLE EXECUTE
; 1971                                    ;TEST INSTRUCTION SHOULD SKIP
```

```
; 1972  .TOC              "MACROS -- SPECIAL DISPATCH MACROS"
; 1973
; 1974  NEXT INST         "DISP/NICOND,SPEC/NICOND,J/NICOND"
; 1975  NEXT INST FETCH   "DISP/NICOND,SPEC/NICOND,J/NICOND-FETCH"
; 1976  EA MODE DISP      "DISP/EAMODE,RAMADR/XR#"
; 1977  AREAD             "DISP/AREAD,WAIT/1,AREAD/1,MEM/1,J/O"
; 1978  B DISP            "DISP/BDISP"
; 1979  BWRITE DISP       "B DISP,MEM/1,BWRITE/1,WRITE CYCLE/1,J/BWRITE"
; 1980  INST DISP         "DISP/DROM,J/O"
; 1981  EXIT              "BWRITE DISP,SPEC/O, WRITE TEST/1""
; 1982  AD FLAGS EXIT     "BWRITE DISP, WRITE TEST/O, AD FLAGS"
; 1983  FL-EXIT           "WRITE CYCLE/1,WRITE TEST/1,MEM/1,BWRITE/1,B DISP,J/FL-BWRITE"
; 1984  TEST DISP         "B DISP,J/TEST-TABLE"
; 1985  SKIP-COMP DISP    "B DISP,J/SKIP-COMP-TABLE"
; 1986  JUMP DISP         "B DISP,J/JUMP-TABLE"
; 1987  DONE              "VMA_[PC],LOAD VMA, FETCH, NEXT INST FETCH"
; 1988  JUMPA             "[PC]_[AR],HOLD LEFT,LOAD VMA,FETCH,NEXT INST FETCH"
; 1989  UUO               "[HR]_[HR].AND.#,#/777740,HOLD RIGHT,J/UUOGO"
; 1990  LUUO              "[AR]_O XWD [40], J/LUUO1"
; 1991  PAGE FAIL TRAP    "TL [FLG], FLG.PI/1, J/PFT"
; 1992  TAKE INTERRUPT    "[FLG]_[FLG].OR.#,FLG.PI/1,HOLD RIGHT,J/PI"
; 1993  INTERRUPT TRAP    "WORK[SV.AR]_[AR], J/ITRAP"
; 1994  MUL DISP          "DISP/MUL"
; 1995  DIV DISP          "DISP/DIV"
; 1996  BYTE DISP         "DISP/BYTE, DT/3T"
; 1997  SCAD DISP         "DISP/SCADO"    ;SKIP (2'S WEIGHT) IS SCAD IS MINUS
; 1998  RETURN []         "DISP/RETURN,J/@1"
; 1999  PI DISP           "DISP/PI"
; 2000  NORM DISP         "DISP/NORM,DT/3T"
; 2001  DISMISS           "TR [PI], #/077400, CALL [JEN1],DT/3T"
; 2002  CALL LOAD PI      "[TO]_[PI] SWAP, CALL [LDPI2]"
; 2003  HALT []           "AD/47,DEST/AD,B/T1,DBM/#,DBUS/DBM,HALT/@1,RSRC/DA,A/MASK, J/HALTED"
; 2004  CLEANUP DISP      "READ [FLG], DBUS/DP, DISP/DP, 3T, J/CLEANUP"
; 2005
```

```
; 2006  .TOC            "DISPATCH ROM MACROS"
; 2007          .DCODE
; 2008
; 2009  ;"A FIELD" MACROS SAY HOW TO FETCH ARGUMENTS
; 2010
; 2011  I       "I/1"
; 2012  I-PF    "I/1,VMA/O,READ/1"
; 2013  R       "A/READ,READ/1"
; 2014  R-PF    "A/RD-PF,READ/1"
; 2015  W       "A/WRITE,TEST/1"
; 2016  RW      "A/READ,TEST/1,READ/1"
; 2017  IW      "I/1,TEST/1"    ;IMMED WHICH STORE IN E. (SETZM, ETC.)
; 2018  IR      "I/1,READ/1"    ;START READ A GO TO EXECUTE CODE
; 2019  DBL R   "A/DREAD,READ/1"        ;AR!ARX _ E!E+1
; 2020  DBL AC  "A/DBLAC"
; 2021  SH      "A/SHIFT,VMA/O,READ/1"
; 2022  SHC     "A/DSHIFT,VMA/O,READ/1"
; 2023  FL-R    "A/FP,READ/1"   ;FLOATING POINT READ
; 2024  FL-RW   "A/FP,READ/1,TEST/1"
; 2025  FL-I    "A/FPI,READ/O"  ;FLOATING POINT IMMEDIATE
; 2026  DBL FL-R "A/DFP,READ/1"
; 2027  IOT     "A/IOT"         ;CHECK FOR IO LEGAL
; 2028
; 2029  ;"B FIELD" MACROS SAY HOW TO STORE RESULTS
; 2030
; 2031  AC      "B/AC"
; 2032  M       "B/MEM,TEST/1,COND FUNC/1"
; 2033  B       "B/BOTH,TEST/1,COND FUNC/1"
; 2034  S       "B/SELF,TEST/1,COND FUNC/1"
; 2035  DAC     "B/DBLAC"
; 2036  DBL B   "B/DBLB,TEST/1,COND FUNC/1"
; 2037  FL-AC   "FL-B/AC"                      ;FLOATING POINT
; 2038  FL-MEM  "FL-B/MEM,TEST/1,COND FUNC/1"  ;FLOATING POINT TO MEMORY
; 2039  FL-BOTH "FL-B/BOTH,TEST/1,COND FUNC/1" ;FLOATING POINT TO BOTH
; 2040  ROUND   "ROUND/1"                      ;FLOATING POINT ROUNDED
; 2041
; 2042
; 2043  ;CONTROL BITS
; 2044  W TEST  "TEST/1"
; 2045  AC DISP "ACDISP/1"
; 2046          .UCODE
; 2047
```

```
U 0000, 0002,3771,0012,4374,4007,0700,0000,0037,7777
U 0002, 0013,3445,1212,4174,4007,0700,0000,0000,0000
U 0013, 0053,3551,1212,4374,4007,0700,0000,0000,0001
U 0053, 0061,3447,1200,4174,4007,0700,0000,0000,0000
U 0061, 0071,3771,0015,4374,4007,0700,0000,0000,0001


U 0071, 0003,4751,1207,4374,4007,0700,0010,0000,0001
U 0003, 0100,4751,1203,4374,4007,0700,0000,0037,6000
U 0100, 0106,3333,0003,7174,4007,0700,0400,0000,0227
U 0106, 0110,4221,0011,4364,4277,0700,0200,0000,0010
U 0110, 0125,4221,0010,4174,4477,0700,0000,0000,0000


U 0125, 0131,4221,0013,4174,4257,0700,0000,0000,0000
U 0131, 0136,3333,0013,7174,4007,0700,0400,0000,0230
U 0136, 0162,3333,0013,7174,4007,0700,0400,0000,0300
U 0162, 0212,3333,0013,7174,4007,0700,0400,0000,0301
U 0212, 0214,4223,0000,1174,4007,0700,0400,0000,1443
U 0214, 0223,3333,0007,1174,4007,0700,0400,0000,1444
U 0223, 0226,4221,0003,4174,4007,0700,2000,0071,0023
U 0226, 0235,3333,0007,7174,4007,0700,0400,0000,0344
U 0235, 0242,4223,0000,7174,4007,0700,0400,0000,0373
U 0242, 0244,3771,0002,4374,4007,0700,0000,0000,0344
U 0244, 0311,3771,0006,4374,4007,0700,0000,0000,0373
U 0311, 0010,0111,0706,4174,4007,0700,0200,0000,0010
U 0010, 0560,3771,0004,1276,6007,0701,0010,0000,1444
U 0012, 0323,3333,0005,6174,4007,0700,0400,0000,0000
U 0323, 0324,0111,0702,4174,4007,0700,0200,0000,0010
U 0324, 0224,3333,0004,6174,4007,0630,2400,0060,0000
U 0224, 0311,4443,0000,4174,4007,0700,0000,0000,0000
U 0225, 0140,6553,0500,4374,4007,0321,0000,0033,0656


U 0140, 0104,4751,1217,4374,4007,0700,0000,0000,1005
```

```
; 2048          .BIN
; 2049  .TOC     "POWER UP SEQUENCE"
; 2050
; 2051          .UCODE
; 2052
; 2053  ;HERE IS WHERE WE FIRE THE MACHINE UP DURING POWER ON
; 2054
; 2055
; 2056  0:       [MASK]_#, #/377777          ;BUILD A MASK WITH
; 2057          [MASK]_[MASK]*2             ; A ONE IN 36-BITS AND O
; 2058          [MASK]_[MASK].OR.#,#/1      ; IN BITS -2,-1,36,37
; 2059          [MAG]_[MASK]*.5             ;MAKE CONSTANT
; 2060          [XWD1]_#, #/1               ;CONSTANT WITH 1 IN EACH
; 2061                                      ; HALF WORD
; 2062          [ONE]_0 XWD [1],            ;THE CONSTANT 1
; 2063          CALL/1                      ;RESET STACK (CAN NEVER RETURN
; 2064                                      ; TO WHERE MR LEFT US)
; 2065  3:       [AR]_0 XWD [376000]         ;ADDRESS OF HALT STATUS
; 2066                                      ; BLOCK
; 2067          WORK[HSBADR]_[AR]           ;SAVE FOR HALT LOOP
; 2068          [UBR]_0, ABORT MEM CYCLE    ;CLEAR THE UBR AND RESET
; 2069                                      ; MEMORY CONTROL LOGIC
; 2070          [EBR]_0, LOAD AC BLOCKS     ;CLEAR THE EBR AND FORCE
; 2071                                      ; PREVIOUS AND CURRENT AC
; 2072                                      ; BLOCKS TO ZERO
; 2073          [FLG]_0, SET APR ENABLES    ;CLEAR THE STATUS FLAGS AND
; 2074                                      ; DISABLE ALL APR CONDITIONS
; 2075          WORK[APR]_[FLG]             ;ZERO REMEMBERED ENABLES
; 2076
; 2077          WORK[TIME0]_[FLG]           ;CLEAR TIME BASE
; 2078          WORK[TIME1]_[FLG]           ; ..
; 2079  .IF/FULL
; 2080          AC[BIN0]_0                  ;COMPUTE A TABLE OF POWERS OF
; 2081          AC[BIN1]_1                  ; TEN
; 2082          [AR]_0, SC_19.              ;WE WANT TO GET 22 NUMBERS
; 2083          WORK[DECLO]_1               ;STARTING WITH 1
; 2084          WORK[DECHI]_0               ; ..
; 2085          [HR]_#, WORK/DECLO          ;ADDRESS OF LOW WORD
; 2086          [BRX]_#, WORK/DECHI         ;ADDRESS OF HIGH WORD
; 2087  TENLP:  [BRX]_[BRX]+1, LOAD VMA     ;ADDRESS THE HIGH WORD
; 2088  =0*     [ARX]_AC[BIN1],             ;LOW WORD TO ARX
; 2089          CALL [DBSLOW]               ;MULTIPLY BY TEN
; 2090          RAM_[BR]                    ;SAVE HIGH WORD
; 2091          [HR]_[HR]+1, LOAD VMA       ;WHERE TO STORE LOW WORD
; 2092          RAM_[ARX], STEP SC          ;STORE LOW WORD AND SEE IF
; 2093                                      ; WE ARE DONE
; 2094  =0       J/TENLP                     ;NOT YET--KEEP GOING
; 2095          [BR].XOR.#, 3T, SKIP ADL.EQ.0, #/330656
; 2096                                      ;DID WE GET THE RIGHT ANSWER
; 2097                                      ; IN THE TOP 18 BITS?
; 2098  =0**0    HALT [MULERR]               ;NO--CPU IS BROKEN
; 2099  .ENDIF/FULL
; 2100
```

```
U 0141, 3610,4221,0014,4174,4007,0700,0010,0000,0000    ; 2101  =0**1   [PI]_0, CALL [LOADPI]   ;CLEAR PI STATE
                                                        ; 2102  =1**1                          ;CLEAR REGISTERS SO NO
                                                        ; 2103                                 ;PARITY ERROR HAPPEN
                                                        ;;2104  .IFNOT/FULL
                                                        ;;2105          [ARX]_0                ;WRITTEN WHILE COMPUTING POWERS
                                                        ;;2106          [BR]_0                 ;OF 10
                                                        ;;2107          [BRX]_0
                                                        ; 2108  .ENDIF/FULL
U 0151, 0334,4751,1217,4374,4007,0700,0000,0000,0120    ; 2109          [T1]_0 XWD [120]       ;RH OF 120 CONTAINS START ADDRESS
                                                        ; 2110                                 ; FOR SIMULATOR. FOR THE REAL
                                                        ; 2111                                 ; MACHINE IT IS JUST DATA WITH
                                                        ; 2112                                 ; GOOD PARITY.
                                                        ; 2113  =
                                                        ; 2114  ;THE CODE UNDER .IF/SIM MUST USE THE SAME ADDRESS AS THE CODE
                                                        ; 2115  ; UNDER .IFNOT/SIM SO THAT MICROCODE ADDRESSES DO NOT CHANGE BETWEEN
                                                        ; 2116  ; VERSIONS
                                                        ;;2117  .IF/SIM
                                                        ;;2118          VMA_[T1], START READ   ;READ THE WORD
                                                        ;;2119          MEM READ, [PC]_MEM, HOLD LEFT, J/START
                                                        ;;2120                                 ;GO FIRE UP SIMULATOR AT THE
                                                        ;;2121                                 ; PROGRAMS STARTING ADDRESS
                                                        ; 2122  .IFNOT/SIM
                                                        ; 2123          [PC]_0,                ;CLEAR LH OF PC
                                                        ; 2124          LEAVE USER,            ;ENTER EXEC MODE
U 0334, 0343,4221,0001,4174,4467,0700,0000,0000,0004    ; 2125          LOAD FLAGS             ;CLEAR TRAP FLAGS
                                                        ; 2126          [T1]_#, HALT/POWER,    ;LOAD T1 WITH POWER UP CODE
U 0343, 0116,3771,0017,4374,4007,0700,0000,0000,0000    ; 2127          J/PWRON                ;ENTER HALT LOOP. DO NOT STORE
                                                        ; 2128                                 ; HALT STATUS BLOCK
                                                        ; 2129  .ENDIF/SIM
                                                        ; 2130
```

```
U 1400, 0110,3443,0100,4174,4156,4700,0200,0014,0012
U 1401, 0110,0111,0701,4170,4156,4700,0200,0014,0012

U 0260, 0110,0111,0701,4170,4156,4700,0200,0014,0012
U 0261, 0110,3443,0100,4174,4156,4700,0200,0014,0012

U 0101, 3475,4751,1203,4374,4367,0700,0000,0000,0423

U 0102, 3475,4751,1203,4374,4367,0700,0000,0000,0422

U 0103, 3475,4751,1203,4374,4367,0700,0000,0000,0421
U 0105, 0104,4751,1217,4374,4007,0700,0000,0000,0002

U 0107, 0117,3443,0100,4174,4007,0700,0200,0014,0012

U 0111, 3475,4751,1203,4374,4367,0700,0000,0000,0423

U 0112, 3475,4751,1203,4374,4367,0700,0000,0000,0422

U 0113, 3475,4751,1203,4374,4367,0700,0000,0000,0421
U 0115, 0104,4751,1217,4374,4007,0700,0000,0000,0002

U 0117, 0346,3771,0002,4365,5617,0700,0200,0000,0002
```

```
; 2131  .TOC    "THE INSTRUCTION LOOP -- START NEXT INSTRUCTION"
; 2132
; 2133  ;ALL INSTRUCTIONS EXCEPT JUMP'S AND UUO'S END UP HERE
; 2134  1400:
; 2135  DONE:   DONE
; 2136  1401:   VMA_[PC]+1, NEXT INST FETCH, FETCH
; 2137  =0
; 2138  SKIP:   VMA_[PC]+1, NEXT INST FETCH, FETCH
; 2139          DONE
; 2140
; 2141
; 2142  ;16-WAY DISPATCH BASED ON NEXT INSTRUCTION
; 2143  =0000
; 2144  NICOND:
; 2145  =0001   [AR]_0 XWD [423],        ;TRAP TYPE 3
; 2146                                   ; GET ADDRESS OF TRAP INST
; 2147          TURN OFF PXCT,           ;CLEAR PXCT
; 2148          J/TRAP                   ;PROCESS TRAP (INOUT.MIC)
; 2149  =0010   [AR]_0 XWD [422],        ;TRAP TYPE 2
; 2150          TURN OFF PXCT,           ;CLEAR PXCT
; 2151          J/TRAP                   ;GO TRAP
; 2152  =0011   [AR]_0 XWD [421],        ;TRAP TYPE 1
; 2153          TURN OFF PXCT,           ;TURN OF PXCT
; 2154          J/TRAP                   ;GO TRAP
; 2155  =0101   HALT [CSL]               ;"HA" COMMAND TO 8080
; 2156  =0111
; 2157  START:  VMA_[PC],                ;LOAD VMA
; 2158          FETCH,                   ;INDICATE INSTRUCTION FETCH
; 2159          J/XCTGO                  ;GO GET INSTRUCTION
; 2160  ;THE NEXT SET OF CASES ARE USED WHEN THERE IS A FETCH
; 2161  ; IN PROGESS
; 2162  =1000
; 2163  NICOND-FETCH:
; 2164  =1001   [AR]_0 XWD [423],        ;TRAP TYPE 3
; 2165          TURN OFF PXCT,
; 2166          J/TRAP
; 2167  =1010   [AR]_0 XWD [422],        ;TRAP TYPE 2
; 2168          TURN OFF PXCT,
; 2169          J/TRAP
; 2170  =1011   [AR]_0 XWD [421],        ;TRAP TYPE 1
; 2171          TURN OFF PXCT,
; 2172          J/TRAP
; 2173  =1101   HALT [CSL]               ;"HA" COMMAND TO 8080
; 2174  =1111
; 2175  XCTGO:  MEM READ,                ;WAIT FOR MEMORY
; 2176          [HR]_MEM,                ;PUT DATA IN HR
; 2177          LOAD INST,               ;LOAD IR & AC #
; 2178          J/INCPC                  ;GO BUMP PC
; 2179  =
; 2180
```

```
                                                ; 2181  ;HERE WE POINT PC TO NEXT INSTRUCTION WHILE WE WAIT FOR
                                                ; 2182  ; EFFECTIVE ADDRESS LOGIC TO SETTLE
                                                ; 2183  INCPC:  VMA_[PC]+1,             ;ADDRESS OF NEXT INSTRUCTION
                                                ; 2184          FETCH/1,               ;INSTRUCTION FETCH
                                                ; 2185          TURN OFF PXCT,         ;CLEAR EFFECT OF PXCT
U 0346, 0201,0111,0701,2170,4366,6700,0200,0010,0010  ; 2186          EA MODE DISP           ;DISPACTH OF INDEXING AND @
                                                ; 2187
                                                ; 2188  ;MAIN EFFECTIVE ADDRESS CALCULATION
                                                ; 2189  =0001
                                                ; 2190  EACALC:
                                                ; 2191  ;
                                                ; 2192  ;       THE FIRST 4 CASES ARE USED ONLY FOR JRST
                                                ; 2193  ;
                                                ; 2194
                                                ; 2195  ;CASE 0 -- JRST 0,FOO(XR)
                                                ; 2196          [PC]_[HR]+XR,          ;UPDATE PC
                                                ; 2197          HOLD LEFT,             ;ONLY RH
                                                ; 2198          LOAD VMA, FETCH,       ;START GETTING IT
U 0201, 0110,0551,0201,2270,4156,4700,0200,0014,0012  ; 2199          NEXT INST FETCH        ;START NEXT INST
                                                ; 2200
                                                ; 2201  ;CASE 2 -- JRST 0,FOO
                                                ; 2202          [PC]_[HR],             ;NEW PC
                                                ; 2203          HOLD LEFT,             ;ONLY RH
                                                ; 2204          LOAD VMA, FETCH,       ;START GETTING IT
U 0203, 0110,3441,0201,4170,4156,4700,0200,0014,0012  ; 2205          NEXT INST FETCH        ;START NEXT INST
                                                ; 2206
                                                ; 2207  ;CASE 4 -- JRST 0,@FOO(XR)
                                                ; 2208          [HR]_[HR]+XR,          ;ADD IN INDEX
                                                ; 2209          START READ,            ;START TO FETCH @ WORD
                                                ; 2210          LOAD VMA,              ;PUT ADDRESS IN VMA
U 0205, 0363,0551,0202,2270,4007,0700,0200,0004,0012  ; 2211          J/FETIND               ;GO DO MEM WAIT (FORGET ABOUT JRST)
                                                ; 2212
                                                ; 2213  ;CASE 6 -- JRST 0,@FOO
                                                ; 2214          VMA_[HR],              ;LOAD UP ADDRESS
                                                ; 2215          START READ,            ;START TO FETCH @ WORD
U 0207, 0363,3443,0200,4174,4007,0700,0200,0004,0012  ; 2216          J/FETIND               ;GO DO MEM WAIT (FORGET ABOUT JRST)
                                                ; 2217
```

```
U 0211, 0213,0551,0202,2270,4007,0700,0000,0000,0000

U 0213, 0000,5741,0203,4174,4001,3700,0200,0000,0342

U 0215, 0363,0551,0202,2270,4007,0700,0200,0004,0112

U 0217, 0363,3443,0200,4174,4007,0700,0200,0004,0112

U 0363, 0366,3771,0002,4361,5217,0700,0200,0000,0102

U 0366, 0201,3443,0100,2174,4006,6700,0200,0010,0010
```

```
; 2218  ;
; 2219  ;THESE 4 ARE FOR THE NON-JRST CASE
; 2220  ;
; 2221
; 2222  ;CASE 10 -- JUST INDEXING
; 2223  INDEX:  [HR]_[HR]+XR,           ;ADD IN INDEX REGISTER
; 2224          HOLD LEFT               ;JUST DO RIGHT HALF
; 2225
; 2226  ;CASE 12 -- NO INDEXING OR INDIRECT
; 2227  NOMOD:  [AR]_EA,                ;PUT 0,,E IN AR
; 2228          PXCT DATA, AREAD        ;DO ONE OR MORE OF THE FOLLWING
; 2229                                  ; ACCORDING TO THE DROM:
; 2230                                  ;1. LOAD VMA
; 2231                                  ;2. START READ OR WRITE
; 2232                                  ;3. DISPATCH TO 40-57
; 2233                                  ;   OR DIRECTLY TO EXECUTE CODE
; 2234
; 2235  ;CASE 14 -- BOTH INDEXING AND INDIRECT
; 2236  BOTH:   [HR]_[HR]+XR,           ;ADD IN INDEX REGISTER
; 2237          LOAD VMA, PXCT EA,      ;PUT ADDRESS IN VMA
; 2238          START READ, J/FETIND    ;START CYCLE AND GO WAIT FOR DATA
; 2239
; 2240  ;CASE 16 -- JUST INDIRECT
; 2241  INDRCT: VMA_[HR],               ;LOAD ADDRESS OF @ WORD
; 2242          START READ, PXCT EA     ;START CYCLE
; 2243
; 2244
; 2245  ;HERE TO FETCH INDIRECT WORD
; 2246  FETIND: MEM READ, [HR]_MEM,     ;GET DATA WORD
; 2247          HOLD LEFT,              ;JUST RIGHT HALF
; 2248          LOAD IND EA             ;RELOAD @ AND INDEX FLOPS
; 2249
; 2250  XCT2:   VMA_[PC],               ;PUT PC BACK IN VMA
; 2251          FETCH/1,                ;TURN ON FETCH FLAG
; 2252          EA MODE DISP,           ;REDO CALCULATION FOR
; 2253          J/EACALC                ; NEW WORD
; 2254
```

```
U 0040, 0000,3771,0003,4365,5001,2700,0200,0000,0002
U 0041, 0000,3771,0003,0276,6001,2700,0000,0000,0000
U 0042, 0371,3771,0003,4365,5007,0700,0200,0000,0002
U 0371, 0401,0111,0702,4170,4007,0700,0200,0004,0312
U 0401, 0000,3771,0004,4365,5001,2700,0200,0000,0002
U 0043, 0406,3771,0003,0276,6007,0700,0000,0000,0000
U 0406, 0000,3771,0004,1276,6001,2701,0000,0000,1441
U 0044, 0000,3333,0003,4174,4001,2530,3000,0041,5777
U 0045, 0415,3772,0000,1275,5007,0701,0000,0000,1441
U 0415, 0416,3776,0005,0274,4007,0701,0000,0000,0000
U 0416, 0044,3446,0505,4174,4007,0700,0000,0000,0000
```

```
; 2255  .TOC    "THE INSTRUCTION LOOP -- FETCH ARGUMENTS"
; 2256  ;HERE ON AREAD DISP TO HANDLE VARIOUS CASES OF ARGUMENT FETCH
; 2257
; 2258  ;CASE 0 -- READ (E)
; 2259  40:     MEM READ,                ;WAIT FOR DATA
; 2260          [AR]_MEM,                ;PUT WORD IN AR
; 2261          INST DISP                ;GO TO EXECUTE CODE
; 2262
; 2263  ;CASE 1 -- WRITE (E)
; 2264  41:     [AR]_AC,                 ;PUT AC IN AR
; 2265          INST DISP                ;GO TO EXECUTE CODE
; 2266
; 2267  ;CASE 2 -- DOUBLE READ
; 2268  42:     MEM READ,                ;WAIT FOR DATA
; 2269          [AR]_MEM                 ;PUT HI WORD IN AR
; 2270          VMA_[HR]+1, PXCT DATA,   ;POINT TO E+1
; 2271          START READ               ;START MEMORY CYCLE
; 2272          MEM READ,                ;WAIT FOR DATA
; 2273          [ARX]_MEM,               ;LOW WORD IN ARX
; 2274          INST DISP                ;GO TO EXECUTE CODE
; 2275
; 2276  ;CASE 3 -- DOUBLE AC
; 2277  43:     [AR]_AC                  ;GET HIGH AC
; 2278          [ARX]_AC[1],             ;PUT C(AC+1) IN ARX
; 2279          INST DISP                ;GO TO EXECUTE CODE
; 2280
; 2281  ;CASE 4 -- SHIFT
; 2282  44:
; 2283  SHIFT:  READ [AR],                ;LOOK AT EFFECTIVE ADDRESS
; 2284          SKIP DP18,               ;SEE IF LEFT OR RIGHT
; 2285          SC_SHIFT-1,              ;PUT NUMBER OF PLACES TO SHIFT IN
; 2286          LOAD FE,                 ; SC AND FE
; 2287          INST DISP                ;GO DO THE SHIFT
; 2288
; 2289  ;CASE 5 -- SHIFT COMBINED
; 2290  45:     Q_AC[1]                  ;PUT LOW WORD IN Q
; 2291          [BR]_AC*.5 LONG          ;PUT AC IN BR & SHIFT BR!Q RIGHT
; 2292          [BR]_[BR]*.5 LONG,       ;SHIFT BR!Q 1 MORE PLACE RIGHT
; 2293          J/SHIFT                  ;GO DO SHIFT SETUP
; 2294
```

```
U 0046, 0372,3770,0303,4344,4007,0700,0000,0000,0000    ; 2295  ;CASE 6 -- FLOATING POINT IMMEDIATE
                                                        ; 2296  46:     [AR]_[AR] SWAP,          ;FLIP BITS TO LEFT HALF
                                                        ; 2297          J/FPRO                   ;JOIN COMMON F.P. CODE
                                                        ; 2298
                                                        ; 2299  ;CASE 7 -- FLOATING POINT
                                                        ; 2300  47:     MEM READ,                ;WAIT FOR MEMORY (SPEC/MEM WAIT)
U 0047, 0372,3771,0003,4365,5007,0700,0200,0000,0002    ; 2301          [AR]_MEM                 ;DATA INTO AR
                                                        ; 2302  =O
                                                        ; 2303  FPRO:   READ [AR],               ;LOOK AT NUMBER
                                                        ; 2304          SC_EXP, FE_EXP,          ;PUT EXPONENT IN SC & FE
                                                        ; 2305          SKIP DPO,                ;SEE IF NEGATIVE
U 0372, 0412,3333,0003,4174,4007,0520,3010,0041,2000    ; 2306          CALL [ARSIGN]            ;EXTEND AR SIGN
                                                        ; 2307  FPR1:   [ARX]_0,                 ;ZERO ARX
U 0373, 0000,4221,0004,4174,4001,2700,0000,0000,0000    ; 2308          INST DISP                ;GO TO EXECUTE CODE
                                                        ; 2309
                                                        ; 2310  ;CASE 10 -- READ THEN PREFETCH
                                                        ; 2311  50:     MEM READ,                ;WAIT FOR DATA
                                                        ; 2312          [AR]_MEM THEN FETCH,     ;PUT DATA IN AR AND START A READ
                                                        ; 2313                                   ; VMA HAS PC+1.
U 0050, 0000,3770,0103,4365,5001,2700,0200,0014,0012    ; 2314          INST DISP                ;GO DO IT
                                                        ; 2315
                                                        ; 2316  ;CASE 11 -- DOUBLE FLOATING READ
                                                        ; 2317  51:     SPEC MEM READ,           ;WAIT FOR DATA
                                                        ; 2318          [BR]_MEM,                ;HOLD IN BR
                                                        ; 2319          SC_EXP, FE_EXP,          ;SAVE EXPONENT
U 0051, 0402,3771,0005,4365,5177,0521,3000,0041,2000    ; 2320          SKIP DPO, 3T             ;SEE IF MINUS
                                                        ; 2321  =O      [AR]_[AR]+1,             ;POINT TO E+1
                                                        ; 2322          LOAD VMA, PXCT DATA,     ;PUT IN VMA
U 0402, 0432,0111,0703,4174,4007,0700,0200,0004,0312    ; 2323          START READ, J/DFPR1       ;GO GET POSITIVE DATA
                                                        ; 2324          [AR]_[AR]+1,             ;POINT TO E+1
                                                        ; 2325          LOAD VMA, PXCT DATA,     ;PUT IN VMA
U 0403, 0431,0111,0703,4174,4007,0700,0200,0004,0312    ; 2326          START READ               ;GO GET NEGATIVE DATA
                                                        ; 2327          [BR]_-SIGN,              ;SMEAR MINUS SIGN
U 0431, 0445,3551,0505,4374,0007,0700,0000,0077,7000    ; 2328          J/DFPR2                  ;CONTINUE BELOW
U 0432, 0445,4551,0505,4374,0007,0700,0000,0000,0777    ; 2329  DFPR1:  [BR]_+SIGN               ;SMEAR PLUS SIGN
                                                        ; 2330  DFPR2:  MEM READ, 3T,            ;WAIT FOR MEMORY
                                                        ; 2331          [ARX]_(MEM.AND.[MAG])*.5,
U 0445, 0451,4557,0004,4365,5007,0701,0200,0000,0002    ; 2332          ASH                      ;SET SHIFT PATHS
U 0451, 0452,3447,0503,4174,4007,0700,0000,0000,0000    ; 2333          [AR]_[BR]*.5             ;SHIFT AR
                                                        ; 2334          [AR]_[AR]*.5,            ;COMPLETE SHIFTING
U 0452, 0467,3447,0303,4174,4007,0700,2000,0011,0000    ; 2335          SC_FE                    ;PAGE FAIL MAY HAVE ZAPPED
                                                        ; 2336                                   ; THE SC.
                                                        ; 2337          VMA_[PC], FETCH,         ;GET NEXT INST
U 0467, 0000,3443,0100,4174,4001,2700,0200,0014,0012    ; 2338          INST DISP                ;DO THIS ONE
                                                        ; 2339
```

```
U 0052, 0404,4443,0000,4174,4007,0040,0000,0000,0000    ; 2340  ;CASE 12 -- TEST FOR IO LEGAL
U 0404, 2740,4551,0202,4374,0007,0700,0000,0077,7740    ; 2341  52:     SKIP IO LEGAL           ;IS IO LEGAL?
U 0405, 0000,4443,0000,4174,4001,2700,0000,0000,0000    ; 2342  =O      UUO                     ;NO
                                                        ; 2343          INST DISP               ;YES--DO THE INSTRUCTION
                                                        ; 2344
                                                        ; 2345
                                                        ; 2346  ;CASE 13 -- RESERVED
                                                        ; 2347  ;53:
                                                        ; 2348
                                                        ; 2349  ;CASE 14 -- RESERVED
                                                        ; 2350  ;54:
                                                        ; 2351
                                                        ; 2352  ;CASE 15 -- RESERVED
                                                        ; 2353  ;55:
                                                        ; 2354
                                                        ; 2355  ;CASE 16 -- RESERVED
                                                        ; 2356  ;56:
                                                        ; 2357
                                                        ; 2358  ;CASE 17 -- RESERVED
                                                        ; 2359  ;57:
                                                        ; 2360
                                                        ; 2361  ;EXTEND AR SIGN.
                                                        ; 2362  ;CALL WITH SKIP ON ARO, RETURNS 1 ALWAYS
                                                        ; 2363  =O
U 0412, 0001,4551,0303,4374,0004,1700,0000,0000,0777    ; 2364  ARSIGN: [AR]_+SIGN, RETURN [1]  ;EXTEND + SIGN
U 0413, 0001,3551,0303,4374,0004,1700,0000,0077,7000    ; 2365          [AR]_-SIGN, RETURN [1]  ;EXTEND - SIGN
                                                        ; 2366
```

```
                                        ; 2367  .TOC    "THE INSTRUCTION LOOP -- STORE ANSWERS"
                                        ; 2368
                                        ; 2369  ;NOTE:  INSTRUCTIONS WHICH STORE IN BOTH AC AND MEMORY
                                        ; 2370  ;       (E.G. ADDB, AOS)  MUST STORE IN MEMORY FIRST
                                        ; 2371  ;       SO THAT IF A PAGE FAIL HAPPENS THE  AC IS
                                        ; 2372  ;       STILL INTACT.
                                        ; 2373
                                        ; 2374  1500:
                                        ; 2375  BWRITE: ;BASE ADDRESS OF BWRITE DISPATCH
                                        ; 2376
                                        ; 2377  ;CASE 0 -- RESERVED
                                        ; 2378  ;1500:
                                        ; 2379
                                        ; 2380  ;CASE 1  --  RESERVED
                                        ; 2381  ;1501:
                                        ; 2382
                                        ; 2383  ;CASE 2  --  RESERVED
                                        ; 2384  ;1502:
                                        ; 2385
                                        ; 2386  ;CASE 3  --  RESERVED
                                        ; 2387  ;1503:
                                        ; 2388
                                        ; 2389  ;CASE 4 -- STORE SELF
                                        ; 2390  1504:
                                        ; 2391  STSELF: SKIP IF ACO,            ;IS AC # ZERO?
U 1504, 0434,4443,0000,4174,4007,0360,0000,0000,0000    ; 2392          J/STBTH1                ;GO TO STORE BOTH CASE
                                        ; 2393
                                        ; 2394  ;CASE 5 -- STORE DOUBLE AC
                                        ; 2395  1505:
                                        ; 2396  DAC:    AC[1]_[ARX],            ;STORE AC 1
U 1505, 1515,3440,0404,1174,4007,0700,0400,0000,1441    ; 2397          J/STAC                  ;GO STORE AC
                                        ; 2398
                                        ; 2399  ;CASE 6 -- STORE DOUBLE BOTH (KA10 STYLE MEM_AR ONLY)
                                        ; 2400  1506:
                                        ; 2401  STDBTH: MEM WRITE,              ;WAIT FOR MEMORY
                                        ; 2402          MEM_[AR],               ;STORE AR
U 1506, 1505,3333,0003,4175,5007,0701,0200,0000,0002    ; 2403          J/DAC                   ;NOW STORE AC & AC+1
                                        ; 2404
                                        ; 2405  ;CASE 7 -- RESERVED
                                        ; 2406  ;1507:
                                        ; 2407
```

```
                                        ; 2408  ;CASE 10  --  RESERVED
                                        ; 2409  ;1510:
                                        ; 2410
                                        ; 2411  ;CASE 11  --  RESERVED
                                        ; 2412  ;1511:
                                        ; 2413
                                        ; 2414  ;CASE 12  --  RESERVED
                                        ; 2415  ;1512:
                                        ; 2416
                                        ; 2417  ;CASE 13  --  RESERVED
                                        ; 2418  ;1513:
                                        ; 2419
                                        ; 2420  ;CASE 14  --  RESERVED
                                        ; 2421  1514:
                                        ; 2422  FL-BWRITE:                      ;THE NEXT 4 CASES ARE ALSO
                                        ; 2423                                  ;USED IN FLOATING POINT
U 1514, 0104,4751,1217,4374,4007,0700,0000,0000,1000    ; 2424          HALT    [BW14]
                                        ; 2425
                                        ; 2426  ;CASE 15 -- STORE AC
                                        ; 2427  1515:
                                        ; 2428  STAC:   AC_[AR],                ;STORE AC
U 1515, 0100,3440,0303,0174,4156,4700,0400,0000,0000    ; 2429          NEXT INST               ;DO NEXT INSTRUCTION
                                        ; 2430
                                        ; 2431  ;CASE 16 -- STORE IN MEMORY
                                        ; 2432  1516:
                                        ; 2433  STMEM:  MEM WRITE,              ;WAIT FOR MEMORY
                                        ; 2434          MEM_[AR],               ;STORE AR
U 1516, 1400,3333,0003,4175,5007,0701,0200,0000,0002    ; 2435          J/DONE                  ;START FETCH OF NEXT
                                        ; 2436
                                        ; 2437  ;CASE 17 -- STORE BOTH
                                        ; 2438  1517:
                                        ; 2439  STBOTH: MEM WRITE,              ;WAIT FOR MEMORY
                                        ; 2440          MEM_[AR],               ;STORE AR
U 1517, 1515,3333,0003,4175,5007,0701,0200,0000,0002    ; 2441          J/STAC                  ;NOW STORE AC
                                        ; 2442
                                        ; 2443  =0
                                        ; 2444  STBTH1: MEM WRITE,              ;WAIT FOR MEMORY
                                        ; 2445          MEM_[AR],               ;STORE AR
U 0434, 1515,3333,0003,4175,5007,0701,0200,0000,0002    ; 2446          J/STAC                  ;NOW STORE AC
                                        ; 2447  STORE:  MEM WRITE,              ;WAIT FOR MEMORY
                                        ; 2448          MEM_[AR],               ;STORE AC
U 0435, 1400,3333,0003,4175,5007,0701,0200,0000,0002    ; 2449          J/DONE                  ;START NEXT INST
                                        ; 2450
```

```
D 0200, 1015,1515,1100
D 0201, 0015,1515,3000
D 0202, 0116,1404,0700
D 0203, 0004,1504,1700

D 0204, 1015,1402,1100
D 0205, 0015,1402,3000
D 0206, 0116,1402,0700
D 0207, 0004,1402,1700

D 0210, 1015,1405,1100
D 0211, 0015,1405,3000
D 0212, 0116,1405,0700
D 0213, 0004,1405,1700

D 0214, 1015,1403,1100
D 0215, 0015,1515,3000
D 0216, 0116,1403,0700
D 0217, 0004,1403,1700


U 1402, 1500,3770,0303,4344,4003,7700,0200,0003,0001

U 1403, 1404,3333,0003,4174,4007,0520,0000,0000,0000

U 1404, 1500,4443,0000,4174,4003,7700,0200,0003,0001


U 1405, 1404,2441,0303,4174,4467,0701,4000,0001,0001
```

```
; 2451  .TOC    "MOVE GROUP"
; 2452
; 2453          .DCODE
; 2454  200:    R-PF,    AC,     J/STAC          ;MOVE
; 2455          I-PF,    AC,     J/STAC          ;MOVEI
; 2456          W,       M,      J/MOVE          ;MOVEM
; 2457          RW,      S,      J/STSELF        ;MOVES
; 2458
; 2459  204:    R-PF,    AC,     J/MOVS          ;MOVS
; 2460          I-PF,    AC,     J/MOVS          ;MOVSI
; 2461          W,       M,      J/MOVS          ;MOVSM
; 2462          RW,      S,      J/MOVS          ;MOVSS
; 2463
; 2464  210:    R-PF,    AC,     J/MOVN          ;MOVN
; 2465          I-PF,    AC,     J/MOVN          ;MOVNI
; 2466          W,       M,      J/MOVN          ;MOVNM
; 2467          RW,      S,      J/MOVN          ;MOVNS
; 2468
; 2469  214:    R-PF,    AC,     J/MOVM          ;MOVM
; 2470          I-PF,    AC,     J/STAC          ;MOVMI
; 2471          W,       M,      J/MOVM          ;MOVMM
; 2472          RW,      S,      J/MOVM          ;MOVNS
; 2473          .UCODE
; 2474
; 2475  1402:
; 2476  MOVS:   [AR]_[AR] SWAP,EXIT
; 2477
; 2478  1403:
; 2479  MOVM:   READ [AR], SKIP DPO, J/MOVE
; 2480
; 2481  1404:
; 2482  MOVE:   EXIT
; 2483  1405:
; 2484  MOVN:   [AR]_-[AR],              ;NEGATE NUMBER
; 2485          AD FLAGS, 3T,           ;UPDATE FLAGS
; 2486          J/MOVE                  ;STORE ANSWER
; 2487
```

```
D 0250, 0015,1406,1500

U 1406, 0471,3771,0005,0276,6007,0700,0200,0003,0002

U 0471, 1515,3333,0005,4175,5007,0701,0200,0000,0002
```

```
; 2488  .TOC    "EXCH"
; 2489
; 2490          .DCODE
; 2491  250:    R,W TEST,       AC,     J/EXCH
; 2492          .UCODE
; 2493
; 2494  1406:
; 2495  EXCH:   [BR]_AC,                ;COPY AC TO THE BR
; 2496          START WRITE             ;START A WRITE CYCLE
; 2497          MEM WRITE,              ;COMPLETE WRITE CYCLE
; 2498          MEM_[BR],               ;STORE BR (AC) IN MEMORY
; 2499          J/STAC                  ;STORE THE AR IN AC. NOTE: AR
; 2500                                  ; WAS LOADED WITH MEMORY OPERAND
; 2501                                  ; AS PART OF INSTRUCTION DISPATCH
; 2502
```

```
D 0500, 1015,1410,1100
D 0501, 0015,1410,3000
D 0502, 0016,1407,1700
D 0503, 0004,1404,1700

D 0504, 1015,1411,1100
D 0505, 0015,1411,3000
D 0506, 0016,1413,1700
D 0507, 0004,1414,1700

D 0510, 1015,1432,1100
D 0511, 0015,1432,3000
D 0512, 0116,1432,0700
D 0513, 0004,1432,1700

D 0514, 1015,1424,1100
D 0515, 0015,1424,3000
D 0516, 0116,1424,0700
D 0517, 0004,1424,1700

D 0520, 1015,1433,1100
D 0521, 0015,1433,3000
D 0522, 0116,1433,0700
D 0523, 0004,1433,1700

D 0524, 1015,1425,1100
D 0525, 0015,1425,3000
D 0526, 0116,1425,0700
D 0527, 0004,1425,1700

D 0530, 1015,1430,1100
D 0531, 0015,1430,3000
D 0532, 0116,1430,0700
D 0533, 0004,1430,1700

D 0534, 1015,1422,1100
D 0535, 0015,1422,3000
D 0536, 0116,1422,0700
D 0537, 0004,1422,1700
```


HALFWORD GROUP

```
; 2503  .TOC    "HALFWORD GROUP"
; 2504  ;       DESTINATION LEFT HALF
; 2505
; 2506          .DCODE
; 2507  500:    R-PF,   AC,     J/HLL
; 2508          I-PF,   AC,     J/HLL
; 2509          RW,     M,      J/HRR           ;HLLM = HRR EXCEPT FOR STORE
; 2510          RW,     S,      J/MOVE          ;HLLS = MOVES
; 2511
; 2512          R-PF,   AC,     J/HRL
; 2513          I-PF,   AC,     J/HRL
; 2514          RW,     M,      J/HRLM
; 2515          RW,     S,      J/HRLS
; 2516
; 2517  510:    R-PF,   AC,     J/HLLZ
; 2518          I-PF,   AC,     J/HLLZ
; 2519          W,      M,      J/HLLZ
; 2520          RW,     S,      J/HLLZ
; 2521
; 2522          R-PF,   AC,     J/HRLZ
; 2523          I-PF,   AC,     J/HRLZ
; 2524          W,      M,      J/HRLZ
; 2525          RW,     S,      J/HRLZ
; 2526
; 2527  520:    R-PF,   AC,     J/HLLO
; 2528          I-PF,   AC,     J/HLLO
; 2529          W,      M,      J/HLLO
; 2530          RW,     S,      J/HLLO
; 2531
; 2532          R-PF,   AC,     J/HRLO
; 2533          I-PF,   AC,     J/HRLO
; 2534          W,      M,      J/HRLO
; 2535          RW,     S,      J/HRLO
; 2536
; 2537  530:    R-PF,   AC,     J/HLLE
; 2538          I-PF,   AC,     J/HLLE
; 2539          W,      M,      J/HLLE
; 2540          RW,     S,      J/HLLE
; 2541
; 2542          R-PF,   AC,     J/HRLE
; 2543          I-PF,   AC,     J/HRLE
; 2544          W,      M,      J/HRLE
; 2545          RW,     S,      J/HRLE
; 2546
```

```
D 0540, 1015,1407,1100
D 0541, 0015,1407,3000
D 0542, 0016,1410,1700
D 0543, 0004,1404,1700

D 0544, 1015,1412,1100
D 0545, 0015,1412,3000
D 0546, 0016,1415,1700
D 0547, 0004,1416,1700

D 0550, 1015,1420,1100
D 0551, 0015,1420,3000
D 0552, 0116,1420,0700
D 0553, 0004,1420,1700

D 0554, 1015,1426,1100
D 0555, 0015,1426,3000
D 0556, 0116,1426,0700
D 0557, 0004,1426,1700

D 0560, 1015,1421,1100
D 0561, 0015,1421,3000
D 0562, 0116,1421,0700
D 0563, 0004,1421,1700

D 0564, 1015,1427,1100
D 0565, 0015,1427,3000
D 0566, 0116,1427,0700
D 0567, 0004,1427,1700

D 0570, 1015,1417,1100
D 0571, 0015,1417,3000
D 0572, 0116,1417,0700
D 0573, 0004,1417,1700

D 0574, 1015,1423,1100
D 0575, 0015,1423,3000
D 0576, 0116,1423,0700
D 0577, 0004,1423,1700
```

```
; 2547  ;       DESTINATION RIGHT HALF
; 2548
; 2549  540:    R-PF,   AC,     J/HRR
; 2550          I-PF,   AC,     J/HRR
; 2551          RW,     M,      J/HLL           ;HRRM = HLL EXCEPT FOR STORE
; 2552          RW,     S,      J/MOVE          ;HRRS = MOVES
; 2553
; 2554          R-PF,   AC,     J/HLR
; 2555          I-PF,   AC,     J/HLR
; 2556          RW,     M,      J/HLRM
; 2557          RW,     S,      J/HLRS
; 2558
; 2559  550:    R-PF,   AC,     J/HRRZ
; 2560          I-PF,   AC,     J/HRRZ
; 2561          W,      M,      J/HRRZ
; 2562          RW,     S,      J/HRRZ
; 2563
; 2564          R-PF,   AC,     J/HLRZ
; 2565          I-PF,   AC,     J/HLRZ
; 2566          W,      M,      J/HLRZ
; 2567          RW,     S,      J/HLRZ
; 2568
; 2569  560:    R-PF,   AC,     J/HRRO
; 2570          I-PF,   AC,     J/HRRO
; 2571          W,      M,      J/HRRO
; 2572          RW,     S,      J/HRRO
; 2573
; 2574          R-PF,   AC,     J/HLRO
; 2575          I-PF,   AC,     J/HLRO
; 2576          W,      M,      J/HLRO
; 2577          RW,     S,      J/HLRO
; 2578
; 2579  570:    R-PF,   AC,     J/HRRE
; 2580          I-PF,   AC,     J/HRRE
; 2581          W,      M,      J/HRRE
; 2582          RW,     S,      J/HRRE
; 2583
; 2584          R-PF,   AC,     J/HLRE
; 2585          I-PF,   AC,     J/HLRE
; 2586          W,      M,      J/HLRE
; 2587          RW,     S,      J/HLRE
; 2588
; 2589          .UCODE
; 2590
```

```
U 1407,  1500,3771,0003,0276,0003,7700,0200,0003,0001    ; 2591  ;FIRST THE GUYS THAT LEAVE THE OTHER HALF ALONE
                                                         ; 2592
                                                         ; 2593  ;THE AR CONTAINS THE MEMORY OPERAND. SO WE WANT TO PUT THE LH OF
                                                         ; 2594  ; AC INTO AR TO DO A HRR. OBVIOUS THING FOR HLL.
                                                         ; 2595  1407:
U 1407,  1500,3771,0003,0276,0003,7700,0200,0003,0001    ; 2596  HRR:    [AR]_AC,HOLD RIGHT,EXIT
                                                         ; 2597  1410:
U 1410,  1500,3771,0003,0270,6003,7700,0200,0003,0001    ; 2598  HLL:    [AR]_AC,HOLD LEFT,EXIT
                                                         ; 2599
                                                         ; 2600  ;HRL FLOW:
                                                         ; 2601  ;AT HRL AR CONTAINS:
                                                         ; 2602  ;
                                                         ; 2603  ;       !-----------------!-----------------!
                                                         ; 2604  ;       !     LH OF (E)   !     RH OF (E)   !
                                                         ; 2605  ;       !-----------------!-----------------!
                                                         ; 2606  ;
                                                         ; 2607  ;AR_AR SWAP GIVES:
                                                         ; 2608  ;
                                                         ; 2609  ;       !-----------------!-----------------!
                                                         ; 2610  ;       !     RH OF (E)   !     LH OF (E)   !
                                                         ; 2611  ;       !-----------------!-----------------!
                                                         ; 2612  ;
                                                         ; 2613  ;AT HLL, AR_AC,HOLD LEFT GIVES:
                                                         ; 2614  ;
                                                         ; 2615  ;       !-----------------!-----------------!
                                                         ; 2616  ;       !     RH OF (E)   !     RH OF AC    !
                                                         ; 2617  ;       !-----------------!-----------------!
                                                         ; 2618  ;
                                                         ; 2619  ;THE EXIT MACRO CAUSES THE AR TO BE STORED IN AC (AT STAC).
                                                         ; 2620  ; THE REST OF THE HALF WORD IN THIS GROUP ARE VERY SIMILAR.
                                                         ; 2621
                                                         ; 2622  1411:
U 1411,  1410,3770,0303,4344,4007,0700,0000,0000,0000    ; 2623  HRL:    [AR]_[AR] SWAP,J/HLL
                                                         ; 2624  1412:
U 1412,  1407,3770,0303,4344,4007,0700,0000,0000,0000    ; 2625  HLR:    [AR]_[AR] SWAP,J/HRR
                                                         ; 2626
                                                         ; 2627  1413:
U 1413,  0506,3770,0303,4344,4007,0700,0000,0000,0000    ; 2628  HRLM:   [AR]_[AR] SWAP
U 0506,  1402,3771,0003,0270,6007,0700,0000,0000,0000    ; 2629          [AR]_AC,HOLD LEFT,J/MOVS
                                                         ; 2630  1414:
U 1414,  1500,3770,0303,4344,0003,7700,0200,0003,0001    ; 2631  HRLS:   [AR]_[AR] SWAP,HOLD RIGHT,EXIT
                                                         ; 2632
                                                         ; 2633  1415:
U 1415,  0511,3770,0303,4344,4007,0700,0000,0000,0000    ; 2634  HLRM:   [AR]_[AR] SWAP
U 0511,  1402,3771,0003,0276,0007,0700,0000,0000,0000    ; 2635          [AR]_AC,HOLD RIGHT,J/MOVS
                                                         ; 2636  1416:
U 1416,  1500,3770,0303,4340,4003,7700,0200,0003,0001    ; 2637  HLRS:   [AR]_[AR] SWAP,HOLD LEFT,EXIT
                                                         ; 2638
```

```
U 1417,  1420,3333,0003,4174,4007,0530,0000,0000,0000
U 1420,  1500,5731,0003,4174,4003,7700,0200,0003,0001
U 1421,  1500,5431,1203,4174,4003,7700,0200,0003,0001

U 1422,  1424,3333,0003,4174,4007,0530,0000,0000,0000
U 1424,  1402,3771,0003,4374,0007,0700,0000,0000,0000
U 1425,  1402,3771,0003,4374,0007,0700,0000,0077,7777

U 1423,  1426,3333,0003,4174,4007,0520,0000,0000,0000
U 1426,  1402,3771,0003,4370,4007,0700,0000,0000,0000
U 1427,  1402,3771,0003,4370,4007,0700,0000,0077,7777

U 1430,  1432,3333,0003,4174,4007,0520,0000,0000,0000
U 1432,  1500,5371,0003,4174,4003,7700,0200,0003,0001
U 1433,  1500,5341,1203,4174,4003,7700,0200,0003,0001
```

HALFWORD GROUP

```
; 2639  ;NOW THE HALFWORD OPS WHICH CONTROL THE "OTHER" HALF.
; 2640  ; ENTER WITH O,,E (E) OR (AC) IN AR
; 2641
; 2642  1417:
; 2643  HRRE:   READ [AR],SKIP DP18
; 2644  1420:
; 2645  HRRZ:   [AR] LEFT_O, EXIT
; 2646  1421:
; 2647  HRRO:   [AR] LEFT_-1, EXIT
; 2648
; 2649  1422:
; 2650  HRLE:   READ [AR],SKIP DP18
; 2651  1424:
; 2652  HRLZ:   [AR]_#,#/O,HOLD RIGHT,J/MOVS
; 2653  1425:
; 2654  HRLO:   [AR]_#,#/777777,HOLD RIGHT,J/MOVS
; 2655
; 2656  1423:
; 2657  HLRE:   READ [AR],SKIP DPO
; 2658  1426:
; 2659  HLRZ:   [AR]_#,#/O,HOLD LEFT,J/MOVS
; 2660  1427:
; 2661  HLRO:   [AR]_#,#/777777,HOLD LEFT,J/MOVS
; 2662
; 2663  1430:
; 2664  HLLE:   READ [AR],SKIP DPO
; 2665  1432:
; 2666  HLLZ:   [AR] RIGHT_O, EXIT
; 2667  1433:
; 2668  HLLO:   [AR] RIGHT_-1, EXIT
; 2669
```

```
D 0120, 0205,1505,1100
D 0121, 0215,1434,1100



U 1434, 3071,4551,0404,4374,0007,0700,0010,0037,7777
U 1436, 1515,3440,0404,1174,4007,0700,0400,0000,1441


D 0124, 0300,1567,0100
D 0125, 0100,1565,0500



U 1565, 3070,3771,0004,1276,6007,0701,0010,0000,1441



U 1567, 0512,0113,0207,4174,4007,0700,0200,0003,0312
U 0512, 0531,3333,0004,4175,5007,0701,0200,0000,0002



U 0531, 0435,3443,0200,4174,4007,0700,0200,0003,0312
```

```
; 2670  .TOC    "DMOVE, DMOVN, DMOVEM, DMOVNM"
; 2671
; 2672          .DCODE
; 2673  120:    DBL R,  DAC,     J/DAC
; 2674          DBL R,  AC,      J/DMOVN
; 2675          .UCODE
; 2676
; 2677  1434:
; 2678  DMOVN:  CLEAR ARX0, CALL [DBLNGA]
; 2679  1436:   AC[1]_[ARX], J/STAC
; 2680
; 2681          .DCODE
; 2682  124:    DBL AC,          J/DMOVN1
; 2683          W,               J/DMOVNM
; 2684          .UCODE
; 2685
; 2686
; 2687  1565:
; 2688  DMOVNM: [ARX]_AC[1],CALL [DBLNEG]
; 2689  1567:
; 2690  DMOVN1: [HR]+[ONE],              ;GET E+1
; 2691          LOAD VMA,                ;PUT THAT IN VMA
; 2692          START WRITE,             ;STORE IN E+1
; 2693          PXCT DATA                ;DATA CYCLE
; 2694          MEM WRITE, MEM_[ARX]     ;STORE LOW WORD
; 2695          VMA_[HR],                ;GET E
; 2696          LOAD VMA,                ;SAVE IN VMA
; 2697          PXCT DATA,               ;OPERAND STORE
; 2698          START WRITE,             ;START MEM CYCLE
; 2699          J/STORE                  ;GO STORE AR
; 2700
```

```
D 0400, 0015,1441,3000
D 0401, 0015,1441,3000
D 0402, 0016,1441,2700
D 0403, 0017,1441,2700

U 1441, 1500,4221,0003,4174,4003,7700,0200,0003,0001

D 0404, 1015,1442,1100
D 0405, 0015,1442,3000
D 0406, 0016,1442,1700
D 0407, 0017,1442,1700

U 1442, 1500,4551,0303,0274,4003,7700,0200,0003,0001

D 0410, 1015,1443,1100
D 0411, 0015,1443,3000
D 0412, 0016,1443,1700
D 0413, 0017,1443,1700

U 1443, 1500,5551,0303,0274,4003,7700,0200,0003,0001

D 0414, 1015,1404,1100
D 0415, 0015,1404,3000
D 0416, 0016,1404,1700
D 0417, 0017,1404,1700

D 0420, 1015,1444,1100
D 0421, 0015,1444,3000
D 0422, 0016,1444,1700
D 0423, 0017,1444,1700

U 1444, 1442,7441,0303,4174,4007,0700,0000,0000,0000

D 0424, 0000,1400,1100
D 0425, 0000,1400,2100
D 0426, 0116,1404,0700
D 0427, 0116,1404,0700
```

```
; 2701  .TOC    "BOOLEAN GROUP"
; 2702
; 2703          .DCODE
; 2704  400:    I-PF,   AC,     J/SETZ
; 2705          I-PF,   AC,     J/SETZ
; 2706          IW,     M,      J/SETZ
; 2707          IW,     B,      J/SETZ
; 2708          .UCODE
; 2709
; 2710  1441:
; 2711  SETZ:   [AR]_0, EXIT
; 2712
; 2713          .DCODE
; 2714  404:    R-PF,   AC,     J/AND
; 2715          I-PF,   AC,     J/AND
; 2716          RW,     M,      J/AND
; 2717          RW,     B,      J/AND
; 2718          .UCODE
; 2719
; 2720  1442:
; 2721  AND:    [AR]_[AR].AND.AC,EXIT
; 2722
; 2723          .DCODE
; 2724  410:    R-PF,   AC,     J/ANDCA
; 2725          I-PF,   AC,     J/ANDCA
; 2726          RW,     M,      J/ANDCA
; 2727          RW,     B,      J/ANDCA
; 2728          .UCODE
; 2729
; 2730  1443:
; 2731  ANDCA:  [AR]_[AR].AND.NOT.AC,EXIT
; 2732
; 2733          .DCODE
; 2734  414:    R-PF,   AC,     J/MOVE   ;SETM = MOVE
; 2735          I-PF,   AC,     J/MOVE
; 2736          RW,     M,      J/MOVE   ;SETMM = NOP THAT WRITES MEMORY
; 2737          RW,     B,      J/MOVE   ;SETMB = MOVE THAT WRITES MEMORY
; 2738
; 2739  420:    R-PF,   AC,     J/ANDCM
; 2740          I-PF,   AC,     J/ANDCM
; 2741          RW,     M,      J/ANDCM
; 2742          RW,     B,      J/ANDCM
; 2743          .UCODE
; 2744
; 2745  1444:
; 2746  ANDCM:  [AR]_.NOT.[AR],J/AND
; 2747
; 2748          .DCODE
; 2749  424:    R,              J/DONE
; 2750          I,              J/DONE
; 2751          W,      M,      J/MOVE          ;SETAM = MOVEM
; 2752          W,      M,      J/MOVE          ;SETAB, TOO
; 2753          .UCODE
; 2754
```

```
D 0430, 1015,1445,1100
D 0431, 0015,1445,3000
D 0432, 0016,1445,1700
D 0433, 0017,1445,1700

U 1445, 1500,6551,0303,0274,4003,7700,0200,0003,0001

D 0434, 1015,1446,1100
D 0435, 0015,1446,3000
D 0436, 0016,1446,1700
D 0437, 0017,1446,1700

U 1446, 1500,3551,0303,0274,4003,7700,0200,0003,0001

D 0440, 1015,1447,1100
D 0441, 0015,1447,3000
D 0442, 0016,1447,1700
D 0443, 0017,1447,1700

U 1447, 1443,7441,0303,4174,4007,0700,0000,0000,0000

D 0444, 1015,1450,1100
D 0445, 0015,1450,3000
D 0446, 0016,1450,1700
D 0447, 0017,1450,1700

U 1450, 1500,7551,0303,0274,4003,7700,0200,0003,0001

D 0450, 0015,1451,3000
D 0451, 0015,1451,3000
D 0452, 0016,1451,2700
D 0453, 0017,1451,2700

U 1451, 1500,7771,0003,0274,4003,7700,0200,0003,0001
```

```
; 2755            .DCODE
; 2756  430:      R-PF,    AC,      J/XOR
; 2757            I-PF,    AC,      J/XOR
; 2758            RW,      M,       J/XOR
; 2759            RW,      B,       J/XOR
; 2760            .UCODE
; 2761
; 2762  1445:
; 2763  XOR:      [AR]_[AR].XOR.AC,EXIT
; 2764
; 2765            .DCODE
; 2766  434:      R-PF,    AC,      J/IOR
; 2767            I-PF,    AC,      J/IOR
; 2768            RW,      M,       J/IOR
; 2769            RW,      B,       J/IOR
; 2770            .UCODE
; 2771
; 2772  1446:
; 2773  IOR:      [AR]_[AR].OR.AC,EXIT
; 2774
; 2775            .DCODE
; 2776  440:      R-PF,    AC,      J/ANDCB
; 2777            I-PF,    AC,      J/ANDCB
; 2778            RW,      M,       J/ANDCB
; 2779            RW,      B,       J/ANDCB
; 2780            .UCODE
; 2781
; 2782  1447:
; 2783  ANDCB:    [AR]_.NOT.[AR],J/ANDCA
; 2784
; 2785            .DCODE
; 2786  444:      R-PF,    AC,      J/EQV
; 2787            I-PF,    AC,      J/EQV
; 2788            RW,      M,       J/EQV
; 2789            RW,      B,       J/EQV
; 2790            .UCODE
; 2791
; 2792  1450:
; 2793  EQV:      [AR]_[AR].EQV.AC,EXIT
; 2794
; 2795            .DCODE
; 2796  450:      I-PF,    AC,      J/SETCA
; 2797            I-PF,    AC,      J/SETCA
; 2798            IW,      M,       J/SETCA
; 2799            IW,      B,       J/SETCA
; 2800            .UCODE
; 2801
; 2802  1451:
; 2803  SETCA:    [AR]_.NOT.AC,EXIT
; 2804
```

```
D 0454, 1015,1452,1100
D 0455, 0015,1452,3000
D 0456, 0016,1452,1700
D 0457, 0017,1452,1700



U 1452, 0532,7771;0005,0274,4007,0700,0000,0000,0000
U 0532, 1500,3111,0503,4174,4003,7700,0200,0003,0001


D 0460, 1015,1453,1100
D 0461, 0015,1453,3000
D 0462, 0016,1453,1700
D 0463, 0017,1453,1700



U 1453, 1500,7441,0303,4174,4003,7700,0200,0003,0001


D 0464, 1015,1454,1100
D 0465, 0015,1454,3000
D 0466, 0016,1454,1700
D 0467, 0017,1454,1700



U 1454, 1446,7441,0303,4174,4007,0700,0000,0000,0000


D 0470, 1015,1455,1100
D 0471, 0015,1455,3000
D 0472, 0016,1455,1700
D 0473, 0017,1455,1700



U 1455, 1453,4551,0303,0274,4007,0700,0000,0000,0000


D 0474, 0015,1456,3000
D 0475, 0015,1456,3000
D 0476, 0016,1456,2700
D 0477, 0017,1456,2700



U 1456, 1500,2441,0703,4174,4003,7700,4200,0003,0001
```

```
; 2805          .DCODE
; 2806  454:    R-PF,    AC,      J/ORCA
; 2807          I-PF,    AC,      J/ORCA
; 2808          RW,      M,       J/ORCA
; 2809          RW,      B,       J/ORCA
; 2810          .UCODE
; 2811
; 2812  1452:
; 2813  ORCA:   [BR]_.NOT.AC
; 2814          [AR]_[AR].OR.[BR],EXIT
; 2815
; 2816          .DCODE
; 2817  460:    R-PF,    AC,      J/SETCM
; 2818          I-PF,    AC,      J/SETCM
; 2819          RW,      M,       J/SETCM
; 2820          RW,      B,       J/SETCM
; 2821          .UCODE
; 2822
; 2823  1453:
; 2824  SETCM:  [AR]_.NOT.[AR],EXIT
; 2825
; 2826          .DCODE
; 2827  464:    R-PF,    AC,      J/ORCM
; 2828          I-PF,    AC,      J/ORCM
; 2829          RW,      M,       J/ORCM
; 2830          RW,      B,       J/ORCM
; 2831          .UCODE
; 2832
; 2833  1454:
; 2834  ORCM:   [AR]_.NOT.[AR],J/IOR
; 2835
; 2836          .DCODE
; 2837  470:    R-PF,    AC,      J/ORCB
; 2838          I-PF,    AC,      J/ORCB
; 2839          RW,      M,       J/ORCB
; 2840          RW,      B,       J/ORCB
; 2841          .UCODE
; 2842
; 2843  1455:
; 2844  ORCB:   [AR]_[AR].AND.AC,J/SETCM
; 2845
; 2846          .DCODE
; 2847  474:    I-PF,    AC,      J/SETO
; 2848          I-PF,    AC,      J/SETO
; 2849          IW,      M,       J/SETO
; 2850          IW,      B,       J/SETO
; 2851          .UCODE
; 2852
; 2853  1456:
; 2854  SETO:   [AR]_-[ONE], EXIT
; 2855
```

```
D 0240, 0400,1622,1000
D 0241, 0400,1632,1000
D 0242, 0400,1612,1000
D 0243, 0000,1462,2100
D 0244, 0000,1466,3000
D 0245, 0500,1470,1000
D 0246, 0500,1464,1000

U 1612, 0564,3771,0003,0276,6007,0700,1000,0031,1777

U 1613, 0572,4551,1203,0276,6007,0700,1000,0041,0001

U 0564, 1515,3445,0303,4174,4007,0700,1020,0041,0001
```

```
; 2856  .TOC    "ROTATES AND LOGICAL SHIFTS -- ROT, LSH, JFFO"
; 2857
; 2858          .DCODE
; 2859  240:    SH,             J/ASH
; 2860          SH,             J/ROT
; 2861          SH,             J/LSH
; 2862          I,              J/JFFO
; 2863          I-PF,           J/ASHC
; 2864  245:    SHC,            J/ROTC
; 2865          SHC,            J/LSHC
; 2866          .UCODE
; 2867
; 2868
; 2869  ;HERE IS THE CODE FOR LOGICAL SHIFT. THE EFFECTIVE ADDRESS IS
; 2870  ; IN AR.
; 2871  1612:
; 2872  LSH:    [AR]_AC,                ;PICK UP AC
; 2873          FE_-FE-1,               ;NEGATIVE SHIFT
; 2874          J/LSHL                  ;SHIFT LEFT
; 2875  1613:   [AR]_AC.AND.MASK,        ;MAKE IT LOOK POSITIVE
; 2876          FE_FE+1,                ;UNDO -1 AT SHIFT
; 2877          J/ASHR                  ;GO SHIFT RIGHT
; 2878
; 2879  LSHL:   [AR]_[AR]*2,            ;SHIFT LEFT
; 2880          SHIFT, J/STAC           ;FAST SHIFT & GO STORE AC
; 2881
```

```
U 1622, 0611,4222,0000,4174,4007,0700,0000,0000,0000
U 1623, 0572,3771,0003,0276,6007,0700,1000,0041,0001
U 0572, 1515,3447,0303,4174,4007,0700,1020,0041,0001
U 0611, 0454,3777,0003,0274,4007,0631,2000,0060,0000
U 0454, 0454,3444,0303,4174,4447,0630,2000,0060,0000
U 0455, 1515,3445,0303,4174,4007,0700,0000,0000,0000
```

```
; 2882  ;HERE IS THE CODE FOR ARITHMETIC SHIFT. THE EFFECTIVE ADDRESS IS
; 2883  ; IN AR.
; 2884
; 2885  ASH36 LEFT      "[AR]_[AR]*2 LONG, ASHC, STEP SC, ASH AROV"
; 2886
; 2887  1622:
; 2888  ASH:    Q_0, J/ASHLO            ;HARDWARE ONLY DOES ASHC
; 2889  1623:   [AR]_AC,                ;GET THE ARGUMENT
; 2890          FE_FE+1                 ;FE HAS NEGATIVE SHIFT COUNT
; 2891  ASHR:   [AR]_[AR]*.5,           ;SHIFT RIGHT
; 2892          ASH, SHIFT,             ;FAST SHIFT
; 2893          J/STAC                  ;STORE AC WHEN DONE
; 2894
; 2895  ASHLO:  [AR]_AC*.5,             ;GET INTO 9 CHIPS
; 2896          STEP SC                 ;SEE IF NULL SHIFT
; 2897  =0
; 2898  ASHL:   ASH36 LEFT, J/ASHL      ;SHIFT LEFT
; 2899                                  ;SLOW BECAUSE WE HAVE TO
; 2900                                  ; TEST FOR OVERFLOW
; 2901
; 2902  ASHX:   [AR]_[AR]*2, J/STAC     ;SHIFT BACK INTO 10 CHIPS
; 2903
```

```
U 1632, 0706,3777,0003,0274,4007,0701,1000,0031,1777
U 1633, 0632,3777,0003,0274,4007,0701,1000,0041,0001
U 0632, 0646,3447,0303,4174,4007,0700,0000,0000,0000
U 0646, 0701,3447,0303,4174,4037,0700,1020,0041,0001
U 0701, 0455,3445,0303,4174,4007,0700,0000,0000,0000
U 0706, 0747,3447,0303,4174,4007,0700,0000,0000,0000
U 0747, 0701,3445,0303,4174,4037,0700,1020,0041,0001
```

```
; 2904  ;HERE IS THE CODE FOR ROTATE. THE EFFECTIVE ADDRESS IS
; 2905  ; IN AR.
; 2906  1632:
; 2907  ROT:    [AR]_AC*.5,             ;PICK UP THE AC (& SHIFT)
; 2908          FE_-FE-1,               ;NEGATIVE SHIFT COUNT
; 2909          J/ROTL                  ;ROTATE LEFT
; 2910  1633:   [AR]_AC*.5,             ;PICK UP THE AC (& SHIFT)
; 2911          FE_FE+1                 ;NEGATIVE SHIFT COUNT
; 2912          [AR]_[AR]*.5            ;PUT IN 9 DIPS
; 2913          [AR]_[AR]*.5,           ;SHIFT RIGHT
; 2914          ROT, SHIFT              ;FAST SHIFT
; 2915  ASHXX:  [AR]_[AR]*2,J/ASHX       ;SHIFT TO STD PLACE
; 2916
; 2917  ROTL:   [AR]_[AR]*.5             ;PUT IN RIGHT 36-BITS
; 2918          [AR]_[AR]*2,            ;ROTATE LEFT
; 2919          ROT, SHIFT,             ;FAST SHIFT
; 2920          J/ASHXX                 ;ALL DONE--SHIFT BACK
; 2921
```

```
U 1462, 0502,4551,1205,0276,6007,0622,0000,0000,0000

U 0502, 0752,3441,0301,4174,4007,0700,0200,0014,0012
U 0503, 1400,4223,0000,1174,4007,0700,0400,0000,1441

U 0752, 0514,4443,0000,4174,4007,0700,1000,0071,1764

U 0514, 0514,3445,0505,4174,4007,0520,1000,0041,0004
U 0515, 0767,3777,0003,4334,4057,0700,0000,0041,0000
U 0767, 1006,4251,0303,4374,4007,0700,0000,0000,0077
U 1006, 0100,3440,0303,1174,4156,4700,0400,0000,1441
```

```
; 2922  1462:
; 2923  JFFO:   [BR]_AC.AND.MASK, 4T,    ;GET AC WITH NO SIGN
; 2924          SKIP AD.EQ.0             ; EXTENSION. SKIP IF
; 2925                                   ; ZERO.
; 2926  =0       [PC]_[AR],              ;NOT ZERO--JUMP
; 2927          LOAD VMA, FETCH,         ;GET NEXT INST
; 2928          J/JFFO1                  ;ENTER LOOP
; 2929          AC[1]_0, J/DONE          ;ZERO--DONE
; 2930
; 2931  JFFO1:  FE_-12.                  ;WHY -12.? WELL THE
; 2932                                   ; HARDWARE LOOKS AT
; 2933                                   ; BIT -2 SO THE FIRST
; 2934                                   ; 2 STEPS MOVE THE BR
; 2935                                   ; OVER. WE ALSO LOOK AT
; 2936                                   ; THE DATA BEFORE THE SHIFT
; 2937                                   ; SO WE END UP GOING 1 PLACE
; 2938                                   ; TOO MANY. THAT MEANS THE
; 2939                                   ; FE SHOULD START AT -3.
; 2940                                   ; HOWEVER, WE COUNT THE FE BY
; 2941                                   ; 4  (BECAUSE THE 2 LOW ORDER
; 2942                                   ; BITS DO NOT COME BACK) SO
; 2943                                   ; FE_-12.
; 2944  =0
; 2945  JFFOL:  [BR]_[BR]*2,             ;SHIFT LEFT
; 2946          FE_FE+4,                 ;COUNT UP BIT NUMBER
; 2947          SKIP DPO, J/JFFOL        ;LOOP TILL WE FIND THE BIT
; 2948          [AR]_FE                  ;GET ANSWER BACK
; 2949          [AR]_[AR].AND.# CLR LH,#/77 ;MASK TO 1 COPY
; 2950          AC[1]_[AR], NEXT INST    ;STORE AND EXIT
; 2951
```

```
U 1464, 0544,4443,0000,4174,4007,0630,2000,0060,0000
U 1465, 1023,3333,0003,4174,4007,0700,2000,0031,5777
U 1023, 0534,4443,0000,4174,4007,0630,2000,0060,0000

U 0534, 0534,3446,0505,4174,4057,0630,2000,0060,0000
U 0535, 1026,3444,0505,4174,4007,0700,0000,0000,0000

U 0544, 0544,3444,0505,4174,4057,0630,2000,0060,0000
U 0545, 1026,3444,0505,4174,4007,0700,0000,0000,0000
U 1026, 1033,3444,0505,4174,4007,0700,0000,0000,0000
U 1033, 1053,3440,0505,0174,4007,0700,0400,0000,0000
```

```
; 2952  .TOC    "ROTATES AND LOGICAL SHIFTS -- LSHC"
; 2953
; 2954  ;SHIFT CONNECTIONS WHEN THE SPECIAL FUNCTION "LSHC" IS DONE:
; 2955  ;
; 2956  ;   !-!   !----!-----------------------------------!
; 2957  ;   !O!-->!0000!         HIGH ORDER 36 BITS        !  RAM FILE
; 2958  ;   !-!   !----!-----------------------------------!
; 2959  ;                                                  ^
; 2960  ;                                                  :
; 2961  ;                .................................
; 2962  ;                :
; 2963  ;          !----!-----------------------------------!
; 2964  ;          !0000!          LOW ORDER 36 BITS        !  Q-REGISTER
; 2965  ;          !----!-----------------------------------!
; 2966  ;                                                  ^
; 2967  ;                                                  :
; 2968  ;                                                !-!
; 2969  ;                                                !O!
; 2970  ;                                                !-!
; 2971  ;
; 2972
; 2973  1464:
; 2974  LSHC:   STEP SC, J/LSHCL
; 2975  1465:   READ [AR], SC_-SHIFT-1
; 2976          STEP SC
; 2977  =O
; 2978  LSHCR:  [BR]_[BR]*.5 LONG,STEP SC,LSHC,J/LSHCR
; 2979          [BR]_[BR]*2 LONG,J/LSHCX
; 2980
; 2981  =O
; 2982  LSHCL:  [BR]_[BR]*2 LONG,LSHC,STEP SC,J/LSHCL
; 2983          [BR]_[BR]*2 LONG
; 2984  LSHCX:  [BR]_[BR]*2 LONG
; 2985          AC_[BR], J/ASHCQ1
; 2986
```

```
U 1466, 0554,3333,0003,4174,4007,0330,3000,0041,4000
U 0554, 1036,3772,0000,1275,5007,0701,0000,0000,1441
U 0555, 0100,4443,0000,4174,4156,4700,0000,0000,0000

U 1036, 0614,3776,0005,0274,4007,0631,0000,0000,0000

U 0614, 0634,3447,0505,4174,4007,0700,2000,0041,1776
U 0615, 0624,3447,0505,4174,4007,0700,2000,0031,1776

U 0624, 0624,3446,0505,4174,4047,0630,2000,0060,0000
U 0625, 1046,3444,0505,4174,4047,0700,0000,0000,0000

U 0634, 0634,3444,0505,4174,4447,0630,2000,0060,0000
U 0635, 1046,3444,0505,4174,4447,0700,0000,0000,0000
U 1046, 0642,0113,0505,0174,4007,0521,0400,0000,0000
U 0642, 1053,4662,0000,4374,0007,0700,0000,0037,7777
U 0643, 1053,3662,0000,4374,0007,0700,0000,0040,0000
U 1053, 0100,3223,0000,1174,4156,4700,0400,0000,1441
```

```
; 2987  .TOC    "ROTATES AND LOGICAL SHIFTS -- ASHC"
; 2988
; 2989
; 2990  1466:
; 2991  ASHC:   READ [AR],                ;PUT AR ON DP
; 2992          SC_SHIFT, LOAD FE,        ;PUT SHIFT IN BOTH SC AND FE
; 2993          SKIP ADR.EQ.0             ;SEE IF NULL SHIFT
; 2994  =0      Q_AC[1],                  ;NOT NULL--GET LOW WORD
; 2995          J/ASHC1                   ;CONTINUE BELOW
; 2996  NIDISP: NEXT INST                 ;NULL--ALL DONE
; 2997  ASHC1:  [BR]_AC*.5 LONG,          ;GET HIGH WORD
; 2998                                    ;AND SHIFT Q
; 2999          SKIP/SC                   ;SEE WHICH DIRECTION
; 3000  =0      [BR]_[BR]*.5,             ;ADJUST POSITION
; 3001          SC_FE+S#, S#/1776,        ;SUBRTACT 2 FROM FE
; 3002          J/ASHCL                   ;GO LEFT
; 3003          [BR]_[BR]*.5,             ;ADJUST POSITION
; 3004          SC_S#-FE, S#/1776         ;SC_-2-FE, SC_+# OF STEPS
; 3005  =0                                ;HERE TO GO RIGHT
; 3006  ASHCR:  [BR]_[BR]*.5 LONG,        ;GO RIGHT
; 3007          ASHC,                     ;SET DATA PATHS FOR ASHC (SEE DPE1)
; 3008          STEP SC, J/ASHCR          ;COUNT THE STEP AND KEEP LOOPING
; 3009          [BR]_[BR]*2 LONG,         ;PUT BACK WHERE IT GOES
; 3010          ASHC, J/ASHCX             ;COMPLETE INSTRUCTION
; 3011
; 3012  =0
; 3013  ASHCL:  [BR]_[BR]*2 LONG,         ;GO LEFT
; 3014          ASHC, ASH AROV,           ;SEE IF OVERFLOW
; 3015          STEP SC, J/ASHCL          ;LOOP OVER ALL PLACES
; 3016          [BR]_[BR]*2 LONG,         ;SHIFT BACK WHERE IT GOES
; 3017          ASHC, ASH AROV            ;CAN STILL OVERFLOW
; 3018  ASHCX:  AC_[BR]+[BR], 3T,         ;PUT BACK HIGH WORD
; 3019          SKIP DPO                  ;SEE HOW TO FIX LOW SIGN
; 3020  =0      Q_Q.AND.#, #/377777,      ;POSITIVE, CLEAR LOW SIGN
; 3021          HOLD RIGHT, J/ASHCQ1      ;GO STORE ANSWER
; 3022          Q_Q.OR.#, #/400000,       ;NEGATIVE, SET LOW SIGN
; 3023          HOLD RIGHT                ;IN LEFT HALF
; 3024  ASHCQ1: AC[1]_Q, NEXT INST         ;PUT BACK Q AND EXIT
; 3025
```

```
                                        ; 3026  .TOC    "ROTATES AND LOGICAL SHIFTS -- ROTC"
                                        ; 3027
                                        ; 3028  ;SHIFT CONNECTIONS WHEN THE SPECIAL FUNCTION "ROTC" IS DONE:
                                        ; 3029  ;
                                        ; 3030  ;         !----!-------------------------------------!
                                        ; 3031  ;    .....>!0000!         HIGH ORDER 36 BITS         !  RAM FILE
                                        ; 3032  ;    :    !----!-------------------------------------!
                                        ; 3033  ;    :                                              ^
                                        ; 3034  ;    :                                              :
                                        ; 3035  ;    :   ..............................................
                                        ; 3036  ;    :   :
                                        ; 3037  ;    :   : !----!-------------------------------------!
                                        ; 3038  ;    :   ..!0000!          LOW ORDER 36 BITS         !  Q-REGISTER
                                        ; 3039  ;    :    !----!-------------------------------------!
                                        ; 3040  ;    :                                              ^
                                        ; 3041  ;    :                                              :
                                        ; 3042  ;    :..............................................:
                                        ; 3043  ;
                                        ; 3044
                                        ; 3045  1470:
U 1470, 0742,4443,0000,4174,4007,0630,2000,0060,0000  ; 3046  ROTC:   STEP SC, J/ROTCL
U 1471, 1064,3333,0003,4174,4007,0700,2000,0031,5777  ; 3047  1471:   READ [AR], SC_-SHIFT-1
U 1064, 0644,4443,0000,4174,4007,0630,2000,0060,0000  ; 3048          STEP SC
                                        ; 3049  =0
U 0644, 0644,3446,0505,4174,4077,0630,2000,0060,0000  ; 3050  ROTCR:  [BR]_[BR]*.5 LONG,STEP SC,ROTC,J/ROTCR
U 0645, 1026,3444,0505,4174,4007,0700,0000,0000,0000  ; 3051          [BR]_[BR]*2 LONG,J/LSHCX
                                        ; 3052
                                        ; 3053  =0
U 0742, 0742,3444,0505,4174,4077,0630,2000,0060,0000  ; 3054  ROTCL:  [BR]_[BR]*2 LONG,ROTC,STEP SC,J/ROTCL
                                        ; 3055          [BR]_[BR]*2 LONG,
U 0743, 1026,3444,0505,4174,4007,0700,0000,0000,0000  ; 3056          J/LSHCX
                                        ; 3057
```

```
; 3058  .TOC    "TEST GROUP"
; 3059
; 3060          .DCODE
; 3061
; 3062  ;SPECIAL MACROS USED ONLY IN B-FIELD OF TEST INSTRUCTIONS
; 3063  TN-              "B/4"
; 3064  TNE              "B/0"
; 3065  WORD-TNE         "B/10"  ;USED IN TIOE
; 3066  TNA              "B/0"
; 3067  TNN              "B/4"
; 3068  WORD-TNN         "B/14"  ;USED IN TION
; 3069  TZ-              "B/5"
; 3070  TZE              "B/1"
; 3071  TZA              "B/1"
; 3072  TZN              "B/5"
; 3073  TC-              "B/6"
; 3074  TCE              "B/2"
; 3075  TCA              "B/2"
; 3076  TCN              "B/6"
; 3077  TO-              "B/7"
; 3078  TOE              "B/3"
; 3079  TOA              "B/3"
; 3080  TON              "B/7"
; 3081
; 3082  600:    I,              J/DONE          ;TRN- IS NOP
; 3083          I,              J/DONE          ;SO IS TLN-
; 3084          I,      TNE,    J/TDXX
; 3085          I,      TNE,    J/TSXX
; 3086          I,      TNA,    J/TDX
; 3087          I,      TNA,    J/TSX
; 3088          I,      TNN,    J/TDXX
; 3089          I,      TNN,    J/TSXX
; 3090
; 3091  610:    I,              J/DONE          ;TDN- IS A NOP
; 3092          I,              J/DONE          ;TSN- ALSO
; 3093          R,      TNE,    J/TDXX
; 3094          R,      TNE,    J/TSXX
; 3095          R,      TNA,    J/TDX
; 3096          R,      TNA,    J/TSX
; 3097          R,      TNN,    J/TDXX
; 3098          R,      TNN,    J/TSXX
; 3099
; 3100  620:    I,      TZ-,    J/TDX
; 3101          I,      TZ-,    J/TSX
; 3102          I,      TZE,    J/TDXX
; 3103          I,      TZE,    J/TSXX
; 3104          I,      TZA,    J/TDX
; 3105          I,      TZA,    J/TSX
; 3106          I,      TZN,    J/TDXX
; 3107          I,      TZN,    J/TSXX
; 3108
```

```
D 0600, 0000,1400,2100
D 0601, 0000,1400,2100
D 0602, 0000,1475,2100
D 0603, 0000,1474,2100
D 0604, 0000,1473,2100
D 0605, 0000,1472,2100
D 0606, 0004,1475,2100
D 0607, 0004,1474,2100

D 0610, 0000,1400,2100
D 0611, 0000,1400,2100
D 0612, 0000,1475,1100
D 0613, 0000,1474,1100
D 0614, 0000,1473,1100
D 0615, 0000,1472,1100
D 0616, 0004,1475,1100
D 0617, 0004,1474,1100

D 0620, 0005,1473,2100
D 0621, 0005,1472,2100
D 0622, 0001,1475,2100
D 0623, 0001,1474,2100
D 0624, 0001,1473,2100
D 0625, 0001,1472,2100
D 0626, 0005,1475,2100
D 0627, 0005,1474,2100
```

```
D 0630, 0005,1473,1100
D 0631, 0005,1472,1100
D 0632, 0001,1475,1100
D 0633, 0001,1474,1100
D 0634, 0001,1473,1100
D 0635, 0001,1472,1100
D 0636, 0005,1475,1100
D 0637, 0005,1474,1100

D 0640, 0006,1473,2100
D 0641, 0006,1472,2100
D 0642, 0002,1475,2100
D 0643, 0002,1474,2100
D 0644, 0002,1473,2100
D 0645, 0002,1472,2100
D 0646, 0006,1475,2100
D 0647, 0006,1474,2100

D 0650, 0006,1473,1100
D 0651, 0006,1472,1100
D 0652, 0002,1475,1100
D 0653, 0002,1474,1100
D 0654, 0002,1473,1100
D 0655, 0002,1472,1100
D 0656, 0006,1475,1100
D 0657, 0006,1474,1100
D 0660, 0007,1473,2100
D 0661, 0007,1472,2100
D 0662, 0003,1475,2100
D 0663, 0003,1474,2100
D 0664, 0003,1473,2100
D 0665, 0003,1472,2100
D 0666, 0007,1475,2100
D 0667, 0007,1474,2100

D 0670, 0007,1473,1100
D 0671, 0007,1472,1100
D 0672, 0003,1475,1100
D 0673, 0003,1474,1100
D 0674, 0003,1473,1100
D 0675, 0003,1472,1100
D 0676, 0007,1475,1100
D 0677, 0007,1474,1100
```

```
; 3109  630:     R,      TZ-,    J/TDX
; 3110            R,      TZ-,    J/TSX
; 3111            R,      TZE,    J/TDXX
; 3112            R,      TZE,    J/TSXX
; 3113            R,      TZA,    J/TDX
; 3114            R,      TZA,    J/TSX
; 3115            R,      TZN,    J/TDXX
; 3116            R,      TZN,    J/TSXX
; 3117
; 3118  640:     I,      TC-,    J/TDX
; 3119            I,      TC-,    J/TSX
; 3120            I,      TCE,    J/TDXX
; 3121            I,      TCE,    J/TSXX
; 3122            I,      TCA,    J/TDX
; 3123            I,      TCA,    J/TSX
; 3124            I,      TCN,    J/TDXX
; 3125            I,      TCN,    J/TSXX
; 3126
; 3127  650:     R,      TC-,    J/TDX
; 3128            R,      TC-,    J/TSX
; 3129            R,      TCE,    J/TDXX
; 3130            R,      TCE,    J/TSXX
; 3131            R,      TCA,    J/TDX
; 3132            R,      TCA,    J/TSX
; 3133            R,      TCN,    J/TDXX
; 3134            R,      TCN,    J/TSXX
; 3135  660:     I,      TO-,    J/TDX
; 3136            I,      TO-,    J/TSX
; 3137            I,      TOE,    J/TDXX
; 3138            I,      TOE,    J/TSXX
; 3139            I,      TOA,    J/TDX
; 3140            I,      TOA,    J/TSX
; 3141            I,      TON,    J/TDXX
; 3142            I,      TON,    J/TSXX
; 3143
; 3144  670:     R,      TO-,    J/TDX
; 3145            R,      TO-,    J/TSX
; 3146            R,      TOE,    J/TDXX
; 3147            R,      TOE,    J/TSXX
; 3148            R,      TOA,    J/TDX
; 3149            R,      TOA,    J/TSX
; 3150            R,      TON,    J/TDXX
; 3151            R,      TON,    J/TSXX
; 3152
```

```
U 1472, 1473,3770,0303,4344,4007,0700,0000,0000,0000
U 1473, 0014,4221,0005,4174,4003,7700,0000,0000,0000
U 1474, 1475,3770,0303,4344,4007,0700,0000,0000,0000
U 1475, 0014,4551,0305,0274,4003,7700,0000,0000,0000
```

```
; 3153          .UCODE
; 3154
; 3155  ;THESE 64 INSTRUCTIONS ARE DECODED BY MASK MODE(IMMEDIATE OR MEMORY)
; 3156  ; IN THE A FIELD, DISPATCH TO HERE ON THE J FIELD, AND RE-DISPATCH
; 3157  ; FOR THE MODIFICATION ON THE B FIELD.
; 3158
; 3159  ; ENTER WITH O,E OR (E) IN AR, B FIELD BITS 2 AND 3 AS FOLLOWS:
; 3160  ; O O   NO MODIFICATION
; 3161  ; O 1   OS
; 3162  ; 1 O   COMPLEMENT
; 3163  ; 1 1   ONES
; 3164  ;   THIS ORDER HAS NO SIGNIFICANCE EXCEPT THAT IT CORRESPONDS TO THE
; 3165  ;   ORDER OF INSTRUCTIONS AT TGROUP.
; 3166
; 3167  ;THE BIT 1 OF THE B FIELD IS USED TO DETERMINE THE SENSE
; 3168  ; OF THE SKIP
; 3169  ; 1     SKIP IF AC.AND.MASK .NE. O (TXX- AND TXXN)
; 3170  ; O     SKIP IF AC.AND.MASK .EQ. O (TXXA AND TXXE)
; 3171
; 3172  ;BIT O IS UNUSED AND MUST BE ZERO
; 3173
; 3174
; 3175  1472:
; 3176  TSX:    [AR]_[AR] SWAP          ;TSXX AND TLXX
; 3177  1473:
; 3178  TDX:    [BR]_O,TEST DISP        ; ALWAYS AND NEVER SKIP CASES
; 3179
; 3180  1474:
; 3181  TSXX:   [AR]_[AR] SWAP          ;TSXE, TSXN, TLXE, TLXN
; 3182  1475:
; 3183  TDXX:   [BR]_[AR].AND.AC,       ;TDXE, TDXN, TRXE, TRXN
; 3184          TEST DISP
; 3185
```

```
                                                    ; 3186  ;TEST DISP DOES AN 8 WAY BRANCH BASED ON THE B-FIELD OF DROM
                                                    ; 3187
                                                    ; 3188  =1100
                                                    ; 3189  TEST-TABLE:
                                                    ; 3190
                                                    ; 3191  ;CASE 0 & 4      -- TXNX
U 0014,  1400,3333,0005,4174,4007,0571,0000,0000,0000    ; 3192  TXXX:   READ [BR], TXXX TEST, 3T, J/DONE
                                                    ; 3193
                                                    ; 3194  ;CASE 1 & 5 -- TXZ AND TXZX
U 0015,  1077,7441,0303,4174,4007,0700,0000,0000,0000    ; 3195          [AR]_.NOT.[AR],J/TXZX
                                                    ; 3196
                                                    ; 3197  ;CASE 2 & 6 -- TXC AND TXCX
U 0016,  1116,6551,0303,0274,4007,0700,0000,0000,0000    ; 3198          [AR]_[AR].XOR.AC,J/TDONE
                                                    ; 3199
                                                    ; 3200  ;CASE 3 & 7 -- TXO AND TXOX
U 0017,  1116,3551,0303,0274,4007,0700,0000,0000,0000    ; 3201          [AR]_[AR].OR.AC,J/TDONE
                                                    ; 3202
                                                    ; 3203  ;THE SPECIAL FUNCTION TXXX TEST CAUSES A MICROCODE SKIP IF
                                                    ; 3204  ; AD.EQ.0 AND DROM B IS 0-3 OR AD.NE.0 AND DROM B IS 4-7.
                                                    ; 3205
U 1077,  1116,4551,0303,0274,4007,0700,0000,0000,0000    ; 3206  TXZX:   [AR]_[AR].AND.AC
U 1116,  0014,3440,0303,0174,4007,0700,0400,0000,0000    ; 3207  TDONE:  AC_[AR],J/TXXX
                                                    ; 3208  ;       READ BR,TXXX TEST,J/DONE
                                                    ; 3209
```

```
U 0250,  0110,3443,0100,4174,4156,4700,0200,0014,0012    ; 3210  .TOC    "COMPARE -- CAI, CAM"
                                                         ; 3211
                                                         ; 3212          .DCODE
                                                         ; 3213
                                                         ; 3214  ;SPECIAL B-FIELD ENCODING USED BY SKIP-JUMP-COMPARE CLASS
                                                         ; 3215  ; INSTRUCTIONS:
                                                         ; 3216
                                                         ; 3217  SJC-     "B/0"   ;NEVER
                                                         ; 3218  SJCL     "B/1"   ;LESS
                                                         ; 3219  SJCE     "B/2"   ;EQUAL
                                                         ; 3220  SJCLE    "B/3"   ;LESS EQUAL
                                                         ; 3221  SJCA     "B/4"   ;ALWAYS
                                                         ; 3222  SJCGE    "B/5"   ;GREATER THAN OR EQUAL
                                                         ; 3223  SJCN     "B/6"   ;NOT EQUAL
                                                         ; 3224  SJCG     "B/7"   ;GREATER
                                                         ; 3225
                                                         ; 3226          .UCODE
                                                         ; 3227
                                                         ; 3228  ;COMPARE TABLE
                                                         ; 3229  =1000
                                                         ; 3230  SKIP-COMP-TABLE:
                                                         ; 3231
                                                         ; 3232  ;CASE 0 -- NEVER
U 0250,  0110,3443,0100,4174,4156,4700,0200,0014,0012    ; 3233          DONE
                                                         ; 3234
                                                         ; 3235  ;CASE 1 -- LESS
U 0251,  1400,3333,0003,4174,4007,0520,0000,0000,0000    ; 3236          READ [AR], SKIP DPO,J/DONE
                                                         ; 3237
                                                         ; 3238  ;CASE 2 -- EQUAL
U 0252,  1400,3333,0003,4174,4007,0621,0000,0000,0000    ; 3239  SKIPE:  READ [AR], SKIP AD.EQ.O,J/DONE
                                                         ; 3240
                                                         ; 3241  ;CASE 3 -- LESS OR EQUAL
U 0253,  1400,3333,0003,4174,4007,0421,0000,0000,0000    ; 3242          READ [AR], SKIP AD.LE.O,J/DONE
                                                         ; 3243
                                                         ; 3244  ;CASE 4 -- ALWAYS
U 0254,  0110,0111,0701,4170,4156,4700,0200,0014,0012    ; 3245          VMA_[PC]+1, NEXT INST FETCH, FETCH
                                                         ; 3246
                                                         ; 3247  ;CASE 5 -- GREATER THAN OR EQUAL
U 0255,  0260,3333,0003,4174,4007,0520,0000,0000,0000    ; 3248          READ [AR], SKIP DPO,J/SKIP
                                                         ; 3249
                                                         ; 3250  ;CASE 6 -- NOT EQUAL
U 0256,  0260,3333,0003,4174,4007,0621,0000,0000,0000    ; 3251          READ [AR], SKIP AD.EQ.O,J/SKIP
                                                         ; 3252
                                                         ; 3253  ;CASE 7 -- GREATER
U 0257,  0260,3333,0003,4174,4007,0421,0000,0000,0000    ; 3254          READ [AR], SKIP AD.LE.O,J/SKIP
                                                         ; 3255
```

```
D 0300, 0000,1400,2100
D 0301, 0001,1476,2100
D 0302, 0002,1476,2100
D 0303, 0003,1476,2100
D 0304, 0004,1476,2100
D 0305, 0005,1476,2100
D 0306, 0006,1476,2100
D 0307, 0007,1476,2100

D 0310, 0000,1476,1100
D 0311, 0001,1476,1100
D 0312, 0002,1476,1100
D 0313, 0003,1476,1100
D 0314, 0004,1476,1100
D 0315, 0005,1476,1100
D 0316, 0006,1476,1100
D 0317, 0007,1476,1100


U 1476, 0250,2551,0303,0274,4003,7701,4000,0000,0000
```

```
; 3256          .DCODE
; 3257  300:    I,      SJC-,   J/DONE  ;CAI
; 3258          I,      SJCL,   J/CAIM
; 3259          I,      SJCE,   J/CAIM
; 3260          I,      SJCLE,  J/CAIM
; 3261          I,      SJCA,   J/CAIM
; 3262          I,      SJCGE,  J/CAIM
; 3263          I,      SJCN,   J/CAIM
; 3264          I,      SJCG,   J/CAIM
; 3265
; 3266  310:    R,      SJC-,   J/CAIM  ;CAM
; 3267          R,      SJCL,   J/CAIM
; 3268          R,      SJCE,   J/CAIM
; 3269          R,      SJCLE,  J/CAIM
; 3270          R,      SJCA,   J/CAIM
; 3271          R,      SJCGE,  J/CAIM
; 3272          R,      SJCN,   J/CAIM
; 3273          R,      SJCG,   J/CAIM
; 3274          .UCODE
; 3275
; 3276  1476:
; 3277  CAIM:   [AR]_AC-[AR], 3T, SKIP-COMP DISP
; 3278
```

```
D 0330, 0000,1477,1100
D 0331, 0001,1477,1100
D 0332, 0002,1477,1100
D 0333, 0003,1477,1100
D 0334, 0004,1477,1100
D 0335, 0005,1477,1100
D 0336, 0006,1477,1100
D 0337, 0007,1477,1100

U 1477, 0744,3770,0303,4174,0007,0360,0000,0000,0000
U 0744, 0250,3440,0303,0174,4003,7700,0400,0000,0000
U 0745, 0250,4443,0000,4174,4003,7700,0000,0000,0000

D 0350, 0000,1431,1500
D 0351, 0001,1431,1500
D 0352, 0002,1431,1500
D 0353, 0003,1431,1500
D 0354, 0004,1431,1500
D 0355, 0005,1431,1500
D 0356, 0006,1431,1500
D 0357, 0007,1431,1500

U 1431, 1123,0111,0703,4174,4467,0701,0000,0001,0001
U 1123, 1136,4443,0000,4174,4007,0700,0200,0003,0002
U 1136, 1477,3333,0003,4175,5007,0701,0200,0000,0002

D 0370, 0000,1437,1500
D 0371, 0001,1437,1500
D 0372, 0002,1437,1500
D 0373, 0003,1437,1500
D 0374, 0004,1437,1500
D 0375, 0005,1437,1500
D 0376, 0006,1437,1500
D 0377, 0007,1437,1500

U 1437, 1123,1111,0703,4174,4467,0701,4000,0001,0001
```

```
; 3279  .TOC    "ARITHMETIC SKIPS -- AOS, SOS, SKIP"
; 3280  ;ENTER WITH (E) IN AR
; 3281
; 3282          .DCODE
; 3283  330:    R,      SJC-,   J/SKIPS ;NOT A NOP IF AC .NE. 0
; 3284          R,      SJCL,   J/SKIPS
; 3285          R,      SJCE,   J/SKIPS
; 3286          R,      SJCLE,  J/SKIPS
; 3287          R,      SJCA,   J/SKIPS
; 3288          R,      SJCGE,  J/SKIPS
; 3289          R,      SJCN,   J/SKIPS
; 3290          R,      SJCG,   J/SKIPS
; 3291          .UCODE
; 3292
; 3293  1477:
; 3294  SKIPS:  FIX [AR] SIGN,
; 3295          SKIP IF ACO
; 3296  =0      AC_[AR],SKIP-COMP DISP
; 3297          SKIP-COMP DISP
; 3298
; 3299          .DCODE
; 3300  350:    RW,     SJC-,   J/AOS
; 3301          RW,     SJCL,   J/AOS
; 3302          RW,     SJCE,   J/AOS
; 3303          RW,     SJCLE,  J/AOS
; 3304          RW,     SJCA,   J/AOS
; 3305          RW,     SJCGE,  J/AOS
; 3306          RW,     SJCN,   J/AOS
; 3307          RW,     SJCG,   J/AOS
; 3308          .UCODE
; 3309
; 3310  1431:
; 3311  AOS:    [AR]_[AR]+1, 3T, AD FLAGS
; 3312  XOS:    START WRITE
; 3313          MEM WRITE,MEM_[AR],J/SKIPS
; 3314
; 3315          .DCODE
; 3316  370:    RW,     SJC-,   J/SOS
; 3317          RW,     SJCL,   J/SOS
; 3318          RW,     SJCE,   J/SOS
; 3319          RW,     SJCLE,  J/SOS
; 3320          RW,     SJCA,   J/SOS
; 3321          RW,     SJCGE,  J/SOS
; 3322          RW,     SJCN,   J/SOS
; 3323          RW,     SJCG,   J/SOS
; 3324          .UCODE
; 3325
; 3326  1437:
; 3327  SOS:    [AR]_[AR]-1, 3T, AD FLAGS, J/XOS
; 3328
```

```
U 0270, 0100,3440,0505,0174,4156,4700,0400,0000,0000    ; 3329  .TOC    "CONDITIONAL JUMPS -- JUMP, AOJ, SOJ, AOBJ"
                                                        ; 3330  ; ENTER WITH E IN AR
                                                        ; 3331
                                                        ; 3332  =1000
                                                        ; 3333  JUMP-TABLE:
                                                        ; 3334
                                                        ; 3335  ;CASE 0 -- NEVER
U 0270, 0100,3440,0505,0174,4156,4700,0400,0000,0000    ; 3336          AC_[BR], NEXT INST
                                                        ; 3337
                                                        ; 3338  ;CASE 1 -- LESS
U 0271, 0762,3770,0505,0174,4007,0520,0400,0000,0000    ; 3339          AC_[BR] TEST, SKIP DPO, J/JUMP-
                                                        ; 3340
                                                        ; 3341  ;CASE 2 -- EQUAL
U 0272, 0762,3770,0505,0174,4007,0621,0400,0000,0000    ; 3342          AC_[BR] TEST, SKIP AD.EQ.0, J/JUMP-
                                                        ; 3343
                                                        ; 3344  ;CASE 3 -- LESS THAN OR EQUAL
U 0273, 0762,3770,0505,0174,4007,0421,0400,0000,0000    ; 3345          AC_[BR] TEST, SKIP AD.LE.0, J/JUMP-
                                                        ; 3346
                                                        ; 3347  ;CASE 4 -- ALWAYS
U 0274, 0764,3440,0505,0174,4007,0700,0400,0000,0000    ; 3348  JMPA:   AC_[BR], J/JUMPA
                                                        ; 3349
                                                        ; 3350  ;CASE 5 -- GREATER THAN OR EQUAL TO
U 0275, 0764,3770,0505,0174,4007,0520,0400,0000,0000    ; 3351          AC_[BR] TEST, SKIP DPO, J/JUMPA
                                                        ; 3352
                                                        ; 3353  ;CASE 6 -- NOT EQUAL
U 0276, 0764,3770,0505,0174,4007,0621,0400,0000,0000    ; 3354          AC_[BR] TEST, SKIP AD.EQ.0, J/JUMPA
                                                        ; 3355
                                                        ; 3356  ;CASE 7 -- GREATER
U 0277, 0764,3770,0505,0174,4007,0421,0400,0000,0000    ; 3357          AC_[BR] TEST, SKIP AD.LE.0, J/JUMPA
                                                        ; 3358
                                                        ; 3359  =0
U 0762, 0110,3443,0100,4174,4156,4700,0200,0014,0012    ; 3360  JUMP-:  DONE
U 0763, 0110,3441,0301,4170,4156,4700,0200,0014,0012    ; 3361          JUMPA
                                                        ; 3362
                                                        ; 3363  =0
U 0764, 0110,3441,0301,4170,4156,4700,0200,0014,0012    ; 3364  JUMPA:  JUMPA
U 0765, 0110,3443,0100,4174,4156,4700,0200,0014,0012    ; 3365          DONE
                                                        ; 3366
                                                        ; 3367
```

```
                                                            ; 3368          .DCODE
D 0320, 0000,1400,2100                                      ; 3369  320:    I,      SJC-,   J/DONE
D 0321, 0001,1440,2100                                      ; 3370          I,      SJCL,   J/JUMP
D 0322, 0002,1440,2100                                      ; 3371          I,      SJCE,   J/JUMP
D 0323, 0003,1440,2100                                      ; 3372          I,      SJCLE,  J/JUMP
D 0324, 0004,1520,2100                                      ; 3373          I,      SJCA,   J/JRST
D 0325, 0005,1440,2100                                      ; 3374          I,      SJCGE,  J/JUMP
D 0326, 0006,1440,2100                                      ; 3375          I,      SJCN,   J/JUMP
D 0327, 0007,1440,2100                                      ; 3376          I,      SJCG,   J/JUMP
                                                            ; 3377          .UCODE
                                                            ; 3378
                                                            ; 3379  1440:
U 1440, 0270,3771,0005,0276,6003,7700,0000,0000,0000        ; 3380  JUMP:   [BR]_AC,JUMP DISP
                                                            ; 3381
                                                            ; 3382          .DCODE
D 0340, 0000,1611,3000                                      ; 3383  340:    I-PF,   SJC-,   J/AOJ
D 0341, 0001,1611,2100                                      ; 3384          I,      SJCL,   J/AOJ
D 0342, 0002,1611,2100                                      ; 3385          I,      SJCE,   J/AOJ
D 0343, 0003,1611,2100                                      ; 3386          I,      SJCLE,  J/AOJ
D 0344, 0004,1611,2100                                      ; 3387          I,      SJCA,   J/AOJ
D 0345, 0005,1611,2100                                      ; 3388          I,      SJCGE,  J/AOJ
D 0346, 0006,1611,2100                                      ; 3389          I,      SJCN,   J/AOJ
D 0347, 0007,1611,2100                                      ; 3390          I,      SJCG,   J/AOJ
                                                            ; 3391          .UCODE
                                                            ; 3392
                                                            ; 3393  1611:
U 1611, 0270,0551,0705,0274,4463,7702,0000,0001,0001        ; 3394  AOJ:    [BR]_AC+1, AD FLAGS, 4T, JUMP DISP
                                                            ; 3395
                                                            ; 3396          .DCODE
D 0360, 0000,1542,3000                                      ; 3397  360:    I-PF,   SJC-,   J/SOJ
D 0361, 0001,1542,2100                                      ; 3398          I,      SJCL,   J/SOJ
D 0362, 0002,1542,2100                                      ; 3399          I,      SJCE,   J/SOJ
D 0363, 0003,1542,2100                                      ; 3400          I,      SJCLE,  J/SOJ
D 0364, 0004,1542,2100                                      ; 3401          I,      SJCA,   J/SOJ
D 0365, 0005,1542,2100                                      ; 3402          I,      SJCGE,  J/SOJ
D 0366, 0006,1542,2100                                      ; 3403          I,      SJCN,   J/SOJ
D 0367, 0007,1542,2100                                      ; 3404          I,      SJCG,   J/SOJ
                                                            ; 3405          .UCODE
                                                            ; 3406
                                                            ; 3407  1542:
U 1542, 0270,2551,0705,0274,4463,7702,4000,0001,0001        ; 3408  SOJ:    [BR]_AC-1, AD FLAGS, 4T, JUMP DISP
                                                            ; 3409
                                                            ; 3410          .DCODE
D 0252, 0005,1547,2100                                      ; 3411  252:    I,      SJCGE,  J/AOBJ
D 0253, 0001,1547,2100                                      ; 3412          I,      SJCL,   J/AOBJ
                                                            ; 3413          .UCODE
                                                            ; 3414
                                                            ; 3415  1547:
                                                            ; 3416  AOBJ:   [BR]_AC+1000001,        ;ADD 1 TO BOTH HALF WORDS
                                                            ; 3417          INH CRY18, 3T,          ;NO CARRY INTO LEFT HALF
U 1547, 0270,0551,1505,0274,4403,7701,0000,0000,0000        ; 3418          JUMP DISP               ;HANDLE EITHER AOBJP OR AOBJN
                                                            ; 3419
```

```
D 0254, 0000,1520,6000

D 0255, 0000,1540,2100

U 1520, 0110,3441,0301,4170,4156,4700,0200,0014,0012
U 1521, 0110,3441,0301,4170,4156,4700,0200,0014,0012

U 1522, 0150,1113,0701,4170,4007,0700,4200,0004,0012
U 1523, 2740,4551,0202,4374,0007,0700,0000,0077,7740
U 1524, 1004,4443,0000,4174,4007,0340,0000,0000,0000

U 1525, 2677,3443,0300,4174,4007,0700,0200,0004,0012
U 1526, 0320,4443,0000,4174,4007,0340,0000,0000,0000
U 1527, 1014,4443,0000,4174,4007,0340,0000,0000,0000

U 1530, 1010,1113,0701,4170,4007,0040,4200,0004,0012
U 1531, 2740,4551,0202,4374,0007,0700,0000,0077,7740

U 1532, 0300,1113,0701,4170,4007,0040,4200,0004,0012
U 1533, 2740,4551,0202,4374,0007,0700,0000,0077,7740
U 1534, 1024,4443,0000,4174,4007,0340,0000,0000,0000
U 1535, 2740,4551,0202,4374,0007,0700,0000,0077,7740
U 1536, 2740,4551,0202,4374,0007,0700,0000,0077,7740
U 1537, 2740,4551,0202,4374,0007,0700,0000,0077,7740
```

```
; 3420  .TOC    "AC DECODE JUMPS -- JRST, JFCL"
; 3421
; 3422          .DCODE
; 3423  254:    I,VMA/0, AC DISP,       J/JRST   ;DISPATCHES TO 1 OF 16
; 3424                                           ; PLACES ON AC BITS
; 3425          I,                      J/JFCL
; 3426          .UCODE
; 3427
; 3428  ;JRST DISPATCHES TO ONE OF 16 LOC'NS ON AC BITS
; 3429
; 3430  =0000
; 3431  1520:
; 3432  JRST:   JUMPA                   ;(0) JRST 0,
; 3433  1521:   JUMPA                   ;(1) PORTAL IS SAME AS JRST
; 3434  1522:   VMA_[PC]-1, START READ, ;(2) JRSTF
; 3435          J/JRSTF
; 3436  1523:   UUO                     ;(3)
; 3437  1524:   SKIP KERNEL, J/HALT     ;(4) HALT
; 3438  1525:
; 3439  XJRSTFO: VMA_[AR], START READ, ;(5) XJRSTF
; 3440          J/XJRSTF
; 3441  1526:   SKIP KERNEL, J/XJEN     ;(6) XJEN
; 3442  1527:   SKIP KERNEL, J/XPCW     ;(7) XPCW
; 3443  1530:   VMA_[PC]-1, START READ, ;(10)
; 3444           SKIP IO LEGAL, J/JRST1O
; 3445  1531:   UUO                     ;(11)
; 3446  1532:   VMA_[PC]-1, START READ, ;(12) JEN
; 3447           SKIP IO LEGAL, J/JEN
; 3448  1533:   UUO                     ;(13)
; 3449  1534:   SKIP KERNEL, J/SFM      ;(14) SFM
; 3450  1535:   UUO                     ;(15)
; 3451  1536:   UUO                     ;(16)
; 3452  1537:   UUO                     ;(17)
; 3453
```

```
U 0150, 1143,3771,0002,4365,5217,0700,0210,0000,0002
U 0152, 0110,3441,0301,4170,4156,4700,0200,0014,0012

U 1143, 0030,4443,0000,2174,4006,6700,0000,0000,0000



U 0030, 0002,3773,0000,2274,4464,1700,0000,0001,0004


U 0032, 0002,3333,0002,4174,4464,1700,0000,0001,0004


U 0034, 2621,0551,0202,2270,4007,0700,0200,0004,0012


U 0036, 2621,3443,0200,4174,4007,0700,0200,0004,0112


U 2621, 1143,3771,0002,4365,5217,0700,0200,0000,0002
```

```
; 3454  =0*
; 3455  JRSTF:  MEM READ,              ;WAIT FOR DATA
; 3456          [HR]_MEM,              ;STICK IN HR
; 3457          LOAD INST EA,          ;LOAD @ AND XR
; 3458          CALL [JRSTO]           ;COMPUTE EA AGAIN
; 3459          JUMPA                  ;JUMP
; 3460
; 3461  JRSTO:  EA MODE DISP           ;WHAT TYPE OF EA?
; 3462  =100*
; 3463          READ XR,               ;INDEXED
; 3464          LOAD FLAGS,            ;GET FLAGS FROM XR
; 3465          UPDATE USER,           ;ALLOW USER TO SET
; 3466          RETURN [2]             ;ALL DONE
; 3467          READ [HR],             ;PLAIN
; 3468          LOAD FLAGS,            ;LOAD FLAGS FROM INST
; 3469          UPDATE USER,           ;ALLOW USER TO SET
; 3470          RETURN [2]             ;RETURN
; 3471          [HR]_[HR]+XR,          ;BOTH
; 3472          LOAD VMA,              ;FETCH IND WORD
; 3473          START READ,            ;START MEM CYCLE
; 3474          J/JRST1                ;CONTINUE BELOW
; 3475          VMA_[HR],              ;INDIRECT
; 3476          START READ,            ;FETCH IND WORD
; 3477          PXCT EA,               ;SETUP PXCT STUFF
; 3478          J/JRST1                ;CONTINUE BELOW
; 3479  JRST1:  MEM READ,              ;WAIT FOR DATA
; 3480          [HR]_MEM,              ;LOAD THE HR
; 3481          LOAD INST EA,          ;LOAD @ AND XR
; 3482          J/JRSTO                ;LOOP BACK
; 3483
```

```
U 1004, 2740,4551,0202,4374,0007,0700,0000,0077,7740    ; 3484  =0
U 1005, 2625,3441,0301,4174,4007,0700,0000,0000,0000    ; 3485  HALT:   UUO                     ;USER MODE
U 2625, 0104,4751,1217,4374,4007,0700,0000,0000,0001    ; 3486          [PC]_[AR]               ;EXEC MODE--CHANGE PC
                                                        ; 3487          HALT [HALT]             ;HALT INSTRUCTION
                                                        ; 3488
                                                        ; 3489  =0
U 1010, 2740,4551,0202,4374,0007,0700,0000,0077,7740    ; 3490  JRST1O: UUO
U 1011, 0303,4443,0000,4174,4007,0700,0000,0000,0000    ; 3491          J/JEN2                  ;DISMISS INTERRUPT
                                                        ; 3492  =0000
U 0300, 2740,4551,0202,4374,0007,0700,0000,0077,7740    ; 3493  JEN:    UUO                     ; FLAGS
                                                        ; 3494          MEM READ,
                                                        ; 3495          [HR]_MEM,               ;GET INST
                                                        ; 3496          LOAD INST EA,           ;LOAD XR & @
U 0301, 1143,3771,0002,4365,5217,0700,0210,0000,0002    ; 3497          CALL [JRSTO]            ;COMPUTE FLAGS
                                                        ; 3498  =0011
U 0303, 2434,4553,1400,4374,4007,0331,0010,0007,7400    ; 3499  JEN2:   DISMISS                 ;DISMISS INTERRUPT
U 0307, 3611,3770,1416,4344,4007,0700,0010,0000,0000    ; 3500  =0111   CALL LOAD PI            ;RELOAD PI HARDWARE
U 0317, 0110,3441,0301,4170,4156,4700,0200,0014,0012    ; 3501  =1111   JUMPA                   ;GO JUMP
                                                        ; 3502  =
                                                        ; 3503
                                                        ; 3504  1540:
                                                        ; 3505  JFCL:   JFCL FLAGS,             ;ALL DONE IN HARDWARE
                                                        ; 3506          SKIP JFCL,              ;SEE IF SKIPS
                                                        ; 3507          3T,                     ;ALLOW TIME
U 1540, 0762,4443,0000,4174,4467,0551,0000,0001,0010    ; 3508          J/JUMP-                 ;JUMP IF WE SHOULD
                                                        ; 3509
```

```
U 0320, 2740,4551,0202,4374,0007,0700,0000,0077,7740
U 0321, 2434,4553,1400,4374,4007,0331,0010,0007,7400
U 0325, 0335,3333,0012,4174,4437,0700,0000,0000,0000

U 0335, 1525,4223,0000,4364,4277,0700,0200,0000,0010


U 2677, 2712,3771,0005,4365,5007,0700,0200,0000,0002


U 2712, 2713,0111,0703,4174,4007,0700,0200,0004,0012

U 2713, 2714,3771,0001,4361,5007,0700,0200,0000,0002

U 2714, 2715,3333,0005,4174,4467,0700,0000,0001,0004

U 2715, 0305,5551,1313,4374,4007,0700,0000,0001,0000




U 1014, 2740,4551,0202,4374,0007,0700,0000,0077,7740
U 1015, 0060,4521,1205,4074,4007,0700,0000,0000,0000


U 0060, 3670,3443,0300,4174,4007,0700,0210,0003,0012

U 0064, 3671,0111,0703,4170,4007,0700,0210,0003,0012

U 0065, 2677,0111,0703,4174,4007,0700,0200,0004,0002



U 1024, 2740,4551,0202,4374,0007,0700,0000,0077,7740
U 1025, 2716,3443,0300,4174,4007,0700,0200,0003,0012
U 2716, 0435,4521,1203,4074,4007,0700,0000,0000,0000
```

```
; 3510  .TOC    "EXTENDED ADDRESSING INSTRUCTIONS"
; 3511
; 3512  =0000
; 3513  XJEN:   UUO                          ;HERE IF USER MODE
; 3514          DISMISS                      ;CLEAR HIGHEST INTERRUPT
; 3515  =0101   READ [MASK], LOAD PI         ;NO MORE INTERRUPTS
; 3516  =1101   ABORT MEM CYCLE,             ;AVOID INTERRUPT PAGE FAIL
; 3517          J/XJRSTFO                    ;START READING FLAG WORD
; 3518  =
; 3519
; 3520  XJRSTF: MEM READ, [BR]_MEM           ;PUT FLAGS IN BR
; 3521          [AR]_[AR]+1,                 ;INCREMENT ADDRESS
; 3522          LOAD VMA,                    ;PUT RESULT IN VMA
; 3523          START READ                   ;START MEMORY
; 3524          MEM READ, [PC]_MEM,          ;PUT DATA IN PC
; 3525          HOLD LEFT                    ;IGNORE SECTION NUMBER
; 3526          READ [BR], LOAD FLAGS,       ;LOAD NEW FLAGS
; 3527          UPDATE USER                  ;ALSO USER FLAG
; 3528  PISET:  [FLG]_[FLG].AND.NOT.#,       ;CLEAR PI CYCLE
; 3529           FLG.PI/1, J/PIEXIT          ;RELOAD PI HARDWARE
; 3530                                       ; INCASE THIS IS AN
; 3531                                       ; INTERRUPT INSTRUCTION
; 3532
; 3533  =O
; 3534  XPCW:   UUO                          ;USER MODE
; 3535          [BR]_FLAGS                   ;PUT FLAGS IN BR
; 3536  =O*O
; 3537  PIXPCW: VMA_[AR], START WRITE,       ;STORE FLAGS
; 3538          CALL [STOBR]                 ;PUT BR IN MEMORY
; 3539  =1*O    VMA_[AR]+1, LOAD VMA,
; 3540          START WRITE,                 ;PREPEARE TO STORE PC
; 3541          CALL [STOPC]                 ;PUT PC IN MEMORY
; 3542  =1*1    [AR]_[AR]+1,                 ;DO NEW PC PART
; 3543          START READ, J/XJRSTF
; 3544  =
; 3545
; 3546  =O
; 3547  SFM:    UUO
; 3548          VMA_[AR], START WRITE        ;STORE FLAGS
; 3549          [AR]_FLAGS, J/STORE          ;STORE AND EXIT
; 3550
```

```
                                                    ; 3551  .TOC    "XCT"
                                                    ; 3552
                                                    ; 3553          .DCODE
D 0256, 0000,1541,1100                              ; 3554  256:    R,              J/XCT   ;OPERAND FETCHED AS DATA
                                                    ; 3555          .UCODE
                                                    ; 3556
                                                    ; 3557  1541:
U 1541, 1034,4443,0000,4174,4007,0340,0000,0000,0000 ; 3558  XCT:    SKIP KERNEL             ;SEE IF MAY BE PXCT
                                                    ; 3559  =0
                                                    ; 3560  XCT1A:  [HR]_[AR],              ;STUFF INTO HR
                                                    ; 3561          DBUS/DP,                ;PLACE ON DBUS FOR IR
                                                    ; 3562          LOAD INST,              ;LOAD IR, AC, XR, ETC.
                                                    ; 3563          PXCT/E1,                ;ALLOW XR TO BE PREVIOUS
U 1034, 2717,3441,0302,4174,4617,0700,0000,0000,0100 ; 3564          J/XCT1                  ;CONTINUE BELOW
                                                    ; 3565
                                                    ; 3566          READ [HR],              ;LOAD PXCT FLAGS
                                                    ; 3567          LOAD PXCT,              ; ..
U 1035, 1034,3333,0002,4174,4167,0700,0000,0000,0000 ; 3568          J/XCT1A                 ;CONTINUE WITH NORMAL FLOW
                                                    ; 3569
                                                    ; 3570  XCT1:   WORK[YSAVE]_[HR] CLR LH,;SAVE FOR IO INSTRUCTIONS
U 2717, 0366,4713,1202,7174,4007,0700,0400,0000,0422 ; 3571          J/XCT2                  ;GO EXECUTE IT
                                                    ; 3572
```

```
D 0260, 0000,1544,2100
D 0261, 0002,1543,3100
D 0262, 0002,1545,2100
D 0263, 0000,1546,2100

U 1543, 2720,3771,0005,4365,5007,0700,0200,0000,0002

U 2720, 1044,0551,1504,0274,4407,0311,0200,0003,0712

U 1544, 2720,3741,0105,4074,4467,0700,0000,0005,0000

U 1044, 0220,3333,0005,4175,5003,7701,0200,0000,0002

U 1045, 2721,3333,0005,4175,5007,0701,0200,0000,0002

U 2721, 0220,4443,0000,4174,4463,7700,0000,0001,2000

U 0220, 0221,3441,0301,4174,4007,0700,0200,0014,0012

U 0221, 0100,3440,0404,0174,4156,4700,0400,0000,0000

U 0222, 1400,3440,0404,0174,4007,0700,0400,0000,0000
```

```
; 3573  .TOC    "STACK INSTRUCTIONS -- PUSHJ, PUSH, POP, POPJ"
; 3574
; 3575          .DCODE
; 3576  260:    I,      B/0,    J/PUSHJ
; 3577          IR,     B/2,    J/PUSH
; 3578          I,      B/2,    J/POP
; 3579          I,              J/POPJ
; 3580          .UCODE
; 3581
; 3582  ;ALL START WITH E IN AR
; 3583  1543:
; 3584  PUSH:   MEM READ,                ;PUT MEMOP IN BR
; 3585          [BR]_MEM                 ; ..
; 3586  PUSH1:  [ARX]_AC+1000001,         ;BUMP BOTH HALVES OF AC
; 3587          INH CRY18,               ;NO CARRY
; 3588          LOAD VMA,                ;START TO STORE ITEM
; 3589          START WRITE,             ;START MEM CYCLE
; 3590          PXCT STACK WORD,         ;THIS IS THE STACK DATA WORD
; 3591          3T,                      ;ALLOW TIME
; 3592          SKIP CRY0,               ;GO TO STMAC, SKIP IF PDL OV
; 3593          J/STMAC                  ; ..
; 3594
; 3595  1544:
; 3596  PUSHJ:  [BR]_PC WITH FLAGS,       ;COMPUTE UPDATED FLAGS
; 3597          CLR FPD,                 ;CLEAR FIRST-PART-DONE
; 3598          J/PUSH1                  ; AND JOIN PUSH CODE
; 3599
; 3600  =0
; 3601  STMAC:  MEM WRITE,               ;WAIT FOR MEMORY
; 3602          MEM_[BR],                ;STORE BR ON STACK
; 3603          B DISP,                  ;SEE IF PUSH OR PUSHJ
; 3604          J/JSTAC                  ;BELOW
; 3605  ;WE MUST STORE THE STACK WORD PRIOR TO SETTING PDL OV IN CASE OF
; 3606  ; PAGE FAIL.
; 3607          MEM WRITE,               ;WAIT FOR MEMORY
; 3608          MEM_[BR]                 ;STORE BR
; 3609  SETPDL: SET PDL OV,               ;OVERFLOW
; 3610          B DISP,                  ;SEE IF PUSH OR PUSHJ
; 3611          J/JSTAC                  ;BELOW
; 3612
; 3613  =00
; 3614  JSTAC:  [PC]_[AR],                ;PUSHJ--LOAD PC
; 3615          LOAD VMA,                ;LOAD ADDRESS
; 3616          FETCH                    ;GET NEXT INST
; 3617  JSTAC1: AC_[ARX],                 ;STORE BACK STACK PTR
; 3618          NEXT INST                ;DO NEXT INST
; 3619          AC_[ARX],                ;UPDATE STACK POINTER
; 3620          J/DONE                   ;DO NEXT INST
; 3621  =
; 3622
```

```
                                                        ; 3623  1545:
                                                        ; 3624  POP:    [ARX]_AC,               ;GET POINTER
                                                        ; 3625          LOAD VMA,               ;ADDRESS OF STACK WORD
                                                        ; 3626          START READ, 3T,         ;START CYCLE
U 1545, 2722,3771,0004,0276,6007,0701,0200,0004,0712    ; 3627          PXCT STACK WORD         ;FOR PXCT
                                                        ; 3628
                                                        ; 3629          MEM READ,               ;LOAD BR (QUIT IF PAGE FAIL)
U 2722, 2723,3771,0005,4365,5007,0700,0200,0000,0002    ; 3630          [BR]_MEM                ;STACK WORD TO BR
                                                        ; 3631
                                                        ; 3632          [ARX]_[ARX]+#,          ;UPDATE POINTER
                                                        ; 3633          #/777777,               ;-1 IN EACH HALF
                                                        ; 3634          INH CRY18, 3T,          ;BUT NO CARRY
U 2723, 1144,0551,0404,4374,4407,0311,0000,0077,7777    ; 3635          SKIP CRY0               ;SEE IF OVERFLOW
                                                        ; 3636
                                                        ; 3637  =0      VMA_[AR],               ;EFFECTIVE ADDRESS
                                                        ; 3638          PXCT DATA,              ;FOR PXCT
                                                        ; 3639          START WRITE,            ;WHERE TO STORE RESULT
U 1144, 2725,3443,0300,4174,4007,0700,0200,0003,0312    ; 3640          J/POPX1                 ;OVERFLOW
                                                        ; 3641
                                                        ; 3642          VMA_[AR],               ;EFFECTIVE ADDRESS
                                                        ; 3643          PXCT DATA,              ;FOR PXCT
U 1145, 2724,3443,0300,4174,4007,0700,0200,0003,0312    ; 3644          START WRITE             ;WHERE TO STORE RESULT
                                                        ; 3645
                                                        ; 3646          MEM WRITE,              ;WAIT FOR MEM
                                                        ; 3647          MEM_[BR],               ;STORE BR
                                                        ; 3648          B DISP,                 ;POP OR POPJ?
U 2724, 0220,3333,0005,4175,5003,7701,0200,0000,0002    ; 3649          J/JSTAC                 ;STORE POINTER
                                                        ; 3650
                                                        ; 3651
                                                        ; 3652  POPX1:  MEM WRITE,              ;WAIT FOR MEMORY
                                                        ; 3653          MEM_[BR],               ;STORE BR
U 2725, 2721,3333,0005,4175,5007,0701,0200,0000,0002    ; 3654          J/SETPDL                ;GO SET PDL OV
                                                        ; 3655
                                                        ; 3656  1546:
                                                        ; 3657  POPJ:   [ARX]_AC,               ;GET POINTER
                                                        ; 3658          LOAD VMA,               ;POINT TO STACK WORD
                                                        ; 3659          PXCT STACK WORD, 3T,    ;FOR PXCT
U 1546, 2726,3771,0004,0276,6007,0701,0200,0004,0712    ; 3660          START READ              ;START READ
                                                        ; 3661          [ARX]_[ARX]+#,          ;UPDATE POINTER
                                                        ; 3662          #/777777,               ;-1 IN BOTH HALFS
                                                        ; 3663          INH CRY18, 3T,          ;INHIBIT CARRY 18
U 2726, 1146,0551,0404,4374,4407,0311,0000,0077,7777    ; 3664          SKIP CRY0               ;SEE IF OVERFLOW
U 1146, 1147,4443,0000,4174,4467,0700,0000,0001,2000    ; 3665  =0      SET PDL OV              ;SET OVERFLOW
                                                        ; 3666          MEM READ, [PC]_MEM,     ;STICK DATA IN PC
                                                        ; 3667          HOLD LEFT,              ;NO FLAGS
U 1147, 0221,3771,0001,4361,5007,0700,0200,0000,0002    ; 3668          J/JSTAC1                ;STORE POINTER
                                                        ; 3669
```

```
D 0105, 0000,1551,3000

U 1551, 2727,3770,0303,4344,0007,0700,0000,0000,0000
U 2727, 1150,3771,0005,0276,6007,0521,0000,0000,0000
U 1150, 1152,0113,0503,0174,4407,0521,0400,0000,0000
U 1151, 1154,0113,0503,0174,4407,0521,0400,0000,0000
U 1152, 0100,4443,0000,4174,4156,4700,0000,0000,0000
U 1153, 0555,4443,0000,4174,4467,0700,0000,0001,2000
U 1154, 0555,4443,0000,4174,4467,0700,0000,0001,2000
U 1155, 0100,4443,0000,4174,4156,4700,0000,0000,0000
```

```
; 3670  .TOC    "STACK INSTRUCTIONS -- ADJSP"
; 3671
; 3672          .DCODE
; 3673  105:    I-PF,   B/0,            J/ADJSP
; 3674          .UCODE
; 3675
; 3676  1551:
; 3677  ADJSP:  [AR]_[AR] SWAP,         ;MAKE 2 COPIES OF RH
; 3678          HOLD RIGHT
; 3679          [BR]_AC,                ;READ AC, SEE IF MINUS
; 3680          3T,
; 3681          SKIP DPO
; 3682  =0      AC_[BR]+[AR],           ;UPDATE AC
; 3683          INH CRY18,              ;NO CARRY
; 3684          SKIP DPO,               ;SEE IF STILL OK
; 3685          3T,                     ;ALLOW TIME
; 3686          J/ADJSP1                ;TEST FOR OFLO
; 3687          AC_[BR]+[AR],           ;UPDATE AC
; 3688          INH CRY18,              ;NO CARRY
; 3689          SKIP DPO,               ;SEE IF STILL MINUS
; 3690          3T,                     ;ALLOW TIME FOR SKIP
; 3691          J/ADJSP2                ;CONTINUE BELOW
; 3692
; 3693  =0
; 3694  ADJSP1: NEXT INST               ;NO OVERFLOW
; 3695          SET PDL OV,             ;SET PDL OV
; 3696          J/NIDISP                ;GO DO NICOND DISP
; 3697
; 3698  =0
; 3699  ADJSP2: SET PDL OV,             ;SET PDL OV
; 3700          J/NIDISP                ;GO DO NICOND DISP
; 3701          NEXT INST               ;NO OVERFLOW
; 3702
```

```
D 0264, 0000,1552,2100
D 0265, 0000,1550,2100
D 0266, 0000,1554,2100
D 0267, 0000,1555,2100

U 1550, 2730,3741,0105,4074,4007,0700,0000,0000,0000
U 2730, 0764,3440,0505,0174,4467,0700,0400,0005,0000

U 1552, 2731,3741,0105,4074,4467,0700,0000,0005,0000
U 2731, 2732,3443,0300,4174,4007,0700,0200,0003,0012
U 2732, 2733,3333,0005,4175,5007,0701,0200,0000,0002

U 2733, 0107,0551,0301,4370,4007,0701,0000,0000,0001

U 1554, 2734,3441,0305,4174,4007,0700,0200,0003,0002
U 2734, 0130,3770,0304,4344,4007,0700,0000,0000,0000
U 0130, 3076,3771,0003,0276,6007,0700,0010,0000,0000
U 0134, 3672,3441,0104,4170,4007,0700,0010,0000,0000

U 0135, 0107,0551,0501,4370,4007,0701,0000,0000,0001

U 1555, 2735,3771,0005,0276,6007,0700,0000,0000,0000
U 2735, 2736,3770,0505,4344,4007,0700,0000,0000,0000
U 2736, 2737,3443,0500,4174,4007,0700,0200,0004,0012

U 2737, 0274,3771,0005,4365,5007,0700,0200,0000,0002
```

```
; 3703  .TOC    "SUBROUTINE CALL/RETURN -- JSR, JSP, JSA, JRA"
; 3704
; 3705          .DCODE
; 3706  264:    I,              J/JSR
; 3707          I,              J/JSP
; 3708          I,              J/JSA
; 3709          I,              J/JRA
; 3710          .UCODE
; 3711
; 3712  1550:
; 3713  JSP:    [BR]_PC WITH FLAGS      ;GET PC WITH FLAGS
; 3714          CLR FPD,                ;CLEAR FIRST-PART-DONE
; 3715          AC_[BR],                ;STORE FLAGS
; 3716          J/JUMPA                 ;GO JUMP
; 3717
; 3718  1552:
; 3719  JSR:    [BR]_PC WITH FLAGS,     ;GET PC WITH FLAGS
; 3720          CLR FPD                 ;CLEAR FIRST-PART-DONE
; 3721          VMA_[AR],               ;EFFECTIVE ADDRESS
; 3722          START WRITE             ;STORE OLD PC WORD
; 3723          MEM WRITE,              ;WAIT FOR MEMORY
; 3724          MEM_[BR]                ;STORE
; 3725          [PC]_[AR]+1000001,      ;PC _ E+1
; 3726          HOLD LEFT,              ;NO JUNK IN LEFT
; 3727          3T,                     ;ALLOW TIME FOR DBM
; 3728          J/START                 ;START AT E+1
; 3729
; 3730
; 3731  1554:
; 3732  JSA:    [BR]_[AR],              ;SAVE E
; 3733          START WRITE             ;START TO STORE
; 3734          [ARX]_[AR] SWAP         ;ARX LEFT _ E
; 3735  =0*0    [AR]_AC,                ;GET OLD AC
; 3736          CALL [IBPX]             ;SAVE AR IN MEMORY
; 3737  =1*0    [ARX]_[PC],             ;ARX NOW HAS E,,PC
; 3738          HOLD LEFT,              ; ..
; 3739          CALL [AC_ARX]           ;GO PUT ARX IN AC
; 3740  =1*1    [PC]_[BR]+1000001,      ;NEW PC
; 3741          3T,                     ;ALLOW TIME
; 3742          HOLD LEFT,              ;NO JUNK IN PC LEFT
; 3743          J/START                 ;START AT E+1
; 3744  =
; 3745
; 3746  1555:
; 3747  JRA:    [BR]_AC                 ;GET AC
; 3748          [BR]_[BR] SWAP          ;OLD E IN BR RIGHT
; 3749          VMA_[BR],               ;LOAD VMA
; 3750          START READ              ;FETCH SAVED AC
; 3751          MEM READ,               ;WAIT FOR MEMORY
; 3752          [BR]_MEM,               ;LOAD BR WITH SAVE AC
; 3753          J/JMPA                  ;GO JUMP
; 3754
```

```
D 0030, 0000,1557,2100
D 0031, 0001,1557,2100
D 0032, 0002,1557,2100
D 0033, 0003,1557,2100
D 0034, 0004,1557,2100
D 0035, 0005,1557,2100
D 0036, 0006,1557,2100
D 0037, 0007,1557,2100

D 0040, 0000,1556,2100
D 0041, 0000,1556,2100
D 0042, 0000,1556,2100
D 0043, 0000,1556,2100
D 0044, 0000,1556,2100
D 0045, 0000,1556,2100
D 0046, 0000,1556,2100
D 0047, 0000,1556,2100
D 0050, 0000,1556;2100
D 0051, 0000,1556,2100
D 0052, 0000,1556,2100
D 0053, 0000,1556,2100
D 0054, 0000,1556,2100
D 0055, 0000,1556,2100
D 0056, 0000,1556,2100
D 0057, 0000,1556,2100
D 0060, 0000,1556,2100
D 0061, 0000,1556,2100
D 0062, 0000,1556,2100
D 0063, 0000,1556,2100
D 0064, 0000,1556,2100
D 0065, 0000,1556,2100
D 0066, 0000,1556,2100
D 0067, 0000,1556,2100
D 0070, 0000,1556,2100
D 0071, 0000,1556,2100
D 0072, 0000,1556,2100
D 0073, 0000,1556,2100
D 0074, 0000,1556,2100
D 0075, 0000,1556,2100
D 0076, 0000,1556,2100
D 0077, 0000,1556,2100

D 0100, 0000,1556,2100
D 0101, 0000,1661,2100
D 0102, 0000,1662,2100
D 0103, 0000,1663,2100
```

```
; 3755  .TOC    "ILLEGAL INSTRUCTIONS AND UUO'S"
; 3756  ;LUUO'S TRAP TO CURRENT CONTEXT
; 3757
; 3758          .DCODE
; 3759  030:    I,      B/0,    J/LUUO
; 3760          I,      B/1,    J/LUUO
; 3761          I,      B/2,    J/LUUO
; 3762          I,      B/3,    J/LUUO
; 3763          I,      B/4,    J/LUUO
; 3764          I,      B/5,    J/LUUO
; 3765          I,      B/6,    J/LUUO
; 3766          I,      B/7,    J/LUUO
; 3767
; 3768  ;MONITOR UUO'S -- TRAP TO EXEC
; 3769
; 3770  040:    I,              J/MUUO          ;CALL
; 3771          I,              J/MUUO          ;INIT
; 3772          I,              J/MUUO
; 3773          I,              J/MUUO
; 3774          I,              J/MUUO
; 3775          I,              J/MUUO
; 3776          I,              J/MUUO
; 3777          I,              J/MUUO          ;CALLI
; 3778          I,              J/MUUO          ;OPEN
; 3779          I,              J/MUUO          ;TTCALL
; 3780          I,              J/MUUO
; 3781          I,              J/MUUO
; 3782          I,              J/MUUO
; 3783          I,              J/MUUO          ;RENAME
; 3784          I,              J/MUUO          ;IN
; 3785          I,              J/MUUO          ;OUT
; 3786          I,              J/MUUO          ;SETSTS
; 3787          I,              J/MUUO          ;STATO
; 3788          I,              J/MUUO          ;GETSTS
; 3789          I,              J/MUUO          ;STATZ
; 3790          I,              J/MUUO          ;INBUF
; 3791          I,              J/MUUO          ;OUTBUF
; 3792          I,              J/MUUO          ;INPUT
; 3793          I,              J/MUUO          ;OUTPUT
; 3794          I,              J/MUUO          ;CLOSE
; 3795          I,              J/MUUO          ;RELEAS
; 3796          I,              J/MUUO          ;MTAPE
; 3797          I,              J/MUUO          ;UGETF
; 3798          I,              J/MUUO          ;USETI
; 3799          I,              J/MUUO          ;USETO
; 3800          I,              J/MUUO          ;LOOKUP
; 3801          I,              J/MUUO          ;ENTER
; 3802
; 3803  ;EXPANSION OPCODES
; 3804
; 3805  100:    I,              J/UUO           ;UJEN
; 3806          I,              J/UUO101
; 3807          I,              J/UUO102
; 3808          I,              J/UUO103
; 3809
```

```
D 0000, 0000,1556,2100
D 0104, 0000,1664,2100
D 0106, 0000,1666,2100
D 0107, 0000,1667,2100
D 0130, 0000,1660,2100
D 0131, 0001,1660,2100
D 0141, 0002,1660,2100
D 0151, 0003,1660,2100
D 0161, 0004,1660,2100
D 0171, 0005,1660,2100
D 0247, 0000,1665,2100


U 1661, 2740,4551,0202,4374,0007,0700,0000,0077,7740

U 1662, 2740,4551,0202,4374,0007,0700,0000,0077,7740

U 1663, 2740,4551,0202,4374,0007,0700,0000,0077,7740

U 1664, 2740,4551,0202,4374,0007,0700,0000,0077,7740

U 1666, 2740,4551,0202,4374,0007,0700,0000,0077,7740

U 1667, 2740,4551,0202,4374,0007,0700,0000,0077,7740

U 1660, 2740,4551,0202,4374,0007,0700,0000,0077,7740

U 1665, 2740,4551,0202,4374,0007,0700,0000,0077,7740
```

ILLEGAL INSTRUCTIONS AND UUO'S

```
; 3810  ;RESERVED OPCODES
; 3811
; 3812  000:     I,              J/UUO
; 3813  104:     I,              J/JSYS          ;JSYS
; 3814  106:     I,              J/UUO106
; 3815           I,              J/UUO107
; 3816  130:     I,      B/0,    J/FP-LONG       ;UFA
; 3817           I,      B/1,    J/FP-LONG       ;DFN
; 3818  141:     I,      B/2,    J/FP-LONG       ;FADL
; 3819  151:     I,      B/3,    J/FP-LONG       ;FSBL
; 3820  161:     I,      B/4,    J/FP-LONG       ;FMPL
; 3821  171:     I,      B/5,    J/FP-LONG       ;FDVL
; 3822  247:     I,              J/UUO247        ;RESERVED
; 3823           .UCODE
; 3824
; 3825  1661:
; 3826  UUO101: UUO
; 3827  1662:
; 3828  UUO102: UUO
; 3829  1663:
; 3830  UUO103: UUO
; 3831  1664:
; 3832  JSYS:   UUO
; 3833  1666:
; 3834  UUO106: UUO
; 3835  1667:
; 3836  UUO107: UUO
; 3837  1660:
; 3838  FP-LONG:UUO
; 3839  1665:
; 3840  UUO247: UUO
; 3841
```

```
U 1556, 2740,4551,0202,4374,0007,0700,0000,0077,7740
U 2740, 1156,4751,1204,4374,4007,0700,0000,0000,0424

U 1156, 1511,0111,1104,4174,4007,0700,0010,0000,0000
U 1157, 1160,3333,0010,4174,4007,0520,0000,0000,0000


U 1160, 0310,3333,0004,4174,4007,0700,0200,0021,1016
U 1161, 1162,3770,0203,4344,4007,0700,0000,0000,0000

U 1162, 2746,4521,1203,4074,0007,0700,0010,0000,0000

U 1163, 0314,3333,0004,4174,4007,0700,0200,0021,1016
U 0314, 2747,3333,0003,4175,5007,0701,0210,0000,0002
U 0316, 0020,3333,0001,4175,5007,0701,0200,0000,0002
U 0020, 2747,4221,0002,4174,0007,0700,0010,0000,0000



U 0022, 3542,3333,0002,4175,5007,0701,0210,0000,0002

U 0023, 3670,0111,0704,4170,4007,0700,0210,0023,1016
```

```
; 3842  ;HERE FOR UUO'S WHICH TRAP TO EXEC
; 3843  1556:
; 3844  UUO:     ;THIS TAG IS USED FOR ILLEGAL THINGS WHICH DO UUO TRAPS
; 3845  MUUO:    ;THIS TAG IS USED FOR MONITOR CALL INSTRUCTIONS
; 3846          [HR]_[HR].AND.#,          ;MASK OUT @ AND XR
; 3847          #/777740,                 ;MASK
; 3848          HOLD RIGHT                ;KEEP RIGHT
; 3849  ;THE UUO MACRO DOES THE ABOVE INSTRUCTION AND GOES TO UUOGO
; 3850  UUOGO:  [ARX]_0 XWD [424]         ;HERE FROM UUO MACRO
; 3851                                    ;GET OFFSET TO UPT
; 3852  =0      [ARX]_[ARX]+[UBR],        ;ADDRESS OF MUUO WORD
; 3853          CALL [ABORT]              ;STOP MEMORY
; 3854          READ [EBR],               ;KL PAGING
; 3855          SKIP DPO                  ; ??
; 3856  =0      READ [ARX],               ;GET THE ADDRESS
; 3857          LOAD VMA,                 ;START WRITE
; 3858          VMA PHYSICAL WRITE,       ;ABSOLUTE ADDRESS
; 3859          J/KIMUUO                  ;GO STORE KI STYLE
; 3860          [AR]_[HR] SWAP            ;PUT IN RIGHT HALF
; 3861  =0      [AR]_FLAGS,               ;FLAGS IN LEFT HALF
; 3862          HOLD RIGHT,               ;JUST WANT FLAGS
; 3863          CALL [UUOFLG]             ;CLEAR TRAP FLAGS
; 3864          READ [ARX],               ;LOOK AT ADDRESS
; 3865          LOAD VMA,                 ;LOAD THE VMA
; 3866          VMA PHYSICAL WRITE        ;STORE FLAG WORD
; 3867  =0*     MEM WRITE,                ;WAIT FOR MEMORY
; 3868          MEM_[AR], CALL [NEXT]     ;STORE
; 3869          MEM WRITE,                ;WAIT FOR MEMORY
; 3870          MEM_[PC]                  ;STORE FULL WORD PC
; 3871  =000    [HR]_0,                   ;SAVE E
; 3872          HOLD RIGHT, CALL [NEXT]   ;BUT CLEAR OPCODE
; 3873  =010
; 3874  UUOPCW: MEM WRITE,                ;WAIT FOR MEMORY
; 3875          MEM_[HR],                 ;STORE INSTRUCTION IN KI
; 3876                                    ; OR FULL WORD E IN KL
; 3877          CALL [GETPCW]             ;GET PROCESS-CONTEXT-WORD
; 3878
; 3879  =011    NEXT [ARX] PHYSICAL WRITE, ;POINT TO NEXT WORD
; 3880          CALL [STOBR]              ;STORE PROCESS CONTEXT WORD
; 3881
```

```
                                                        ;NOW WE MUST PICK ONE OF 8 NEW PC WORDS BASED ON PC FLAGS
U 0027, 2741,4751,1205,4374,4007,0700,0000,0000,0430    =111    [BR]_0 XWD [430]        ;OFFSET INTO UPT
                                                        =
U 2741, 2742,0111,1105,4174,4007,0700,0000,0000,0000            [BR]_[BR]+[UBR]         ;ADDRESS OF WORD
U 2742, 2743,4521,1203,4074,4007,0700,0000,0000,0000            [AR]_FLAGS              ;GET FLAGS
                                                                TL [AR],                ;LOOK AT FLAGS
U 2743, 1164,4553,0300,4374,4007,0321,0000,0000,0600            #/600                   ;TRAP SET?
                                                        =0      [BR]_[BR].OR.#,         ;YES--POINT TO TRAP CASE
                                                                #/1,                    ; ..
U 1164, 1165,3551,0505,4370,4007,0700,0000,0000,0001            HOLD LEFT               ;LEAVE LEFT ALONE
                                                                TL [AR],                ;USER OR EXEC
U 1165, 1166,4553,0300,4374,4007,0321,0000,0001,0000            #/10000                 ; ..
                                                        =0      [BR]_[BR].OR.#,         ;USER
                                                                #/4,                    ;POINT TO USER WORDS
U 1166, 1167,3551,0505,4370,4007,0700,0000,0000,0004            HOLD LEFT
                                                                READ [BR],              ;LOOK AT ADDRESS
                                                                LOAD VMA,               ;PLACE IN VMA
                                                                VMA PHYSICAL,           ;PHYSICAL ADDRESS
U 1167, 2744,3333,0005,4174,4007,0700,0200,0024,1016            START READ              ;GET NEW PC WORD
                                                        GOEXEC: MEM READ,               ;WAIT FOR DATA
U 2744, 2745,3771,0003,4365,5007,0700,0200,0000,0002            [AR]_MEM                ;STICK IN AR
                                                                READ [AR],              ;LOOK AT DATA
                                                                LOAD FLAGS,             ;LOAD NEW FLAGS
                                                                LEAVE USER,             ;ALLOW USER TO LOAD
                                                                LOAD PCU,               ;SET PCU FROM USER
U 2745, 0764,3333,0003,4174,4467,0700,0000,0000,0404            J/JUMPA                 ;JUMP


                                                        ;HERE FOR TOPS-10 STYLE PAGING

                                                        =00
                                                        KIMUUO: MEM WRITE,              ;STORE INSTRUCTION
U 0310, 2747,3333,0002,4175,5007,0701,0210,0000,0002            MEM_[HR], CALL [NEXT]   ;IN MEMORY
                                                        =10     [AR]_PC WITH FLAGS,     ;GET PC WORD
U 0312, 2746,3741,0103,4074,4007,0700,0010,0000,0000            CALL [UUOFLG]           ;CLEAR TRAP FLAGS
                                                        =11     MEM WRITE,              ;STORE PC WORD
                                                                MEM_[AR],               ; ..
U 0313, 0022,3333,0003,4175,5007,0701,0200,0000,0002            J/UUOPCW                ;GO STORE PROCESS CONTEXT

                                                        UUOFLG: [AR]_[AR].AND.NOT.#,    ;CLEAR TRAP FLAGS
                                                                #/600, HOLD RIGHT,      ; IN WORD TO SAVE
U 2746, 0001,5551,0303,4374,0004,1700,0000,0000,0600            RETURN [1]              ; BACK TO CALLER

                                                        NEXT:   NEXT [ARX] PHYSICAL WRITE, ;POINT TO NEXT WORD
U 2747, 0002,0111,0704,4170,4004,1700,0200,0023,1016            RETURN [2]
```

```
U 1557, 0400,4751,1203,4374,4007,0700,0000,0000,0040

U 0400, 2750,3333,0003,4174,4007,0700,0200,0003,0012

U 2750, 2751,4551,0202,4374,0007,0700,0000,0077,7740

U 2751, 2752,3333,0002,4175,5007,0701,0200,0000,0002

U 2752, 2503,0111,0703,4170,4007,0700,0200,0004,0012
```

```
; 3928  ;HERE FOR LUUO'S
; 3929  1557:
; 3930  LUUO:   [AR]_0 XWD [40]          ;AR GET CONSTANT 40
; 3931  ;THE LUUO MACRO DOES THE ABOVE INSTRUCTION AND GOES TO LUUO1
; 3932  400:                             ;FOR SIMULATOR
; 3933  LUUO1:  READ [AR],               ;LOAD 40 INTO
; 3934          LOAD VMA,                ; THE VMA AND
; 3935          START WRITE              ; PREPARE TO STORE
; 3936          [HR]_[HR].AND.#,         ;CLEAR OUT INDEX AND @
; 3937          #/777740,                ; ..
; 3938          HOLD RIGHT
; 3939          MEM WRITE,               ;STORE LUUO IN 40
; 3940          MEM_[HR]
; 3941          VMA_[AR]+1,              ;POINT TO 41
; 3942          LOAD VMA,                ;PUT 41 IN VMA
; 3943          START READ,              ;START FETCH
; 3944          J/CONT1                  ;GO EXECUTE THE INSTRUCTION
; 3945
```

```
                                                    ; 3946  .TOC    "ARITHMETIC -- ADD, SUB"
                                                    ; 3947
                                                    ; 3948          .DCODE
D 0270, 1015,1560,1100                              ; 3949  270:    R-PF,   AC,     J/ADD
D 0271, 0015,1560,3000                              ; 3950          I-PF,   AC,     J/ADD
D 0272, 0016,1560,1700                              ; 3951          RW,     M,      J/ADD
D 0273, 0017,1560,1700                              ; 3952          RW,     B,      J/ADD
                                                    ; 3953          .UCODE
                                                    ; 3954
                                                    ; 3955  1560:
                                                    ; 3956  ADD:    [AR]_[AR]+AC,           ;DO THE ADD
U 1560, 1500,0551,0303,0274,4463,7701,0200,0001,0001; 3957          AD FLAGS EXIT, 3T       ;UPDATE CARRY FLAGS
                                                    ; 3958                                  ;STORE ANSWER
                                                    ; 3959                                  ;MISSES 3-TICKS BY 3 NS.
                                                    ; 3960
                                                    ; 3961
                                                    ; 3962          .DCODE
D 0274, 1015,1561,1100                              ; 3963  274:    R-PF,   AC,     J/SUB
D 0275, 0015,1561,3000                              ; 3964          I-PF,   AC,     J/SUB
D 0276, 0016,1561,1700                              ; 3965          RW,     M,      J/SUB
D 0277, 0017,1561,1700                              ; 3966          RW,     B,      J/SUB
                                                    ; 3967          .UCODE
                                                    ; 3968
                                                    ; 3969  1561:
                                                    ; 3970  SUB:    [AR]_AC-[AR],           ;DO THE SUBTRACT
U 1561, 1500,2551,0303,0274,4463,7701,4200,0001,0001; 3971          AD FLAGS EXIT, 3T       ;UPDATE PC CARRY FLAGS
                                                    ; 3972                                  ;ALL DONE
                                                    ; 3973                                  ;MISSES 3-TICKS BY 3 NS.
                                                    ; 3974
```

```
D 0114, 0205,1457,1100
D 0115, 0205,1615,1100



U 1457, 1170,0551,0404,1274,4007,0562,0000,0000,1441



U 1170, 2754,0551,0303,0274,4467,0702,4000,0001,0001
U 1171, 2753,7441,1205,4174,4007,0700,0000,0000,0000

U 2753, 1170,3111,0503,4170,4007,0700,0000,0000,0000



U 1615, 1172,2551,0404,1274,4007,0562,4000,0000,1441

U 1172, 2754,2551,0303,0274,4467,0702,0000,0001,0001

U 1173, 2754,2551,0303,0274,4467,0702,4000,0001,0001

U 2754, 1174,3770,0303,4174,0007,0520,0000,0000,0000
U 1174, 1404,4551,0404,4374,0007,0700,0000,0037,7777
U 1175, 1404,3551,0404,4374,0007,0700,0000,0040,0000
```

ARITHMETIC -- DADD, DSUB

```
; 3975  .TOC    "ARITHMETIC -- DADD, DSUB"
; 3976
; 3977          .DCODE
; 3978  114:    DBL R,  DAC,    J/DADD
; 3979          DBL R,  DAC,    J/DSUB
; 3980          .UCODE
; 3981
; 3982  1457:
; 3983  DADD:   [ARX]_[ARX]+AC[1], 4T,  ;ADD LOW WORDS
; 3984          SKIP CRY1               ;SEE IF CARRY TO HIGH WORD
; 3985  =0
; 3986  DADD1:  [AR]_[AR]+AC,           ;ADD HIGH WORDS
; 3987          ADD .25,                ;ADD IN ANY CARRY FROM LOW WORD
; 3988          AD FLAGS, 4T,           ;UPDATE PC FLAGS
; 3989          J/CPYSGN                ;COPY SIGN TO LOW WORD
; 3990          [BR]_.NOT.[MASK]        ;SET BITS 35 AND 36 IN
; 3991          [AR]_[AR].OR.[BR],      ; AR SO THAT ADD .25 WILL
; 3992          HOLD LEFT, J/DADD1      ; ADD 1.
; 3993
; 3994  1615:
; 3995  DSUB:   [ARX]_AC[1]-[ARX], 4T,  ;SUBTRACT LOW WORD
; 3996          SKIP CRY1               ;SEE IF CARRY
; 3997  =0      [AR]_AC-[AR]-.25,       ;NO CARRY
; 3998          AD FLAGS, 4T,           ;UPDATE PC FLAGS
; 3999          J/CPYSGN                ;GO COPY SIGN
; 4000          [AR]_AC-[AR], 4T,       ;THERE WAS A CARRY
; 4001          AD FLAGS                ;UPDATE CARRY FLAGS
; 4002
; 4003  CPYSGN: FIX [AR] SIGN, SKIP DPO
; 4004  =0      [ARX]_[ARX].AND.#, #/377777, HOLD RIGHT, J/MOVE
; 4005          [ARX]_[ARX].OR.#, #/400000, HOLD RIGHT, J/MOVE
; 4006
; 4007
```

```
D 0220, 1015,1641,1100
D 0221, 0015,1641,3000
D 0222, 0016,1641,1700
D 0223, 0017,1641,1700


U 1641, 2755,3441,0306,0174,4007,0700,0000,0000,0000
U 2755, 0021,3772,0000,0275,5007,0700,2000,0071,0043

U 0021, 3001,3446,0606,4174,4007,0700,0010,0000,0000
U 0025, 1176,3333,0004,4174,4007,0621,0000,0000,0000
U 1176, 2756,3445,0404,4174,4007,0700,0000,0000,0000
U 1177, 1500,3221,0003,4174,4003,7700,0200,0003,0001


U 2756, 1200,5113,0412,4174,4007,0621,0000,0000,0000

U 1200, 1202,3770,0404,4174,0007,0520,0000,0000,0000
U 1201, 1500,7001,0003,4174,4003,7700,0200,0003,0001

U 1202, 1404,3221,0003,4174,4467,0700,0000,0041,1000
U 1203, 1404,7001,0003,4174,4467,0700,0000,0041,1000



D 0224, 1005,1571,1100
D 0225, 0005,1571,3000
D 0226, 0016,1571,1700
D 0227, 0006,1571,1700



U 1571, 2757,3442,0300,0174,4007,0700,0000,0000,0000
U 2757, 2760,3441,0316,4174,4007,0700,0000,0000,0000
U 2760, 0031,3771,0006,0276,6007,0700,2000,0071,0043

U 0031, 3001,3446,0606,4174,4007,0700,0010,0000,0000
U 0035, 2761,3445,0403,4174,4007,0700,0000,0000,0000
U 2761, 1204,3770,0303,4174,0007,0520,0000,0000,0000

U 1204, 1500,4001,0004,4174,4003,7700,0200,0003,0001
U 1205, 1206,4113,0616,4174,4007,0520,0000,0000,0000
U 1206, 1500,7001,0004,4174,4003,7700,0200,0003,0001

U 1207, 1404,7001,0004,4174,4467,0700,0000,0041,1000
```

```
; 4008  .TOC    "ARITHMETIC -- MUL, IMUL"
; 4009
; 4010          .DCODE
; 4011  220:    R-PF,    AC,     J/IMUL
; 4012          I-PF,    AC,     J/IMUL
; 4013          RW,      M,      J/IMUL
; 4014          RW,      B,      J/IMUL
; 4015          .UCODE
; 4016  1641:
; 4017  IMUL:   [BRX]_[AR], AC              ;COPY C(E)
; 4018          Q_AC, SC_35.                ;GET THE AC
; 4019  =0**    [BRX]_[BRX]*.5 LONG,        ;SHIFT RIGHT
; 4020          CALL [MULSUB]               ;MULTIPLY
; 4021          READ [ARX], SKIP AD.EQ.0 ;SEE IF FITS
; 4022  =0      [ARX]_[ARX]*2, J/IMUL2      ;NOT ZERO--SHIFT LEFT
; 4023  IMUL1:  [AR]_Q, EXIT                ;POSITIVE
; 4024
; 4025  IMUL2:  [MASK].AND.NOT.[ARX],       ;SEE IF ALL SIGN BITS
; 4026          SKIP AD.EQ.0                ; ..
; 4027  =0      FIX [ARX] SIGN,             ;NOT ALL SIGN BITS
; 4028          SKIP DPO, J/IMUL3           ;GIVE + OR - OVERFLOW
; 4029          [AR]_[MAG].EQV.Q, EXIT      ;NEGATIVE
; 4030  =0
; 4031  IMUL3:  [AR]_Q, SET AROV, J/MOVE
; 4032          [AR]_[MAG].EQV.Q, SET AROV, J/MOVE
; 4033
; 4034
; 4035          .DCODE
; 4036  224:    R-PF,    DAC,    J/MUL
; 4037          I-PF,    DAC,    J/MUL
; 4038          RW,      M,      J/MUL
; 4039          RW,      DBL B,  J/MUL
; 4040          .UCODE
; 4041
; 4042
; 4043  1571:
; 4044  MUL:    Q_[AR], AC                  ;COPY C(E)
; 4045          [TO]_[AR]                   ;SAVE FOR OVERFLOW TEST
; 4046          [BRX]_AC, SC_35.            ;GET THE AC
; 4047  =0**    [BRX]_[BRX]*.5 LONG,        ;SHIFT OVER
; 4048          CALL [MULSUB]               ;MULTIPLY
; 4049          [AR]_[ARX]*2                ;SHIFT OVER
; 4050          FIX [AR] SIGN, SKIP DPO ;SEE IF NEGATIVE
; 4051  =0      [ARX]_[MAG].AND.Q,          ;POSITIVE
; 4052          EXIT
; 4053          [TO].AND.[BRX], SKIP DPO ;TRIED TO SQUARE 1BO?
; 4054  =0      [ARX]_[MAG].EQV.Q, EXIT ;NO
; 4055          [ARX]_[MAG].EQV.Q,          ;YES
; 4056          SET AROV, J/MOVE
; 4057
; 4058
```

```
D 0116, 0205,1566,1100

U 1566, 2762,3447,0303,4174,4007,0700,0000,0000,0000
U 2762, 2763,4117,0004,4174,4007,0700,0000,0000,0000

U 2763, 0120,3441,0405,4174,4007,0350,0000,0000,0000

U 0120, 2775,4557,0006,1274,4007,0701,0010,0000,1441

U 0121, 2765,4557,0004,1274,4007,0701,0000,0000,1442
U 0124, 0171,3223,0000,1174,4007,0700,0400,0000,1443

U 0171, 0563,3442,0300,4174,4007,0700,2010,0071,0043
U 0175, 2764,3441,0416,4174,4007,0700,0000,0000,0000

U 2764, 0410,3227,0004,1174,4007,0700,0400,0000,1442
U 2765, 0410,3777,0016,1276,6007,0701,0000,0000,1441

U 0410, 2776,3777,0006,0274,4007,0701,0010,0000,0000

U 0411, 3000,0113,1616,1174,4007,0701,0400,0000,1441
U 0414, 2766,3223,0000,1174,4007,0700,0400,0000,1442

U 2766, 0543,0111,1604,4174,4007,0700,0000,0000,0000

U 0543, 0563,3442,0300,4174,4007,0700,2010,0071,0043

U 0547, 2767,3445,0404,4174,4467,0700,0000,0005,0000
```

```
; 4059  .TOC    "ARITHMETIC -- DMUL"
; 4060
; 4061          .DCODE
; 4062  116:    DBL R,  DAC,                J/DMUL
; 4063          .UCODE
; 4064
; 4065  .IF/FULL
; 4066  1566:
; 4067  DMUL:   [AR]_[AR]*.5                ;SHIFT MEM OPERAND RIGHT
; 4068          [ARX]_([ARX].AND.[MAG])*.5
; 4069          [BR]_[ARX],                 ;COPY LOW WORD
; 4070          SKIP FPD                    ;SEE IF FIRST PART DONE
; 4071  ;
; 4072  ; BRX * BR ==> C(E+1) * C(AC+1)
; 4073  ;
; 4074  =000    [BRX]_(AC[1].AND.[MAG])*.5, 3T, ;GET LOW AC
; 4075          CALL [DMULGO]               ;START MULTIPLY
; 4076          [ARX]_(AC[2].AND.[MAG])*.5, 3T, ;FIRST PART DONE
; 4077          J/DMUL1                     ;GO DO SECOND PART
; 4078  =100    AC[3]_Q                     ;SALT AWAY 1 WORD OF PRODUCT
; 4079  =
; 4080  ;
; 4081  ; BRX * Q ==> C(E) * C(AC+1)
; 4082  ;
; 4083  =0**    Q_[AR], SC_35.,             ;GO MULT NEXT HUNK
; 4084          CALL [QMULT]                ; ..
; 4085          [TO]_[ARX]                  ;SAVE PRODUCT
; 4086          AC[2]_Q, [ARX]_Q*.5,        ;SAVE PRODUCT
; 4087          J/DMUL2                     ;GO DO HIGH HALF
; 4088  DMUL1:  [TO]_AC[1]*.5               ;RESTORE TO
; 4089  =0*0
; 4090  ;
; 4091  ; BRX * BR ==> C(AC) * C(E+1)
; 4092  ;
; 4093  DMUL2:  [BRX]_AC*.5,                ;PREPARE TO DO HIGH HALF
; 4094          CALL [DBLMUL]               ; GO DO IT
; 4095          AC[1]_[TO]*2, 3T,           ;INTERRUPT, SAVE TO
; 4096          J/DMLINT                    ;SET FPD AND INTERRUPT
; 4097          AC[2]_Q                     ;SAVE PRODUCT
; 4098  =
; 4099          [ARX]_[ARX]+[TO]            ;PREPARE FOR LAST MUL
; 4100  ;
; 4101  ; BRX * Q ==> C(AC) * C(E)
; 4102  ;
; 4103  =0**    Q_[AR], SC_35.,             ;DO THE LAST MULTIPLY
; 4104          CALL [QMULT]                ; GO DO IT
; 4105          [ARX]_[ARX]*2,              ;SHIFT BACK
; 4106          CLR FPD                     ;CLEAR FPD
; 4107
```

```
U 2767, 1210,3770,0404,0174,4007,0520,0400,0000,0000
U 1210, 2774,3223,0000,1174,4007,0700,0400,0000,1441
U 1211, 2770,7003,0000,1174,4007,0700,0400,0000,1441
U 2770, 2771,3772,0000,1275,5007,0701,0000,0000,1442
U 2771, 2772,7003,0000,1174,4007,0700,0400,0000,1442
U 2772, 2773,3772,0000,1275,5007,0701,0000,0000,1443
U 2773, 2774,7003,0000,1174,4007,0700,0400,0000,1443

U 2774, 1212,3741,0103,4074,4007,0520,0000,0000,0000
U 1212, 0110,3443,0100,4174,4156,4700,0200,0014,0012
U 1213, 1400,4443,0000,4174,4467,0700,0000,0041,1000



U 2775, 2776,4221,0004,4174,4007,0700,0000,0000,0000
U 2776, 2777,3442,0500,4174,4007,0700,2000,0071,0043
U 2777, 0563,3447,0606,4174,4007,0700,0000,0000,0000


U 0563, 3003,3446,1200,4174,4007,0700,0010,0000,0000

U 0567, 0004,0113,0404,4174,4464,1701,0000,0001,0001

U 3000, 2675,4443,0000,4174,4467,0700,0000,0003,0000
```

```
; 4108          AC_[ARX] TEST, SKIP DPO ;PUT BACK INTO AC
; 4109  =0       AC[1]_Q, J/DMTRAP       ;POSITIVE
; 4110          AC[1]_[MAG].EQV.Q       ;NEGATIVE
; 4111          Q_AC[2]
; 4112          AC[2]_[MAG].EQV.Q
; 4113          Q_AC[3]
; 4114          AC[3]_[MAG].EQV.Q
; 4115  DMTRAP: [AR]_PC WITH FLAGS,     ;LOOK AT FLAGS
; 4116          SKIP DPO                ;SEE IF AROV SET?
; 4117  =0       DONE                    ;NO--ALL DONE
; 4118          SET AROV, J/DONE        ;YES--FORCE TRAP 1 ALSO
; 4119
; 4120
; 4121  ;WAYS TO CALL MULTIPLY
; 4122  DMULGO: [ARX]_0                 ;CLEAR ARX
; 4123  DBLMUL: Q_[BR], SC_35.
; 4124          [BRX]_[BRX]*.5
; 4125  =0**
; 4126  QMULT:  Q_Q*.5,
; 4127          CALL [MULTIPLY]
; 4128          [ARX]+[ARX], AD FLAGS,  ;TEST FOR OVERFLOW
; 4129          3T, RETURN [4]          ;AND RETURN
; 4130
; 4131  DMLINT: SET FPD, J/FIXPC        ;SET FPD, BACKUP PC
; 4132                                  ; INTERRUPT
;;4133  .IFNOT/FULL
;;4134  1566:
;;4135  DMUL:   UUO
; 4136  .ENDIF/FULL
; 4137
```

```
                                        ; 4138  ;MULTIPLY SUBROUTINE
                                        ; 4139  ;ENTERED WITH:
                                        ; 4140  ;       MULTIPLIER IN Q
                                        ; 4141  ;       MULTIPLICAND IN BRX
                                        ; 4142  ;RETURNS 4 WITH PRODUCT IN ARX!Q
                                        ; 4143
                                        ; 4144  MUL STEP        "A/BRX,B/ARX,DEST/Q_Q*.5,ASHC,STEP SC,MUL DISP"
                                        ; 4145  MUL FINAL       "A/BRX,B/ARX,DEST/Q_Q*2"
                                        ; 4146
U 3001, 3002,3446,0606,4174,4007,0700,0000,0000,0000    ; 4147  MULSUB: [BRX]_[BRX]*.5 LONG
                                        ; 4148  MULSB1: [ARX]_0*.5 LONG,        ;CLEAR ARX AND SHIFT Q
                                        ; 4149          STEP SC,                ;COUNT FIRST STEP
U 3002, 0122,4226,0004,4174,4007,0630,2000,0060,0000    ; 4150          J/MUL+                  ;ENTER LOOP
                                        ; 4151
                                        ; 4152  ;MULTIPLY SUBROUTINE
                                        ; 4153  ;ENTERED WITH:
                                        ; 4154  ;       MULTIPLIER IN Q
                                        ; 4155  ;       MULTIPLICAND IN BRX
                                        ; 4156  ;       PARTIAL PRODUCT IN ARX
                                        ; 4157  ;RETURNS 4 WITH Q*BRX+ARX IN ARX!Q
                                        ; 4158
                                        ; 4159  MULTIPLY:
                                        ; 4160          Q_Q*.5,                 ;SHIFT Q
                                        ; 4161          STEP SC,                ;COUNT FIRST STEP
U 3003, 0122,3446,1200,4174,4007,0630,2000,0060,0000    ; 4162          J/MUL+                  ;ENTER LOOP
                                        ; 4163
```

```
                                                    ; 4164  ;HERE FOR POSITIVE STEPS
                                                    ; 4165  =010                            ;0 IN A POSITIVE STEP
                                                    ; 4166  MUL+:   AD/B,                   ;DON'T ADD
                                                    ; 4167          MUL STEP,               ;SHIFT
U 0122, 0122,3336,0604,4174,4046,2630,2000,0060,0000  ; 4168          J/MUL+                  ;KEEP POSITIVE
                                                    ; 4169  =011                            ;DONE
                                                    ; 4170          AD/B,                   ;DON'T ADD
                                                    ; 4171          MUL FINAL,              ;SHIFT
U 0123, 0004,3334,0604,4174,4004,1700,0000,0000,0000  ; 4172          RETURN [4]              ;SHIFT Q AND RETURN
                                                    ; 4173  =110                            ;1 IN A POSITIVE STEP
                                                    ; 4174          AD/B-A-.25, ADD .25,    ;SUBTRACT
                                                    ; 4175          MUL STEP,               ;SHIFT AND COUNT
U 0126, 0142,1116,0604,4174,4046,2630,6000,0060,0000  ; 4176          J/MUL-                  ;NEGATIVE NOW
                                                    ; 4177  =111                            ;DONE
                                                    ; 4178          AD/B-A-.25, ADD .25,    ;SUBTRACT
                                                    ; 4179          MUL FINAL,              ;SHIFT
U 0127, 0004,1114,0604,4174,4004,1700,4000,0000,0000  ; 4180          RETURN [4]              ; AND RETURN
                                                    ; 4181
                                                    ; 4182  ;HERE FOR NEGATIVE STEPS
                                                    ; 4183  =010                            ;0 IN NEGATIVE STEP
                                                    ; 4184  MUL-:   AD/A+B,                 ;ADD
                                                    ; 4185          MUL STEP,               ;SHIFT AND COUNT
U 0142, 0122,0116,0604,4174,4046,2630,2000,0060,0000  ; 4186          J/MUL+                  ;POSITIVE NOW
                                                    ; 4187  =011                            ;DONE
                                                    ; 4188          AD/A+B,                 ;ADD
                                                    ; 4189          MUL FINAL,              ;SHIFT
U 0143, 0004,0114,0604,4174,4004,1700,0000,0000,0000  ; 4190          RETURN [4]                      ;FIX Q AND RETURN
                                                    ; 4191  =110                            ;1 IN NEGATIVE STEP
                                                    ; 4192          AD/B,                   ;DON'T ADD
                                                    ; 4193          MUL STEP,               ;SHIFT AND COUNT
U 0146, 0142,3336,0604,4174,4046,2630,2000,0060,0000  ; 4194          J/MUL-                  ;STILL NEGATIVE
                                                    ; 4195  =111                            ;DONE
                                                    ; 4196          AD/B,                   ;DON'T ADD
                                                    ; 4197          MUL FINAL,              ;SHIFT
U 0147, 0004,3334,0604,4174,4004,1700,0000,0000,0000  ; 4198          RETURN [4]                      ;FIX Q AND RETURN
                                                    ; 4199
```

```
D 0230, 1005,1600,1100
D 0231, 0005,1600,3000
D 0232, 0016,1600,1700
D 0233, 0006,1600,1700

D 0234, 1005,1601,1100
D 0235, 0005,1601,3000
D 0236, 0016,1601,1700
D 0237, 0006,1601,1700



U 1600, 3004,3441,0305,0174,4007,0700,0000,0000,0000

U 3004, 1214,3772,0000,0275,5007,0520,0000,0000,0000

U 1214, 0161,4221,0003,4174,4007,0700,0000,0000,0000

U 1215, 0161,2441,0703,4174,4007,0700,4000,0000,0000


U 1601, 3005,3441,0305,4174,4007,0700,0000,0000,0000
U 3005, 3006,3771,0003,0276,6007,0700,0000,0000,0000
U 3006, 3007,3772,0000,1275,5007,0701,0000,0000,1441

U 3007, 0160,3333,0003,4174,4007,0621,0000,0000,0000

U 0160, 1216,7443,0300,4174,4007,0621,0000,0000,0000

U 0161, 0164,3333,0005,4174,4007,0621,0000,0000,0000

U 0164, 0370,4443,0000,4174,4007,0700,2010,0071,0042
U 0165, 0555,4443,0000,4174,4467,0700,0000,0051,1000

U 0166, 1177,3441,0304,4174,4007,0700,0000,0000,0000




U 1216, 3010,3441,0306,4174,4007,0700,0000,0000,0000


U 1217, 0164,3333,0005,4174,4007,0621,0000,0000,0000
```

```
; 4200  .TOC    "ARITHMETIC -- DIV, IDIV"
; 4201
; 4202          .DCODE
; 4203  230:    R-PF,    DAC,     J/IDIV
; 4204          I-PF,    DAC,     J/IDIV
; 4205          RW,      M,       J/IDIV
; 4206          RW,      DBL B,   J/IDIV
; 4207
; 4208  234:    R-PF,    DAC,     J/DIV
; 4209          I-PF,    DAC,     J/DIV
; 4210          RW,      M,       J/DIV
; 4211          RW,      DBL B,   J/DIV
; 4212          .UCODE
; 4213
; 4214  1600:
; 4215  IDIV:   [BR]_[AR], AC            ;COPY MEMORY OPERAND
; 4216          Q_AC,                    ;LOAD Q
; 4217          SKIP DPO                 ;SEE IF MINUS
; 4218  =0      [AR]_0,                  ;EXTEND + SIGN
; 4219          J/DIV1                   ;NOW SAME AS DIV
; 4220          [AR]_-1,                 ;EXTEND - SIGN
; 4221          J/DIV1                   ;SAME AS DIV
; 4222
; 4223  1601:
; 4224  DIV:    [BR]_[AR]                ;COPY MEM OPERAND
; 4225          [AR]_AC                  ;GET AC
; 4226          Q_AC[1]                  ;AND AC+1
; 4227          READ [AR],               ;TEST FOR NO DIVIDE
; 4228          SKIP AD.EQ.0
; 4229  =000    .NOT.[AR],               ;SEE IF ALL SIGN BITS IN AR
; 4230          SKIP AD.EQ.0,            ; ..
; 4231          J/DIVA                   ;CONTINUE BELOW
; 4232  =001
; 4233  DIV1:   READ [BR],               ;SEE IF DIVIDE BY
; 4234          SKIP AD.EQ.0             ; ZERO
; 4235  =100
; 4236  DIV2:   SC_34.,                  ;NOT ZERO--LOAD STEP COUNT
; 4237          CALL [DIVSUB]            ;DIVIDE
; 4238  =101    NO DIVIDE                ;DIVIDE BY ZERO
; 4239  =110    [ARX]_[AR],              ;COPY REMAINDER
; 4240          J/IMUL1                  ;STORE ANSWER
; 4241  =
; 4242
; 4243
; 4244  =0
; 4245  DIVA:   [BRX]_[AR],              ;HIGH WORD IS NOT SIGNS
; 4246          J/DIVB                   ;GO TEST FOR NO DIVIDE
; 4247          READ [BR],               ;ALL SIGN BITS
; 4248          SKIP AD.EQ.0,            ;SEE IF ZERO DIVIDE
; 4249          J/DIV2                   ;BACK TO MAIN FLOW
; 4250
```

```
U 3010, 3011,3221,0004,4174,4007,0700,0000,0000,0000
U 3011, 0330,3333,0003,4174,4007,0520,0000,0000,0000
U 0330, 1220,3333,0005,4174,4007,0520,0000,0000,0000
U 0331, 3072,4551,0404,4374,0007,0700,0010,0037,7777
U 0333, 1220,3333,0005,4174,4007,0520,0000,0000,0000
U 1220, 1222,2113,0305,4174,4007,0521,4000,0000,0000
U 1221, 1222,0113,0305,4174,4007,0521,0000,0000,0000
U 1222, 0555,4443,0000,4174,4467,0700,0000,0051,1000
U 1223, 0161,3441,0603,4174,4007,0700,0000,0000,0000
```

```
; 4251  DIVB:   [ARX]_Q                ;MAKE ABS VALUES
; 4252          READ [AR],             ;SEE IF +
; 4253          SKIP DPO
; 4254  =OO     READ [BR],             ;SEE IF +
; 4255          SKIP DPO,
; 4256          J/DIVC                 ;CONTINUE BELOW
; 4257          CLEAR [ARX]O,          ;FLUSH DUPLICATE SIGN
; 4258          CALL [DBLNG1]          ;NEGATE AR!ARX
; 4259  =11     READ [BR],             ;SEE IF TOO BIG
; 4260          SKIP DPO,
; 4261          J/DIVC
; 4262  =
; 4263  =O
; 4264  DIVC:   [AR]-[BR],             ;COMPUTE DIFFERENCE
; 4265          SKIP DPO,              ;SEE IF IT GOES
; 4266          3T,                    ;ALLOW TIME
; 4267          J/NODIV                ;TEST
; 4268          [AR]+[BR],
; 4269          SKIP DPO,              ;SAME TEST FOR -VE BR
; 4270          3T,
; 4271          J/NODIV
; 4272  =O
; 4273  NODIV:  NO DIVIDE              ;TOO BIG
; 4274          [AR]_[BRX],            ;FITS
; 4275          J/DIV1                 ;GO BACK AND DIVIDE
; 4276
```

```
D 0117, 0205,1627,1100

U 1627, 3012,4112,0400,4174,4007,0700,0000,0000,0000
U 3012, 1224,3447,0305,4174,4007,0421,0000,0000,0000
U 1224, 1230,3446,0505,4174,4007,0700,0000,0000,0000
U 1225, 1226,3446,0505,4174,4007,0520,0000,0000,0000
U 1226, 1230,4003,0000,4174,4007,0621,0000,0000,0000
U 1227, 3013,4751,1217,4374,4007,0700,0000,0000,0005
U 3013, 3014,3662,0000,4374,0007,0700,0000,0060,0000
U 3014, 3015,2222,0000,4174,4007,0700,4000,0000,0000
U 3015, 3017,2446,0505,4174,4047,0700,0040,0000,0000
U 1230, 3016,3446,0505,4174,4047,0700,0000,0000,0000
U 1231, 0555,4443,0000,4174,4467,0700,0000,0051,1000
U 3016, 3017,4751,1217,4374,4007,0700,0000,0000,0004
U 3017, 3020,3221,0006,0174,4007,0700,0000,0000,0000
U 3020, 0054,3777,0003,0274,4007,0520,0000,0000,0000
U 0054, 1232,4552,0000,1275,5007,0701,0000,0000,1441
U 0055, 3061,6551,1717,4374,4007,0700,0010,0000,0007
U 0075, 0054,3447,0303,4174,4007,0700,0000,0000,0000
U 1232, 3043,3446,0303,4174,4007,0700,0010,0000,0000
U 1233, 1234,2113,0305,4174,4007,0521,4000,0000,0000
U 1234, 1236,2113,0305,4174,4007,0620,4000,0000,0000
U 1235, 3021,3221,0004,4174,4007,0700,0000,0000,0000
U 1236, 0033,3333,0017,4174,4003,5701,0000,0000,0000
U 1237, 1240,1003,0600,4174,4007,0521,4000,0000,0000
U 1240, 0033,3333,0017,4174,4003,5701,0000,0000,0000
U 1241, 3021,3221,0004,4174,4007,0700,0000,0000,0000
```

```
; 4277  .TOC    "ARITHMETIC -- DDIV"
; 4278
; 4279          .DCODE
; 4280  117:    DBL R,  DAC,    J/DDIV
; 4281          .UCODE
; 4282
; 4283  .IF/FULL
; 4284  1627:
; 4285  DDIV:   Q_[ARX].AND.[MAG]          ;COPY LOW WORD
; 4286          [BR]_[AR]*.5,              ;COPY MEMORY OPERAND
; 4287          SKIP AD.LE.0               ;SEE IF POSITIVE
; 4288  =0      [BR]_[BR]*.5 LONG,         ;POSITIVE
; 4289          J/DDIV1                    ;CONTINUE BELOW
; 4290          [BR]_[BR]*.5 LONG,         ;NEGATIVE OR ZERO
; 4291          SKIP DPO                   ;SEE WHICH?
; 4292  =0      [MAG].AND.Q,               ;SEE IF ALL ZERO
; 4293          SKIP AD.EQ.0, J/DDIV1      ;CONTINUE BELOW
; 4294          [T1]_0 XWD [5]             ;NEGATE MEM OP
; 4295          Q_Q.OR.#, #/600000,        ;SIGN EXTEND THE LOW
; 4296          HOLD RIGHT                 ; WORD
; 4297          Q_-Q                       ;MAKE Q POSITIVE
; 4298          [BR]_(-[BR]-.25)*.5 LONG,  ;NEGATE HIGH WORD
; 4299          ASHC, MULTI PREC/1,        ;USE CARRY FROM LOW WORD
; 4300          J/DDIV3                    ;CONTINUE BELOW
; 4301  =0
; 4302  DDIV1:  [BR]_[BR]*.5 LONG,         ;SHIFT OVER 1 PLACE
; 4303          ASHC, J/DDIV2              ;CONTINUE BELOW
; 4304          NO DIVIDE                  ;DIVIDE BY ZERO
; 4305  DDIV2:  [T1]_0 XWD [4]             ;MEM OPERAND IS POSITIVE
; 4306  DDIV3:  [BRX]_Q, AC                ;COPY Q
; 4307
; 4308          [AR]_AC*.5, 2T, SKIP DPO ;GET AC--SEE IF NEGATIVE
; 4309  =0*1*0
; 4310  DDIV3A: Q_AC[1].AND.[MAG],         ;POSITIVE (OR ZERO)
; 4311          J/DDIV4                    ;CONTINUE BELOW
; 4312          [T1]_[T1].XOR.#,           ;NEGATIVE
; 4313          #/7, CALL [QDNEG]          ;UPDATE SAVED FLAGS
; 4314  =1*1*1  [AR]_[AR]*.5,              ;SHIFT AR OVER
; 4315          J/DDIV3A                   ;GO BACK AND LOAD Q
; 4316  =
; 4317  =0
; 4318  DDIV4:  [AR]_[AR]*.5 LONG,         ;SHIFT AR OVER
; 4319          CALL [DDIVS]               ;SHIFT 1 MORE PLACE
; 4320          [AR]-[BR], 3T, SKIP DPO ;TEST MAGNITUDE
; 4321  =0      [AR]-[BR], 2T,
; 4322          SKIP AD.EQ.0, J/DDIV5
; 4323          [ARX]_Q, J/DDIV5A          ;ANSWER FITS
; 4324
; 4325  =0
; 4326  DDIV5:  READ [T1], 3T, DISP/DP, J/NODDIV
; 4327          Q-[BRX], 3T, SKIP DPO
; 4328  =0      READ [T1], 3T, DISP/DP, J/NODDIV
; 4329          [ARX]_Q                    ;COPY LOW WORD
; 4330
```

```
U 3021, 0354,4552,0000,1275,5007,0701,0000,0000,1442
U 0354, 1264,3446,1200,4174,4007,0700,2010,0071,0042
U 0356, 3022,3224,0016,4174,4007,0700,0000,0000,0000
U 3022, 3023,0002,1600,4174,4007,0700,0000,0000,0000
U 3023, 1242,4003,0000,1174,4007,0700,0400,0000,1440
U 1242, 3043,3442,0400,4174,4007,0700,0010,0000,0000
U 1243, 3024,3221,0004,4174,4007,0700,0000,0000,0000
U 3024, 0551,4552,0000,1275,5007,0701,0000,0000,1443



U 0551, 1264,3446,0316,4174,4007,0520,2010,0071,0042
U 0553, 3025,3224,0016,4174,4007,0700,0000,0000,0000
U 3025, 0056,3333,0017,4174,4003,5701,0000,0000,0000

U 0056, 3030,0001,1616,4174,4007,0700,0000,0000,0000
U 0057, 3026,2225,0016,4174,4007,0700,4000,0000,0000



U 3026, 3027,1772,0000,0274,4007,0701,0040,0000,0000
U 3027, 1244,3223,0000,0174,4007,0621,0400,0000,0000
U 1244, 3031,3440,1616,1174,4007,0700,0400,0000,1441
U 1245, 3033,4223,0000,1174,4007,0700,0400,0000,1441

U 3030, 3033,4113,1600,1174,4007,0700,0400,0000,1441

U 3031, 3032,3551,1616,4374,0007,0700,0000,0040,0000
U 3032, 3033,3440,1616,1174,4007,0700,0400,0000,1441

U 3033, 1246,3333,0003,4174,4007,0520,0000,0000,0000

U 1246, 3037,3442,0400,4174,4007,0700,0000,0000,0000
U 1247, 3034,0112,0406,4174,4007,0700,0000,0000,0000

U 3034, 3035,0111,0503,4174,4007,0700,0040,0000,0000
U 3035, 3036,0002,0600,4174,4007,0700,0000,0000,0000

U 3036, 3037,0111,0503,4174,4007,0700,0040,0000,0000
U 3037, 0355,3333,0017,4174,4003,5701,0000,0000,0000



U 0355, 3041,3444,0303,4174,4047,0700,0000,0000,0000
U 0357, 3040,2222,0000,4174,4007,0700,4000,0000,0000

U 3040, 3041,2444,0303,4174,4047,0700,0040,0000,0000
```

```
; 4331  ;HERE WITH EVERYTHING SETUP AND READY TO GO
; 4332  DDIV5A: Q_AC[2].AND.[MAG]
; 4333  =0*     Q_Q*.5, SC_34., CALL [DBLDIV]
; 4334          [TO]_Q*2 LONG
; 4335          Q_Q+[TO]
; 4336          AC[0]_Q.AND.[MAG]          ;STORE ANSWER
; 4337  =0      Q_[ARX], CALL [DDIVS]      ;SHIFT OUT EXTRA ZERO BIT
; 4338          [ARX]_Q                    ; ..
; 4339          Q_AC[3].AND.[MAG]
; 4340  =0*     [TO]_[AR]*.5 LONG,         ;SHIFT Q, PUT AR ON DP
; 4341          SC_34.,                    ;LOAD SHIFT COUNT
; 4342          SKIP DPO,                  ;LOOK AT AR SIGN
; 4343          CALL [DBLDIV]              ;GO DIVIDE
; 4344          [TO]_Q*2 LONG
; 4345          READ [T1], 3T, DISP/DP     ;WHAT SIGN IS QUO
; 4346  =1110   [TO]_[TO]+Q,               ;POSITIVE QUO
; 4347          J/DDIV5B                   ;CONTINUE BELOW
; 4348          [TO]_-Q*2                  ;NEGATIVE QUO
; 4349          AD/-D-.25, DBUS/RAM, 3T,
; 4350          RAMADR/AC#, DEST/Q_AD,
; 4351          MULTI PREC/1
; 4352          AC_Q, SKIP AD.EQ.0
; 4353  =0      AC[1]_[TO], J/DDIV5C
; 4354          AC[1]_0, J/DDIV6
; 4355
; 4356  DDIV5B: AC[1]_[TO].AND.[MAG], J/DDIV6   ;STORE LOW WORD IN + CASE
; 4357
; 4358  DDIV5C: [TO]_[TO].OR.#, #/400000, HOLD RIGHT
; 4359          AC[1]_[TO]
; 4360
; 4361  DDIV6:  READ [AR], SKIP DPO      ;LOOK AT AR SIGN
; 4362  =0
; 4363  DDIV7:  Q_[ARX], J/DDIV8
; 4364          Q_[ARX]+[BRX]
; 4365          [AR]_[AR]+[BR],
; 4366          MULTI PREC/1
; 4367          Q_Q+[BRX]
; 4368          [AR]_[AR]+[BR],
; 4369          MULTI PREC/1
; 4370  DDIV8:  READ [T1], 3T, DISP/DP
; 4371  =1101
; 4372  DDIV8A: [AR]_[AR]*2 LONG, ASHC, ;POSITIVE REMAINDER
; 4373          J/DDIV9                    ;CONTINUE BELOW
; 4374          Q_-Q                       ;NEGATE REMAINDER IN AR!Q
; 4375          [AR]_(-[AR]-.25)*2 LONG,
; 4376          MULTI PREC/1, ASHC
; 4377
```

```
U 3041, 1250,0113,0303,1174,4007,0521,0400,0000,1442

U 1250, 0100,4003,0000,1174,4156,4700,0400,0000,1443
U 1251, 3042,4002,0000,1174,4007,0700,0000,0000,1443

U 3042, 0100,7003,0000,1174,4156,4700,0400,0000,1443



U 0033, 3061,4443,0000,4174,4007,0700,0010,0000,0000
U 0037, 0555,4443,0000,4174,4467,0700,0000,0051,1000

U 3043, 0001,3446,0303,4174,4044,1700,0000,0000,0000
```



```
; 4378  DDIV9:  AC[2]_[AR]+[AR], 3T,
; 4379          SKIP DPO
; 4380  =0      AC[3]_Q.AND.[MAG],
; 4381          NEXT INST
; 4382          Q_Q.AND.[MAG], AC[3]
; 4383          AC[3]_[MAG].EQV.Q,
; 4384          NEXT INST
; 4385
; 4386
; 4387  ;HERE IF WE WANT TO SET NO DIVIDE
; 4388  =11011
; 4389  NODDIV: CALL [QDNEG]              ;FIXUP AC TO AC+3
; 4390          NO DIVIDE                 ;ABORT DIVIDE
; 4391
; 4392  DDIVS:  [AR]_[AR]*.5 LONG, ASHC, RETURN [1]
;;4393  .IFNOT/FULL
;;4394  1627:
;;4395  DDIV:   UUO
; 4396  .ENDIF/FULL
; 4397
```

```
U 0370, 3044,4662,0000,4374,0007,0700,0010,0037,7777
U 0374, 0002,4443,0000,4174,4004,1700,0000,0000,0000
U 0375, 0002,2222,0000,4174,4004,1700,4000,0000,0000
U 0376, 0377,2222,0000,4174,4007,0700,4000,0000,0000
U 0377, 0002,2441,0303,4174,4004,1700,4000,0000,0000
```

```
; 4398  .TOC    "ARITHMETIC -- DIVIDE SUBROUTINE"
; 4399
; 4400  ;HERE IS THE SUBROUTINE TO DO DIVIDE
; 4401  ;ENTER WITH:
; 4402  ;       AR!Q ← D'END
; 4403  ;       BR = D'SOR
; 4404  ;RETURN 2 WITH:
; 4405  ;       AR = REMAINDER
; 4406  ;       Q = QUOTIENT
; 4407  ;CALLER MUST CHECK FOR ZERO DIVIDE PRIOR TO CALL
; 4408  ;
; 4409  =1000
; 4410  DIVSUB: Q_Q.AND.#,              ;CLEAR SIGN BIT IN
; 4411          #/377777,               ;MASK
; 4412          HOLD RIGHT,             ;JUST CLEAR BIT 0
; 4413          CALL [DIVSGN]           ;DO REAL DIVIDE
; 4414  =1100   RETURN [2]              ;ALL POSITIVE
; 4415  =1101   Q_-Q, RETURN [2]        ;-QUO +REM
; 4416  =1110   Q_-Q                    ;ALL NEGATIVE
; 4417  =1111   [AR]_-[AR], RETURN [2]  ;NEGATIVE REMAINDER
; 4418
```

```
U 3044,  1252,3333,0003,4174,4007,0520,0000,0000,0000
U 1252,  3046,4221,0004,4174,4007,0700,0000,0000,0000
U 1253,  1254,2222,0000,4174,4007,0621,4000,0000,0000
U 1254,  3045,7441,0303,4174,4007,0700,0000,0000,0000
U 1255,  3045,2441,0303,4174,4007,0700,4000,0000,0000

U 3045,  3046,3771,0004,4374,4007,0700,0000,0010,0000
U 3046,  1256,3333,0005,4174,4007,0520,0000,0000,0000


U 1256,  3050,3446,0303,4174,4007,0700,0000,0000,0000
U 1257,  3047,2441,0505,4174,4007,0700,4000,0000,0000

U 3047,  1256,3551,0404,4374,4007,0700,0000,0004,0000
U 3050,  3051,3447,0303,4174,4007,0700,0000,0000,0000
U 3051,  3052,3447,0505,4174,4007,0700,0000,0000,0000
U 3052,  3053,3447,0505,4174,4007,0700,0000,0000,0000
U 3053,  1260,1114,0503,4174,4067,0630,6000,0060,0000


U 1260,  1260,0114,0503,4174,4067,0630,2100,0060,0000
U 1261,  3054,3444,1717,4174,4067,0700,0100,0000,0000
U 3054,  1262,3447,0303,4174,4007,0520,0000,0000,0000

U 1262,  3055,3444,0303,4174,4007,0700,0000,0000,0000
U 1263,  1262,0111,0503,4174,4007,0700,0000,0000,0000
U 3055,  3056,3444,0303,4174,4007,0700,0000,0000,0000
U 3056,  3057,4002,1200,4174,4007,0700,0000,0000,0000


U 3057,  0014,3333,0004,4174,4000,1701,0000,0000,0000
```

```
; 4419  ;HERE IS THE INNER DIVIDE SUBROUTINE
; 4420  ;SAME SETUP AS DIVSUB
; 4421  ;RETURNS WITH AR AND Q POSITIVE AND
; 4422  ;       14 IF ALL POSITIVE
; 4423  ;       15 IF -QUO
; 4424  ;       16 IF ALL NEGATIVE
; 4425  ;       17 IF NEGATIVE REMAINDER
; 4426
; 4427  BASIC DIV STEP  "DEST/Q_Q*2, DIV, A/BR, B/AR, STEP SC"
; 4428  DIV STEP        "BASIC DIV STEP, AD/A+B, DIVIDE/1"
; 4429  FIRST DIV STEP  "BASIC DIV STEP, AD/B-A-.25, ADD .25"
; 4430
; 4431  DIVSGN: READ [AR], SKIP DPO
; 4432  =0      [ARX]_0, J/DVSUB2        ;REMAINDER IS POSITIVE
; 4433         Q_-Q, SKIP AD.EQ.0       ;COMPLEMENT LOW WORD
; 4434  =0      [AR]_.NOT.[AR], J/DVSUB1 ;COMPLEMENT HI WORD
; 4435         [AR]_-[AR]               ;TWO'S COMPLEMENT HI WORD SINCE
; 4436                                  ; LOW WORD WAS ZERO
; 4437  DVSUB1: [ARX]_#, #/100000        ;REMAINDER IS NEGATIVE
; 4438  DVSUB2: READ [BR], SKIP DPO      ;IS THE DIVISOR NEGATIVE
; 4439  =0
; 4440  DVSUB3: [AR]_[AR]*.5 LONG,       ;START TO PUT IN 9-CHIPS
; 4441         J/DIVSET                 ;JOIN MAIN STREAM
; 4442         [BR]_-[BR]               ;COMPLEMENT DIVISOR
; 4443         [ARX]_[ARX].OR.#,        ;ADJUST SIGN OF QUOTIENT
; 4444         #/400000, J/DVSUB3       ;USE 9 CHIPS
; 4445  DIVSET: [AR]_[AR]*.5
; 4446         [BR]_[BR]*.5
; 4447         [BR]_[BR]*.5
; 4448         FIRST DIV STEP
; 4449  ;HERE IS THE MAIN DIVIDE LOOP
; 4450  =0
; 4451  DIVIDE: DIV STEP, J/DIVIDE
; 4452         [T1]_[T1]*2 LONG, DIVIDE/1, DIV
; 4453         [AR]_[AR]*.5, SKIP DPO
; 4454  =0
; 4455  FIX++:  [AR]_[AR]*2 LONG, J/FIX1++
; 4456         [AR]_[AR]+[BR], J/FIX++
; 4457  FIX1++: [AR]_[AR]*2 LONG
; 4458         Q_[MASK].AND.Q
; 4459         READ [ARX], 3T,          ;RETURN TO 1 OF 4 PLACES
; 4460         DISP/1,                  ;BASED ON SIGN OF RESULT
; 4461         J/14                     ;RETURN
; 4462
```

```
U 1264, 3060,1114,0604,4174,4057,0700,4000,0000,0000
U 1265, 3060,0114,0604,4174,4057,0700,0000,0000,0000
U 3060, 1266,0115,0503,4174,4007,0630,2040,0060,0000
U 1266, 3060,0114,0604,4174,4067,0700,0100,0000,0000
U 1267, 0002,4444,0002,4174,4064,1700,0100,0000,0000
```

```
; 4463  .TOC    "ARITHMETIC -- DOUBLE DIVIDE SUBROUTINE"
; 4464  .IF/FULL
; 4465  ;CALL WITH:
; 4466  ;       AR!ARX!Q = 3 WORD DV'END
; 4467  ;       BR!BRX   = 2 WORD DV'SOR
; 4468  ;RETURN 2 WITH:
; 4469  ;       AR!ARX   = 2 WORD REMAINDER
; 4470  ;                       CORRECT IF POSITIVE (Q IS ODD)
; 4471  ;                       WRONG (BY BR!BRX) IF NEGATIVE (Q IS EVEN)
; 4472  ;       Q        = 1 WORD QUOTIENT
; 4473  ;CALLER MUST CHECK FOR ZERO DIVIDE PRIOR TO CALL
; 4474  ;
; 4475  ;NOTE: THIS SUBROUTINE ONLY WORKS FOR POSITIVE NUMBERS
; 4476  ;
; 4477  =O
; 4478  ;HERE FOR NORMAL STARTUP
; 4479  DBLDIV: [ARX]_([ARX]-[BRX])*2 LONG,  ;SUBTRACT LOW WORD
; 4480          LSHC, J/DIVHI               ;GO ENTER LOOP
; 4481  ;SKIP ENTRY POINT IF FINAL STEP IN PREVIOUS ENTRY WAS IN ERROR
; 4482          [ARX]_([ARX]+[BRX])*2 LONG, ;CORRECTION STEP
; 4483          LSHC, J/DIVHI               ;GO ENTER LOOP
; 4484
; 4485  ;HERE IS DOUBLE DIVIDE LOOP
; 4486  DIVHI:  AD/A+B,                      ;ADD (HARDWARE MAY OVERRIDE)
; 4487          A/BR, B/AR,                 ;OPERANDS ARE AR AND BR
; 4488          DEST/AD*2,                  ;SHIFT LEFT
; 4489          SHSTYLE/NORM,               ;SET SHIFT PATHS (SEE DPE1)
; 4490          MULTI PREC/1,               ;INJECT SAVED BITS
; 4491          STEP SC                     ;COUNT DOWN LOOP
; 4492  =O      AD/A+B,                     ;ADD (HARDWARE MAY OVERRIDE)
; 4493          A/BRX, B/ARX,               ;LOW WORDS
; 4494          DEST/Q_Q*2,                 ;SHIFT WHOLE MESS LEFT
; 4495          SHSTYLE/DIV,                ;SET SHIFT PATHS (SEE DPE1)
; 4496          DIVIDE/1,                   ;SAVE BITS
; 4497          J/DIVHI                     ;KEEP LOOPING
; 4498  ;HERE WHEN ALL DONE
; 4499          DEST/Q_Q*2, DIV,            ;SHIFT IN LAST Q BIT
; 4500          DIVIDE/1,                   ;GENERATE BIT
; 4501          B/HR, RETURN [2]            ;ZERO HR AND RETURN
; 4502
```

```
U 3061, 3062,1772,0000,1274,4007,0701,4000,0000,1443
U 3062, 1270,4003,0000,1174,4007,0621,0400,0000,1443
U 1270, 3065,7772,0000,1274,4007,0701,0000,0000,1442
U 1271, 3063,1772,0000,1274,4007,0701,4000,0000,1442
U 3063, 1272,4003,0000,1174,4007,0621,0400,0000,1442
U 1272, 3066,7772,0000,1274,4007,0701,0000,0000,1441
U 1273, 3064,1772,0000,1274,4007,0701,4000,0000,1441
U 3064, 1274,4003,0000,1174,4007,0621,0400,0000,1441
U 1274, 3067,7771,0003,0274,4007,0700,0000,0000,0000
U 1275, 3067,1771,0003,0274,4007,0701,4000,0000,0000
U 3065, 1272,4003,0000,1174,4007,0700,0400,0000,1442
U 3066, 1274,4003,0000,1174,4007,0700,0400,0000,1441
U 3067, 0024,3440,0303,0174,4004,1700,0400,0000,0000
U 3070, 3071,4551,0404,4374,0007,0700,0000,0037,7777
U 3071, 1276,2441,0404,4174,4007,0621,4000,0000,0000
U 1276, 2222,7441,0303,4174,4467,0700,0000,0001,0001
U 1277, 2222,2441,0303,4174,4467,0701,4000,0001,0001
U 3072, 1300,2441,0404,4174,4007,0621,4000,0000,0000
U 1300, 2222,7441,0303,4174,4007,0700,0000,0000,0000
U 1301, 2222,2441,0303,4174,4007,0700,4000,0000,0000
```

```
; 4503  .TOC    "ARITHMETIC -- SUBROUTINES FOR ARITHMETIC"
; 4504
; 4505  ;QUAD WORD NEGATE
; 4506  ;ARGUMENT IN AC!AC1!AC2!AC3
; 4507  ;LEAVES COPY OF AC!AC1 IN AR!Q
; 4508  ;RETURNS TO CALL!24
; 4509  QDNEG:  Q_-AC[3]
; 4510          AC[3]_Q.AND.[MAG],      ;PUT BACK LOW WORD
; 4511          SKIP AD.EQ.0            ;SEE IF ANY CARRY
; 4512  =0
; 4513  COM2A:  Q_.NOT.AC[2], J/COM2     ;CARRY--DO 1'S COMPLEMENT
; 4514          Q_-AC[2]                ;NEXT WORD
; 4515          AC[2]_Q.AND.[MAG],      ;PUT BACK WORD
; 4516          SKIP AD.EQ.0
; 4517  =0
; 4518  COM1A:  Q_.NOT.AC[1], J/COM1
; 4519          Q_-AC[1]
; 4520          AC[1]_Q.AND.[MAG],
; 4521          SKIP AD.EQ.0
; 4522  =0
; 4523  COMOA:  [AR]_.NOT.AC, J/COMO
; 4524          [AR]_-AC, 3T, J/COMO
; 4525
; 4526  COM2:   AC[2]_Q.AND.[MAG], J/COM1A
; 4527  COM1:   AC[1]_Q.AND.[MAG], J/COMOA
; 4528  COMO:   AC_[AR], RETURN [24]
; 4529  .ENDIF/FULL
; 4530
; 4531  ;DOUBLE WORD NEGATE
; 4532  ;ARGUMENT IN AR AND ARX
; 4533  ;RETURNS TO CALL!2
; 4534
; 4535  DBLNEG: CLEAR ARXO              ;FLUSH DUPLICATE SIGN
; 4536  DBLNGA: [ARX]_-[ARX],           ;FLIP LOW WORD
; 4537          SKIP AD.EQ.0            ;SEE IF CARRY
; 4538  =0      [AR]_.NOT.[AR],         ;NO CARRY-- 1 COMP
; 4539          AD FLAGS, J/CLARXO      ;CLEAR LOW SIGN
; 4540          [AR]_-[AR],             ;CARRY
; 4541          AD FLAGS, 3T, J/CLARXO
; 4542
; 4543  ;SAME THING BUT DOES NOT SET PC FLAGS
; 4544  DBLNG1: [ARX]_-[ARX], SKIP AD.EQ.0
; 4545  =0      [AR]_.NOT.[AR], J/CLARXO
; 4546          [AR]_-[AR], J/CLARXO
; 4547
```

```
; 4548          .NOBIN
; 4549  .TOC    "BYTE GROUP -- IBP, ILDB, LDB, IDPB, DPB"
; 4550
; 4551
; 4552  ;ALL FIVE INSTRUCTIONS OF THIS GROUP ARE CALLED WITH THE BYTE POINTER
; 4553  ;IN THE AR.  ALL INSTRUCTIONS SHARE COMMON SUBROUTINES.
; 4554
; 4555  ;IBP OR ADJBP
; 4556  ;IBP IF AC#0, ADJBP OTHERWISE
; 4557  ; HERE WITH THE BASE POINTER IN AR
; 4558
; 4559  ;HERE IS A MACRO TO DO IBP. WHAT HAPPENS IS:
; 4560  ;       THE AR IS PUT ON THE DP.
; 4561  ;       THE BR IS LOADED FROM THE DP WITH BITS 0-5 FROM SCAD
; 4562  ;       THE SCAD COMPUTES P-S
; 4563  ;       IBPS IS CALLED WITH A 4-WAY DISPATCH ON SCAD0 AND FIRST-PART-DONE
; 4564  ;THE MACRO IS WRITTEN WITH SEVERAL SUB-MACROS BECAUSE OF RESTRICTIONS
; 4565  ; IN THE MICRO ASSEMBLER
; 4566
; 4567  IBP DP          "AD/D, DEST/A, A/AR, B/BR, DBUS/DBM, DBM/DP, BYTE/BYTE1"
; 4568  IBP SCAD        "SCAD/A-B, SCADA/BYTE1, SCADB/SIZE"
; 4569  IBP SPEC        "SCAD DISP, SKIP FPD"
; 4570  CALL IBP        "IBP DP, IBP SCAD, IBP SPEC, CALL [IBPS], DT/3T"
; 4571
; 4572  SET P TO 36-S   "AD/D,DEST/A,A/BR,B/AR,DBUS/DBM,DBM/DP,SCAD/A-B,SCADB/SIZE,BYTE/BYTE1,SCADA/PTR44"
; 4573
; 4574  ;THE FOLLOWING MACRO IS USED FOR COUNTING SHIFTS IN THE BYTE ROUTINES. IT
; 4575  ; USES THE FE AND COUNTS BY 8. NOTE: BYTE STEP IS A 2S WEIGHT SKIP NOT 1S.
; 4576  BYTE STEP       "SCAD/A+B,SCADA/S#,S#/1770,SCADB/FE,LOAD FE, 3T,SCAD DISP"
; 4577
```

```
D 0133, 0015,1610,1100
D 0134, 0000,1620,1500

D 0135, 0000,1624,1100
D 0136, 0000,1630,1500
D 0137, 0000,1634,1100

U 1610, 0240,4443,0000,4174,4007,0360,0000,0000,0000

U 0240, 3130,3333,0003,7174,4007,0700,0400,0000,0223
U 0241, 0350,3770,0305,4334,4016,7351,0010,0033,6000
U 0245, 0110,3443,0100,4174,4156,4700,0200,0014,0012

U 1620, 0350,3770,0305,4334,4016,7351,0010,0033,6000

U 1624, 3100,3333,0003,4174,4217,0701,1010,0073,0500
U 1625, 0604,3443,0100,4174,4007,0700,0200,0014,0012

U 0604, 0340,3333,0003,4174,4006,5701,1010,0051,0770

U 0606, 0555,3440,0303,0174,4467,0700,0400,0005,0000

U 1630, 0350,3770,0305,4334,4016,7351,0010,0033,6000

U 1634, 3073,3775,0004,0274,4007,0701,0000,0000,0000

U 3073, 3074,3443,0400,4174,4007,0701,1000,0077,0000
U 3074, 0264,3771,0004,0276,6007,0700,0000,0000,0000

U 0264, 3100,3333,0003,4174,4217,0700,0010,0000,0500

U 0265, 0360,3333,0003,4174,4006,5701,0010,0000,0000
U 0267, 1400,4443,0000,4174,4467,0700,0000,0005,0000
```

```
; 4578          .BIN
; 4579
; 4580          .DCODE
; 4581  133:    R,      AC,     J/IBP           ;OR ADJBP
; 4582  134:    R,W TEST,       J/ILDB          ;CAN'T USE RPW BECAUSE OF FP
; 4583          R,              J/LDB
; 4584          R,W TEST,       J/IDPB
; 4585          R,              J/DPB
; 4586          .UCODE
; 4587  1610:
; 4588  IBP:    SKIP IF ACO                     ;SEE IF ADJBP
; 4589  =000    WORK[ADJPTR]_[AR],              ;SAVE POINTER
; 4590          J/ADJBP                         ;GO ADJUST BYTE POINTER
; 4591  =001    CALL IBP                        ;BUMP BYTE POINTER
; 4592  =101    DONE                            ;POINTER STORED
; 4593  =
; 4594
; 4595  1620:
; 4596  ILDB:   CALL IBP                        ;BUMP BYTE POINTER
; 4597  1624:
; 4598  LDB:    READ [AR],                      ;LOOK AT POINTER
; 4599          LOAD BYTE EA, FE_P, 3T,         ;GET STUFF OUT OF POINTER
; 4600          CALL [BYTEA]                    ;COMPUTE EFFECTIVE ADDRESS
; 4601  1625:   VMA_[PC], FETCH                 ;START FETCH OF NEXT INST
; 4602  =0*     READ [AR],                      ;LOOK AT POINTER
; 4603          FE_FE.AND.S#, S#/0770,          ;MASK OUT JUNK IN FE
; 4604          BYTE DISP,                      ;DISPATCH ON BYTE SIZE
; 4605          CALL [LDB1]                     ;GET BYTE
; 4606          AC_[AR], CLR FPD,               ;STORE AC
; 4607          J/NIDISP                        ;GO DO NEXT INST
; 4608
; 4609  1630:
; 4610  IDPB:   CALL IBP                        ;BUMP BYTE POINTER
; 4611  1634:
; 4612  DPB:    [ARX]_AC*2                      ;PUT 7 BIT BYTE IN 28-34
; 4613          AD/A, A/ARX, SCAD/A,            ;PUT THE BYTE INTO
; 4614          SCADA/BYTE5, 3T,                ; INTO THE FE REGISTER
; 4615          LOAD FE                         ; FE REGISTER
; 4616          [ARX]_AC                        ;PUT BYTE IN ARX
; 4617  =100    READ [AR],                      ;LOOK AT BYTE POINTER
; 4618          LOAD BYTE EA,                   ;LOAD UP EFFECTIVE ADDRESS
; 4619          CALL [BYTEA]                    ;COMPUTE EFFECTIVE ADDRESS
; 4620          READ [AR],                      ;LOOK AT POINTER AGAIN
; 4621          BYTE DISP,                      ;DISPATCH ON SIZE
; 4622          CALL [DPB1]                     ;GO STORE BYTE
; 4623  =111    CLR FPD, J/DONE                 ;ALL DONE
; 4624  =
; 4625
```

```
U 0350, 3076,3441,0503,4174,4007,0700,0200,0003,0002
U 0351, 0004,4443,0000,4174,4004,1700,0000,0000,0000
U 0352, 3075,3770,0503,4334,4017,0700,0000,0032,6000
U 0353, 0004,4443,0000,4174,4004,1700,0000,0000,0000

U 3075, 3076,0111,0703,4170,4007,0700,0200,0003,0002
U 3076, 0004,3333,0003,4175,5004,1701,0200,0000,0002

U 3077, 0070,4443,0000,2174,4006,6700,0000,0000,0000
U 3100, 0070,4443,0000,2174,4466,6700,0000,0003,0000

U 0070, 3102,0553,0300,2274,4007,0700,0200,0004,0712

U 0072, 3102,3443,0300,4174,4007,0700,0200,0004,0712

U 0074, 3101,0553,0300,2274,4007,0700,0200,0004,0512

U 0076, 3101,3443,0300,4174,4007,0700,0200,0004,0512

U 3101, 3077,3771,0003,4361,5217,0700,0200,0000,0502

U 3102, 0001,4551,1205,4365,5004,1700,0200,0000,0002
```

```
;      BYTE GROUP -- INCREMENT BYTE POINTER SUBROUTINE

; 4626  .TOC    "BYTE GROUP -- INCREMENT BYTE POINTER SUBROUTINE"
; 4627
; 4628  =00
; 4629  IBPS:   [AR]_[BR], START WRITE, J/IBPX  ;NO OVERFLOW, BR HAS ANSWER
; 4630     .    RETURN [4]                      ;FIRST PART DONE SET
; 4631          SET P TO 36-S, J/NXTWRD         ;WORD OVERFLOW
; 4632          RETURN [4]                      ;FPD WAS SET IGNORE OVERFLOW
; 4633                                          ; AND RETURN
; 4634
; 4635  NXTWRD: [AR]_[AR]+1, HOLD LEFT, START WRITE     ;BUMP Y AND RETURN
; 4636  IBPX:   MEM WRITE, MEM_[AR], RETURN [4]
; 4637
; 4638
; 4639  .TOC    "BYTE GROUP -- BYTE EFFECTIVE ADDRESS EVALUATOR"
; 4640
; 4641  ;ENTER WITH POINTER IN AR
; 4642  ;RETURN1 WITH (EA) IN VMA AND WORD IN BR
; 4643  BYTEAS: EA MODE DISP,            ;HERE TO AVOID FPD
; 4644          J/BYTEAO                ;GO COMPUTE EA
; 4645  BYTEA:  SET FPD,                 ;SET FIRST-PART-DONE
; 4646          EA MODE DISP            ;DISPATCH
; 4647  =100*
; 4648  BYTEAO: VMA_[AR]+XR,            ;INDEXING
; 4649          START READ,             ;FETCH DATA WORD
; 4650          PXCT BYTE DATA,         ;FOR PXCT
; 4651          J/BYTFET                ;GO WAIT
; 4652          VMA_[AR],               ;PLAIN
; 4653          START READ,             ;START CYCLE
; 4654          PXCT BYTE DATA,         ;FOR PXCT
; 4655          J/BYTFET                ;GO WAIT
; 4656          VMA_[AR]+XR,            ;BOTH
; 4657          START READ,             ;START CYCLE
; 4658          PXCT BYTE PTR EA,       ;FOR PXCT
; 4659          J/BYTIND                ;GO DO INDIRECT
; 4660          VMA_[AR],               ;JUST @
; 4661          START READ,             ;START READ
; 4662          PXCT BYTE PTR EA        ;FOR PXCT
; 4663  BYTIND: MEM READ,               ;WAIT FOR @ WORD
; 4664          [AR]_MEM,               ;PUT IN AR
; 4665          HOLD LEFT,              ;JUST IN RH (SAVE P & S)
; 4666          LOAD BYTE EA,           ;LOOP BACK
; 4667          J/BYTEAS                ; ..
; 4668
; 4669  BYTFET: MEM READ,               ;WAIT FOR BYTE DATA
; 4670          [BR]_MEM.AND.MASK,      ; WORD. UNSIGNED
; 4671          RETURN [1]              ;RETURN TO CALLER
; 4672
```

```
                                                ; 4673  .TOC    "BYTE GROUP -- LOAD BYTE SUBROUTINE"
                                                ; 4674
                                                ; 4675  ;CALL WITH:
                                                ; 4676  ;       WORD IN BR
                                                ; 4677  ;       POINTER IN AR
                                                ; 4678  ;       P IN FE
                                                ; 4679  ;       BYTE DISPATCH
                                                ; 4680  ;RETURN2 WITH BYTE IN AR
                                                ; 4681  LDB SCAD        "SCAD/A,BYTE/BYTE5"
                                                ; 4682  7-BIT LDB       "AD/D,DBUS/DBM,DBM/DP,DEST/A,A/BR,B/BR, LDB SCAD"
                                                ; 4683
                                                ; 4684  =000
                                                ; 4685  LDB1:   GEN 17-FE, 3T,          ;GO SEE IF ALL THE BITS
                                                ; 4686          SCAD DISP,              ; ARE IN THE LEFT HALF
U 0340, 0550,4443,0000,4174,4006,7701,0000,0031,0210    ; 4687          J/LDBSWP                ;GO TO LDBSWP & SKIP IF LH
                                                ; 4688
                                                ; 4689  ;HERE ARE THE 7-BIT BYTES
U 0341, 3103,3770,0505,4334,4057,0700,0000,0073,0000    ; 4690  =001    7-BIT LDB, SCADA/BYTE1, J/LDB7
U 0342, 3103,3770,0505,4334,4057,0700,0000,0074,0000    ; 4691  =010    7-BIT LDB, SCADA/BYTE2, J/LDB7
U 0344, 3103,3770,0505,4334,4057,0700,0000,0075,0000    ; 4692  =100    7-BIT LDB, SCADA/BYTE3, J/LDB7
U 0345, 3103,3770,0505,4334,4057,0700,0000,0076,0000    ; 4693  =101    7-BIT LDB, SCADA/BYTE4, J/LDB7
U 0347, 3103,3770,0505,4334,4057,0700,0000,0077,0000    ; 4694  =111    7-BIT LDB, SCADA/BYTE5, J/LDB7
                                                ; 4695  =
                                                ; 4696
                                                ; 4697  ;FOR 7-BIT BYTES WE HAVE BYTE IN BR 28-35 AND JUNK IN REST OF BR.
                                                ; 4698  ; WE JUST MASK THE SELECTED BYTE AND SHIFT ONE PLACE RIGHT.
                                                ; 4699  LDB7:   AD/ZERO,RSRC/DA,        ;LH_ZERO, RH_D.AND.A
                                                ; 4700          DBUS/DBM,DBM/#,#/376,   ;D INPUT IS 376
                                                ; 4701          A/BR,                   ;A IS BR
                                                ; 4702          B/AR,                   ;PUT RESULT IN AR
                                                ; 4703          DEST/AD*.5, 3T,         ;SHIFT RESULT 1 PLACE
U 3103, 0002,4257,0503,4374,4004,1701,0000,0000,0376    ; 4704          RETURN [2]              ;RETURN TO CALLER
                                                ; 4705
```

```
U 0550, 3105,4443,0000,4174,4007,0700,1000,0031,0000
U 0552, 3104,3770,0505,4344,4007,0700,0000,0000,0000

U 3104, 3105,4221,0005,4174,0007,0700,1000,0031,0220

U 3105, 3106,3447,0505,4174,4007,0700,1020,0041,0010
U 3106, 3107,3333,0003,4174,4007,0700,1000,0031,7770
U 3107, 3110,4222,0000,4174,4007,0700,0000,0000,0000

U 3110, 3111,4224,0003,4174,4027,0700,1020,0041,0010
U 3111, 3112,4224,0003,4174,4027,0700,0000,0000,0000
U 3112, 0002,4001,0503,4174,4004,1700,0000,0000,0000
```

```
; 4706  ;HERE FOR NORMAL BYTES
; 4707  =00
; 4708  LDBSWP: FE_-FE,                    ;MAKE P NEGATIVE
; 4709          J/LDBSH                    ;JOIN MAIN LDB LOOP
; 4710  =10     [BR]_[BR] SWAP             ;SHIFT 18 STEPS
; 4711  =
; 4712          [BR]_0, HOLD RIGHT,        ;PUT ZERO IN LH
; 4713          FE_-FE+S#, S#/220          ;UPDATE FE
; 4714  LDBSH:  [BR]_[BR]*.5,              ;SHIFT RIGHT
; 4715          FE_FE+10,                  ;UPDATE THE FE
; 4716          MULTI SHIFT/1              ;FAST SHIFT
; 4717          READ [AR], FE_-S-10        ;GET SIZE
; 4718          Q_0                        ;CLEAR Q
; 4719          GEN MSK [AR],              ;PUT MASK IN Q (WIPEOUT AR)
; 4720          FE_FE+10,                  ;COUNT UP ALL STEPS
; 4721          MULTI SHIFT/1              ;FAST SHIFT
; 4722          GEN MSK [AR]               ;ONE MORE BIT
; 4723          [AR]_[BR].AND.Q, RETURN [2]
; 4724
```

```
; 4725          .NOBIN
; 4726  .TOC    "BYTE GROUP -- DEPOSIT BYTE IN MEMORY"
; 4727
; 4728  ;FLOW FOR DPB (NOT 7-BIT BYTE)
; 4729  ;
; 4730  ;FIRST SET ARX TO -1 AND Q TO ZERO AND ROTATE LEFT
; 4731  ; S PLACES GIVING:
; 4732
; 4733  ;               ARX             Q
; 4734  ;       +-----------------!-----------------+
; 4735  ;       !1111111111111000000!0000000000000111111!
; 4736  ;       +-----------------!-----------------+
; 4737  ;                                       !<--->!
; 4738  ;                                       S BITS
; 4739  ;
; 4740
; 4741  ;NOW THE AC IS LOAD INTO THE ARX AND BOTH THE ARX AND Q
; 4742  ; ARE SHIFTED LEFT P BITS GIVING:
; 4743
; 4744  ;       +-----------------!-----------------+
; 4745  ;       !??????BBBBBB000000!000000111111000000!
; 4746  ;       +-----------------!-----------------+
; 4747  ;        <----><----->           <----><---->
; 4748  ;         JUNK  BYTE              MASK P BITS
; 4749  ;
; 4750
; 4751  ;AT THIS POINT WE ARE ALMOST DONE. WE NEED TO AND
; 4752  ; THE BR WITH .NOT. Q TO ZERO THE BITS FOR THE BYTE
; 4753  ; AND AND ARX WITH Q TO MASK OUT THE JUNK THIS GIVES:
; 4754  ;
; 4755  ;               ARX
; 4756  ;       +-----------------+
; 4757  ;       !000000BBBBBB000000!
; 4758  ;       +-----------------!
; 4759  ;
; 4760  ;               AR
; 4761  ;       +-----------------+
; 4762  ;       !DDDDDD000000DDDDDD!
; 4763  ;       +-----------------+
; 4764  ;
; 4765  ;WE NOW OR THE AR WITH ARX TO GENERATE THE ANSWER.
; 4766
```

```
U 0360, 3115,3333,0003,4174,4007,0700,1000,0031,7770
U 0361, 3113,3770,0503,4334,4017,0700,0000,0041,0000
U 0362, 3113,3770,0503,4334,4027,0700,0000,0041,0000
U 0364, 3113,3770,0503,4334,4037,0700,0000,0041,0000
U 0365, 3113,3770,0503,4334,4047,0700,0000,0041,0000
U 0367, 3113,3770,0503,4334,4057,0700,0000,0041,0000

U 3113, 3114,3447,1200,4174,4007,0700,0200,0003,0002
U 3114, 0002,3333,0003,4175,5004,1701,0200,0000,0002


U 3115, 3116,4222,0000,4174,4007,0700,0000,0000,0000


U 3116, 3117,4224,0000,4174,4027,0700,1020,0041,0010
U 3117, 3120,4224,0000,4174,4027,0700,0000,0000,0000
U 3120, 3121,3333,0003,4174,4007,0701,1000,0073,0000
U 3121, 3122,4443,0000,4174,4007,0700,1000,0051,0770

U 3122, 3123,4002,1200,4174,4007,0700,1000,0031,0000


U 3123, 3124,3444,0404,4174,4007,0700,1020,0041,0010


U 3124, 3125,7221,0003,4174,4007,0700,0000,0000,0000
U 3125, 3126,4111,0503,4174,4007,0700,0000,0000,0000
U 3126, 3127,4001,0404,4174,4007,0700,0000,0000,0000

U 3127, 3113,3111,0403,4174,4007,0700,0000,0000,0000
```

```
; 4767          .BIN
; 4768
; 4769  ;DEPOSIT BYTE SUBROUTINE
; 4770  ;CALL WITH:
; 4771  ;       BYTE POINTER IN AR
; 4772  ;       BYTE TO STORE IN ARX
; 4773  ;       WORD TO MERGE WITH IN BR
; 4774  ;       (E) OF BYTE POINTER IN VMA
; 4775  ;       7-BIT BYTE IN FE
; 4776  ;       BYTE DISPATCH
; 4777  ;RETURN2 WITH BYTE IN MEMORY
; 4778  ;
; 4779  DPB SCAD        "SCAD/A+B,SCADA/S#,SCADB/FE,S#/O"
; 4780  7-BIT DPB       "AD/D,DEST/A,A/BR,DBUS/DBM,DBM/DP,B/AR, DPB SCAD"
; 4781
; 4782  =000
; 4783  DPB1:   READ [AR], FE_-S-10, J/DPBSLO   ;NOT SPECIAL
; 4784  =001    7-BIT DPB, BYTE/BYTE1, J/DPB7
; 4785  =010    7-BIT DPB, BYTE/BYTE2, J/DPB7
; 4786  =100    7-BIT DPB, BYTE/BYTE3, J/DPB7
; 4787  =101    7-BIT DPB, BYTE/BYTE4, J/DPB7
; 4788  =111    7-BIT DPB, BYTE/BYTE5, J/DPB7
; 4789  =
; 4790  DPB7:   [MAG]_[MASK]*.5, START WRITE
; 4791          MEM WRITE, MEM_[AR], RETURN [2]
; 4792
; 4793
; 4794  DPBSLO: Q_0                      ;CLEAR Q
; 4795          GEN MSK [MAG],           ;GENERATE MASK IN Q (ZAP MAG)
; 4796          FE_FE+10,                ;COUNT STEPS
; 4797          MULTI SHIFT/1            ;FAST SHIFT
; 4798          GEN MSK [MAG]            ;ONE MORE BITS
; 4799          READ [AR], 3T, FE_P      ;AMOUNT TO SHIFT
; 4800          FE_FE.AND.S#, S#/0770    ;MASK OUT JUNK
; 4801          Q_Q.AND.[MASK],          ;CLEAR BITS 36 AND 37
; 4802          FE_-FE                   ;MINUS NUMBER OF STEPS
; 4803          [ARX]_[ARX]*2 LONG,      ;SHIFT BYTE AND MASK
; 4804          FE_FE+10,                ;COUNT OUT STEPS
; 4805          MULTI SHIFT/1            ;FAST SHIFT
; 4806  ;AT THIS POINT WE HAVE DONE ALL THE SHIFTING WE NEED. THE BYTE IS
; 4807  ; IN ARX AND THE MASK IS IN Q.
; 4808          [AR]_.NOT.Q
; 4809          [AR]_[AR].AND.[BR]
; 4810          [ARX]_[ARX].AND.Q
; 4811          [AR]_[AR].OR.[ARX],
; 4812          J/DPB7
; 4813
```

```
U 3130,  1302,3770,0304,4344,4007,0700,2000,0071,0011
U 1302,  1302,3447,0404,4174,4007,0630,2000,0060,0000
U 1303,  3131,4557,0404,4374,4007,0701,0000,0000,0176
U 3131,  3132,3771,0004,4374,0007,0700,0000,0000,0000
U 3132,  3133,3333,0004,7174,4007,0700,0400,0000,0221
U 3133,  3134,4557,0305,4374,4007,0701,0000,0000,7700
U 3134,  3135,3770,0505,4344,4007,0700,2000,0071,0003
U 3135,  1304,4221,0005,4174,0007,0700,0000,0000,0000
```

```
; 4814  .TOC    "BYTE GROUP -- ADJUST BYTE POINTER"
; 4815  .IF/FULL
; 4816  ;FIRST THE NUMBER OF BYTES PER WORD IS COMPUTED FROM THE
; 4817  ; FOLLOWING FORMULA:
; 4818  ;
; 4819  ;                          (  P  )      ( 36-P )
; 4820  ;  BYTES PER WORD = INT( --- ) + INT( ---- )
; 4821  ;                          (  S  )      (  S   )
; 4822  ;
; 4823  ;THIS GIVES 2 BYTES PER WORD FOR THE FOLLOWING 12 BIT BYTE:
; 4824  ;       !====================================!
; 4825  ;       !  6  !////////////!     12     !  6   !
; 4826  ;       !====================================!
; 4827  ;              P=18 AND S=12
; 4828  ;
; 4829  ;WE GET 3 BYTES/WORD IF THE BYTES FALL IN THE NATURAL PLACE:
; 4830  ;       !====================================!
; 4831  ;       !    12     !\\\\\\\\\\\\!     12     !
; 4832  ;       !====================================!
; 4833  ;              P=12 AND S=12
; 4834
; 4835  ;WE COME HERE WITH THE BYTE POINTER IN AR, AND ADJPTR
; 4836  ADJBP:  [ARX]_[AR] SWAP,          ;MOVE SIZE OVER
; 4837          SC_9.                    ;READY TO SHIFT
; 4838  =O
; 4839  ADJBPO: [ARX]_[ARX]*.5,           ;SHIFT P OVER
; 4840          STEP SC,                 ; ..
; 4841          J/ADJBPO                 ; ..
; 4842          [ARX]_([ARX].AND.#)*.5,  ;SHIFT AND MASK
; 4843          3T,                      ;WAIT
; 4844          #/176                    ;6 BIT MASK
; 4845          [ARX]_#,                 ;CLEAR LH
; 4846          #/0,                     ; ..
; 4847          HOLD RIGHT               ; ..
; 4848          WORK[ADJP]_[ARX]         ;SAVE P
; 4849          [BR]_([AR].AND.#)*.5,    ;START ON S
; 4850          3T,                      ;EXTRACT S
; 4851          #/007700                 ; ..
; 4852          [BR]_[BR] SWAP,          ;SHIFT 18 PLACES
; 4853          SC_3                     ; ..
; 4854          [BR]_0,                  ;CLEAR LH
; 4855          HOLD RIGHT               ; ..
; 4856
```

```
U 1304,  1304,3447,0505,4174,4007,0630,2000,0060,0000
U 1305,  1306,3333,0005,7174,4007,0621,0400,0000,0222
U 1306,  0660,3442,0400,4174,4007,0700,2000,0071,0042
U 1307,  1404,3771,0003,7274,4007,0701,0000,0000,0223
U 0660,  0370,3771,0003,4374,4007,0700,0010,0000,0000
U 0662,  3136,3223,0000,7174,4007,0700,0400,0000,0224
U 3136,  3137,3772,0000,4370,4007,0700,0000,0000,0044
U 3137,  3140,1662,0000,7274,4007,0701,4000,0000,0221
U 3140,  3141,3771,0005,7274,4007,0701,0000,0000,0222
U 3141,  0664,4443,0000,4174,4007,0700,2000,0071,0042
U 0664,  0370,3771,0003,4374,4007,0700,0010,0000,0000
U 0666,  3142,3333,0003,7174,4007,0700,0400,0000,0225
U 3142,  3143,3771,0003,4374,4007,0700,0000,0077,7777
U 3143,  1310,0661,0005,7274,4007,0622,0000,0000,0224
U 1310,  0570,0662,0000,0274,4007,0522,2000,0071,0042
U 1311,  0555,4443,0000,4174,4467,0700,0000,0051,1000
```

```
; 4857  =O
; 4858  ADJBP1: [BR]_[BR]*.5,          ;SHIFT S OVER
; 4859          STEP SC,               ; ..
; 4860          J/ADJBP1               ; ..
; 4861          WORK[ADJS]_[BR],       ;SALT S AWAY
; 4862          SKIP AD.EQ.O           ;SEE IF ZERO
; 4863  =O      Q_[ARX],               ;DIVIDE P BY S
; 4864          SC_34.,                ;STEP COUNT
; 4865          J/ADJBP2               ;SKIP NEXT WORD
; 4866          [AR]_WORK[ADJPTR], J/MOVE       ;S=O -- SAME AS MOVE
; 4867  =O*
; 4868  ADJBP2: [AR]_#,                ;FILL AR WITH SIGN BITS
; 4869          #/O,                   ;POSITIVE
; 4870          CALL [DIVSUB]          ;GO DIVIDE
; 4871          WORK[ADJQ1]_Q          ;SAVE QUOTIENT
; 4872          Q_#,                   ;COMPUTE (36-P)/S
; 4873          #/36.,                 ; ..
; 4874          HOLD LEFT              ;SMALL ANSWER
; 4875          Q_Q-WORK[ADJP]         ;SUBTRACT P
; 4876          [BR]_WORK[ADJS]        ;DIVIDE BY S
; 4877          SC_34.                 ;STEP COUNT
; 4878  =O*     [AR]_#,                ;MORE SIGN BITS
; 4879          #/O,                   ; ..
; 4880          CALL [DIVSUB]          ;GO DIVIDE
; 4881          WORK[ADJR2]_[AR]       ;SAVE REMAINDER
; 4882          [AR]_#,                ;ASSUME NEGATIVE ADJ
; 4883          #/777777               ;EXTEND SIGN
; 4884          AD/D+Q,                ;BR_(P/S)+((36-P)/S)
; 4885          DEST/AD,               ; ..
; 4886          B/BR,                  ; ..
; 4887          RAMADR/#,              ; ..
; 4888          DBUS/RAM,              ; ..
; 4889          WORK/ADJQ1,            ; ..
; 4890          4T,                    ; ..
; 4891          SKIP AD.EQ.O           ;SEE IF ZERO
; 4892  =O      Q_Q+AC,                ;GET ADJUSTMENT
; 4893          SC_34.,                ;STEP COUNT
; 4894          SKIP DPO,              ;GO DO DIVIDE
; 4895          4T,                    ;WAIT FOR DP
; 4896          J/ADJBP3               ;BELOW
; 4897          NO DIVIDE              ;O BYTES/WORD
; 4898
```

```
U 0570, 0571,3771,0003,4374,4007,0700,0000,0000,0000

U 0571, 0370,3333,0005,7174,4007,0700,0410,0000,0226

U 0573, 3144,3772,0000,4374,0007,0700,0000,0000,0000

U 3144, 1312,3333,0003,4174,4007,0421,0000,0000,0000

U 1312, 3146,0661,0005,7274,4407,0701,0000,0000,0223

U 1313, 3145,1002,0700,4170,4007,0700,4000,0000,0000

U 3145, 1312,0551,0303,7274,4007,0701,0000,0000,0226
```

```
; 4899  ;WE NOW DIVIDE THE ADJUSTMENT BY THE BYTES PER WORD AND FORCE THE
; 4900  ; REMAINDER (R) TO BE A POSITIVE NUMBER (MUST NOT BE ZERO). THE
; 4901  ; QUOTIENT IS ADDED TO THE Y FIELD IN THE BYTE POINTER AND THE NEW
; 4902  ; P FIELD IS COMPUTED BY:
; 4903  ;
; 4904  ;              (                ( 36-P ))
; 4905  ; NEW P = 36-((R * S) +  RMDR( ---- ))
; 4906  ;              (                (   S  ))
; 4907  ;
; 4908  ;WE NOW HAVE BYTES/WORD IN BR AND ADJUSTMENT IN Q. DIVIDE TO GET
; 4909  ; WORDS TO ADJUST BY.
; 4910  =OO
; 4911  ADJBP3: [AR]_#,                  ;POSITIVE ADJUSTMENT
; 4912          #/O.
; 4913          WORK[ADJBPW]_[BR],       ;SAVE BYTES/WORD & COMPUTE
; 4914          CALL [DIVSUB]            ; ADJ/(BYTES/WORD)
; 4915  ;WE NOW WANT TO ADJUST THE REMAINDER SO THAT IT IS POSITIVE
; 4916  =11     Q_#,                     ;ONLY RIGHT HALF
; 4917          #/O,                     ; ..
; 4918          HOLD RIGHT               ; ..
; 4919  =
; 4920          READ [AR],               ;ALREADY +
; 4921          SKIP AD.LE.O             ; ..
; 4922  =O
; 4923  ADJBP4: AD/D+Q,                  ;ADD Q TO POINTER AND STORE
; 4924          DEST/AD,                 ; ..
; 4925          B/BR,                    ;RESULT TO BR
; 4926          RAMADR/#,                ;PTR IS IN RAM
; 4927          DBUS/RAM,                ; ..
; 4928          WORK/ADJPTR,             ; ..
; 4929          INH CRY18,               ;JUST RH
; 4930          3T,                      ;WAIT FOR RAM
; 4931          J/ADJBP5                 ;CONTINUE BELOW
; 4932          Q_Q-1,                   ;NO--MAKE Q SMALLER
; 4933          HOLD LEFT                ; ..
; 4934          [AR]_[AR]+WORK[ADJBPW],  ;MAKE REM BIGGER
; 4935          J/ADJBP4                 ;NOW HAVE + REMAINDER
; 4936
```

```
U 3146, 3147,3441,0306,4174,4007,0700,2000,0071,0043
U 3147, 0062,3772,0000,7274,4007,0701,0000,0000,0222

U 0062, 3001,3446,0606,4174,4007,0700,0010,0000,0000




U 0066, 3150,0661,0003,7274,4007,0701,0000,0000,0225


U 3150, 3151,2555,0303,4374,4007,0701,4000,0000,0044

U 3151, 3152,3770,0303,4344,4007,0700,2000,0071,0011


U 3152, 1314,3771,0003,4370,4007,0700,0000,0000,0000




U 1314, 1314,3445,0303,4174,4007,0630,2000,0060,0000


U 1315, 3153,4551,0505,4374,0007,0700,0000,0000,7777

U 3153, 1400,3113,0305,0174,4007,0700,0400,0000,0000
```

```
; 4937  ADJBP5: [BRX]_[AR],              ;COMPUTE R*S
; 4938           SC_35.                  ;STEP COUNT
; 4939           Q_WORK[ADJS]            ;GET S
; 4940  =01*     [BRX]_[BRX]*.5 LONG,    ;SHIFT OVER
; 4941           CALL [MULSUB]           ; ..
; 4942           AD/D+Q,                 ;AR_(R*S)+RMDR(36-P)/S
; 4943           DEST/AD,                ; ..
; 4944           B/AR,                   ; ..
; 4945           RAMADR/#,               ; ..
; 4946           3T,                     ; ..
; 4947           DBUS/RAM,               ; ..
; 4948           WORK/ADJR2              ; ..
; 4949           [AR]_(#-[AR])*2,        ;COMPUTE 36-AR
; 4950           3T,                     ;AND START LEFT
; 4951           #/36.                   ; ..
; 4952           [AR]_[AR] SWAP,         ;PUT THE POSITION BACK
; 4953           SC_9.                   ; ..
; 4954           [AR]_#,                 ;CLEAR JUNK FROM RH
; 4955           #/0,                    ; ..
; 4956           HOLD LEFT               ; ..
; 4957  =0
; 4958  ADJBP6: [AR]_[AR]*2,             ;LOOP OVER ALL BITS
; 4959           STEP SC,                ; ..
; 4960           J/ADJBP6                ; ..
; 4961           [BR]_[BR].AND.#,        ; ..
; 4962           #/007777,               ; ..
; 4963           HOLD RIGHT              ; ..
; 4964           AC_[AR].OR.[BR],        ;ALL DONE
; 4965           J/DONE
;;4966  .IFNOT/FULL
;;4967
;;4968  ADJBP:  UUO                      ;NO ADJBP IN SMALL
;;4969                                           ; MICROCODE
; 4970  .ENDIF/FULL
; 4971
```

```
; 4972          .NOBIN
; 4973  .TOC    "BLT"
; 4974
; 4975  ;THIS CODE PROVIDES A GUARANTEED RESULT IN AC ON COMPLETION OF
; 4976  ; THE TRANSFER (EXCEPT IN THE CASE AC IS PART OF BUT NOT THE LAST WORD
; 4977  ; OF THE DESTINATION BLOCK).  WHEN AC IS NOT PART OF THE DESTINATION
; 4978  ; BLOCK, IT IS LEFT CONTAINING THE ADDRESSES OF THE FIRST WORD FOLLOWING
; 4979  ; THE SOURCE BLOCK (IN THE LH), AND THE FIRST WORD FOLLOWING THE DEST-
; 4980  ; INATION BLOCK (IN THE RH).  IF AC IS THE LAST WORD OF THE DESTINATION
; 4981  ; BLOCK, IT WILL BE A COPY OF THE LAST WORD OF THE SOURCE BLOCK.
; 4982
; 4983  ;IN ADDITION, A SPECIAL-CASE CHECK IS MADE FOR THE CASE IN WHICH EACH
; 4984  ; WORD STORED IS USED AS THE SOURCE OF THE NEXT TRANSFER.  IN THIS CASE,
; 4985  ; ONLY ONE READ NEED BE PERFORMED, AND THAT DATA MAY BE STORED FOR EACH
; 4986  ; TRANSFER.  THUS THE COMMON USE OF BLT TO CLEAR CORE IS SPEEDED UP.
; 4987
```

```
D 0251, 0000,1640,2100

U 1640, 3154,3771,0006,0276,6007,0700,0000,0000,0000
U 3154, 1316,3770,0604,4344,4007,0700,0000,0000,0000

U 1316, 3664,3443,0400,4174,4007,0700,0210,0004,0712

U 1317, 3155,4221,0006,4174,0007,0700,0000,0000,0000
U 3155, 3156,2112,0306,4174,4007,0700,4000,0000,0000
U 3156, 3157,0001,0705,4174,4007,0700,0000,0000,0000

U 3157, 3160,3770,0505,4344,0007,0700,0000,0000,0000

U 3160, 3161,0551,0505,0274,4407,0701,0000,0000,0000
U 3161, 1320,3771,0013,4370,4007,0700,0000,0000,0001

U 1320, 3663,3440,0505,0174,4007,0700,0410,0000,0000

U 1321, 3162,0551,0405,4370,4007,0701,0000,0000,0001

U 3162, 1322,2113,0506,4174,4007,0331,4000,0000,0000

U 1322, 3165,3443,0600,4174,4007,0700,0200,0003,0312
```

```
        BLT

; 4988          .BIN
; 4989          .DCODE
; 4990  251:    I,              J/BLT
; 4991          .UCODE
; 4992
; 4993  1640:
; 4994  BLT:    [BRX]_AC                ;FETCH THE AC (DEST IN RH)
; 4995          [ARX]_[BRX] SWAP        ;COPY TO ARX (SRC IN RH)
; 4996  =O      VMA_[ARX],              ;ADDRESS OF FIRST WORD
; 4997          START READ,
; 4998          PXCT BLT SRC,
; 4999          CALL [CLARXL]           ;CLEAR THE LEFT HALF OF
; 5000          [BRX]_O,                ; BOTH SRC AND DEST
; 5001          HOLD RIGHT
; 5002          Q_[AR]-[BRX]            ;NUMBER OF WORDS TO MOVE
; 5003          [BR]_Q+1                ;LENGTH +1
; 5004          [BR]_[BR] SWAP,         ;COPY TO BOTH HALFS
; 5005          HOLD RIGHT
; 5006          [BR]_AC+[BR],           ;FINAL AC
; 5007          INH CRY18               ;KEEP AC CORRECT IF DEST IS 777777
; 5008          STATE_[BLT]             ;SET PAGE FAIL FLAGS
; 5009  =O      AC_[BR],                ;STORE BACK IN AC
; 5010          CALL [LOADQ]            ;LOAD FIRST WORD INTO Q
; 5011          [BR]_[ARX]+1000001,     ;SRC+1
; 5012          3T,
; 5013          HOLD LEFT
; 5014          [BR]-[BRX], 3T,         ;IS THIS THE CORE CLEAR CASE
; 5015          SKIP ADR.EQ.O
; 5016  =O
; 5017  BLTLP1: VMA_[BRX],
; 5018          START WRITE,
; 5019          PXCT BLT DEST,          ;WHERE TO STORE
; 5020          J/BLTGO
; 5021
```

```
U 1323,  3163,3443,0600,4174,4007,0700,0200,0003,0312
U 3163,  1324,3223,0000,4174,4007,0671,0200,0000,0002
U 1324,  3166,4443,0000,4174,4007,0700,0000,0000,0000
U 1325,  1326,2113,0603,4174,4007,0521,4000,0000,0000
U 1326,  1400,4221,0013,4170,4007,0700,0000,0000,0000
U 1327,  1330,0111,0704,4174,4007,0370,0000,0000,0000
U 1330,  3163,0111,0706,4170,4007,0700,0200,0003,0312
U 1331,  3165,0111,0706,4170,4007,0700,0200,0003,0312
U 3164,  1322,3772,0000,4365,5007,0700,0200,0000,0002
U 3165,  3166,3223,0000,4174,4007,0701,0200,0000,0002
U 3166,  1332,2113,0603,4174,4007,0521,4000,0000,0000
U 1332,  1400,4221,0013,4170,4007,0700,0000,0000,0000
U 1333,  3167,0111,0706,4174,4007,0700,0000,0000,0000
U 3167,  3164,0111,0704,4170,4007,0700,0200,0004,0712
U 3170,  3171,3770,0303,4344,4007,0700,0000,0000,0000
U 3171,  3172,3771,0003,7270,4007,0701,0000,0000,0214
U 3172,  1100,3440,0303,0174,4007,0700,0400,0000,0000
```

```
; 5022  ;CLEAR CORE CASE
; 5023          VMA_[BRX],
; 5024          START WRITE,
; 5025          PXCT BLT DEST
; 5026  BLTCLR: MEM WRITE,                ;STORE WORD
; 5027          MEM_Q,
; 5028          SKIP/-1 MS                ;1 MS TIMER UP
; 5029  =O      J/BLTGOT                  ;GO TAKE INTERRUPT
; 5030          [BRX]-[AR],               ;BELOW E?
; 5031          3T,
; 5032          SKIP DPO
; 5033  =O      END BLT,                  ;NO--STOP BLT
; 5034          J/DONE
; 5035          [ARX]_[ARX]+1,            ;FOR PAGE FAIL LOGIC
; 5036          SKIP IRPT
; 5037  =O      VMA_[BRX]+1,
; 5038          LOAD VMA,
; 5039          PXCT BLT DEST,
; 5040          START WRITE,              ;YES--KEEP STORING
; 5041          J/BLTCLR
; 5042          VMA_[BRX]+1,              ;INTERRUPT
; 5043          LOAD VMA,
; 5044          PXCT BLT DEST,
; 5045          START WRITE,
; 5046          J/BLTGO
; 5047
; 5048  ;HERE FOR NORMAL BLT
; 5049  BLTLP:  MEM READ,                 ;FETCH
; 5050          Q_MEM,
; 5051          J/BLTLP1
; 5052  BLTGO:  MEM WRITE,                ;STORE
; 5053          MEM_Q
; 5054  BLTGOT: [BRX]-[AR],               ;BELOW E?
; 5055          3T,
; 5056          SKIP DPO
; 5057  =O      END BLT,                  ;NO--STOP BLT
; 5058          J/DONE
; 5059          [BRX]_[BRX]+1             ;UPDATE DEST ADDRESS
; 5060          VMA_[ARX]+1,
; 5061          LOAD VMA,
; 5062          PXCT BLT SRC,
; 5063          START READ,               ;YES--MOVE 1 MORE WORD
; 5064          J/BLTLP
; 5065
; 5066  ;HERE TO CLEAN UP AFTER BLT PAGE FAILS
; 5067  BLT-CLEANUP:
; 5068          [AR]_[AR] SWAP            ;PUT SRC IN LEFT HALF
; 5069          [AR]_WORK[SV.BRX],
; 5070          HOLD LEFT
; 5071          AC_[AR],                  ;STORE THE AC AND RETURN
; 5072          J/CLEANED
; 5073
```

```
D 0140, 0701,1577,1100
D 0142, 0702,1577,1700
D 0143, 0703,1577,1700
D 0144, 0711,1577,1100
D 0145, 0611,1577,0100
D 0146, 0712,1577,1700
D 0147, 0713,1577,1700

D 0150, 0701,1576,1100
D 0152, 0702,1576,1700
D 0153, 0703,1576,1700
D 0154, 0711,1576,1100
D 0155, 0611,1576,0100
D 0156, 0712,1576,1700
D 0157, 0713,1576,1700



U 1576, 1577,2441,0303,4174,4007,0700,4000,0000,0000
;5074  .TOC    "FLOATING POINT -- FAD, FSB"
;5075
;5076          .DCODE
;5077  140:    FL-R,   FL-AC,           J/FAD
;5078  142:    FL-RW,  FL-MEM,          J/FAD
;5079          FL-RW,  FL-BOTH,         J/FAD
;5080          FL-R,   FL-AC, ROUND,    J/FAD
;5081          FL-I,   FL-AC, ROUND,    J/FAD
;5082          FL-RW,  FL-MEM, ROUND,   J/FAD
;5083          FL-RW,  FL-BOTH, ROUND,  J/FAD
;5084
;5085  150:    FL-R,   FL-AC,           J/FSB
;5086  152:    FL-RW,  FL-MEM,          J/FSB
;5087          FL-RW,  FL-BOTH,         J/FSB
;5088          FL-R,   FL-AC, ROUND,    J/FSB
;5089          FL-I,   FL-AC, ROUND,    J/FSB
;5090          FL-RW,  FL-MEM, ROUND,   J/FSB
;5091          FL-RW,  FL-BOTH, ROUND,  J/FSB
;5092          .UCODE
;5093
;5094  ;BOTH FAD & FSB ARE ENTERED WITH THE MEMORY OPERAND IN AR
;5095  ; SIGN SMEARED. THE EXPONENT IN BOTH SC AND FE.
;5096  1576:
;5097  FSB:    [AR]_-[AR]               ;MAKE MEMOP NEGATIVE
;5098
;5099  1577:
U 1577, 0670,3771,0005,0276,6006,7701,2000,0020,2000
;5100  FAD:    [BR]_AC, SC_SC-EXP-1, 3T, SCAD DISP
;5101  =0*
U 0670, 1336,3333,0005,4174,4007,0520,0000,0000,0000
;5102  FAS1:   READ [BR], SKIP DPO, J/FAS2     ;BR .LE. AR
U 0672, 3173,3441,0304,4174,4007,0700,0000,0000,0000
;5103          [ARX]_[AR]               ;SWAP AR AND BR
U 3173, 3174,3441,0503,4174,4007,0700,2000,0041,2000
;5104          [AR]_[BR], SC_EXP
U 3174, 3175,3441,0405,4174,4007,0700,2000,0020,0000
;5105          [BR]_[ARX], SC_SC-FE-1   ;NUMBER OF SHIFT STEPS
U 3175, 1334,3333,0003,4174,4007,0520,1000,0041,2000
;5106          READ [AR], FE_EXP, 2T, SKIP DPO
U 1334, 3176,4551,0303,4374,0007,0700,0000,0000,0777
;5107  =0       [AR]_+SIGN, J/FAS3
U 1335, 3176,3551,0303,4374,0007,0700,0000,0077,7000
;5108          [AR]_-SIGN, J/FAS3
;5109
;5110  =0       ;SIGN SMEAR BR AND UNNORMALIZE
U 1336, 3176,4551,0505,4374,0007,0700,0000,0000,0777
;5111  FAS2:   [BR]_+SIGN, J/FAS3
U 1337, 3176,3551,0505,4374,0007,0700,0000,0077,7000
;5112          [BR]_-SIGN, J/FAS3
;5113
U 3176, 1340,4222,0000,4174,4007,0630,2000,0060,0000
;5114  FAS3:   Q_0, STEP SC
;5115  =0
U 1340, 1340,3446,0505,4174,4047,0630,2000,0060,0000
;5116  FAS4:   [BR]_[BR]*.5 LONG, STEP SC, ASHC, J/FAS4
U 1341, 0420,0111,0503,4174,4003,4701,0000,0000,0000
;5117          [AR]_[AR]+[BR], NORM DISP, J/SNORM
;5118
```

```
D 0160, 0701,1570,1100
D 0162, 0702,1570,1700
D 0163, 0703,1570,1700

D 0164, 0711,1570,1100
D 0165, 0611,1570,0100
D 0166, 0712,1570,1700
D 0167, 0713,1570,1700




U 1570, 1342,3771,0006,0276,6007,0521,1000,0040,2000
U 1342, 3177,4551,0606,4374,0007,0700,0000,0000,0777
U 1343, 3177,3551,0606,4374,0007,0700,0000,0077,7000
U 3177, 0163,3442,0300,4174,4007,0700,2000,0071,0033

U 0163, 3001,3446,0606,4174,4007,0700,0010,0000,0000

U 0167, 3200,4662,0000,4370,4007,0700,0000,0077,7000
U 3200, 3201,3441,0403,4174,4007,0700,1000,0041,0002


U 3201, 0420,3446,0303,4174,4003,4701,1000,0041,1600
```

```
; 5119  .TOC     "FLAOTING POINT -- FMP"
; 5120
; 5121          .DCODE
; 5122  160:     FL-R,    FL-AC,              J/FMP
; 5123  162:     FL-RW,   FL-MEM,             J/FMP
; 5124          FL-RW,   FL-BOTH,            J/FMP
; 5125
; 5126          FL-R,    FL-AC, ROUND,       J/FMP
; 5127          FL-I,    FL-AC, ROUND,       J/FMP
; 5128          FL-RW,   FL-MEM, ROUND,      J/FMP
; 5129          FL-RW,   FL-BOTH, ROUND,     J/FMP
; 5130          .UCODE
; 5131
; 5132  1570:
; 5133  FMP:     [BRX]_AC,                    ;GET AC
; 5134          FE_SC+EXP, 3T,               ;EXPONENT OF ANSWER
; 5135          SKIP.DPO                     ;GET READY TO SMEAR SIGN
; 5136  =0       [BRX]_+SIGN, J/FMP1          ;POSITIVE
; 5137          [BRX]_-SIGN, J/FMP1          ;NEGATIVE
; 5138  FMP1:    Q_[AR], SC_27.               ;GET MEMORY OPERAND
; 5139  =01*     [BRX]_[BRX]*.5 LONG,         ;SHIFT RIGHT
; 5140          CALL [MULSUB]                ;MULTIPLY
; 5141          Q_Q.AND.#, #/777000,         ;WE ONLY COMPUTED
; 5142          HOLD LEFT                    ; 27 BITS
; 5143          [AR]_[ARX], FE_FE+2          ;SET SHIFT PATHS
; 5144          [AR]_[AR]*.5 LONG,           ;SHIFT OVER
; 5145          FE_FE-200,                   ;ADJUST EXPONENT
; 5146          NORM DISP, J/SNORM           ;NORMALIZE & EXIT
; 5147
```

```
D 0170, 0701,1574,1100
D 0172, 0702,1574,1700
D 0173, 0703,1574,1700

D 0174, 0711,1574,1100
D 0175, 0611,1574,0100
D 0176, 0712,1574,1700
D 0177, 0713,1574,1700



U 1574, 1344,3441,0305,0174,4007,0621,0000,0000,0000


U 1344, 1346,3771,0003,0276,6007,0520,1000,0030,2000
U 1345, 0555,4443,0000,4174,4467,0700,0000,0071,1000

U 1346, 3202,4551,0303,4374,0007,0700,0000,0000,0777
U 1347, 3203,3551,0303,4374,0007,0700,0000,0077,7000
U 3202, 3204,3441,0304,4174,4007,0700,1000,0031,0200
U 3203, 3204,2441,0304,4174,4007,0700,5000,0031,0200
U 3204, 1350,3445,0506,4174,4007,0520,0000,0000,0000

U 1350, 1352,2113,0406,4174,4007,0311,4000,0000,0000
U 1351, 1350,2445,0506,4174,4007,0700,4000,0000,0000

U 1352, 1354,3447,0606,4174,4007,0700,0000,0000,0000
U 1353, 0555,4443,0000,4174,4467,0700,0000,0071,1000


U 1354, 3666,3445,0303,4174,4007,0700,0010,0000,0000
U 1355, 0144,2113,0604,4174,4007,0421,4000,0000,0000

U 0144, 3044,4222,0000,4174,4007,0700,2010,0071,0033
U 0145, 0144,3447,0303,4174,4007,0700,1000,0041,0001

U 0154, 3205,3227,0003,4174,4007,0700,0000,0000,0000

U 0155, 2056,3333,0003,4174,4007,0621,0010,0000,0000

U 0156, 2056,3333,0003,4174,4007,0621,0010,0000,0000
U 0157, 3205,3227,0003,4174,4007,0700,0000,0000,0000
U 0177, 3205,2227,0003,4174,4007,0700,4000,0000,0000

U 3205, 1365,4222,0000,4174,4007,0700,0000,0000,0000
```

```
; 5148  .TOC    "FLOATING POINT -- FDV"
; 5149
; 5150          .DCODE
; 5151  170:    FL-R,    FL-AC,            J/FDV
; 5152  172:    FL-RW,   FL-MEM,           J/FDV
; 5153          FL-RW,   FL-BOTH,          J/FDV
; 5154
; 5155          FL-R,    FL-AC, ROUND,     J/FDV
; 5156          FL-I,    FL-AC, ROUND,     J/FDV
; 5157          FL-RW,   FL-MEM, ROUND,    J/FDV
; 5158          FL-RW,   FL-BOTH, ROUND,   J/FDV
; 5159          .UCODE
; 5160
; 5161
; 5162  1574:
; 5163  FDV:    [BR]_[AR], SKIP AD.EQ.O, AC     ;COPY DIVSOR SEE IF O
; 5164  =O
; 5165          [AR]_AC, FE_SC-EXP, SKIP DPO,   ;GET AC & COMPUTE NEW
; 5166                  J/FDVO                  ; EXPONENT
; 5167          FL NO DIVIDE                    ;DIVIDE BY ZERO
; 5168  =O
; 5169  FDVO:   [AR]_+SIGN, J/FDV1
; 5170          [AR]_-SIGN, J/FDV2
; 5171  FDV1:   [ARX]_[AR],FE_-FE+200,J/FDV3    ;COMPUTE 2*DVND
; 5172  FDV2:   [ARX]_-[AR],FE_-FE+200,J/FDV3   ;ABSOLUTE VALUE
; 5173  FDV3:   [BRX]_[BR]*2, SKIP DPO  ;ABSOLUTE VALUE
; 5174  =O
; 5175  FDV4:   [ARX]-[BRX], SKIP CRYO, 3T, J/FDV5      ;FLOATING NO DIV?
; 5176          [BRX]_-[BR]*2, J/FDV4           ;FORCE ABSOLUTE VALUE
; 5177  =O
; 5178  FDV5:   [BRX]_[BRX]*.5, J/FDV6          ;SHIFT BACK ARX
; 5179          FL NO DIVIDE                    ;UNNORMALIZED INPUT
; 5180  =O
; 5181  FDV6:   [AR]_[AR]*2,                    ;DO NOT DROP A BIT
; 5182          CALL [SBRL]                     ;AT FDV7+1
; 5183          [BRX]-[ARX], SKIP AD.LE.O       ;IS ANSWER .LE. 1?
; 5184  =00100
; 5185  FDV7:   Q_O, SC_27., CALL [DIVSGN]      ;DIVIDE
; 5186  =00101  [AR]_[AR]*.5, FE_FE+1, J/FDV7   ;SCALE DV'END
; 5187  =01100
; 5188  FDV8:   [AR]_Q*.5, J/FDV9               ;PUT ANSWER IN AR
; 5189  =01101  READ [AR], SKIP AD.EQ.O,        ;-VE ANSWER, LOOK AT RMDR
; 5190          CALL [SETSN]                    ; SEE HOW TO NEGATE
; 5191  =01110  READ [AR], SKIP AD.EQ.O,        ;-VE ANSWER, LOOK AT RMDR
; 5192          CALL [SETSN]                    ; SEE HOW TO NEGATE
; 5193  =01111  [AR]_Q*.5, J/FDV9               ;PUT ANSWER IN AR
; 5194  =11111  [AR]_-Q*.5, J/FDV9              ;ZERO RMDR
; 5195
; 5196  FDV9:   Q_O, J/SNORMO                   ;GO NORMALIZE
; 5197
```

```
D 0127, 0011,1616,1100
D 0132, 0001,1621,2100



U 1616, 1356,4553,0300,4374,4007,0321,0000,0077,7000
U 1356, 1360,2441,0305,4174,4007,0521,4000,0000,0000
U 1357, 1365,4222,0000,4174,4007,0700,1000,0071,0233


U 1360, 1362,4553,0500,4374,4007,0321,0000,0077,7000
U 1361, 3206,4222,0000,4174,4007,0700,1000,0071,0244

U 1362, 3206,4222,0000,4174,4007,0700,1000,0071,0244
U 1363, 1365,4222,0000,4174,4007,0700,1000,0071,0233



U 3206, 1364,3446,0303,4174,4047,0700,2000,0071,0006


U 1364, 1364,3446,0303,4174,4047,0630,2000,0060,0000
U 1365, 0420,3333,0003,4174,4003,4701,0000,0000,0000



U 1621, 3207,3333,0003,4174,4007,0700,2000,0041,4000
U 3207, 3210,4222,0000,0174,4007,0700,0000,0000,0000
U 3210, 1366,3771,0003,0276,6007,0520,1000,0040,2000
U 1366, 1365,4551,0303,4374,0007,0700,0000,0000,0777
U 1367, 1365,3551,0303,4374,0007,0700,0000,0077,7000
```

```
; 5198  .TOC    "FLOATING POINT -- FLTR, FSC"
; 5199
; 5200          .DCODE
; 5201  127:    R,      FL-AC,ROUND,    J/FLTR
; 5202  132:    I,      FL-AC,          J/FSC
; 5203          .UCODE
; 5204
; 5205  1616:
; 5206  FLTR:   [AR].AND.#, #/777000, 3T, SKIP ADL.EQ.0 ;SMALL POS NUMBER?
; 5207  =0      [BR]_-[AR], SKIP DPO, 3T, J/FLTR1       ;NO--SEE IF MINUS
; 5208          Q_0, FE_S#, S#/233, J/SNORMO    ;FITS IN 27 BITS
; 5209  =0
; 5210  FLTR1:  [BR].AND.#, #/777000, 3T,
; 5211                  SKIP ADL.EQ.0, J/FLTR1A ;SMALL NEGATIVE NUMBER
; 5212          Q_0, FE_S#, S#/244, J/FLTR2     ;LARGE POS NUMBER
; 5213  =0
; 5214  FLTR1A: Q_0, FE_S#, S#/244, J/FLTR2     ;BIG NUMBER
; 5215          Q_0, FE_S#, S#/233, J/SNORMO    ;FITS IN 27 BITS
; 5216  ;AT THIS POINT WE KNOW THE NUMBER TAKES MORE THAN 27 BITS. WE JUST
; 5217  ; SHIFT 8 PLACES RIGHT AND NORMALIZE. WE COULD BE MORE CLEVER BUT
; 5218  ; THIS IS THE RARE CASE ANYWAY.
; 5219  FLTR2:  [AR]_[AR]*.5 LONG, ASHC, SC_6   ;SHOVE OVER TO THE RIGHT
; 5220  =0
; 5221  FLTR3:  [AR]_[AR]*.5 LONG, ASHC,        ;SHIFT RIGHT 9 PLACES
; 5222                  STEP SC, J/FLTR3        ; SO IT WILL FIT
; 5223  SNORMO: READ [AR], NORM DISP, J/SNORM   ;NORMALIZE ANSWER
; 5224
; 5225
; 5226  1621:
; 5227  FSC:    READ [AR], SC_SHIFT
; 5228          Q_0, AC                         ;DON'T SHIFT IN JUNK
; 5229          [AR]_AC, FE_SC+EXP, SKIP DPO    ;SIGN SMEAR
; 5230  =0      [AR]_+SIGN, J/SNORMO
; 5231          [AR]_-SIGN, J/SNORMO
; 5232
```

```
D 0122, 0701,1626,1100
D 0126, 0711,1626,1100



U 1626, 0674,4222,0000,4174,4006,7701,0000,0041,1534
U 0674, 0555,4443,0000,4174,4467,0700,0000,0041,1000
U 0676, 0720,4443,0000,4174,4006,7701,2000,0041,1544
U 0720, 1372,4443,0000,4174,4007,0630,2000,0060,0000
U 0722, 3211,4443,0000,4174,4007,0700,2000,0031,0232

U 3211, 1370,4443,0000,4174,4007,0630,2000,0060,0000


U 1370, 1370,3446,0303,4174,4047,0630,2000,0060,0000
U 1371, 0063,3447,0705,4174,4003,7700,0000,0000,0000


U 1372, 1372,3445,0303,4174,4007,0630,2000,0060,0000
U 1373, 0100,3440,0303,0174,4156,4700,0400,0000,0000


U 0063, 1374,3333,0003,4174,4007,0520,0000,0000,0000
U 0073, 1514,0111,0503,4174,4003,7700,0200,0003,0001

U 1374, 0100,3440,0303,0174,4156,4700,0400,0000,0000
U 1375, 1376,3223,0000,4174,4007,0621,0000,0000,0000
U 1376, 1514,0111,0703,4174,4003,7700,0200,0003,0001

U 1377, 0073,7441,1205,4174,4007,0700,0000,0000,0000
```



```
; 5233  .TOC    "FLOATING POINT -- FIX AND FIXR"
; 5234
; 5235      .   .DCODE
; 5236  122:    FL-R,   FL-AC,          J/FIX
; 5237  126:    FL-R,   FL-AC,ROUND,    J/FIX
; 5238          .UCODE
; 5239
; 5240  1626:
; 5241  FIX:    Q_0, SCAD/A+B, SCADA/S#,        ;CLEAR Q, SEE IF
; 5242                 S#/1534, SCADB/FE, 3T,   ; ANSWER FITS IN
; 5243                 SCAD DISP                ; 35 BITS.
; 5244  =0*     SET AROV, J/NIDISP              ;TOO BIG
; 5245          SC_FE+S#, S#/1544, 3T, SCAD DISP ;NEED TO MOVE LEFT?
; 5246  =0*     STEP SC, J/FIXL
; 5247          SC_S#-FE, S#/232                ;NUMBER OF PLACES TO SHIFT
; 5248                                          ; RIGHT
; 5249          STEP SC                         ;ALREADY THERE
; 5250  =0
; 5251  FIXR:   [AR]_[AR]*.5 LONG, ASHC,        ;SHIFT BINARY POINT
; 5252                 STEP SC, J/FIXR          ; TO BIT 35.5
; 5253          [BR]_[ONE]*.5, B DISP, J/FIXX   ;WHICH KIND OF FIX?
; 5254
; 5255  =0
; 5256  FIXL:   [AR]_[AR]*2, STEP SC, J/FIXL    ;SHIFT LEFT
; 5257          AC_[AR], NEXT INST              ;WE ARE NOW DONE
; 5258
; 5259  =0*11
; 5260  FIXX:   READ [AR], SKIP DPO, J/FIXT     ;FIX--SEE IF MINUS
; 5261  FIXX1:  [AR]_[AR]+[BR], FL-EXIT         ;FIXR--ROUND UP
; 5262  =0
; 5263  FIXT:   AC_[AR], NEXT INST              ;FIX & +, TRUNCATE
; 5264          READ Q, SKIP AD.EQ.0            ;NEGATIVE--ANY FRACTION?
; 5265  =0      [AR]_[AR]+1, FL-EXIT            ;YES--ROUND UP
; 5266          [BR]_.NOT.[MASK],               ;MAYBE--GENERATE .75
; 5267          J/FIXX1                         ;ROUND UP IF BIT 36 OR
; 5268                                          ; 37 SET
; 5269
```

```
U 0420, 0420,3444,0303,4174,4063,4701,1000,0041,1777
U 0421, 2002,2222,0000,4174,4007,0311,4000,0000,0000
U 0422, 0262,3333,0003,4174,4003,4701,0010,0000,0000
U 0423, 2002,2222,0000,4174,4007,0311,4000,0000,0000
U 0424, 0262,3447,0303,4174,4007,0700,1010,0041,0001
U 0425, 2002,2222,0000,4174,4007,0311,4000,0000,0000
U 0426, 0262,3447,0303,4174,4007,0700,1010,0041,0001
U 0427, 2002,2222,0000,4174,4007,0311,4000,0000,0000
U 0430, 2000,3223,0000,4174,4007,0621,0000,0000,0000
U 0436, 2001,3770,0303,4324,0457,0700,0000,0041,0000


U 2000, 0420,3444,0303,4174,4063,4701,1000,0041,1777
U 2001, 1514,4443,0000,4174,4003,7700,0200,0003,0001
```

```
; 5270  .TOC    "FLOATING POINT -- SINGLE PRECISION NORMALIZE"
; 5271
; 5272  ;NORMALIZE DISPATCH IS A 9-WAY DISPATCH. THE HARDWARE LOOKS AT
; 5273  ; 4 SIGNALS: DP=0, DP BIT 8, DP BIT 9, DP BIT -2. THE 9 CASES
; 5274  ; ARE:
; 5275
; 5276  ;       DP=0    DP08    DP09    DP00    ACTION TO TAKE
; 5277  ;       0       0       0       0       SHIFT LEFT
; 5278  ;
; 5279  ;       0       0       0       1       NEGATE AND RETRY
; 5280  ;
; 5281  ;       0       0       1       0       ALL DONE
; 5282  ;
; 5283  ;       0       0       1       1       NEGATE AND RETRY
; 5284  ;
; 5285  ;       0       1       0       0       SHIFT RIGHT
; 5286  ;
; 5287  ;       0       1       0       1       NEGATE AND RETRY
; 5288  ;
; 5289  ;       0       1       1       0       SHIFT RIGHT
; 5290  ;
; 5291  ;       0       1       1       1       NEGATE AND RETRY
; 5292  ;
; 5293  ;       1       -       -       -       LOOK AT Q BITS
; 5294
; 5295  ;ENTER HERE WITH UNNORMALIZED NUMBER IN AR!Q. FE HOLDS THE NEW
; 5296  ; EXPONENT. CALL WITH NORM DISP
; 5297  =0000              ;9-WAY DISPATCH
; 5298  SNORM:  [AR]_[AR]*2 LONG, DIV, FE_FE-1, NORM DISP, J/SNORM
; 5299          Q_-Q, SKIP CRYO, 3T, J/SNNEG
; 5300          READ [AR], NORM DISP, CALL [SROUND]
; 5301          Q_-Q, SKIP CRYO, 3T, J/SNNEG
; 5302          [AR]_[AR]*.5, FE_FE+1, CALL [SROUND]
; 5303          Q_-Q, SKIP CRYO, 3T, J/SNNEG
; 5304          [AR]_[AR]*.5, FE_FE+1, CALL [SROUND]
; 5305          Q_-Q, SKIP CRYO, 3T, J/SNNEG
; 5306          READ Q, SKIP AD.EQ.0, J/SNORM1
; 5307  =1110   [AR]_EXP, J/FLEX
; 5308  =
; 5309  =0
; 5310  SNORM1: [AR]_[AR]*2 LONG, DIV, FE_FE-1, NORM DISP, J/SNORM
; 5311  FLEX:   FL-EXIT
; 5312
```

```
U 2002, 0440,7441,0303,4174,4003,4701,0000,0000,0000
U 2003, 0440,2441,0303,4174,4003,4701,4000,0000,0000

U 0440, 0440,3444,0303,4174,4063,4701,1000,0041,1777
U 0442, 0262,3333,0003,4174,4003,4701,0010,0000,0000
U 0444, 0262,3447,0303,4174,4007,0700,1010,0041,0001
U 0446, 0262,3447,0303,4174,4007,0700,1010,0041,0001
U 0450, 0440,3444,0303,4174,4063,4701,1000,0041,1777
U 0456, 0327,3770,0303,4324,0453,7700,0000,0041,0000

U 0327, 2004,4553,1300,4374,4007,0321,0000,0000,2000

U 0337, 2007,4111,1203,4174,4007,0700,0000,0000,0000

U 2004, 3212,7441,0303,4174,4007,0700,0000,0000,0000
U 2005, 2006,3223,0000,4174,4007,0621,0000,0000,0000
U 2006, 3212,7441,0303,4174,4007,0700,0000,0000,0000
U 2007, 3212,2441,0303,4174,4007,0700,4000,0000,0000
U 3212, 1514,4221,0013,4174,4003,7700,0200,0003,0001




U 0262, 0407,3447,0705,4174,4003,7700,0000,0000,0000
U 0266, 0262,3447,0303,4174,4007,0700,1000,0041,0001

U 0407, 0016,4443,0000,4174,4004,1700,0000,0000,0000
U 0417, 0302,0111,0503,4174,4003,4701,0000,0000,0000
U 0302, 0016,4443,0000,4174,4004,1700,0000,0000,0000
U 0306, 0016,3447,0303,4174,4004,1700,1000,0041,0001
```

```
; 5313  =0
; 5314  SNNEG:  [AR]_.NOT.[AR], NORM DISP, J/SNNORM ;NEGATE HIGH WORD
; 5315                                              ; (NO CARRY)
; 5316          [AR]_-[AR], NORM DISP, J/SNNORM ;NEGATE HIGH WORD (W/CARRY)
; 5317  =0000
; 5318  SNNORM: [AR]_[AR]*2 LONG, DIV, FE_FE-1, NORM DISP, J/SNNORM
; 5319  =0010   READ [AR], NORM DISP, CALL [SROUND]
; 5320  =0100   [AR]_[AR]*.5, FE_FE+1, CALL [SROUND]
; 5321  =0110   [AR]_[AR]*.5, FE_FE+1, CALL [SROUND]
; 5322  =1000   [AR]_[AR]*2 LONG, DIV, FE_FE-1, NORM DISP, J/SNNORM
; 5323  =1110   [AR]_EXP, B DISP
; 5324  =
; 5325  =0111   TL [FLG], FLG.SN/1, J/SNNOT
; 5326          [AR]_[AR].AND.[MASK],   ;CLEAR ANY LEFT OVER BITS
; 5327          J/SNNOT1
; 5328  =0
; 5329  SNNOT:  [AR]_.NOT.[AR], J/SNNOT2
; 5330          READ Q, SKIP AD.EQ.0
; 5331  =0      [AR]_.NOT.[AR], J/SNNOT2
; 5332  SNNOT1: [AR]_-[AR], J/SNNOT2    ;NORMAL NEGATE AND EXIT
; 5333  SNNOT2: [FLG]_0, FL-EXIT
; 5334
; 5335
; 5336
; 5337  .TOC    "FLOATING POINT -- ROUND ANSWER"
; 5338
; 5339  =*01*
; 5340  SROUND: [BR]_[ONE]*.5, B DISP, J/SRND1
; 5341          [AR]_[AR]*.5, FE_FE+1, J/SROUND ;WE WENT TOO FAR
; 5342  =0111
; 5343  SRND1:  RETURN [16]                     ;NOT ROUNDING INSTRUCTION
; 5344          [AR]_[AR]+[BR], NORM DISP
; 5345  =*01*   RETURN [16]
; 5346          [AR]_[AR]*.5, FE_FE+1, RETURN [16]
; 5347
```

```
D 0110, 1100,1637,1100
D 0111, 1100,1635,1100

U 1635, 3213,2441,0404,4174,4007,0700,4000,0000,0000
U 3213, 1637,2441,0303,4174,4007,0700,0040,0000,0000

U 1637, 3214,4557,0006,1274,4007,0701,0000,0000,1441

U 3214, 2010,3777,0005,0274,4007,0521,2000,0020,2000

U 2010, 2012,5547,0505,0374,4007,0631,0000,0077,7400

U 2011, 2012,3547,0505,0374,4007,0631,0000,0077,7400

U 2012, 2014,3442,0600,4174,4007,0700,0000,0000,0000

U 2013, 3216,3771,0016,0276,6007,0700,2000,0041,2000
```

```
; 5348  .TOC    "FLOATING POINT -- DFAD, DFSB"
; 5349
; 5350          .DCODE
; 5351  110:    DBL FL-R,                J/DFAD
; 5352  111:    DBL FL-R,                J/DFSB
; 5353          .UCODE
; 5354
; 5355  ;ENTER FROM A-READ CODE WITH:
; 5356  ;FE/    EXP
; 5357  ;SC/    EXP
; 5358  ;AR/    C(E) SHIFT RIGHT 2 PLACES
; 5359  ;ARX/   C(E+1) SHIFTED RIGHT 1 PLACE
; 5360  1635:
; 5361  DFSB:   [ARX]_-[ARX]             ;NEGATE LOW WORD
; 5362          [AR]_-[AR]-.25, MULTI PREC/1
; 5363  1637:
; 5364  DFAD:   [BRX]_(AC[1].AND.[MAG])*.5, 3T ;GET LOW WORD
; 5365          [BR]_AC*.5, 3T,          ;GET AC AND START TO SHIFT
; 5366          SC_SC-EXP-1,             ;NUMBER OF PLACES TO SHIFT
; 5367          SKIP DPO                 ;SEE WHAT SIGN
; 5368  =0      [BR]_+SIGN*.5, 3T,       ;SIGN SMEAR
; 5369          AC, SKIP/SC, J/DFAS1     ;SEE WHICH IS BIGGER
; 5370          [BR]_-SIGN*.5, 3T,       ;SIGN SMEAR
; 5371          AC, SKIP/SC, J/DFAS1     ;SEE WHICH IS BIGGER
; 5372  =0
; 5373  DFAS1:  Q_[BRX],                 ;AR IS BIGGER
; 5374          J/DFAS2                  ;ADJUST BR!Q
; 5375          [TO]_AC,                 ;BR IS BIGGER OR EQUAL
; 5376          SC_EXP, 2T, J/DFAS3      ;SET SC TO THAT EXPONENT
; 5377
```

```
U 2014, 0153,3441,0516,4174,4007,0700,0010,0000,0000
U 2015, 3215,3441,1605,4174,4007,0700,0000,0000,0000
U 3215, 3221,0002,0400,4174,4007,0700,0000,0000,0000

U 3216, 3217,3442,0400,4174,4007,0700,2000,0020,0000
U 3217, 2016,3333,0016,4174,4007,0700,1000,0041,2000
U 2016, 0153,3441,0316,4174,4007,0700,0010,0000,0000
U 2017, 3220,3441,1603,4174,4007,0700,0000,0000,0000
U 3220, 3221,0002,0600,4174,4007,0700,0000,0000,0000

U 3221, 3222,0116,0503,4174,4047,0700,0040,0000,0000

U 3222, 0433,3444,0303,4174,4046,2700,0000,0000,0000

U 0433, 3223,5111,1217,4174,4007,0700,0000,0000,0000
U 0437, 0433,0222,0000,4174,4007,0700,4000,0000,0000

U 3223, 3224,3444,0303,4174,4047,0700,1000,0041,1777

U 3224, 3225,3444,0303,4174,4047,0700,1000,0041,1777
U 3225, 2031,3002,1700,4170,4007,0700,0000,0000,0000
```

```
; 5378  ;HERE IF AR!ARX IS GREATER THAN BR!BRX
; 5379  =0
; 5380  DFAS2:  [TO]_[BR], CALL [DFADJ] ;ADJUST BR!Q
; 5381          [BR]_[TO]               ;PUT ANSWER BACK
; 5382          Q_Q+[ARX], J/DFAS5      ;ADD LOW WORDS
; 5383
; 5384  ;HERE IS BR!BRX IF GREATER THAN OR EQUAL TO AR!ARX
; 5385  DFAS3:  Q_[ARX],                ;SETUP TO SHIFT AR!ARX
; 5386          SC_SC-FE-1              ;COMPUTE # OF PLACES
; 5387          READ [TO], FE_EXP       ;EXPONENT OF ANSWER
; 5388  =0      [TO]_[AR], CALL [DFADJ] ;ADJUST AR!Q
; 5389          [AR]_[TO]               ;PUT ANSWER BACK
; 5390          Q_Q+[BRX], J/DFAS5      ;ADD LOW WORDS
; 5391
; 5392  ;BIT DIDDLE TO GET THE ANSWER (INCLUDING 2 GUARD BITS) INTO
; 5393  ; AR!Q
; 5394  DFAS5:  [AR]_([AR]+[BR])*.5 LONG, ;ADD HIGH WORDS
; 5395          MULTI PREC/1, ASHC      ;INJECT SAVED CRY2
; 5396          [AR]_[AR]*2 LONG,       ;SHIFT BACK LEFT
; 5397          ASHC,'MUL DISP          ;SEE IF WE LOST A 1
; 5398  =1011
; 5399  DFAS6:  [T1]_[T1].AND.NOT.[MASK], J/DFAS7
; 5400          Q_Q+.25, J/DFAS6
; 5401  DFAS7:  [AR]_[AR]*2 LONG, ASHC, ;PUT IN GUARD BITS
; 5402          FE_FE-1
; 5403          [AR]_[AR]*2 LONG, ASHC,
; 5404          FE_FE-1
; 5405          Q_[T1].OR.Q, HOLD LEFT, J/DNORMO
; 5406
```

```
U 0153, 2057,3444,1616,4174,4067,0700,0010,0000,0000
U 0173, 3226,3444,1616,4174,4067,0700,0000,0000,0000
U 3226, 3227,3444,1616,4174,4067,0700,0000,0000,0000

U 3227, 0472,3446,1616,4174,4047,0630,2000,0060,0000


U 0472, 0472,3446,1616,4174,4046,2630,2000,0060,0000

U 0473, 0453,3221,0017,4174,4006,2700,0000,0000,0000
U 0476, 2020,3551,1313,4374,0007,0700,0000,0000,2000
U 0477, 2021,3551,1313,4374,0007,0700,0000,0000,2000


U 2020, 2020,3446,1616,4174,4047,0630,2000,0060,0000
U 2021, 0453,3221,0017,4174,4007,0700,0000,0000,0000

U 0453, 3230,3446,1616,4174,4047,0700,0000,0000,0000
U 0457, 0453,3551,1313,4374,0007,0700,0000,0000,2000
U 3230, 0001,3446,1616,4174,4044,1700,0000,0000,0000
```

```
; 5407  ;SUBROUTINE TO ADJUST NUMBER IN TO!Q
; 5408  ;RETURNS 1 WITH
; 5409  ;       TO!Q ADJUSTED
; 5410  ;       FLG.SN=1 IF WE SHIFTED OUT ANY 1 BITS (STICKY BIT)
; 5411  ;       T1 HAS Q TWO STEPS PRIOR TO BEING DONE
; 5412  DFADJ   "STEP SC, ASHC, MUL DISP"
; 5413
; 5414  =0**11
; 5415  DFADJ:  [TO]_[TO]*2 LONG, DIV,  ;MOVE EVERYTHING 2 PLACES
; 5416          CALL [CLRSN]
; 5417          [TO]_[TO]*2 LONG, DIV
; 5418          [TO]_[TO]*2 LONG, DIV
; 5419          [TO]_[TO]*.5 LONG, ASHC, ;SHIFT AT LEAST 1 PLACE
; 5420          STEP SC
; 5421  =1010
; 5422  DFADJ1: [TO]_[TO]*.5 LONG,       ;UNNORMALIZE TO!Q
; 5423          DFADJ, J/DFADJ1          ;LOOP TILL DONE
; 5424  DFADJ2: [T1]_Q,                  ;SAVE GUARD BITS
; 5425          MUL DISP, J/DFADJ5       ;LOOK AT LAST BIT
; 5426          [FLG]_[FLG].OR.#, FLG.SN/1, HOLD RIGHT, J/DFADJ3
; 5427          [FLG]_[FLG].OR.#, FLG.SN/1, HOLD RIGHT, J/DFADJ4
; 5428
; 5429  =0
; 5430  DFADJ3: [TO]_[TO]*.5 LONG, ASHC, STEP SC, J/DFADJ3
; 5431  DFADJ4: [T1]_Q                   ;SAVE 2 GUARD BITS
; 5432  =1011
; 5433  DFADJ5: [TO]_[TO]*.5 LONG, ASHC, J/DFADJ6
; 5434          [FLG]_[FLG].OR.#, FLG.SN/1, HOLD RIGHT, J/DFADJ5
; 5435  DFADJ6: [TO]_[TO]*.5 LONG, ASHC, RETURN [1]
; 5436
```

```
D 0112, 1105,1631,1100

U 1631, 2022,3442,0400,4174,4007,0700,2000,0071,0006

U 2022, 2022,3444,0303,4174,4047,0630,2000,0060,0000
U 2023, 3231,3446,1200,4174,4007,0700,0000,0000,0000
U 3231, 3232,4662,0000,4374,0007,0700,0000,0007,7777
U 3232, 3233,3221,0005,4174,4007,0700,0000,0000,0000

U 3233, 0623,4557,0006,1274,4007,0700,0000,0000,1441
U 0623, 3002,3447,0606,4174,4007,0700,2010,0071,0043

U 0627, 0730,3442,0300,4174,4007,0700,2000,0071,0043
U 0730, 3003,4443,0000,4174,4007,0700,0010,0000,0000
U 0734, 3234,3441,0416,4174,4007,0700,0000,0000,0000
U 3234, 3235,3227,0004,4174,4007,0700,2000,0011,0000

U 3235, 2024,3777,0006,0274,4007,0521,1000,0040,2000
U 2024, 0732,5547,0606,4374,4007,0701,0000,0077,7400
U 2025, 0732,3547,0606,4374,4007,0701,0000,0077,7400

U 0732, 3003,3442,0500,4174,4007,0700,2010,0071,0043
U 0736, 3236,3221,0017,4174,4007,0700,0000,0000,0000
U 3236, 1012,0111,1604,4174,4007,0700,0000,0000,0000

U 1012, 3003,3442,0300,4174,4007,0700,2010,0071,0043
```

```
; 5437  .TOC    "FLOATING POINT -- DFMP"
; 5438
; 5439          .DCODE
; 5440  112:    DBL FL-R,       DAC,    J/DFMP
; 5441          .UCODE
; 5442
; 5443  ;SAME ENTRY CONDITIONS AS DFAD/DFSB
; 5444  1631:
; 5445  DFMP:   Q_[ARX], SC_6           ;SHIFT MEM OP 8 PLACES
; 5446  =0
; 5447  DFMP1:  [AR]_[AR]*2 LONG; ASHC, ;SHIFT
; 5448          STEP SC, J/DFMP1
; 5449          Q_Q*.5
; 5450          Q_Q.AND.#, #/077777, HOLD RIGHT
; 5451          [BR]_Q                  ;COPY LOW WORD
; 5452  ;
; 5453  ; BRX * BR ==> C(E+1) * C(AC+1)
; 5454  ;
; 5455          [BRX]_(AC[1].AND.[MAG])*.5 ;GET LOW AC
; 5456  =0**    [BRX]_[BRX]*.5, SC_35., CALL [MULSB1]
; 5457  ;
; 5458  ; BRX * Q ==> C(E) * C(AC+1)
; 5459  ;
; 5460          Q_[AR], SC_35.          ;GO MULT NEXT HUNK
; 5461  =0**    CALL [MULTIPLY]
; 5462          [TO]_[ARX]              ;SAVE PRODUCT
; 5463          [ARX]_Q*.5, SC_FE       ;PUT IN NEXT STEP
; 5464  ;
; 5465  ; BRX * BR ==> C(AC) * C(E+1)
; 5466  ;
; 5467          [BRX]_AC*.5,            ;PREPARE TO DO HIGH HALF
; 5468          FE_SC+EXP,              ;EXPONENT ON ANSWER
; 5469          SKIP DPO, 3T
; 5470  =0      [BRX]_+SIGN*.5, 3T, J/DFMP2
; 5471          [BRX]_-SIGN*.5, 3T
; 5472  =0**
; 5473  DFMP2:  Q_[BR], SC_35., CALL [MULTIPLY] ;GO MULTIPLY
; 5474          [T1]_Q                  ;SAVE FOR ROUNDING
; 5475          [ARX]_[ARX]+[TO]        ;PREPARE FOR LAST MUL
; 5476  ;
; 5477  ; BRX * Q ==> C(AC) * C(E)
; 5478  ;
; 5479  =0**    Q_[AR], SC_35.,         ;DO THE LAST MULTIPLY
; 5480          CALL [MULTIPLY]         ; ..
; 5481
```

```
U 1016, 0243,3446,0403,4174,4007,0700,1000,0041,1576

U 0243, 2056,3333,0017,4174,4007,0621,0010,0000,0000
U 0263, 3237,3444;0303,4174,4047,0700,0000,0000,0000
U 3237, 3240,3447,0705,4174,4007,0700,0000,0000,0000
U 3240, 2026,4553,1700,4374,4007,0321,0000,0020,0000
U 2026, 2027,0002,0500,4174,4007,0700,0000,0000,0000
U 2027, 3241,3444,0303,4174,4047,0700,0000,0000,0000
U 3241, 2030,4553,1700,4374,4007,0321,0000,0010,0000
U 2030, 2031,0002,0500,4174,4007,0700,0000,0000,0000

U 2031, 0520,3333,0003,4174,4003,4701,1000,0041,0002
```

```
; 5482  ;OK, WE NOW HAVE THE PRODUCT IN ARX!Q!T1. ALL WE NEED TO DO
; 5483  ; IS SOME BIT DIDDLES TO GET EVERYTHING IN THE RIGHT PLACE
; 5484          [AR]_[ARX]*.5 LONG,      ;SHIFT THE ANSWER
; 5485          FE_FE+S#, S#/1576        ;CORRECT EXPONENT
; 5486  =0**11  READ [T1], SKIP AD.EQ.0, ;SEE IF LOW ORDER 1
; 5487          CALL [SETSN]             ; BITS AROUND SOMEPLACE
; 5488          [AR]_[AR]*2 LONG, ASHC   ;SHIFT LEFT
; 5489          [BR]_[ONE]*.5            ;PLACE TO INSTERT BITS
; 5490          TL [T1], #/200000        ;ANYTHING TO INJECT?
; 5491  =0      Q_Q+[BR]                 ;YES--PUT IT IN
; 5492          [AR]_[AR]*2 LONG, ASHC   ;MAKE ROOM FOR MORE
; 5493          TL [T1], #/100000        ;ANOTHER BIT NEEDED
; 5494  =0      Q_Q+[BR]                 ;YES--PUT IN LAST BIT
; 5495  DNORMO: READ [AR], NORM DISP,    ;SEE WHAT WE NEED TO DO
; 5496          FE_FE+S#, S#/2, J/DNORM  ;ADJUST FOR INITIAL SHIFTS
; 5497
```

```
D 0113, 1105,1636,1100


U 1636, 0132,3441,0406,4174,4007,0700,0000,0000,0000

U 0132, 2057,4221,0017,4174,4007,0700,0010,0000,0000

U 0133, 2032,3441,0305,1174,4007,0421,0000,0000,1441


U 2032, 3244,4557,0004,1274,4007,0700,0000,0000,1441
U 2033, 3242,7441,1717,4174,4007,0700,0000,0000,0000
U 3242, 3243,2441,0606,4174,4007,0700,4000,0000,0000



U 3243, 2032,2331,0005,1174,4007,0521,0040,0000,1441


U 3244, 2034,3777,0003,0274,4007,0521,1000,0030,2000
U 2034, 2036,5547,0303,4374,4007,0701,0000,0077,7400
U 2035, 3245,7441,1717,4174,4007,0700,0000,0000,0000
U 3245, 3246,3547,0303,4374,4007,0701,0000,0077,7400
U 3246, 3247,2442,0400,4174,4007,0700,4000,0000,0000


U 3247, 2037,2446,0303,4174,4047,0700,0040,0000,0000


U 2036, 3043,3442,0400,4174,4007,0700,0010,0000,0000
U 2037, 2040,2113,0305,4174,4007,0521,4000,0000,0000
U 2040, 0555,4443,0000,4174,4467,0700,0000,0071,1000
U 2041, 0724,3221,0004,4174,4007,0700,0000,0000,0000
U 0724, 1264,4222,0000,4174,4007,0700,2010,0071,0032
U 0726, 1054,3221,0016,4174,4007,0700,2000,0071,0043


U 1054, 1264,5002,0000,4174,4007,0621,0010,0000,0000
U 1056, 3250,3446,1200,4174,4007,0700,0000,0000,0000


U 3250, 0513,3444,1616,4174,4046,2700,0000,0000,0000



U 0513, 2042,3333,0017,4174,4007,0520,1000,0031,0202
U 0517, 0513,0222,0000,4174,4007,0700,4000,0000,0000

U 2042, 0520,3441,1603,4174,4003,4701,0000,0000,0000
U 2043, 0200,3441,1603,4174,4003,4701,0000,0000,0000
```

```
; 5498  .TOC    "FLOATING POINT -- DFDV"
; 5499
; 5500          .DCODE
; 5501  113:    DBL FL-R,       DAC,    J/DFDV
; 5502          .UCODE
; 5503  1636:
; 5504  DFDV:   [BRX]_[ARX]                    ;COPY OPERAND (COULD SAVE TIME
; 5505                                         ; WITH SEPERATE A-READ FOR DFDV)
; 5506  =1**10  [T1]_0, CALL [CLRSN]           ;CLEAR FLAG
; 5507          [BR]_[AR], SKIP AD.LE.0,       ;SEE IF POSITIVE
; 5508          AC[1]                          ;WARM UP RAM
; 5509  =0
; 5510  DFDV1:  [ARX]_(AC[1].AND.[MAG])*.5,    ;POSITIVE--GET AC
; 5511          J/DFDV2                        ; AND CONTINUE BELOW
; 5512          [T1]_.NOT.[T1]                 ;DV'SOR NEGATIVE (OR ZERO)
; 5513          [BRX]_-[BRX]                   ;NEGATE LOW WORD
; 5514          AD/-B-.25, B/BR, DEST/AD,      ;NEGATE HIGH WORD
; 5515          MULTI PREC/1, 3T,              ;ADDING IN CRY02
; 5516          SKIP DPO, AC[1],               ;SEE IF STILL NEGATIVE
; 5517          J/DFDV1                        ; ..
; 5518  DFDV2:  [AR]_AC*.5,                    ;GET AC AND SHIFT
; 5519          FE_SC-EXP, 3T,                 ;COMPUTE NEW EXPONENT
; 5520          SKIP DPO                       ;SEE IF NEGATIVE
; 5521  =0      [AR]_+SIGN*.5, 3T, J/DFDV3     ;POSITIVE
; 5522          [T1]_.NOT.[T1]                 ;NEGATIVE OR ZERO
; 5523          [AR]_-SIGN*.5, 3T              ;SIGN SMEAR
; 5524          Q_-[ARX]                       ;NEGATE OPERAND
; 5525          [AR]_(-[AR]-.25)*.5 LONG,      ;NEGATE HIGH WORD
; 5526          MULTI PREC/1,                  ;USE SAVED CARRY
; 5527          ASHC, J/DFDV4                  ;CONTINUE BELOW
; 5528  =0
; 5529  DFDV3:  Q_[ARX],                       ;COPY OPERAND
; 5530          CALL [DDIVS]                   ;SHIFT OVER
; 5531  DFDV4:  [AR]-[BR], 3T, SKIP DPO        ;SEE IF OVERFLOW
; 5532  =0      FL NO DIVIDE
; 5533          [ARX]_Q                        ;START DIVISION
; 5534  =0*     Q_0, SC_26., CALL [DBLDIV]
; 5535          [T0]_Q, SC_35.
; 5536  =0*     Q_Q.AND.NOT.[MAG],             ;SEE IF ODD
; 5537          SKIP AD.EQ.0,                  ;SKIP IF EVEN
; 5538          CALL [DBLDIV]                  ;GO DIVIDE
; 5539          Q_Q*.5                         ;MOVE ANSWER OVER
; 5540  =
; 5541          [T0]_[T0]*2 LONG, ASHC,        ;DO FIRST NORM STEP
; 5542          MUL DISP                       ; SEE IF A 1 FELL OUT
; 5543  =1011
; 5544  DFDV4A: READ [T1], SKIP DPO,           ;SHOULD RESULT BE NEGATIVE
; 5545          FE_S#-FE, S#/202,              ;CORRECT EXPONENT
; 5546          J/DFDV4B                       ;LOOK BELOW
; 5547          Q_Q+.25, J/DFDV4A              ;PUT BACK THE BIT
; 5548  =0
; 5549  DFDV4B: [AR]_[T0], NORM DISP, J/DNORM ;PLUS
; 5550          [AR]_[T0], NORM DISP, J/DNNORM ;MINUS
; 5551
```

```
U 0520, 0520,3444,0303,4174,4043,4701,1000,0041,1777
U 0521, 2046,4553,1300,4374,4007,0321,0000,0000,2000

U 0522, 0322,3333,0003,4174,4003,4701,0010,0000,0000
U 0523, 2046,4553,1300,4374,4007,0321,0000,0000,2000

U 0524, 0322,3446,0303,4174,4047,0700,1010,0041,0001
U 0525, 2046,4553,1300,4374,4007,0321,0000,0000,2000

U 0526, 0322,3446,0303,4174,4047,0700,1010,0041,0001
U 0527, 2046,4553,1300,4374,4007,0321,0000,0000,2000

U 0530, 3252,4002,0000,4174,0007,0700,0000,0000,0000
U 0536, 3251,4221,0013,4174,4007,0700,0000,0000,0000


U 3251, 1515,4113,0400,1174,4007,0700,0400,0000,1441

U 3252, 2044,3223,0000,4174,4007,0621,0000,0000,0000


U 2044, 0520,3444,0303,4174,4043,4701,1000,0041,1777
U 2045, 1515,3440,0303,1174,4007,0700,0400,0000,1441
```

```
; 5552  .TOC    "FLOATING POINT -- DOUBLE PRECISION NORMALIZE"
; 5553
; 5554  ;NORMALIZE AR!Q
; 5555  ;DNORMO:        READ [AR], NORM DISP,   ;SEE WHAT WE NEED TO DO
; 5556  ;       FE_FE+S#, S#/2, J/DNORM ;ADJUST FOR INITIAL SHIFTS
; 5557  =0000
; 5558  DNORM:  [AR]_[AR]*2 LONG,        ;SHIFT LEFT
; 5559          FE_FE-1, ASHC,          ;ADJUST EXPONENT
; 5560          NORM DISP, J/DNORM      ;TRY AGAIN
; 5561          TL [FLG], FLG.SN/1, J/DNEG ;RESULT IS NEGATIVE
; 5562          READ [AR], NORM DISP,   ;SEE IF WE WENT TOO FAR
; 5563          CALL [DROUND]           ; AND ROUND ANSWER
; 5564          TL [FLG], FLG.SN/1, J/DNEG ;RESULT IS NEGATIVE
; 5565          [AR]_[AR]*.5 LONG, ASHC,
; 5566          FE_FE+1, CALL [DROUND]
; 5567          TL [FLG], FLG.SN/1, J/DNEG ;RESULT IS NEGATIVE
; 5568          [AR]_[AR]*.5 LONG, ASHC,
; 5569          FE_FE+1, CALL [DROUND]
; 5570          TL [FLG], FLG.SN/1, J/DNEG ;RESULT IS NEGATIVE
; 5571          Q_[MAG].AND.Q,          ;HIGH WORD IS ZERO
; 5572          HOLD RIGHT, J/DNORM1    ;GO TEST LOW WORD
; 5573  =1110  [FLG]_0                 ;[122] CLEAR FLAG WORD
; 5574  =
; 5575          AC[1]_[ARX].AND.[MAG],  ;STORE LOW WORD
; 5576          J/STAC                  ;GO DO HIGH WORD
; 5577
; 5578
; 5579  DNORM1: READ Q, SKIP AD.EQ.0    ;TEST LOW WORD
; 5580  =0      [AR]_[AR]*2 LONG,       ;LOW WORD IS NON-ZERO
; 5581          FE_FE-1, ASHC,          ;ADJUST EXPONENT
; 5582          NORM DISP, J/DNORM      ;KEEP LOOKING
; 5583          AC[1]_[AR], J/STAC      ;WHOLE ANSWER IS ZERO
; 5584
```

```
U 2046, 3253,7222,0000,4174,4007,0700,0000,0000,0000
U 2047, 2050,2222,0000,4174,4007,0511,4000,0000,0000
U 3253, 2050,4221,0013,4174,4007,0700,0000,0000,0000

U 2050, 0200,7441,0303,4174,4003,4701,0000,0000,0000

U 2051, 0200,2441,0303,4174,4003,4701,4000,0000,0000

U 0200, 0200,3444,0303,4174,4043,4701,1000,0041,1777

U 0202, 0322,3333,0003,4174,4003,4701,0010,0000,0000

U 0204, 0322,3446,0303,4174,4047,0700,1010,0041,0001

U 0206, 0322,3446,0303,4174,4047,0700,1010,0041,0001

U 0210, 3256,4002,0000,4174,0007,0700,0000,0000,0000
U 0216, 0610,4111,1204,4174,4007,0700,0000,0000,0000

U 0610, 3257,4111,0004,4174,4007,0700,0010,0000,0000

U 0612, 3254,6111,0004,4174,4007,0700,0000,0000,0000

U 0613, 2052,2441,0404,4174,4007,0561,4000,0000,0000

U 3254, 2052,4221,0013,4174,4007,0700,0000,0000,0000

U 2052, 3255,7333,0003,0174,4007,0700,0400,0000,0000
U 2053, 3255,2443,0300,0174,4007,0701,4400,0000,0000

U 3255, 0100,4113,0400,1174,4156,4700,0400,0000,1441

U 3256, 2054,3223,0000,4174,4007,0621,0000,0000,0000


U 2054, 0200,3444,0303,4174,4043,4701,1000,0041,1777
U 2055, 1515,3440,0303,1174,4007,0700,0400,0000,1441
```

```
; 5585  ;HERE TO NORMALIZE NEGATIVE D.P. RESULTS
; 5586  =0
; 5587  DNEG:   Q_.NOT.Q, J/DNEG1        ;ONES COMP
; 5588          Q_-Q, SKIP CRY2, J/DNEG2
; 5589  DNEG1:  [FLG]_0
; 5590  =0
; 5591  DNEG2:  [AR]_.NOT.[AR],          ;NO CARRY
; 5592          NORM DISP, J/DNNORM      ;GO NORMALIZE
; 5593          [AR]_-[AR],              ;CARRY
; 5594          NORM DISP, J/DNNORM      ;NORMALIZE
; 5595
; 5596  =000*
; 5597  DNNORM: [AR]_[AR]*2 LONG,         ;SHIFT 1 PLACE
; 5598          FE_FE-1, ASHC,           ;ADJUST EXPONENT
; 5599          NORM DISP, J/DNNORM      ;LOOP TILL DONE
; 5600  =001*  READ [AR], NORM DISP,     ;SEE IF WE WENT TOO FAR
; 5601          CALL [DROUND]            ; AND ROUND ANSWER
; 5602  =010*  [AR]_[AR]*.5 LONG, ASHC,
; 5603          FE_FE+1, CALL [DROUND]
; 5604  =011*  [AR]_[AR]*.5 LONG, ASHC,
; 5605          FE_FE+1, CALL [DROUND]
; 5606  =100*  Q_[MAG].AND.Q,            ;HIGH WORD IS ZERO
; 5607          HOLD RIGHT, J/DNNRM1     ;GO TEST LOW WORD
; 5608  =111*  [ARX]_[ARX].AND.[MASK]    ;REMOVE ROUNDING BIT
; 5609  =
; 5610  =00    [ARX]_[ARX].AND.[MAG],    ;ALSO CLEAR SIGN
; 5611          CALL [CHKSN]             ;ONES COMP?
; 5612  =10    [ARX]_[ARX].XOR.[MAG],    ;YES--ONES COMP
; 5613          J/DNN1                   ;CONTINUE BELOW
; 5614  =11    [ARX]_-[ARX], 3T,         ;NEGATE RESULT
; 5615          SKIP CRY1, J/DNN2
; 5616  =
; 5617  DNN1:  [FLG]_0                   ;CLEAR FLAG
; 5618  =0
; 5619  DNN2:  AC_.NOT.[AR], J/DNORM2
; 5620          AC_-[AR], 3T
; 5621  DNORM2: AC[1]_[ARX].AND.[MAG],    ;STORE LOW WORD
; 5622          NEXT INST                ;ALL DONE
; 5623
; 5624  DNNRM1: READ Q, SKIP AD.EQ.0      ;TEST LOW WORD
; 5625  =0      [AR]_[AR]*2 LONG,         ;LOW WORD IS NON-ZERO
; 5626          FE_FE-1, ASHC,           ;ADJUST EXPONENT
; 5627          NORM DISP, J/DNNORM      ;KEEP LOOKING
; 5628          AC[1]_[AR], J/STAC       ;WHOLE ANSWER IS ZERO
; 5629
```

```
U 3257, 0002,4553,1300,4374,4004,1321,0000,0000,2000    ; 5630  CHKSN:  TL [FLG], FLG.SN/1, RETURN [2]
                                                        ; 5631
                                                        ; 5632  ;SUBROUTINE TO SET/CLEAR FLG.SN
                                                        ; 5633  ;CALL WITH:
                                                        ; 5634  ;       CALL [SETSN], SKIP IF WE SHOULD CLEAR
                                                        ; 5635  ;RETURNS 23
                                                        ; 5636  =0
U 2056, 0023,3551,1313,4374,0004,1700,0000,0000,2000    ; 5637  SETSN:  [FLG]_[FLG].OR.#, FLG.SN/1, HOLD RIGHT, RETURN [23]
U 2057, 0023,5551,1313,4374,0004,1700,0000,0000,2000    ; 5638  CLRSN:  [FLG]_[FLG].AND.NOT.#, FLG.SN/1, HOLD RIGHT, RETURN [23]
                                                        ; 5639
                                                        ; 5640
                                                        ; 5641  ;SUBROUTINE TO ROUND A FLOATING POINT NUMBER
                                                        ; 5642  ;CALL WITH:
                                                        ; 5643  ;       NUMBER IN AR!Q AND NORM DISP
                                                        ; 5644  ;RETURNS 16 WITH ROUNDED NUMBER IN AR!ARX
                                                        ; 5645  ;
                                                        ; 5646  =*01*
                                                        ; 5647  DROUND: [ARX]_(Q+1)*.5,          ;ROUND AND SHIFT
                                                        ; 5648          SKIP CRY2,               ;SEE IF OVERFLOW
U 0322, 0462,0007,0704,4174,4007,0511,0000,0000,0000    ; 5649          J/DRND1                  ;COMPLETE ROUNDING
                                                        ; 5650          [AR]_[AR]*.5 LONG,       ;WE WENT TOO FAR
U 0326, 0322,3446,0303,4174,4047,0700,1000,0041,0001    ; 5651          FE_FE+1, ASHC, J/DROUND ;SHIFT BACK AND ROUND
                                                        ; 5652  =*010
U 0462, 0016,3770,0303,4324,0454,1700,0000,0041,0000    ; 5653  DRND1:  [AR]_EXP, RETURN [16]    ;NO OVERFLOW
                                                        ; 5654  =011    [AR]_[AR]+.25,           ;ADD CARRY (BITS 36 AND 37
                                                        ; 5655                                   ; ARE COPIES OF Q BITS)
                                                        ; 5656          NORM DISP,               ;SEE IF OVERFLOW
U 0463, 0462,0441,0303,4174,4003,4701,4000,0000,0000    ; 5657          J/DRND1          ; ..
                                                        ; 5658  =110    [AR]_[AR]*.5,            ;SHIFT RIGHT
                                                        ; 5659          FE_FE+1,                 ;KEEP EXP RIGHT
U 0466, 0462,3447,0303,4174,4007,0700,1000,0041,0001    ; 5660          J/DRND1          ;ALL SET NOW
                                                        ; 5661  =
                                                        ; 5662
```

```
D 0001, 0001,1740,2100
D 0002, 0002,1740,2100
D 0003, 0003,1740,2100
D 0004, 0002,1741,2100
D 0005, 0005,1740,2100
D 0006, 0006,1740,2100
D 0007, 0007,1740,2100

D 0010, 0001,1742,2100
D 0011, 0004,1742,2100
D 0012, 0001,1743,2100
D 0013, 0000,1743,2100

D 0014, 0001,1744,2100
D 0015, 0000,1744,2100
D 0016, 0002,1744,2100
D 0017, 0003,1744,2100

D 0020, 0000,1746,2100
D 0021, 0000,1747,2100
D 0022, 0000,1750,2100
D 0023, 0000,1751,2100
D 0024, 0001,1751,2100
D 0025, 0002,1751,2100
D 0026, 0004,1751,2100
D 0027, 0010,1751,2100
```

```
; 5663  .TOC    "EXTEND -- DISPATCH ROM ENTRIES"
; 5664
; 5665          .DCODE
; 5666  001:    I,      SJCL,   J/L-CMS
; 5667          I,      SJCE,   J/L-CMS
; 5668          I,      SJCLE,  J/L-CMS
; 5669          I,      B/2,    J/L-EDIT
; 5670          I,      SJCGE,  J/L-CMS
; 5671          I,      SJCN,   J/L-CMS
; 5672          I,      SJCG,   J/L-CMS
; 5673
; 5674  010:    I,      B/1,    J/L-DBIN        ;CVTDBO
; 5675          I,      B/4,    J/L-DBIN        ;CVTDBT
; 5676          I,      B/1,    J/L-BDEC        ;CVTBDO
; 5677          I,      B/0,    J/L-BDEC        ;CVTBDT
; 5678
; 5679  014:    I,      B/1,    J/L-MVS         ;MOVSO
; 5680          I,      B/0,    J/L-MVS         ;MOVST
; 5681          I,      B/2,    J/L-MVS         ;MOVSLJ
; 5682          I,      B/3,    J/L-MVS         ;MOVSRJ
; 5683
; 5684  020:    I,              J/L-XBLT
; 5685          I,              J/L-SPARE-A
; 5686          I,              J/L-SPARE-B
; 5687          I,      B/0,    J/L-SPARE-C
; 5688          I,      B/1,    J/L-SPARE-C
; 5689          I,      B/2,    J/L-SPARE-C
; 5690          I,      B/4,    J/L-SPARE-C
; 5691          I,      B/10,   J/L-SPARE-C
; 5692          .UCODE
; 5693
```

```
U 1740, 0400,4751,1203,4374,4007,0700,0000,0000,0040
U 1741, 0400,4751,1203,4374,4007,0700,0000,0000,0040
U 1742, 0400,4751,1203,4374,4007,0700,0000,0000,0040
U 1743, 0400,4751,1203,4374,4007,0700,0000,0000,0040
U 1744, 0400,4751,1203,4374,4007,0700,0000,0000,0040
U 1746, 0400,4751,1203,4374,4007,0700,0000,0000,0040
U 1747, 0400,4751,1203,4374,4007,0700,0000,0000,0040
U 1750, 0400,4751,1203,4374,4007,0700,0000,0000,0040
U 1751, 0400,4751,1203,4374,4007,0700,0000,0000,0040
```

```
; 5694  1740:
; 5695  L-CMS:  LUUO
; 5696  1741:
; 5697  L-EDIT: LUUO
; 5698  1742:
; 5699  L-DBIN: LUUO
; 5700  1743:
; 5701  L-BDEC: LUUO
; 5702  1744:
; 5703  L-MVS:  LUUO
; 5704  1746:
; 5705  L-XBLT: LUUO
; 5706  1747:
; 5707  L-SPARE-A: LUUO
; 5708  1750:
; 5709  L-SPARE-B: LUUO
; 5710  1751:
; 5711  L-SPARE-C: LUUO
; 5712
; 5713  ;NOTE: WE DO NOT NEED TO RESERVE 3746 TO 3751 BECAUSE THE CODE
; 5714  ;       AT EXTEND DOES A RANGE CHECK.
; 5715
```

```
D 0123, 0000,1467,3100

U 1467, 1062,3771,0005,4365,5007,0700,0200,0000,0002
U 1062, 3556,4553,0500,4374,4007,0321,0010,0076,0740
U 1066, 3260,4521,0206,4374,4007,0700,0000,0000,0740
U 3260, 3261,3111,0605,4174,0417,0700,0000,0000,0000
U 3261, 3262,3333,0005,4174,4217,0700,0000,0000,0500
U 3262, 3263,3333,0003,7174,4007,0700,0400,0000,0240
U 3263, 0170,4443,0000,2174,4006,6700,0000,0000,0000
U 0170, 0172,0551,0505,2270,4007,0700,0000,0000,0000
U 0172, 0556,5741,0505,4174,4003,7700,0200,0000,0010
U 0174, 3264,0551,0505,2270,4007,0700,0200,0004,0512
U 0176, 3264,3443,0500,4174,4007,0700,0200,0004,0512
U 3264, 3263,3771,0005,4361,5217,0700,0200,0000,0502

U 0556, 3400,3333,0005,7174,4001,2700,0400,0000,0241
U 0557, 2060,3333,0005,4174,4007,0530,0000,0000,0000
U 2060, 3400,3333,0005,7174,4001,2700,0400,0000,0241
U 2061, 0556,3771,0005,4374,0007,0700,0000,0077,7777
```

```
; 5716  .TOC    "EXTEND -- INSTRUCTION SET DECODING"
; 5717
; 5718  ;EACH INSTRUCTION IN THE RANGE 1-23 GOES TO 1 OF 2 PLACES
; 5719  ; 1740-1747 IF NOT UNDER EXTEND
; 5720  ; 3740-3747 IF UNDER EXTEND
; 5721
; 5722          .DCODE
; 5723  123:    I,READ/1,                J/EXTEND
; 5724          .UCODE
; 5725
; 5726  1467:
; 5727  EXTEND: MEM READ, [BR]_MEM        ;FETCH INSTRUCTION
; 5728  =0**    TL [BR], #/760740,       ;IN RANGE 0-17 (AND AC#=0)
; 5729          CALL [BITCHK]            ;TRAP IF NON-ZERO BITS FOUND
; 5730          [BRX]_[HR].AND.# CLR RH, ;SPLIT OUT AC NUMBER
; 5731          #/000740                 ; FROM EXTEND INSTRUCTION
; 5732          [BR]_[BR].OR.[BRX],      ;LOAD IR AND AC #
; 5733          HOLD RIGHT, LOAD IR      ; ..
; 5734          READ [BR], LOAD BYTE EA,         ;LOAD XR #
; 5735              J/EXTEA0                     ;COMPUTE E1
; 5736
; 5737  EXTEA0: WORK[E0]_[AR]
; 5738  EXTEA1: EA MODE DISP
; 5739  =100*
; 5740  EXTEA:  [BR]_[BR]+XR
; 5741  EXTDSP: [BR]_EA FROM [BR], LOAD VMA,
; 5742          B DISP, J/EXTEXT
; 5743          [BR]_[BR]+XR, START READ, PXCT EXTEND EA, LOAD VMA, J/EXTIND
; 5744          VMA_[BR], START READ, PXCT EXTEND EA
; 5745
; 5746  EXTIND: MEM READ, [BR]_MEM, HOLD LEFT, LOAD BYTE EA, J/EXTEA1
; 5747
; 5748  ;HERE TO EXTEND SIGN FOR OFFSET MODES
; 5749  =1110
; 5750  EXTEXT: WORK[E1]_[BR],                    ;SAVE E1
; 5751          DISP/DROM, J/3400                ;GO TO EXTENDED EXECUTE CODE
; 5752          READ [BR], SKIP DP18             ;NEED TO EXTEND SIGN
; 5753  =0      WORK[E1]_[BR],                   ;POSITIVE
; 5754          DISP/DROM, J/3400
; 5755          [BR]_#, #/777777, HOLD RIGHT,    ;NEGATIVE
; 5756          J/EXTEXT
; 5757
```

```
U 3744, 3457,0111,0703,4174,4007,0700,0210,0004,0012
U 3754, 2062,3771,0005,1276,6007,0701,0000,0000,1443

U 2062, 3556,4553,0500,4374,4007,0321,0010,0077,7000
U 2066, 2064,3771,0003,0276,6007,0700,0000,0000,0000

U 2064, 3461,4521,0306,4374,4007,0700,0010,0077,7000

U 2065, 2070,2113,0305,1174,4007,0521,4400,0000,1443

U 2070, 3265,7441,0503,4174,4007,0700,0000,0000,0000
U 2071, 3265,7441,0303,4174,4007,0700,0000,0000,0000

U 3265, 0574,3333,0003,7174,4003,7700,0400,0000,0242

U 0574, 0500,3771,0013,4370,4007,0700,0000,0000,0003
U 0575, 3266,3771,0005,1276,6007,0701,0000,0000,1444

U 0576, 3306,3441,0304,4174,4007,0700,0000,0000,0000

U 0577, 0630,3771,0004,1276,6007,0522,0000,0000,1443

U 3266, 3267,3333,0005,4174,4007,0700,1000,0041,6020
U 3267, 2074,4222,0000,4174,4006,7701,1000,0041,1770

U 2074, 2074,4224,0003,4174,4026,7701,1000,0041,1770
U 2076, 3270,7221,0003,4174,4007,0700,0000,0000,0000

U 3270, 0507,4113,0312,7174,4007,0700,0400,0000,0243
```


EXTEND -- MOVE STRING -- SETUP

```
; 5758  .TOC    "EXTEND -- MOVE STRING -- SETUP"
; 5759
; 5760  ;HERE TO MOVE A STRING
; 5761  ;COME HERE WITH:
; 5762  ;       AR/ E0
; 5763  ;       BR/ E1
; 5764  ;
; 5765  3744:
; 5766  MVS:    [AR]_[AR]+1,            ;GO FETCH FILL
; 5767          LOAD VMA,               ; BYTE
; 5768          START READ,             ; ..
; 5769          CALL [GTFILL]           ;SUBROUTINE TO COMPLETE
; 5770  3754:   [BR]_AC[DLEN]           ;GET DEST LENGTH AND FLAGS
; 5771  =0**    TL [BR], #/777000,      ;ANY FLAGS SET?
; 5772          CALL [BITCHK]           ;SEE IF ILLEGAL
; 5773          [AR]_AC                 ;GET SRC LENGTH AND FLAGS
; 5774  =0      [BRX]_[AR].AND.# CLR RH, ;COPY FLAGS TO BRX
; 5775          #/777000,               ; ..
; 5776          CALL [CLRFLG]           ;CLEAR FLAGS IN AR
; 5777                                  ;NEW DLEN IS <SRC LEN>-<DST LEN>
; 5778          AC[DLEN]_[AR]-[BR], 3T, ;COMPUTE DIFFERENCE
; 5779          SKIP DPO                ;WHICH IS SHORTER?
; 5780  =0      [AR]_.NOT.[BR],         ;DESTINATION
; 5781          J/MVS1                  ;GET NEGATIVE LENGTH
; 5782          [AR]_.NOT.[AR]          ;SOURCE
; 5783  MVS1:   WORK[SLEN]_[AR],        ; ..
; 5784          B DISP                  ;SEE WHAT TYPE OF MOVE
; 5785  ;SLEN NOW HAS -<LEN OF SHORTER STRING>-1
; 5786  =1100
; 5787          STATE_[SRC], J/MOVELP   ;TRANSLATE--ALL SET
; 5788          [BR]_AC[DSTP], J/MVS0   ;OFFSET BUILD MASK
; 5789          [ARX]_[AR],             ;LEFT JUSTIFY
; 5790          J/MOVSTO                ; ..
; 5791          [ARX]_AC[DLEN],         ;RIGHT JUSTIFY
; 5792          SKIP DPO, 4T,           ;WHICH IS SHORTER?
; 5793          J/MOVRJ
; 5794
; 5795  MVS0:   READ [BR], FE_S+2       ;GET DST BYTE SIZE
; 5796          Q_0, BYTE STEP          ;BUILD AN S BIT MASK
; 5797  =0*
; 5798  MVS01:  GEN MSK [AR], BYTE STEP, J/MVS01
; 5799          [AR]_.NOT.Q             ;BITS WHICH MUST NOT BE SET
; 5800          WORK[MSK]_[AR].AND.[MASK], ;SAVE FOR SRCMOD
; 5801          J/MOVLP0                ;GO ENTER LOOP
; 5802
```

```
U 0500, 1074,0551,0703,7274,4007,0701,0010,0000,0242
U 0501, 0750,3441,0304,4174,4007,0520,0000,0000,0000
U 0504, 3271,1771,0003,7274,4007,0701,4000,0000,0242
U 0505, 3451,3771,0013,4370,4007,0700,0010,0000,0005
U 0507, 0500,3771,0013,4370,4007,0700,0000,0000,0003

U 3271, 2072,3771,0005,1276,6007,0522,0000,0000,1443
U 2072, 3272,3440,0303,1174,4007,0700,0400,0000,1443
U 2073, 2072,1111,0503,4174,4007,0700,4000,0000,0000
U 3272, 3273,7771,0003,7274,4007,0701,0000,0000,0242
U 3273, 2100,3333,0005,4174,4007,0520,0000,0000,0000
U 2100, 2101,0111,0503,4174,4007,0700,0000,0000,0000
U 2101, 3274,3111,0603,4174,4007,0700,0000,0000,0000
U 3274, 1515,4221,0013,4170,4007,0700,0000,0000,0000
```

```
; 5803  .TOC    "EXTEND -- MOVE STRING -- OFFSET/TRANSLATE"
; 5804
; 5805  ;HERE IS THE LOOP FOR OFFSET AND TRANSLATED MOVES
; 5806  =000
; 5807  MOVELP: [AR]_WORK[SLEN]+1,       ;UPDATE STRING LENGTH
; 5808          CALL [SRCMOD]            ;GET A SOURCE BYTE
; 5809  =001    [ARX]_[AR], SKIP DPO,    ;(1) LENGTH EXHAUSTED
; 5810          J/MOVST2                 ;    SEE IF FILL IS NEEDED
; 5811  =100    [AR]_-WORK[SLEN],        ;(4) ABORT
; 5812          J/MVABT                  ; ..
; 5813          STATE_[SRC+DST],         ;(5) NORMAL--STORE DST BYTE
; 5814          CALL [PUTDST]            ;     ..
; 5815  =111
; 5816  MOVLPO: STATE_[SRC], J/MOVELP    ;(7) DPB DONE
; 5817  =
; 5818
; 5819  ;HERE TO ABORT A STRING MOVE DUE TO TRANSLATE OR OFFSET FAILURE
; 5820
; 5821  MVABT:  [BR]_AC[DLEN],           ;WHICH STRING IS LONGER
; 5822          SKIP DPO, 4T
; 5823  =0
; 5824  MVABT1: AC[DLEN]_[AR], J/MVABT2 ;PUT AWAY DEST LEN
; 5825          [AR]_[AR]-[BR],          ;DEST LEN WAS GREATER
; 5826          J/MVABT1                 ;STICK BACK IN AC
; 5827
; 5828  MVABT2: [AR]_.NOT.WORK[SLEN]     ;GET UNDECREMENTED SLEN
; 5829          READ [BR], SKIP DPO      ;NEED TO FIXUP SRC?
; 5830  =0      [AR]_[AR]+[BR]           ;SRC LONGER BY (DLEN)
; 5831  MVEND:  [AR]_[AR].OR.[BRX]       ;PUT BACK SRC FLAGS
; 5832          END STATE, J/STAC        ;ALL DONE
; 5833
```

```
U 0630, 3275,3771,0003,1276,6007,0701,0000,0000,1441

U 0631, 2265,3771,0013,4370,4007,0700,0010,0000,0006

U 0633, 3307,0551,0704,7274,4007,0701,0000,0000,0242


U 3275, 2102,3440,0303,1174,4007,0670,0400,0000,1441
U 2102, 3300,3333,0003,7174,4007,0700,0400,0000,0211
rrupt.

U 2103, 2104,1111,0704,4174,4007,0521,4000,0000,0000


U 2104, 0650,3770,0305,4334,4016,7371,0000,0033,6000

U 2105, 0546,4223,0000,1174,4007,0700,0400,0000,1443


U 0650, 3275,3441,0503,4174,4007,0700,0000,0000,0000

U 0651, 3276,7771,0003,7274,4007,0701,0000,0000,0242

U 0652, 3277,3770,0503,4334,4017,0700,0000,0032,6000
U 0653, 3276,7771,0003,7274,4007,0701,0000,0000,0242
U 3276, 2106,3440,0303,1174,4007,0700,0400,0000,1443

U 2106, 3665,0111,0403,4174,4007,0700,0010,0000,0000

U 2107, 3721,3113,0306,0174,4007,0700,0400,0000,0000


U 3277, 3275,0111,0703,4170,4007,0700,0000,0000,0000


U 3300, 3301,3333,0005,7174,4007,0700,0400,0000,0213
U 3301, 3302,3333,0004,7174,4007,0700,0400,0000,0212
U 3302, 2075,3333,0006,7174,4007,0700,0400,0000,0214
U 2075, 3561,4443,0000,4174,4007,0700,0010,0000,0000
U 2077, 3303,3771,0003,7274,4007,0701,0000,0000,0211
U 3303, 3304,3771,0005,7274,4007,0701,0000,0000,0213
U 3304, 3305,3771,0004,7274,4007,0701,0000,0000,0212

U 3305, 3275,3771,0006,7274,4007,0701,0000,0000,0214
```

```
; 5834  .TOC    "EXTEND -- MOVE STRING -- MOVSRJ"
; 5835
; 5836  =OO
; 5837  MOVRJ:  [AR]_AC[SRCP], J/MVSKP  ;SRC LONGER, SKIP OVER SOME
; 5838          STATE_[DSTF],           ;DST LONGER, FILL IT
; 5839          CALL [MOVFIL]           ; ..
; 5840  =11     [ARX]_WORK[SLEN]+1,     ;DONE FILLING
; 5841          J/MOVST1                ;GO MOVE STRING
; 5842
; 5843  ;HERE TO SKIP OVER EXTRA SOURCE BYTES
; 5844  MVSKP:  AC[SRCP]_[AR], SKIP -1MS ;[121] Is there a timer interrupt?
; 5845  =O      WORK[SV.AR]_[AR], J/MVSK2 ;[121][123] Yes, save regs for int
; 5846          [ARX]_[ARX]-1, 3T,      ;DONE SKIPPING?
; 5847          SKIP DPO
; 5848  =O      IBP DP, IBP SCAD,       ;NO--START THE IBP
; 5849          SCAD DISP, SKIP IRPT,   ;4-WAY DISPATCH
; 5850          3T, J/MVSKP1            ;GO BUMP POINTER
; 5851          AC[DLEN]_O,             ;LENGTHS ARE NOW EQUAL
; 5852          J/MOVST4                ;GO MOVE STRING
; 5853
; 5854  =OO
; 5855  MVSKP1: [AR]_[BR], J/MVSKP      ;NO OVERFLOW
; 5856          [AR]_.NOT.WORK[SLEN],   ;INTERRUPT
; 5857          J/MVSK3                 ; ..
; 5858          SET P TO 36-S,          ;WORD OVERFLOW
; 5859          J/MVSKP2                ;FIXUP Y
; 5860          [AR]_.NOT.WORK[SLEN]    ;[121] INTERRUPT or timer.
; 5861  MVSK3:  AC[DLEN]_[AR]            ;RESET DLEN
; 5862  =O      [AR]_[AR]+[ARX],
; 5863          CALL [INCAR]            ;ADD 1 TO AR
; 5864          AC_[AR].OR.[BRX],       ;PUT BACK FLAGS
; 5865          J/ITRAP                 ;DO INTERRUPT TRAP
; 5866
; 5867  MVSKP2: [AR]_[AR]+1, HOLD LEFT, ;BUMP Y
; 5868          J/MVSKP          ;KEEP GOING
; 5869
; 5870                                  ;BEGIN EDIT [123]
; 5871  MVSK2:  WORK[SV.BR]_[BR]         ;SAVE ALL
; 5872          WORK[SV.ARX]_[ARX]      ;THE REGISTERS
; 5873          WORK[SV.BRX]_[BRX]      ;FOR THE TICK
; 5874  =O*     CALL [TICK]             ;UPDATE CLOCK AND SET INTERUPT
; 5875          [AR]_WORK[SV.AR]        ;NOW PUT
; 5876          [BR]_WORK[SV.BR]        ;THEM ALL
; 5877          [ARX]_WORK[SV.ARX]      ;BACK SO WE
; 5878          [BRX]_WORK[SV.BRX],     ;CAN CONTINUE
; 5879                  J/MVSKP
; 5880                                  ;END EDIT [123]
; 5881
```

```
U 3306, 3307,0111,0704,4174,4007,0700,0000,0000,0000
U 3307, 0540,3771,0013,4370,4007,0700,0000,0000,0003


U 0540, 2276,3333,0004,7174,4007,0520,0410,0000,0242

U 0541, 0750,3441,0304,4174,4007,0520,0000,0000,0000

U 0542, 3451,3771,0013,4370,4007,0700,0010,0000,0005


U 0546, 3307,0551,0704,7274,4007,0701,0000,0000,0242


U 0750, 3310,4223,0000,1174,4007,0700,0400,0000,1443
U 0751, 2265,3771,0013,4370,4007,0700,0010,0000,0004
U 0753, 2151,3440,0606,0174,4007,0700,0400,0000,0000

U 3310, 3311,3113,0406,0174,4007,0700,0400,0000,0000
U 3311, 0252,4221,0013,4170,4007,0700,0000,0000,0000
```

```
; 5882  .TOC    "EXTEND -- MOVE STRING -- SIMPLE MOVE LOOP"
; 5883
; 5884  ;HERE FOR NO-MODIFICATION STRING MOVES
; 5885  MOVSTO: [ARX]_[ARX]+1            ;CANT DO [ARX]_[AR]+1
; 5886  MOVST1: STATE_[SRC]              ;PREPARE FOR PAGE FAIL
; 5887  =000
; 5888         WORK[SLEN]_[ARX],         ;GO GET A SOURCE BYTE
; 5889         SKIP DPO, CALL [GSRC]     ; ..
; 5890  MOVSTX: [ARX]_[AR],              ;SHORT STRING RAN OUT
; 5891         SKIP DPO, J/MOVST2        ;GO SEE IF FILL NEEDED
; 5892  =010   STATE_[SRC+DST],          ;WILL NEED TO BACK UP BOTH POINTERS
; 5893         CALL [PUTDST]             ;STORE BYTE
; 5894  =110
; 5895  MOVST4: [ARX]_WORK[SLEN]+1,      ;COUNT DOWN LENGTH
; 5896         J/MOVST1                  ;LOOP OVER STRING
; 5897  =
; 5898  =00
; 5899  MOVST2: AC[DLEN]_0, J/MOVST3     ;CLEAR DEST LEN, REBUILD SRC
; 5900         STATE_[DST], CALL [MOVFIL] ;FILL OUT DEST
; 5901  =11    AC_[BRX], J/ENDSKP        ;ALL DONE
; 5902
; 5903  MOVST3: AC_[ARX].OR.[BRX]        ;REBUILD SRC
; 5904         END STATE, J/SKIPE        ; ..
; 5905
```

```
U 3740, 2063,3771,0004,1276,6007,0701,0000,0000,1443
U 2063, 3556,4553,0400,4374,4007,0321,0010,0077,7000
U 2067, 2110,3771,0006,0276,6007,0700,0000,0000,0000
U 2110, 3556,4553,0600,4374,4007,0321,0010,0077,7000
U 2114, 2112,2113,0604,4174,4007,0521,4000,0000,0000
U 2112, 2113,0111,0703,4174,4007,0700,0000,0000,0000
U 2113, 2116,0111,0703,4170,4007,0700,0200,0004,0012


U 2116, 3663,4221,0003,4174,4007,0700,0010,0000,0000

U 2117, 3321,3223,0000,7174,4007,0700,0400,0000,0244
```

```
; 5906  .TOC    "EXTEND -- COMPARE STRING"
; 5907
; 5908  3740:
; 5909  CMS:    [ARX]_AC[DLEN]          ;GET DEST LEN
; 5910  =0**    TL [ARX], #/777000, CALL [BITCHK]
; 5911          [BRX]_AC                ;GET SRC LEN
; 5912  =0**    TL [BRX], #/777000, CALL [BITCHK]
; 5913          [BRX]-[ARX], 3T, SKIP DPO ;WHICH STRING IS LONGER?
; 5914  =0      [AR]_[AR]+1             ;SRC STRING IS LONGER
; 5915          VMA_[AR]+1, START READ  ;DST STRING
; 5916  =0      [AR]_0,                 ;FORCE FIRST COMPARE TO BE
; 5917                                  ;EQUAL
; 5918          CALL [LOADQ]            ;PUT FILL INTO Q
; 5919          WORK[FILL]_Q,           ;SAVE FILLER
; 5920          J/CMS2                  ;ENTER LOOP
; 5921
```

```
U 2120, 0250,4221,0013,4170,4003,7700,0000,0000,0000
U 2121, 3312,3771,0003,1276,6007,0701,0000,0000,1441
U 3312, 1020,3333,0006,4174,4007,0520,0000,0000,0000

U 1020, 2277,3771,0013,4370,4007,0700,0010,0000,0011

U 1021, 2122,3333,0004,4174,4007,0520,0000,0000,0000
U 1022, 3313,3333,0003,7174,4007,0700,0400,0000,0245

U 3313, 3314,3440,0606,0174,4007,0700,0400,0000,0000
U 3314, 3315,3771,0013,4370,4007,0700,0000,0000,0010
U 3315, 1030,3333,0004,4174,4007,0520,0000,0000,0000

U 1030, 2124,4443,0000,4174,4007,0700,0010,0000,0000

U 1031, 3316,3771,0003,7274,4007,0701,0000,0000,0244
U 1032, 3316,3440,0404,1174,4007,0700,0400,0000,1443


U 3316, 3317,4111,1203,7174,4007,0700,0000,0000,0245
U 3317, 3320,4551,1205,7274,4007,0700,0000,0000,0245
U 3320, 3321,2111,0503,4174,4007,0700,4000,0000,0000
U 3321, 3322,1111,0704,4174,4007,0700,4000,0000,0000
U 3322, 3323,1111,0706,4174,4007,0700,4000,0000,0000
U 3323, 2120,3333,0003,4174,4007,0621,0000,0000,0000


U 2122, 3324,3772,0000,7274,4007,0701,0000,0000,0244
U 2123, 2120,4221,0003,4174,4007,0700,0000,0000,0000
U 3324, 3325,3771,0013,4370,4007,0700,0000,0000,0012
U 3325, 1030,3223,0000,7174,4007,0700,0400,0000,0245



U 2124, 3452,3771,0003,1276,6007,0701,0010,0000,1444


U 2125, 0340,3333,0003,4174,4006,5701,1000,0051,0770
```

```
; 5922  ;HERE IS THE COMPARE LOOP.
; 5923  ; ARX/ CONATINS REMAINING DEST LENGTH
; 5924  ; BRX/ CONTAINS REMAINING SOURCE LENGTH
; 5925  =O
; 5926  CMS3:                                ;BYTES ARE NOT EQUAL
; 5927          END STATE,                   ;NO MORE SPECIAL PAGE FAIL ACTION
; 5928          SKIP-COMP DISP               ;SEE SKIP-COMP-TABLE
; 5929  CMS4:   [AR]_AC[SRCP]                ;GET BYTE POINTER
; 5930          READ [BRX], SKIP DPO         ;MORE IN SOURCE STRING?
; 5931  =OO      STATE_[EDIT-SRC],            ;PREPARE FOR PAGE FAIL
; 5932          CALL [GETSRC]                ; GO GET BYTE
; 5933          READ [ARX], SKIP DPO,        ;NO MORE SRC--SEE IF MORE DEST
; 5934          J/CMS5                       ; ..
; 5935          WORK[CMS]_[AR]               ;SAVE SRC BYTE
; 5936  =
; 5937          AC_[BRX]                     ;PUT BACK SRC LEN
; 5938          STATE_[COMP-DST]             ;HAVE TO BACK UP IF DST FAILS
; 5939          READ [ARX], SKIP DPO         ;ANY MORE DEST?
; 5940  =OO
; 5941  CMS6:   CALL [CMPDST]                ;MORE DEST BYTES
; 5942          [AR]_WORK[FILL],             ;OUT OF DEST BYTES
; 5943          J/CMS7                       ;GO DO COMPARE
; 5944          AC[DLEN]_[ARX]               ;GOT A BYTE, UPDATE LENGTH
; 5945  =
; 5946  CMS7:   [AR]_[AR].AND.[MASK],        ;MAKE MAGNITUDES
; 5947          WORK[CMS]                    ;WARM UP RAM
; 5948          [BR]_[MASK].AND.WORK[CMS], 2T ;GET SRC MAGNITUDE
; 5949          [AR]_[BR]-[AR] REV           ;UNSIGNED COMPARE
; 5950  CMS2:   [ARX]_[ARX]-1                 ;UPDATE LENGTHS
; 5951          [BRX]_[BRX]-1                ; ..
; 5952          READ [AR], SKIP AD.EQ.O, J/CMS3 ;SEE IF EQUAL
; 5953
; 5954  =O
; 5955  CMS5:   Q_WORK[FILL], J/CMS8          ;MORE DST--GET SRC FILL
; 5956          [AR]_O, J/CMS3               ;STRINGS ARE EQUAL
; 5957  CMS8:   STATE_[EDIT-DST]             ;JUST DST POINTER ON PAGE FAIL
; 5958          WORK[CMS]_Q, J/CMS6          ;MORE DST--SAVE SRC FILL
; 5959
; 5960  =O
; 5961  CMPDST: [AR]_AC[DSTP],               ;GET DEST POINTER
; 5962          CALL [IDST]                  ;UPDATE IT
; 5963          READ [AR],                   ;LOOK AT BYTE POINTER
; 5964          FE_FE.AND.S#, S#/0770,       ;MASK OUT BIT 6
; 5965          BYTE DISP, J/LDB1            ;GO LOAD BYTE
; 5966
```

```
U 3742, 3326,4571,1203,4374,4007,0700,0000,0077,7777
U 3326, 3327,3333,0003,7174,4007,0700,0400,0000,0243

U 3327, 3330,3771,0003,0276,6007,0700,0000,0000,0000

U 3330, 2130,4521,0306,4374,4007,0700,0000,0077,7000

U 2130, 2222,3771,0004,1276,6007,0701,0010,0000,1444
U 2132, 2126,3440,0404,1174,4007,0700,0400,0000,1444

U 2126, 2156,3333,0006,4174,4007,0520,0010,0000,0000

U 2127, 0616,4551,0303,4374,0003,7700,0000,0000,0777

U 0616, 3331,3771,0013,4370,4007,0700,0000,0000,0007

U 0617, 0616,3551,0606,4374,0007,0700,0000,0040,0000
U 3331, 0460,7333,0003,7174,4007,0700,0400,0000,0242

U 0460, 1074,0551,0703,7274,4007,0701,0010,0000,0242

U 0461, 2144,4553,0600,4374,4007,0321,0000,0010,0000

U 0464, 3336,7771,0003,7274,4007,0701,0000,0000,0242

U 0465, 2134,1553,0300,4374,4007,0532,4000,0000,0012

U 2134, 3336,7771,0003,7274,4007,0701,0000,0000,0242
```

```
; 5967  .TOC    "EXTEND -- DECIMAL TO BINARY CONVERSION"
; 5968
; 5969  3742:
; 5970  DBIN:   [AR]_[777777] XWD 0      ;IF WE ARE IN OFFSET MODE
; 5971          WORK[MSK]_[AR]           ; ONLY ALLOW 18 BITS
; 5972                                   ;RANGE CHECKED (0-10) LATER
; 5973          [AR]_AC                  ;GET SRC LENGTH
; 5974          [BRX]_[AR].AND.# CLR RH, ;SPLIT OUT FLAGS
; 5975          #/777000                 ; ..
; 5976  =0*     [ARX]_AC[BIN1],          ;GET LOW WORD
; 5977          CALL [CLARX0]            ;CLEAR BIT 0 OF ARX
; 5978          AC[BIN1]_[ARX]           ;STORE BACK
; 5979  =0      READ [BRX], SKIP DPO,    ;IS S ALREADY SET?
; 5980          CALL [CLRBIN]            ;GO CLEAR BIN IF NOT
; 5981          [AR]_[AR].AND.#,         ;CLEAR FLAGS FROM LENGTH
; 5982          #/000777, HOLD RIGHT,    ; ..
; 5983          B DISP                   ;SEE IF OFFSET OR TRANSLATE
; 5984  =1110
; 5985  DBIN1:  STATE_[CVTDB], J/DBIN2   ;TRANSLATE--LEAVE S ALONE
; 5986          [BRX]_[BRX].OR.#,        ;OFFSET--FORCE S TO 1
; 5987          #/400000, HOLD RIGHT,
; 5988          J/DBIN1
; 5989  DBIN2:  WORK[SLEN]_.NOT.[AR]     ;STORE -SLEN-1
; 5990
; 5991  ;HERE IS THE MAIN LOOP
; 5992  =0*0
; 5993  DBINLP: [AR]_WORK[SLEN]+1, CALL [SRCMOD] ;(0) GET MODIFIED SRC BYTE
; 5994          TL [BRX], #/100000,      ;(1) DONE, IS M SET?
; 5995          J/DBXIT
; 5996          [AR]_.NOT.WORK[SLEN],    ;(4) ABORT
; 5997          J/DBABT                  ;       ..
; 5998          [AR]-#, #/10.,           ;(5) NORMAL--SEE IF 0-9
; 5999          4T, SKIP DP18            ; ..
; 6000  =0      [AR]_.NOT.WORK[SLEN],    ;DIGIT TOO BIG
; 6001          J/DBABT                  ;GO ABORT CVT
; 6002
```

```
U 2135,  1050,3771,0005,1276,6007,0622,0000,0000,1443

U 1050,  0560,3771,0004,1276,6007,0701,0010,0000,1444

U 1051,  3332,3771,0005,1276,6007,0701,0000,0000,1444
U 1052,  0460,4443,0000,4174,4007,0700,0000,0000,0000

U 3332,  2136,4553,0500,4374,4007,0321,0000,0076,0000
U 2136,  1050,4443,0000,4174,4007,0700,0000,0000,0000
U 2137,  3333,3775,0005,1276,6007,0701,0000,0000,1444
U 3333,  2140,3445,0505,1174,4007,0700,0000,0000,1444

U 2140,  3666,0551,0505,1274,4007,0700,0010,0000,1444

U 2141,  0460,0113,0305,1174,4007,0701,0400,0000,1444
```

```
; 6003  ;HERE TO ADD IN A DIGIT
; 6004          [BR]_AC[BINO], 4T,      ;GET HIGH BINARY
; 6005          SKIP AD.EQ.O            ;SEE IF SMALL
; 6006  =OO
; 6007  DBSLO:  [ARX]_AC[BIN1],         ;TOO BIG
; 6008          CALL [DBSLOW]           ;GO USE DOUBLE PRECISION PATHS
; 6009          [BR]_AC[BIN1],          ;GET LOW WORD
; 6010          J/DBFAST                ;MIGHT FIT IN 1 WORD
; 6011          J/DBINLP                ;RETURN FROM DBSLOW
; 6012                                  ;GO DO NEXT DIGIT
; 6013  =
; 6014  DBFAST: TL [BR], #/760000       ;WILL RESULT FIT IN 36 BITS?
; 6015  =O      J/DBSLO                 ;MAY NOT FIT--USE DOUBLE WORD
; 6016          [BR]_AC[BIN1]*2         ;COMPUTE AC*2
; 6017          [BR]_[BR]*2, AC[BIN1]   ;COMPUTE AC*4
; 6018  =O      [BR]_[BR]+AC[BIN1], 2T, ;COMPUTE AC*5
; 6019          CALL [SBRL]             ;COMPUTE AC*10
; 6020          AC[BIN1]_[AR]+[BR], 3T, ;NEW BINARY RESULT
; 6021          J/DBINLP                ;DO NEXT DIGIT
; 6022
```

```
U 0560, 0620,3771,0005,1276,6007,0701,0010,0000,1443
U 0561, 2142,0551,0404,1274,4007,0562,0010,0000,1444
U 0565, 0600,0551,0505,1274,4007,0701,0000,0000,1443
U 0600, 0621,4443,0000,4174,4007,0700,0010,0000,0000
U 0601, 2142,0111,0304,4174,4007,0561,0010,0000,0000
U 0605, 3334,3440,0404,1174,4007,0700,0400,0000,1444
U 3334, 0002,3440,0505,1174,4004,1700,0400,0000,1443
U 0620, 0621,4443,0000,4174,4007,0700,0010,0000,0000
U 0621, 0622,0111,0505,4174,4007,0700,0000,0000,0000
U 0622, 2142,0111,0404,4174,4007,0561,0010,0000,0000
U 0626, 0001,4443,0000,4174,4004,1700,0000,0000,0000
U 2142, 0004,4443,0000,4174,4004,1700,0000,0000,0000
U 2143, 3335,4551,0404,4374,0007,0700,0000,0037,7777
U 3335, 0004,0111,0705,4174,4004,1700,0000,0000,0000
```

```
; 6023  ;HERE IF NUMBER DOES NOT FIT IN ONE WORD
; 6024
; 6025  =000
; 6026  DBSLOW: [BR]_AC[BINO],          ;FETCH HIGH WORD
; 6027          CALL [MULBY4]           ;MULTIPLY BY 4
; 6028          [ARX]_[ARX]+AC[BIN1],   ;COMPUTE VALUE * 5
; 6029          SKIP CRY1, 4T,          ;SEE IF OVERFLOW
; 6030          CALL [ADDCRY]           ;GO ADD CARRY
; 6031  =101    [BR]_[BR]+AC[BINO]      ;ADD IN HIGH WORD
; 6032  =
; 6033  =000    CALL [DBLDBL]           ;MAKE * 10
; 6034          [ARX]_[ARX]+[AR], 3T,   ;ADD IN NEW DIGIT
; 6035          SKIP CRY1,              ;SEE IF OVERFLOW
; 6036          CALL [ADDCRY]           ;ADD IN THE CARRY
; 6037  =101    AC[BIN1]_[ARX]          ;PUT BACK ANSWER
; 6038  =
; 6039          AC[BINO]_[BR],          ; ..
; 6040          RETURN [2]              ;GO DO NEXT BYTE
; 6041
; 6042  ;HERE TO DOUBLE BR!ARX
; 6043  =000
; 6044  MULBY4: CALL [DBLDBL]           ;DOUBLE TWICE
; 6045  DBLDBL: [BR]_[BR]+[BR]          ;DOUBLE HIGH WORD FIRST
; 6046                                 ;(SO WE DON'T DOUBLE CARRY)
; 6047          [ARX]_[ARX]+[ARX],      ;DOUBLE LOW WORD
; 6048          SKIP CRY1, 3T,          ;SEE IF CARRY
; 6049          CALL [ADDCRY]           ;ADD IN CARRY
; 6050  =110    RETURN [1]              ;ALL DONE
; 6051  =
; 6052
; 6053  ;HERE TO ADD THE CARRY
; 6054  =0
; 6055  ADDCRY: RETURN [4]              ;NO CARRY
; 6056          CLEAR [ARX]0            ;KEEP LOW WORD POSITIVE
; 6057          [BR]_[BR]+1,            ;ADD CARRY
; 6058          RETURN [4]              ;ALL DONE
; 6059
```

```
U 3336, 3337,3111,0306,4174,4007,0700,0000,0000,0000

U 3337, 2145,1111,0701,4170,4007,0700,4000,0000,0000




U 2144, 3342,3771,0004,1276,6007,0701,0000,0000,1444
U 2145, 3340,3771,0003,1276,6007,0701,0000,0000,1444

U 3340, 2146,3771,0005,1276,6007,0522,0000,0000,1443
U 2146, 3341,4551,0303,4374,0007,0700,0000,0037,7777
U 2147, 3341,3551,0303,4374,0007,0700,0000,0040,0000
U 3341, 2150,3440,0303,1174,4007,0700,0400,0000,1444

U 2150, 2156,3770,0606,0174,4007,0520,0410,0000,0000
U 2151, 0260,4221,0013,4170,4007,0700,0000,0000,0000

U 3342, 3343,4551,0404,4374,0007,0700,0000,0037,7777

U 3343, 2152,2441,0404,1174,4007,0621,4000,0000,1443
U 2152, 2155,7771,0003,1274,4007,0700,0000,0000,1443

U 2153, 2154,1771,0003,1274,4007,0621,4000,0000,1443
U 2154, 2155,4571,1204,4374,4007,0700,0000,0040,0000

U 2155, 3344,3440,0303,1174,4007,0700,0400,0000,1443
U 3344, 2145,3440,0404,1174,4007,0700,0400,0000,1444




U 2156, 3345,4223,0000,1174,4007,0700,0400,0000,1443
U 2157, 0001,4443,0000,4174,4004,1700,0000,0000,0000
U 3345, 0001,4223,0000,1174,4004,1700,0400,0000,1444
```

```
; 6060  ;HERE TO ABORT CONVERSION
; 6061  DBABT:   [BRX]_[BRX].OR.[AR]      ;PUT BACK UNUSED LENGTH
; 6062           [PC]_[PC]-1, HOLD LEFT,  ;DO NOT SKIP
; 6063           J/DBDONE                 ;GO FIX UP SIGN COPY
; 6064
; 6065  ;HERE AT END
; 6066  =O
; 6067  DBXIT:   [ARX]_AC[BIN1],          ;GET LOW WORD
; 6068           J/DBNEG                  ;GO NEGATE
; 6069  DBDONE:  [AR]_AC[BIN1]            ;FETCH LOW WORD
; 6070           [BR]_AC[BINO], 4T,       ;GET HIGH WORD
; 6071           SKIP DPO                 ;WHAT SIGN
; 6072  =O       CLEAR [AR]O, J/DBDN1     ;POSITIVE
; 6073           [AR]_[AR].OR.#, #/400000, HOLD RIGHT
; 6074  DBDN1:   AC[BIN1]_[AR]            ;STORE AC BACK
; 6075  =O       AC_[BRX] TEST,   ;RETURN FLAGS
; 6076           SKIP DPO, CALL [CLRBIN] ;CLEAR BIN IS S=O
; 6077  ENDSKP:  END STATE, J/SKIP         ;NO--ALL DONE
; 6078
; 6079  DBNEG:   CLEAR ARXO               ;CLEAR EXTRA SIGN BIT
; 6080           [ARX]_-[ARX], 3T,        ;NEGATE AND SEE IF
; 6081           SKIP AD.EQ.O, AC[BINO]   ; ANY CARRY
; 6082  =O       [AR]_.NOT.AC[BINO], 2T, J/STAC34 ;NO CARRY
; 6083           [AR]_-AC[BINO], 3T,      ;CARRY
; 6084           SKIP AD.EQ.O             ;SEE IF ALL ZERO
; 6085  =O       [ARX]_[400000] XWD O     ;MAKE COPY OF SIGN
; 6086                                    ; UNLESS HIGH WORD IS ZERO
; 6087  STAC34:  AC[BINO]_[AR]            ;PUT BACK ANSWER
; 6088           AC[BIN1]_[ARX], J/DBDONE        ; ..
; 6089
; 6090  ;HELPER SUBROUTINE TO CLEAR AC[BINO] AND AC[BIN1] IF S=O
; 6091  ;CALL WITH:
; 6092  ;       READ [BRX], SKIP DPO, CALL [CLRBIN]
; 6093  ;RETURNS 1 ALWAYS
; 6094  =O
; 6095  CLRBIN:  AC[BINO]_O, J/CLRB1
; 6096           RETURN [1]
; 6097  CLRB1:   AC[BIN1]_O, RETURN [1]
; 6098
```

```
U 3743, 2160,3771,0006,1276,6007,0351,0000,0000,1443

U 2160, 3346,4551,0606,4374,4007,0700,0000,0077,7000
U 2161, 3364,3771,0003,0276,6007,0700,0000,0000,0000
U 3346, 3347,3771,0004,1276,6007,0701,0000,0000,1441
U 3347, 2131,3771,0003,0276,6007,0700,2000,0071,0024

U 2131, 2222,3333,0004,7174,4007,0700,0410,0000,0250

U 2133, 2162,3333,0003,7174,4007,0520,0400,0000,0247

U 2162, 2172,4221,0006,4170,4007,0700,0000,0000,0000

U 2163, 2164,3551,0606,4374,0007,0700,0000,0010,0000

U 2164, 3072,4551,0404,4374,0007,0700,0010,0037,7777

U 2166, 2170,3770,0303,0174,4007,0520,0400,0000,0000

U 2170, 2162,3440,0404,1174,4007,0700,0400,0000,1441
U 2171, 3350,0111,0704,4174,4007,0700,0000,0000,0000

U 3350, 2164,3551,0606,4374,0007,0700,0000,0004,0000

U 2172, 0441,3771,0003,0276,6007,0700,0000,0000,0000

U 2173, 2176,3551,0606,4374,0007,0700,0000,0020,0000

U 0441, 2216,3771,0004,1276,6007,0701,0010,0000,1441

U 0443, 2172,0111,0706,4174,4007,0630,2000,0060,0000
U 0447, 2174,4553,0600,4374,4007,0331,0000,0077,7777

U 2174, 2176,3551,0606,4374,0007,0700,0000,0020,0000
U 2175, 2176,0111,0706,4174,4007,0700,0000,0000,0000
```

```
; 6099  .TOC    "EXTEND -- BINARY TO DECIMAL CONVERSION"
; 6100
; 6101  3743:
; 6102  BDEC:   [BRX]_AC[DLEN],          ;GET LENGTH AND FLAGS
; 6103          SKIP FPD                 ;CONTINUE FROM INTERUPT?
; 6104  =0      [BRX]_[BRX].AND.#,       ;JUST KEEP THE FLAGS
; 6105          #/777000,                ; ..
; 6106          J/BDECO                  ;COMPUTE NEW FLAGS
; 6107  DOCVT:  [AR]_AC, J/DOCVT1        ;ALL SET PRIOR TO TRAP
; 6108  BDECO:  [ARX]_AC[1]              ;GET LOW BINARY
; 6109          [AR]_AC, SC_20.          ;GET HIGH WORD, SET STEP COUNT
; 6110  =0*     WORK[BDL]_[ARX],         ;SAVE IN CASE OF ABORT
; 6111          CALL [CLARXO]            ;MAKE SURE BIT 0 IS OFF
; 6112          WORK[BDH]_[AR],          ;SAVE HIGH WORD AND
; 6113          SKIP DPO                 ; TEST SIGN
; 6114  =0
; 6115  BDEC1:  [BRX]_0, HOLD LEFT,      ;POSITIVE, CLEAR RH OF BRX
; 6116          J/BDEC3                  ;COMPUTE # OF DIGITS REQUIRED
; 6117          [BRX]_[BRX].OR.#,        ;NEGATIVE, SET M
; 6118          #/100000, HOLD RIGHT     ; ..
; 6119  =0*
; 6120  BDEC2:  CLEAR ARXO, CALL [DBLNG1] ;NEGATE AR!ARX
; 6121          AC_[AR] TEST,            ;PUT BACK ANSWER
; 6122          SKIP DPO                 ;IF STILL MINUS WE HAVE
; 6123                                   ; 1B0, AND NO OTHER BITS
; 6124  =0      AC[1]_[ARX], J/BDEC1     ;POSITIVE NOW
; 6125          [ARX]_[ARX]+1            ;JUST 1B0--ADD 1
; 6126          [BRX]_[BRX].OR.#,        ;AND REMEMBER THAT WE DID
; 6127          #/040000, HOLD RIGHT,    ; IN LEFT HALF OF AC+3
; 6128          J/BDEC2                  ; NEGATE IT AGAIN
; 6129  =0
; 6130  BDEC3:  [AR]_AC, J/BDEC4         ;GET HIGH AC
; 6131          [BRX]_[BRX].OR.#,        ;NO LARGER POWER OF 10 FITS
; 6132          #/200000,                ;SET N FLAG (CLEARLY NOT 0)
; 6133          HOLD RIGHT, J/BDEC5      ;SETUP TO FILL, ETC.
; 6134  =001
; 6135  BDEC4:  [ARX]_AC[1],             ;GET HIGH WORD
; 6136          CALL [BDSUB]             ;SEE IF 10**C(BRX) FITS
; 6137  =011    [BRX]_[BRX]+1,   ;NUMBER FITS--TRY A LARGER ONE
; 6138          STEP SC, J/BDEC3         ;UNLESS WE ARE OUT OF NUMBERS
; 6139  =111    TR [BRX], #/777777       ;ANY DIGITS REQUIRED?
; 6140  =
; 6141  =0      [BRX]_[BRX].OR.#,        ;SOME DIGITS NEEDED,
; 6142          #/200000, HOLD RIGHT,    ; SET N FLAG
; 6143          J/BDEC5                  ;CONTINUE BELOW
; 6144          [BRX]_[BRX]+1            ;ZERO--FORCE AT LEAST 1 DIGIT
; 6145
```

```
U 2176, 3461,3771,0003,1276,6007,0701,0010,0000,1443
U 2177, 3351,4221,0005,4174,4007,0700,0000,0000,0000
U 3351, 3352,3441,0605,4170,4007,0700,0000,0000,0000

U 3352, 2200,1111,0305,4174,4007,0421,4000,0000,0000

U 2200, 3374,3771,0004,7274,4007,0701,0000,0000,0250
U 2201, 2202,3333,0006,4174,4007,0520,0000,0000,0000

U 2202, 2161,3440,0606,1174,4007,0700,0400,0000,1443



U 2203, 3353,3441,0603,4174,0007,0700,0000,0000,0000
U 3353, 3354,3440,0303,1174,4007,0700,0400,0000,1443
U 3354, 3355,3771,0003,7274,4007,0701,0000,0000,0240

U 3355, 3356,0111,0703,4174,4007,0700,0200,0004,0012
U 3356, 3357,3771,0016,4365,5007,0700,0200,0000,0002
U 3357, 3360,3771,0013,4370,4007,0700,0000,0000,0012
U 3360, 3361,2113,0507,7174,4007,0701,4400,0000,0242
U 3361, 3362,3441,1603,7174,4007,0700,0000,0000,0242



U 3362, 0640,0551,0705,7274,4007,0521,0000,0000,0242
U 0640, 2161,3440,0606,1174,4007,0700,0400,0000,1443

U 0641, 3451,3333,0005,7174,4007,0700,0410,0000,0242
U 0647, 3363,2551,0705,1274,4007,0701,4000,0000,1443

U 3363, 3361,3440,0505,1174,4007,0700,0400,0000,1443
```

```
; 6146  =O
; 6147  BDEC5:  [AR]_AC[DLEN],          ;GET LENGTH
; 6148          CALL [CLRFLG]           ;REMOVE FLAGS FROM AR
; 6149          [BR]_O
; 6150          [BR]_[BRX], HOLD LEFT   ;GET # OF DIGITS NEEDED
; 6151          [BR]_[BR]-[AR],         ;NUMBER OF FILLS NEEDED
; 6152          SKIP AD.LE.O            ;SEE IF ENOUGH ROOM
; 6153  =O      [ARX]_WORK[BDL],        ;DOES NOT FIT IN SPACE ALLOWED
; 6154          J/BDABT                 ; DO NOT DO CONVERT
; 6155          READ [BRX], SKIP DPO    ;IS L ALREADY SET
; 6156  =O      AC[DLEN]_[BRX],         ;NO--NO FILLERS
; 6157          J/DOCVT                 ;GO CHURN OUT THE NUMBER
; 6158
; 6159
; 6160  ;HERE TO STORE LEADING FILLERS
; 6161          [AR]_[BRX], HOLD RIGHT  ;MAKE SURE THE FLAGS GET SET
; 6162          AC[DLEN]_[AR]           ; BEFORE WE PAGE FAIL
; 6163          [AR]_WORK[EO]           ;ADDRESS OF FILL (-1)
; 6164          [AR]_[AR]+1, LOAD VMA,  ;FETCH FILLER
; 6165          START READ
; 6166          MEM READ, [TO]_MEM      ;GET FILLER INTO AR
; 6167          STATE_[EDIT-DST]        ;PAGE FAILS BACKUP DST
; 6168          WORK[SLEN]_[BR]-1, 3T   ;SAVE # OF FILLERS
; 6169  BDFILL: [AR]_[TO], WORK[SLEN]   ;RESTORE FILL BYTE AND
; 6170                                  ; WARM UP RAM FILE
; 6171          [BR]_WORK[SLEN]+1, 3T,  ;MORE FILLERS NEEDED?
; 6172          SKIP DPO
; 6173  =000    AC[DLEN]_[BRX], J/DOCVT ;ALL DONE FIX FLAGS AND CONVERT
; 6174  =001    WORK[SLEN]_[BR],        ;SAVE UPDATED LENGTH
; 6175          CALL [PUTDST]           ; AND STORE FILLER
; 6176  =111    [BR]_AC[DLEN]-1         ;COUNT DOWN STRING LENGTH
; 6177  =
; 6178          AC[DLEN]_[BR], J/BDFILL ;KEEP FILLING
; 6179
```

```
U 3364, 3373,3771,0004,1276,6007,0701,0000,0000,1441

U 0562, 2216,0111,0705,4174,4007,0700,0010,0000,0000
U 0566, 3365,3333,0003,7174,4007,0700,0400,0000,0247

U 3365, 0636,0551,0503,7274,4003,7701,0000,0000,0241

U 0636, 2210,3333,0003,4174,4007,0700,0200,0004,0012
U 0637, 0510,3333,0004,7174,4007,0700,0400,0000,0250
U 0510, 3451,3771,0013,4370,4007,0700,0010,0000,0012
U 0516, 3366,2551,0705,1274,4007,0701,4000,0000,1443
U 3366, 3367,3771,0003,7274,4007,0701,0000,0000,0247
U 3367, 3370,3771,0004,7274,4007,0701,0000,0000,0250
U 3370, 2204,4553,0500,4374,4007,0321,0000,0004,0000
U 2204, 3375,0111,0704,4174,4007,0700,0000,0000,0000
U 2205, 3371,3440,0303,0174,4007,0700,0400,0000,0000
U 3371, 3372,3440,0404,1174,4007,0700,0400,0000,1441
U 3372, 3373,3440,0505,1174,4007,0700,0400,0000,1443
U 3373, 2206,1111,0706,4174,4007,0531,4000,0000,0000
U 2206, 0562,2441,0705,4174,4467,0701,4000,0003,0000
U 2207, 0260,4221,0013,4170,4467,0700,0000,0005,0000

U 2210, 3661,4221,0013,4170,4007,0700,0010,0000,0000
U 2211, 2212,4553,0600,4374,4007,0331,0000,0077,7777
U 2212, 0637,4221,0003,4174,0007,0700,0000,0000,0000
U 2213, 2214,4553,0600,4374,4007,0321,0000,0010,0000
U 2214, 2215,3770,0303,4344,4007,0700,0000,0000,0000
U 2215, 0637,4221,0003,4174,0007,0700,0000,0000,0000

U 3374, 1505,3771,0003,7274,4007,0701,0000,0000,0247

U 3375, 2205,5551,0505,4374,0007,0700,0000,0004,0000
```

```
; 6180  ;HERE TO STORE THE ANSWER
; 6181
; 6182  DOCVT1: [ARX]_AC[1],              ;GET LOW WORD
; 6183          J/DOCVT2                 ;ENTER LOOP FROM BOTTOM
; 6184  =010
; 6185  BDECLP: [BR]_[BR]+1,              ;COUNT DIGITS
; 6186          CALL [BDSUB]             ;KEEP SUBTRACTING 10**C(BRX)
; 6187  =110    WORK[BDH]_[AR]           ;SAVE BINARY
; 6188  =
; 6189          [AR]_[BR]+WORK[E1],      ;OFFSET DIGIT
; 6190          B DISP                   ;SEE WHICH MODE
; 6191  =1110   READ [AR], LOAD VMA,     ;TRANSLATE, START READING TABLE
; 6192          START READ, J/BDTBL      ; GO GET ENTRY FROM TABLE
; 6193  BDSET:  WORK[BDL]_[ARX]          ;SAVE LOW BINARY
; 6194  =00*    STATE_[EDIT-DST], CALL [PUTDST]
; 6195  =11*    [BR]_AC[DLEN]-1          ;UPDATE STRING LENGTH
; 6196          [AR]_WORK[BDH]
; 6197          [ARX]_WORK[BDL]
; 6198          TL [BR], #/040000        ;ARE WE CONVERTING 1B0?
; 6199  =0      [ARX]_[ARX]+1, J/BDCFLG  ;YES--FIX THE NUMBER AND CLEAR FLAG
; 6200  DOCVT3: AC_[AR]
; 6201          AC[1]_[ARX]
; 6202          AC[DLEN]_[BR]            ;STORE BACK NEW STRING LENGTH
; 6203  DOCVT2: [BRX]_[BRX]-1, 3T, SKIP DP18
; 6204  =0      [BR]_-1, SET FPD, 3T, J/BDECLP
; 6205          END STATE, CLR FPD, J/SKIP
; 6206
; 6207  ;HERE TO TRANSLATE 1 DIGIT
; 6208  =0
; 6209  BDTBL:  END STATE,                ;DON'T CHANGE BYTE POINTER IF
; 6210                                    ; THIS PAGE FAILS
; 6211          CALL [LOADAR]            ;GO PUT WORD IN AR
; 6212          TR [BRX], #/777777       ;LAST DIGIT
; 6213  =0      [AR]_0, HOLD RIGHT, J/BDSET
; 6214          TL [BRX], #/100000       ;AND NEGATIVE
; 6215  =0      [AR]_[AR] SWAP           ;LAST AND MINUS, USE LH
; 6216          [AR]_0, HOLD RIGHT, J/BDSET
; 6217
; 6218  BDABT:  [AR]_WORK[BDH], J/DAC
; 6219
; 6220  BDCFLG: [BR]_[BR].AND.NOT.#,      ;CLEAR FLAG THAT TELLS US
; 6221          #/040000, HOLD RIGHT,    ; TO SUBTRACT 1 AND
; 6222          J/DOCVT3                 ; CONTINUE CONVERTING
; 6223
```

```
U 2216, 2165,0551,0616,4374,4007,0701,0000,0000,0344
U 2217, 2675,4443,0000,4174,4007,0700,0000,0000,0000


U 2165, 2222,3441,1617,4174,4007,0700,0210,0000,0010

U 2167, 2220,1551,0404,6274,4007,0561,4000,0000,0000
U 2220, 2221,1111,0703,4174,4007,0700,4000,0000,0000
U 2221, 3376,0551,0616,4374,4007,0701,0000,0000,0373
U 3376, 3377,3333,0016,4174,4007,0700,0200,0000,0010

U 3377, 2222,1551,0303,6274,4007,0522,4000,0000,0000


U 2222, 0002,4551,0404,4374,0004,1700,0000,0037,7777
U 2223, 3400,0551,0303,6274,4007,0700,0000,0000,0000
U 3400, 3401,3333,0017,4174,4007,0700,0200,0000,0010
U 3401, 2224,0551,0404,6274,4007,0561,0000,0000,0000


U 2224, 0006,4551,0404,4374,0004,1700,0000,0037,7777

U 2225, 2224,0111,0703,4174,4007,0700,0000,0000,0000
```

```
; 6224  ;SUBROUTINE TO SUBRTACT A POWER OF 10 FROM AR!ARX
; 6225  ;CALL WITH:
; 6226  ;       AR!ARX/ NUMBER TO BE CONVERTED
; 6227  ;       BRX(RIGHT)/ POWER OF 10
; 6228  ;RETURNS:
; 6229  ;       2 RESULT IS STILL POSITIVE
; 6230  ;       6 RESULT WOULD HAVE BEEN NEGATIVE (RESTORE DONE)
; 6231  =0
; 6232  BDSUB:  [TO]_[BRX]+#, 3T, WORK/DECLO, ;ADDRESS OF LOW WORD
; 6233          J/BDSUB1                ;NO INTERRUPT
; 6234          J/FIXPC                 ;INTERRUPT
; 6235  =0*
; 6236  BDSUB1: [T1]_[TO], LOAD VMA,     ;PUT IN VMA,
; 6237          CALL [CLARXO]           ;FIX UP SIGN OF LOW WORD
; 6238          [ARX]_[ARX]-RAM, 3T,    ;SUBTRACT
; 6239          SKIP CRY1               ;SEE IF OVERFLOW
; 6240  =0      [AR]_[AR]-1             ;PROCESS CARRY
; 6241          [TO]_[BRX]+#, 3T, WORK/DECHI ;ADDRESS OF HIGH WORD
; 6242          READ [TO], LOAD VMA     ;PLACE IN VMA
; 6243          [AR]_[AR]-RAM, 4T,      ;SUBTRACT
; 6244          SKIP DPO                ;SEE IF IT FIT
; 6245  =0
; 6246  CLARXO: CLEAR ARXO,             ;IT FIT, KEEP LOW WORD +
; 6247          RETURN [2]              ; AND RETURN
; 6248          [AR]_[AR]+RAM           ;RESTORE
; 6249          READ [T1], LOAD VMA
; 6250          [ARX]_[ARX]+RAM, 3T, SKIP CRY1
; 6251  =0
; 6252  BDSUB2: CLEAR ARXO,             ;KEEP LOW WORD +
; 6253          RETURN [6]              ;RETURN OVERFLOW
; 6254          [AR]_[AR]+1,            ;ADD BACK THE CARRY
; 6255          J/BDSUB2                ;COMPLETE SUBTRACT
; 6256
```

```
U 3741, 3457,0111,0703,4170,4007,0700,0210,0004,0012
U 3751, 2111,3771,0006,0276,6007,0700,0000,0000,0000

U 2111, 3556,4553,0600,4374,4007,0321,0010,0004,7777
U 2115, 3402,3443,0600,4174,4007,0700,0200,0004,0012
U 3402, 3403,4221,0013,4170,4007,0700,0000,0000,0000
U 3403, 2226,3770,0605,4344,4007,0700,0000,0000,0000

U 2226, 3661,0115,0505,4174,4007,0700,0010,0000,0000
U 2227, 0654,3333,0005,4174,4003,1701,0000,0000,0000

U 0654, 2230,3770,0303,4344,4007,0700,2000,0071,0007
U 0655, 2231,3770,0303,4344,4007,0700,0000,0000,0000
U 0656, 2230,3447,0303,4174,4007,0700,2000,0071,0006
U 0657, 3404,4551,0303,4374,4007,0700,0000,0000,0777

U 2230, 2230,3447,0303,4174,4007,0630,2000,0060,0000
U 2231, 3404,4551,0303,4374,4007,0700,0000,0000,0777
```

```
; 6257  .TOC    "EXTEND -- EDIT -- MAIN LOOP"
; 6258
; 6259  ;HERE FOR EDIT INSTRUCTION
; 6260  ;CALL WITH:
; 6261  ;       AR/     E0      ADDRESS OF FILL, FLOAT, AND MESSAGE TABLE
; 6262  ;       BR/     E1      TRANSLATE TABLE
; 6263  ;
; 6264  3741:
; 6265  EDIT:   VMA_[AR]+1, START READ, ;FIRST GET FILL BYTE
; 6266          CALL [GTFILL]           ;GO GET IT
; 6267  3751:   [BRX]_AC                ;GET PATTERN POINTER
; 6268  =0**     TL [BRX], #/047777,     ;MAKE SURE SECTION 0
; 6269          CALL [BITCHK]           ; ..
; 6270  EDITLP: VMA_[BRX], START READ   ;FETCH PATTERN WORD
; 6271          END STATE               ;NO SPECIAL PAGE FAIL ACTION
; 6272          [BR]_[BRX] SWAP         ;GET PBN IN BITS 20 & 21
; 6273  =0       [BR]_[BR]*4,          ; ..
; 6274          CALL [LOADAR]           ;GET PATTERN WORD
; 6275          READ [BR], 3T, DISP/DP LEFT
; 6276  =1100
; 6277          [AR]_[AR] SWAP, SC_7, J/MOVPAT  ;(0) BITS 0-8
; 6278          [AR]_[AR] SWAP, J/MSKPAT        ;(1) BITS 9-17
; 6279          [AR]_[AR]*.5, SC_6, J/MOVPAT    ;(2) BITS 18-27
; 6280          [AR]_[AR].AND.#, #/777, J/EDISP ;(3) BITS 28-35
; 6281  =0
; 6282  MOVPAT: [AR]_[AR]*.5, STEP SC, J/MOVPAT ;SHIFT OVER
; 6283  MSKPAT: [AR]_[AR].AND.#, #/777
; 6284
```

```
U 3404, 2232,3447,0305,4174,4007,0700,2000,0071,0002
U 2232, 2232,3447,0505,4174,4007,0630,2000,0060,0000
U 2233, 0661,3333,0005,4174,4003,5701,0000,0000,0000
U 0661, 2234,1553,0300,4374,4007,0532,4000,0000,0005
U 0663, 2252,3333,0006,4174,4007,0520,0000,0000,0000
U 0665, 3424,4443,0000,4174,4007,0700,0000,0000,0000
U 0667, 3424,4443,0000,4174,4007,0700,0000,0000,0000
U 0671, 3424,4443,0000,4174,4007,0700,0000,0000,0000
U 0673, 2256,4553,0600,4374,4007,0321,0000,0010,0000
U 0675, 2256,4553,0600,4374,4007,0321,0000,0020,0000
U 0677, 2256,4443,0000,4174,4007,0700,0000,0000,0000
U 2234, 3424,4443,0000,4174,4007,0700,0000,0000,0000
U 2235, 0710,3333,0003,4174,4003,5701,0000,0000,0000
U 0710, 3405,0111,0701,4174,4007,0700,0000,0000,0000
U 0711, 2244,3771,0013,4370,4007,0700,0000,0000,0011
U 0712, 0006,3333,0006,4174,4007,0520,0000,0000,0000
U 0713, 3424,4551,0606,4374,0007,0700,0000,0007,7777
U 0714, 0715,3771,0005,1276,6007,0701,0000,0000,1443
U 0715, 2240,3443,0500,4174,4007,0700,0200,0004,0012
```

```
; 6285  ;HERE WITH PATTERN BYTE RIGHT ADJUSTED IN AR
; 6286  EDISP:  [BR]_[AR]*.5, SC_2       ;SHIFT OVER
; 6287  =O
; 6288  EDISP1: [BR]_[BR]*.5, STEP SC, J/EDISP1
; 6289          READ [BR], 3T, DISP/DP   ;LOOK AT HIGH 3 BITS
; 6290  =0001                           ;(O) OPERATE GROUP
; 6291          [AR]-#, #/5, 4T,         ;       SEE IF O-4
; 6292          SKIP DP18, J/EDOPR
; 6293                                   ;(1) MESSAGE BYTE
; 6294          READ [BRX], SKIP DPO,
; 6295          J/EDMSG
; 6296                                   ;(2) UNDEFINED
; 6297          J/EDNOP
; 6298                                   ;(3) UNDEFINED
; 6299          J/EDNOP
; 6300                                   ;(4) UNDEFINED
; 6301          J/EDNOP
; 6302                                   ;(5) SKIP IF M SET
; 6303          TL [BRX], #/100000,
; 6304          J/EDSKP
; 6305                                   ;(6) SKIP IF N SET
; 6306          TL [BRX], #/200000,
; 6307          J/EDSKP
; 6308                                   ;(7) SKIP ALWAYS
; 6309          J/EDSKP
; 6310
; 6311  .TOC    "EXTEND -- EDIT -- DECODE OPERATE GROUP"
; 6312
; 6313  ;HERE FOR OPERATE GROUP. SKIP IF IN RANGE
; 6314  =O
; 6315  EDOPR:  J/EDNOP                 ;OUT OF RANGE
; 6316          READ [AR], 3T, DISP/DP   ;DISPATCH ON TYPE
; 6317  =1000   [PC]_[PC]+1, J/EDSTOP    ;(O) STOP EDIT
; 6318          STATE_[EDIT-SRC],        ;(1) SELECT SOURCE BYTE
; 6319          J/EDSEL
; 6320          READ [BRX], SKIP DPO,    ;(2) START SIGNIFICANCE
; 6321          J/EDSSIG
; 6322          [BRX]_[BRX].AND.#,       ;(3) FIELD SEPERATOR
; 6323          #/77777, HOLD RIGHT,
; 6324          J/EDNOP
; 6325          [BR]_AC[MARK]            ;(4) EXCHANGE MARK AND DEST
; 6326          VMA_[BR], START READ,
; 6327          J/EDEXMD
; 6328  =
; 6329
```

```
U 3405,  3406,7441,0605,4174,4007,0700,1000,0071,0010
U 3406,  3407,3441,0603,4174,4007,0701,1000,0043,0000

U 3407,  2236,4553,0500,4374,4007,0321,0000,0003,0000

U 2236,  1515,3770,0303,4334,4017,0700,0000,0041,0000


U 2237,  2236,0111,0703,4174,4007,0700,1000,0051,0700

U 0006,  3413,4443,0000,4174,4007,0700,0010,0000,0000
U 0007,  3424,4443,0000,4174,4007,0700,0000,0000,0000

U 2240,  3661,3772,0000,1275,5007,0701,0010,0000,1444
U 2241,  3410,4443,0000,4174,4007,0700,0200,0003,0002

U 3410,  3411,3223,0000,4174,4007,0701,0200,0000,0002
U 3411,  3424,3440,0303,1174,4007,0700,0400,0000,1444
```

```
; 6330  .TOC    "EXTEND -- EDIT -- STOP EDIT"
; 6331
; 6332  ;HERE TO END AN EDIT OPERATION. PC IS SET TO SKIP IF NORMAL END
; 6333  ; OR NON-SKIP IF ABORT
; 6334  EDSTOP: [BR]_.NOT.[BRX],          ;AD WILL NOT DO D.AND.NOT.A
; 6335          FE_S#, S#/10              ;PRESET FE
; 6336          [AR]_[BRX], 3T, FE_FE+P ;MOVE POINTER, UPBATE PBN
; 6337          [BR].AND.#, 3T,           ;WAS OLD NUMBER 3?
; 6338          #/030000, SKIP ADL.EQ.0 ; ..
; 6339  =0
; 6340  EDSTP1: [AR]_P, J/STAC            ;NO--ALL DONE
; 6341          [AR]_[AR]+1,              ;YES--BUMP WORD #
; 6342          FE_FE.AND.S#, S#/0700,    ;KEEP ONLY FLAG BITS
; 6343          J/EDSTP1                  ;GO STOP EDIT
; 6344
; 6345  .TOC    "EXTEND -- EDIT -- START SIGNIFICANCE"
; 6346
; 6347  ;HERE WITH DST POINTER IN AR
; 6348  =110
; 6349  EDSSIG: CALL [EDFLT]              ;STORE FLT CHAR
; 6350          J/EDNOP                   ;DO NEXT PATTERN BYTE
; 6351
; 6352  .TOC    "EXTEND -- EDIT -- EXCHANGE MARK AND DESTINATION"
; 6353
; 6354  ;HERE WITH ADDRESS OF MARK POINTER IN BR
; 6355  =0
; 6356  EDEXMD: Q_AC[DSTP],              ;GET DEST POINTER
; 6357          CALL [LOADAR]             ;GO PUT MARK IN AR
; 6358          START WRITE               ;START WRITE. SEPERATE STEP TO AVOID
; 6359                                    ; PROBLEM ON DPM5
; 6360          MEM WRITE, MEM_Q          ;PUT OLD DEST IN MARK
; 6361          AC[DSTP]_[AR], J/EDNOP    ;PUT BACK DEST POINTER
; 6362
```

```
U 2244, 2277,3771,0003,1276,6007,0701,0010,0000,1441
U 2246, 0700,3447,0303,7174,4007,0700,0000,0000,0241

U 0700, 3444,0551,0303,7274,4007,0700,0010,0000,0241



U 0702, 2242,3333,0003,4174,4007,0621,0000,0000,0000

U 0703, 0246,3771,0013,4370,4007,0700,0000,0000,0011
U 0704, 3405,4443,0000,4174,4007,0700,0000,0000,0000


U 0705, 3451,3771,0013,4370,4007,0700,0010,0000,0013

U 0707, 3424,4443,0000,4174,4007,0700,0000,0000,0000




U 2242, 0705,4443,0000,4174,4007,0700,0000,0000,0000
U 2243, 3424,4443,0000,4174,4007,0700,0000,0000,0000




U 0246, 3413,3333,0004,7174,4007,0700,0410,0000,0246
U 0247, 3412,3771,0003,7274,4007,0701,0000,0000,0246


U 3412, 0705,4251,0303,4374,4007,0700,0000,0007,7777
```

```
; 6363  .TOC    "EXTEND -- EDIT -- PROCESS SOURCE BYTE"
; 6364
; 6365  =0*
; 6366  EDSEL:  [AR]_AC[SRCP],          ;PICK UP SRC POINTER
; 6367          CALL [GETSRC]           ;GET SOURCE BYTE
; 6368          [AR]_[AR]*.5, WORK[E1]  ;PREPARE TO TRANSLATE
; 6369  =000    [AR]_[AR]+WORK[E1],     ;GO TRANSLATE BY HALFWORDS
; 6370          2T, CALL [TRNAR]        ; ..
; 6371  =010
; 6372  EDFILL: READ [AR],              ;(2) NO SIGNIFICANCE, GO FILL
; 6373          SKIP AD.EQ.0,           ;    SEE IF ANY FILLER
; 6374          J/EDFIL1                ;    GO TO IT
; 6375          STATE_[EDIT-SRC],       ;(3) SIG START, DO FLOAT CHAR
; 6376          J/EDSFLT
; 6377  =100    J/EDSTOP                ;(4) ABORT
; 6378  =101
; 6379  EDSPUT: STATE_[EDIT-S+D],       ;(5) NORMAL, STORE AT DST
; 6380          CALL [PUTDST]           ;    ..
; 6381  =111
; 6382          J/EDNOP                 ;(7) BYTE STORED
; 6383  =
; 6384
; 6385  ;HERE TO COMPLETE STORING FILL
; 6386  =0
; 6387  EDFIL1: J/EDSPUT                ;STORE FILLER
; 6388          J/EDNOP                 ;NO FILLER TO STORE
; 6389
; 6390  ;HERE TO DO FLOAT BYTE
; 6391  =110
; 6392  EDSFLT: WORK[FSIG]_[ARX],        ;SAVE SIG CHAR
; 6393          CALL [EDFLT]            ;STORE FLOAT CHAR
; 6394          [AR]_WORK[FSIG]         ;RESTORE CHAR
; 6395          [AR]_[AR].AND.# CLR LH, ;JUST KEEP THE BYTE IN CASE
; 6396          #/77777,                ; DEST BYTE .GT. 15 BITS
; 6397          J/EDSPUT                ;GO STORE CHAR WHICH STARTED THIS AL

; 6398
```

```
U 3413, 3414,3771,0005,1276,6007,0701,0000,0000,1443
U 3414, 3415,3443,0500,4174,4007,0700,0200,0003,0012
U 3415, 3416,3771,0005,1276,6007,0701,0000,0000,1444
U 3416, 2250,3333,0005,4175,5007,0701,0200,0000,0002

U 2250, 3420,4751,1203,4374,4007,0700,0010,0000,0002

U 2251, 0740,3771,0003,4365,5007,0621,0200,0000,0002




U 0740, 3451,3551,1313,4370,4007,0700,0010,0000,0012

U 0741, 3417,3551,0606,4374,0007,0700,0000,0040,0000

U 0746, 3417,3551,0606,4374,0007,0700,0000,0040,0000




U 3417, 0007,3440,0606,0174,4004,1700,0400,0000,0000




U 2252, 0760,3771,0003,7274,4007,0622,0000,0000,0244

U 2253, 2254,4251,0303,4374,4007,0700,0000,0000,0077

U 2254, 3420,0111,0703,7174,4007,0700,0010,0000,0240
U 2255, 0760,3771,0003,4365,5007,0700,0200,0000,0002


U 0760, 3451,3771,0013,4370,4007,0700,0010,0000,0012
U 0761, 3424,4443,0000,4174,4007,0700,0000,0000,0000
U 0766, 3424,4443,0000,4174,4007,0700,0000,0000,0000

U 3420, 3421,0551,0303,7274,4007,0701,0000,0000,0240

U 3421, 0001,3443,0300,4174,4004,1700,0200,0004,0012
```

```
; 6399  ;SUBRUTINE TO PROCESS FLOAT CHAR
; 6400  ;CALL WITH:
; 6401  ;       AR/ POINTER TO STORE @ MARK
; 6402  ;RETURN 7 WITH FLOAT STORED
; 6403  EDFLT:  [BR]_AC[MARK]           ;ADDRESS OF MARK POINTER
; 6404          VMA_[BR], START WRITE   ;READY TO STORE
; 6405          [BR]_AC[DSTP]           ;GET DST POINTER
; 6406          MEM WRITE, MEM_[BR]     ;STORE POINTER
; 6407  =0      [AR]_0 XWD [2],         ;FETCH FLOAT CHAR
; 6408          CALL [EDBYTE]           ;GET TBL BYTE
; 6409          MEM READ, [AR]_MEM,     ;GET FLOAT CHAR
; 6410          SKIP AD.EQ.0            ;SEE IF NULL
; 6411  =000
; 6412          [FLG]_[FLG].OR.#,       ;REMEMBER TO BACKUP DST POINTER
; 6413          STATE/EDIT-DST,         ; WILL ALSO BACKUP SRC IF CALLED
; 6414          HOLD LEFT,              ; FROM SELECT
; 6415          CALL [PUTDST]           ; STORE FLOAT
; 6416  =001    [BRX]_[BRX].OR.#, #/400000,
; 6417          HOLD RIGHT,  J/EDFLT1   ;NULL
; 6418  =110    [BRX]_[BRX].OR.#, #/400000,
; 6419          HOLD RIGHT,  J/EDFLT1   ;MARK STORED
; 6420  =
; 6421  EDFLT1: AC_[BRX],               ;SAVE FLAGS SO WE DON'T
; 6422                                  ;TRY TO DO THIS AGAIN IF
; 6423                                  ;NEXT STORE PAGE FAILS
; 6424          RETURN [7]              ;AND RETURN
; 6425
; 6426  .TOC    "EXTEND -- EDIT -- MESSAGE BYTE"
; 6427
; 6428  ;HERE WITH SKIP ON S
; 6429  =0
; 6430  EDMSG:  [AR]_WORK[FILL],        ;GET FILL BYTE
; 6431          SKIP AD.EQ.0, 4T,       ;SEE IF NULL
; 6432          J/EDMSG1                ;GO STORE
; 6433          [AR]_[AR].AND.# CLR LH, ;GET OFFSET INTO TABLE
; 6434          #/77
; 6435  =0      [AR]_[AR]+1, WORK[EO],  ;PLUS 1
; 6436          CALL [EDBYTE]           ;GET TBL BYTE
; 6437          MEM READ, [AR]_MEM      ;FROM MEMORY
; 6438  =000
; 6439  EDMSG1: STATE_[EDIT-DST],       ;WHAT TO DO ON PAGE FAILS
; 6440          CALL [PUTDST]           ;STORE MESSAGE BYTE
; 6441  =001    J/EDNOP                 ;NULL FILLER
; 6442  =110    J/EDNOP                 ;NEXT BYTE
; 6443  =
; 6444
; 6445  EDBYTE: [AR]_[AR]+WORK[EO]      ;GET OFFSET INTO TABLE
; 6446          VMA_[AR], START READ,   ;START MEMORY CYCLE
; 6447          RETURN [1]              ;RETURN TO CALLER
; 6448
```

```
U 2256, 3422,4551,0303,4374,4007,0700,0000,0000,0077
U 2257, 3424,4443,0000,4174,4007,0700,0000,0000,0000
U 3422, 3423,0115,0703,4174,4007,0700,0000,0000,0000
U 3423, 3425,3333,0003,4174,4007,0701,2000,0007,0000
U 3424, 3425,4443,0000,4174,4007,0700,2000,0071,0000
U 3425, 3426,3333,0006,4174,4007,0701,1000,0073,0000
U 3426, 3427,4443,0000,4174,4007,0700,1000,0051,0030
U 3427, 3430,4443,0000,4174,4007,0700,1000,0040,0000
U 3430, 3431,4443,0000,4174,4007,0700,1000,0041,0010
U 3431, 3432,3777,0003,4334,4057,0700,2000,0041,0000
U 3432, 2260,4251,0303,4374,4007,0630,0000,0000,0170
U 2260, 2262,3447,0303,4174,4007,0700,2000,0071,0000
U 2261, 2260,3551,0303,4370,4007,0700,0000,0000,0200
U 2262, 2262,3447,0303,4174,4007,0630,2000,0060,0000
U 2263, 3433,0111,0306,4170,4007,0700,0000,0000,0000
U 3433, 3434,3770,0303,4334,4017,0700,0000,0041,0000
U 3434, 3435,4551,0606,4374,0007,0700,0000,0070,0000
U 3435, 3436,4551,0303,4374,4007,0700,0000,0003,0000
U 3436, 3437,3111,0306,4174,0007,0700,0000,0000,0000
U 3437, 2115,3440,0606,0174,4007,0700,0400,0000,0000
```

```
; 6449  .TOC    "EXTEND -- EDIT -- SKIP"
; 6450
; 6451  =0
; 6452  ;HERE TO SKIP ALWAYS
; 6453  EDSKP:  [AR]_[AR].AND.#, #/77,   ;JUST KEEP SKIP DISTANCE
; 6454          J/EDSKP1                ;CONTINUE BELOW
; 6455  ;HERE IF WE DO NOT WANT TO SKIP
; 6456          J/EDNOP
; 6457  EDSKP1: [AR]_([AR]+1)*2          ;GIVE 1 EXTRA SKIP
; 6458          READ [AR], SCAD/A*2,    ;PUT THE ADJUSTMENT
; 6459          SCADA/BYTE5, 3T, LOAD SC, ; THE SC
; 6460          J/EDNOP1                ;JOIN MAIN LOOP
; 6461
; 6462
; 6463  .TOC    "EXTEND -- EDIT -- ADVANCE PATTERN POINTER"
; 6464
; 6465  EDNOP:  SC_0                     ;NO SKIP
; 6466  EDNOP1: READ [BRX], 3T, FE_P     ;PUT PBN IN FE
; 6467          FE_FE.AND.S#, S#/30     ;JUST BYTE #
; 6468          FE_FE+SC                ;ADD IN ANY SKIP DISTANCE
; 6469          FE_FE+S#, S#/10         ;BUMP PBN
; 6470          [AR]_FE,                ;GET NUMBER OF WORDS
; 6471          LOAD SC                 ;PUT MSB WHERE IT CAN BE TESTED
; 6472                                  ; QUICKLY
; 6473          [AR]_[AR].AND.# CLR LH, ;KEEP ONLY 1 COPY
; 6474          #/170, SKIP/SC          ; ..
; 6475  =0
; 6476  EDN1A:  [AR]_[AR]*.5, SC_0,
; 6477          J/EDNOP2                ;READY TO SHIFT OFF BYTE WITHIN
; 6478                                  ; WORD
; 6479          [AR]_[AR].OR.#, #/200,  ;GET THE SIGN BIT OF THE FE
; 6480          HOLD LEFT,              ; INTO THE AR. ONLY HAPPENS ON
; 6481          J/EDN1A                 ; SKP 76 OR SKP 77
; 6482  =0
; 6483  EDNOP2: [AR]_[AR]*.5, STEP SC, J/EDNOP2
; 6484          [BRX]_[BRX]+[AR],       ;UPDATE WORD ADDRESS
; 6485          HOLD LEFT
; 6486          [AR]_P                  ;PUT PBN BACK IN BRX
; 6487          [BRX]_[BRX].AND.#,      ;JUST KEEP FLAGS
; 6488          #/700000,               ; ..
; 6489          HOLD RIGHT
; 6490          [AR]_[AR].AND.#,        ;JUST KEEP PBN
; 6491          #/030000
; 6492          [BRX]_[BRX].OR.[AR],    ;FINAL ANSWER
; 6493          HOLD RIGHT
; 6494          AC_[BRX], J/EDITLP      ;DO NEXT FUNCTION
; 6495
```

```
U 0332, 3451,3771,0003,7274,4007,0700,0010,0000,0244
U 0336, 3440,3771,0003,1276,6007,0701,0000,0000,1443

U 3440, 2264,0113,0703,1174,4007,0521,0400,0000,1443
U 2264, 0002,4443,0000,4174,4004,1700,0000,0000,0000
U 2265, 0332,4443,0000,7174,4007,0700,0000,0000,0244
```

```
; 6496  .TOC    "EXTEND SUBROUTINES -- FILL OUT DESTINATION"
; 6497
; 6498  ;CALL WITH
; 6499  ;       AC[DLEN]/ NEGATIVE NUMBER OF BYTES LEFT IN DEST
; 6500  ;       FILL/  FILL BYTE
; 6501  ;       RETURN [2] WITH FILLERS STORED
; 6502  ;
; 6503  ;NOTE: THIS ROUTINE NEED NOT TEST FOR INTERRUPTS ON EACH BYTE
; 6504  ;       BECAUSE EVERY BYTE STORE DOES A MEMORY READ.
; 6505  ;
; 6506  =O1*
; 6507  MOVF1:  [AR]_WORK[FILL], 2T,     ;GET FILL BYTE
; 6508          CALL [PUTDST]           ;PLACE IN DEST
; 6509          [AR]_AC[DLEN]           ;AMOUNT LEFT
; 6510          AC[DLEN]_[AR]+1, 3T,    ;STORE UPDATED LEN
; 6511          SKIP* DPO               ; AND SEE IF DONE
; 6512  =O      RETURN [2]              ;DONE
; 6513  MOVFIL: WORK[FILL], J/MOVF1     ;DO ANOTHER BYTE
; 6514                                  ;ENTERING HERE SAVES 150NS
; 6515                                  ; PER BYTE BUT COSTS 300NS
; 6516                                  ; PER FIELD MOVED. I ASSUME (BUT DO
; 6517                                  ; NOT KNOW) THAT THIS SPEEDS
; 6518                                  ; THINGS UP.
; 6519
```

```
U 1074, 2276,3333,0003,7174,4007,0520,0410,0000,0242
U 1075, 0001,4221,0013,4170,4004,1700,0000,0000,0000
U 1076, 0716,4443,0000,7174,4003,7700,0000,0000,0241

U 0716, 3443,3447,0303,4174,4007,0700,0000,0000,0000
U 0717, 3441,3770,0303,7174,0007,0700,0000,0000,0241




U 3441, 3442,0551,0303,7274,4007,0700,0000,0000,0241

U 3442, 0004,4553,0300,7274,4004,1622,0000,0000,0243
```

```
; 6520  .TOC"EXTEND SUBROUTINES -- GET MODIFIED SOURCE BYTE"
; 6521
; 6522  ;CALL WITH:
; 6523  ;SLEN = MINUS LENGTH OF STRING
; 6524  ;MSK = MASK FOR BYTE SIZE (1 IF BIT MUST BE ZERO)
; 6525  ;E1 = EFFECTIVE ADDRESS OF OPERATION WORD (SIGN EXTENDED IF OFFSET)
; 6526  ;       [AR]_WORK[SLEN]+1, CALL [SRCMOD]
; 6527  ;RETURNS:
; 6528  ;       1 LENGTH EXHAUSTED
; 6529  ;       2 (EDIT ONLY) NO SIGNIFICANCE
; 6530  ;       3 (EDIT ONLY) SIGNIFICANCE START:
; 6531  ;       4 ABORT: OUT OF RANGE OR TRANSLATE FAILURE
; 6532  ;       5 NORMAL: BYTE IN AR
; 6533  ;
; 6534  ;DROM B SET AS FOLLOWS:
; 6535  ;       0 TRANSLATE
; 6536  ;       1 OFFSET
; 6537  ;       2 EDIT
; 6538  ;       4 CVTDBT
; 6539  =OO
; 6540  SRCMOD: WORK[SLEN]_[AR],          ;PUT BACK SOURCE LENGTH
; 6541          SKIP DPO,                ;SEE IF DONE
; 6542          CALL [GSRC]              ;GET A SOURCE BYTE
; 6543          END STATE, RETURN [1]    ;DONE
; 6544          WORK[E1], B DISP         ;OFFSET OR TRANSLATE?
; 6545  =
; 6546  =1110    [AR]_[AR]*.5, J/XLATE    ;TRANSLATE
; 6547          FIX [AR] SIGN, WORK[E1] ;IF WE ARE PROCESSING FULL WORD
; 6548                                   ; BYTES, AND THEY ARE NEGATIVE,
; 6549                                   ; AND THE OFFSET IS POSITIVE THEN
; 6550                                   ; WE HAVE TO MAKE BITS -1 AND -2
; 6551                                   ; COPIES OF THE SIGN BIT.
; 6552          [AR]_[AR]+WORK[E1], 2T   ;OFFSET
; 6553          [AR].AND.WORK[MSK],      ;VALID BYTE?
; 6554          SKIP AD.EQ.O, 4T,        ;SKIP IF OK
; 6555          RETURN [4]               ;RETURN 4 IF BAD, 5 IF OK
; 6556
```

```
U 3443, 3444,0551,0303,7274,4007,0701,0000,0000,0241

U 3444, 2266,3333,0003,4174,4007,0700,0200,0004,0012



U 2266, 3662,3445,0303,4174,4007,0700,0010,0000,0000
U 2267, 2270,4553,0300,4374,4007,0331,0000,0000,0001



U 2270, 0721,3441,0403,4174,4003,1701,0000,0000,0000

U 2271, 2270,3770,0404,4344,4007,0700,0000,0000,0000
```

```
; 6557  .TOC    "EXTEND SUBROUTINES -- TRANSLATE"
; 6558
; 6559  ;HERE WITH BYTE IN AR 1-36. FETCH TABLE ENTRY.
; 6560  XLATE:  [AR]_[AR]+WORK[E1]       ;COMPUTE ADDRESS
; 6561  TRNAR:  READ [AR], LOAD VMA,     ;FETCH WORD
; 6562          START READ               ; ..
; 6563  =0      [AR]_[AR]*2,             ;GET BACK LSB
; 6564                                   ;BIT 36 IS NOT PRESERVED
; 6565                                   ; BY PAGE FAILS
; 6566          CALL [LOADARX]           ;PUT ENTRY IN ARX
; 6567          TR [AR], #/1             ;WHICH HALF?
; 6568  =0
; 6569  XLATE1: [AR]_[ARX], 3T,          ;RH -- COPY TO AR
; 6570          DISP/DP LEFT,            ;DISPATCH ON CODE
; 6571          J/TRNFNC                 ;DISPATCH TABLE
; 6572          [ARX]_[ARX] SWAP,        ;LH -- FLIP AROUND
; 6573          J/XLATE1                 ;START SHIFT
; 6574
```

```
U 0721, 2272,3333,0006,4174,4007,0520,0000,0000,0000
U 0723, 0004,4443,0000,4174,4004,1700,0000,0000,0000
U 0725, 0721,5551,0606,4374,0007,0700,0000,0010,0000
U 0727, 0721,3551,0606,4374,0007,0700,0000,0010,0000
U 0731, 0721,3551,0606,4374,0007,0700,0000,0020,0000
U 0733, 0004,3551,0606,4374,0004,1700,0000,0020,0000
U 0735, 0731,5551,0606,4374,0007,0700,0000,0010,0000
U 0737, 0721,3551,0606,4374,0007,0700,0000,0030,0000
```

```
; 6575  ;HERE ON TRANSLATE OPERATION TO PERFORM FUNCTIONS REQUIRED BY
; 6576  ; THE 3 HIGH ORDER BITS OF THE TRANSLATE FUNCTION HALFWORD. WE
; 6577  ; DISPATCH ON FUNCTION AND HAVE:
; 6578  ;       BRX/    FLAGS
; 6579  ;       ARX/    TABLE ENTRY IN RH
; 6580  ;
; 6581  =0001
; 6582                                          ;(0) NOP
; 6583  TRNFNC: READ [BRX], SKIP DPO,            ;S FLAG ALREADY SET?
; 6584          J/TRNRET                        ; ..
; 6585                                          ;(1) ABORT
; 6586          RETURN [4]
; 6587                                          ;(2) CLEAR M FLAG
; 6588          [BRX]_[BRX].AND.NOT.#,
; 6589          #/100000, HOLD RIGHT,
; 6590          J/TRNFNC
; 6591                                          ;(3) SET M FLAG
; 6592          [BRX]_[BRX].OR.#,
; 6593          #/100000, HOLD RIGHT,
; 6594          J/TRNFNC
; 6595                                          ;(4) SET N FLAG
; 6596  TRNSIG: [BRX]_[BRX].OR.#,
; 6597          #/200000, HOLD RIGHT,
; 6598          J/TRNFNC
; 6599                                          ;(5) SET N FLAG THEN ABORT
; 6600          [BRX]_[BRX].OR.#,
; 6601          #/200000, HOLD RIGHT,
; 6602          RETURN [4]
; 6603                                          ;(6) CLEAR M THEN SET N
; 6604          [BRX]_[BRX].AND.NOT.#,
; 6605          #/100000, HOLD RIGHT,
; 6606          J/TRNSIG
; 6607                                          ;(7) SET N AND M
; 6608          [BRX]_[BRX].OR.#,
; 6609          #/300000, HOLD RIGHT,
; 6610          J/TRNFNC
; 6611
```

```
U 2272, 0754,3333,0004,4174,4003,7530,0000,0000,0000
U 2273, 0005,4251,0403,4374,4004,1700,0000,0007,7777


U 0754, 0533,3771,0003,1276,6003,7701,0000,0000,1443


U 0755, 2273,3551,0606,4374,0007,0700,0000,0040,0000
U 0756, 0002,3771,0003,7274,4004,1701,0000,0000,0244
U 0757, 0003,3771,0003,1276,6004,1701,0000,0000,1444


U 0533, 3445,1111,0703,4174,4007,0700,4000,0000,0000
U 0537, 1074,0551,0703,7274,4007,0701,0000,0000,0242
U 3445, 2274,3770,0303,1174,4007,0520,0400,0000,1443
U 2274, 1074,3771,0003,7274,4007,0701,0000,0000,0242
U 2275, 1074,0551,0703,7274,4007,0701,0000,0000,0242
```

```
; 6612  ;HERE TO COMPLETE A TRANSLATE
; 6613
; 6614  =0
; 6615  TRNRET: READ [ARX], SKIP DP18,   ;S-FLAG IS ZERO
; 6616          B DISP, SKIP DP18,      ;SEE IF EDIT OR SIG START
; 6617          J/TRNSS                 ; ..
; 6618  TRNSS1: [AR]_[ARX].AND.# CLR LH, ;S IS SET, JUST RETURN BYTE
; 6619          #/77777, RETURN [5]     ; ..
; 6620
; 6621  =1100
; 6622  TRNSS:  [AR]_AC[DLEN],           ;NO SIG ON MOVE OR D2B
; 6623          B DISP, J/TRNNS1        ;SEE IF D2B
; 6624          [BRX]_[BRX].OR.#,       ;SIG START ON MOVE OR D2B
; 6625          #/400000, HOLD RIGHT,
; 6626          J/TRNSS1                ;RETURN BYTE
; 6627          [AR]_WORK[FILL],        ;EDIT--NO SIG RETURN FILL
; 6628          RETURN [2]              ; ..
; 6629          [AR]_AC[DSTP],          ;EDIT--START OF SIG
; 6630          RETURN [3]              ; ..
; 6631
; 6632  =1011
; 6633  TRNNS1: [AR]_[AR]-1, J/TRNNS2   ;COMPENSATE FOR IGNORING SRC
; 6634          [AR]_WORK[SLEN]+1,      ;DEC TO BIN HAS NO DEST LENGTH
; 6635          J/SRCMOD                ;JUST UPDATE SRC LENTH
; 6636  TRNNS2: AC[DLEN]_[AR] TEST,     ;PUT BACK DLEN AND
; 6637          SKIP DPO                ; SEE WHICH IS NOW SHORTER
; 6638  =0      [AR]_WORK[SLEN],        ;DEST IS SHORTER. DO NOT CHANGE
; 6639          J/SRCMOD                ; AMOUNT LEFT
; 6640          [AR]_WORK[SLEN]+1,      ;GO LOOK AT NEXT BYTE
; 6641          J/SRCMOD
; 6642
```

```
                                                ; 6643  .TOC    "EXTEND SUBROUTINES -- GET UNMODIFIED SOURCE BYTE"
                                                ; 6644
                                                ; 6645  ;CALL:
                                                ; 6646  ;       GSRC WITH SKIP ON SOURCE LENGTH
                                                ; 6647  ;       GETSRC IF LENGHT IS OK
                                                ; 6648  ;WITH:
                                                ; 6649  ;       AC1/ SOURCE BYTE POINTER
                                                ; 6650  ;RETURNS:
                                                ; 6651  ;       1 IF LENGTH RAN OUT
                                                ; 6652  ;       2 IF OK (BYTE IN AR)
                                                ; 6653  ;
                                                ; 6654  =O
U 2276, 0001,3771,0003,1276,6004,1701,0000,0000,1443    ; 6655  GSRC:   [AR]_AC[DLEN],          ;LENGTH RAN OUT
U 2277, 3446,3771,0003,1276,6007,0701,0000,0000,1441    ; 6656          RETURN [1]              ;RESTORE AR AND RETURN
                                                ; 6657  GETSRC: [AR]_AC[SRCP]          ;GET SRC PTR
                                                ; 6658          IBP DP, IBP SCAD,       ;UPDATE BYTE POINTER
U 3446, 0231,3770,0305,4334,4016,7701,0000,0033,6000    ; 6659          SCAD DISP, 3T           ;SEE IF OFLOW
U 0231, 3450,3441,0503,4174,4007,0700,0000,0000,0000    ; 6660  =O1     [AR]_[BR], J/GSRC1      ;NO OFLOW
U 0233, 3447,3770,0503,4334,4017,0700,0000,0032,6000    ; 6661          SET P TO 36-S           ;RESET P
U 3447, 3450,0111,0703,4170,4007,0700,0000,0000,0000    ; 6662          [AR]_[AR]+1, HOLD LEFT  ;BUMP Y
                                                ; 6663
U 3450, 2300,3440,0303,1174,4007,0700,0400,0000,1441    ; 6664  GSRC1:  AC[SRCP]_[AR]           ;STORE UPDATED POINTER
                                                ; 6665  =O      READ [AR], LOAD BYTE EA,;SETUP TO FIGURE OUT
U 2300, 3077,3333,0003,4174,4217,0701,1010,0073,0500    ; 6666          FE_P, 3T, CALL [BYTEAS] ; EFFECTIVE ADDRESS
                                                ; 6667          READ [AR],              ;LOOK AT POINTER
                                                ; 6668          BYTE DISP,              ;SEE IF 7 BIT
                                                ; 6669          FE_FE.AND.S#, S#/0770,  ;MASK OUT P FIELD
U 2301, 0340,3333,0003,4174,4006,5701,1000,0051,0770    ; 6670          J/LDB1                  ;GO GET THE BYTE
                                                ; 6671
```

```
U 3451, 2302,3441,0304,4174,4007,0700,0000,0000,0000

U 2302, 3452,3771,0003,1276,6007,0701,0010,0000,1444


U 2303, 2245,0113,0404,4174,4007,0701,1000,0077,0000

U 2245, 0360,3333,0003,4174,4006,5701,0010,0000,0000
U 2247, 0006,4443,0000,4174,4004,1700,0000,0000,0000
```

```
; 6672  .TOC    "EXTEND SUBROUTINES -- STORE BYTE IN DESTINATION STRING"
; 6673
; 6674  ;CALL WITH:
; 6675  ;       AR/ BYTE TO STORE
; 6676  ;       AC4/ DESTINATION BYTE POINTER
; 6677  ;RETURNS:
; 6678  ;       AR & AC4/ UPDATED BYTE POINTER
; 6679  ;       ARX/ BYTE TO STORE
; 6680  ;       BR/ WORD TO MERGE WITH
; 6681  ;       6 ALWAYS
; 6682  ;
; 6683  PUTDST: [ARX]_[AR]              ;SAVE BYTE
; 6684  =O      [AR]_AC[DSTP],          ;GET DEST POINTER
; 6685          CALL [IDST]             ;BUMP DEST POINTER
; 6686          AD/A+B, A/ARX, B/ARX,   ;SHIFT 7-BIT BYTE TO
; 6687          SCAD/A, 3T,             ; NATURAL PLACE, AND PUT
; 6688          SCADA/BYTE5, LOAD FE    ; INTO FE
; 6689  =O*     READ [AR], BYTE DISP,   ;GO PUT BYTE IN MEMORY
; 6690          CALL [DPB1]             ; ..
; 6691          RETURN [6]              ;ALL DONE
; 6692
```

```
                                                ; 6693  .TOC    "EXTEND SUBROUTINES -- UPDATE DEST STRING POINTERS"
                                                ; 6694
                                                ; 6695
                                                ; 6696  ;SUBROUTINE TO BUMP DST POINTERS
                                                ; 6697  ;CALL WITH:
                                                ; 6698  ;       AR/     AC[DSTP]
                                                ; 6699  ;       RETURN 1 WITH UPDATED POINTER STORED
                                                ; 6700  ;
U 3452, 2304,3770,0305,4334,4016,7701,0000,0033,6000    ; 6701  IDST:   IBP DP, IBP SCAD, SCAD DISP, 3T
U 2304, 3454,3441,0503,4174,4217,0700,0000,0000,0600    ; 6702  =0*     [AR]_[BR], LOAD DST EA, J/IDSTX
U 2306, 3453,3770,0503,4334,4017,0700,0000,0032,6000    ; 6703          SET P TO 36-S
U 3453, 3454,0111,0703,4170,4217,0700,0000,0000,0600    ; 6704          [AR]_[AR]+1, HOLD LEFT, LOAD DST EA
                                                ; 6705  IDSTX:  AC[DSTP]_[AR], 3T,      ;STORE PTR BACK
U 3454, 0230,3440,0303,1174,4006,6701,1400,0073,1444    ; 6706          FE_P, DISP/EAMODE       ;SAVE P FOR CMPDST
                                                ; 6707  =100*
U 0230, 3102,0553,0300,2274,4007,0701,0200,0004,0712    ; 6708  DSTEA:  VMA_[AR]+XR, START READ, PXCT BYTE DATA, 3T, J/BYTFET
U 0232, 3102,3443,0300,4174,4007,0700,0200,0004,0712    ; 6709          VMA_[AR], START READ, PXCT BYTE DATA, J/BYTFET
U 0234, 3455,0553,0300,2274,4007,0701,0200,0004,0612    ; 6710          VMA_[AR]+XR, START READ, PXCT/BIS-DST-EA, 3T, J/DSTIND
U 0236, 3455,3443,0300,4174,4007,0700,0200,0004,0612    ; 6711          VMA_[AR], START READ, PXCT/BIS-DST-EA, J/DSTIND
                                                ; 6712
U 3455, 3456,3771,0003,4361,5217,0700,0200,0000,0602    ; 6713  DSTIND: MEM READ, [AR]_MEM, HOLD LEFT, LOAD DST EA
U 3456, 0230,4443,0000,2174,4006,6700,0000,0000,0000    ; 6714          EA MODE DISP, J/DSTEA
                                                ; 6715
                                                ; 6716
                                                ; 6717  ;HERE TO TEST ILLEGAL BITS SET
                                                ; 6718  ;CALL WITH:
                                                ; 6719  ;       SKIP IF ALL BITS LEGAL
                                                ; 6720  ;       RETURN [4] IF OK, ELSE DO UUO
                                                ; 6721  ;
                                                ; 6722  3556:           ;EXTEND OF 0 COMES HERE
U 3556, 2740,4551,0202,4374,0007,0700,0000,0077,7740    ; 6723  BITCHK: UUO
U 3557, 0004,4443,0000,4174,4004,1700,0000,0000,0000    ; 6724  3557:   RETURN [4]
                                                ; 6725
                                                ; 6726  ;HERE TO PUT FILL IN [AR] AND WORK[FILL]
                                                ; 6727  GTFILL: MEM READ,               ;WAIT FOR DATA
U 3457, 3460,3771,0003,4365,5007,0700,0200,0000,0002    ; 6728          [AR]_MEM                ;PLACE IN AR
                                                ; 6729          WORK[FILL]_[AR],        ;SAVE FOR LATER
U 3460, 0010,3333,0003,7174,4004,1700,0400,0000,0244    ; 6730          RETURN [10]             ;RETURN TO CALLER
                                                ; 6731
                                                ; 6732  ;SUBROUTINE TO CLEAR FLAGS IN AR
                                                ; 6733  CLRFLG: [AR]_[AR].AND.#,        ;CLEAR FLAGS IN AR
                                                ; 6734          #/000777,               ; ..
U 3461, 0001,4551,0303,4374,0004,1700,0000,0000,0777    ; 6735          HOLD RIGHT, RETURN [1]
                                                ; 6736
```

```
U 2310, 3474,3771,0003,1276,6007,0701,0010,0000,1441
U 2311, 2676,3440,0505,1174,4007,0700,0400,0000,1441

U 3462, 3463,3771,0003,7274,4007,0701,0000,0000,0214
U 3463, 2312,0113,0703,0174,4007,0701,0400,0000,0000



U 2312, 3474,3771,0003,1276,6007,0701,0010,0000,1444
U 2313, 2676,3440,0505,1174,4007,0700,0400,0000,1444


U 3464, 3465,1771,0003,7274,4007,0701,4000,0000,0242

U 3465, 2314,3771,0005,1276,6007,0522,0000,0000,1443

U 2314, 3467,3440,0303,1174,4007,0700,0400,0000,1443
U 2315, 2316,3441,0304,4174,4007,0700,0000,0000,0000

U 2316, 3672,3551,0404,7274,4007,0701,0010,0000,0214
U 2317, 3466,1111,0503,4174,4007,0700,4000,0000,0000

U 3466, 2676,3440,0303,1174,4007,0700,0400,0000,1443

U 3467, 3471,0111,0503,4174,4007,0700,0000,0000,0000

U 3470, 3471,1771,0003,7274,4007,0701,4000,0000,0242
U 3471, 3472,3551,0303,7274,4007,0701,0000,0000,0214
U 3472, 2676,3440,0303,0174,4007,0700,0400,0000,0000

U 3473, 3465,7771,0003,7274,4007,0701,0000,0000,0242


U 3474, 0001,3770,0305,4334,4014,1700,0000,0043,6000
```

```
; 6737  .TOC    "EXTEND -- PAGE FAIL CLEANUP"
; 6738
; 6739  ;BACK UP SOURCE POINTER
; 6740  =O
; 6741  BACKS:  [AR]_AC[SRCP],
; 6742          CALL [BACKBP]           ;BACKUP BP
; 6743          AC[SRCP]_[BR], J/CLDISP
; 6744
; 6745  CMSDST: [AR]_WORK[SV.BRX]       ;GET OLD SRC LEN
; 6746          AC_[AR]+1, 3T           ;BACK UP
; 6747  ;BACK UP DESTINATION POINTER
; 6748  =O
; 6749  BACKD:  [AR]_AC[DSTP],
; 6750          CALL [BACKBP]
; 6751          AC[DSTP]_[BR], J/CLDISP
; 6752
; 6753  ;FAILURES DURING MOVE STRING (BACKUP LENGTHS)
; 6754  STRPF:  [AR]_-WORK[SLEN]        ;GET AMOUNT LEFT
; 6755  STRPFO: [BR]_AC[DLEN], 4T,      ;WHICH STRING IS LONGER?
; 6756          SKIP DPO
; 6757  =O
; 6758  STRPF1: AC[DLEN]_[AR], J/STPF1A ;SRC LONGER
; 6759          [ARX]_[AR]              ;COPY SRC LENGTH
; 6760  =O      [ARX]_[ARX].OR.WORK[SV.BRX], ;REBUILD FLAGS
; 6761          CALL [AC_ARX]           ;RESET AC]SLEN]
; 6762          [AR]_[AR]-[BR]          ;MAKE DEST LEN
; 6763  STRPF3: AC[DLEN]_[AR],          ;PUT BACK DEST LEN
; 6764          J/CLDISP                ;DO NEXT CLEANUP
; 6765
; 6766  STPF1A: [AR]_[AR]+[BR], J/STRPF2
; 6767
; 6768  PFDBIN: [AR]_-WORK[SLEN]        ;RESTORE LENGTH
; 6769  STRPF2: [AR]_[AR].OR.WORK[SV.BRX]
; 6770  PFGACO: AC_[AR], J/CLDISP       ;PUT BACK SRC LEN AND FLAGS
; 6771
; 6772  STRPF4: [AR]_.NOT.WORK[SLEN], J/STRPFO
; 6773
; 6774  BACKBP: IBP DP, SCAD/A+B, SCADA/BYTE1, SCADB/SIZE, ;P_P+S
; 6775          RETURN [1]
; 6776
```

```
                                                ; 6777          .TOC    "TRAPS"
                                                ; 6778
U 3475, 3476,3741,0104,4074,4007,0700,0000,0000,0000    ; 6779  TRAP:   [ARX]_PC WITH FLAGS     ;SAVE THE PC WHICH CAUSED THE
                                                ; 6780          WORK[TRAPPC]_[ARX],     ; TRAP
U 3476, 2320,3333,0004,7174,4007,0340,0400,0000,0425    ; 6781          SKIP KERNEL             ;SEE IF UBR OR EBR
                                                ; 6782  =O      [AR]_[AR]+[UBR],        ;ADDRESS OF INSTRUCTION
                                                ; 6783          MEM READ,               ;WAIT FOR PREFETCH TO GET INTO
                                                ; 6784                                  ; THE CACHE. MAY PAGE FAIL BUT
                                                ; 6785                                  ; THAT IS OK
                                                ; 6786          START READ,             ;START FETCH
                                                ; 6787          VMA PHYSICAL,           ;ABSOLUTE ADDRESSING
U 2320, 3477,0111,1103,4364,4007,0700,0200,0024,1016    ; 6788          J/TRP1                  ;JOIN COMMON CODE
                                                ; 6789
                                                ; 6790          [AR]_[AR]+[EBR],        ;WE COME HERE IN EXEC MODE
                                                ; 6791          MEM READ,               ;WAIT FOR PREFETCH TO GET INTO
                                                ; 6792                                  ; THE CACHE. MAY PAGE FAIL BUT
                                                ; 6793                                  ; THAT IS OK
                                                ; 6794          START READ,             ;START FETCH
                                                ; 6795          VMA PHYSICAL,           ;ABSOLUTE ADDRESSING
U 2321, 3477,0111,1003,4364,4007,0700,0200,0024,1016    ; 6796          J/TRP1                  ;JOIN COMMON CODE
                                                ; 6797
                                                ; 6798  TRP1:   MEM READ, [HR]_MEM,     ;PLACE INSTRUCTION IN HR
U 3477, 3500,3771,0002,4365,5617,0700,0200,0000,0002    ; 6799          LOAD INST               ;LOAD IR, XR, @
                                                ; 6800          [HR].AND.#,             ;TEST TO SEE IF THIS
                                                ; 6801          #/700000, 3T,           ; IS A UUO
U 3500, 2322,4553,0200,4374,4007,0321,0000,0070,0000    ; 6802          SKIP ADL.EQ.0
                                                ; 6803  =O      CHANGE FLAGS,           ;NOT A UUO
                                                ; 6804          HOLD USER/1,            ;CLEAR TRAP FLAGS
U 2322, 2717,4443,0000,4174,4467,0700,0000,0001,0000    ; 6805          J/XCT1                  ;DO THE INSTRUCTION
U 2323, 2740,4551,0202,4374,0007,0700,0000,0077,7740    ; 6806          UUO                     ;DO THE UUO
                                                ; 6807
```

```
D 0700, 1200,1700,4100
D 0701, 1200,1720,4100

U 1701, 2740,4551,0202,4374,0007,0700,0000,0077,7740
U 1702, 2740,4551,0202,4374,0007,0700,0000,0077,7740
U 1703, 2740,4551,0202,4374,0007,0700,0000,0077,7740
U 1706, 3503,3771,0005,4304,4007,0701,0000,0000,0000
U 1707, 3501,3771,0005,4304,4007,0701,0000,0000,0000

U 1710, 2740,4551,0202,4374,0007,0700,0000,0077,7740
U 1711, 2740,4551,0202,4374,0007,0700,0000,0077,7740
U 1712, 2740,4551,0202,4374,0007,0700,0000,0077,7740
U 1713, 2740,4551,0202,4374,0007,0700,0000,0077,7740
U 1716, 3504,3441,1405,4174,4007,0700,0000,0000,0000
U 1717, 3502,3441,1405,4174,4007,0700,0000,0000,0000

U 1720, 2740,4551,0202,4374,0007,0700,0000,0077,7740
U 1726, 2740,4551,0202,4374,0007,0700,0000,0077,7740
U 1727, 2740,4551,0202,4374,0007,0700,0000,0077,7740

U 1730, 2740,4551,0202,4374,0007,0700,0000,0077,7740
U 1731, 2740,4551,0202,4374,0007,0700,0000,0077,7740
U 1732, 2740,4551,0202,4374,0007,0700,0000,0077,7740
U 1733, 2740,4551,0202,4374,0007,0700,0000,0077,7740
U 1734, 2740,4551,0202,4374,0007,0700,0000,0077,7740
U 1735, 2740,4551,0202,4374,0007,0700,0000,0077,7740
U 1736, 2740,4551,0202,4374,0007,0700,0000,0077,7740
U 1737, 2740,4551,0202,4374,0007,0700,0000,0077,7740

U 3501, 3502,4251,0505,4374,4007,0700,0000,0000,7770
U 3502, 0260,4113,0305,4174,4007,0330,0000,0000,0000

U 3503, 3504,4251,0505,4374,4007,0700,0000,0000,7770
U 3504, 1400,4113,0305,4174,4007,0330,0000,0000,0000
```

```
; 6808          .TOC    "IO -- INTERNAL DEVICES"
; 6809
; 6810          .DCODE
; 6811  700:    IOT,AC DISP,    J/GRP700
; 6812          IOT,AC DISP,    J/GRP701
; 6813          .UCODE
; 6814
; 6815  1701:   UUO             ;DATAI APR,
; 6816  1702:   UUO             ;BLKO APR,
; 6817  1703:   UUO             ;DATAO APR,
; 6818  1706:   [BR]_APR, J/APRSZ ;CONSZ APR,
; 6819  1707:   [BR]_APR, J/APRSO ;CONSO APR,
; 6820  1710:
; 6821  RDERA:  UUO             ;BLKI PI,
; 6822  1711:   UUO             ;DATAI PI,
; 6823  1712:   UUO             ;BLKO PI,
; 6824  1713:   UUO             ;DATAO PI,
; 6825  1716:   [BR]_[PI], J/CONSZ ;CONSZ PI,
; 6826  1717:   [BR]_[PI], J/CONSO ;CONSO PI,
; 6827
; 6828  1720:
; 6829  GRP701: UUO             ;BLKI PAG,
; 6830  1726:   UUO             ;CONSZ PAG,
; 6831  1727:   UUO             ;CONSO PAG,
; 6832
; 6833  ;680I AND CACHE SWEEP STUFF
; 6834  1730:   UUO
; 6835  1731:   UUO
; 6836  1732:   UUO
; 6837  1733:   UUO
; 6838  1734:   UUO
; 6839  1735:   UUO
; 6840  1736:   UUO
; 6841  1737:   UUO
; 6842
; 6843  APRSO:  [BR]_[BR].AND.# CLR LH, #/7770
; 6844  CONSO:  [BR].AND.[AR], SKIP ADR.EQ.0, J/SKIP
; 6845
; 6846  APRSZ:  [BR]_[BR].AND.# CLR LH, #/7770
; 6847  CONSZ:  [BR].AND.[AR], SKIP ADR.EQ.0, J/DONE
; 6848
```

```
U 1700, 0137,3771,0005,4374,4007,0700,0000,0001,0001

U 0137, 3614,3771,0005,4374,0007,0700,0000,0000,0123

U 1704, 3505,3771,0005,7274,4007,0701,0000,0000,0230

U 3505, 3506,5551,0505,4370,4007,0700,0000,0000,0007
U 3506, 3507,4551,0304,4374,4007,0700,0000,0000,0007
U 3507, 3510,3111,0405,4174,4007,0700,0000,0000,0000

U 3510, 3511,4551,0304,4374,4007,0700,0000,0000,7760
U 3511, 2324,4553,0300,4374,4007,0331,0000,0010,0000
U 2324, 2325,3111,0405,4174,4007,0700,0000,0000,0000
U 2325, 2326,4553,0300,4374,4007,0331,0000,0004,0000
U 2326, 2327,5111,0405,4174,4007,0700,0000,0000,0000
U 2327, 3512,3771,0006,4304,4007,0701,0000,0000,0000
U 3512, 2330,4553,0300,4374,4007,0331,0000,0002,0000
U 2330, 2331,5111,0406,4174,4007,0700,0000,0000,0000
U 2331, 2332,4553,0300,4374,4007,0331,0000,0001,0000
U 2332, 2333,3111,0406,4174,4007,0700,0000,0000,0000
U 2333, 2334,4553,0300,4374,4007,0331,0000,0003,0000

U 2334, 3515,3333,0006,4174,4007,0700,0000,0000,0000
U 2335, 3513,3333,0005,4174,4007,0700,0000,0000,0000

U 3513, 3514,3333,0005,4174,4257,0700,0000,0000,0000

U 3514, 1400,3333,0005,7174,4007,0700,0400,0000,0230

U 3515, 3516,3333,0006,4174,4237,0700,0000,0000,0000

U 3516, 3517,5551,0606,4370,4007,0700,0000,0000,2000

U 3517, 2335,3333,0006,4174,4237,0700,0000,0000,0000
```

```
; 6849  1700:
; 6850  GRP700:
; 6851  APRID:  [BR]_#,
; 6852          #/4097.
; 6853  137:    [BR]_#,
; 6854          MICROCODE OPTIONS/OPT,
; 6855          MICROCODE VERSION/UCV,
; 6856          HOLD RIGHT,
; 6857          J/RTNREG
; 6858
; 6859  1704:
; 6860  WRAPR:  [BR]_WORK[APR]
; 6861          [BR]_[BR].AND.NOT.#,     ;CLEAR THE OLD PIA
; 6862          #/7, HOLD LEFT           ; ..
; 6863          [ARX]_[AR].AND.#, #/7    ;PUT NEW PIA IN ARX
; 6864          [BR]_[BR].OR.[ARX]       ;PUT NEW PIA IN BR
; 6865          [ARX]_[AR].AND.#,        ;MASK THE DATA BITS
; 6866          #/007760                 ; DOWN TO ENABLES
; 6867          TR [AR], #/100000        ;WANT TO ENABLE ANY?
; 6868  =0      [BR]_[BR].OR.[ARX]       ;YES--SET THEM
; 6869          TR [AR], #/40000         ;WANT TO DISABLE ANY?
; 6870  =0      [BR]_[BR].AND.NOT.[ARX]  ;YES--CLEAR THEM
; 6871          [BRX]_APR                ;GET CURRENT STATUS
; 6872          TR [AR], #/20000         ;WANT TO CLEAR FLAGS?
; 6873  =0      [BRX]_[BRX].AND.NOT.[ARX] ;YES--CLEAR BITS
; 6874          TR [AR], #/10000         ;WANT TO SET ANY FLAGS?
; 6875  =0      [BRX]_[BRX].OR.[ARX]     ;YES--SET FLAGS
; 6876          TR [AR], #/30000         ;ANY CHANGE AT ALL?
; 6877  =0      READ [BRX],              ;YES--LOAD NEW FLAGS
; 6878          J/WRAPR2                 ;TURN OFF INTERRUPT 8080
; 6879  WRAPR1: READ [BR]                ;FIX DPM TIMING BUG
; 6880          READ [BR],               ;ENABLE CONDITIONS
; 6881          SET APR ENABLES
; 6882          WORK[APR]_[BR],          ;SAVE FOR RDAPR
; 6883          J/DONE                   ;ALL DONE
; 6884
; 6885  WRAPR2: READ [BRX],              ;LOAD NEW FLAGS
; 6886          SPEC/APR FLAGS           ; ..
; 6887          [BRX]_[BRX].AND.NOT.#,   ;CLEAR INTERRUPT THE 8080
; 6888          #/002000, HOLD LEFT      ; FLAG
; 6889          READ [BRX],              ;LOAD NEW FLAGS
; 6890          SPEC/APR FLAGS,          ; ..
; 6891          J/WRAPR1                 ;LOOP BACK
; 6892
```

```
U 1705, 3520,3771,0005,7274,4007,0701,0000,0000,0230
U 3520, 3521,3770,0505,4344,0007,0700,0000,0000,0000
U 3521, 3522,4551,0505,4374,0007,0700,0000,0000,7760
U 3522, 3523,4551,0505,4370,4007,0700,0000,0000,0007
U 3523, 3524,3771,0004,4304,4007,0701,0000,0000,0000
U 3524, 3525,4251,0404,4374,4007,0700,0000,0000,7770
U 3525, 3614,3111,0405,4174,4007,0700,0000,0000,0000
```

```
; 6893  1705:
; 6894  RDAPR:  [BR]_WORK[APR]
; 6895          [BR]_[BR] SWAP,          ;PUT ENABLES IN BOTH
; 6896          HOLD RIGHT               ; HALVES
; 6897          [BR]_[BR].AND.#,         ;SAVE ENABLES IN LH
; 6898          #/7760,                  ;
; 6899          HOLD RIGHT
; 6900          [BR]_[BR].AND.#,         ;SAVE PIA IN RH
; 6901          #/7,
; 6902          HOLD LEFT
; 6903          [ARX]_APR                ;READ THE APR FLAGS
; 6904          [ARX]_[ARX].AND.# CLR LH, ;MASK OUT JUNK
; 6905          #/007770                 ;KEEP 8 FLAGS
; 6906          [BR]_[BR].OR.[ARX],      ;MASH THE STUFF TOGETHER
; 6907          J/RTNREG                 ;RETURN
; 6908
```

```
U 1723, 3526,3443,0300,4174,4007,0700,0200,0004,0012
U 3526, 2336,3771,0003,4365,5007,0521,0200,0000,0002
U 2336, 2340,4551,0303,4374,0007,0321,0000,0010,0000
U 2337, 3527,4551,1111,4374,0007,0700,0000,0077,0077
U 3527, 2340,4553,0300,4374,4007,0321,0000,0010,0000
U 2340, 3530,4551,0305,4374,4007,0700,0000,0001,7777
U 2341, 1400,3111,0311,4174,0477,0700,0000,0000,0000
U 3530, 2342,4221,0005,4174,0007,0700,2000,0071,0007
U 2342, 2342,3445,0505,4174,4007,0630,2000,0060,0000
U 2343, 3531,4551,1111,4374,0007,0700,0000,0077,7774
U 3531, 3532,4221,0011,4170,4007,0700,0000,0000,0000
U 3532, 3533,3111,0511,4174,4007,0700,0000,0000,0000
U 3533, 2410,3111,0311,4174,0477,0700,0000,0000,0000
```

```
; 6909  .TOC    "IO -- INTERNAL DEVICES -- EBR & UBR"
; 6910
; 6911  1723:
; 6912  WRUBR:  VMA_[AR],                ;LOAD E INTO VMA
; 6913          START READ               ;START MEMORY
; 6914          MEM READ,                ;WAIT FOR DATA
; 6915          [AR]_MEM, 3T,            ;PUT IT INTO THE AR
; 6916          SKIP DPO                 ;SEE IF WE WANT TO LOAD
; 6917                                   ; AC BLOCK NUMBERS
; 6918  =O      [AR]_[AR].AND.#,         ;NO--CLEAR JUNK IN AR
; 6919          #/100000,                ; LEAVE ONLY LOAD UBR
; 6920          HOLD RIGHT,              ; IN LEFT HALF
; 6921          SKIP ADL.EQ.O, 3T,               ;SEE IF WE WANT TO LOAD
; 6922          J/ACBSET                 ;SKIP AROUND UBR LOAD
; 6923          [UBR]_[UBR].AND.#,       ;MASK OUT THE OLD
; 6924          #/770077,                ; AC BLOCK NUMBERS
; 6925          HOLD RIGHT               ;IN THE LEFT HALF
; 6926          [AR].AND.#,              ;SEE IF WE WANT TO LOAD
; 6927          #/100000, 3T,            ; UBR ALSO
; 6928          SKIP ADL.EQ.O
; 6929  =O
; 6930  ACBSET: [BR]_[AR].AND.#,         ;COPY UBR PAGE NUMBER
; 6931          #/17777,                 ; INTO BR
; 6932          J/SETUBR                 ;GO LOAD UBR
; 6933          [UBR]_[UBR].OR.[AR],     ;DO NOT LOAD UBR
; 6934                                   ; PUT AC BLOCK # IN
; 6935          HOLD RIGHT,              ; THE LEFT HALF
; 6936          LOAD AC BLOCKS,          ;LOAD HARDWARE
; 6937          J/DONE                   ;ALL DONE
; 6938  SETUBR: [BR]_0,                  ;CLEAR BR LEFT
; 6939          SC_7,                    ;PUT THE COUNT IN SC
; 6940          HOLD RIGHT
; 6941  =O
; 6942  STUBRS: [BR]_[BR]*2,             ;SHIFT BR OVER
; 6943          STEP SC,                 ; 9 PLACES
; 6944          J/STUBRS
; 6945          [UBR]_[UBR].AND.#,       ;MASK OUT OLD UBR
; 6946          #/777774,                ; BITS IN
; 6947          HOLD RIGHT               ; LEFT HALF
; 6948          [UBR]_0,                 ;CLEAR RIGHT HALF
; 6949          HOLD LEFT
; 6950          [UBR]_[UBR].OR.[BR]      ;PUT IN PAGE TABLE ADDRESS
; 6951          [UBR]_[UBR].OR.[AR],     ;PUT IN AC BLOCK #
; 6952          HOLD RIGHT,              ; IN LEFT HALF
; 6953          LOAD AC BLOCKS,          ;TELL HARDWARE
; 6954          J/SWEEP                  ;CLEAR CACHE
; 6955
```

```
U 1724, 2344,3445,0303,4174,4007,0700,2000,0071,0006
U 2344, 2344,3445,0303,4174,4007,0630,2000,0060,0000

U 2345, 3534,3771,0005,7274,4007,0701,0000,0000,0230
U 3534, 3535,4551,0505,4370,4007,0700,0000,0074,7777
U 3535, 2346,4553,0300,4374,4007,0321,0000,0000,0020
U 2346, 2347,3551,0505,4370,4007,0700,0000,0003,0000
U 2347, 3536,3333,0005,4174,4257,0700,0000,0000,0000
U 3536, 3537,3333,0005,7174,4007,0700,0400,0000,0230

U 3537, 3540,3441,0310,4174,4007,0700,0000,0000,0000
U 3540, 2350,4553,1000,4374,4007,0321,0000,0000,0040
U 2350, 2410,3551,1010,4374,0007,0700,0000,0040,0000
U 2351, 2410,5551,1010,4374,0007,0700,0000,0040,0000

U 1725, 2352,3447,1005,4174,4007,0700,2000,0071,0006

U 2352, 2352,3447,0505,4174,4007,0630,2000,0060,0000
U 2353, 3541,4551,0505,4374,4007,0700,0000,0006,3777

U 3541, 3614,4221,0005,4174,0007,0700,0000,0000,0000

U 1721, 2354,3441,1105,4174,4007,0700,0000,0000,0000
U 2354, 2356,3447,0506,4174,4007,0700,2010,0071,0006

U 2355, 3614,4551,0505,4374,0007,0700,0000,0050,7700

U 3542, 3543,3441,1105,4174,4007,0700,0000,0000,0000
U 3543, 2356,3447,0506,4174,4007,0700,2000,0071,0006

U 2356, 2356,3447,0606,4174,4007,0630,2000,0060,0000
U 2357, 3544,4551,0606,4374,4007,0700,0000,0001,7777
U 3544, 0001,3441,0605,4170,4004,1700,0000,0000,0000
```

```
; 6956  1724:
; 6957  WREBR:  [AR]_[AR]*2, SC_6
; 6958  =0
; 6959  WREBR1: [AR]_[AR]*2, STEP SC, J/WREBR1
; 6960  .IF/FULL                          ;DO NOT ENABLE PAGING IN SMALL
; 6961                                   ; MICROCODE.
; 6962          [BR]_WORK[APR]
; 6963          [BR]_[BR].AND.#, #/747777, HOLD LEFT
; 6964          [AR].AND.#, #/20, 3T, SKIP ADL.EQ.0
; 6965  =0      [BR]_[BR].OR.#, #/030000, HOLD LEFT
; 6966          READ [BR], SET APR ENABLES
; 6967          WORK[APR]_[BR]
; 6968  .ENDIF/FULL
; 6969          [EBR]_[AR]
; 6970          [EBR].AND.#, #/40, 3T, SKIP ADL.EQ.0
; 6971  =0      [EBR]_[EBR].OR.#, #/400000, HOLD RIGHT, J/SWEEP
; 6972          [EBR]_[EBR].AND.NOT.#, #/400000, HOLD RIGHT, J/SWEEP
; 6973
; 6974  1725:
; 6975  RDEBR:  [BR]_[EBR]*.5, SC_6
; 6976  =0
; 6977  RDEBR1: [BR]_[BR]*.5, STEP SC, J/RDEBR1
; 6978          [BR]_[BR].AND.#, #/63777 ;MASK TO JUST EBR
; 6979          [BR]_0,                  ;CLEAR LEFT HALF
; 6980          HOLD RIGHT,              ; BITS
; 6981          J/RTNREG                 ;RETURN ANSWER
; 6982
; 6983  1721:
; 6984  RDUBR:  [BR]_[UBR]
; 6985  =0      [BRX]_[BR]*.5, SC_6, CALL [GTPCW1]
; 6986          [BR]_[BR].AND.#,         ;JUST RETURN USEFUL
; 6987          #/507700, HOLD RIGHT,    ; BITS
; 6988          J/RTNREG
; 6989
; 6990
; 6991  GETPCW: [BR]_[UBR]
; 6992          [BRX]_[BR]*.5, SC_6
; 6993  =0
; 6994  GTPCW1: [BRX]_[BRX]*.5, STEP SC, J/GTPCW1
; 6995          [BRX]_[BRX].AND.#, #/17777
; 6996          [BR]_[BRX], HOLD LEFT, RETURN [1]
; 6997
```

```
D 0702, 1216,1760,4700

U 1760, 3614,3771,0005,7274,4007,0701,0000,0000,0215
U 1761, 3614,3771,0005,7274,4007,0701,0000,0000,0216
U 1762, 3614,3771,0005,7274,4007,0701,0000,0000,0220
U 1763, 3614,3771,0005,7274,4007,0701,0000,0000,0217
U 1766, 3614,3771,0005,7274,4007,0701,0000,0000,0227
U 1767, 2740,4551,0202,4374,0007,0700,0000,0077,7740

U 1770, 3545,4443,0000,4174,4007,0700,0200,0004,0002
U 3545, 3546,3771,0003,4365,5007,0700,0200,0000,0002
U 3546, 1400,3333,0003,7174,4007,0700,0400,0000,0215

U 1771, 3547,4443,0000,4174,4007,0700,0200,0004,0002
U 3547, 3550,3771,0003,4365,5007,0700,0200,0000,0002
U 3550, 1400,3333,0003,7174,4007,0700,0400,0000,0216

U 1772, 3551,4443,0000,4174,4007,0700,0200,0004,0002
U 3551, 3552,3771,0003,4365,5007,0700,0200,0000,0002
U 3552, 1400,3333,0003,7174,4007,0700,0400,0000,0220

U 1773, 3553,4443,0000,4174,4007,0700,0200,0004,0002
U 3553, 3554,3771,0003,4365,5007,0700,0200,0000,0002
U 3554, 1400,3333,0003,7174,4007,0700,0400,0000,0217

U 1776, 3555,4443,0000,4174,4007,0700,0200,0004,0002
U 3555, 3560,3771,0003,4365,5007,0700,0200,0000,0002
U 3560, 1400,3333,0003,7174,4007,0700,0400,0000,0227
U 1777, 2740,4551,0202,4374,0007,0700,0000,0077,7740
```

```
; 6998  .TOC    "IO -- INTERNAL DEVICES -- KL PAGING REGISTERS"
; 6999
; 7000          .DCODE
; 7001  702:    IOT,AC DISP,    M,      J/GRP702
; 7002          .UCODE
; 7003
; 7004  1760:
; 7005  GRP702:
; 7006  RDSPB:  [BR]_WORK[SBR], J/RTNREG
; 7007  1761:
; 7008  RDCSB:  [BR]_WORK[CBR], J/RTNREG
; 7009  1762:
; 7010  RDPUR:  [BR]_WORK[PUR], J/RTNREG
; 7011  1763:
; 7012  RDCSTM: [BR]_WORK[CSTM], J/RTNREG
; 7013  1766:
; 7014  RDHSB:  [BR]_WORK[HSBADR], J/RTNREG
; 7015  1767:   UUO
; 7016
; 7017  1770:
; 7018  WRSPB:  START READ
; 7019          MEM READ, [AR]_MEM
; 7020          WORK[SBR]_[AR], J/DONE
; 7021  1771:
; 7022  WRCSB:  START READ
; 7023          MEM READ, [AR]_MEM
; 7024          WORK[CBR]_[AR], J/DONE
; 7025  1772:
; 7026  WRPUR:  START READ
; 7027          MEM READ, [AR]_MEM
; 7028          WORK[PUR]_[AR], J/DONE
; 7029  1773:
; 7030  WRCSTM: START READ
; 7031          MEM READ, [AR]_MEM
; 7032          WORK[CSTM]_[AR], J/DONE
; 7033  1776:
; 7034  WRHSB:  START READ
; 7035          MEM READ, [AR]_MEM
; 7036          WORK[HSBADR]_[AR], J/DONE
; 7037  1777:   UUO
; 7038
```

```
U 3561, 3562,3771,0003,7274,4117,0701,0000,0000,0301

U 3562, 3563,4751,1205,4374,4007,0700,0000,0001,0000
U 3563, 3564,0111,0503,4174,4007,0700,0000,0000,0000
U 3564, 2360,3770,0303,4174,0007,0520,0000,0000,0000


U 2360, 3565,3333,0003,7174,4007,0700,0400,0000,0301
U 2361, 2305,3771,0003,7274,4007,0701,0000,0000,0300

U 2305, 3573,0111,0703,4174,4007,0700,0010,0000,0000

U 2307, 2360,4221,0003,4174,4007,0700,0000,0000,0000
U 3565, 3566,3771,0003,7274,4007,0701,0000,0000,0303

U 3566, 2362,1111,0503,4174,4007,0421,4000,0000,0000


U 2362, 0002,3333,0003,7174,4004,1700,0400,0000,0303
U 2363, 3567,3771,0003,7274,4007,0701,0000,0000,0302
U 3567, 3570,3771,0005,4304,4007,0701,0000,0000,0000
U 3570, 3571,3551,0505,4374,4007,0700,0000,0000,0040


U 3571, 2362,3333,0005,4174,4237,0700,0000,0000,0000
```

```
; 7039  .TOC    "IO -- INTERNAL DEVICES -- TIMER CONTROL"
; 7040
; 7041
; 7042                                        ;BEGIN [123]
; 7043  TICK:   [AR]_WORK[TIME1],             ;GET LOW WORD
; 7044          SPEC/CLRCLK                   ;CLEAR CLOCK FLAG
; 7045                                        ;END [123]
; 7046  TOCK:   [BR]_0 XWD [10000]            ;2^12 UNITS PER MS
; 7047          [AR]_[AR]+[BR]                ;INCREMENT THE TIMER
; 7048          FIX [AR] SIGN, SKIP DPO       ;SEE IF IT OVERFLOWED
; 7049  =0
; 7050  TOCK1:  WORK[TIME1]_[AR],             ;STORE THE NEW TIME
; 7051          J/TOCK2                       ;SKIP OVER THE OVERFLOW CODE
; 7052          [AR]_WORK[TIME0]               ;GET HIGH WORD
; 7053  =0*     [AR]_[AR]+1,                  ;BUMP IT
; 7054          CALL [WRTIM1]                 ;STORE BACK IN RAM
; 7055          [AR]_0,                       ;CAUSE LOW WORD WORD
; 7056          J/TOCK1                       ; TO GET STORED
; 7057  TOCK2:  [AR]_WORK[TTG]
; 7058          [AR]_[AR]-[BR],               ;COUNT DOWN TIME TO GO
; 7059          SKIP AD.LE.0                  ;SEE IF IT TIMED OUT
; 7060  =0
; 7061  TOCK3:  WORK[TTG]_[AR],               ;SAVE NEW TIME TO GO
; 7062          RETURN [2]                    ;ALL DONE
; 7063          [AR]_WORK[PERIOD]
; 7064          [BR]_APR                      ;GET CURRENT FLAGS
; 7065          [BR]_[BR].OR.#, #/40          ;SET TIMER INTERRUPT FLAG
; 7066          READ [BR],                    ;PLACE ON DP AND
; 7067          SPEC/APR FLAGS,               ; LOAD INTO HARDWARE
; 7068          J/TOCK3                       ;ALL DONE
; 7069
```

```
U 1774, 3572,4443,0000,4174,4007,0700,0200,0004,0002
U 3572, 1114,3771,0003,4365,5007,0700,0200,0000,0002

U 1114, 3662,0111,0702,4170,4007,0700,0210,0004,0012

U 1115, 3573,4551,0404,4370,4007,0700,0010,0077,0000

U 1117, 1400,3333,0004,7174,4007,0700,0400,0000,0301

U 3573, 0002,3333,0003,7174,4004,1700,0400,0000,0300

U 1764, 3574,4451,1205,4324,4007,0700,0000,0000,0000
U 3574, 3575,4451,1204,4324,4007,0700,0000,0000,0000
U 3575, 3576,4451,1206,4324,4007,0700,0000,0000,0000

U 3576, 2364,6113,0405,4174,4007,0621,0000,0000,0000
U 2364, 2365,3441,0604,4174,4007,0700,0000,0000,0000
U 2365, 3577,3771,0005,7274,4007,0701,0000,0000,0300

U 3577, 1120,0551,0404,7274,4007,0671,0000,0000,0301

U 1120, 3562,3771,0003,7274,4117,0700,0010,0000,0301

U 1121, 3600,3333,0002,4174,4007,0700,0200,0003,0012
U 1122, 1764,4443,0000,4174,4007,0700,0000,0000,0000

U 3600, 3601,3333,0005,4175,5007,0701,0200,0000,0002
U 3601, 3602,0111,0702,4170,4007,0700,0200,0003,0012
U 3602, 1400,3333,0004,4175,5007,0701,0200,0000,0002
```

```
; 7070  .TOC    "IO -- INTERNAL DEVICES -- WRTIME & RDTIME"
; 7071
; 7072  1774:
; 7073  WRTIME: START READ                ;FETCH WORD AT E
; 7074          MEM READ,                 ;WAIT FOR DATA
; 7075          [AR]_MEM                  ;PUT WORD IN AR
; 7076  =OO     VMA_[HR]+1,               ;BUMP E
; 7077          START READ,               ;START MEMORY
; 7078          CALL [LOADARX]            ;PUT DATA IN ARX
; 7079          [ARX]_[ARX].AND.#,        ;CLEAR PART HELD IN
; 7080          #/770000,                 ; HARDWARE COUNTER
; 7081          HOLD LEFT,  CALL [WRTIM1]
; 7082  =11     WORK[TIME1]_[ARX],        ;IN WORK SPACE
; 7083          J/DONE                    ;NEXT INSTRUCTION
; 7084  =
; 7085  WRTIM1: WORK[TIMEO]_[AR],         ;SAVE THE NEW VALUE
; 7086          RETURN [2]
; 7087
; 7088  1764:
; 7089  RDTIME: [BR]_TIME                 ;READ THE TIME
; 7090          [ARX]_TIME                ; AGAIN
; 7091          [BRX]_TIME                ; AGAIN
; 7092          [BR].XOR.[ARX],           ;SEE IF STABLE
; 7093          SKIP AD.EQ.O              ; ..
; 7094  =O     [ARX]_[BRX]               ;NO THEN NEXT TRY MUST BE OK
; 7095          [BR]_WORK[TIMEO]
; 7096          [ARX]_[ARX]+WORK[TIME1], ;COMBINE PARTS
; 7097          SKIP/-1 MS                ;SEE IF OVERFLOW HAPPENED
; 7098  =OO     SPEC/CLRCLK,              ;CLEAR CLOCK FLAG
; 7099          [AR]_WORK[TIME1], 2T,     ;GET LOW WORD FOR TOCK
; 7100          CALL [TOCK]               ;UPDATE CLOCKS
; 7101          READ [HR], LOAD VMA,      ;DID NOT OVERFLOW
; 7102          START WRITE, J/RDTIM1     ;STORE ANSWER
; 7103          J/RDTIME                  ;TRY AGAIN
; 7104  =
; 7105  RDTIM1: MEM WRITE, MEM_[BR]
; 7106          VMA_[HR]+1, LOAD VMA, START WRITE
; 7107          MEM WRITE, MEM_[ARX], J/DONE
; 7108
```

```
U 1775,  3603,4443,0000,4174,4007,0700,0200,0004,0002
U 3603,  3604,3771,0003,4365,5007,0700,0200,0000,0002
U 3604,  3605,3333,0003,7174,4007,0700,0400,0000,0302

U 3605,  1400,3333,0003,7174,4007,0700,0400,0000,0303


U 1765,  3614,3771,0005,7274,4007,0701,0000,0000,0302
```

```
; 7109  .TOC    "IO -- INTERNAL DEVICES -- WRINT & RDINT"
; 7110
; 7111
; 7112  1775:
; 7113  WRINT:  START READ
; 7114          MEM READ, [AR]_MEM
; 7115          WORK[PERIOD]_[AR]
; 7116          WORK[TTG]_[AR],
; 7117          J/DONE
; 7118
; 7119  1765:
; 7120  RDINT:  [BR]_WORK[PERIOD],
; 7121          J/RTNREG
; 7122
```

```
                                                        ; 7123  .TOC    "IO -- INTERNAL DEVICES -- RDPI & WRPI"
                                                        ; 7124
                                                        ; 7125  1715:
U 1715, 3614,3441,1405,4174,4007,0700,0000,0000,0000    ; 7126  RDPI:   [BR]_[PI], J/RTNREG
                                                        ; 7127
                                                        ; 7128  1714:
U 1714, 2366,4553,0300,4374,4007,0331,0000,0001,0000    ; 7129  WRPI:   TR [AR], PI.CLR/1
U 2366, 2367,4221,0014,4174,4007,0700,0000,0000,0000    ; 7130  =O      [PI]_O
U 2367, 2370,4553,0300,4374,4007,0331,0000,0074,0000    ; 7131          TR [AR], PI.MBZ/17
U 2370, 2740,4551,0202,4374,0007,0700,0000,0077,7740    ; 7132  =O      UUO
U 2371, 3606,4551,0305,4374,4007,0700,0000,0000,0177    ; 7133          [BR]_[AR].AND.#,#/177
U 3606, 3607,3770,0505,4344,0007,0700,0000,0000,0000    ; 7134          [BR]_[BR] SWAP, HOLD RIGHT
U 3607, 2372,4553,0300,4374,4007,0331,0000,0002,0000    ; 7135          TR [AR], PI.DIR/1
U 2372, 2373,5111,0514,4174,0007,0700,0000,0000,0000    ; 7136  =O      [PI]_[PI].AND.NOT.[BR], HOLD RIGHT
U 2373, 2374,4553,0300,4374,4007,0331,0000,0000,4000    ; 7137          TR [AR], PI.REQ/1
U 2374, 2375,3111,0514,4174,0007,0700,0000,0000,0000    ; 7138  =O      [PI]_[PI].OR.[BR], HOLD RIGHT
U 2375, 2376,4553,0300,4374,4007,0331,0000,0000,0200    ; 7139          TR [AR], PI.TSN/1
U 2376, 2377,3551,1414,4370,4007,0700,0000,0000,0200    ; 7140  =O      [PI]_[PI].OR.#,PI.ON/1, HOLD LEFT
U 2377, 2400,4553,0300,4374,4007,0331,0000,0000,0400    ; 7141          TR [AR], PI.TSF/1
U 2400, 2401,5551,1414,4370,4007,0700,0000,0000,0200    ; 7142  =O      [PI]_[PI].AND.NOT.#,PI.ON/1, HOLD LEFT
U 2401, 2402,4553,0300,4374,4007,0331,0000,0000,2000    ; 7143          TR [AR], PI.TCN/1
U 2402, 2403,3111,0514,4170,4007,0700,0000,0000,0000    ; 7144  =O      [PI]_[PI].OR.[BR], HOLD LEFT
U 2403, 0304,4553,0300,4374,4007,0331,0000,0000,1000    ; 7145          TR [AR], PI.TCF/1
U 0304, 0305,5111,0514,4170,4007,0700,0000,0000,0000    ; 7146  =O**O   [PI]_[PI].AND.NOT.[BR], HOLD LEFT
U 0305, 3611,3770,1416,4344,4007,0700,0010,0000,0000    ; 7147  PIEXIT: CALL LOAD PI
                                                        ; 7148  =1**1
U 0315, 0110,3443,0100,4174,4156,4700,0200,0014,0012    ; 7149          DONE
                                                        ; 7150  =
                                                        ; 7151
                                                        ; 7152  ;SUBROUTINE TO LOAD PI HARDWARE
                                                        ; 7153  ;CALL WITH:
                                                        ; 7154  ;       CALL LOAD PI
                                                        ; 7155  ;RETURNS IO WITH PI HARDWARE LOADED
                                                        ; 7156  ;
U 3610, 3611,3770,1416,4344,4007,0700,0000,0000,0000    ; 7157  LOADPI: [TO]_[PI] SWAP          ;PUT ACTIVE CHANS IN LH
U 3611, 3612,2441,0716,4170,4007,0700,4000,0000,0000    ; 7158  LDPI2:  [TO]_-1, HOLD LEFT      ;DONT MASK RH
U 3612, 3613,4111,1416,4174,4007,0700,0000,0000,0000    ; 7159          [TO]_[TO].AND.[PI]      ;ONLY REQUEST CHANS THAT ARE ON
                                                        ; 7160          .NOT.[TO], LOAD PI,     ;RELOAD HARDWARE
U 3613, 0010,7443,1600,4174,4434,1700,0000,0000,0000    ; 7161           RETURN [IO]            ;RETURN TO CALLER
                                                        ; 7162
```

```
U 3614, 3615,3443,0300,4174,4007,0700,0200,0003,0012
U 3615, 1400,3333,0005,4175,5007,0701,0200,0000,0002
```

```
; 7163  .TOC    "IO -- INTERNAL DEVICES -- SUBROUTINES"
; 7164
; 7165
; 7166  ;HERE WITH SOMETHING IN BR STORE IT @AR
; 7167  RTNREG: VMA_[AR], START WRITE
; 7168          MEM WRITE, MEM_[BR], J/DONE
; 7169
```

```
U 1722, 3616,3443,0300,4174,4147,0700,0200,0000,0010
U 3616, 2404,4221,0003,4174,4007,0700,0000,0000,0000

U 2404, 3622,3771,0003,4374,4007,0700,0010,0037,7377

U 2405, 3617,3771,0005,4374,4247,0700,0000,0000,1001
U 3617, 2406,3333,0003,4174,4247,0700,0000,0000,1000




U 2406, 2406,1111,0503,4174,4247,0630,6000,0060,1000
U 2407, 2413,3333,0003,4174,4007,0700,0000,0000,0000



U 2410, 3622,3771,0003,4374,4007,0700,0010,0037,7377

U 2411, 3620,3771,0005,4374,4347,0700,0000,0000,1001
U 3620, 2412,3333,0003,4174,4347,0700,0000,0000,1000




U 2412, 2412,1111,0503,4174,4347,0630,6000,0060,1000

U 2413, 3621,4223,0000,7174,4007,0700,0400,0000,0424

U 3621, 1400,4223,0000,7174,4007,0700,0400,0000,0423


U 3622, 0001,4443,0000,4174,4004,1700,2000,0071,0375
```

```
; 7170  ;CACHE SWEEP
; 7171
; 7172  1722:
; 7173  CLRPT:  VMA_[AR],                ;PUT CORRECT ADDRESS IN VMA
; 7174          LOAD PAGE TABLE          ;GET SET TO WRITE PAGE TABLE
; 7175          [AR]_0                   ;CLEAR ENTRY
; 7176  =0      [AR]_#,#/377377,         ;INITIAL VMA VALUE
; 7177          CALL [SSWEEP]            ;LOAD THE SC
; 7178          [BR]_#, #/1001,          ;CONSTANT TO KEEP ADDING
; 7179          CLRCSH                   ;START TO CLEAR CACHE
; 7180          READ [AR], CLRCSH        ;FIRST THING TO CLEAR
; 7181  =0
; 7182  CLRPTL: [AR]_[AR]-[BR],          ;UPDATE AR (AND PUT ON DP)
; 7183          CLRCSH,                  ;SWEEP ON NEXT STEP
; 7184          STEP SC,                 ;SKIP IF WE ARE DONE
; 7185          J/CLRPTL                 ;LOOP FOR ALL ENTRIES
; 7186          READ [AR], J/ZAPPTA      ;CLEAR LAST ENTRY
; 7187
; 7188  =0
; 7189  SWEEP:  [AR]_#,#/377377,         ;INITIAL VMA VALUE
; 7190          CALL [SSWEEP]            ;LOAD NUMBER OF STEPS INTO SC
; 7191          [BR]_#, #/1001,          ;CONSTANT TO KEEP ADDING
; 7192          SWEEP                    ;START SWEEP
; 7193          READ [AR], SWEEP         ;FIRST THING TO CLEAR
; 7194  =0
; 7195  SWEEPL: [AR]_[AR]-[BR],          ;UPDATE AR (AND PUT ON DP)
; 7196          SWEEP,                   ;SWEEP ON NEXT STEP
; 7197          STEP SC,                 ;SKIP IF WE ARE DONE
; 7198          J/SWEEPL                 ;LOOP FOR ALL ENTRIES
; 7199                                   ;CLEAR LAST ENTRY AND
; 7200  ZAPPTA: WORK[PTA.U]_0            ; FORGET PAGE TABLE ADDRESS
; 7201          WORK[PTA.E]_0,           ;FORGET PAGE TABLE ADDRESS
; 7202          J/DONE                   ;ALL DONE
; 7203
; 7204  SSWEEP: SC_S#, S#/375,          ;NUMBER OF STEPS
; 7205          RETURN [1]               ;RETURN
; 7206
```

```
U 0770, 0770,3773,0000,4074,4003,1701,0000,0000,0000
U 0771, 3623,3551,1414,4370,4007,0700,0000,0004,0000
U 0772, 3624,3551,1414,4370,4007,0700,0000,0002,0000
U 0773, 3625,3551,1414,4370,4007,0700,0000,0001,0000
U 0774, 3626,3551,1414,4370,4007,0700,0000,0000,4000
U 0775, 3627,3551,1414,4370,4007,0700,0000,0000,2000
U 0776, 3630,3551,1414,4370,4007,0700,0000,0000,1000
U 0777, 3631,3551,1414,4370,4007,0700,0000,0000,0400
U 3623, 3632,4751,1206,4374,4007,0700,0000,0000,0001
U 3624, 3632,4751,1206,4374,4007,0700,0000,0000,0002
U 3625, 3632,4751,1206,4374,4007,0700,0000,0000,0003
U 3626, 3632,4751,1206,4374,4007,0700,0000,0000,0004
U 3627, 3632,4751,1206,4374,4007,0700,0000,0000,0005
U 3630, 3632,4751,1206,4374,4007,0700,0000,0000,0006
U 3631, 3632,4751,1206,4374,4007,0700,0000,0000,0007


U 3632, 3633,4251,1403,4374,4007,0700,0000,0007,7577
U 3633, 3634,7443,0300,4174,4437,0700,0000,0000,0000
U 3634, 2414,4223,0000,4364,4277,0700,0200,0000,0010


U 2414, 3667,4571,1203,4374,4007,0700,0010,0024,1300


U 2415, 2416,3771,0003,4364,4007,0331,0200,0000,0002
U 2416, 3643,4221,0004,4174,4007,0700,0000,0000,0000
U 2417, 3635,3445,0603,4174,4007,0700,0000,0000,0000

U 3635, 3636,0551,0303,4370,4007,0701,0000,0000,0040

U 3636, 3637,0111,1003,4174,4007,0700,0000,0000,0000
```

```
; 7207  ;WE COME HERE EITHER FROM NEXT INSTRUCTION DISPATCH OR PAGE FAIL
; 7208  ; LOGIC. IN ALL CASES, THE CURRENT INSTRUCTION IS CORRECTLY SETUP
; 7209  ; TO RESTART PROPERLY.
; 7210
; 7211  ;FIRST SET THE CORRECT PI IN PROGRESS BIT
; 7212  ;       [FLG]_[FLG].OR.#,FLG.PI/1, HOLD RIGHT,
; 7213  ;               J/PI            ;SET PI CYCLE AND PROCESS PI
; 7214  =1000
; 7215  PI:     AD/D, DBUS/PI NEW,      ;LOOK AT NEW LEVEL
; 7216          DISP/DP LEFT, 3T,       ;DISPATCH ON IT
; 7217          J/PI                    ;GO TO 1 OF NEXT 7 PLACES
; 7218  =1001   [PI]_[PI].OR.#, #/040000, HOLD LEFT, J/PIP1
; 7219  =1010   [PI]_[PI].OR.#, #/020000, HOLD LEFT, J/PIP2
; 7220  =1011   [PI]_[PI].OR.#, #/010000, HOLD LEFT, J/PIP3
; 7221  =1100   [PI]_[PI].OR.#, #/004000, HOLD LEFT, J/PIP4
; 7222  =1101   [PI]_[PI].OR.#, #/002000, HOLD LEFT, J/PIP5
; 7223  =1110   [PI]_[PI].OR.#, #/001000, HOLD LEFT, J/PIP6
; 7224  =1111   [PI]_[PI].OR.#, #/000400, HOLD LEFT, J/PIP7
; 7225  PIP1:   [BRX]_0 XWD [1], J/PI10 ;REMEMBER WE ARE AT LEVEL 1
; 7226  PIP2:   [BRX]_0 XWD [2], J/PI10 ;REMEMBER WE ARE AT LEVEL 2
; 7227  PIP3:   [BRX]_0 XWD [3], J/PI10 ;REMEMBER WE ARE AT LEVEL 3
; 7228  PIP4:   [BRX]_0 XWD [4], J/PI10 ;REMEMBER WE ARE AT LEVEL 4
; 7229  PIP5:   [BRX]_0 XWD [5], J/PI10 ;REMEMBER WE ARE AT LEVEL 5
; 7230  PIP6:   [BRX]_0 XWD [6], J/PI10 ;REMEMBER WE ARE AT LEVEL 6
; 7231  PIP7:   [BRX]_0 XWD [7], J/PI10 ;REMEMBER WE ARE AT LEVEL 7
; 7232
; 7233  PI10:   [AR]_[PI].AND.# CLR LH,  ;TURN OFF PI SYSTEM
; 7234          #/077577                ; TILL WE ARE DONE
; 7235          .NOT.[AR], LOAD PI      ;  ..
; 7236          ABORT MEM CYCLE         ;NO MORE TRAPS
; 7237  =0     [AR]_VMA IO READ,       ;SETUP TO READ WRU BITS
; 7238          WRU CYCLE/1,            ; ..
; 7239          CALL [STRTIO]           ;START THE CYCLE
; 7240          MEM READ,               ;WAIT FOR DATA
; 7241          [AR]_IO DATA, 3T,       ;PUT DATA IN AR
; 7242          SKIP ADR.EQ.0           ;SEE IF ANYONE THERE
; 7243  =0     [ARX]_0, J/VECINT       ;YES--VECTORED INTERRUPT
; 7244          [AR]_[BRX]*2            ;N*2
; 7245          [AR]_[AR]+#, #/40, 3T,  ;2*N+40
; 7246          HOLD LEFT               ; ..
; 7247          [AR]_[AR]+[EBR],        ;ABSOULTE ADDRESS OF
; 7248          J/PI40                  ; INTERRUPT INSTRUCTION
; 7249
```

```
U 3637, 3640,3443,0300,4174,4007,0700,0200,0024,1016

U 3640, 3641,3771,0003,4365,5007,0701,0200,0020,0012
U 3641, 2420,6553,0300,4374,4007,0321,0000,0025,4340
U 2420, 2422,6553,0300,4374,4007,0321,0000,0026,4000
U 2421, 3642,4521,1205,4074,4007,0700,0000,0000,0000

U 3642, 0060,4223,0000,4174,4467,0700,0000,0000,0004

U 2422, 0104,4751,1217,4374,4007,0700,0000,0000,0101
U 2423, 0470,4443,0000,4174,4007,0700,0200,0023,0002

U 0470, 3670,3741,0105,4074,4007,0700,0010,0000,0000

GS
U 0474, 3665,3771,0003,4374,0007,0700,0010,0000,0000

U 0475, 2715,3441,0301,4174,4467,0700,0000,0000,0004
```

```
; 7250  ;HERE WITH ABSOLUTE ADDRESS OF INTERRUPT INSTRUCTION IN [AR]
; 7251  PI40:   VMA_[AR], VMA PHYSICAL READ      ;FETCH THE INSTRUCTION
; 7252  PI50:   MEM READ, [AR]_MEM, LOAD VMA,    ;FETCH INSTRUCTION
; 7253          3T, FORCE EXEC                   ;E IS EXEC MODE
; 7254          [AR].XOR.#, #/254340, 3T, SKIP ADL.EQ.0
; 7255  =0       [AR].XOR.#, #/264000, SKIP ADL.EQ.0, 3T, J/PIJSR
; 7256          [BR]_FLAGS                       ;SAVE FLAGS
; 7257          AD/ZERO, LOAD FLAGS,
; 7258          J/PIXPCW                         ;ENTER EXEC MODE AND ASSUME
; 7259                                           ; WE HAVE AN XPCW
; 7260  ;IF WE HALT HERE ON A VECTORED INTERRUPT, WE HAVE
; 7261  ;       TO/ WHAT WE READ FROM BUS AS VECTOR
; 7262  ;       ARX/ EPT+100+DEVICE
; 7263  ;       BR/  ADDRESS OF ILLEGAL INSTRUCTION
; 7264  ;       BRX/ VECTOR (MASKED AND SHIFTED)
; 7265  =0
; 7266  PIJSR:  HALT [ILLII]                      ;NOT A JSR OR XPCW
; 7267          START WRITE, FORCE EXEC          ;PREPARE TO STORE OLD PC
; 7268  =0*0    [BR]_PC WITH FLAGS,              ;OLD PC
; 7269          CALL [STOBR]                     ;STORE OLD PC
; 7270  =1*0    [AR]_#, #/0, HOLD RIGHT,                ;PREPARE TO CLEAR FL
; 7271          CALL [INCAR]                     ;BUMP POINTER
; 7272  =1*1    [PC]_[AR], LOAD FLAGS,           ;NEW PC
; 7273          J/PISET                          ;CLEAR PI CYCLE & START
; 7274                                           ; INTERRUPT PROGRAM
; 7275  =
; 7276
```

```
U 3643, 2424,3445,0303,4174,4007,0530,0000,0000,0000

U 2424, 3643,0111,1504,4174,4007,0700,0000,0000,0000

U 2425, 2426,4571,1203,4374,4007,0700,0000,0024,1240

U 2426, 3667,3111,0403,4174,4007,0700,0010,0000,0000

U 2427, 2430,3771,0016,4364,4007,0700,0200,0000,0002

U 2430, 3664,0551,1005,4374,4007,0701,0010,0000,0100

U 2431, 3644,0111,0504,4174,4007,0700,0200,0024,1016

U 3644, 2432,3771,0005,4365,5007,0331,0200,0000,0002

U 2432, 3645,4557,1606,4374,4007,0701,0000,0000,0774
U 2433, 0104,4751,1217,4374,4007,0700,0000,0000,0102
U 3645, 3646,3447,0606,4174,4007,0700,0000,0000,0000


U 3646, 3640,0111,0605,4174,4007,0700,0200,0024,0012
```

```
; 7277  ;HERE TO PROCESS A VECTORED INTERRUPT. AT THIS POINT:
; 7278  ;       AR/ WRU BITS (BIT 18 FOR DEVICE 0)
; 7279  ;       ARX/ 0
; 7280  VECINT: [AR]_[AR]*2,              ;SHIFT LEFT (UNSHIFTED ON DP)
; 7281          SKIP DP18                 ;ANYONE THERE?
; 7282  =0      [ARX]_[ARX]+[XWD1],       ;NO--BUMP BOTH HALVES
; 7283          J/VECINT                  ;KEEP LOOKING
; 7284          [AR]_VMA IO READ,         ;SETUP FOR VECTOR CYCLE
; 7285          VECTOR CYCLE/1            ; ..
; 7286  =0      [AR]_[AR].OR.[ARX],       ;PUT IN UNIT NUMBER
; 7287          CALL [STRTIO]             ;START CYCLE
; 7288          MEM READ,                 ;WAIT FOR VECTOR (SEE DPM5)
; 7289          [TO]_IO DATA              ;GET VECTOR
; 7290  =0      [BR]_[EBR]+#, 3T, #/100,          ;EPT+100
; 7291          CALL [CLARXL]             ;CLEAR ARX LEFT
; 7292          [ARX]_[ARX]+[BR],         ;EPT+100+DEVICE
; 7293          VMA PHYSICAL READ         ;FETCH WORD
; 7294          MEM READ, [BR]_MEM, 3T,   ;GET POINTER
; 7295          SKIP ADR.EQ.0             ;SEE IF NON-ZERO
; 7296  =0      [BRX]_([TO].AND.#)*.5, 3T, ;OK--MAKE VECTOR MOD 400
; 7297          #/774, J/VECIN1           ; AND SHIFT OVER
; 7298          HALT [ILLINT]
; 7299  VECIN1: [BRX]_[BRX]*.5            ;SHIFT 1 MORE PLACE
; 7300          [BR]_[BR]+[BRX],          ;ADDRESS OF WORD TO USE
; 7301          LOAD VMA, FORCE EXEC,     ;FORCE EXEC VIRTUAL ADDRESS
; 7302          START READ, J/PI50        ;GO GET INSTRUCTION
; 7303
```

```
U 2434, 3647,3771,0005,4374,4007,0700,0000,0004,0000
U 2435, 0004,4443,0000,4174,4004,1700,0000,0000,0000

U 3647, 2436,4113,0514,4174,4007,0330,0000,0000,0000
U 2436, 0004,5111,0514,4170,4004,1700,0000,0000,0000
U 2437, 3647,3447,0505,4174,4007,0700,0000,0000,0000
```

```
; 7304  .TOC    "PRIORITY INTERRUPTS -- DISMISS SUBROUTINE"
; 7305
; 7306  ;SUBROUTINE TO DISMISS THE HIGHEST PI IN PROGRESS
; 7307  ;RETURNS 4 ALWAYS
; 7308
; 7309  ;DISMISS:
; 7310  ;       TR [PI], #/077400       ;ANY PI IN PROGRESS?
; 7311  =0
; 7312  JEN1:   [BR]_#, PI.IP1/1, J/DSMS1 ;YES--START LOOP
; 7313          RETURN [4]              ;NO--JUST RETURN
; 7314
; 7315  DSMS1:  [PI].AND.[BR], SKIP ADR.EQ.0
; 7316  =0      [PI]_[PI].AND.NOT.[BR], HOLD LEFT, RETURN [4]
; 7317          [BR]_[BR]*.5, J/DSMS1
; 7318
```

```
D 0710, 1210,1614,0100
D 0711, 1214,1614,0100
D 0720, 1200,1614,0100
D 0721, 1204,1614,0100

U 1614, 2440,4443,0000,4174,4007,0700,0010,0000,0000
U 1617, 0014,4551,0305,0274,4003,7700,0000,0000,0000

D 0712, 1210,1460,0100
D 0713, 1210,1461,0100
D 0722, 1200,1460,0100
D 0723, 1200,1461,0100

U 1460, 2440,4443,0000,4174,4007,0700,0010,0000,0000
U 1463, 1400,3440,0303,0174,4007,0700,0400,0000,0000

U 1461, 2450,3771,0005,0276,6007,0700,0000,0000,0000

D 0714, 1210,1644,0100
D 0715, 1214,1644,0100
D 0724, 1200,1644,0100
D 0725, 1204,1644,0100

U 1644, 2440,3441,0306,4174,4007,0700,0010,0000,0000
U 1647, 1013,3441,0305,4174,4003,7700,0000,0000,0000
U 1013, 3650,3551,0505,0274,4007,0700,0000,0000,0000
U 1017, 3650,5551,0505,0274,4007,0700,0000,0000,0000

U 3650, 2450,3441,0603,4174,4007,0700,0000,0000,0000
```

```
; 7319  .TOC    "EXTERNAL IO INSTRUCTIONS"
; 7320
; 7321          .DCODE
; 7322  710:    IOT,     WORD-TNE,        J/TIOX
; 7323  711:    IOT,     WORD-TNN,        J/TIOX
; 7324  720:    IOT,     TNE,             J/TIOX
; 7325  721:    IOT,     TNN,             J/TIOX
; 7326          .UCODE
; 7327
; 7328  1614:
; 7329  TIOX:   CALL [IORD]
; 7330  1617:   [BR]_[AR].AND.AC, TEST DISP
; 7331
; 7332          .DCODE
; 7333  712:    IOT,     B/10,            J/RDIO
; 7334  713:    IOT,     B/10,            J/WRIO
; 7335  722:    IOT,     B/0,             J/RDIO
; 7336  723:    IOT,     B/0,             J/WRIO
; 7337          .UCODE
; 7338
; 7339  1460:
; 7340  RDIO:   CALL [IORD]
; 7341  1463:   AC_[AR], J/DONE
; 7342
; 7343  1461:
; 7344  WRIO:   [BR]_AC, J/IOWR
; 7345
; 7346          .DCODE
; 7347  714:    IOT,             B/10,   J/BIXUB
; 7348  715:    IOT,             B/14,   J/BIXUB
; 7349  724:    IOT,             B/0,    J/BIXUB
; 7350  725:    IOT,             B/4,    J/BIXUB
; 7351          .UCODE
; 7352
; 7353  1644:
; 7354  BIXUB:  [BRX]_[AR],              ;SAVE EFFECTIVE ADDRESS
; 7355          CALL [IORD]              ;GO GET THE DATA
; 7356  1647:   [BR]_[AR],               ;COPY DATA ITEM
; 7357          B DISP                   ;SEE IF SET OR CLEAR
; 7358  =1011   [BR]_[BR].OR.AC,         ;SET BITS
; 7359          J/BIXUB1                 ;GO DO WRITE
; 7360          [BR]_[BR].AND.NOT.AC,    ;CLEAR BITS
; 7361          J/BIXUB1                 ;GO DO WRITE
; 7362
; 7363  BIXUB1: [AR]_[BRX],              ;RESTORE ADDRESS
; 7364          J/IOWR
; 7365
```

```
                                        EXTERNAL IO INSTRUCTIONS

                                ; 7366  ;SUBROUTINE TO READ FROM AN IO DEVICE
                                ; 7367  ;CALL WITH:
                                ; 7368  ;       SECTION 0 EFFECTIVE ADDRESS IN AR
                                ; 7369  ;       INSTRUCTION IN HR
                                ; 7370  ;RETURN 3 WITH WORD OR BYTE IN AR
                                ; 7371  ;
                                ; 7372  =0
                                ; 7373  IORD:   CLR IO BUSY,             ;CLEAR BUSY
U 2440, 2460,4443,0000,4174,4137,0700,0010,0000,0000
                                ; 7374          CALL [IOEA]             ;COMPUTE IO EA
U 2441, 0067,4443,0000,4174,4003,7700,0000,0000,0000
                                ; 7375          B DISP
                                ; 7376  =10111  [BR]_VMA IO READ,       ;BYTE MODE
                                ; 7377          IO BYTE/1,              ;SET BYTE FLAG
U 0067, 2442,4571,1205,4374,4007,0700,0000,0024,1220
                                ; 7378          J/IORD1                 ;GO DO C/A CYCLE
U 0077, 2442,4571,1205,4374,4007,0700,0000,0024,1200
                                ; 7379  =11111  [BR]_VMA IO READ        ;WORD MODE
                                ; 7380  =
                                ; 7381  =0
                                ; 7382  IORD1:  VMA_[AR].OR.[BR] WITH FLAGS,
U 2442, 3657,3113,0305,4174,4007,0701,0210,0000,0036
                                ; 7383          CALL [IOWAIT]           ;WAIT FOR THINGS COMPLETE
                                ; 7384          MEM READ,               ;MAKE SURE REALLY READY
                                ; 7385          [BR]_IO DATA,           ;PUT DATA IN BR
U 2443, 1027,3771,0005,4364,4003,7700,0200,0000,0002
                                ; 7386          B DISP                  ;SEE IF BYTE MODE
U 1027, 2444,4553,0300,4374,4007,0331,0000,0000,0001
                                ; 7387  =0111   TR [AR], #/1, J/IORD2   ;BYTE MODE SEE IF ODD
U 1037, 0003,3441,0503,4174,4004,1700,0000,0000,0000
                                ; 7388          [AR]_[BR], RETURN [3]   ;ALL DONE
                                ; 7389
                                ; 7390  ;HERE ON WORD MODE
                                ; 7391  =0
                                ; 7392  IORD2:  [BR]_[BR]*.5, SC_5,     ;LEFT BYTE
U 2444, 2446,3447,0505,4174,4007,0700,2000,0071,0005
                                ; 7393          J/IORD3                 ;GO SHIFT IT
                                ; 7394          [AR]_[BR].AND.#,        ;MASK IT
U 2445, 0003,4551,0503,4374,4004,1700,0000,0000,0377
                                ; 7395          #/377, RETURN [3]       ;ALL DONE
                                ; 7396
                                ; 7397  =0
                                ; 7398  IORD3:  [BR]_[BR]*.5,           ;SHIFT OVER
U 2446, 2446,3447,0505,4174,4007,0630,2000,0060,0000
                                ; 7399          STEP SC, J/IORD3        ; ..
                                ; 7400          [AR]_[BR].AND.#,        ;MASK IT
U 2447, 0003,4551,0503,4374,4004,1700,0000,0000,0377
                                ; 7401          #/377, RETURN [3]       ;ALL DONE
                                ; 7402
```

```
U 2450, 2460,4443,0000,4174,4137,0700,0010,0000,0000
U 2451, 0227,4443,0000,4174,4003,7700,0000,0000,0000
U 0227, 2454,4553,0300,4374,4007,0331,0000,0000,0001
U 0237, 3651,4571,1204,4374,4007,0700,0000,0021,1200

U 3651, 2452,3113,0304,4174,4007,0701,0200,0000,0036

U 2452, 3657,3333,0005,4175,5007,0701,0210,0000,0002
U 2453, 0110,3443,0100,4174,4156,4700,0200,0014,0012




U 2454, 2456,3445,0505,4174,4007,0700,2000,0071,0005

U 2455, 3651,4571,1204,4374,4007,0700,0000,0021,1220



U 2456, 2456,3445,0505,4174,4007,0630,2000,0060,0000

U 2457, 3651,4571,1204,4374,4007,0700,0000,0021,1220
```

```
; 7403  ;ROUTINE TO WRITE TO AN IO DEVICE
; 7404  ;CALL WITH:
; 7405  ;       SECTION 0 EFFECTIVE ADDRESS IN AR
; 7406  ;       INSTRUCTION IN HR
; 7407  ;       WORD OR BYTE IN BR
; 7408  ;RETURNS BACK TO USER
; 7409  ;
; 7410  =0
; 7411  IOWR:   CLR IO BUSY,              ;CLEAR BUSY
; 7412          CALL [IOEA]               ;COMPUTE IO EA
; 7413          B DISP
; 7414  =10111  TR [AR], #/1, J/IOWR2     ;BYTE MODE
; 7415  =11111  [ARX]_VMA IO WRITE        ;SETUP FLAGS
; 7416  =
; 7417  IOWR1:  VMA_[AR].OR.[ARX] WITH FLAGS
; 7418  =0      MEM WRITE, MEM_[BR],      ;SEND DATA
; 7419          CALL [IOWAIT]             ;WAIT FOR DATA
; 7420          DONE                      ;RETURN
; 7421
; 7422  ;HERE FOR BYTE MODE
; 7423  =0
; 7424  IOWR2:  [BR]_[BR]*2, SC_5,        ;ODD--MOVE LEFT
; 7425          J/IOWR3                   ; ..
; 7426          [ARX]_VMA IO WRITE,       ;SETUP FLAGS
; 7427          IO BYTE/1, J/IOWR1        ; ..
; 7428
; 7429  =0
; 7430  IOWR3:  [BR]_[BR]*2, STEP SC,     ;SHIFT LEFT
; 7431          J/IOWR3                   ;KEEP SHIFTING
; 7432          [ARX]_VMA IO WRITE,       ;SETUP FLAGS
; 7433          IO BYTE/1, J/IOWR1        ; ..
; 7434
```

```
                                        EXTERNAL IO INSTRUCTIONS

U 2460, 3661,1113,0701,4170,4007,0700,4210,0004,0012    ; 7435  ;HERE TO COMPUTE IO EFFECTIVE ADDRESS
U 2461, 3652,7441,0306,4174,4007,0700,0000,0000,0000    ; 7436  ;CALL WITH:
U 3652, 2462,4553,0600,4374,4007,0321,0000,0070,0000    ; 7437  ;       SECTION 0 EFFECTIVE ADDRESS IN AR
                                                        ; 7438  ;       INSTRUCTION IN HR
U 2462, 2464,4553,0200,4374,4007,0321,0000,0000,0020    ; 7439  ;RETURN 1 WITH EA IN AR
                                                        ; 7440  ;
U 2463, 2462,4713,1203,7174,4007,0700,0400,0000,0422    ; 7441  =0
                                                        ; 7442  IOEA:   VMA_[PC]-1,             ;GET INSTRUCTION
U 2464, 3653,3771,0003,7274,4007,0701,0000,0000,0422    ; 7443          START READ,             ; ..
U 2465, 1055,4443,0000,2174,4006,6700,0000,0000,0000    ; 7444          CALL [LOADAR]           ;PUT WORD IN AR
                                                        ; 7445          [BRX]_.NOT.[AR]         ;SEE IF IN RANGE 700-777
U 1055, 2470,3771,0004,2274,4007,0120,0000,0000,0000    ; 7446          TL [BRX], #/700000      ; ..
                                                        ; 7447  =0
U 1057, 0001,3771,0003,7274,4124,1701,0000,0000,0422    ; 7448  IOEA1:  TL [HR], #/20, J/IOEA2  ;INDIRECT?
                                                        ; 7449          WORK[YSAVE]_[AR] CLR LH, ;DIRECT IO INSTRUCTION
                                                        ; 7450          J/IOEA1                 ;SAVE Y FOR EA CALCULATION
U 3653, 3654,3333,0002,4174,4217,0700,0000,0000,0000    ; 7451  =0
U 3654, 2466,4553,0200,4374,4007,0321,0000,0000,0017    ; 7452  IOEA2:  [AR]_WORK[YSAVE],       ;@--GET SAVED Y
U 2466, 2467,0551,0303,2270,4007,0701,0000,0000,0000    ; 7453          J/IOEAI                 ;GET Y AND GO
U 2467, 3655,3443,0300,4174,4007,0700,0200,0004,0012    ; 7454          EA MODE DISP            ;WAS THERE INDEXING?
                                                        ; 7455  =1101   [ARX]_XR, SKIP ADL.LE.0, ;SEE IF LOCAL OR GLOBAL INDEXING
U 3655, 0001,3771,0003,4365,5124,1700,0200,0000,0002    ; 7456          2T, J/IOEAX             ; ..
                                                        ; 7457          [AR]_WORK[YSAVE],       ;JUST PLAIN IO
                                                        ; 7458          CLR IO LATCH, RETURN [1]
                                                        ; 7459
U 2470, 0001,0551,0403,7274,4124,1701,0000,0000,0422    ; 7460  IOEAI:  READ [HR], DBUS/DP,     ;LOAD XR FLOPS IN CASE
U 2471, 3656,0551,0403,7274,4007,0701,0000,0000,0422    ; 7461          LOAD INST EA            ; THERE IS INDEXING
                                                        ; 7462          TL [HR], #/17           ;WAS THERE ALSO INDEXING
U 3656, 0001,4221,0003,4174,0124,1700,0000,0000,0000    ; 7463  =0      [AR]_[AR]+XR, 3T, HOLD LEFT ;YES--ADD IN INDEX VALUE
                                                        ; 7464          VMA_[AR], START READ    ;FETCH DATA WORD
                                                        ; 7465          MEM READ, [AR]_MEM,     ;GO GET DATA WORD
                                                        ; 7466          CLR IO LATCH, RETURN [1]
                                                        ; 7467
                                                        ; 7468  =0
                                                        ; 7469  IOEAX:  [AR]_[ARX]+WORK[YSAVE], ;GLOBAL INDEXING
                                                        ; 7470          CLR IO LATCH, RETURN [1]
                                                        ; 7471          [AR]_[ARX]+WORK[YSAVE]  ;LOCAL INDEXING
                                                        ; 7472          [AR]_0, HOLD RIGHT,
                                                        ; 7473          CLR IO LATCH, RETURN [1]
                                                        ; 7474
```

```
U 3657,  1124,3771,0016,4354,4007,0650,2000,0071,0200
U 1124,  3660,3333,0016,7174,4127,0700,0400,0000,0210
U 1125,  1124,4443,0000,4174,4006,7653,2000,0060,0000
U 1126,  3660,3333,0016,7174,4127,0700,0400,0000,0210
U 1127,  2475,3333,0016,7174,4007,0700,0400,0000,0210
U 3660,  2472,4443,0000,4174,4007,0650,2000,0071,0777
U 2472,  2474,4443,0000,4174,4127,0630,2000,0060,0000
U 2473,  0001,4443,0000,4174,4004,1700,0000,0000,0000
U 2474,  2472,4443,0000,4174,4127,0653,0000,0000,0000
U 2475,  1502,4571,1206,4374,4007,0700,0000,0020,0000
```

```
; 7475  ;WAIT FOR IO TO COMPLETE
; 7476  ;RETURNS 1 OR PAGE FAILS
; 7477  ;
; 7478  IOWAIT: SC_S#, S#/200,          ;DELAY
; 7479          [TO]_VMA,               ;GET VMA
; 7480          SKIP/-IO BUSY           ;SEE IF BUSY YET
; 7481  =00
; 7482  IOW1:   CLR IO LATCH,           ;WENT BUSY
; 7483          WORK[SV.VMA]_[TO],      ;MAKE SURE SV.VMA IS SETUP
; 7484          J/IOW2                  ;WAIT FOR IT TO CLEAR
; 7485          SC_SC-1, SCAD DISP, 5T, ;SEE IF DONE YET
; 7486          SKIP/-IO BUSY,          ; ..
; 7487          J/IOW1                  ;BACK TO LOOP
; 7488          CLR IO LATCH,           ;WENT BUSY AND TIMEOUT
; 7489          WORK[SV.VMA]_[TO],      ;MAKE SURE SV.VMA IS SETUP
; 7490          J/IOW2                  ; ..
; 7491          WORK[SV.VMA]_[TO],      ;MAKE SURE SV.VMA IS SETUP
; 7492          J/IOW5                  ;GO TRAP
; 7493
; 7494  IOW2:   SC_S#, S#/777,          ;GO TIME IO
; 7495          SKIP/-IO BUSY           ; ..
; 7496  =0
; 7497  IOW3:   CLR IO LATCH,           ;TRY TO CLEAR LATCH
; 7498          STEP SC, J/IOW4         ;STILL BUSY
; 7499          RETURN [1]              ;IDLE
; 7500
; 7501  =0
; 7502  IOW4:   CLR IO LATCH, 5T,       ;TRY TO CLEAR LATCH
; 7503          SKIP/-IO BUSY,          ;SEE IF STILL BUSY
; 7504          J/IOW3                  ; ..
; 7505  IOW5:   [BRX]_[200000] XWD 0, J/HARD
; 7506
```

```
U 3661, 0001,3771,0003,4365,5004,1700,0200,0000,0002
U 3662, 0001,3771,0004,4365,5004,1700,0200,0000,0002
U 3663, 0001,3772,0000,4365,5004,1700,0200,0000,0002
U 3664, 0001,4221,0004,4174,0004,1700,0000,0000,0000
U 3665, 0001,0111,0703,4174,4004,1700,0000,0000,0000
U 3666, 0001,3445,0505,4174,4004,1700,0000,0000,0000
U 3667, 0001,3443,0300,4174,4004,1701,0200,0000,0036
U 3670, 0004,3333,0005,4175,5004,1701,0200,0000,0002
U 3671, 0001,3333,0001,4175,5004,1701,0200,0000,0002
U 3672, 0001,3440,0404,0174,4004,1700,0400,0000,0000
```

```
; 7507  .TOC    "SMALL SUBROUTINES"
; 7508
; 7509  ;HERE ARE A COLLECTION ON 1-LINE SUBROUTINES
; 7510  LOADAR: MEM READ, [AR]_MEM,      ;FROM MEMORY TO AR
; 7511           RETURN [1]              ;RETURN TO CALLER
; 7512
; 7513  LOADARX: MEM READ, [ARX]_MEM, RETURN [1]
; 7514
; 7515  LOADQ:  MEM READ, Q_MEM, RETURN [1]
; 7516
; 7517  CLARXL: [ARX]_0, HOLD RIGHT, RETURN [1]
; 7518
; 7519  INCAR:  [AR]_[AR]+1, RETURN [1]
; 7520
; 7521  SBRL:   [BR]_[BR]*2, RETURN [1]
; 7522
; 7523  STRTIO: VMA_[AR] WITH FLAGS, RETURN [1]
; 7524
; 7525  STOBR:  MEM WRITE, MEM_[BR], RETURN [4]
; 7526
; 7527  STOPC:  MEM WRITE, MEM_[PC], RETURN [1]
; 7528
; 7529  AC_ARX: AC_[ARX], RETURN [1]
; 7530
```

```
D 0703, 0003,1650,2100
D 0706, 0006,1650,2100
D 0707, 0007,1650,2100

D 0716, 0006,1651,2100
D 0717, 0007,1651,2100

D 0726, 0006,1652,2100
D 0727, 0007,1652,2100

D 0730, 0000,1653,2100
D 0731, 0001,1653,2100
D 0732, 0002,1653,2100
D 0733, 0003,1653,2100
D 0734, 0004,1653,2100
D 0735, 0005,1653,2100
D 0736, 0006,1653,2100
D 0737, 0007,1653,2100

D 0740, 0000,1654,2100
D 0741, 0001,1654,2100
D 0742, 0002,1654,2100
D 0743, 0003,1654,2100
D 0744, 0004,1654,2100
D 0745, 0005,1654,2100
D 0746, 0006,1654,2100
D 0747, 0007,1654,2100

D 0750, 0000,1655,2100
D 0751, 0001,1655,2100
D 0752, 0002,1655,2100
D 0753, 0003,1655,2100
D 0754, 0004,1655,2100
D 0755, 0005,1655,2100
D 0756, 0006,1655,2100
D 0757, 0007,1655,2100

D 0760, 0000,1656,2100
D 0761, 0001,1656,2100
D 0762, 0002,1656,2100
D 0763, 0003,1656,2100
D 0764, 0004,1656,2100
D 0765, 0005,1656,2100
D 0766, 0006,1656,2100
D 0767, 0007,1656,2100
```

```
; 7531  .TOC    "UNDEFINED IO INSTRUCTIONS"
; 7532
; 7533          .DCODE
; 7534  703:     I,      B/3,    J/IOT700
; 7535  706:     I,      B/6,    J/IOT700
; 7536           I,      B/7,    J/IOT700
; 7537
; 7538  716:     I,      B/6,    J/IOT710
; 7539           I,      B/7,    J/IOT710
; 7540
; 7541  726:     I,      B/6,    J/IOT720
; 7542           I,      B/7,    J/IOT720
; 7543
; 7544  730:     I,      B/0,    J/IOT730
; 7545           I,      B/1,    J/IOT730
; 7546           I,      B/2,    J/IOT730
; 7547           I,      B/3,    J/IOT730
; 7548           I,      B/4,    J/IOT730
; 7549           I,      B/5,    J/IOT730
; 7550           I,      B/6,    J/IOT730
; 7551           I,      B/7,    J/IOT730
; 7552
; 7553  740:     I,      B/0,    J/IOT740
; 7554           I,      B/1,    J/IOT740
; 7555           I,      B/2,    J/IOT740
; 7556           I,      B/3,    J/IOT740
; 7557           I,      B/4,    J/IOT740
; 7558           I,      B/5,    J/IOT740
; 7559           I,      B/6,    J/IOT740
; 7560           I,      B/7,    J/IOT740
; 7561
; 7562  750:     I,      B/0,    J/IOT750
; 7563           I,      B/1,    J/IOT750
; 7564           I,      B/2,    J/IOT750
; 7565           I,      B/3,    J/IOT750
; 7566           I,      B/4,    J/IOT750
; 7567           I,      B/5,    J/IOT750
; 7568           I,      B/6,    J/IOT750
; 7569           I,      B/7,    J/IOT750
; 7570
; 7571  760:     I,      B/0,    J/IOT760
; 7572           I,      B/1,    J/IOT760
; 7573           I,      B/2,    J/IOT760
; 7574           I,      B/3,    J/IOT760
; 7575           I,      B/4,    J/IOT760
; 7576           I,      B/5,    J/IOT760
; 7577           I,      B/6,    J/IOT760
; 7578           I,      B/7,    J/IOT760
; 7579
```

```
D 0770, 0000,1657,2100
D 0771, 0001,1657,2100
D 0772, 0002,1657,2100
D 0773, 0003,1657,2100
D 0774, 0004,1657,2100
D 0775, 0005,1657,2100
D 0776, 0006,1657,2100
D 0777, 0007,1657,2100


U 1650, 2740,4551,0202,4374,0007,0700,0000,0077,7740

U 1651, 2740,4551,0202,4374,0007,0700,0000,0077,7740

U 1652, 2740,4551,0202,4374,0007,0700,0000,0077,7740

U 1653, 2740,4551,0202,4374,0007,0700,0000,0077,7740

U 1654, 2740,4551,0202,4374,0007,0700,0000,0077,7740

U 1655, 2740,4551,0202,4374,0007,0700,0000,0077,7740

U 1656, 2740,4551,0202,4374,0007,0700,0000,0077,7740

U 1657, 2740,4551,0202,4374,0007,0700,0000,0077,7740
```

```
; 7580  770:    I,      B/0,    J/IOT770
; 7581           I,      B/1,    J/IOT770
; 7582           I,      B/2,    J/IOT770
; 7583           I,      B/3,    J/IOT770
; 7584           I,      B/4,    J/IOT770
; 7585           I,      B/5,    J/IOT770
; 7586           I,      B/6,    J/IOT770
; 7587           I,      B/7,    J/IOT770
; 7588           .UCODE
; 7589
; 7590  1650:
; 7591  IOT700: UUO
; 7592  1651:
; 7593  IOT710: UUO
; 7594  1652:
; 7595  IOT720: UUO
; 7596  1653:
; 7597  IOT730: UUO
; 7598  1654:
; 7599  IOT740: UUO
; 7600  1655:
; 7601  IOT750: UUO
; 7602  1656:
; 7603  IOT760: UUO
; 7604  1657:
; 7605  IOT770: UUO
; 7606
```

```
D 0704, 1200,1754,0100
D 0705, 1200,1755,0100

U 1754, 3673,3443,0300,4174,4207,0700,0200,0004,0012
U 3673, 1515,3771,0003,4365,5007,0700,0200,0000,0002
U 1755, 3674,3443,0300,4174,4207,0700,0200,0003,0012
U 3674, 1516,3771,0003,0276,6007,0700,0000,0000,0000
```

```
; 7607  .TOC    "UMOVE AND UMOVEM"
; 7608
; 7609          .DCODE
; 7610  704:    IOT,     J/UMOVE
; 7611          IOT,     J/UMOVEM
; 7612          .UCODE
; 7613
; 7614  1754:
; 7615  UMOVE:  VMA_[AR],                ;LOAD VMA
; 7616          START READ,              ;START MEMORY
; 7617          SPEC/PREV                ;FORCE PREVIOUS
; 7618          MEM READ,                ;WAIT FOR MEMORY
; 7619          [AR]_MEM,                ;PUT DATA IN AR
; 7620          J/STAC                   ;GO PUT AR IN AC
; 7621
; 7622  1755:
; 7623  UMOVEM: VMA_[AR],                ;LOAD VMA
; 7624          START WRITE,             ;START MEMORY
; 7625          SPEC/PREV                ;FORCE PREVIOUS
; 7626          [AR]_AC,                 ;FETCH AC
; 7627          J/STMEM                  ;STORE IN MEMORY
; 7628
```

```
U 0104, 3677,3333,0004,7174,4007,0700,0410,0000,0212
U 0114, 3676,4223,0000,4364,4277,0700,0210,0000,0010
U 0116, 3675,4221,0004,4174,4007,0700,0200,0021,1016
U 3675, 2476,3333,0017,4175,5007,0701,0200,0000,0002
U 2476, 3671,0111,0704,4170,4007,0700,0210,0023,1016
U 2477, 0005,4443,0000,4174,4107,0700,0000,0000,0074


U 0004, 2500,4443,0000,4174,4107,0640,0000,0000,0062
U 0005, 0004,4443,0000,4174,4007,0660,0000,0000,0000



U 2500, 0117,3443,0100,4174,4007,0700,0200,0014,0012
U 2501, 2502,4571,1203,4374,4007,0700,0000,0024,1200


U 2502, 3667,3551,0303,4370,4007,0700,0010,0020,0000



U 2503, 2717,3771,0002,4365,5617,0700,0200,0000,0002
```

```
; 7629  ;HERE WITH HALT CODE IN THE T1
; 7630  =010*
; 7631  HALTED: WORK[SV.ARX]_[ARX],      ;SAVE TEMP REGISTER
; 7632          CALL [SAVVMA]           ;PUT VMA IN WORK[SV.VMA]
; 7633  =110*   ABORT MEM CYCLE,                ;ABORT CYCLE IN PROGRESS
; 7634          CALL [WRTHSB]           ;WRITE HALT STATUS BLOCK
; 7635  =111*
; 7636  PWRON:  [ARX]_0, VMA PHYSICAL WRITE ;STORE HALT CODE
; 7637  =
; 7638          MEM WRITE, MEM_[T1]     ; IN LOCATION 0
; 7639  =0      NEXT [ARX] PHYSICAL WRITE,
; 7640          CALL [STOPC]
; 7641  H1:     SET HALT, J/HALTLP      ;TELL CONSOLE WE HAVE HALTED
; 7642
; 7643
; 7644  4:      UNHALT,                 ;RESET CONSOLE
; 7645          SKIP EXECUTE, J/CONT    ;SEE IF CO OR EX
; 7646  5:
; 7647  HALTLP: SKIP/-CONTINUE, J/4     ;WAIT FOR CONTINUE
; 7648
; 7649  =0
; 7650  CONT:   VMA_[PC],               ;LOAD PC INTO VMA
; 7651          FETCH,                  ;START READ
; 7652          J/XCTGO                 ;DO THE INSTRUCTION
; 7653          [AR]_VMA IO READ        ;PUT FLAGS IN AR
; 7654  =0      [AR]_[AR].OR.#,         ;PUT IN ADDRESS
; 7655          #/200000, HOLD LEFT,    ; OF CSL REGISTER
; 7656          CALL [STRTIO]
; 7657  CONT1:  MEM READ,               ;WAIT FOR DATA
; 7658          [HR]_MEM,               ;PUT IN HR
; 7659          LOAD INST,              ;LOAD IR, ETC.
; 7660          J/XCT1                  ;GO DO THE INSTRUCTION
; 7661
```

```
; 7662  .TOC    "WRITE HALT STATUS BLOCK"
; 7663
; 7664  ;THE HALT STATUS BLOCK LOOKS LIKE:
; 7665
; 7666  ;       !=========================================================!
; 7667  ;       !00!                          MAG                        !
; 7668  ;       !--------------------------------------------------------!
; 7669  ;       !01!                           PC                        !
; 7670  ;       !--------------------------------------------------------!
; 7671  ;       !02!                           HR                        !
; 7672  ;       !--------------------------------------------------------!
; 7673  ;       !03!                           AR                        !
; 7674  ;       !--------------------------------------------------------!
; 7675  ;       !04!                          ARX                        !
; 7676  ;       !--------------------------------------------------------!
; 7677  ;       !05!                           BR                        !
; 7678  ;       !--------------------------------------------------------!
; 7679  ;       !06!                          BRX                        !
; 7680  ;       !--------------------------------------------------------!
; 7681  ;       !07!                          ONE                        !
; 7682  ;       !--------------------------------------------------------!
; 7683  ;       !10!                          EBR                        !
; 7684  ;       !--------------------------------------------------------!
; 7685  ;       !11!                          UBR                        !
; 7686  ;       !--------------------------------------------------------!
; 7687  ;       !12!                          MASK                       !
; 7688  ;       !--------------------------------------------------------!
; 7689  ;       !13!                          FLG                        !
; 7690  ;       !--------------------------------------------------------!
; 7691  ;       !14!                           PI                        !
; 7692  ;       !--------------------------------------------------------!
; 7693  ;       !15!                          XWD1                       !
; 7694  ;       !--------------------------------------------------------!
; 7695  ;       !16!                           TO                        !
; 7696  ;       !--------------------------------------------------------!
; 7697  ;       !17!                           T1                        !
; 7698  ;       !=========================================================!
; 7699  ;       !          VMA FLAGS          !            VMA            !
; 7700  ;       !=========================================================!
; 7701
```

```
U 0001, 3677,3333,0004,7174,4007,0700,0410,0000,0212
U 0011, 0024,3771,0004,7274,4007,0701,0000,0000,0227
U 0024, 3701,4223,0000,4364,4277,0700,0210,0000,0010
U 0026, 2477,4443,0000,4174,4107,0700,0000,0000,0074



U 3676, 2504,3771,0004,7274,4007,0422,0000,0000,0227

U 2504, 3701,3333,0012,4174,4437,0700,0000,0000,0000

U 2505, 0002,3771,0004,7274,4004,1701,0000,0000,0212

U 3677, 3700,3771,0004,4354,4007,0700,0000,0000,0000

U 3700, 0010,3333,0004,7174,4004,1700,0400,0000,0210
```

WRITE HALT STATUS BLOCK

```
; 7702  ;START AT 1 TO DUMP 2901 REGISTERS INTO MAIN MEMORY
; 7703  1:       WORK[SV.ARX]_[ARX],      ;SAVE TEMP REGISTER
; 7704           CALL [SAVVMA]            ;WORK[SV.VMA]_VMA
; 7705  11:      [ARX]_WORK[HSBADR]
; 7706  =10*     ABORT MEM CYCLE, CALL [DUMP]
; 7707           SET HALT, J/H1
; 7708
; 7709
; 7710  WRTHSB: [ARX]_WORK[HSBADR], ;GET ADDRESS OF HSB
; 7711           SKIP AD.LE.0, 4T         ;SEE IF VALID
; 7712  =0       READ [MASK], LOAD PI,    ;TURN OFF PI SYSTEM
; 7713           J/DUMP                   ; AND GO TAKE DUMP
; 7714           [ARX]_WORK[SV.ARX],
; 7715           RETURN [2]               ;DO NOT DUMP ANYTHING
; 7716
; 7717  SAVVMA: [ARX]_VMA
; 7718           WORK[SV.VMA]_[ARX],
; 7719           RETURN [10]
; 7720
```

```
U 3701, 2510,3333,0004,4174,4007,0700,0200,0021,1016    ; 7721  ;DUMP OUT THE 2901
U 2510, 2747,3333,0000,4175,5007,0701,0210,0000,0002    ; 7722  DUMP:   READ [ARX], VMA PHYSICAL WRITE
U 2512, 3702,3333,0001,4175,5007,0701,0200,0000,0002    ; 7723  =O*     MEM WRITE, MEM_[MAG], CALL [NEXT]
U 3702, 2511,0111,0704,4170,4007,0700,0200,0023,1016    ; 7724          MEM WRITE, MEM_[PC]
U 2511, 2747,3333,0002,4175,5007,0701,0210,0000,0002    ; 7725          NEXT [ARX] PHYSICAL WRITE
U 2513, 2514,3333,0003,4175,5007,0701,0200,0000,0002    ; 7726  =O*     MEM WRITE, MEM_[HR], CALL [NEXT]
U 2514, 2747,3333,0003,7174,4007,0700,0410,0000,0211    ; 7727          MEM WRITE, MEM_[AR]
U 2516, 2515,3771,0003,7274,4007,0701,0000,0000,0212    ; 7728  =O*     WORK[SV.AR]_[AR], CALL [NEXT]
U 2515, 2747,3333,0003,4175,5007,0701,0210,0000,0002    ; 7729          [AR]_WORK[SV.ARX]
U 2517, 3703,3333,0005,4175,5007,0701,0200,0000,0002    ; 7730  =O*     MEM WRITE, MEM_[AR], CALL [NEXT]
U 3703, 2520,0111,0704,4170,4007,0700,0200,0023,1016    ; 7731          MEM WRITE, MEM_[BR]
U 2520, 2747,3333,0006,4175,5007,0701,0210,0000,0002    ; 7732          NEXT [ARX] PHYSICAL WRITE
U 2522, 3704,3333,0007,4175,5007,0701,0200,0000,0002    ; 7733  =O*     MEM WRITE, MEM_[BRX], CALL [NEXT]
U 3704, 2521,0111,0704,4170,4007,0700,0200,0023,1016    ; 7734          MEM WRITE, MEM_[ONE]
U 2521, 2747,3333,0010,4175,5007,0701,0210,0000,0002    ; 7735          NEXT [ARX] PHYSICAL WRITE
U 2523, 3705,3333,0011,4175,5007,0701,0200,0000,0002    ; 7736  =O*     MEM WRITE, MEM_[EBR], CALL [NEXT]
U 3705, 2524,0111,0704,4170,4007,0700,0200,0023,1016    ; 7737          MEM WRITE, MEM_[UBR]
U 2524, 2747,3333,0012,4175,5007,0701,0210,0000,0002    ; 7738          NEXT [ARX] PHYSICAL WRITE
U 2526, 3706,3333,0013,4175,5007,0701,0200,0000,0002    ; 7739  =O*     MEM WRITE, MEM_[MASK], CALL [NEXT]
U 3706, 2525,0111,0704,4170,4007,0700,0200,0023,1016    ; 7740          MEM WRITE, MEM_[FLG]
U 2525, 2747,3333,0014,4175,5007,0701,0210,0000,0002    ; 7741          NEXT [ARX] PHYSICAL WRITE
U 2527, 3707,3333,0015,4175,5007,0701,0200,0000,0002    ; 7742  =O*     MEM WRITE, MEM_[PI], CALL [NEXT]
U 3707, 2530,0111,0704,4170,4007,0700,0200,0023,1016    ; 7743          MEM WRITE, MEM_[XWD1]
U 2530, 2747,3333,0016,4175,5007,0701,0210,0000,0002    ; 7744          NEXT [ARX] PHYSICAL WRITE
U 2532, 2531,3333,0017,4175,5007,0701,0200,0000,0002    ; 7745  =O*     MEM WRITE, MEM_[TO], CALL [NEXT]
U 2531, 2747,3771,0003,7274,4007,0701,0010,0000,0210    ; 7746          MEM WRITE, MEM_[T1]
U 2533, 3710,3333,0003,4175,5007,0701,0200,0000,0002    ; 7747  =O*     [AR]_WORK[SV.VMA], CALL [NEXT]
U 3710, 3711,3771,0003,7274,4007,0701,0000,0000,0211    ; 7748          MEM WRITE, MEM_[AR]
U 3711, 3712,3771,0004,7274,4007,0701,0000,0000,0210    ; 7749  HSBDON: [AR]_WORK[SV.AR]
U 3712, 3713,3443,0400,4174,4007,0700,0200,0000,0010    ; 7750          [ARX]_WORK[SV.VMA]
                                                        ; 7751          VMA_[ARX]
                                                        ; 7752          [ARX]_WORK[SV.ARX],
U 3713, 0006,3771,0004,7274,4004,1701,0000,0000,0212    ; 7753          RETURN [6]
                                                        ; 7754
```

```
; 7755          .NOBIN
; 7756          .TOC    "PAGE FAIL REFIL LOGIC"
; 7757
; 7758  ;WHEN THE CPU CAN NOT COMPLETE A MEMORY REFERENCE BECAUSE THE PAGE
; 7759  ; TABLE DOES NOT CONTAIN VALID INFORMATION FOR THE VIRTUAL PAGE INVOLVED
; 7760  ; THE HARDWARE CALLS THIS ROUTINE TO RELOAD THE HARDWARE PAGE TABLE.
; 7761  ;
; 7762  ;THIS CODE WILL EITHER DO THE RELOAD OR GENERATE A PAGE FAIL FOR THE
; 7763  ; SOFTWARE. THE INFORMATION LOADED CONSISTS OF THE PHYSICAL PAGE NUMBER,
; 7764  ; THE CACHE ENABLE BIT AND THE WRITE ENABLE BIT.
; 7765
; 7766  ;THIS LOGIC USES MANY VARIABLES. THEY ARE DESCRIBED BRIEFLY HERE:
; 7767
; 7768  ;THING                   WHERE KEPT                  USE
; 7769  ;OLD VMA                 WORKSPACE WORD 210          SAVES VMA
; 7770  ;OLD AR                  WORKSPACE WORD 211          SAVES AR
; 7771  ;OLD ARX                 WORKSPACE WORD 212          SAVES ARX
; 7772  ;OLD BR                  WORKSPACE WORD 213          SAVES BR
; 7773  ;OLD BRX                 WORKSPACE WORD 214          SAVES BRX
; 7774  ;KL PAGING BIT           EBR BIT 1 (IN 2901)         INDICATES KL STYLE (TOPS-20) PAGING
; 7775  ;                                                    INSTEAD OF KI STYLE (TOPS-10 AND DIAGNOSTIC)
; 7776  ;                                                    MODE PAGING
; 7777  ;W BIT                   FLG BIT 4                   PAGE CAN BE WRITTEN
; 7778  ;C BIT                   FLG BIT 6                   DATA IN THIS PAGE MAY BE PUT
; 7779  ;                                                    INTO CACHE
; 7780  ;PI CYCLE                FLG BIT 5                   STORING OLD PC DURING PI
; 7781  ;MAP FLAG                FLG BIT 18                  MAP INSTRUCTION IN PROGRESS
; 7782  ;CLEANUP CODE            FLG BITS 32-35              WHAT TO DO SO INSTRUCTION MAY BE
; 7783  ;                                                    RESTARTED
; 7784  ;SPT BASE                WORKSPACE WORD 215          ADDRESS OF SHARED-POINTER-TABLE
; 7785  ;CST BASE                WORKSPACE WORD 216          ADDRESS OF CORE-STATUS-TABLE
; 7786  ;CST MASK                WORKSPACE WORD 217          BITS TO KEEP ON CST UPDATE
; 7787  ;CST DATA (PUR)          WORKSPACE WORD 220          BITS TO SET ON CST UPDATE
; 7788  ;PAGE TABLE ADDRESS      AR                          WHERE THIS PAGE TABLE IS LOCATED
; 7789  ;PHYSICAL PAGE # (PPN)   AR                          RESULT OF THIS PROCESS
; 7790  ;CST ENTRY               AR                          CORE STATUS TABLE ENTRY
; 7791  ;SPT ENTRY               AR                          WORD FROM SPT
; 7792  ;PAGE TABLE ENTRY        AR                          WORD FROM PT
; 7793  ;PAGE NUMBER             BR                          INDEX INTO CURENT PAGE TABLE
; 7794  ;PAGE FAIL WORD          BRX                         WHAT HAPPENED (ALSO MAP RESULT)
; 7795
```

```
; 7796  ;
; 7797  ;
; 7798  ;
; 7799  ;                    KL10 PAGING - WORD FORMATS
; 7800  ;
; 7801  ;Section Pointer
; 7802  ;
; 7803  ;The section pointer is found in the user or exec section table.
; 7804  ;(Part of UPT or EPT.)
; 7805  ;
; 7806  ;Section pointer provides (via the SPT) the physical address  of
; 7807  ;the PAGE TABLE for the given section.
; 7808  ;
; 7809  ;         Code:   0       No-access (trap)
; 7810  ;                 1       Immediate
; 7811  ;                 2       Share
; 7812  ;                 3       Indirect
; 7813  ;                 4-7     Unused, reserved
; 7814  ;
; 7815  ;         0 1 2 3 4 5 6           18                        35
; 7816  ;         +---+-+-+-+-+---------+-------------------------+
; 7817  ;         !CODE!P!W! !C!/////////!  PAGE TABLE IDENTIFIER  !
; 7818  ;         !010 ! ! ! ! !/////////!        (SPT INDEX)      !
; 7819  ;         +---+-+-+-+-+---------+-------------------------+
; 7820  ;
; 7821  ;                NORMAL SECTION POINTER (Code = 2)
; 7822  ;
; 7823  ;         0   2 3 4 5 6     9           18                       35
; 7824  ;         +---+-+-+-+-+---+-----------+-----------------------+
; 7825  ;         !CODE!P!W! !C!///!SECTION    !SECTION TABLE IDENTIFIER!
; 7826  ;         !011 ! ! ! ! !///!TABLE INDEX!       (SPT INDEX)      !
; 7827  ;         +---+-+-+-+-+---+-----------+-----------------------+
; 7828  ;
; 7829  ;               INDIRECT SECTION POINTER (Code = 3)
; 7830
```

```
; 7831  ;PAGE POINTERS
; 7832  ;
; 7833  ;FOUND IN PAGE TABLES
; 7834  ;
; 7835  ;        0 1 2 3 4 5 6      12                              35
; 7836  ;        +----+-+-+-+-+----+-------------------------------+
; 7837  ;        !CODE!P!W! !C!////!   PHYSICAL ADDRESS OF PAGE    !
; 7838  ;        !001 ! ! ! ! !////!                                !
; 7839  ;        +----+-+-+-+-+----+-------------------------------+
; 7840  ;
; 7841  ;                IMMEDIATE POINTER (code field = 1)
; 7842  ;
; 7843  ;        B12-35  give PHYSICAL ADDRESS OF PAGE
; 7844  ;            if  B12-17 >< 0, page not in core-trap
; 7845  ;            if  B12-17 =  0, B23-35 give CORE PAGE
; 7846  ;                             NUMBER of page, B18-22 MBZ
; 7847  ;
; 7848  ;
; 7849  ;
; 7850  ;
; 7851  ;
; 7852  ;        0    2 3     6           18                        35
; 7853  ;        +-----+-------+---------+-------------------------+
; 7854  ;        !CODE !SAME AS!/////////!        SPT INDEX        !
; 7855  ;        !010  ! IMMED.!/////////!                         !
; 7856  ;        +-----+-------+---------+-------------------------+
; 7857  ;
; 7858  ;                SHARED POINTER (code field = 2)
; 7859  ;
; 7860  ;        B18-35  Give SPT INDEX (SPTX).  SPTX + SPT BASE
; 7861  ;                ADDRESS = physical core address of word
; 7862  ;                holding physical address of page.
; 7863
```

```
; 7864  ;        0 1 2 3      6     9    17 18                    35
; 7865  ;        +----+--------+---+-------+----------------------+
; 7866  ;        !CODE!SAME AS !///! PAGE  ! PAGE TABLE IDENTIFIER!
; 7867  ;        !011 ! IMMED. !///!NUMBER !     (SPT INDEX)      !
; 7868  ;        +----+--------+---+-------+----------------------+
; 7869  ;
; 7870  ;                INDIRECT POINTER (code field = 3)
; 7871  ;
; 7872  ;        This pointer type causes another pointer to be  fetched
; 7873  ;        and  interpreted.   The  new pointer is found in word N
; 7874  ;        (B9-17) of the page addressed by C(SPT + SPTX).
; 7875  ;
; 7876  ;
; 7877  ;
; 7878  ;        SPT ENTRY
; 7879  ;
; 7880  ;        Found in the SPT, i.e., when fetching C(SPT +SPTX)
; 7881  ;
; 7882  ;                              12                        35
; 7883  ;        +--------------------+-------------------------+
; 7884  ;        !////////////////////!  PHYSICAL ADDRESS OF PAGE !
; 7885  ;        !////////////////////!          OR PAGE TABLE    !
; 7886  ;        +--------------------+-------------------------+
; 7887  ;
; 7888  ;                B12-35  Give PHYSICAL ADDRESS of page.
; 7889  ;
; 7890  ;        The base address (physical core  address)  of  the  SPT
; 7891  ;        resides in one AC of the reserved AC block.
; 7892
```

```
; 7893  ;PHYSICAL STORAGE ADDRESS
; 7894  ;
; 7895  ;Found in B12-35 of IMMEDIATE POINTERS and SPT ENTRIES.
; 7896  ;
; 7897  ;                        12      17 18   23              35
; 7898  ;                        +---------+----+----------------+
; 7899  ;                        !         !MBZ ! CORE PAGE NUMBER!
; 7900  ;                        !         !    !   IF B12-17 = 0 !
; 7901  ;                        +---------+----+----------------+
; 7902  ;
; 7903  ;       If B12-17 = 0, then B23-35 are CORE PAGE NUMBER  (i.e.,
; 7904  ;       B14-26  of  physical  core  address) of page and B18-22
; 7905  ;       MBZ.  If B12-17 >< 0, then  address  is  not  core  and
; 7906  ;       pager traps.
; 7907  ;
; 7908  ;
; 7909  ;
; 7910  ;CORE STATUS TABLE ENTRY
; 7911  ;
; 7912  ;Found when fetching C(CBR + CORE PAGENO)
; 7913  ;
; 7914  ;       0      5                                  32  34 35
; 7915  ;       +-------+----------------------------------+------+-+
; 7916  ;       !  CODE !                                  !      !M!
; 7917  ;       +-------+----------------------------------+------+-+
; 7918  ;
; 7919  ;       B0-5    are code field:
; 7920  ;
; 7921  ;               0 - unavailable, trap
; 7922  ;
; 7923  ;               1-77 - available
; 7924  ;
; 7925  ;
; 7926  ;
; 7927  ;       B32-34 reserved for future hardware specification.
; 7928  ;
; 7929  ;       B35 is "modified" bit, set on any write ref to page.
; 7930
```

```
; 7931  ;QUANTITIES IN HARDWARE REGISTERS
; 7932  ;
; 7933  ;SPT     SPT Base Register
; 7934  ;
; 7935  ;                 14                                35
; 7936  ;                 +---------------------------------+
; 7937  ;                 !   PHYSICAL CORE WORD ADDRESS    !
; 7938  ;                 +---------------------------------+
; 7939  ;
; 7940  ;CBR     CST Base Register
; 7941  ;
; 7942  ;                 14                                35
; 7943  ;                 +---------------------------------+
; 7944  ;                 !   PHYSICAL CORE WORD ADDRESS    !
; 7945  ;                 +---------------------------------+
; 7946  ;
; 7947  ;CSTMSK  CST Update Mask
; 7948  ;
; 7949  ;        0                                        32  35
; 7950  ;        +----------------------------------------+---+-+
; 7951  ;        !                  MASK                  !111!1!
; 7952  ;        +----------------------------------------+---+-+
; 7953  ;
; 7954  ;                ANDed with CST word during update
; 7955  ;
; 7956  ;(B32-35 must be all 1's to preserve existing CST information)
; 7957  ;
; 7958  ;CSTDATA CST Update Data
; 7959  ;
; 7960  ;        0                                    32 34 35
; 7961  ;        +----------------------------------------+---+-+
; 7962  ;        !                  DATA                  !000!0!
; 7963  ;        +----------------------------------------+---+-+
; 7964  ;
; 7965  ;                IORed with CST word during update
; 7966  ;
; 7967  ;(B32-35 must be all 0's to preserve existing CST information)
; 7968  ;
; 7969  ;All unspecified bits and fields are reserved for future
; 7970  ;specification by DEC.
; 7971  ;
; 7972
```

```
D 0257, 1215,1553,0100

U 1553, 3714,3551,0303,4374,0007,0700,0000,0016,0000
U 3714, 3715,3771,0006,4354,4007,0700,0000,0000,0000

U 3715, 3716,4551,0606,4374,0007,0700,0000,0040,0000
U 3716, 3717,3333,0006,7174,4007,0700,0400,0000,0210
U 3717, 3720,3771,0005,7274,4007,0701,0000,0000,0230
U 3720, 2506,4553,0500,4374,4007,0331,0000,0003,0000
U 2506, 3730,3771,0013,4370,4007,0700,0000,0040,0002
U 2507, 0100,3440,0303,0174,4156,4700,0400,0000,0000

U 3777, 3721,3333,0003,7174,4007,0700,0400,0000,0211
U 3721, 3722,3333,0006,7174,4007,0700,0400,0000,0214
U 3722, 3723,3771,0006,4354,4007,0700,0000,0000,0000
U 3723, 3724,3333,0006,7174,4007,0700,0400,0000,0210

U 3724, 1060,3333,0004,7174,4007,0370,0400,0000,0212

U 1060, 1060,3773,0000,4304,4003,1702,0000,0000,0000

U 1061, 1507,3333,0005,7174,4007,0700,0400,0000,0213

U 1063, 3725,3771,0006,4374,4007,0700,0000,0000,0000

U 1065, 1502,4571,1206,4374,4007,0700,0000,0037,0000

U 1067, 1502,4571,1206,4374,4007,0700,0000,0037,0000

U 1070, 3730,3333,0005,7174,4007,0700,0400,0000,0213

U 1071, 1507,3333,0005,7174,4007,0700,0400,0000,0213

U 1072, 3730,3333,0005,7174,4007,0700,0400,0000,0213

U 1073, 3730,3333,0005,7174,4007,0700,0400,0000,0213

U 3725, 3726,3333,0006,7174,4007,0700,0400,0000,0160
U 3726, 3727,3333,0006,7174,4007,0700,0400,0000,0161
```

```
; 7973          .BIN
; 7974
; 7975          .DCODE
; 7976  257:     IOT,     AC,      J/MAP
; 7977          .UCODE
; 7978
; 7979  1553:
; 7980  MAP:     [AR]_[AR].OR.#,          ;ASSUME PHYSICAL REF
; 7981           #/160000,                ;FAKE ANSWER
; 7982           HOLD RIGHT               ; ..
; 7983           [BRX]_VMA                ;PUT VMA AND FLAGS IN BRX
; 7984           [BRX]_[BRX].AND.#,       ;JUST KEEP USER BIT
; 7985           #/400000, HOLD RIGHT     ; ..
; 7986           WORK[SV.VMA]_[BRX]       ;SAVE IN WORKSPACE
; 7987           [BR]_WORK[APR]           ;GET APR FLAGS
; 7988           TR [BR], #/030000        ;PAGING ENABLED?
; 7989  =0       STATE_[MAP], J/PFMAP     ;YES--DO REAL MAP
; 7990           AC_[AR], NEXT INST       ;NO--RETURN VIRTUAL ADDRESS
; 7991  ;HARDWARE COMES HERE ON PAGE TABLE NOT VALID (OR INTERRUPT) WHEN
; 7992  ; STARTING A MEMORY REFERENCE. MICOWORD ADDRESS OF INSTRUCTION DOING
; 7993  ; MEM WAIT IS SAVED ON THE STACK.
; 7994  3777:
; 7995  PAGE-FAIL:
; 7996           WORK[SV.AR]_[AR]
; 7997  ITRAP:  WORK[SV.BRX]_[BRX]
; 7998           [BRX]_VMA
; 7999           WORK[SV.VMA]_[BRX]
; 8000           WORK[SV.ARX]_[ARX],
; 8001           SKIP IRPT                ;SEE IF INTERRUPT (SAVE DISPATCH)
; 8002  =0000
; 8003  PFD:     DBM/PF DISP, DBUS/DBM,   ;BRING CODE TO 2901'S
; 8004           AD/D, DEST/PASS, 4T,     ;PUT ON DP 18-21
; 8005           DISP/DP LEFT, J/PFD      ;DISPATCH ON IT
; 8006  =0001                            ;(1) INTERRUPT
; 8007           WORK[SV.BR]_[BR], J/PFPI1
; 8008  =0011                            ;(3) BAD DATA FROM MEMORY
; 8009           [BRX]_IO DATA,           ;GET THE BAD DATA
; 8010           AD PARITY OK/0,          ; DO NOT LOOK AT PARITY
; 8011           J/BADDATA                ;SAVE IN AC BLK 7
; 8012  =0101                            ;(5) NXM ERROR
; 8013           [BRX]_[370000] XWD 0, J/HARD
; 8014  =0111                            ;(7) NXM & BAD DATA
; 8015           [BRX]_[370000] XWD 0, J/HARD
; 8016  =1000                            ;(10) WRITE VIOLATION
; 8017           WORK[SV.BR]_[BR], J/PFMAP
; 8018  =1001                            ;[123] (11) 1 ms timer and movsrj
; 8019           WORK[SV.BR]_[BR], J/PFPI1
; 8020  =1010                            ;(12) PAGE NOT VALID
; 8021           WORK[SV.BR]_[BR], J/PFMAP
; 8022  =1011                            ;(13) EXEC/USER MISMATCH
; 8023           WORK[SV.BR]_[BR], J/PFMAP
; 8024  =
; 8025
; 8026  BADDATA:
; 8027           WORK[BADW0]_[BRX]        ;SAVE BAD WORD
; 8028           WORK[BADW1]_[BRX]        ;AGAIN
```

```
U 3727,  1502,4571,1206,4374,4007,0700,0000,0036,0000    ; 8029          [BRX]_[360000] XWD 0, J/HARD
                                                         ; 8030
```

```
U 3730, 3731,4223,0000,4364,4277,0700,0200,0000,0010
U 3731, 3732,3551,1313,4374,0007,0700,0000,0002,4000
U 3732, 2534,4553,0600,4374,4007,0321,0000,0002,0000
U 2534, 2535,3551,0606,4374,0007,0700,0000,0001,0000
U 2535, 3733,4551,0606,4374,0007,0700,0000,0041,1000
U 3733, 3734,6551,0606,4374,0007,0700,0000,0000,1000
U 3734, 2536,3441,0605,4174,4007,0700,2000,0071,0007
U 2536, 2536,3447,0505,4174,4007,0630,2000,0060,0000
U 2537, 3735,4251,0505,4374,4007,0700,0000,0000,0777
U 3735, 2540,4553,1000,4374,4007,0321,0000,0000,0040
U 2540, 2542,3333,0006,4174,4007,0520,0000,0000,0000
U 2541, 3775,3447,0504,4174,4007,0700,0000,0000,0000
```

```
; 8031  ;WE HAVE SAVED AR, ARX, BR AND BRX. WE MERGE IN HERE FROM MAP
; 8032  ; INSTRUCTION, SAVE THE VMA AND START THE PAGE FAIL WORD.
; 8033  PFMAP:  ABORT MEM CYCLE          ;CLEAR PAGE FAIL
; 8034          [FLG]_[FLG].OR.#,       ;PRESET W AND C TO 1
; 8035          FLG.W/1, FLG.C/1,       ;BITS INVOLVED
; 8036          HOLD RIGHT              ;LEAVE RH ALONE
; 8037          TL [BRX], WRITE TEST/1  ;IS THIS A WRITE TEST?
; 8038  =0      [BRX]_[BRX].OR.#,
; 8039          #/10000,
; 8040          HOLD RIGHT              ;YES--TURN INTO WRITE REF
; 8041          [BRX]_[BRX].AND.#,      ;START PAGE FAIL WORD
; 8042          #/411000,               ;SAVE 3 INTERESTING BITS
; 8043          HOLD RIGHT              ;SAVE VIRTUAL ADDRESS
; 8044                                  ;USER ADDR (400000)
; 8045                                  ;WRITE REF (010000)
; 8046                                  ;PAGED REF (001000)
; 8047          [BRX]_[BRX].XOR.#,      ;FIX BIT 8
; 8048          #/1000, HOLD RIGHT
; 8049          [BR]_[BRX],             ;COPY VIRTUAL ADDRESS
; 8050          SC_7                    ;PREPARE TO SHIFT 9 PLACES
; 8051  =0
; 8052  PF25:   [BR]_[BR]*.5,           ;RIGHT ADJUST PAGE #
; 8053          STEP SC,                ;COUNT SHIFT STEPS
; 8054          J/PF25                  ;LOOP FOR 9
; 8055          [BR]_[BR].AND.# CLR LH, ;MASK TO 9 BITS
; 8056          #/777                   ; ..
; 8057          TL [EBR],               ;KI MODE REFILL?
; 8058          #/40                    ;FLAG BIT
; 8059  =0      READ [BRX],             ;USER REF? (KL MODE)
; 8060          SKIP DPO,               ; ..
; 8061          J/PF30                  ;CONTINUE AT PF30
; 8062          [ARX]_[BR]*.5,          ;KI10 MODE REFILL
; 8063          J/KIFILL                ;GO HANDLE EASY CASE
; 8064
```

```
U 2542, 2546,3771,0004,7274,4007,0622,0000,0000,0423
U 2543, 2544,3771,0004,7274,4007,0622,0000,0000,0424

U 2544, 1130,0113,0405,4174,4007,0700,0200,0024,1016

U 2545, 3736,0551,1103,4374,4007,0701,0000,0000,0540

U 2546, 1130,0113,0405,4174,4007,0700,0200,0024,1016
U 2547, 3736,0551,1003,4374,4007,0701,0000,0000,0540

U 3736, 3737,3443,0300,4174,4007,0700,0200,0024,1016
U 3737, 1000,3771,0003,4365,5007,0700,0200,0000,0002

U 1000, 3770,4443,0000,4174,4007,0700,2010,0071,0007

U 1001, 2554,4553,0300,4374,4007,0321,0000,0000,0077
U 1002, 2552,0551,0303,7274,4007,0701,0000,0000,0215
U 1003, 3773,0551,0303,7274,4007,0701,0010,0000,0215
U 1007, 2550,4553,0300,4374,4007,0321,0000,0000,0077
U 2550, 2670,4553,1300,4374,4007,0321,0000,0001,0000
U 2551, 0602,3445,0303,4174,4007,0630,2000,0060,0000

U 0602, 0602,3445,0303,4174,4007,0630,2000,0060,0000
U 0603, 3773,4443,0000,4174,4007,0700,0010,0000,0000
U 0607, 1000,4443,0000,4174,4007,0700,0000,0000,0000
```

```
; 8065  ;HERE IN TOPS-20 MODE
; 8066  ;PICK UP CORRECT SECTION POINTER
; 8067  =0
; 8068  PF30:   [ARX]_WORK[PTA.E],      ;EXEC MODE
; 8069          SKIP AD.EQ.0, 4T,       ;SEE IF VALID
; 8070          J/PF35                  ;CONTINUE BELOW
; 8071          [ARX]_WORK[PTA.U],      ;USER MODE
; 8072          SKIP AD.EQ.0, 4T        ;SEE IF VALID
; 8073  =0      VMA_[ARX]+[BR],         ;POINTER VALID
; 8074          VMA PHYSICAL READ,      ;START MEMORY
; 8075          J/PF77                  ;CONTINUE BELOW
; 8076          [AR]_[UBR]+#, 3T,       ;USER MODE
; 8077          #/540,                  ;OFFSET TO UPT
; 8078          J/PF40                  ;GO GET POINTER
; 8079
; 8080  =0
; 8081  PF35:   VMA_[ARX]+[BR],         ;POINTER VALID
; 8082          VMA PHYSICAL READ,      ;START MEMORY
; 8083          J/PF77                  ;CONTINUE BELOW
; 8084          [AR]_[EBR]+#, 3T,       ;EXEC MODE
; 8085          #/540                   ;OFFSET TO EPT
; 8086  PF40:   VMA_[AR],                ;LOAD THE VMA
; 8087          START READ,             ;START THE MEMORY CRANKING
; 8088          VMA PHYSICAL            ;ABSOLUTE ADDRESS
; 8089          MEM READ,               ;WAIT FOR MEMORY
; 8090          [AR]_MEM                ;POINT POINTER IN AR
; 8091  ;LOOK AT SECTION POINTER AND DISPATCH ON TYPE
; 8092  =000
; 8093  PF45:   SC_7,                   ;FETCH SECTION 0 POINTER
; 8094          CALL [SETPTR]           ;FIGURE OUT POINTER TYPE
; 8095  SECIMM: TL [AR],                ;IMMEDIATE POINTER
; 8096          #/77,                   ;TEST FOR 12-17 = 0
; 8097          J/PF50                  ;CONTINUE AT PF50
; 8098          [AR]_[AR]+WORK[SBR],    ;SHARED SECTION
; 8099          J/SECSHR                ;GO FETCH POINTER FROM SPT
; 8100          [AR]_[AR]+WORK[SBR],    ;INDIRECT SECTION POINTER
; 8101          CALL [RDPT]             ;GO FETCH SPT ENTRY
; 8102  =111    TL [AR],                ;12 TO 17 = 0?
; 8103          #/77                    ; ..
; 8104  =
; 8105  =0      PAGE FAIL TRAP          ;NO
; 8106          [AR]_[AR]*2,            ;FIRST SHIFT
; 8107          STEP SC                 ;SC WAS LOADED AT PF45
; 8108  =0*0
; 8109  PF60:   [AR]_[AR]*2,            ;CONVERT TO ADDRESS OF
; 8110          STEP SC,                ; SECTION TABLE
; 8111          J/PF60
; 8112          CALL [RDPT]             ;READ SECTION TABLE
; 8113  =1*1    J/PF45                  ;TRY AGAIN
; 8114  =
; 8115
```

```
U 2552, 3773,4443,0000,4174,4007,0700,0010,0000,0000
U 2556, 2554,4553,0300,4374,4007,0321,0000,0000,0077



U 2554, 2670,4553,1300,4374,4007,0321,0000,0001,0000
U 2555, 2560,4251,0304,4374,4007,0700,0000,0000,3777
U 2560, 2620,3441,0403,4174,4007,0700,0010,0000,0000
U 2562, 2553,3551,0303,7274,4007,0701,0000,0000,0220


U 2553, 3076,4443,0000,4174,4007,0700,0210,0001,0002
U 2557, 2564,4443,0000,4174,4007,0700,2000,0071,0007



U 2564, 2564,3445,0404,4174,4007,0630,2000,0060,0000
```

```
; 8116  ;HERE FOR SHARED SECTION. AR GETS THE ADDRESS OF PAGE TABLE
; 8117  =0**
; 8118  SECSHR: CALL [RDPT]              ;READ WORD FROM SPT
; 8119          TL [AR], #/77            ;TEST FOR BITS 12-17 = 0
; 8120
; 8121  ;HERE WITH ADDRESS OF PAGE TABLE IN AR AND SKIP ON
; 8122  ; BITS 12 THRU 17 EQUAL TO ZERO
; 8123  =0
; 8124  PF50:   PAGE FAIL TRAP           ;BITS 12-17 .NE. 0
; 8125          [ARX]_[AR].AND.# CLR LH, ;PAGE NUMBER OF PAGE TABLE
; 8126          #/3777                   ;11 BIT PHYSICAL PAGE #
; 8127  =0*     [AR]_[ARX],              ;COPY ADDRESS
; 8128          CALL [UPCST]             ;UPDATE CSTO
; 8129  PF70:   [AR]_[AR].OR.WORK[PUR]   ;PUT IN NEW AGE AND
; 8130                                   ; USE BITS
; 8131  =0**    START NO TEST WRITE,     ;START MEMORY WRITE
; 8132          CALL [IBPX]              ;GO STORE IN MEMORY
; 8133          SC_7                     ;THIS CAN BE BUMMED
; 8134  =0
; 8135  PF75:   [ARX]_[ARX]*2,           ;CONVERT PAGE NUMBER TO
; 8136          STEP SC,                 ; PAGE ADDRESS
; 8137          J/PF75                   ;LOOP OVER 9 STEPS
; 8138
```

```
U 2565, 1040,3333,0006,4174,4007,0520,0000,0000,0000

U 1040, 2640,3771,0003,7274,4007,0622,0010,0000,0423

U 1041, 2640,3771,0003,7274,4007,0622,0010,0000,0424

U 1042, 1047,3333,0004,7174,4007,0700,0400,0000,0423

U 1043, 1047,3333,0004,7174,4007,0700,0400,0000,0424

U 1047, 1130,0113,0405,4174,4007,0700,0200,0024,1016

U 1130, 3770,3771,0003,4365,5007,0700,0210,0000,0002

U 1131, 1134,4553,0300,4374,4007,0321,0000,0000,0077

U 1132, 2572,0551,0303,7274,4007,0701,0000,0000,0215
```

```
; 8139  ;WE NOW HAVE THE ADDRESS OF THE PAGE TABLE ENTRY. GO
; 8140  ; READ IT AND START ANALYSIS
; 8141
; 8142  ;IF WE ARE HERE FOR THE FIRST TIME FOR THE USER OR EXEC SAVE THE
; 8143  ; ADDRESS OF THE PAGE TABLE IN PTA.E OR PTA.U SO THAT WE DO NOT
; 8144  ; HAVE TO DO THE SECTION LOOKUP EVERY TIME.
; 8145          READ [BRX], SKIP DPO      ;USER OR EXEC REF?
; 8146  =000     [AR]_WORK[PTA.E],         ;EXEC MODE
; 8147          SKIP AD.EQ.0, 4T,         ;SEE IF SET YET
; 8148          CALL [SHDREM]             ;SHOULD WE REMEMBER PTR
; 8149          [AR]_WORK[PTA.U],         ;USER MODE
; 8150          SKIP AD.EQ.0, 4T,         ;SEE IF SET YET
; 8151          CALL [SHDREM]             ;SHOULD WE REMEMBER PTR
; 8152          WORK[PTA.E]_[ARX],        ;SAVE FOR EXEC
; 8153          J/PF76                    ;CONTINUE BELOW
; 8154          WORK[PTA.U]_[ARX],        ;SAVE FOR USER
; 8155          J/PF76                    ;CONTINUE BELOW
; 8156  =111
; 8157  PF76:   VMA_[ARX]+[BR],           ;READ PAGE POINTER
; 8158          START READ,
; 8159          VMA PHYSICAL
; 8160  =
; 8161  =00
; 8162  PF77:   MEM READ,                 ;START ANALYSIS OF POINTER
; 8163          [AR]_MEM,
; 8164          CALL [SETPTR]
; 8165  PTRIMM: TL [AR],                  ;IMMEDIATE POINTER
; 8166          #/77,                     ;CHECK FOR BITS 0-5
; 8167          J/PF80                    ;GO TO PF80
; 8168          [AR]_[AR]+WORK[SBR],      ;SHARED POINTER
; 8169          J/PTRSHR                  ;GO TO READ SPT
; 8170
```

```
U 1133, 2566,3770,0305,4344,4007,0670,0000,0000,0000
U 2566, 2561,3333,0003,7174,4007,0700,0400,0000,0426
U 2567, 2570,4251,0505,4374,4007,0370,0000,0000,0777
U 2570, 1000,4551,0303,4374,0007,0700,0000,0027,7000
U 2571, 2670,4553,1300,4374,4007,0321,0000,0001,0000
U 2572, 3773,4443,0000,4174,4007,0700,0010,0000,0000
U 2576, 1134,4553,0300,4374,4007,0321,0000,0000,0077
U 1134, 2670,4553,1300,4374,4007,0321,0000,0001,0000
U 1135, 2620,4251,0304,4374,4007,0700,0010,0000,3777
U 1137, 3745,3551,0606,4374,0007,0700,0000,0010,0000
U 3745, 2574,4553,1300,4374,4007,0321,0000,0002,0000
U 2574, 3746,3551,0606,4374,4007,0700,0000,0002,0000
U 2575, 2600,4553,0600,4374,4007,0321,0000,0003,0000
U 2600, 2670,4553,1300,4374,4007,0321,0000,0001,0000
U 2601, 2573,3551,0303,7274,4007,0701,0000,0000,0220
U 2561, 3562,3771,0003,7274,4117,0701,0010,0000,0301
U 2563, 1133,3771,0003,7274,4007,0701,0000,0000,0426
```

```
; 8171  ;INDIRECT POINTER. CHANGE PAGE # AND LOOK FOR PAGE TABLE
; 8172  PTRIND: [BR]_[AR] SWAP,          ;PUT IN RIGHT HALF
; 8173          SKIP/-1 MS              ;DID CLOCK GO OFF
; 8174  =0      WORK[SV.AR1]_[AR],      ;YES--UPDATE CLOCK
; 8175          J/PFTICK                ; ..
; 8176          [BR]_[BR].AND.# CLR LH, ;UPDATE PAGE # AND RESTART
; 8177          #/777,                  ;MASK FOR PAGE #
; 8178          SKIP IRPT               ;SEE IF THIS IS A LOOP
; 8179  =0      [AR]_[AR].AND.#,        ;CHANGE INDIRECT POINTER
; 8180          #/277000,               ; INTO SHARE POINTER
; 8181          HOLD RIGHT,             ; ..
; 8182          J/PF45                  ;GO BACK AND TRY AGAIN
; 8183          PAGE FAIL TRAP          ;POINTER LOOP
; 8184
; 8185  =0**
; 8186  PTRSHR: CALL [RDPT]             ;GO LOOK AT POINTER
; 8187          TL [AR],                ;BITS 12-17 .EQ. 0?
; 8188          #/77
; 8189
; 8190  ;HERE WITH FINAL POINTER. SKIP IF 12-17 NOT EQUAL TO ZERO
; 8191  =00
; 8192  PF80:   PAGE FAIL TRAP          ;NO--TAKE A TRAP
; 8193          [ARX]_[AR].AND.# CLR LH, ;SAVE PHYSICAL PAGE #
; 8194          #/3777,                 ;MASK TO 13 BITS
; 8195          CALL [UPCST]            ;UPDATE CSTO
; 8196  ;HERE WE HAVE CST ENTRY IN AR, PAGE FAIL WORD IN BRX. GO LOOK
; 8197  ; AT WRITABLE AND WRITTEN BITS
; 8198  =11
; 8199  PF90:   [BRX]_[BRX].OR.#,       ;TRANSLATION IS VALID
; 8200          #/100000, HOLD RIGHT    ; ..
; 8201          TL [FLG], FLG.W/1       ;IS THIS PAGE WRITABLE?
; 8202  =0      [BRX]_[BRX].OR.#,       ;YES--INDICATE THAT IN PFW
; 8203          #/020000,
; 8204          J/PF100                 ;NOT WRITE VIOLATION
; 8205          TL [BRX],               ;IS THIS A WRITE REF?
; 8206          WRITE TEST/1, WRITE CYCLE/1
; 8207  =0      PAGE FAIL TRAP          ;WRITE VIOLATION
; 8208  PF107:  [AR]_[AR].OR.WORK[PUR], ;PUT IN NEW AGE
; 8209          J/PF110                 ;GO TO STORE CST ENTRY
; 8210
; 8211  =0*
; 8212  PFTICK: [AR]_WORK[TIME1],       ;UPDATE TIMER
; 8213          SPEC/CLRCLK, CALL [TOCK]
; 8214          [AR]_WORK[SV.AR1],      ;RESTORE AR
; 8215          J/PTRIND                ;GO TRY AGAIN
; 8216
```

```
U 3746, 2602,4553,0600,4374,4007,0321,0000,0001,0000

U 2602, 2604,3551,0303,4370,4007,0700,0000,0000,0001

U 2603, 2604,4553,0300,4374,4007,0331,0000,0000,0001

U 2604, 2601,3551,0606,4374,0007,0700,0000,0004,0000

U 2605, 2601,5551,1313,4374,0007,0700,0000,0002,0000

U 2573, 3076,4443,0000,4174,4007,0700,0210,0001,0002

U 2577, 2606,4553,1300,4374,4007,0331,0000,0040,0000

U 2606, 2610,3441,0403,4174,4007,0700,2000,0071,0007

U 2607, 3755,3771,0003,7274,4007,0701,0000,0000,0210

U 2610, 2610,3445,0303,4174,4007,0630,2000,0060,0000

U 2611, 3747,4551,0303,4374,0007,0700,0000,0000,0003
U 3747, 3750,4221,0013,4170,4007,0700,0000,0000,0000

U 3750, 3752,3551,0606,4374,0007,0700,0000,0010,0000
U 3752, 2612,4553,1300,4374,4007,0321,0000,0000,4000

U 2612, 2613,3551,0606,4374,0007,0700,0000,0000,2000

U 2613, 3753,4551,0606,4370,4007,0700,0000,0000,0777

U 3753, 1500,3111,0603,4174,4003,7700,0200,0003,0001
```

```
; 8217  ;HERE IF PAGE IS WRITABLE
; 8218  PF100:  TL [BRX], WRITE CYCLE/1 ;IS THIS A WRITE REF?
; 8219  =0      [AR]_[AR].OR.#,         ;YES--SET WRITTEN BIT
; 8220          #/1,
; 8221          HOLD LEFT,
; 8222          J/PF105
; 8223          TR [AR],                ;NOT WRITE, ALREADY WRITTEN?
; 8224          #/1
; 8225  =0
; 8226  PF105:  [BRX]_[BRX].OR.#,       ;WRITTEN SET BIT
; 8227          #/040000,               ;MARK PAGE AS
; 8228          HOLD RIGHT,             ;WRITABLE
; 8229          J/PF107                 ;STORE CST WORD
; 8230          [FLG]_[FLG].AND.NOT.#,  ;NOT WRITTEN, CAUSE TRAP ON
; 8231          FLG.W/1,                ; WRITE ATTEMPT
; 8232          HOLD RIGHT,             ;ONLY CLEAR LH
; 8233          J/PF107
; 8234  =0**
; 8235  PF110:  START NO TEST WRITE,
; 8236          CALL [IBPX]             ;STORE CST ENTRY
; 8237
; 8238  ;HERE WHEN WE HAVE FIGURED OUT PHYSICAL ADDRESS (IN ARX) AND FLAGS
; 8239  ; (IN BRX) RELOAD PAGE TABLE.
; 8240  PFDONE: TR [FLG],               ;MAP INSTRUCTION?
; 8241          #/400000
; 8242  =0      [AR]_[ARX],             ;GET PHYSCIAL PAGE #
; 8243          SC_7,                   ;PREPARE TO CONVERT TO
; 8244          J/PF130                 ; WORD ADDRESS
; 8245          [AR]_WORK[SV.VMA],;RESTORE VMA
; 8246          J/PF120
; 8247  =0
; 8248  PF130:  [AR]_[AR]*2,            ;CONVERT TO WORD #
; 8249          STEP SC,
; 8250          J/PF130
; 8251          [AR]_[AR].AND.#,        ;JUST ADDRESS BITS
; 8252          #/3,
; 8253          HOLD RIGHT
; 8254          END MAP                 ;CLEAR MAP FLAGS
; 8255          [BRX]_[BRX].OR.#,       ;TURN ON THE TRANSLATION
; 8256          #/100000,               ; VALID BIT
; 8257          HOLD RIGHT              ; IN LEFT HALF ONLY
; 8258          TL [FLG], FLG.C/1       ;CACHE BIT SET?
; 8259  =0      [BRX]_[BRX].OR.#,       ;YES--SET IN MAP WORD
; 8260          #/002000, HOLD RIGHT    ; ..
; 8261          [BRX]_[BRX].AND.#,      ;PRESERVE WORD #
; 8262          #/777, HOLD LEFT        ; IN PAGE FAIL WORD
; 8263          [AR]_[AR].OR.[BRX],     ;COMPLETE MAP INSTRUCTION
; 8264          EXIT
; 8265
```

```
U 3755, 3756,3441,0305,4174,4007,0700,0000,0000,0000    ; 8266  PF120:  [BR]_[AR]                ;COPY PAGE FAIL WORD
                                                        ; 8267          [BR]_[AR].AND.NOT.#,    ;CLEAR BITS WHICH START A CYCLE
                                                        ; 8268          READ CYCLE/1,           ; ..
                                                        ; 8269          WRITE CYCLE/1,          ; ..
                                                        ; 8270          WRITE TEST/1,           ; ..
U 3756, 3757,5551,0305,4374,0007,0700,0000,0007,0000    ; 8271          HOLD RIGHT              ;JUST DO LEFT HALF
                                                        ; 8272          VMA_[BR], 3T,           ;RESTORE VMA
U 3757, 3760,3443,0500,4174,4007,0701,0200,0000,0030    ; 8273          DP FUNC/1               ;SET USER ACCORDING TO WHAT IT WAS
                                                        ; 8274          [ARX]_[ARX].AND.# CLR LH, ;JUST KEEP PAGE #
U 3760, 3761,4251,0404,4374,4007,0700,0000,0000,3777    ; 8275          #/3777                  ; ..
U 3761, 3762,3551,0406,4374,4007,0700,0000,0040,0000    ; 8276          [BRX]_[ARX].OR.#, #/400000 ;SET VALID BITS
U 3762, 2614,4553,1300,4374,4007,0321,0000,0002,0000    ; 8277          TL [FLG], FLG.W/1       ;WANT WRITE SET?
U 2614, 2615,3551,0606,4374,4007,0700,0000,0004,0000    ; 8278  =0      [BRX]_[BRX].OR.#, #/040000 ;SET WRITE BIT
                                                        ; 8279          TL [FLG], FLG.C/1,      ;WANT CACHE SET?
U 2615, 2616,4553,1300,4374,4147,0321,0000,0000,4000    ; 8280          LOAD PAGE TABLE         ;LOAD PAGE TABLE ON NEXT
                                                        ; 8281                                  ; MICRO INSTRUCTION
                                                        ; 8282  =0      [BRX]_[BRX].OR.#,       ;SET CACHE BIT
U 2616, 3763,3551,0606,4374,4007,0700,0000,0002,0000    ; 8283          #/020000, J/PF125       ;CACHE BIT
U 2617, 3763,3333,0006,4174,4007,0700,0000,0000,0000    ; 8284          READ [BRX]              ;LOAD PAGE TABLE
U 3763, 3764,3771,0004,7274,4007,0701,0000,0000,0212    ; 8285  PF125:  [ARX]_WORK[SV.ARX]
U 3764, 3765,3771,0005,7274,4007,0701,0000,0000,0213    ; 8286          [BR]_WORK[SV.BR]
U 3765, 3766,3771,0006,7274,4007,0701,0000,0000,0214    ; 8287          [BRX]_WORK[SV.BRX]
                                                        ; 8288          VMA_[AR],               ;MAKE MEM REQUEST
                                                        ; 8289          DP FUNC/1, 3T,          ;FROM DATA PATH
U 3766, 3767,3443,0300,4174,4007,0701,0200,0000,0032    ; 8290          WAIT/1                  ;WAIT FOR PREVIOUS CYCLE TO
                                                        ; 8291                                  ; COMPLETE. (NEED THIS TO
                                                        ; 8292                                  ; START ANOTHER CYCLE)
                                                        ; 8293          [AR]_WORK[SV.AR],
U 3767, 0000,3771,0003,7274,4004,1701,0000,0000,0211    ; 8294          RETURN [0]
                                                        ; 8295  ;SUBROUTINE TO START CST UPDATE
                                                        ; 8296  ;CALL WITH:
                                                        ; 8297  ;       AR/ PHYSICAL PAGE NUMBER
                                                        ; 8298  ;RETURN 2 WITH ENTRY IN AR, PAGE FAIL IF AGE TOO SMALL
                                                        ; 8299  =0**
                                                        ; 8300  UPCST:  [AR]_[AR]+WORK[CBR],    ;ADDRESS OF CSTO ENTRY
U 2620, 3773,0551,0303,7274,4007,0701,0010,0000,0216    ; 8301          CALL [RDPT]             ;READ OLD VALUE
                                                        ; 8302          TL [AR],                ;0 - 5 = 0?
U 2624, 2622,4553,0300,4374,4007,0321,0000,0077,0000    ; 8303          #/770000                ; ..
                                                        ; 8304  =0      [AR]_[AR].AND.WORK[CSTM],       ;CLEAR AGE FIELD
U 2622, 0002,4551,0303,7274,4004,1701,0000,0000,0217    ; 8305          RETURN [2]              ;AGE IS NOT ZERO
U 2623, 2670,4553,1300,4374,4007,0321,0000,0001,0000    ; 8306          PAGE FAIL TRAP          ;AGE TOO SMALL
                                                        ; 8307
```

```
U 3770, 3771,3551,0304,4374,4007,0700,0000,0075,3777
U 3771, 3772,4111,0413,4174,0007,0700,0000,0000,0000
U 3772, 2626,3333,0003,4174,4007,0520,0000,0000,0000

U 2626, 2630,4553,0300,4374,4007,0321,0000,0030,0000
U 2627, 2670,4553,1300,4374,4007,0321,0000,0001,0000

U 2630, 2632,4553,0300,4374,4007,0321,0000,0010,0000
U 2631, 2670,4553,1300,4374,4007,0321,0000,0001,0000

U 2632, 2634,4553,0300,4374,4007,0321,0000,0020,0000
U 2633, 0002,4443,0000,4174,4004,1700,0000,0000,0000

U 2634, 0003,4443,0000,4174,4004,1700,0000,0000,0000
U 2635, 0001,4443,0000,4174,4004,1700,0000,0000,0000
```

```
; 8308  ;SUBROUTINE TO LOOK AT PAGE POINTER
; 8309  ;CALL WITH POINTER IN AR
; 8310  ;RETURNS 1 IF TYPE 1
; 8311  ;RETURNS 2 IF TYPE 2
; 8312  ;RETURNS 3 IF TYPE 3
; 8313  ;GOES TO PFT IF TYPE 0 OR 4 THRU 7
; 8314  SETPTR: [ARX]_[AR].OR.#,          ;AND C AND W BITS
; 8315          #/753777                 ; OF ALL POINTERS
; 8316          [FLG]_[FLG].AND.[ARX],   ; ..
; 8317          HOLD RIGHT               ;KEEP IN LH OF FLG
; 8318          READ [AR],               ;TYPE 4,5,6 OR 7?
; 8319          SKIP DPO                 ; ..
; 8320  =0      TL [AR],                 ;HERE WE TEST FOR TYPE
; 8321          #/300000,                ; ZERO POINTER
; 8322          J/STPTR1                 ;CHECK AT STPTR1
; 8323          PAGE FAIL TRAP           ;BAD TYPE
; 8324  =0
; 8325  STPTR1: TL [AR],                 ;NOT ZERO
; 8326          #/100000,                ;SEPERATE TYPE 2
; 8327          J/STPTR2                 ; ..
; 8328          PAGE FAIL TRAP           ;TYPE 0
; 8329
; 8330  =0
; 8331  STPTR2: TL [AR],                 ;SEPERATE TYPE 1
; 8332          #/200000,                ; AND 3
; 8333          J/STPTR3                 ; ..
; 8334          RETURN [2]               ;TYPE 2
; 8335
; 8336  =0
; 8337  STPTR3: RETURN [3]               ;TYPE 3
; 8338          RETURN [1]               ;TYPE 1
; 8339
```

```
U 3773, 2636,3443,0300,4174,4007,0370,0200,0024,1016

U 2636, 0004,3771,0003,4365,5004,1700,0200,0000,0002
U 2637, 2670,4553,1300,4374,4007,0321,0000,0001,0000

U 2640, 0007,4443,0000,4174,4004,1700,0000,0000,0000
U 2641, 3774,7441,1303,4174,4007,0700,0000,0000,0000
U 3774, 2642,4553,0300,4374,4007,0321,0000,0002,4000
U 2642, 0007,4443,0000,4174,4004,1700,0000,0000,0000
U 2643, 0002,4443,0000,4174,4004,1700,0000,0000,0000
```

```
; 8340  ;SUBROUTINE TO FETCH A PAGE POINTER OR CST ENTRY
; 8341  ;CALL WITH ADDRESS IN AR
; 8342  ;RETURN 4 WITH WORD IN AR
; 8343  ;
; 8344  RDPT:   VMA_[AR],              ;LOAD THE VMA
; 8345          START READ,            ;START MEM CYCLE
; 8346          VMA PHYSICAL,          ;ABSOLUTE ADDRESS
; 8347          SKIP IRPT              ;CHECK FOR INTERRUPTS
; 8348  =O      MEM READ,              ;NO INTERRUPTS
; 8349          [AR]_MEM,              ;PUT THE DATA INTO AR
; 8350          RETURN [4]             ;AND RETURN
; 8351          PAGE FAIL TRAP         ;INTERRUPT
; 8352
; 8353
; 8354  ;SUBROUTINE TO SEE IF WE SHOULD REMEMBER AN EXEC SECTION PTR
; 8355  ;CALL WITH SKIP ON ADR.EQ.O
; 8356  ;RETURNS 2 IF WE SHOULD STORE AND 7 IF WE SHOULD NOT
; 8357  ;
; 8358  =O
; 8359  SHDREM: RETURN [7]             ;INDIRECT PTR
; 8360          [AR]_.NOT.[FLG]        ;FLIP BITS
; 8361          TL [AR], FLG.W/1, FLG.C/1 ;BOTH BITS SET
; 8362  =O      RETURN [7]             ;NO--DON'T STORE
; 8363          RETURN [2]             ;STORE
; 8364
```

```
U 3775, 2644,3333,0006,4174,4007,0520,0000,0000,0000
U 2644, 2646,1553,0500,4374,4007,0532,4000,0000,0340
U 2645, 3776,0111,1104,4174,4007,0700,0200,0024,1016
U 2646, 2650,1553,0500,4374,4007,0532,4000,0000,0400
U 2647, 2650,0551,0404,4374,4007,0701,0000,0000,0600
U 2650, 3776,0111,1004,4174,4007,0700,0200,0024,1016
U 2651, 2645,0551,0404,4374,4007,0701,0000,0000,0220
U 3776, 1435,3771,0004,4365,5007,0700,0200,0000,0002
U 1435, 2652,4553,0500,4374,4007,0331,0000,0000,0001
```

```
; 8365  ;HERE IN KI10 MODE
; 8366  ;BR CONTAINS PAGE # AND ARX CONTAINS PAGE #/2
; 8367
; 8368  KIFILL: READ [BRX],              ;USER REF?
; 8369         SKIP DPO                  ; ..
; 8370  =0      [BR]-#,                   ;EXEC--LESS THAN 340?
; 8371         #/340,                    ; ..
; 8372         SKIP DP18, 4T,            ; ..
; 8373         J/KIF10                   ;FOLLOW EXEC PATH
; 8374  KIUPT:  [ARX]_[ARX]+[UBR],        ;POINTER TO PAGE MAP ENTRY
; 8375         LOAD VMA,                 ;PUT ADDRESS IN VMA
; 8376         VMA PHYSICAL,             ;ABSOLUTE ADDRESS
; 8377         START READ,               ;FETCH UPT WORD
; 8378         J/KIF30                   ;JOIN COMMON CODE
; 8379  =0
; 8380  KIF10:  [BR]-#,                   ;EXEC ADDRESS .GE. 340
; 8381         #/400,                    ; SEE IF .GT. 400
; 8382         SKIP DP18, 4T,            ; ..
; 8383         J/KIEPT                   ;LOOK AT KIF20
; 8384         [ARX]_[ARX]+#, 3T,        ;EXEC ADDRESS .LT. 340
; 8385         #/600,                    ;IN EBR+600
; 8386         J/KIEPT                   ;JOIN COMMON CODE
; 8387
; 8388  =0
; 8389  KIEPT:  [ARX]_[ARX]+[EBR],        ;ADD OFFSET TO
; 8390         LOAD VMA,                 ; EPT
; 8391         START READ,               ;START FETCH
; 8392         VMA PHYSICAL,             ;ABSOLUTE ADDRESS
; 8393         J/KIF30                   ;GO GET POINTER
; 8394         [ARX]_[ARX]+#,            ;PER PROCESS PAGE
; 8395         #/220, 3T,                ; IS IN UPT + 400
; 8396         J/KIUPT                   ;JOIN COMMON CODE
; 8397  KIF30:  MEM READ,                 ;WAIT FOR DATA
; 8398         [ARX]_MEM                 ;PLACE IT IN ARX
; 8399         TR [BR],                  ;SEE IF EVEN OR ODD
; 8400         #/1                       ; ..
; 8401
```

```
U 2652, 2654,3333,0004,4174,4007,0530,0000,0000,0000
U 2653, 2652,3770,0404,4344,4007,0700,0000,0000,0000

U 2654, 2670,4553,1300,4374,4007,0321,0000,0001,0000

U 2655, 1500,5551,1313,4374,4007,0700,0000,0002,4000
U 1500, 2656,4553,0400,4374,4007,0331,0000,0002,0000

U 2656, 2657,3551,1313,4374,0007,0700,0000,0000,4000
U 2657, 2660,4553,0400,4374,4007,0331,0000,0004,0000

U 2660, 2661,3551,0606,4374,0007,0700,0000,0002,0000

U 2661, 2662,4553,0400,4374,4007,0331,0000,0010,0000

U 2662, 1501,3551,1313,4374,0007,0700,0000,0002,0000

U 2663, 2664,4553,0600,4374,4007,0321,0000,0001,0000

U 2664, 2654,3551,0606,4374,0007,0700,0000,0010,0000
U 2665, 2577,4443,0000,4174,4007,0700,0000,0000,0000

U 1501, 2577,3551,0606,4374,4007,0700,0000,0004,0000
```

```
; 8402  =0
; 8403  KIF40:  READ [ARX],              ;ODD
; 8404          SKIP DP18,               ;SEE IF VALID
; 8405          J/KIF50                  ;JOIN COMMON CODE
; 8406          [ARX]_[ARX] SWAP,        ;EVEN--FLIP AROUND
; 8407          J/KIF40                  ; AND CONTINUE
; 8408
; 8409  =0
; 8410  KIF50:  PAGE FAIL TRAP
; 8411  ;AT THIS POINT WE HAVE THE PAGE MAP ENTRY IN RH OF AR
; 8412          [FLG]_[FLG].AND.NOT.#,   ;CLEAR W AND C
; 8413          FLG.W/1, FLG.C/1         ; FLAGS
; 8414          TR [ARX], #/020000       ;CACHE ENABLED?
; 8415  =0      [FLG]_[FLG].OR.#,        ;SET CACHE BITS
; 8416          FLG.C/1, HOLD RIGHT      ; ..
; 8417          TR [ARX], #/040000       ;DO NOT CACHE
; 8418                                   ;SEE IF CACHE BIT SET
; 8419  =0      [BRX]_[BRX].OR.#,        ;COPY BITS TO BRX
; 8420          #/020000,
; 8421          HOLD RIGHT
; 8422          TR [ARX],                ; ..
; 8423          #/100000
; 8424  =0      [FLG]_[FLG].OR.#,        ;SAVE W
; 8425          FLG.W/1,                 ; ..
; 8426          HOLD RIGHT,              ; ..
; 8427          J/KIF90                  ;ALL DONE
; 8428          TL [BRX],                ;W=0, WRITE REF?
; 8429          WRITE CYCLE/1
; 8430  =0
; 8431  KIF80:  [BRX]_[BRX].OR.#,        ;WRITE FAILURE
; 8432          #/100000, HOLD RIGHT,    ;INDICATE THAT ACCESS WAS ON
; 8433          J/KIF50                  ;GO PAGE FAIL
; 8434          J/PFDONE                 ;ALL DONE
; 8435
; 8436  KIF90:  [BRX]_[BRX].OR.#,        ;PAGE IS WRITABLE
; 8437          #/40000,                 ;TURN ON IN BRX
; 8438          J/PFDONE                 ;ALL SET
; 8439
```

```
U 1502, 2666,3333,0005,7174,4007,0700,0400,0000,0213
U 2666, 1511,3771,0005,4354,4007,0700,0010,0000,0000
U 2667, 1503,4551,0505,4374,0007,0700,0000,0040,1237
U 1503, 2654,3111,0506,4174,4007,0700,0000,0000,0000
U 1507, 1140,4443,0000,4174,4007,0370,0000,0000,0000
U 1140, 3562,3771,0003,7274,4117,0701,0010,0000,0301
U 1141, 2671,4443,0000,4174,4007,0700,0000,0000,0000
U 1142, 1510,4223,0000,4364,4277,0700,0200,0000,0010
U 1510, 3763,3771,0003,7274,4007,0701,0000,0000,0210
U 1511, 0001,4223,0000,4364,4274,1700,0200,0000,0010
```

```
; 8440  ;HERE ON HARD PAGE FAILURES
; 8441  HARD:   WORK[SV.BR]_[BR]          ;SAVE BR (CLEANUP MAY NEED IT)
; 8442  =0      [BR]_VMA,                 ;BUILD PAGE FAIL WORD
; 8443          CALL [ABORT]              ;CLEAR ERROR
; 8444          [BR]_[BR].AND.#,          ;SAVE THE FLAGS
; 8445          #/401237,                 ; ..
; 8446          HOLD RIGHT                ; ..
; 8447          [BRX]_[BRX].OR.[BR],      ;COMPLETE PAGE FAIL WORD
; 8448          J/KIF50                   ;GO TRAP
; 8449
; 8450  PFPI1:  SKIP IRPT                 ;TIMER TRAP?
; 8451  =00
; 8452          [AR]_WORK[TIME1],         ;YES--GET LOW WORD
; 8453          SPEC/CLRCLK,              ;CLEAR CLOCK FLAG
; 8454          CALL [TOCK]               ;DO THE UPDATE
; 8455          J/PFT1                    ;EXTERNAL INTERRUPT
; 8456          ABORT MEM CYCLE           ;CLEAR 1MS FLAGS
; 8457  =
; 8458  PFPI2:  [AR]_WORK[SV.VMA],        ;RESTORE VMA
; 8459          J/PF125
; 8460
; 8461
; 8462  ABORT:  ABORT MEM CYCLE, RETURN [1]
; 8463
```

```
U 2670, 0104,4751,1217,4374,4007,0700,0000,0000,0100
U 2671, 2672,3771,0003,7274,4007,0611,0000,0000,0210
U 2672, 2674,4553,0300,4374,4007,0321,0000,0010,0000
U 2673, 1512,3771,0003,7274,4007,0701,0000,0000,0425
U 1512, 2676,3333,0003,4174,4467,0700,0000,0000,0004
U 2674, 1100,4443,0000,4174,4007,0700,0000,0000,0000
U 2675, 2676,1111,0701,4170,4007,0700,4000,0000,0000
U 2676, 1100,3333,0013,4174,4003,5701,0000,0000,0000
U 1100, 2700,4221,0013,4170,4007,0370,0000,0000,0000
U 1101, 3170,3771,0003,7274,4007,0701,0000,0000,0212
U 1102, 1563,0111,0701,4174,4007,0700,0000,0000,0000
U 1103, 3464,3771,0013,4370,4007,0700,0000,0000,0011
U 1104, 3464,3771,0013,4370,4007,0700,0000,0000,0012
U 1105, 2312,3771,0013,4370,4007,0700,0000,0000,0003
U 1106, 3473,3771,0013,4370,4007,0700,0000,0000,0012
U 1107, 3470,3771,0013,4370,4007,0700,0000,0000,0011
U 1110, 3462,3771,0013,4370,4007,0700,0000,0000,0011
U 1111, 2310,4221,0013,4170,4007,0700,0000,0000,0000
U 1112, 2312,4221,0013,4170,4007,0700,0000,0000,0000
U 1113, 2312,3771,0013,4370,4007,0700,0000,0000,0011
```

```
; 8464  ;HERE ON PAGE FAIL TRAP
; 8465  =O
; 8466  PFT:    HALT [IOPF]                ;IO PAGE FAILURE
; 8467  PFT1:   [AR]_WORK[SV.VMA],
; 8468          SKIP/TRAP CYCLE            ;SEE IF TRAP CYCLE
; 8469  =O      TL [AR], FETCH/1,          ;IS THIS AN INSTRUCTION FETCH
; 8470          J/PFT1A                    ;GO LOOK BELOW
; 8471          [AR]_WORK[TRAPPC]          ;RESTORE PC
; 8472          READ [AR], LOAD FLAGS, J/CLDISP
; 8473  =O
; 8474  PFT1A:  J/CLEANED                   ;YES--NO PC TO BACK UP
; 8475  FIXPC:  [PC]_[PC]-1, HOLD LEFT      ;DATA FAILURE--BACKUP PC
; 8476  =O
; 8477  CLDISP: CLEANUP DISP                ;GO CLEANUP AFTER PAGE FAIL
; 8478  =OOOO
; 8479  CLEANUP:
; 8480  CLEANED:                            ;(O) NORMAL CASE
; 8481          END STATE, SKIP IRPT,      ;NO MORE CLEANUP NEEDED
; 8482          J/PFT2                     ;HANDLE PAGE FAIL OR INTERRUPT
; 8483          [AR]_WORK[SV.ARX],         ;(1) BLT
; 8484          J/BLT-CLEANUP
; 8485          [PC]_[PC]+1,               ;(2) MAP
; 8486          J/MAPDON
; 8487          STATE_[EDIT-SRC],          ;(3) SRC IN STRING MOVE
; 8488          J/STRPF
; 8489          STATE_[EDIT-DST],          ;(4) DST IN STRING MOVE
; 8490          J/STRPF
; 8491          STATE_[SRC],               ;(5) SRC+DST IN STRING MOVE
; 8492          J/BACKD
; 8493          STATE_[EDIT-DST],          ;(6) FILL IN MOVSRJ
; 8494          J/STRPF4
; 8495          STATE_[EDIT-SRC],          ;(7) DEC TO BIN
; 8496          J/PFDBIN
; 8497          STATE_[EDIT-SRC],          ;(10) SRC+DST IN COMP
; 8498          J/CMSDST
; 8499          END STATE, J/BACKS         ;(11) EDIT SRC FAIL
; 8500          END STATE, J/BACKD         ;(12) EDIT DST FAIL
; 8501          STATE_[EDIT-SRC],          ;(13) SRC+DST IN EDIT
; 8502          J/BACKD
; 8503  =
; 8504
```

```
U 2700,  1513,0551,1103,4374,4007,0701,0000,0000,0500
U 2701,  0770,3551,1313,4374,0007,0700,0000,0001,0000

U 1513,  2702,3443,0300,4174,4007,0700,0200,0021,1016


U 2702,  1572,3333,0006,4175,5007,0701,0210,0000,0002

U 2703,  2704,4553,1000,4374,4007,0321,0000,0040,0000
U 2704,  2710,4521,1205,4074,4007,0700,0000,0000,0000
U 2705,  1562,3741,0105,4074,4007,0700,0000,0000,0000


U 1562,  1564,3333,0005,4175,5007,0701,0200,0000,0002


U 1563,  2706,4221,0013,4170,4007,0370,0000,0000,0000

U 2706,  1500,3441,0603,4174,4003,7700,0200,0003,0001
U 2707,  2701,1111,0701,4174,4007,0700,4000,0000,0000
```

```
; 8505  =0
; 8506  PFT2:   [AR]_[UBR]+#,          ;PREPARE TO STORE PFW
; 8507          #/500, 3T,
; 8508          J/PFT10
; 8509  PFT3:   TAKE INTERRUPT         ;PROCESS INTERRUPT
; 8510  PFT10:  VMA_[AR],              ;WHERE TO STORE PFW
; 8511          VMA PHYSICAL WRITE
; 8512  =0      MEM WRITE,             ;STORE PFW
; 8513          MEM_[BRX],
; 8514          CALL [NEXTAR]          ;ADVANCE POINTER TO
; 8515                                 ;PREPARE TO STORE PC
; 8516          TL [EBR], #/400000     ;KL PAGING?
; 8517  =0      [BR]_FLAGS,J/EAPF      ;YES--DO EXTENDED THING
; 8518          [BR]_PC WITH FLAGS     ;GET OLD PC
; 8519          MEM WRITE,             ;STORE OLD PC
; 8520          MEM_[BR],
; 8521          J/EAPF1
; 8522
; 8523  MAPDON: END STATE,             ;CLEAR MAP BIT
; 8524          SKIP IRPT              ;ANY INTERRUPT?
; 8525  =0      [AR]_[BRX],            ;RETURN PAGE FAIL WORD
; 8526          EXIT
; 8527          [PC]_[PC]-1, J/PFT3    ;INTERRUPTED OUT OF MAP
; 8528                                 ; RETRY INSTRUCTION
; 8529
```

```
                                                                ; 8530
                                                                ; 8531  =O
                                                                ; 8532  EAPF:   MEM WRITE, MEM_[BR],     ;STORE FLAGS
U 2710, 1572,3333,0005,4175,5007,0701,0210,0000,0002            ; 8533          CALL [NEXTAR]           ;STORE PC WORD
U 2711, 1564,3333,0001,4175,5007,0701,0200,0000,0002            ; 8534          MEM WRITE, MEM_[PC]     ; ..
                                                                ; 8535  EAPF1:  [AR]_[AR]+1,
                                                                ; 8536          VMA PHYSICAL READ,
U 1564, 2744,0111,0703,4174,4007,0700,0200,0024,1016            ; 8537          J/GOEXEC
                                                                ; 8538
U 1572, 0001,0111,0703,4170,4004,1700,0200,0023,1016            ; 8539  NEXTAR: NEXT [AR] PHYSICAL WRITE, RETURN [1]
                                                                ; 8540


; Number of microwords used:
;       D words= 512
;       U words= 2023, Highest= 2047

        END
```

(U) A

585 #

AR
589 # 2296 2334 2364 2365 2428 2476 2484 2623 2625 2628 2631
2634 2637 2721 2731 2746 2763 2773 2783 2793 2824 2834 2844
2879 2891 2898 2902 2912 2913 2915 2917 2918 2926 2949 2950
3176 3181 3183 3195 3198 3201 3206 3207 3277 3294 3296 3361
3364 3432 3433 3439 3459 3486 3501 3537 3548 3560 3614 3637
3642 3677 3721 3725 3732 3734 3887 3892 3921 3956 3970 3986
3997 4000 4003 4017 4044 4045 4050 4067 4083 4103 4215 4224
4229 4239 4245 4264 4268 4286 4314 4318 4320 4321 4340 4372
4375 4378 4392 4417 4434 4435 4440 4445 4453 4455 4457 4528
4538 4540 4545 4546 4591 4596 4606 4610 4648 4652 4656 4660
4836 4849 4934 4937 4949 4952 4958 4964 5002 5068 5071 5097
5103 5107 5108 5138 5144 5163 5169 5170 5171 5172 5181 5186
5206 5207 5219 5221 5230 5231 5251 5256 5257 5263 5298 5302
5304 5307 5310 5314 5316 5318 5320 5321 5322 5323 5329 5331
5332 5341 5346 5362 5388 5396 5401 5403 5447 5460 5479 5488
5492 5507 5521 5523 5525 5531 5558 5565 5568 5580 5583 5591
5593 5597 5602 5604 5620 5625 5628 5650 5653 5654 5658 5774
5778 5782 5789 5800 5809 5824 5844 5848 5861 5864 5890 5974
5981 5998 6020 6034 6061 6072 6073 6074 6087 6121 6151 6162
6200 6215 6243 6248 6277 6278 6279 6280 6282 6283 6286 6291
6340 6361 6368 6369 6395 6433 6445 6446 6453 6473 6476 6479
6483 6484 6486 6490 6492 6546 6547 6552 6553 6560 6563 6567
6636 6658 6664 6683 6701 6705 6708 6709 6710 6711 6733 6758
6759 6763 6769 6770 6774 6844 6847 6863 6865 6867 6869 6872
6874 6876 6912 6918 6926 6930 6933 6951 6957 6959 6964 6969
7048 7129 7131 7133 7135 7137 7139 7141 7143 7145 7167 7173
7235 7245 7251 7254 7255 7272 7280 7330 7341 7354 7356 7382
7387 7414 7417 7445 7463 7464 7523 7615 7623 7654 7980 7990
8086 8095 8098 8100 8102 8106 8109 8119 8125 8129 8165 8168
8172 8179 8187 8193 8208 8219 8223 8248 8251 8266 8267 8288
8300 8302 8304 8314 8320 8325 8331 8344 8361 8469 8510

ARX
590 # 2396 2678 2679 3617 3619 3632 3661 3983 3995 4004 4005
4022 4025 4027 4049 4069 4085 4105 4108 4128 4257 4285 4337
4363 4364 4443 4535 4536 4544 4613 4803 4810 4811 4839 4842
4863 4996 5011 5105 5143 5175 5361 5382 5385 5445 5462 5484
5504 5524 5529 5575 5614 5621 5862 5903 5910 5944 5978 6028
6037 6047 6056 6079 6080 6088 6120 6124 6201 6238 6246 6250
6252 6569 6572 6618 6686 6760 6864 6868 6870 6873 6875 6904
6906 7079 7092 7096 7286 7469 7471 7529 7751 8073 8081 8127
8135 8157 8242 8274 8276 8316 8384 8394 8406 8414 8417 8422

BR
591 # 2095 2292 2327 2329 2333 2814 2945 2978 2979 2982 2983
2984 2985 3000 3003 3006 3009 3013 3016 3018 3050 3051 3054
3055 3336 3339 3342 3345 3348 3351 3354 3357 3682 3687 3715
3740 3748 3749 3889 3894 3991 4123 4288 4290 4298 4302 4365
4368 4442 4446 4447 4448 4451 4456 4487 4629 4631 4690 4691
4692 4693 4694 4701 4710 4714 4723 4784 4785 4786 4787 4788
4809 4852 4858 4961 5004 5006 5009 5014 5104 5111 5112 5116
5117 5173 5176 5210 5261 5344 5368 5370 5380 5394 5473 5491
5494 5728 5740 5741 5743 5744 5771 5780 5825 5830 5855 5858
5949 6014 6017 6018 6031 6039 6045 6168 6178 6189 6198 6202
6220 6273 6288 6326 6337 6404 6660 6661 6702 6703 6743 6751
6762 6766 6843 6846 6861 6895 6897 6900 6942 6950 6963 6965
6977 6978 6985 6986 6992 7047 7058 7065 7134 7136 7138 7144
7146 7182 7195 7292 7315 7316 7317 7358 7360 7388 7392 7394

Cross Reference Listing

		7398	7400	7424	7430	7521	7988	8052	8055	8062	8176	8272	8370
		8380	8399	8444	8447								
	BRX	592 #	4019	4047	4053	4124	4147	4167	4171	4175	4179	4185	4189
		4193	4197	4274	4327	4367	4479	4482	4493	4940	4995	5017	5023
		5030	5054	5136	5137	5139	5178	5183	5373	5390	5456	5470	5471
		5513	5732	5831	5901	5912	5913	5937	5986	5994	6075	6104	6117
		6126	6131	6139	6141	6150	6156	6161	6173	6212	6214	6232	6241
		6268	6270	6272	6303	6306	6322	6334	6336	6416	6418	6421	6487
		6494	6588	6592	6596	6600	6604	6608	6624	6887	6994	6995	6996
		7094	7244	7299	7300	7363	7446	7984	8037	8038	8041	8047	8049
		8199	8202	8205	8218	8226	8255	8259	8261	8263	8278	8282	8419
		8428	8431	8436	8525								
	EBR	594 #	6790	6970	6971	6972	6975	7247	7290	8057	8084	8389	8516
	FLG	597 #	3528	5325	5426	5427	5434	5561	5564	5567	5570	5630	5637
		5638	6412	8034	8105	8124	8183	8192	8201	8207	8230	8240	8258
		8277	8279	8306	8323	8328	8351	8360	8410	8412	8415	8424	8509
	HR	588 #	2196	2202	2208	2214	2223	2227	2236	2241	2342	2690	2695
		3436	3445	3448	3450	3451	3452	3471	3475	3485	3490	3493	3513
		3534	3547	3826	3828	3830	3832	3834	3836	3838	3840	3846	3860
		3936	5730	6723	6800	6806	6815	6816	6817	6821	6822	6823	6824
		6829	6830	6831	6834	6835	6836	6837	6838	6839	6840	6841	7015
		7037	7132	7448	7462	7591	7593	7595	7597	7599	7601	7603	7605
	MAG	586 #	2331	4029	4032	4051	4054	4055	4068	4074	4076	4110	4112
		4114	4292	4310	4332	4336	4339	4380	4382	4383	4510	4515	4520
		4526	4527	5364	5455	5510	5536	5571	5606	5610	5612		
	MASK	596 #	2057	2058	2059	2062	2065	2098	2109	2145	2149	2152	2155
		2164	2167	2170	2173	2424	2647	2668	2875	2923	3487	3535	3549
		3570	3850	3861	3883	3886	3930	3990	4126	4160	4294	4305	4333
		4458	4670	4790	4801	5266	5326	5399	5449	5539	5608	5695	5697
		5699	5701	5703	5705	5707	5709	5711	5946	5948	5970	6085	6407
		7046	7089	7090	7091	7225	7226	7227	7228	7229	7230	7231	7237
		7256	7266	7284	7298	7376	7379	7415	7426	7432	7449	7505	7653
		8013	8015	8029	8466	8517							
	ONE	593 #	2087	2091	2136	2138	2183	2270	2321	2324	2854	3245	3311
		3327	3394	3408	3434	3443	3446	3521	3539	3542	3879	3925	3941
		4220	4635	4932	5003	5035	5037	5042	5059	5060	5253	5265	5340
		5489	5647	5766	5807	5840	5846	5867	5885	5895	5914	5915	5950
		5951	5993	6057	6062	6125	6137	6144	6164	6171	6176	6185	6195
		6199	6203	6204	6240	6254	6265	6317	6341	6435	6457	6510	6633
		6634	6640	6662	6704	6746	7053	7076	7106	7158	7442	7519	7639
		7725	7732	7735	7738	7741	7744	8475	8485	8527	8535	8539	
	PC	587 #	2135	2139	2157	2250	2312	2337	3233	3360	3365	3596	3713
		3719	3737	3915	4115	4117	4592	4601	6779	7149	7268	7420	7650
		8518											
	PI	598 #	3499	3500	3514	6825	6826	7126	7140	7142	7147	7157	7159
		7218	7219	7220	7221	7222	7223	7224	7233				
	TO	600 #	4095	4099	4335	4346	4353	4356	4358	4359	5381	5389	5415
		5417	5418	5419	5422	5430	5433	5435	5475	5541	5549	5550	6169
		6236	7160	7296									
	T1	601 #	4312	4452	5405	5490	5493	5512	5522				
	UBR	595 #	3852	3885	6782	6923	6945	6984	6991	8076	8374	8506	
	XWD1	599 #	3416	3586	7282								
(D) A		1222 #											
	DBLAC	1226 #	2682										
	DFP	1232 #	5351	5352	5440	5501							

Cross Reference Listing

```
        DREAD           1225 #  2673    2674    3978    3979    4062    4280
        DSHIFT          1228 #  2864    2865
        FP              1230 #  5077    5078    5079    5080    5082    5083    5085    5086    5087    5088    5090
                        5091    5122    5123    5124    5126    5128    5129    5151    5152    5153    5155    5157
                        5158    5236    5237
        FPI             1229 #  5081    5089    5127    5156
        IOT             1233 #  6811    6812    7001    7322    7323    7324    7325    7333    7334    7335    7336
                        7347    7348    7349    7350    7610    7611    7976
        RD-PF           1231 #  2454    2459    2464    2469    2507    2512    2517    2522    2527    2532    2537
                        2542    2549    2554    2559    2564    2569    2574    2579    2584    2714    2724    2734
                        2739    2756    2766    2776    2786    2806    2817    2827    2837    3949    3963    4011
                        4036    4203    4208
        READ            1223 #  2457    2462    2467    2472    2491    2509    2510    2514    2515    2520    2525
                        2530    2535    2540    2545    2551    2552    2556    2557    2562    2567    2572    2577
                        2582    2587    2716    2717    2726    2727    2736    2737    2741    2742    2749    2758
                        2759    2768    2769    2778    2779    2788    2789    2808    2809    2819    2820    2829
                        2830    2839    2840    3093    3094    3095    3096    3097    3098    3109    3110    3111
                        3112    3113    3114    3115    3116    3127    3128    3129    3130    3131    3132    3133
                        3134    3144    3145    3146    3147    3148    3149    3150    3151    3266    3267    3268
                        3269    3270    3271    3272    3273    3283    3284    3285    3286    3287    3288    3289
                        3290    3300    3301    3302    3303    3304    3305    3306    3307    3316    3317    3318
                        3319    3320    3321    3322    3323    3554    3951    3952    3965    3966    4013    4014
                        4038    4039    4205    4206    4210    4211    4581    4582    4583    4584    4585    5201
        SHIFT           1227 #  2859    2860    2861
        WRITE           1224 #  2456    2461    2466    2471    2519    2524    2529    2534    2539    2544    2561
                        2566    2571    2576    2581    2586    2683    2751    2752
(U) ACALU               1167 #
        AC+N            1169 #  2080    2081    2088    2278    2290    2396    2679    2688    2929    2950    2994
                        3024    3983    3995    4074    4076    4078    4086    4088    4095    4097    4109    4110
                        4111    4112    4113    4114    4226    4310    4332    4336    4339    4353    4354    4356
                        4359    4378    4380    4382    4383    4509    4510    4513    4514    4515    4518    4519
                        4520    4526    4527    5364    5455    5508    5510    5516    5575    5583    5621    5628
                        5770    5778    5788    5791    5821    5824    5837    5844    5851    5861    5899    5909
                        5929    5944    5961    5976    5978    6004    6007    6009    6016    6017    6018    6020
                        6026    6028    6031    6037    6039    6067    6069    6070    6074    6081    6082    6083
                        6087    6088    6095    6097    6102    6108    6124    6135    6147    6156    6162    6173
                        6176    6178    6182    6195    6201    6202    6325    6356    6361    6366    6403    6405
                        6509    6510    6622    6629    6636    6655    6657    6664    6684    6705    6741    6743
                        6749    6751    6755    6758    6763
        B               1168 #
(D) ACDISP              1253 #  3423    6811    6812    7001
(U) ACN                 1170 #  2278    2290    2396    2679    2688    2929    2950    2994    3024    3983    3995
                        4074    4076    4078    4086    4088    4095    4097    4109    4110    4111    4112    4113
                        4114    4226    4310    4332    4336    4339    4353    4354    4356    4359    4378    4380
                        4382    4383    4509    4510    4513    4514    4515    4518    4519    4520    4526    4527
                        5364    5455    5508    5510    5516    5575    5583    5621    5628    6108    6124    6135
                        6182    6201
        BINO            1177 #  2080    6004    6026    6031    6039    6070    6081    6082    6083    6087    6095
        BIN1            1178 #  2081    2088    5976    5978    6007    6009    6016    6017    6018    6020    6028
                        6037    6067    6069    6074    6088    6097
        DLEN            1174 #  5770    5778    5791    5821    5824    5851    5861    5899    5909    5944    6102
                        6147    6156    6162    6173    6176    6178    6195    6202    6509    6510    6622    6636
                        6655    6755    6758    6763
        DSTP            1175 #  5788    5961    6356    6361    6405    6629    6684    6705    6749    6751
        MARK            1176 #  6325    6403
```

Cross Reference Listing

	SRCLEN	1172 #											
	SRCP	1173 #	5837	5844	5929	6366	6657	6664	6741	6743			
(U)	AD	482 #	2062	2065	2098	2109	2145	2149	2152	2155	2164	2167	2170
		2173	2227	2424	2645	2647	2666	2668	3487	3570	3850	3883	3930
		4294	4305	5695	5697	5699	5701	5703	5705	5707	5709	5711	5741
		5970	6085	6407	7046	7089	7090	7091	7225	7226	7227	7228	7229
		7230	7231	7237	7266	7284	7298	7376	7379	7415	7426	7432	7449
		7505	7653	8013	8015	8029	8466						
	A	511 #	2057	2059	2135	2139	2157	2202	2214	2241	2250	2292	2333
		2334	2337	2396	2428	2679	2695	2879	2891	2898	2902	2912	2913
		2915	2917	2918	2926	2945	2950	2978	2979	2982	2983	2984	2985
		3000	3003	3006	3009	3013	3016	3050	3051	3054	3055	3207	3233
		3296	3336	3348	3360	3361	3364	3365	3432	3433	3439	3459	3475
		3486	3501	3537	3548	3560	3614	3617	3619	3637	3642	3715	3721
		3732	3737	3749	4017	4019	4022	4044	4045	4047	4049	4067	4069
		4083	4085	4103	4105	4117	4123	4124	4126	4147	4160	4215	4224
		4239	4245	4274	4286	4288	4290	4302	4314	4318	4333	4337	4340
		4353	4359	4363	4372	4392	4440	4445	4446	4447	4452	4453	4455
		4457	4528	4592	4601	4606	4613	4629	4652	4660	4714	4790	4803
		4839	4858	4863	4937	4940	4958	4996	5009	5017	5023	5071	5103
		5104	5105	5116	5138	5139	5143	5144	5163	5171	5173	5178	5181
		5186	5219	5221	5251	5253	5256	5257	5263	5298	5302	5304	5310
		5318	5320	5321	5322	5340	5341	5346	5373	5380	5381	5385	5388
		5389	5396	5401	5403	5415	5417	5418	5419	5422	5430	5433	5435
		5445	5447	5449	5456	5460	5462	5473	5479	5484	5488	5489	5492
		5504	5507	5529	5539	5541	5549	5550	5558	5565	5568	5580	5583
		5597	5602	5604	5625	5628	5650	5658	5744	5789	5809	5824	5844
		5855	5861	5890	5901	5937	5944	5978	6017	6037	6039	6074	6087
		6088	6124	6150	6156	6161	6162	6169	6173	6178	6200	6201	6202
		6236	6270	6279	6282	6286	6288	6326	6336	6361	6368	6404	6421
		6446	6476	6483	6494	6546	6563	6569	6660	6664	6683	6702	6705
		6709	6711	6743	6751	6758	6759	6763	6770	6825	6826	6912	6942
		6957	6959	6969	6975	6977	6984	6985	6991	6992	6994	6996	7094
		7126	7149	7167	7173	7244	7251	7272	7280	7299	7317	7341	7354
		7356	7363	7388	7392	7398	7420	7424	7430	7464	7521	7523	7529
		7615	7623	7650	7751	7990	8049	8052	8062	8086	8106	8109	8127
		8135	8242	8248	8266	8272	8288	8344	8510	8525			
	A+B	484 #	2087	2091	2136	2138	2183	2270	2321	2324	2690	3018	3245
		3311	3521	3539	3542	3682	3687	3852	3879	3885	3925	3941	4095
		4099	4128	4184	4188	4268	4364	4365	4368	4378	4451	4456	4482
		4486	4492	4635	5035	5037	5042	5059	5060	5117	5261	5265	5344
		5394	5475	5766	5830	5862	5867	5885	5914	5915	6020	6034	6045
		6047	6057	6125	6137	6144	6164	6185	6199	6254	6265	6273	6317
		6341	6435	6457	6484	6510	6662	6686	6704	6746	6766	6782	6790
		7047	7053	7076	7106	7247	7282	7292	7300	7519	7639	7725	7732
		7735	7738	7741	7744	8073	8081	8157	8374	8389	8485	8535	8539
	A+Q	483 #	4335	4346	4367	5003	5382	5390	5491	5494	5647		
	A-.25	495 #											
	A-B-.25	500 #	4264	4320	4321	5002	5014	5030	5054	5175	5183	5531	5778
		5913	5949	6168									
	A-D-.25	496 #	5998	6238	6243	6291	8370	8380					
	A-Q-.25	499 #											
	A.AND.B	516 #	4053	4068	4285	4356	4809	5326	5575	5608	5610	5621	5800
		5946	6844	6847	7159	7315	8316						
	A.AND.Q	515 #	4051	4292	4336	4380	4382	4458	4510	4515	4520	4526	4527

Cross Reference Listing


```
                4723    4801    4810    5571    5606
A.EQV.B         543 #
A.EQV.Q         542 #   4029    4032    4054    4055    4110    4112    4114    4383
A.OR.B          508 #   2814    3991    4811    4964    5732    5831    5864    5903    6061    6492    6864
                6868    6875    6906    6933    6950    6951    7138    7144    7286    7382    7417    8263
                8447
A.OR.Q          507 #   5405
A.XOR.B         535 #   5612    7092
A.XOR.Q         534 #
B               510 #   2067    2075    2077    2078    2081    2083    2090    2092    2283    2303    2402
                2434    2440    2445    2448    2479    2498    2643    2650    2657    2664    2694    2975
                2991    3047    3192    3236    3239    3242    3248    3251    3254    3313    3467    3515
                3526    3566    3602    3608    3647    3653    3724    3854    3856    3864    3868    3870
                3875    3897    3903    3914    3918    3933    3940    4021    4166    4170    4192    4196
                4227    4233    4247    4252    4254    4259    4326    4328    4345    4361    4370    4431
                4438    4459    4589    4598    4602    4617    4620    4636    4717    4783    4791    4799
                4848    4861    4881    4913    4920    5102    5106    5189    5191    5223    5227    5260
                5300    5319    5387    5486    5495    5544    5562    5600    5734    5737    5750    5752
                5753    5783    5795    5829    5845    5871    5872    5873    5888    5930    5933    5935
                5939    5952    5963    5971    5979    6110    6112    6155    6174    6187    6191    6193
                6242    6249    6275    6289    6294    6316    6320    6372    6392    6406    6458    6466
                6540    6561    6583    6615    6665    6667    6689    6729    6780    6877    6879    6880
                6882    6885    6889    6966    6967    7020    7024    7028    7032    7036    7050    7061
                7066    7082    7085    7101    7105    7107    7115    7116    7168    7180    7186    7193
                7418    7460    7483    7489    7491    7525    7527    7631    7638    7703    7712    7718
                7722    7723    7724    7726    7727    7728    7730    7731    7733    7734    7736    7737
                7739    7740    7742    7743    7745    7746    7748    7986    7996    7997    7999    8000
                8007    8017    8019    8021    8023    8027    8028    8059    8145    8152    8154    8174
                8284    8318    8368    8403    8441    8472    8477    8513    8520    8532    8534
B-.25           494 #
B-A-.25         492 #   3327    3434    3443    3446    4174    4178    4448    4479    5825    5846    5950
                5951    6062    6151    6203    6240    6633    6762    7058    7182    7195    7442    8475
                8527
D               514 #   2056    2060    2085    2086    2088    2126    2176    2246    2260    2264    2269
                2273    2277    2278    2290    2291    2296    2301    2312    2318    2476    2495    2596
                2598    2623    2625    2628    2629    2631    2634    2635    2637    2652    2654    2659
                2661    2688    2872    2889    2895    2907    2910    2948    2994    2997    3176    3181
                3294    3339    3342    3345    3351    3354    3357    3380    3456    3463    3480    3495
                3500    3520    3524    3585    3596    3624    3630    3657    3666    3677    3679    3713
                3719    3734    3735    3747    3748    3752    3860    3902    3915    4003    4018    4027
                4046    4050    4088    4093    4108    4111    4113    4115    4216    4225    4226    4308
                4437    4591    4596    4610    4612    4616    4631    4664    4690    4691    4692    4693
                4694    4710    4784    4785    4786    4787    4788    4836    4845    4852    4866    4868
                4872    4876    4878    4882    4911    4916    4939    4952    4954    4994    4995    5004
                5008    5050    5068    5069    5100    5133    5165    5229    5307    5323    5365    5375
                5467    5518    5653    5727    5746    5755    5770    5773    5787    5788    5791    5813
                5816    5821    5837    5838    5848    5858    5875    5876    5877    5878    5886    5892
                5900    5909    5911    5929    5931    5938    5942    5955    5957    5961    5973    5976
                5985    6004    6007    6009    6016    6026    6067    6069    6070    6075    6102    6107
                6108    6109    6121    6130    6135    6147    6153    6163    6166    6167    6182    6194
                6196    6197    6215    6218    6267    6272    6277    6278    6318    6325    6340    6356
                6366    6375    6379    6394    6403    6405    6409    6430    6437    6439    6470    6486
                6507    6509    6547    6572    6622    6627    6629    6636    6638    6655    6657    6658
                6661    6684    6701    6703    6713    6728    6741    6745    6749    6755    6774    6779
                6798    6818    6819    6851    6853    6860    6871    6894    6895    6903    6915    6962
```

Cross Reference Listing

	7006	7008	7010	7012	7014	7019	7023	7027	7031	7035	7043	7048
	7052	7057	7063	7064	7075	7095	7099	7114	7120	7134	7147	7157
	7176	7178	7189	7191	7215	7241	7252	7268	7270	7289	7294	7312
	7344	7385	7452	7455	7457	7465	7479	7510	7513	7515	7619	7626
	7658	7705	7710	7714	7717	7729	7747	7749	7750	7752	7983	7987
	7989	7998	8004	8009	8068	8071	8090	8146	8149	8163	8172	8212
	8214	8245	8285	8286	8287	8293	8349	8398	8406	8442	8452	8458
	8467	8471	8483	8487	8489	8491	8493	8495	8497	8501	8518	
D+A	488 #	2196	2208	2223	2236	3394	3416	3471	3586	3632	3661	3725
	3740	3956	3983	3986	4648	4656	4934	5006	5011	5740	5743	5807
	5840	5895	5993	6018	6028	6031	6171	6189	6232	6241	6248	6250
	6369	6445	6552	6560	6634	6640	6708	6710	7096	7245	7290	7463
	7469	7471	8076	8084	8098	8100	8168	8300	8384	8394	8506	
D+Q	489 #	4884	4892	4923	4942							
D-.25	506 #											
D-A-.25	504 #	3277	3408	3970	3995	3997	4000	4949	6176	6195		
D-Q-.25	505 #											
D.AND.A	523 #	2329	2331	2342	2364	2678	2721	2844	2875	2923	3183	3206
	3436	3445	3448	3450	3451	3452	3485	3490	3493	3499	3513	3514
	3534	3535	3547	3549	3826	3828	3830	3832	3834	3836	3838	3840
	3846	3861	3886	3887	3892	3936	4004	4074	4076	4257	4310	4332
	4339	4535	4670	4842	4849	4961	5107	5111	5136	5169	5206	5210
	5230	5325	5364	5455	5490	5493	5510	5561	5564	5567	5570	5630
	5728	5730	5771	5774	5910	5912	5948	5974	5981	5994	6014	6056
	6072	6079	6104	6120	6139	6198	6212	6214	6246	6252	6268	6280
	6283	6303	6306	6322	6337	6453	6487	6490	6553	6567	6723	6733
	6800	6806	6815	6816	6817	6821	6822	6823	6824	6829	6830	6831
	6834	6835	6836	6837	6838	6839	6840	6841	6863	6865	6867	6869
	6872	6874	6876	6897	6900	6918	6923	6926	6930	6945	6963	6964
	6970	6978	6986	6995	7015	7037	7079	7129	7131	7132	7133	7135
	7137	7139	7141	7143	7145	7256	7296	7330	7387	7394	7400	7414
	7446	7448	7462	7591	7593	7595	7597	7599	7601	7603	7605	7984
	7988	8037	8041	8057	8095	8102	8105	8119	8124	8165	8179	8183
	8187	8192	8201	8205	8207	8218	8223	8240	8251	8258	8261	8277
	8279	8302	8304	8306	8320	8323	8325	8328	8331	8351	8361	8399
	8410	8414	8417	8422	8428	8444	8469	8516	8517			
D.AND.Q	524 #	3020	4410	5141	5450							
D.EQV.A	547 #	2793										
D.EQV.Q	548 #											
D.OR.A	512 #	2058	2327	2365	2773	3201	3889	3894	4005	4358	4443	5108
	5112	5137	5170	5231	5370	5426	5427	5434	5471	5523	5637	5986
	6073	6117	6126	6131	6141	6412	6416	6418	6479	6592	6596	6600
	6608	6624	6760	6769	6965	6971	7065	7140	7218	7219	7220	7221
	7222	7223	7224	7358	7654	7980	8034	8038	8129	8199	8202	8208
	8219	8226	8255	8259	8276	8278	8282	8314	8415	8419	8424	8431
	8436	8509										
D.OR.Q	513 #	3022	4295									
D.XOR.A	539 #	2095	2763	3198	4312	7254	7255	8047				
D.XOR.Q	540 #											
Q	509 #	3024	4023	4031	4078	4086	4086	4097	4109	4251	4306	4323
	4329	4334	4338	4344	4352	4871	5027	5053	5188	5193	5264	5306
	5330	5424	5431	5451	5463	5474	5533	5535	5579	5624	5919	5958
	6360											
Q-.25	493 #											
Q-A-.25	491 #	4327	4932									

Cross Reference Listing

Cross Reference Listing

	3918	3921	3930	3933	3941	3956	3970	3986	3991	3991	3997	4000
	4003	4023	4029	4031	4032	4049	4050	4067	4115	4218	4220	4225
	4227	4252	4274	4308	4314	4318	4361	4365	4365	4368	4368	4372
	4375	4378	4392	4417	4431	4434	4435	4440	4445	4448	4451	4453
	4455	4456	4456	4457	4487	4523	4524	4528	4538	4540	4545	4546
	4589	4598	4602	4606	4617	4620	4629	4631	4635	4635	4636	4664
	4702	4717	4719	4722	4723	4783	4784	4785	4786	4787	4788	4791
	4799	4808	4809	4809	4811	4811	4866	4868	4878	4881	4882	4911
	4920	4934	4944	4949	4952	4954	4958	5030	5054	5068	5069	5071
	5097	5104	5106	5107	5108	5117	5117	5143	5144	5165	5169	5170
	5181	5186	5188	5189	5191	5193	5194	5219	5221	5223	5227	5229
	5230	5231	5251	5256	5257	5260	5261	5261	5263	5265	5265	5298
	5300	5302	5304	5307	5310	5314	5316	5318	5319	5320	5321	5322
	5323	5326	5326	5329	5331	5332	5341	5344	5344	5346	5362	5389
	5394	5394	5396	5401	5403	5447	5484	5488	5492	5495	5518	5521
	5523	5525	5549	5550	5558	5562	5565	5568	5580	5583	5591	5593
	5597	5600	5602	5604	5619	5625	5628	5650	5653	5654	5658	5695
	5697	5699	5701	5703	5705	5707	5709	5711	5737	5766	5766	5773
	5780	5782	5783	5798	5799	5807	5811	5824	5825	5825	5828	5830
	5830	5831	5831	5837	5844	5845	5855	5856	5858	5860	5861	5862
	5862	5867	5867	5875	5914	5914	5915	5916	5929	5935	5942	5946
	5946	5949	5949	5952	5956	5961	5963	5970	5971	5973	5981	5989
	5993	5996	6000	6069	6072	6073	6074	6082	6083	6087	6107	6109
	6112	6121	6130	6147	6161	6162	6163	6164	6164	6169	6187	6189
	6191	6196	6200	6213	6215	6216	6218	6240	6243	6248	6254	6254
	6265	6277	6278	6279	6280	6282	6283	6316	6336	6340	6341	6341
	6361	6366	6368	6369	6372	6394	6395	6407	6409	6430	6433	6435
	6435	6437	6445	6453	6457	6457	6458	6470	6473	6476	6479	6483
	6486	6490	6507	6509	6510	6540	6546	6547	6552	6560	6561	6563
	6569	6618	6622	6627	6629	6633	6634	6636	6638	6640	6655	6657
	6660	6661	6662	6662	6664	6665	6667	6684	6689	6702	6703	6704
	6704	6705	6713	6728	6729	6733	6741	6745	6746	6749	6754	6758
	6762	6762	6763	6766	6766	6768	6769	6770	6772	6782	6782	6790
	6790	6915	6918	6957	6959	7019	7020	7023	7024	7027	7028	7031
	7032	7035	7036	7043	7047	7047	7048	7050	7052	7053	7053	7055
	7057	7058	7058	7061	7063	7075	7085	7099	7114	7115	7116	7175
	7176	7180	7182	7182	7186	7189	7193	7195	7195	7233	7237	7241
	7244	7245	7247	7247	7252	7270	7280	7284	7286	7286	7341	7363
	7388	7394	7400	7449	7452	7457	7463	7465	7469	7471	7472	7510
	7519	7519	7619	7626	7653	7654	7727	7728	7729	7730	7747	7748
	7749	7980	7990	7996	8076	8084	8090	8098	8100	8106	8109	8127
	8129	8146	8149	8163	8168	8174	8179	8208	8212	8214	8219	8242
	8245	8248	8251	8263	8263	8293	8300	8304	8318	8349	8360	8452
	8458	8467	8471	8472	8483	8506	8525	8535	8535	8539		
ARX	610 #	2088	2092	2273	2278	2307	2331	2396	2678	2679	2688	2694
	3586	3617	3619	3624	3632	3657	3661	3734	3737	3850	3852	3852
	3856	3864	3879	3925	3983	3995	4004	4005	4021	4022	4027	4051
	4054	4055	4068	4068	4076	4086	4099	4099	4105	4108	4122	4128
	4148	4167	4171	4175	4179	4185	4189	4193	4197	4239	4251	4257
	4323	4329	4338	4432	4437	4443	4459	4479	4479	4482	4482	4493
	4535	4536	4544	4612	4616	4803	4810	4836	4839	4842	4845	4848
	4995	5035	5035	5060	5103	5171	5172	5183	5361	5463	5475	5475
	5510	5533	5608	5608	5610	5610	5612	5612	5614	5647	5789	5791
	5809	5840	5846	5872	5877	5885	5885	5888	5890	5895	5909	5913
	5933	5939	5944	5950	5976	5978	6007	6028	6034	6034	6037	6047

Cross Reference Listing

	6047	6056	6067	6079	6080	6085	6088	6108	6110	6120	6124	6125
	6125	6135	6153	6182	6193	6197	6199	6199	6201	6238	6246	6250
	6252	6392	6572	6615	6683	6686	6759	6760	6779	6780	6863	6865
	6903	6904	7079	7082	7090	7094	7096	7107	7243	7282	7282	7292
	7292	7415	7417	7426	7432	7455	7513	7517	7529	7631	7636	7639
	7703	7705	7710	7714	7717	7718	7722	7725	7732	7735	7738	7741
	7744	7750	7752	8000	8062	8068	8071	8125	8135	8152	8154	8193
	8274	8285	8314	8374	8374	8384	8389	8389	8394	8398	8403	8406
BR	611 #	2090	2291	2292	2318	2327	2329	2495	2498	2813	2923	2945
	2978	2979	2982	2983	2984	2985	2997	3000	3003	3006	3009	3013
	3016	3018	3050	3051	3054	3055	3178	3183	3192	3336	3339	3342
	3345	3348	3351	3354	3357	3380	3394	3408	3416	3520	3526	3535
	3585	3596	3602	3608	3630	3647	3653	3679	3713	3715	3719	3724
	3732	3747	3748	3752	3883	3885	3885	3889	3894	3897	3990	4069
	4215	4224	4233	4247	4254	4259	4264	4268	4286	4288	4290	4298
	4302	4320	4321	4438	4442	4446	4447	4591	4596	4610	4670	4690
	4691	4692	4693	4694	4710	4712	4714	4849	4852	4854	4858	4861
	4876	4886	4913	4925	4961	4964	5003	5004	5006	5009	5011	5100
	5102	5105	5111	5112	5116	5163	5207	5253	5266	5340	5365	5368
	5370	5381	5451	5489	5507	5514	5531	5727	5732	5732	5734	5740
	5741	5743	5746	5750	5752	5753	5755	5770	5778	5788	5795	5821
	5829	5848	5871	5876	5948	6004	6009	6016	6017	6018	6020	6026
	6031	6039	6045	6045	6057	6057	6070	6149	6150	6151	6151	6171
	6174	6176	6178	6185	6185	6195	6202	6204	6220	6272	6273	6275
	6286	6288	6289	6325	6334	6403	6405	6406	6658	6701	6743	6751
	6755	6774	6818	6819	6825	6826	6843	6844	6846	6847	6851	6853
	6860	6861	6864	6864	6868	6868	6870	6870	6879	6880	6882	6894
	6895	6897	6900	6906	6906	6930	6938	6942	6962	6963	6965	6966
	6967	6975	6977	6978	6979	6984	6986	6991	6996	7006	7008	7010
	7012	7014	7046	7064	7065	7066	7089	7092	7095	7105	7120	7126
	7133	7134	7168	7178	7191	7256	7268	7290	7294	7300	7300	7312
	7317	7330	7344	7356	7358	7360	7376	7379	7382	7385	7392	7398
	7418	7424	7430	7521	7525	7731	7987	8007	8017	8019	8021	8023
	8049	8052	8055	8073	8081	8157	8172	8176	8266	8267	8286	8441
	8442	8444	8517	8518	8520	8532						
BRX	612 #	2086	2087	2087	4017	4019	4046	4047	4074	4093	4124	4147
	4245	4306	4364	4937	4940	4994	5000	5002	5014	5037	5042	5059
	5059	5133	5136	5137	5139	5173	5175	5176	5178	5364	5455	5456
	5467	5470	5471	5504	5513	5730	5774	5864	5873	5878	5901	5903
	5911	5930	5937	5951	5974	5979	5986	6061	6061	6075	6102	6104
	6115	6117	6126	6131	6137	6137	6141	6144	6144	6155	6156	6173
	6203	6267	6294	6320	6322	6416	6418	6421	6466	6484	6484	6487
	6492	6492	6494	6583	6588	6592	6596	6600	6604	6608	6624	6871
	6873	6873	6875	6875	6877	6885	6887	6889	6985	6992	6994	6995
	7091	7225	7226	7227	7228	7229	7230	7231	7296	7299	7354	7445
	7505	7733	7983	7984	7986	7997	7998	7999	8009	8013	8015	8027
	8028	8029	8038	8041	8047	8059	8145	8199	8202	8226	8255	8259
	8261	8276	8278	8282	8284	8287	8368	8419	8431	8436	8447	8447
	8513											
EBR	614 #	2070	3854	6969	6971	6972	7736					
FLG	617 #	2073	2075	2077	2078	3528	5008	5033	5057	5333	5426	5427
	5434	5573	5589	5617	5637	5638	5787	5813	5816	5832	5838	5886
	5892	5900	5904	5927	5931	5938	5957	5985	6077	6167	6194	6205
	6209	6271	6318	6375	6379	6412	6439	6543	7740	7989	8034	8230
	8254	8316	8316	8412	8415	8424	8477	8481	8487	8489	8491	8493

;

Cross Reference Listing

		8495	8497	8499	8500	8501	8509	8523				
	HR	608 #	2085	2091	2091	2176	2208	2223	2236	2246	2270	2342	3436
		3445	3448	3450	3451	3452	3456	3467	3471	3480	3485	3490	3493
		3495	3513	3534	3547	3560	3566	3570	3826	3828	3830	3832	3834
		3836	3838	3840	3846	3871	3875	3914	3936	3940	4501	6723	6798
		6806	6815	6816	6817	6821	6822	6823	6824	6829	6830	6831	6834
		6835	6836	6837	6838	6839	6840	6841	7015	7037	7076	7101	7106
		7132	7460	7591	7593	7595	7597	7599	7601	7603	7605	7658	7726
	MAG	606 #	2059	4126	4160	4285	4333	4356	4790	4795	4798	5449	5539
		5575	5621	7723									
	MASK	616 #	2056	2057	2058	3515	4025	5800	7712	7739			
	ONE	613 #	2062	2081	2083	2690	6168	7734					
	PC	607 #	2123	2136	2138	2183	2196	2202	2926	3245	3361	3364	3432
		3433	3434	3443	3446	3459	3486	3501	3524	3614	3666	3725	3740
		3870	6062	6317	6317	7272	7442	7527	7724	8475	8485	8485	8527
		8534											
	PI	618 #	2101	7130	7136	7136	7138	7138	7140	7142	7144	7144	7146
		7146	7218	7219	7220	7221	7222	7223	7224	7315	7316	7316	7742
	TO	620 #	3500	4045	4053	4085	4088	4095	4334	4340	4344	4346	4348
		4353	4358	4359	5375	5380	5387	5388	5415	5417	5418	5419	5422
		5430	5433	5435	5462	5535	5541	6166	6232	6241	6242	7147	7157
		7158	7159	7159	7289	7479	7483	7489	7491	7745			
	T1	621 #	2098	2109	2126	2155	2173	2424	3487	4294	4305	4312	4326
		4328	4345	4370	4452	5399	5399	5424	5431	5474	5486	5506	5512
		5522	5544	6236	6249	7266	7298	7638	7746	8466			
	UBR	615 #	2068	6923	6933	6933	6945	6948	6950	6950	6951	6951	7737
	XWD1	619 #	2060	7743									
(D)	B	1235 #	3084	3085	3086	3087	3088	3089	3093	3094	3095	3096	3097
		3098	3100	3101	3102	3103	3104	3105	3106	3107	3109	3110	3111
		3112	3113	3114	3115	3116	3118	3119	3120	3121	3122	3123	3124
		3125	3127	3128	3129	3130	3131	3132	3133	3134	3135	3136	3137
		3138	3139	3140	3141	3142	3144	3145	3146	3147	3148	3149	3150
		3151	3257	3258	3259	3260	3261	3262	3263	3264	3266	3267	3268
		3269	3270	3271	3272	3273	3283	3284	3285	3286	3287	3288	3289
		3290	3300	3301	3302	3303	3304	3305	3306	3307	3316	3317	3318
		3319	3320	3321	3322	3323	3369	3370	3371	3372	3373	3374	3375
		3376	3383	3384	3385	3386	3387	3388	3389	3390	3397	3398	3399
		3400	3401	3402	3403	3404	3411	3412	3576	3577	3578	3673	3759
		3760	3761	3762	3763	3764	3765	3766	3816	3817	3818	3819	3820
		3821	5666	5667	5668	5669	5670	5671	5672	5674	5675	5676	5677
		5679	5680	5681	5682	5687	5688	5689	5690	5691	7322	7323	7324
		7325	7333	7334	7335	7336	7347	7348	7349	7350	7534	7535	7536
		7538	7539	7541	7542	7544	7545	7546	7547	7548	7549	7550	7551
		7553	7554	7555	7556	7557	7558	7559	7560	7562	7563	7564	7565
		7566	7567	7568	7569	7571	7572	7573	7574	7575	7576	7577	7578
		7580	7581	7582	7583	7584	7585	7586	7587				
	AC	1239 #	2454	2455	2459	2460	2464	2465	2469	2470	2491	2507	2508
		2512	2513	2517	2518	2522	2523	2527	2528	2532	2533	2537	2538
		2542	2543	2549	2550	2554	2555	2559	2560	2564	2565	2569	2570
		2574	2575	2579	2580	2584	2585	2674	2704	2705	2714	2715	2724
		2725	2734	2735	2739	2740	2756	2757	2766	2767	2776	2777	2786
		2787	2796	2797	2806	2807	2817	2818	2827	2828	2837	2838	2847
		2848	3949	3950	3963	3964	4011	4012	4581	7976			
	BOTH	1241 #	2707	2717	2727	2737	2742	2759	2769	2779	2789	2799	2809
		2820	2830	2840	2850	3952	3966	4014					

Cross Reference Listing

```
        DBLAC           1237 #  2673    3978    3979    4036    4037    4062    4203    4204    4208    4209    4280
                        5440    5501
        DBLB            1238 #  4039    4206    4211
        MEM             1240 #  2456    2461    2466    2471    2509    2514    2519    2524    2529    2534    2539
                        2544    2551    2556    2561    2566    2571    2576    2581    2586    2706    2716    2726
                        2736    2741    2751    2752    2758    2768    2778    2788    2798    2808    2819    2829
                        2839    2849    3951    3965    4013    4038    4205    4210    7001
        SELF            1236 #  2457    2462    2467    2472    2510    2515    2520    2525    2530    2535    2540
                        2545    2552    2557    2562    2567    2572    2577    2582    2587
(U) BWRITE              1106 #  2476    2482    2596    2598    2631    2637    2645    2647    2666    2668    2711
                        2721    2731    2763    2773    2793    2803    2814    2824    2854    3957    3971    4023
                        4029    4052    4054    5261    5265    5311    5333    8264    8526
(U) BYTE                765 #
        BYTE1           766 #   4591    4596    4610    4631    4784    5848    5858    6340    6486    6658    6661
                        6701    6703    6774
        BYTE2           767 #   4785
        BYTE3           768 #   4786
        BYTE4           769 #   4787
        BYTE5           770 #   2948    4690    4691    4692    4693    4694    4788    6470
(U) CALL                917 #   2063    2089    2101    2306    2678    2688    3458    3497    3499    3500    3514
                        3538    3541    3736    3739    3853    3863    3868    3872    3877    3880    3914    3916
                        4020    4048    4075    4084    4094    4104    4127    4237    4258    4313    4319    4333
                        4337    4343    4389    4413    4591    4596    4600    4605    4610    4619    4622    4870
                        4880    4914    4941    4999    5010    5140    5182    5185    5190    5192    5300    5302
                        5304    5319    5320    5321    5380    5388    5416    5456    5461    5473    5480    5487
                        5506    5530    5534    5538    5563    5566    5569    5601    5603    5605    5611    5729
                        5769    5772    5776    5808    5814    5839    5863    5874    5889    5893    5900    5910
                        5912    5918    5932    5941    5962    5977    5980    5993    6008    6019    6027    6030
                        6033    6036    6044    6049    6076    6111    6120    6136    6148    6175    6186    6194
                        6211    6237    6266    6269    6274    6349    6357    6367    6370    6380    6393    6408
                        6415    6436    6440    6508    6542    6566    6666    6685    6690    6742    6750    6761
                        6985    7054    7078    7081    7100    7147    7177    7190    7239    7269    7271    7287
                        7291    7329    7340    7355    7374    7383    7412    7419    7444    7632    7634    7640
                        7656    7704    7706    7723    7726    7728    7730    7733    7736    7739    7742    7745
                        7747    8094    8101    8112    8118    8128    8132    8148    8151    8164    8186    8195
                        8213    8236    8301    8443    8454    8514    8533
(U) CHKL                670 #   2176    2246    2260    2269    2273    2290    2301    2312    2318    2331    2402
                        2434    2440    2445    2448    2498    2694    2994    3313    3456    3480    3495    3520
                        3524    3585    3602    3608    3630    3647    3653    3666    3724    3752    3868    3870
                        3875    3902    3914    3918    3940    4018    4111    4113    4216    4226    4310    4332
                        4339    4636    4664    4670    4791    5050    5727    5746    6166    6356    6406    6409
                        6437    6713    6728    6798    6915    7019    7023    7027    7031    7035    7075    7105
                        7107    7114    7168    7252    7294    7418    7465    7510    7513    7515    7525    7527
                        7619    7638    7658    7723    7724    7726    7727    7730    7731    7733    7734    7736
                        7737    7739    7740    7742    7743    7745    7746    7748    8090    8163    8349    8398
                        8513    8520    8532    8534
(U) CHKR                677 #   2176    2246    2260    2269    2273    2290    2301    2312    2318    2331    2402
                        2434    2440    2445    2448    2498    2694    2994    3313    3456    3480    3495    3520
                        3524    3585    3602    3608    3630    3647    3653    3666    3724    3752    3868    3870
                        3875    3902    3914    3918    3940    4018    4111    4113    4216    4226    4310    4332
                        4339    4636    4664    4670    4791    5050    5727    5746    6166    6356    6406    6409
                        6437    6713    6728    6798    6915    7019    7023    7027    7031    7035    7075    7105
                        7107    7114    7168    7252    7294    7418    7465    7510    7513    7515    7525    7527
                        7619    7638    7658    7723    7724    7726    7727    7730    7731    7733    7734    7736
                        7737    7739    7740    7742    7743    7745    7746    7748    8090    8163    8349    8398
```

```
                8513  8520  8532  8534
(U) CLKL        666 #  2136  2138  2183  2196  2197  2203  2208  2223  2224  2236  2247
                2270  2598  2629  2637  2659  2661  3245  3361  3364  3432  3433  3434
                3443  3446  3459  3471  3501  3525  3539  3667  3726  3738  3742  3879
                3891  3896  3925  3941  3992  4635  4665  4874  4933  4956  5008  5013
                5033  5037  5042  5057  5060  5070  5142  5405  5740  5743  5746  5787
                5813  5816  5832  5838  5867  5886  5892  5900  5904  5915  5927  5931
                5938  5957  5985  6062  6077  6115  6150  6167  6194  6205  6209  6265
                6271  6318  6375  6379  6414  6439  6480  6485  6543  6662  6704  6713
                6862  6888  6902  6949  6963  6965  6996  7076  7081  7106  7140  7142
                7144  7146  7158  7218  7219  7220  7221  7222  7223  7224  7246  7316
                7442  7463  7463  7639  7655  7725  7732  7735  7738  7741  7744  7989
                8221  8254  8262  8475  8481  8487  8489  8491  8493  8495  8497  8499
                8500  8501  8523  8539
(U) CLKR        673 #  2327  2329  2342  2364  2365  2596  2631  2635  2652  2654  2678
                3021  3023  3294  3436  3445  3448  3450  3451  3452  3485  3490  3493
                3513  3534  3547  3678  3826  3828  3830  3832  3834  3836  3838  3840
                3848  3862  3872  3922  3938  4003  4004  4005  4027  4050  4257  4296
                4358  4412  4535  4712  4847  4855  4918  4963  5001  5005  5107  5108
                5111  5112  5136  5137  5169  5170  5230  5231  5307  5323  5426  5427
                5434  5450  5572  5607  5637  5638  5653  5733  5755  5982  5987  6056
                6072  6073  6079  6118  6120  6127  6133  6142  6161  6213  6216  6221
                6246  6252  6323  6417  6419  6489  6493  6547  6589  6593  6597  6601
                6605  6609  6625  6723  6735  6806  6815  6816  6817  6821  6822  6823
                6824  6829  6830  6831  6834  6835  6836  6837  6838  6839  6840  6841
                6856  6896  6899  6920  6925  6935  6940  6947  6952  6971  6972  6980
                6987  7015  7037  7048  7132  7134  7136  7138  7270  7472  7517  7591
                7593  7595  7597  7599  7601  7603  7605  7982  7985  8036  8040  8043
                8048  8181  8200  8228  8232  8253  8257  8260  8271  8317  8416  8421
                8426  8432  8446  8509
(U) CLRFPD      1035 #  3597  3714  3720  4106  4606  4623  6205
(D) COND FUNC   1258 #  2456  2457  2461  2462  2466  2467  2471  2472  2509  2510  2514
                2515  2519  2520  2524  2525  2529  2530  2534  2535  2539  2540  2544
                2545  2551  2552  2556  2557  2561  2562  2566  2567  2571  2572  2576
                2577  2581  2582  2586  2587  2706  2707  2716  2717  2726  2727  2736
                2737  2741  2742  2751  2752  2758  2759  2768  2769  2778  2779  2788
                2789  2798  2799  2808  2809  2819  2820  2829  2830  2839  2840  2849
                2850  3951  3952  3965  3966  4013  4014  4038  4039  4205  4206  4210
                4211  5078  5079  5082  5083  5086  5087  5090  5091  5123  5124  5128
                5129  5152  5153  5157  5158  7001
(U) CRY38       900 #  2484  2854  3277  3327  3408  3434  3443  3446  3970  3987  3995
                4000  4174  4178  4220  4264  4297  4320  4321  4327  4348  4374  4415
                4416  4417  4433  4435  4442  4448  4479  4509  4514  4519  4524  4536
                4540  4544  4546  4875  4932  4949  5002  5014  5030  5054  5097  5172
                5175  5176  5183  5194  5207  5299  5301  5303  5305  5316  5332  5361
                5400  5513  5524  5531  5547  5588  5593  5614  5620  5654  5778  5811
                5825  5846  5913  5949  5950  5951  5998  6062  6080  6083  6151  6168
                6176  6195  6203  6204  6238  6240  6243  6291  6633  6754  6762  6768
                7058  7158  7182  7195  7442  8370  8380  8475  8527
(U) DBM         645 #
    APR FLAGS   648 #  6818  6819  6871  6903  7064
    BYTES       649 #
    DP          651 #  2948  4591  4596  4610  4631  4690  4691  4692  4693  4694  4784
                4785  4786  4787  4788  5848  5858  6340  6470  6486  6658  6661  6701
                6703  6774
```

Cross Reference Listing


```
        DP SWAP         652 #   2296    2476    2623    2625    2628    2631    2634    2637    3176    3181    3500
                        3677    3734    3748    3860    4710    4836    4852    4952    4995    5004    5068    6215
                        6272    6277    6278    6572    6895    7134    7147    7157    8172    8406
        EXP             650 #   5307    5323    5653    7089    7090    7091
        MEM             654 #   2068    2175    2246    2259    2268    2272    2300    2311    2317    2330    3455
                        3479    3494    3516    3520    3524    3584    3629    3666    3751    3901    4663    4669
                        5049    5727    5746    6166    6409    6437    6713    6727    6783    6791    6798    6914
                        7019    7023    7027    7031    7035    7074    7114    7236    7240    7252    7288    7294
                        7384    7465    7510    7513    7515    7618    7633    7657    7706    8033    8089    8162
                        8348    8397    8456    8462
        PF DISP         647 #   8003
        SCAD DIAG       646 #
        VMA             653 #   7479    7717    7983    7998    8442
        #               655 #   2056    2058    2060    2062    2065    2085    2086    2095    2098    2109    2126
                        2145    2149    2152    2155    2164    2167    2170    2173    2327    2329    2342    2364
                        2365    2424    2652    2654    2659    2661    2678    2949    3020    3022    3436    3445
                        3448    3450    3451    3452    3485    3487    3490    3493    3499    3513    3514    3528
                        3534    3547    3632    3661    3725    3740    3826    3828    3830    3832    3834    3836
                        3838    3840    3846    3850    3883    3887    3889    3892    3894    3921    3930    3936
                        4004    4005    4257    4294    4295    4305    4312    4358    4410    4437    4443    4535
                        4700    4842    4845    4849    4868    4872    4878    4882    4911    4916    4949    4954
                        4961    5008    5011    5107    5108    5111    5112    5136    5137    5141    5169    5170
                        5206    5210    5230    5231    5325    5368    5370    5426    5427    5434    5450    5470
                        5471    5490    5493    5521    5523    5561    5564    5567    5570    5630    5637    5638
                        5695    5697    5699    5701    5703    5705    5707    5709    5711    5728    5730    5755
                        5771    5774    5787    5813    5816    5838    5886    5892    5900    5910    5912    5931
                        5938    5957    5970    5974    5981    5985    5986    5994    5998    6014    6056    6072
                        6073    6079    6085    6104    6117    6120    6126    6131    6139    6141    6167    6194
                        6198    6212    6214    6220    6232    6241    6246    6252    6268    6280    6283    6291
                        6303    6306    6318    6322    6337    6375    6379    6395    6407    6412    6416    6418
                        6433    6439    6453    6473    6479    6487    6490    6567    6588    6592    6596    6600
                        6604    6608    6618    6624    6723    6733    6800    6806    6815    6816    6817    6821
                        6822    6823    6824    6829    6830    6831    6834    6835    6836    6837    6838    6839
                        6840    6841    6843    6846    6851    6853    6861    6863    6865    6867    6869    6872
                        6874    6876    6887    6897    6900    6904    6918    6923    6926    6930    6945    6963
                        6964    6965    6970    6971    6972    6978    6986    6995    7015    7037    7046    7065
                        7079    7129    7131    7132    7133    7135    7137    7139    7140    7141    7142    7143
                        7145    7176    7178    7189    7191    7218    7219    7220    7221    7222    7223    7224
                        7225    7226    7227    7228    7229    7230    7231    7233    7237    7245    7254    7255
                        7266    7270    7284    7290    7296    7298    7312    7376    7379    7387    7394    7400
                        7414    7415    7426    7432    7446    7448    7462    7505    7591    7593    7595    7597
                        7599    7601    7603    7605    7653    7654    7980    7984    7988    7989    8013    8015
                        8029    8034    8037    8038    8041    8047    8055    8057    8076    8084    8095    8102
                        8105    8119    8124    8125    8165    8176    8179    8183    8187    8192    8193    8199
                        8201    8202    8205    8207    8218    8219    8223    8226    8230    8240    8251    8255
                        8258    8259    8261    8267    8274    8276    8277    8278    8279    8282    8302    8306
                        8314    8320    8323    8325    8328    8331    8351    8361    8370    8380    8384    8394
                        8399    8410    8412    8414    8415    8417    8419    8422    8424    8428    8431    8436
                        8444    8466    8469    8487    8489    8491    8493    8495    8497    8501    8506    8509
                        8516
(U) DBUS                636 #
        DBM             642 #   2056    2058    2060    2062    2065    2068    2085    2086    2095    2098    2109
                        2126    2145    2149    2152    2155    2164    2167    2170    2173    2175    2176    2246
                        2246    2259    2260    2268    2269    2272    2273    2296    2300    2301    2311    2312
                        2317    2318    2327    2329    2330    2331    2342    2364    2365    2424    2476    2623
```

Cross Reference Listing

	2625	2628	2631	2634	2637	2652	2654	2659	2661	2678	2948	2949
	3020	3022	3176	3181	3436	3445	3448	3450	3451	3452	3455	3456
	3479	3480	3485	3487	3490	3493	3494	3495	3499	3500	3513	3514
	3516	3520	3520	3524	3524	3528	3534	3547	3584	3585	3629	3630
	3632	3661	3666	3666	3677	3725	3734	3740	3748	3751	3752	3826
	3828	3830	3832	3834	3836	3838	3840	3846	3850	3860	3883	3887
	3889	3892	3894	3901	3902	3921	3930	3936	4004	4005	4257	4294
	4295	4305	4312	4358	4410	4437	4443	4535	4591	4596	4610	4631
	4663	4664	4669	4670	4690	4691	4692	4693	4694	4700	4710	4784
	4785	4786	4787	4788	4836	4842	4845	4849	4852	4868	4872	4878
	4882	4911	4916	4949	4952	4954	4961	4995	5004	5008	5011	5049
	5050	5068	5107	5108	5111	5112	5136	5137	5141	5169	5170	5206
	5210	5230	5231	5307	5323	5325	5368	5370	5426	5427	5434	5450
	5470	5471	5490	5493	5521	5523	5561	5564	5567	5570	5630	5637
	5638	5653	5695	5697	5699	5701	5703	5705	5707	5709	5711	5727
	5727	5728	5730	5746	5746	5755	5771	5774	5787	5813	5816	5838
	5848	5858	5886	5892	5900	5910	5912	5931	5938	5957	5970	5974
	5981	5985	5986	5994	5998	6014	6056	6072	6073	6079	6085	6104
	6117	6120	6126	6131	6139	6141	6166	6166	6167	6194	6198	6212
	6214	6215	6220	6232	6241	6246	6252	6268	6272	6277	6278	6280
	6283	6291	6303	6306	6318	6322	6337	6340	6375	6379	6395	6407
	6409	6409	6412	6416	6418	6433	6437	6437	6439	6453	6470	6473
	6479	6486	6487	6490	6567	6572	6588	6592	6596	6600	6604	6608
	6618	6624	6658	6661	6701	6703	6713	6713	6723	6727	6728	6733
	6774	6783	6791	6798	6798	6800	6806	6815	6816	6817	6818	6819
	6821	6822	6823	6824	6829	6830	6831	6834	6835	6836	6837	6838
	6839	6840	6841	6843	6846	6851	6853	6861	6863	6865	6867	6869
	6871	6872	6874	6876	6887	6895	6897	6900	6903	6904	6914	6915
	6918	6923	6926	6930	6945	6963	6964	6965	6970	6971	6972	6978
	6986	6995	7015	7019	7019	7023	7023	7027	7027	7031	7031	7035
	7035	7037	7046	7064	7065	7074	7075	7079	7089	7090	7091	7114
	7114	7129	7131	7132	7133	7134	7135	7137	7139	7140	7141	7142
	7143	7145	7147	7157	7176	7178	7189	7191	7218	7219	7220	7221
	7222	7223	7224	7225	7226	7227	7228	7229	7230	7231	7233	7236
	7237	7240	7241	7245	7252	7252	7254	7255	7266	7270	7284	7288
	7289	7290	7294	7294	7296	7298	7312	7376	7379	7384	7385	7387
	7394	7400	7414	7415	7426	7432	7446	7448	7462	7465	7465	7479
	7505	7510	7510	7513	7513	7515	7515	7591	7593	7595	7597	7599
	7601	7603	7605	7618	7619	7633	7653	7654	7657	7658	7706	7717
	7980	7983	7984	7988	7989	7998	8003	8009	8013	8015	8029	8033
	8034	8037	8038	8041	8047	8055	8057	8076	8084	8089	8090	8095
	8102	8105	8119	8124	8125	8162	8163	8165	8172	8176	8179	8183
	8187	8192	8193	8199	8201	8202	8205	8207	8218	8219	8223	8226
	8230	8240	8251	8255	8258	8259	8261	8267	8274	8276	8277	8278
	8279	8282	8302	8306	8314	8320	8323	8325	8328	8331	8348	8349
	8351	8361	8370	8380	8384	8394	8397	8398	8399	8406	8410	8412
	8414	8415	8417	8419	8422	8424	8428	8431	8436	8442	8444	8456
	8462	8466	8469	8487	8489	8491	8493	8495	8497	8501	8506	8509
	8516											
DP	640 #	2080	2081	2396	2402	2428	2434	2440	2445	2448	2498	2679
	2694	2929	2950	2985	3018	3024	3207	3294	3296	3313	3336	3339
	3339	3342	3342	3345	3345	3348	3351	3351	3354	3354	3357	3357
	3561	3602	3608	3617	3619	3647	3653	3682	3687	3715	3724	3868
	3870	3875	3914	3918	3940	4003	4027	4050	4078	4086	4095	4097
	4108	4108	4109	4110	4112	4114	4336	4352	4353	4354	4356	4359

Cross Reference Listing

		4378	4380	4383	4510	4515	4520	4526	4527	4528	4606	4636	4791
		4964	5009	5027	5053	5071	5257	5263	5575	5583	5619	5620	5621
		5628	5778	5824	5844	5851	5861	5864	5899	5901	5903	5937	5944
		5978	6020	6037	6039	6074	6075	6075	6087	6088	6095	6097	6121
		6121	6124	6156	6162	6173	6178	6200	6201	6202	6360	6361	6406
		6421	6494	6510	6547	6636	6636	6664	6705	6743	6751	6758	6763
		6770	7048	7105	7107	7168	7341	7418	7460	7525	7527	7529	7638
		7723	7724	7726	7727	7730	7731	7733	7734	7736	7737	7739	7740
		7742	7743	7745	7746	7748	7990	8477	8513	8520	8532	8534	
	PC FLAGS	637 #	3535	3549	3596	3713	3719	3861	3886	3915	4115	6779	7256
		7268	8517	8518									
	PI NEW	638 #	7215										
	RAM	641 #	2088	2196	2208	2223	2236	2264	2277	2278	2290	2291	2495
		2596	2598	2629	2635	2688	2721	2731	2763	2773	2793	2803	2813
		2844	2872	2875	2889	2895	2907	2910	2923	2994	2997	3183	3198
		3201	3206	3277	3380	3394	3408	3416	3463	3471	3586	3624	3657
		3679	3735	3747	3956	3970	3983	3986	3995	3997	4000	4018	4046
		4074	4076	4088	4093	4111	4113	4216	4225	4226	4308	4310	4332
		4339	4349	4509	4513	4514	4518	4519	4523	4524	4612	4616	4648
		4656	4866	4875	4876	4888	4892	4927	4934	4939	4947	4994	5006
		5069	5100	5133	5165	5229	5364	5365	5375	5455	5467	5510	5518
		5740	5743	5770	5773	5788	5791	5807	5811	5821	5828	5837	5840
		5856	5860	5875	5876	5877	5878	5895	5909	5911	5929	5942	5948
		5955	5961	5973	5976	5993	5996	6000	6004	6007	6009	6016	6018
		6026	6028	6031	6067	6069	6070	6082	6083	6102	6107	6108	6109
		6130	6135	6147	6153	6163	6171	6176	6182	6189	6195	6196	6197
		6218	6238	6243	6248	6250	6267	6325	6356	6366	6369	6394	6403
		6405	6430	6445	6507	6509	6552	6553	6560	6622	6627	6629	6634
		6638	6640	6655	6657	6684	6708	6710	6741	6745	6749	6754	6755
		6760	6768	6769	6772	6860	6894	6962	7006	7008	7010	7012	7014
		7043	7052	7057	7063	7095	7096	7099	7120	7330	7344	7358	7360
		7452	7455	7457	7463	7469	7471	7626	7705	7710	7714	7729	7747
		7749	7750	7752	7987	8068	8071	8098	8100	8129	8146	8149	8168
		8208	8212	8214	8245	8285	8286	8287	8293	8300	8304	8452	8458
		8467	8471	8483									
(U) DEST		571 #											
	A	572 #	2296	2312	2396	2428	2476	2623	2625	2628	2631	2634	2637
		2679	2950	2985	3176	3181	3207	3294	3296	3336	3339	3342	3345
		3348	3351	3354	3357	3500	3617	3619	3677	3715	3734	3748	3860
		4003	4027	4050	4108	4353	4359	4528	4591	4596	4606	4610	4631
		4690	4691	4692	4693	4694	4710	4784	4785	4786	4787	4788	4836
		4852	4952	4995	5004	5009	5068	5071	5257	5263	5307	5323	5583
		5628	5653	5824	5844	5848	5858	5861	5901	5937	5944	5978	6037
		6039	6074	6075	6087	6088	6121	6124	6156	6162	6173	6178	6200
		6201	6202	6215	6272	6277	6278	6340	6361	6421	6486	6494	6547
		6572	6636	6658	6661	6664	6701	6703	6705	6743	6751	6758	6763
		6770	6774	6895	7048	7134	7147	7157	7341	7529	7990	8172	8406
	AD	574 #	2056	2058	2060	2062	2065	2068	2070	2073	2082	2085	2086
		2087	2088	2091	2098	2101	2109	2123	2126	2136	2138	2145	2149
		2152	2155	2164	2167	2170	2173	2176	2183	2196	2202	2208	2223
		2227	2236	2246	2260	2264	2269	2270	2273	2277	2278	2301	2307
		2318	2321	2324	2327	2329	2342	2364	2365	2424	2484	2495	2596
		2598	2629	2635	2645	2647	2652	2654	2659	2661	2666	2668	2678
		2688	2711	2721	2731	2746	2763	2773	2783	2793	2803	2813	2814
		2824	2834	2844	2854	2872	2875	2889	2923	2926	2949	3178	3183

Cross Reference Listing

3195	3198	3201	3206	3245	3277	3311	3327	3361	3364	3380	3394
3408	3416	3432	3433	3436	3445	3448	3450	3451	3452	3456	3459
3471	3480	3485	3486	3487	3490	3493	3495	3501	3513	3520	3521
3524	3528	3534	3535	3539	3542	3547	3549	3560	3585	3586	3596
3614	3624	3630	3632	3657	3661	3666	3679	3713	3719	3725	3732
3735	3737	3740	3747	3752	3826	3828	3830	3832	3834	3836	3838
3840	3846	3850	3852	3861	3871	3879	3883	3885	3886	3889	3894
3902	3915	3921	3925	3930	3936	3941	3956	3970	3983	3986	3990
3991	3995	3997	4000	4004	4005	4017	4023	4029	4031	4032	4045
4046	4051	4054	4055	4069	4085	4099	4115	4122	4215	4218	4220
4224	4225	4239	4245	4251	4257	4274	4294	4305	4306	4312	4323
4329	4338	4346	4358	4365	4368	4417	4432	4434	4435	4437	4442
4443	4456	4523	4524	4535	4536	4538	4540	4544	4545	4546	4616
4629	4635	4664	4670	4712	4723	4808	4809	4810	4811	4845	4854
4866	4868	4876	4878	4882	4885	4911	4924	4934	4937	4943	4954
4961	4994	5000	5003	5006	5008	5011	5033	5035	5037	5042	5057
5059	5060	5069	5097	5100	5103	5104	5105	5107	5108	5111	5112
5117	5133	5136	5137	5143	5163	5165	5169	5170	5171	5172	5207
5229	5230	5231	5261	5265	5266	5314	5316	5326	5329	5331	5332
5333	5344	5361	5362	5375	5380	5381	5388	5389	5399	5424	5426
5427	5431	5434	5451	5462	5474	5475	5504	5506	5507	5512	5513
5514	5522	5533	5535	5549	5550	5573	5589	5591	5593	5608	5610
5612	5614	5617	5637	5638	5654	5695	5697	5699	5701	5703	5705
5707	5709	5711	5727	5730	5732	5740	5741	5743	5746	5755	5766
5770	5773	5774	5780	5782	5787	5788	5789	5791	5799	5807	5809
5811	5813	5816	5821	5825	5828	5830	5831	5832	5837	5838	5840
5846	5855	5856	5860	5862	5867	5875	5876	5877	5878	5885	5886
5890	5892	5895	5900	5904	5909	5911	5914	5915	5916	5927	5929
5931	5938	5942	5946	5948	5949	5950	5951	5956	5957	5961	5970
5973	5974	5976	5981	5985	5986	5993	5996	6000	6004	6007	6009
6018	6026	6028	6031	6034	6045	6047	6056	6057	6061	6062	6067
6069	6070	6072	6073	6077	6079	6080	6082	6083	6085	6102	6104
6107	6108	6109	6115	6117	6120	6125	6126	6130	6131	6135	6137
6141	6144	6147	6149	6150	6151	6153	6161	6163	6164	6166	6167
6169	6171	6176	6182	6185	6189	6194	6195	6196	6197	6199	6203
6204	6205	6209	6213	6216	6218	6220	6232	6236	6238	6240	6241
6243	6246	6248	6250	6252	6254	6265	6267	6271	6280	6283	6317
6318	6322	6325	6334	6336	6341	6366	6369	6375	6379	6394	6395
6403	6405	6407	6409	6412	6416	6418	6430	6433	6435	6437	6439
6445	6453	6473	6479	6484	6487	6490	6492	6507	6509	6543	6552
6560	6569	6588	6592	6596	6600	6604	6608	6618	6622	6624	6627
6629	6633	6634	6638	6640	6655	6657	6660	6662	6683	6684	6702
6704	6713	6723	6728	6733	6741	6745	6749	6754	6755	6759	6760
6762	6766	6768	6769	6772	6779	6782	6790	6798	6806	6815	6816
6817	6818	6819	6821	6822	6823	6824	6825	6826	6829	6830	6831
6834	6835	6836	6837	6838	6839	6840	6841	6843	6846	6851	6853
6860	6861	6863	6864	6865	6868	6870	6871	6873	6875	6887	6894
6897	6900	6903	6904	6906	6915	6918	6923	6930	6933	6938	6945
6948	6950	6951	6962	6963	6965	6969	6971	6972	6978	6979	6984
6986	6991	6995	6996	7006	7008	7010	7012	7014	7015	7019	7023
7027	7031	7035	7037	7043	7046	7047	7052	7053	7055	7057	7058
7063	7064	7065	7075	7076	7079	7089	7090	7091	7094	7095	7096
7099	7106	7114	7120	7126	7130	7132	7133	7136	7138	7140	7142
7144	7146	7158	7159	7175	7176	7178	7182	7189	7191	7195	7218
7219	7220	7221	7222	7223	7224	7225	7226	7227	7228	7229	7230

Cross Reference Listing

	7231	7233	7237	7241	7243	7245	7247	7252	7256	7266	7268	7270
	7272	7282	7284	7286	7289	7290	7292	7294	7298	7300	7312	7316
	7330	7344	7354	7356	7358	7360	7363	7376	7379	7385	7388	7394
	7400	7415	7426	7432	7445	7452	7455	7457	7463	7465	7469	7471
	7472	7479	7505	7510	7513	7517	7519	7591	7593	7595	7597	7599
	7601	7603	7605	7619	7626	7636	7639	7653	7654	7658	7705	7710
	7714	7717	7725	7729	7732	7735	7738	7741	7744	7747	7749	7750
	7752	7980	7983	7984	7987	7989	7998	8009	8013	8015	8029	8034
	8038	8041	8047	8049	8055	8068	8071	8076	8084	8090	8098	8100
	8125	8127	8129	8146	8149	8163	8168	8176	8179	8193	8199	8202
	8208	8212	8214	8219	8226	8230	8242	8245	8251	8254	8255	8259
	8261	8263	8266	8267	8274	8276	8278	8282	8285	8286	8287	8293
	8300	8304	8314	8316	8349	8360	8374	8384	8389	8394	8398	8412
	8415	8419	8424	8431	8436	8442	8444	8447	8452	8458	8466	8467
	8471	8475	8481	8483	8485	8487	8489	8491	8493	8495	8497	8499
	8500	8501	8506	8509	8517	8518	8523	8525	8527	8535	8539	
AD*.5	581 #	2059	2331	2333	2334	2891	2895	2907	2910	2912	2913	2917
	2948	3000	3003	4067	4068	4074	4076	4086	4088	4093	4124	4286
	4308	4314	4445	4446	4447	4453	4703	4714	4790	4839	4842	4849
	4858	5178	5186	5188	5193	5194	5253	5302	5304	5320	5321	5340
	5341	5346	5364	5365	5368	5370	5455	5456	5463	5467	5470	5471
	5489	5510	5518	5521	5523	5647	5658	6279	6282	6286	6288	6368
	6470	6476	6483	6546	6975	6977	6985	6992	6994	7296	7299	7317
	7392	7398	8052	8062								
AD*2	579 #	2057	2879	2902	2915	2918	2945	4022	4049	4105	4348	4488
	4612	4949	4958	5173	5176	5181	5256	6016	6017	6273	6457	6563
	6942	6957	6959	7244	7280	7424	7430	7521	8106	8109	8135	8248
PASS	576 #	2067	2075	2077	2078	2081	2083	2084	2090	2092	2135	2139
	2157	2214	2241	2250	2337	2402	2434	2440	2445	2448	2498	2694
	2695	3018	3233	3313	3360	3365	3439	3475	3537	3548	3570	3602
	3608	3637	3642	3647	3653	3682	3687	3721	3724	3749	3868	3870
	3875	3914	3918	3940	4025	4053	4095	4110	4112	4114	4117	4292
	4336	4356	4378	4380	4383	4510	4515	4520	4526	4527	4589	4592
	4601	4636	4652	4660	4791	4848	4861	4871	4881	4913	4996	5017
	5023	5575	5619	5620	5621	5737	5744	5750	5753	5778	5783	5800
	5845	5871	5872	5873	5888	5919	5935	5958	5971	5989	6020	6110
	6112	6168	6174	6187	6193	6270	6326	6392	6404	6406	6446	6510
	6540	6709	6711	6729	6746	6780	6844	6847	6882	6912	6967	7020
	7024	7028	7032	7036	7050	7061	7082	7085	7092	7105	7107	7115
	7116	7149	7167	7168	7173	7200	7201	7251	7315	7382	7417	7418
	7420	7449	7464	7483	7489	7491	7523	7525	7527	7615	7623	7631
	7638	7650	7703	7718	7723	7724	7726	7727	7728	7730	7731	7733
	7734	7736	7737	7739	7740	7742	7743	7745	7746	7748	7751	7986
	7996	7997	7999	8000	8004	8007	8017	8019	8021	8023	8027	8028
	8073	8081	8086	8152	8154	8157	8174	8272	8288	8344	8441	8510
	8513	8520	8532	8534								
Q_AD	575 #	2290	2888	2994	3020	3022	4018	4044	4083	4103	4111	4113
	4123	4216	4226	4285	4295	4297	4310	4332	4335	4337	4339	4350
	4363	4364	4367	4374	4382	4410	4415	4416	4433	4458	4509	4513
	4514	4518	4519	4718	4794	4801	4863	4872	4875	4892	4916	4932
	4939	5002	5050	5114	5138	5141	5185	5196	5208	5212	5214	5215
	5228	5241	5299	5301	5303	5305	5373	5382	5385	5390	5400	5405
	5445	5450	5460	5473	5479	5491	5494	5524	5529	5534	5536	5547
	5571	5587	5588	5606	5796	5955	6356	7515				
Q_Q*.5	580 #	2291	2292	2978	2997	3006	3050	4019	4047	4126	4147	4148

Cross Reference Listing

		4160	4167	4175	4185	4193	4288	4290	4298	4302	4318	4333	4340
		4392	4440	4940	5116	5139	5144	5219	5221	5251	5394	5419	5422
		5430	5433	5435	5449	5484	5525	5539	5565	5568	5602	5604	5650
	Q_Q*2	578 #	2898	2979	2982	2983	2984	3009	3013	3016	3051	3054	3055
		4171	4179	4189	4197	4334	4344	4372	4375	4448	4451	4452	4455
		4457	4479	4482	4494	4499	4719	4722	4795	4798	4803	5298	5310
		5318	5322	5396	5401	5403	5415	5417	5418	5447	5488	5492	5541
		5558	5580	5597	5625	5798							
(U)	DISP	810 #	4460										
	ADISP	817 #											
	AREAD	813 #	2228										
	BDISP	818 #	2476	2482	2596	2598	2631	2637	2645	2647	2666	2668	2711
		2721	2731	2763	2773	2793	2803	2814	2824	2854	3178	3184	3277
		3296	3297	3380	3394	3408	3418	3603	3610	3648	3957	3971	4023
		4029	4052	4054	5253	5261	5265	5311	5323	5333	5340	5742	5784
		5928	5983	6190	6544	6616	6623	7330	7357	7375	7386	7413	8264
		8526											
	BYTE	823 #	4604	4621	5965	6668	6689						
	CONSOLE	811 #											
	DP	816 #	4326	4328	4345	4370	6289	6316	8477				
	DP LEFT	814 #	6275	6570	7216	8005							
	DROM	812 #	2261	2265	2274	2279	2287	2308	2314	2338	2343	5751	5754
	EAMODE	824 #	2186	2252	3461	4643	4646	5738	6706	6714	7454		
	MUL	820 #	4167	4175	4185	4193	5397	5423	5425	5542			
	NICOND	822 #	2135	2136	2138	2139	2199	2205	2429	2950	2996	3024	3233
		3245	3336	3360	3361	3364	3365	3432	3433	3459	3501	3618	3694
		3701	4117	4381	4384	4592	5257	5263	5622	7149	7420	7990	
	NORM	815 #	5117	5146	5223	5298	5300	5310	5314	5316	5318	5319	5322
		5344	5495	5549	5550	5560	5562	5582	5592	5594	5599	5600	5627
		5656											
	PAGE FAIL	821 #											
	RETURN	819 #	2364	2365	3466	3470	3923	3926	4129	4172	4180	4190	4198
		4392	4414	4415	4417	4501	4528	4630	4632	4636	4671	4704	4723
		4791	5343	5345	5346	5435	5630	5637	5638	5653	6040	6050	6055
		6058	6096	6097	6247	6253	6424	6447	6512	6543	6555	6586	6602
		6619	6628	6630	6656	6691	6724	6730	6735	6775	6996	7062	7086
		7161	7205	7313	7316	7388	7395	7401	7458	7466	7470	7473	7499
		7511	7513	7515	7517	7519	7521	7523	7525	7527	7529	7715	7719
		7753	8294	8305	8334	8337	8338	8350	8359	8362	8363	8462	8539
	SCADO	825 #	4591	4596	4610	4686	5100	5243	5245	5796	5798	5849	6659
		6701	7485										
(U)	DIVIDE	911 #	4451	4452	4496	4500							
(U)	DONT CACHE	1081 #											
(U)	DP FUNC	1098 #	7382	7417	7523	8273	8289						
(U)	DT	880 #											
	2T	883 #											
	3T	884 #	2088	2278	2290	2291	2401	2433	2439	2444	2447	2497	2688
		2694	2895	2907	2910	2924	2994	2997	3239	3242	3251	3254	3313
		3342	3345	3354	3357	3499	3499	3514	3514	3601	3607	3646	3652
		3723	3867	3869	3874	3887	3892	3913	3917	3939	3983	3995	4021
		4026	4088	4093	4111	4113	4226	4228	4230	4234	4248	4287	4293
		4308	4310	4322	4332	4339	4352	4433	4509	4511	4513	4514	4516
		4518	4519	4521	4537	4544	4591	4596	4604	4610	4612	4621	4636
		4791	4862	4866	4875	4876	4891	4921	4934	4939	5006	5026	5052
		5069	5117	5146	5163	5183	5189	5191	5223	5264	5298	5300	5306

Cross Reference Listing

	5310	5314	5316	5318	5319	5322	5325	5330	5344	5365	5467	5486
	5490	5493	5495	5507	5518	5537	5549	5550	5560	5561	5562	5564
	5567	5570	5579	5582	5588	5592	5594	5599	5600	5624	5627	5630
	5648	5656	5728	5770	5771	5788	5791	5807	5811	5821	5828	5837
	5840	5856	5860	5875	5876	5877	5878	5895	5909	5910	5912	5929
	5942	5948	5952	5955	5961	5965	5976	5993	5994	5996	6000	6004
	6005	6007	6009	6014	6016	6018	6026	6028	6031	6067	6069	6070
	6081	6082	6083	6084	6102	6108	6135	6139	6147	6152	6153	6163
	6171	6176	6182	6189	6195	6196	6197	6198	6212	6214	6218	6268
	6303	6306	6325	6356	6360	6366	6369	6373	6394	6403	6405	6406
	6410	6430	6431	6445	6507	6509	6552	6553	6554	6560	6567	6622
	6627	6629	6634	6638	6640	6655	6657	6668	6684	6689	6741	6745
	6749	6754	6755	6760	6768	6769	6772	6818	6819	6860	6867	6869
	6871	6872	6874	6876	6894	6903	6962	7006	7008	7010	7012	7014
	7043	7052	7057	7059	7063	7064	7093	7095	7096	7099	7105	7107
	7120	7129	7131	7135	7137	7139	7141	7143	7145	7168	7382	7387
	7414	7417	7418	7446	7448	7452	7455	7457	7462	7469	7471	7523
	7525	7527	7638	7705	7710	7711	7714	7723	7724	7726	7727	7729
	7730	7731	7733	7734	7736	7737	7739	7740	7742	7743	7745	7746
	7747	7748	7749	7750	7752	7987	7988	8037	8057	8068	8069	8071
	8072	8095	8098	8100	8102	8105	8119	8124	8129	8146	8147	8149
	8150	8165	8168	8183	8187	8192	8201	8205	8207	8208	8212	8214
	8218	8223	8240	8245	8258	8277	8279	8285	8286	8287	8293	8300
	8302	8304	8306	8320	8323	8325	8328	8331	8351	8361	8399	8410
	8414	8417	8422	8428	8452	8458	8467	8469	8471	8483	8512	8516
	8519	8532	8534									
4T	885 #											
5T	886 #											
(U) EXT ADR	1104 #	3858	3866	3879	3899	3925	6787	6795	7251	7293	7382	7417
	7523	7636	7639	7722	7725	7732	7735	7738	7741	7744	8074	8082
	8088	8159	8346	8376	8392	8511	8536	8539				
(U) FETCH	1070 #	2135	2136	2138	2139	2158	2184	2198	2204	2251	2312	2337
	2927	3233	3245	3360	3361	3364	3365	3432	3433	3459	3501	3616
	4117	4592	4601	7149	7420	7651	8469					
(D) FL-B	1246 #											
AC	1247 #	5077	5080	5081	5085	5088	5089	5122	5126	5127	5151	5155
	5156	5201	5202	5236	5237							
BOTH	1249 #	5079	5083	5087	5091	5124	5129	5153	5158			
MEM	1248 #	5078	5082	5086	5090	5123	5128	5152	5157			
(U) FLG.C	1212 #	8035	8258	8279	8361	8413	8416					
(U) FLG.PI	1211 #	3529	8105	8124	8183	8192	8207	8306	8323	8328	8351	8410
	8509											
(U) FLG.SN	1213 #	5325	5426	5427	5434	5561	5564	5567	5570	5630	5637	5638
(U) FLG.W	1210 #	8035	8201	8231	8277	8361	8413	8425				
(U) FMWRITE	906 #	2067	2075	2077	2078	2080	2081	2083	2084	2090	2092	2396
	2428	2679	2929	2950	2985	3018	3024	3207	3296	3336	3339	3342
	3345	3348	3351	3354	3357	3570	3617	3619	3682	3687	3715	4078
	4086	4095	4097	4108	4109	4110	4112	4114	4336	4352	4353	4354
	4356	4359	4378	4380	4383	4510	4515	4520	4526	4527	4528	4589
	4606	4848	4861	4871	4881	4913	4964	5009	5071	5257	5263	5575
	5583	5619	5620	5621	5628	5737	5750	5753	5778	5783	5800	5824
	5844	5845	5851	5861	5864	5871	5872	5873	5888	5899	5901	5903
	5919	5935	5937	5944	5958	5971	5978	5989	6020	6037	6039	6074
	6075	6087	6088	6095	6097	6110	6112	6121	6124	6156	6162	6168
	6173	6174	6178	6187	6193	6200	6201	6202	6361	6392	6421	6494

```
                    6510    6540    6636    6664    6705    6729    6743    6746    6751    6758    6763    6770
                    6780    6882    6967    7020    7024    7028    7032    7036    7050    7061    7082    7085
                    7115    7116    7200    7201    7341    7449    7483    7489    7491    7529    7631    7703
                    7718    7728    7986    7990    7996    7997    7999    8000    8007    8017    8019    8021
                    8023    8027    8028    8152    8154    8174    8441
(U) FORCE EXEC      1068 #  3858    3866    3879    3899    3925    6787    6795    7237    7251    7253    7267
                    7284    7293    7301    7376    7379    7415    7426    7432    7636    7639    7653    7722
                    7725    7732    7735    7738    7741    7744    8074    8082    8088    8159    8346    8376
                    8392    8511    8536    8539
(U) FORCE USER      1067 #  3858    3866    3879    3899    3925    6787    6795    7237    7251    7284    7293
                    7376    7379    7415    7426    7432    7636    7639    7653    7722    7725    7732    7735
                    7738    7741    7744    8074    8082    8088    8159    8346    8376    8392    8511    8536
                    8539
(U) GENL            668 #   2136    2138    2183    2196    2197    2203    2208    2223    2224    2236    2247
                    2270    2598    2629    2637    2659    2661    3245    3361    3364    3432    3433    3434
                    3443    3446    3459    3471    3501    3525    3539    3667    3726    3738    3742    3879
                    3891    3896    3925    3941    3992    4635    4665    4874    4933    4956    5008    5013
                    5033    5037    5042    5057    5060    5070    5142    5405    5740    5743    5746    5787
                    5813    5816    5832    5838    5867    5886    5892    5900    5904    5915    5927    5931
                    5938    5957    5985    6062    6077    6115    6150    6167    6194    6205    6209    6265
                    6271    6318    6375    6379    6414    6439    6480    6485    6543    6662    6704    6713
                    6862    6888    6902    6949    6963    6965    6996    7076    7081    7106    7140    7142
                    7144    7146    7158    7218    7219    7220    7221    7222    7223    7224    7246    7316
                    7442    7463    7463    7639    7655    7725    7732    7735    7738    7741    7744    7989
                    8221    8254    8262    8475    8481    8487    8489    8491    8493    8495    8497    8499
                    8500    8501    8523    8539
(U) GENR            675 #   2327    2329    2342    2364    2365    2596    2631    2635    2652    2654    2678
                    3021    3023    3294    3436    3445    3448    3450    3451    3452    3485    3490    3493
                    3513    3534    3547    3678    3826    3828    3830    3832    3834    3836    3838    3840
                    3848    3862    3872    3922    3938    4003    4004    4005    4027    4050    4257    4296
                    4358    4412    4535    4712    4847    4855    4918    4963    5001    5005    5107    5108
                    5111    5112    5136    5137    5169    5170    5230    5231    5307    5323    5426    5427
                    5434    5450    5572    5607    5637    5638    5653    5733    5755    5982    5987    6056
                    6072    6073    6079    6118    6120    6127    6133    6142    6161    6213    6216    6221
                    6246    6252    6323    6417    6419    6489    6493    6547    6589    6593    6597    6601
                    6605    6609    6625    6723    6735    6806    6815    6816    6817    6821    6822    6823
                    6824    6829    6830    6831    6834    6835    6836    6837    6838    6839    6840    6841
                    6856    6896    6899    6920    6925    6935    6940    6947    6952    6971    6972    6980
                    6987    7015    7037    7048    7132    7134    7136    7138    7270    7472    7517    7591
                    7593    7595    7597    7599    7601    7603    7605    7982    7985    8036    8040    8043
                    8048    8181    8200    8228    8232    8253    8257    8260    8271    8317    8416    8421
                    8426    8432    8446    8509
(U) HALT            1189 #
        BW14        1199 #  2424
        CSL         1193 #  2155    2173
        HALT        1192 #  3487
        ILLII       1196 #  7266
        ILLINT      1197 #  7298
        IOPF        1195 #  8466
        MULERR      1201 #  2098
        NICOND 5    1200 #
        POWER       1191 #  2126
(U) HOLD USER       1037 #  2124    2485    3311    3327    3394    3408    3465    3469    3505    3527    3597
                    3609    3665    3695    3699    3714    3720    3905    3957    3971    3988    3998    4001
                    4031    4032    4056    4106    4118    4128    4131    4238    4273    4304    4390    4539
```

Cross Reference Listing

	4541	4606	4623	4645	4897	5167	5179	5244	5532	6204	6205	6804
(D) I	1254 #	2455	2460	2465	2470	2508	2513	2518	2523	2528	2533	2538
	2543	2550	2555	2560	2565	2570	2575	2580	2585	2704	2705	2706
	2707	2715	2725	2735	2740	2750	2757	2767	2777	2787	2796	2797
	2798	2799	2807	2818	2828	2838	2847	2848	2849	2850	2862	2863
	3082	3083	3084	3085	3086	3087	3088	3089	3091	3092	3100	3101
	3102	3103	3104	3105	3106	3107	3118	3119	3120	3121	3122	3123
	3124	3125	3135	3136	3137	3138	3139	3140	3141	3142	3257	3258
	3259	3260	3261	3262	3263	3264	3369	3370	3371	3372	3373	3374
	3375	3376	3383	3384	3385	3386	3387	3388	3389	3390	3397	3398
	3399	3400	3401	3402	3403	3404	3411	3412	3423	3425	3576	3577
	3578	3579	3673	3706	3707	3708	3709	3759	3760	3761	3762	3763
	3764	3765	3766	3770	3771	3772	3773	3774	3775	3776	3777	3778
	3779	3780	3781	3782	3783	3784	3785	3786	3787	3788	3789	3790
	3791	3792	3793	3794	3795	3796	3797	3798	3799	3800	3801	3805
	3806	3807	3808	3812	3813	3814	3815	3816	3817	3818	3819	3820
	3821	3822	3950	3964	4012	4037	4204	4209	4990	5202	5666	5667
	5668	5669	5670	5671	5672	5674	5675	5676	5677	5679	5680	5681
	5682	5684	5685	5686	5687	5688	5689	5690	5691	5723	7534	7535
	7536	7538	7539	7541	7542	7544	7545	7546	7547	7548	7549	7550
	7551	7553	7554	7555	7556	7557	7558	7559	7560	7562	7563	7564
	7565	7566	7567	7568	7569	7571	7572	7573	7574	7575	7576	7577
	7578	7580	7581	7582	7583	7584	7585	7586	7587			
(U) I.CO3	1132 #											
(U) I.CO4	1133 #											
(U) I.CO5	1134 #											
(U) I.CO6	1135 #											
(U) I.CO7	1136 #											
(U) IO BYTE	1117 #	7377	7427	7433								
(U) IO CYCLE	1111 #	7237	7284	7376	7379	7415	7426	7432	7653			
(U) J	473 #	2228	2261	2265	2274	2279	2287	2308	2314	2338	2343	2364
	2365	3466	3470	3923	3926	4129	4172	4180	4190	4198	4392	4414
	4415	4417	4461	4501	4528	4630	4632	4636	4671	4704	4723	4791
	5343	5345	5346	5435	5630	5637	5638	5653	5751	5754	6040	6050
	6055	6058	6096	6097	6247	6253	6424	6447	6512	6543	6555	6586
	6602	6619	6628	6630	6656	6691	6724	6730	6735	6775	6996	7062
	7086	7161	7205	7313	7316	7388	7395	7401	7458	7466	7470	7473
	7499	7511	7513	7515	7517	7519	7521	7523	7525	7527	7529	7647
	7715	7719	7753	8294	8305	8334	8337	8338	8350	8359	8362	8363
	8462	8539										
ABORT	8462 #	3853	8443									
ACBSET	6930 #	6922										
AC_ARX	7529 #	3739	6761									
ADD	3956 #	3949	3950	3951	3952							
ADDCRY	6055 #	6030	6036	6049								
ADJBP	4836 #	4590										
ADJBPO	4839 #	4841										
ADJBP1	4858 #	4860										
ADJBP2	4868 #	4865										
ADJBP3	4911 #	4896										
ADJBP4	4923 #	4935										
ADJBP5	4937 #	4931										
ADJBP6	4958 #	4960										
ADJSP	3677 #	3673										
ADJSP1	3694 #	3686										

```
ADJSP2          3699 #  3691
AND             2721 #  2714   2715   2716   2717   2746
ANDCA           2731 #  2724   2725   2726   2727   2783
ANDCB           2783 #  2776   2777   2778   2779
ANDCM           2746 #  2739   2740   2741   2742
AOBJ            3416 #  3411   3412
AOJ             3394 #  3383   3384   3385   3386   3387   3388   3389   3390
AOS             3311 #  3300   3301   3302   3303   3304   3305   3306   3307
APRID           6851 #
APRSO           6843 #  6819
APRSZ           6846 #  6818
ARSIGN          2364 #  2306
ASH             2888 #  2859
ASHC            2991 #  2863
ASHC1           2997 #  2995
ASHCL           3013 #  3002   3015
ASHCQ1          3024 #  2985   3021
ASHCR           3006 #  3008
ASHCX           3018 #  3010
ASHL            2898 #  2898
ASHLO           2895 #  2888
ASHR            2891 #  2877
ASHX            2902 #  2915
ASHXX           2915 #  2920
BACKBP          6774 #  6742   6750
BACKD           6749 #  8492   8500   8502
BACKS           6741 #  8499
BADDATA         8026 #  8011
BDABT           6218 #  6154
BDCFLG          6220 #  6199
BDEC            6102 #
BDECO           6108 #  6106
BDEC1           6115 #  6124
BDEC2           6120 #  6128
BDEC3           6130 #  6116   6138
BDEC4           6135 #  6130
BDEC5           6147 #  6133   6143
BDECLP          6185 #  6204
BDFILL          6169 #  6178
BDSET           6193 #  6213   6216
BDSUB           6232 #  6136   6186
BDSUB1          6236 #  6233
BDSUB2          6252 #  6255
BDTBL           6209 #  6192
BITCHK          6723 #  5729   5772   5910   5912   6269
BIXUB           7354 #  7347   7348   7349   7350
BIXUB1          7363 #  7359   7361
BLT             4994 #  4990
BLT-CLEANUP     5067 #  8484
BLTCLR          5026 #  5041
BLTGOT          5054 #  5029
BLTGO           5052 #  5020   5046
BLTLP           5049 #  5064
BLTLP1          5017 #  5051
BOTH            2236 #
```

```
BWRITE      2375 #  2476  2482  2596  2598  2631  2637  2645  2647  2666  2668  2711
            2721    2731  2763  2773  2793  2803  2814  2824  2854  3957  3971  4023
            4029    4052  4054  8264  8526
BYTEAS      4643 #  4667  6666
BYTEA       4645 #  4600  4619
BYTEAO      4648 #  4644
BYTFET      4669 #  4651  4655  6708  6709
BYTIND      4663 #  4659
CAIM        3277 #  3258  3259  3260  3261  3262  3263  3264  3266  3267  3268  3269
            3270    3271  3272  3273
CHKSN       5630 #  5611
CLARXL      7517 #  4999  7291
CLARXO      6246 #  4539  4541  4545  4546  5977  6111  6237
CLDISP      8477 #  6743  6751  6764  6770  8472
CLEANED     8480 #  5072  8474
CLEANUP     8479 #  8477
CLRB1       6097 #  6095
CLRBIN      6095 #  5980  6076
CLRFLG      6733 #  5776  6148
CLRPTL      7182 #  7185
CLRPT       7173 #
CLRSN       5638 #  5416  5506
CMPDST      5961 #  5941
CMS         5909 #
CMS2        5950 #  5920
CMS3        5926 #  5952  5956
CMS4        5929 #
CMS5        5955 #  5934
CMS6        5941 #  5958
CMS7        5946 #  5943
CMS8        5957 #  5955
CMSDST      6745 #  8498
COMO        4528 #  4523  4524
COMOA       4523 #  4527
COM1        4527 #  4518
COM1A       4518 #  4526
COM2        4526 #  4513
COM2A       4513 #
CONSO       6844 #  6826
CONSZ       6847 #  6825
CONT        7650 #  7645
CONT1       7657 #  3944
CPYSGN      4003 #  3989  3999
DAC         2396 #  2403  2673  6218
DADD        3983 #  3978
DADD1       3986 #  3992
DBABT       6061 #  5997  6001
DBDN1       6074 #  6072
DBDONE      6069 #  6063  6088
DBFAST      6014 #  6010
DBIN        5970 #
DBIN1       5985 #  5988
DBIN2       5989 #  5985
DBINLP      5993 #  6011  6021
DBLDBL      6045 #  6033  6044
```

DBLDIV	4479 #	4333	4343	5534	5538	
DBLMUL	4123 #	4094				
DBLNEG	4535 #	2688				
DBLNGA	4536 #	2678				
DBLNG1	4544 #	4258	6120			
DBNEG	6079 #	6068				
DBSLOW	6026 #	2089	6008			
DBSLO	6007 #	6015				
DBXIT	6067 #	5995				
DDIV	4285 #	4280				
DDIV1	4302 #	4289	4293			
DDIV2	4305 #	4303				
DDIV3A	4310 #	4315				
DDIV3	4306 #	4300				
DDIV4	4318 #	4311				
DDIV5A	4332 #	4323				
DDIV5B	4356 #	4347				
DDIV5C	4358 #	4353				
DDIV5	4326 #	4322				
DDIV6	4361 #	4354	4356			
DDIV7	4363 #					
DDIV8A	4372 #					
DDIV8	4370 #	4363				
DDIV9	4378 #	4373				
DDIVS	4392 #	4319	4337	5530		
DFAD	5364 #	5351				
DFADJ	5415 #	5380	5388			
DFADJ1	5422 #	5423				
DFADJ2	5424 #					
DFADJ3	5430 #	5426	5430			
DFADJ4	5431 #	5427				
DFADJ5	5433 #	5425	5434			
DFADJ6	5435 #	5433				
DFAS1	5373 #	5369	5371			
DFAS2	5380 #	5374				
DFAS3	5385 #	5376				
DFAS5	5394 #	5382	5390			
DFAS6	5399 #	5400				
DFAS7	5401 #	5399				
DFDV	5504 #	5501				
DFDV1	5510 #	5517				
DFDV2	5518 #	5511				
DFDV3	5529 #	5521				
DFDV4A	5544 #	5547				
DFDV4B	5549 #	5546				
DFDV4	5531 #	5527				
DFMP	5445 #	5440				
DFMP1	5447 #	5448				
DFMP2	5473 #	5470				
DFPR1	2329 #	2323				
DFPR2	2330 #	2328				
DFSB	5361 #	5352				
DIV	4224 #	4208	4209	4210	4211	
DIV1	4233 #	4219	4221	4275		
DIV2	4236 #	4249				

```
DIVA            4245 #  4231
DIVB            4251 #  4246
DIVC            4264 #  4256    4261
DIVHI           4486 #  4480    4483    4497
DIVIDE          4451 #  4451
DIVSET          4445 #  4441
DIVSGN          4431 #  4413    5185
DIVSUB          4410 #  4237    4870    4880    4914
DMLINT          4131 #  4096
DMOVNM          2688 #  2683
DMOVN           2678 #  2674
DMOVN1          2690 #  2682
DMTRAP          4115 #  4109
DMUL            4067 #  4062
DMUL1           4088 #  4077
DMUL2           4093 #  4087
DMULGO          4122 #  4075
DNEG            5587 #  5561    5564    5567    5570
DNEG1           5589 #  5587
DNEG2           5591 #  5588
DNN1            5617 #  5613
DNN2            5619 #  5615
DNNORM          5597 #  5550    5592    5594    5599    5627
DNNRM1          5624 #  5607
DNORM           5558 #  5496    5549    5560    5582
DNORMO          5495 #  5405
DNORM1          5579 #  5572
DNORM2          5621 #  5619
DOCVT           6107 #  6157    6173
DOCVT1          6182 #  6107
DOCVT2          6203 #  6183
DOCVT3          6200 #  6222
DONE            2135 #  2435    2449    2749    2750    2929    3082    3083    3091    3092    3192    3236
                3239    3242    3257    3369    3620    4118    4623    4965    5034    5058    6847    6883
                6937    7020    7024    7028    7032    7036    7083    7107    7117    7168    7202    7341
DPB             4612 #  4585
DPB1            4783 #  4622    6690
DPB7            4790 #  4784    4785    4786    4787    4788    4812
DPBSLO          4794 #  4783
DRND1           5653 #  5649    5657    5660
DROUND          5647 #  5563    5566    5569    5601    5603    5605    5651
DSMS1           7315 #  7312    7317
DSTEA           6708 #  6714
DSTIND          6713 #  6710    6711
DSUB            3995 #  3979
DUMP            7722 #  7706    7713
DVSUB1          4437 #  4434
DVSUB2          4438 #  4432
DVSUB3          4440 #  4444
EACALC          2190 #  2253
EAPF            8532 #  8517
EAPF1           8535 #  8521
EDBYTE          6445 #  6408    6436
EDEXMD          6356 #  6327
EDFILL          6372 #
```

```
EDFIL1          6387 #  6374
EDFLT           6403 #  6349    6393
EDFLT1          6421 #  6417    6419
EDISP           6286 #  6280
EDISP1          6288 #  6288
EDIT            6265 #
EDITLP          6270 #  6494
EDMSG           6430 #  6295
EDMSG1          6439 #  6432
EDN1A           6476 #  6481
EDNOP           6465 #  6297    6299    6301    6315    6324    6350    6361    6382    6388    6441    6442
                6456
EDNOP1          6466 #  6460
EDNOP2          6483 #  6477    6483
EDOPR           6315 #  6292
EDSEL           6366 #  6319
EDSFLT          6392 #  6376
EDSKP           6453 #  6304    6307    6309
EDSKP1          6457 #  6454
EDSPUT          6379 #  6387    6397
EDSSIG          6349 #  6321
EDSTOP          6334 #  6317    6377
EDSTP1          6340 #  6343
ENDSKP          6077 #  5901
EQV             2793 #  2786    2787    2788    2789
EXCH            2495 #  2491
EXTDSP          5741 #
EXTEA           5740 #
EXTEAO          5737 #  5735
EXTEA1          5738 #  5746
EXTEND          5727 #  5723
EXTEXT          5750 #  5742    5756
EXTIND          5746 #  5743
FAD             5100 #  5077    5078    5079    5080    5081    5082    5083
FAS1            5102 #
FAS2            5111 #  5102
FAS3            5114 #  5107    5108    5111    5112
FAS4            5116 #  5116
FDV             5163 #  5151    5152    5153    5155    5156    5157    5158
FDVO            5169 #  5166
FDV1            5171 #  5169
FDV2            5172 #  5170
FDV3            5173 #  5171    5172
FDV4            5175 #  5176
FDV5            5178 #  5175
FDV6            5181 #  5178
FDV7            5185 #  5186
FDV8            5188 #
FDV9            5196 #  5188    5193    5194
FETIND          2246 #  2211    2216    2238
FIX             5241 #  5236    5237
FIX++           4455 #  4456
FIX1++          4457 #  4455
FIXL            5256 #  5246    5256
FIXPC           8475 #  4131    6234
```

Cross Reference Listing

FIXR	5251 #	5252							
FIXT	5263 #	5260							
FIXX	5260 #	5253							
FIXX1	5261 #	5267							
FL-BWRITE	2422 #	5261	5265	5311	5333				
FLEX	5311 #	5307							
FLTR	5206 #	5201							
FLTR1A	5214 #	5211							
FLTR1	5210 #	5207							
FLTR2	5219 #	5212	5214						
FLTR3	5221 #	5222							
FMP	5133 #	5122	5123	5124	5126	5127	5128	5129	
FMP1	5138 #	5136	5137						
FP-LONG	3838 #	3816	3817	3818	3819	3820	3821		
FPRO	2303 #	2297							
FPR1	2307 #								
FSB	5097 #	5085	5086	5087	5088	5089	5090	5091	
FSC	5227 #	5202							
GETPCW	6991 #	3877							
GETSRC	6657 #	5932	6367						
GOEXEC	3901 #	8537							
GRP7OO	6850 #	6811							
GRP7O1	6829 #	6812							
GRP7O2	7005 #	7001							
GSRC	6655 #	5889	6542						
GSRC1	6664 #	6660							
GTFILL	6727 #	5769	6266						
GTPCW1	6994 #	6985	6994						
H1	7641 #	7707							
HALT	3485 #	3437							
HALTED	7631 #	2098	2155	2173	2424	3487	7266	7298	8466
HALTLP	7647 #	7641							
HARD	8441 #	7505	8013	8015	8029				
HLL	2598 #	2507	2508	2551	2623				
HLLE	2664 #	2537	2538	2539	2540				
HLLO	2668 #	2527	2528	2529	2530				
HLLZ	2666 #	2517	2518	2519	2520				
HLR	2625 #	2554	2555						
HLRE	2657 #	2584	2585	2586	2587				
HLRM	2634 #	2556							
HLRO	2661 #	2574	2575	2576	2577				
HLRS	2637 #	2557							
HLRZ	2659 #	2564	2565	2566	2567				
HRL	2623 #	2512	2513						
HRLE	2650 #	2542	2543	2544	2545				
HRLM	2628 #	2514							
HRLO	2654 #	2532	2533	2534	2535				
HRLS	2631 #	2515							
HRLZ	2652 #	2522	2523	2524	2525				
HRR	2596 #	2509	2549	2550	2625				
HRRE	2643 #	2579	2580	2581	2582				
HRRO	2647 #	2569	2570	2571	2572				
HRRZ	2645 #	2559	2560	2561	2562				
HSBDON	7749 #								
IBP	4588 #	4581							

IBPS	4629 #	4591	4596	4610					
IBPX	4636 #	3736	4629	8132	8236				
IDIV	4215 #	4203	4204	4205	4206				
IDPB	4610 #	4584							
IDST	6701 #	5962	6685						
IDSTX	6705 #	6702							
ILDB	4596 #	4582							
IMUL	4017 #	4011	4012	4013	4014				
IMUL1	4023 #	4240							
IMUL2	4025 #	4022							
IMUL3	4031 #	4028							
INCAR	7519 #	5863	7271						
INCPC	2183 #	2178							
INDEX	2223 #								
INDRCT	2241 #								
IOEA	7442 #	7374	7412						
IOEA1	7448 #	7450							
IOEA2	7452 #	7448							
IOEAI	7460 #	7453							
IOEAX	7469 #	7456							
IOR	2773 #	2766	2767	2768	2769	2834			
IORD	7373 #	7329	7340	7355					
IORD1	7382 #	7378							
IORD2	7392 #	7387							
IORD3	7398 #	7393	7399						
IOT700	7591 #	7534	7535	7536					
IOT710	7593 #	7538	7539						
IOT720	7595 #	7541	7542						
IOT730	7597 #	7544	7545	7546	7547	7548	7549	7550	7551
IOT740	7599 #	7553	7554	7555	7556	7557	7558	7559	7560
IOT750	7601 #	7562	7563	7564	7565	7566	7567	7568	7569
IOT760	7603 #	7571	7572	7573	7574	7575	7576	7577	7578
IOT770	7605 #	7580	7581	7582	7583	7584	7585	7586	7587
IOW1	7482 #	7487							
IOW2	7494 #	7484	7490						
IOW3	7497 #	7504							
IOW4	7502 #	7498							
IOW5	7505 #	7492							
IOWAIT	7478 #	7383	7419						
IOWR	7411 #	7344	7364						
IOWR1	7417 #	7427	7433						
IOWR2	7424 #	7414							
IOWR3	7430 #	7425	7431						
ITRAP	7997 #	5865							
JEN	3493 #	3447							
JEN1	7312 #	3499	3514						
JEN2	3499 #	3491							
JFCL	3505 #	3425							
JFFO	2923 #	2862							
JFFO1	2931 #	2928							
JFFOL	2945 #	2947							
JMPA	3348 #	3753							
JRA	3747 #	3709							
JRST	3432 #	3373	3423						
JRSTO	3461 #	3458	3482	3497					

JRST1	3479 #	3474	3478									
JRST10	3490 #	3444										
JRSTF	3455 #	3435										
JSA	3732 #	3708										
JSP	3713 #	3707										
JSR	3719 #	3706										
JSTAC	3614 #	3604	3611	3649								
JSTAC1	3617 #	3668										
JSYS	3832 #	3813										
JUMP	3380 #	3370	3371	3372	3374	3375	3376					
JUMP-TABLE	3333 #	3380	3394	3408	3418							
JUMP-	3360 #	3339	3342	3345	3508							
JUMPA	3364 #	3348	3351	3354	3357	3716	3907					
KIEPT	8389 #	8383	8386									
KIF10	8380 #	8373										
KIF30	8397 #	8378	8393									
KIF40	8403 #	8407										
KIF50	8410 #	8405	8433	8448								
KIF80	8431 #											
KIF90	8436 #	8427										
KIFILL	8368 #	8063										
KIMUUO	3913 #	3859										
KIUPT	8374 #	8396										
L-BDEC	5701 #	5676	5677									
L-CMS	5695 #	5666	5667	5668	5670	5671	5672					
L-DBIN	5699 #	5674	5675									
L-EDIT	5697 #	5669										
L-MVS	5703 #	5679	5680	5681	5682							
L-SPARE-A	5707 #	5685										
L-SPARE-B	5709 #	5686										
L-SPARE-C	5711 #	5687	5688	5689	5690	5691						
L-XBLT	5705 #	5684										
LDB	4598 #	4583										
LDB1	4685 #	4605	5965	6670								
LDB7	4699 #	4690	4691	4692	4693	4694						
LDBSH	4714 #	4709										
LDBSWP	4708 #	4687										
LDPI2	7158 #	3500	7147									
LOADAR	7510 #	6211	6274	6357	7444							
LOADARX	7513 #	6566	7078									
LOADPI	7157 #	2101										
LOADQ	7515 #	5010	5918									
LSH	2872 #	2861										
LSHC	2974 #	2865										
LSHCL	2982 #	2974	2982									
LSHCR	2978 #	2978										
LSHCX	2984 #	2979	3051	3056								
LSHL	2879 #	2874										
LUUO	3930 #	3759	3760	3761	3762	3763	3764	3765	3766			
LUUO1	3933 #	5695	5697	5699	5701	5703	5705	5707	5709	5711		
MAP	7980 #	7976										
MAPDON	8523 #	8486										
MOVE	2482 #	2456	2479	2486	2510	2552	2734	2735	2736	2737	2751	2752
	4004	4005	4031	4032	4056	4866						
MOVELP	5807 #	5787	5816									

```
MOVF1                6507 #  6513
MOVFIL               6513 #  5839    5900
MOVLPO               5816 #  5801
MOVM                 2479 #  2469    2471    2472
MOVN                 2484 #  2464    2465    2466    2467
MOVPAT               6282 #  6277    6279    6282
MOVRJ                5837 #  5793
MOVS                 2476 #  2459    2460    2461    2462    2629    2635    2652    2654    2659    2661
MOVSTX               5890 #
MOVSTO               5885 #  5790
MOVST1               5886 #  5841    5896
MOVST2               5899 #  5810    5891
MOVST3               5903 #  5899
MOVST4               5895 #  5852
MSKPAT               6283 #  6278
MUL                  4044 #  4036    4037    4038    4039
MUL+                 4166 #  4150    4162    4168    4186
MUL-                 4184 #  4176    4194
MULBY4               6044 #  6027
MULSB1               4148 #  5456
MULSUB               4147 #  4020    4048    4941    5140
MULTIPLY             4159 #  4127    5461    5473    5480
MUUO                 3845 #  3770    3771    3772    3773    3774    3775    3776    3777    3778    3779    3780
                     3781    3782    3783    3784    3785    3786    3787    3788    3789    3790    3791    3792
                     3793    3794    3795    3796    3797    3798    3799    3800    3801
MVABT                5821 #  5812
MVABT1               5824 #  5826
MVABT2               5828 #  5824
MVEND                5831 #
MVS                  5766 #
MVS1                 5783 #  5781
MVSK2                5871 #  5845
MVSK3                5861 #  5857
MVSKP                5844 #  5837    5855    5868    5879
MVSKP1               5855 #  5850
MVSKP2               5867 #  5859
MVSO                 5795 #  5788
MVSO1                5798 #  5798
NEXT                 3925 #  3868    3872    3914    7723    7726    7728    7730    7733    7736    7739    7742
                     7745    7747
NEXTAR               8539 #  8514    8533
NICOND               2144 #  2429    2950    2996    3024    3336    3618    3694    3701    4381    4384    5257
                     5263    5622    7990
NICOND-FETCH         2163 #  2135    2136    2138    2139    2199    2205    3233    3245    3360    3361    3364
                     3365    3432    3433    3459    3501    4117    4592    7149    7420
NIDISP               2996 #  3696    3700    4238    4273    4304    4390    4607    4897    5167    5179    5244
                     5532
NODDIV               4389 #  4326    4328
NODIV                4273 #  4267    4271
NOMOD                2227 #
NXTWRD               4635 #  4631
ORCA                 2813 #  2806    2807    2808    2809
ORCB                 2844 #  2837    2838    2839    2840
ORCM                 2834 #  2827    2828    2829    2830
PAGE-FAIL            7995 #
```

PF100	8218 #	8204									
PF105	8226 #	8222									
PF107	8208 #	8229	8233								
PF110	8235 #	8209									
PF120	8266 #	8246									
PF125	8285 #	8283	8459								
PF130	8248 #	8244	8250								
PF25	8052 #	8054									
PF30	8068 #	8061									
PF35	8081 #	8070									
PF40	8086 #	8078									
PF45	8093 #	8113	8182								
PF50	8124 #	8097									
PF60	8109 #	8111									
PF70	8129 #										
PF75	8135 #	8137									
PF76	8157 #	8153	8155								
PF77	8162 #	8075	8083								
PF80	8192 #	8167									
PF90	8199 #										
PFD	8003 #	8005									
PFDBIN	6768 #	8496									
PFDONE	8240 #	8434	8438								
PFGACO	6770 #										
PFMAP	8033 #	7989	8017	8021	8023						
PFPI1	8450 #	8007	8019								
PFPI2	8458 #										
PFT	8466 #	8105	8124	8183	8192	8207	8306	8323	8328	8351	8410
PFT1	8467 #	8455									
PFT10	8510 #	8508									
PFT1A	8474 #	8470									
PFT2	8506 #	8482									
PFT3	8509 #	8527									
PFTICK	8212 #	8175									
PI	7215 #	7217	8509								
PI10	7233 #	7225	7226	7227	7228	7229	7230	7231			
PI40	7251 #	7248									
PI50	7252 #	7302									
PIEXIT	7147 #	3529									
PIJSR	7266 #	7255									
PIP1	7225 #	7218									
PIP2	7226 #	7219									
PIP3	7227 #	7220									
PIP4	7228 #	7221									
PIP5	7229 #	7222									
PIP6	7230 #	7223									
PIP7	7231 #	7224									
PISET	3528 #	7273									
PIXPCW	3537 #	7258									
POP	3624 #	3578									
POPJ	3657 #	3579									
POPX1	3652 #	3640									
PTRIMM	8165 #										
PTRIND	8172 #	8215									
PTRSHR	8186 #	8169									

```
PUSH                    3584 #  3577
PUSH1                   3586 #  3598
PUSHJ                   3596 #  3576
PUTDST                  6683 #  5814    5893    6175    6194    6380    6415    6440    6508
PWRON                   7636 #  2127
QDNEG                   4509 #  4313    4389
QMULT                   4126 #  4084    4104
RDAPR                   6894 #
RDCSB                   7008 #
RDCSTM                  7012 #
RDEBR                   6975 #
RDEBR1                  6977 #  6977
RDERA                   6821 #
RDHSB                   7014 #
RDINT                   7120 #
RDIO                    7340 #  7333    7335
RDPI                    7126 #
RDPT                    8344 #  8101    8112    8118    8186    8301
RDPUR                   7010 #
RDSPB                   7006 #
RDTIME                  7089 #  7103
RDTIM1                  7105 #  7102
RDUBR                   6984 #
ROT                     2907 #  2860
ROTC                    3046 #  2864
ROTCL                   3054 #  3046    3054
ROTCR                   3050 #  3050
ROTL                    2917 #  2909
RTNREG                  7167 #  6857    6907    6981    6988    7006    7008    7010    7012    7014    7121    7126
SAVVMA                  7717 #  7632    7704
SBRL                    7521 #  5182    6019
SECIMM                  8095 #
SECSHR                  8118 #  8099
SETCA                   2803 #  2796    2797    2798    2799
SETCM                   2824 #  2817    2818    2819    2820    2844
SETO                    2854 #  2847    2848    2849    2850
SETPDL                  3609 #  3654
SETPTR                  8314 #  8094    8164
SETSN                   5637 #  5190    5192    5487
SETUBR                  6938 #  6932
SETZ                    2711 #  2704    2705    2706    2707
SFM                     3547 #  3449
SHDREM                  8359 #  8148    8151
SHIFT                   2283 #  2293
SKIP                    2138 #  3248    3251    3254    6077    6205    6844
SKIP-COMP-TABLE         3230 #  3277    3296    3297    5928
SKIPE                   3239 #  5904
SKIPS                   3294 #  3283    3284    3285    3286    3287    3288    3289    3290    3313
SNNEG                   5314 #  5299    5301    5303    5305
SNNORM                  5318 #  5314    5316    5318    5322
SNNOT                   5329 #  5325
SNNOT1                  5332 #  5327
SNNOT2                  5333 #  5329    5331    5332
SNORM                   5298 #  5117    5146    5223    5298    5310
SNORMO                  5223 #  5196    5208    5215    5230    5231
```

Cross Reference Listing

SNORM1	5310 #	5306										
SOJ	3408 #	3397	3398	3399	3400	3401	3402	3403	3404			
SOS	3327 #	3316	3317	3318	3319	3320	3321	3322	3323			
SRCMOD	6540 #	5808	5993	6635	6639	6641						
SRND1	5343 #	5340										
SROUND	5340 #	5300	5302	5304	5319	5320	5321	5341				
SSWEEP	7204 #	7177	7190									
STAC	2428 #	2397	2441	2446	2454	2455	2470	2499	2679	2880	2893	2902
	5576	5583	5628	5832	6340	7620						
STAC34	6087 #	6082										
START	2157 #	3728	3743									
STBOTH	2439 #											
STBTH1	2444 #	2392										
STDBTH	2401 #											
STMAC	3601 #	3593										
STMEM	2433 #	7627										
STOBR	7525 #	3538	3880	7269								
STOPC	7527 #	3541	7640									
STORE	2447 #	2699	3549									
STPF1A	6766 #	6758										
STPTR1	8325 #	8322										
STPTR2	8331 #	8327										
STPTR3	8337 #	8333										
STRPF	6754 #	8488	8490									
STRPFO	6755 #	6772										
STRPF1	6758 #											
STRPF2	6769 #	6766										
STRPF3	6763 #											
STRPF4	6772 #	8494										
STRTIO	7523 #	7239	7287	7656								
STSELF	2391 #	2457										
STUBRS	6942 #	6944										
SUB	3970 #	3963	3964	3965	3966							
SWEEPL	7195 #	7198										
SWEEP	7189 #	6954	6971	6972								
TDONE	3207 #	3198	3201									
TDX	3178 #	3086	3095	3100	3104	3109	3113	3118	3122	3127	3131	3135
	3139	3144	3148									
TDXX	3183 #	3084	3088	3093	3097	3102	3106	3111	3115	3120	3124	3129
	3133	3137	3141	3146	3150							
TENLP	2087 #	2094										
TEST-TABLE	3189 #	3178	3184	7330								
TICK	7043 #	5874										
TIOX	7329 #	7322	7323	7324	7325							
TOCK	7046 #	7100	8213	8454								
TOCK1	7050 #	7056										
TOCK2	7057 #	7051										
TOCK3	7061 #	7068										
TRAP	6779 #	2148	2151	2154	2166	2169	2172					
TRNAR	6561 #	6370										
TRNFNC	6583 #	6571	6590	6594	6598	6610						
TRNNS1	6633 #	6623										
TRNNS2	6636 #	6633										
TRNRET	6615 #	6584										
TRNSIG	6596 #	6606										

Cross Reference Listing


```
TRNSS           6622 #  6617
TRNSS1          6618 #  6626
TRP1            6798 #  6788    6796
TSX             3176 #  3087    3096    3101    3105    3110    3114    3119    3123    3128    3132    3136
                3140    3145    3149
TSXX            3181 #  3085    3089    3094    3098    3103    3107    3112    3116    3121    3125    3130
                3134    3138    3142    3147    3151
TXXX            3192 #  3207
TXZX            3206 #  3195
UMOVEM          7623 #  7611
UMOVE           7615 #  7610
UPCST           8300 #  8128    8195
UUO             3844 #  3805    3812
UUO1O1          3826 #  3806
UUO1O2          3828 #  3807
UUO1O3          3830 #  3808
UUO1O6          3834 #  3814
UUO1O7          3836 #  3815
UUO247          3840 #  3822
UUOFLG          3921 #  3863    3916
UUOGO           3850 #  2342    3436    3445    3448    3450    3451    3452    3485    3490    3493    3513
                3534    3547    3826    3828    3830    3832    3834    3836    3838    3840    6723    6806
                6815    6816    6817    6821    6822    6823    6824    6829    6830    6831    6834    6835
                6836    6837    6838    6839    6840    6841    7015    7037    7132    7591    7593    7595
                7597    7599    7601    7603    7605
UUOPCW          3874 #  3919
VECINT          7280 #  7243    7283
VECIN1          7299 #  7297
WRAPR           6860 #
WRAPR1          6879 #  6891
WRAPR2          6885 #  6878
WRCSB           7022 #
WRCSTM          7030 #
WREBR           6957 #
WREBR1          6959 #  6959
WRHSB           7034 #
WRINT           7113 #
WRIO            7344 #  7334    7336
WRPI            7129 #
WRPUR           7026 #
WRSPB           7018 #
WRTHSB          7710 #  7634
WRTIME          7073 #
WRTIM1          7085 #  7054    7081
WRUBR           6912 #
XCT             3558 #  3554
XCT1            3570 #  3564    6805    7660
XCT1A           3560 #  3568
XCT2            2250 #  3571
XCTGO           2175 #  2159    7652
XJEN            3513 #  3441
XJRSTF          3520 #  3440    3543
XJRSTFO         3439 #  3517
XLATE           6560 #  6546
XLATE1          6569 #  6573
```

Cross Reference Listing


```
        XOR                     2763 #  2756    2757    2758    2759
        XOS                     3312 #  3327
        XPCW                    3534 #  3442
        ZAPPTA                  7200 #  7186
(D) J                           1251 #
(U) JFCLFLG                     1057 #  3505
(U) LD FLAGS                    1061 #  2125    3464    3468    3526    3904    7257    7272    8472
(U) LD PCU                      1049 #  3906
(U) LDVMA                       1100 #  2068    2087    2091    2135    2135    2136    2138    2139    2139    2157    2183
                                2198    2204    2210    2214    2237    2241    2250    2270    2312    2322    2325    2337
                                2691    2695    2696    2927    3233    3233    3245    3360    3360    3361    3364    3365
                                3365    3432    3433    3434    3439    3443    3446    3459    3472    3475    3501    3516
                                3522    3537    3539    3539    3548    3588    3615    3625    3637    3642    3658    3721
                                3749    3857    3858    3858    3865    3866    3866    3879    3879    3898    3899    3925
                                3925    3934    3941    3942    4117    4117    4592    4592    4601    4648    4652    4656
                                4660    4996    5017    5023    5037    5038    5042    5043    5060    5061    5741    5743
                                5744    5767    5915    6164    6191    6236    6242    6249    6265    6270    6326    6404
                                6446    6561    6708    6709    6710    6711    6787    6795    6912    7076    7101    7106
                                7106    7149    7149    7167    7173    7236    7251    7251    7251    7252    7293    7293
                                7301    7382    7417    7420    7420    7442    7464    7523    7615    7623    7633    7636
                                7636    7639    7639    7650    7706    7722    7722    7725    7725    7732    7732    7735
                                7735    7738    7738    7741    7741    7744    7744    7751    8033    8073    8074    8074
                                8081    8082    8082    8086    8088    8157    8159    8272    8288    8344    8346    8375
                                8376    8390    8392    8456    8462    8510    8511    8511    8536    8536    8539    8539
(U) LOADFE                      904 #   2286    2304    2319    2873    2876    2880    2890    2892    2908    2911    2914
                                2919    2931    2946    2992    4599    4603    4615    4708    4713    4715    4717    4720
                                4783    4796    4799    4800    4802    4804    5106    5134    5143    5145    5165    5171
                                5172    5186    5208    5212    5214    5215    5229    5298    5302    5304    5310    5318
                                5320    5321    5322    5341    5346    5387    5402    5404    5468    5485    5496    5519
                                5545    5559    5566    5569    5581    5598    5603    5605    5626    5651    5659    5795
                                5796    5798    5964    6335    6336    6342    6466    6467    6468    6469    6666    6669
                                6688    6706
(U) LOADSC                      902 #   2082    2092    2285    2304    2319    2335    2896    2898    2974    2975    2976
                                2978    2982    2992    3001    3004    3008    3015    3046    3047    3048    3050    3054
                                4018    4046    4083    4103    4123    4149    4161    4167    4175    4185    4193    4236
                                4333    4341    4448    4451    4491    4837    4840    4853    4859    4864    4877    4893
                                4938    4953    4959    5100    5104    5105    5114    5116    5138    5185    5219    5222
                                5227    5245    5246    5247    5249    5252    5256    5366    5376    5386    5420    5423
                                5430    5445    5448    5456    5460    5463    5473    5479    5534    5535    6109    6138
                                6277    6279    6282    6286    6288    6459    6465    6471    6476    6483    6939    6943
                                6957    6959    6975    6977    6985    6992    6994    7184    7197    7204    7392    7399
                                7424    7430    7478    7485    7494    7498    8050    8053    8093    8107    8110    8133
                                8136    8243    8249
(U) LSRC                        554 #
(U) MACRO%
        ABORT MEM CYCLE         1906 #  2068    3516    7236    7633    7706    8033    8456    8462
        AC                      1756 #  4017    4044    4215    4306    5163    5228    5369    5371
        AC[]                    1757 #  4382    5508    5516    6017    6081
        AC[]_Q                  1658 #  3024    4078    4086    4097    4109
        AC[]_Q.AND.[]           1647 #  4336    4380    4510    4515    4520    4526    4527
        AC[]_[]                 1634 #  2396    2679    2950    4353    4359    5583    5628    5824    5844    5861    5944
                                5978    6037    6039    6074    6087    6088    6124    6156    6162    6173    6178    6201
                                6202    6361    6664    6705    6743    6751    6758    6763
        AC[]_[] TEST            1635 #  6636
        AC[]_[] VIA AD          1632 #
```

Cross Reference Listing


```
AC[]_[]*2               1637 #  4095
AC[]_[]+1               1636 #  6510
AC[]_[]+Q               1642 #
AC[]_[]+[]              1644 #  4378    6020
AC[]_[]-[]              1643 #  5778
AC[]_[].AND.[]          1646 #  4356    5575    5621
AC[]_[].EQV.Q           1648 #  4110    4112    4114    4383
AC[]_-Q                 1654 #
AC[]_-[]                1649 #
AC[]_.NOT.[]            1652 #
AC[]_0                  1656 #  2080    2929    4354    5851    5899    6095    6097
AC[]_1                  1657 #  2081
AC_-[]                  1650 #  5620
AC_.NOT.[]              1653 #  5619
AC_Q                    1655 #  4352
AC_[]                   1638 #  2428    2985    3207    3296    3336    3348    3617    3619    3715    4528    4606
                        5009    5071    5257    5263    5901    5937    6200    6421    6494    6770    7341    7529
                        7990
AC_[] TEST              1639 #  3339    3342    3345    3351    3354    3357    4108    6075    6121
AC_[] VIA AD            1633 #
AC_[]+1                 1640 #  6746
AC_[]+Q                 1641 #
AC_[]+[]                1645 #  3018    3682    3687
AC_[].OR.[]             1651 #  4964    5864    5903
AD FLAGS                1934 #  2485    3311    3327    3394    3408    3957    3971    3988    3998    4001    4128
                        4539    4541
AD FLAGS EXIT           1982 #  3957    3971
AD PARITY               1860 #  2088    2264    2277    2278    2495    2596    2598    2629    2635    2688    2872
                        2875    2889    2923    3380    3624    3657    3679    3735    3747    4046    4088    4225
                        4616    4994    5100    5133    5165    5229    5375    5770    5773    5788    5791    5821
                        5837    5909    5911    5929    5961    5973    5976    6004    6007    6009    6016    6026
                        6067    6069    6070    6102    6107    6108    6109    6130    6135    6147    6182    6267
                        6325    6366    6403    6405    6509    6622    6629    6655    6657    6684    6741    6749
                        6755    7344    7626
ADD .25                 1891 #  2484    2854    3277    3327    3408    3434    3443    3446    3970    3987    3995
                        4000    4174    4178    4220    4264    4297    4320    4321    4327    4348    4374    4415
                        4416    4417    4433    4435    4442    4448    4479    4509    4514    4519    4524    4536
                        4540    4544    4546    4875    4932    4949    5002    5014    5030    5054    5097    5172
                        5175    5176    5183    5194    5207    5299    5301    5303    5305    5316    5332    5361
                        5400    5513    5524    5531    5547    5588    5593    5614    5620    5654    5778    5811
                        5825    5846    5913    5949    5950    5951    5998    6062    6080    6083    6151    6168
                        6176    6195    6203    6204    6238    6240    6243    6291    6633    6754    6762    6768
                        7058    7158    7182    7195    7442    8370    8380    8475    8527
ADL PARITY              1853 #
ADR PARITY              1857 #
AREAD                   1977 #  2228
ASH                     1868 #  2332    2892
ASH AROV                1919 #  2898    3014    3017
ASH36 LEFT              2885 #  2898
ASHC                    1873 #  2898    3007    3010    3014    3017    4167    4175    4185    4193    4299    4303
                        4372    4376    4392    5116    5219    5221    5251    5395    5397    5401    5403    5419
                        5423    5430    5433    5435    5447    5488    5492    5527    5541    5559    5565    5568
                        5581    5598    5602    5604    5626    5651
B DISP                  1978 #  2476    2482    2596    2598    2631    2637    2645    2647    2666    2668    2711
                        2721    2731    2763    2773    2793    2803    2814    2824    2854    3178    3184    3277
```

Cross Reference Listing


```
                        3296    3297    3380    3394    3408    3418    3603    3610    3648    3957    3971    4023
                        4029    4052    4054    5253    5261    5265    5311    5323    5333    5340    5742    5784
                        5928    5983    6190    6544    6616    6623    7330    7357    7375    7386    7413    8264
                        8526
BAD PARITY              1862 #
BASIC DIV STEP          4427 #  4448    4451
BWRITE DISP             1979 #  2476    2482    2596    2598    2631    2637    2645    2647    2666    2668    2711
                        2721    2731    2763    2773    2793    2803    2814    2824    2854    3957    3971    4023
                        4029    4052    4054    8264    8526
BYTE DISP               1996 #  4604    4621    5965    6668    6689
BYTE STEP               4576 #  5796    5798
CALL IBP                4570 #  4591    4596    4610
CALL LOAD PI            2002 #  3500    7147
CALL []                 1892 #  2089    2101    2306    2678    2688    3458    3497    3499    3500    3514    3538
                        3541    3736    3739    3853    3863    3868    3872    3877    3880    3914    3916    4020
                        4048    4075    4084    4094    4104    4127    4237    4258    4313    4319    4333    4337
                        4343    4389    4413    4591    4596    4600    4605    4610    4619    4622    4870    4880
                        4914    4941    4999    5010    5140    5182    5185    5190    5192    5300    5302    5304
                        5319    5320    5321    5380    5388    5416    5456    5461    5473    5480    5487    5506
                        5530    5534    5538    5563    5566    5569    5601    5603    5605    5611    5729    5769
                        5772    5776    5808    5814    5839    5863    5874    5889    5893    5900    5910    5912
                        5918    5932    5941    5962    5977    5980    5993    6008    6019    6027    6030    6033
                        6036    6044    6049    6076    6111    6120    6136    6148    6175    6186    6194    6211
                        6237    6266    6269    6274    6349    6357    6367    6370    6380    6393    6408    6415
                        6436    6440    6508    6542    6566    6666    6685    6690    6742    6750    6761    6985
                        7054    7078    7081    7100    7147    7177    7190    7239    7269    7271    7287    7291
                        7329    7340    7355    7374    7383    7412    7419    7444    7632    7634    7640    7656
                        7704    7706    7723    7726    7728    7730    7733    7736    7739    7742    7745    7747
                        8094    8101    8112    8118    8128    8132    8148    8151    8164    8186    8195    8213
                        8236    8301    8443    8454    8514    8533
CHANGE FLAGS            1912 #  2124    2125    2485    3311    3327    3394    3408    3464    3465    3468    3469
                        3505    3526    3527    3597    3609    3665    3695    3699    3714    3720    3904    3905
                        3906    3957    3971    3988    3998    4001    4031    4032    4056    4106    4118    4128
                        4131    4238    4273    4304    4390    4539    4541    4606    4623    4645    4897    5167
                        5179    5244    5532    6204    6205    6803    7257    7272    8472
CHK PARITY              1861 #  2176    2246    2260    2269    2273    2290    2301    2312    2318    2331    2402
                        2434    2440    2445    2448    2498    2694    2994    3313    3456    3480    3495    3520
                        3524    3585    3602    3608    3630    3647    3653    3666    3724    3752    3868    3870
                        3875    3902    3914    3918    3940    4018    4111    4113    4216    4226    4310    4332
                        4339    4636    4664    4670    4791    5050    5727    5746    6166    6356    6406    6409
                        6437    6713    6728    6798    6915    7019    7023    7027    7031    7035    7075    7105
                        7107    7114    7168    7252    7294    7418    7465    7510    7513    7515    7525    7527
                        7619    7638    7658    7723    7724    7726    7727    7730    7731    7733    7734    7736
                        7737    7739    7740    7742    7743    7745    7746    7748    8090    8163    8349    8398
                        8513    8520    8532    8534
CHK PARITY L            1854 #
CHK PARITY R            1858 #
CLEANUP DISP            2004 #  8477
CLEAR ARXO              1610 #  2678    4535    6079    6120    6246    6252
CLEAR CONTINUE          1901 #
CLEAR EXECUTE           1902 #
CLEAR RUN               1903 #
CLEAR []O               1609 #  2678    4257    4535    6056    6072    6079    6120    6246    6252
CLR FPD                 1921 #  3597    3714    3720    4106    4606    4623    6205
CLR IO BUSY             1907 #  7373    7411
```

Cross Reference Listing

FM WRITE	1630	#	2067	2075	2077	2078	2080	2081	2083	2084	2090	2092	2396
	2428		2679	2929	2950	2985	3018	3024	3207	3296	3336	3339	3342
	3345		3348	3351	3354	3357	3570	3617	3619	3682	3687	3715	4078
	4086		4095	4097	4108	4109	4110	4112	4114	4336	4352	4353	4354
	4356		4359	4378	4380	4383	4510	4515	4520	4526	4527	4528	4589
	4606		4848	4861	4871	4881	4913	4964	5009	5071	5257	5263	5575
	5583		5619	5620	5621	5628	5737	5750	5753	5778	5783	5800	5824
	5844		5845	5851	5861	5864	5871	5872	5873	5888	5899	5901	5903
	5919		5935	5937	5944	5958	5971	5978	5989	6020	6037	6039	6074
	6075		6087	6088	6095	6097	6110	6112	6121	6124	6156	6162	6168
	6173		6174	6178	6187	6193	6200	6201	6202	6361	6392	6421	6494
	6510		6540	6636	6664	6705	6729	6743	6746	6751	6758	6763	6770
	6780		6882	6967	7020	7024	7028	7032	7036	7050	7061	7082	7085
	7115		7116	7200	7201	7341	7449	7483	7489	7491	7529	7631	7703
	7718		7728	7986	7990	7996	7997	7999	8000	8007	8017	8019	8021
	8023		8027	8028	8152	8154	8174	8441					
FORCE EXEC	1693	#	7253	7267	7301								
GEN 17-FE	1848	#	4685										
GEN MSK []	1626	#	4719	4722	4795	4798	5738						
HALT []	2003	#	2098	2155	2173	2424	3487	7266	7298	8466			
HOLD LEFT	1852	#	2136	2138	2183	2196	2197	2203	2208	2223	2224	2236	2247
	2270		2598	2629	2637	2659	2661	3245	3361	3364	3432	3433	3434
	3443		3446	3459	3471	3501	3525	3539	3667	3726	3738	3742	3879
	3891		3896	3925	3941	3992	4635	4665	4874	4933	4956	5008	5013
	5033		5037	5042	5057	5060	5070	5142	5405	5740	5743	5746	5787
	5813		5816	5832	5838	5867	5886	5892	5900	5904	5915	5927	5931
	5938		5957	5985	6062	6077	6115	6150	6167	6194	6205	6209	6265
	6271		6318	6375	6379	6414	6439	6480	6485	6543	6662	6704	6713
	6862		6888	6902	6949	6963	6965	6996	7076	7081	7106	7140	7142
	7144		7146	7158	7218	7219	7220	7221	7222	7223	7224	7246	7316
	7442		7463	7463	7639	7655	7725	7732	7735	7738	7741	7744	7989
	8221		8254	8262	8475	8481	8487	8489	8491	8493	8495	8497	8499
	8500		8501	8523	8539								
HOLD RIGHT	1856	#	2327	2329	2342	2364	2365	2596	2631	2635	2652	2654	2678
	3021		3023	3294	3436	3445	3448	3450	3451	3452	3485	3490	3493
	3513		3534	3547	3678	3826	3828	3830	3832	3834	3836	3838	3840
	3848		3862	3872	3922	3938	4003	4004	4005	4027	4050	4257	4296
	4358		4412	4535	4712	4847	4855	4918	4963	5001	5005	5107	5108
	5111		5112	5136	5137	5169	5170	5230	5231	5307	5323	5426	5427
	5434		5450	5572	5607	5637	5638	5653	5733	5755	5982	5987	6056
	6072		6073	6079	6118	6120	6127	6133	6142	6161	6213	6216	6221
	6246		6252	6323	6417	6419	6489	6493	6547	6589	6593	6597	6601
	6605		6609	6625	6723	6735	6806	6815	6816	6817	6821	6822	6823
	6824		6829	6830	6831	6834	6835	6836	6837	6838	6839	6840	6841
	6856		6896	6899	6920	6925	6935	6940	6947	6952	6971	6972	6980
	6987		7015	7037	7048	7132	7134	7136	7138	7270	7472	7517	7591
	7593		7595	7597	7599	7601	7603	7605	7982	7985	8036	8040	8043
	8048		8181	8200	8228	8232	8253	8257	8260	8271	8317	8416	8421
	8426		8432	8446	8509								
IBP DP	4567	#	4591	4596	4610	5848	6658	6701	6774				
IBP SCAD	4568	#	4591	4596	4610	5848	6658	6701					
IBP SPEC	4569	#	4591	4596	4610								
INH CRY18	1864	#	3417	3587	3634	3663	3683	3688	4929	5007			
INST DISP	1980	#	2261	2265	2274	2279	2287	2308	2314	2338	2343		
INTERRUPT TRAP	1993	#											

Cross Reference Listing


```
JFCL FLAGS              1930 #  3505
JUMP DISP               1986 #  3380  3394  3408  3418
JUMPA                   1988 #  3361  3364  3432  3433  3459  3501
LDB SCAD                4681 #  4690  4691  4692  4693  4694
LEAVE USER              1928 #  2124  3905
LOAD AC BLOCKS          1896 #  2070  6936  6953
LOAD BYTE EA            1887 #  4599  4618  4666  5734  5746  6665
LOAD DST EA             1890 #  6702  6704  6713
LOAD FE                 1770 #  2286  2304  2319  2873  2876  2880  2890  2892  2908  2911  2914
                        2919    2931  2946  2992  4599  4603  4615  4708  4713  4715  4717  4720
                        4783    4796  4799  4800  4802  4804  5106  5134  5143  5145  5165  5171
                        5172    5186  5208  5212  5214  5215  5229  5298  5302  5304  5310  5318
                        5320    5321  5322  5341  5346  5387  5402  5404  5468  5485  5496  5519
                        5545    5559  5566  5569  5581  5598  5603  5605  5626  5651  5659  5795
                        5796    5798  5964  6335  6336  6342  6466  6467  6468  6469  6666  6669
                        6688    6706
LOAD FLAGS              1932 #  2125  3464  3468  3526  3904  7257  7272  8472
LOAD IND EA             1888 #  2248
LOAD INST               1885 #  2177  3562  6799  7659
LOAD INST EA            1886 #  3457  3481  3496  7461
LOAD IR                 1881 #  5733
LOAD PAGE TABLE         1895 #  7174  8280
LOAD PCU                1926 #  3906
LOAD PI                 1899 #  3515  7160  7235  7712
LOAD PXCT               1893 #  3567
LOAD SC                 1769 #  2082  2092  2285  2304  2319  2335  2896  2898  2974  2975  2976
                        2978    2982  2992  3001  3004  3008  3015  3046  3047  3048  3050  3054
                        4018    4046  4083  4103  4123  4149  4161  4167  4175  4185  4193  4236
                        4333    4341  4448  4451  4491  4837  4840  4853  4859  4864  4877  4893
                        4938    4953  4959  5100  5104  5105  5114  5116  5138  5185  5219  5222
                        5227    5245  5246  5247  5249  5252  5256  5366  5376  5386  5420  5423
                        5430    5445  5448  5456  5460  5463  5473  5479  5534  5535  6109  6138
                        6277    6279  6282  6286  6288  6459  6465  6471  6476  6483  6939  6943
                        6957    6959  6975  6977  6985  6992  6994  7184  7197  7204  7392  7399
                        7424    7430  7478  7485  7494  7498  8050  8053  8093  8107  8110  8133
                        8136    8243  8249
LOAD SRC EA             1889 #
LOAD VMA                1692 #  2068  2087  2091  2135  2135  2136  2138  2139  2139  2157  2183
                        2198    2204  2210  2214  2237  2241  2250  2270  2312  2322  2325  2337
                        2691    2695  2696  2927  3233  3233  3245  3360  3360  3361  3364  3365
                        3365    3432  3433  3434  3439  3443  3446  3459  3472  3475  3501  3516
                        3522    3537  3539  3539  3548  3588  3615  3625  3637  3642  3658  3721
                        3749    3857  3858  3858  3865  3866  3866  3879  3879  3898  3899  3925
                        3925    3934  3941  3942  4117  4117  4592  4592  4601  4648  4652  4656
                        4660    4996  5017  5023  5037  5038  5042  5043  5060  5061  5741  5743
                        5744    5767  5915  6164  6191  6236  6242  6249  6265  6270  6326  6404
                        6446    6561  6708  6709  6710  6711  6787  6795  6912  7076  7101  7106
                        7106    7149  7149  7167  7173  7236  7251  7251  7251  7252  7293  7293
                        7301    7382  7417  7420  7420  7442  7464  7523  7615  7623  7633  7636
                        7636    7639  7639  7650  7706  7722  7722  7725  7725  7732  7732  7735
                        7735    7738  7738  7741  7741  7744  7744  7751  8033  8073  8074  8074
                        8081    8082  8082  8086  8088  8157  8159  8272  8288  8344  8346  8375
                        8376    8390  8392  8456  8462  8510  8511  8511  8536  8536  8539  8539
LSH                     1869 #
LSHC                    1872 #  2978  2982  4480  4483
```

Cross Reference Listing

LUUO
1990 # 5695 5697 5699 5701 5703 5705 5707 5709 5711

MEM CYCLE
1689 # 2068 2087 2091 2135 2135 2135 2136 2136 2138 2138 2139
2139 2139 2157 2158 2175 2183 2198 2198 2204 2204 2209 2210
2214 2215 2237 2238 2241 2242 2246 2250 2259 2268 2270 2271
2272 2300 2311 2312 2312 2322 2323 2325 2326 2330 2337 2337
2401 2433 2439 2444 2447 2496 2497 2691 2692 2694 2695 2696
2698 2927 2927 3233 3233 3233 3245 3245 3312 3313 3360 3360
3360 3361 3361 3364 3364 3365 3365 3365 3432 3432 3433 3433
3434 3434 3439 3439 3443 3443 3446 3446 3455 3459 3459 3472
3473 3475 3476 3479 3494 3501 3501 3516 3520 3522 3523 3524
3537 3537 3539 3539 3540 3543 3548 3548 3584 3588 3589 3601
3607 3615 3616 3625 3626 3629 3637 3639 3642 3644 3646 3652
3658 3660 3666 3721 3722 3723 3733 3749 3750 3751 3857 3858
3858 3865 3866 3866 3867 3869 3874 3879 3879 3879 3898 3899
3900 3901 3913 3917 3925 3925 3925 3934 3935 3939 3941 3942
3943 4117 4117 4117 4592 4592 4592 4601 4601 4629 4635 4636
4648 4649 4652 4653 4656 4657 4660 4661 4663 4669 4790 4791
4996 4997 5017 5018 5023 5024 5026 5037 5038 5040 5042 5043
5045 5049 5052 5060 5061 5063 5727 5741 5743 5743 5744 5744
5746 5767 5768 5915 5915 6164 6165 6166 6191 6192 6236 6242
6249 6265 6265 6270 6270 6326 6326 6358 6360 6404 6404 6406
6409 6437 6446 6446 6561 6562 6708 6708 6709 6709 6710 6710
6711 6711 6713 6727 6783 6786 6787 6791 6794 6795 6798 6912
6913 6914 7018 7019 7022 7023 7026 7027 7030 7031 7034 7035
7073 7074 7076 7077 7101 7102 7105 7106 7106 7106 7107 7113
7114 7149 7149 7149 7167 7167 7168 7173 7236 7240 7251 7251
7251 7252 7252 7267 7288 7293 7293 7294 7301 7302 7382 7384
7417 7418 7420 7420 7420 7442 7443 7464 7464 7465 7510 7513
7515 7523 7525 7527 7615 7616 7618 7623 7624 7633 7636 7636
7638 7639 7639 7639 7650 7651 7657 7706 7722 7722 7723 7724
7725 7725 7725 7726 7727 7730 7731 7732 7732 7732 7733 7734
7735 7735 7735 7736 7737 7738 7738 7738 7739 7740 7741 7741
7741 7742 7743 7744 7744 7744 7745 7746 7748 7751 8033 8073
8074 8074 8081 8082 8082 8086 8087 8088 8089 8131 8157 8158
8159 8162 8235 8272 8288 8344 8345 8346 8348 8375 8376 8377
8390 8391 8392 8397 8456 8462 8510 8511 8511 8512 8519 8532
8534 8536 8536 8539 8539 8539

MEM READ
1717 # 2175 2246 2259 2268 2272 2300 2311 2330 3455 3479 3494
3520 3524 3584 3629 3666 3751 3901 4663 4669 5049 5727 5746
6166 6409 6437 6713 6727 6783 6791 6798 6914 7019 7023 7027
7031 7035 7074 7114 7240 7252 7288 7294 7384 7465 7510 7513
7515 7618 7657 8089 8162 8348 8397

MEM WAIT
1716 # 2175 2246 2259 2268 2272 2300 2311 2330 2401 2433 2439
2444 2447 2497 2694 3313 3455 3479 3494 3520 3524 3584 3601
3607 3629 3646 3652 3666 3723 3751 3867 3869 3874 3901 3913
3917 3939 4636 4663 4669 4791 5026 5049 5052 5727 5746 6166
6360 6406 6409 6437 6713 6727 6783 6791 6798 6914 7019 7023
7027 7031 7035 7074 7105 7107 7114 7168 7240 7252 7288 7294
7384 7418 7465 7510 7513 7515 7525 7527 7618 7638 7657 7723
7724 7726 7727 7730 7731 7733 7734 7736 7737 7739 7740 7742
7743 7745 7746 7748 8089 8162 8348 8397 8512 8519 8532 8534

MEM WRITE
1718 # 2401 2433 2439 2444 2447 2497 2694 3313 3601 3607 3646
3652 3723 3867 3869 3874 3913 3917 3939 4636 4791 5026 5052
6360 6406 7105 7107 7168 7418 7525 7527 7638 7723 7724 7726
7727 7730 7731 7733 7734 7736 7737 7739 7740 7742 7743 7745

```
                        7746    7748    8512    8519    8532    8534
MEM_Q                   1725 #  5027    5053    6360
MEM_[]                  1724 #  2402    2434    2440    2445    2448    2498    2694    3313    3602    3608    3647
                        3653    3724    3868    3870    3875    3914    3918    3940    4636    4791    6406    7105
                        7107    7168    7418    7525    7527    7638    7723    7724    7726    7727    7730    7731
                        7733    7734    7736    7737    7739    7740    7742    7743    7745    7746    7748    8513
                        8520    8532    8534
MUL DISP                1994 #  4167    4175    4185    4193    5397    5423    5425    5542
MUL FINAL               4145 #  4171    4179    4189    4197
MUL STEP                4144 #  4167    4175    4185    4193
NEXT INST               1974 #  2429    2950    2996    3024    3336    3618    3694    3701    4381    4384    5257
                        5263    5622    7990
NEXT INST FETCH         1975 #  2135    2136    2138    2139    2199    2205    3233    3245    3360    3361    3364
                        3365    3432    3433    3459    3501    4117    4592    7149    7420
NEXT [] PHYSICAL WRI    1747 #  3879    3925    7639    7725    7732    7735    7738    7741    7744    8539
NO DIVIDE               1936 #  4238    4273    4304    4390    4897
NORM DISP               2000 #  5117    5146    5223    5298    5300    5310    5314    5316    5318    5319    5322
                        5344    5495    5549    5550    5560    5562    5582    5592    5594    5599    5600    5627
                        5656
ONES                    1875 #  4719    4722    4795    4798    5798
PAGE FAIL TRAP          1991 #  8105    8124    8183    8192    8207    8306    8323    8328    8351    8410
PI DISP                 1999 #
PXCT BLT DEST           1701 #  5019    5025    5039    5044
PXCT BLT SRC            1705 #  4998    5062
PXCT BYTE DATA          1703 #  4650    4654    6708    6709
PXCT BYTE PTR EA        1702 #  4658    4662
PXCT DATA               1700 #  2228    2270    2322    2325    2693    2697    3638    3643
PXCT EA                 1699 #  2237    2242    3477
PXCT EXTEND EA          1706 #  5743    5744
PXCT STACK WORD         1704 #  3590    3627    3659
Q-[]                    1574 #  4327
Q.AND.NOT.[]            1575 #
Q_#                     1590 #  4872    4916
Q_-1                    1582 #
Q_-AC[]                 1583 #  4509    4514    4519
Q_-Q                    1584 #  4297    4374    4415    4416    4433    5299    5301    5303    5305    5588
Q_-[]                   1581 #  5524
Q_.NOT.AC[]             1580 #  4513    4518
Q_.NOT.Q                1589 #  5587
Q_0                     1591 #  2888    4718    4794    5114    5185    5196    5208    5212    5214    5215    5228
                        5241    5534    5796
Q_0 XWD []              1592 #
Q_AC                    1585 #  4018    4216
Q_AC[]                  1586 #  2290    2994    4111    4113    4226    6356
Q_AC[].AND.MASK         1587 #
Q_AC[].AND.[]           1588 #  4310    4332    4339
Q_MEM                   1735 #  5050    7515
Q_Q*.5                  1597 #  4126    4160    4333    5449    5539
Q_Q*2                   1598 #
Q_Q+.25                 1593 #  5400    5547
Q_Q+1                   1594 #
Q_Q+AC                  1596 #  4892
Q_Q+[]                  1603 #  4335    4367    5382    5390    5491    5494
Q_Q-1                   1595 #  4932
Q_Q-WORK[]              1682 #  4875
```

```
Q_Q.AND.#               1600 #  3020    4410    5141    5450
Q_Q.AND.NOT.[]          1602 #  5536
Q_Q.AND.[]              1601 #  4382    4801
Q_Q.OR.#                1599 #  3022    4295
Q_WORK[]                1664 #  4939    5955
Q_[]                    1576 #  4044    4083    4103    4123    4337    4363    4863    5138    5373    5385    5445
                        5460    5473    5479    5529
Q_[]+[]                 1578 #  4364
Q_[]-[]                 1577 #  5002
Q_[].AND.Q              1604 #  4458    5571    5606
Q_[].AND.[]             1579 #  4285
Q_[].OR.Q               1605 #  5405
RAM_[]                  1685 #  2090    2092
READ Q                  1615 #  5264    5306    5330    5579    5624
READ XR                 1613 #  3463
READ []                 1614 #  2283    2303    2479    2643    2650    2657    2664    2975    2991    3047    3192
                        3236    3239    3242    3248    3251    3254    3467    3515    3526    3566    3854    3856
                        3864    3897    3903    3933    4021    4227    4233    4247    4252    4254    4259    4326
                        4328    4345    4361    4370    4431    4438    4459    4598    4602    4617    4620    4717
                        4783    4799    4920    5102    5106    5189    5191    5223    5227    5260    5300    5319
                        5387    5486    5495    5544    5562    5600    5734    5752    5795    5829    5930    5933
                        5939    5952    5963    5979    6155    6191    6242    6249    6275    6289    6294    6316
                        6320    6372    6458    6466    6561    6583    6615    6665    6667    6689    6877    6879
                        6880    6885    6889    6966    7066    7101    7180    7186    7193    7460    7712    7722
                        8059    8145    8284    8318    8368    8403    8472    8477
RETURN []               1998 #  2364    2365    3466    3470    3923    3926    4129    4172    4180    4190    4198
                        4392    4414    4415    4417    4501    4528    4630    4632    4636    4671    4704    4723
                        4791    5343    5345    5346    5435    5630    5637    5638    5653    6040    6050    6055
                        6058    6096    6097    6247    6253    6424    6447    6512    6543    6555    6586    6602
                        6619    6628    6630    6656    6691    6724    6730    6735    6775    6996    7062    7086
                        7161    7205    7313    7316    7388    7395    7401    7458    7466    7470    7473    7499
                        7511    7513    7515    7517    7519    7521    7523    7525    7527    7529    7715    7719
                        7753    8294    8305    8334    8337    8338    8350    8359    8362    8363    8462    8539
ROT                     1871 #  2914    2919
ROTC                    1874 #  3050    3054
SCAD DISP               1997 #  4591    4596    4610    4686    5100    5243    5245    5796    5798    5849    6659
                        6701    7485
SC_-1                   1814 #
SC_-2                   1815 #
SC_-SHIFT               1778 #
SC_-SHIFT-1             1779 #  2975    3047
SC_-SHIFT-2             1780 #
SC_0                    1813 #  6465    6476
SC_1                    1812 #
SC_11.                  1803 #
SC_14.                  1802 #
SC_19.                  1801 #  2082
SC_2                    1811 #  6286
SC_20.                  1800 #  6109
SC_22.                  1799 #
SC_24.                  1798 #
SC_26.                  1797 #  5534
SC_27.                  1796 #  5138    5185
SC_28.                  1795 #
SC_3                    1810 #  4853
```

```
SC_34.                  1794 #  4236  4333  4341  4864  4877  4893
SC_35.                  1793 #  4018  4046  4083  4103  4123  4938  5456  5460  5473  5479  5535
SC_36.                  1792 #
SC_4                    1809 #
SC_5                    1808 #  7392  7424
SC_6                    1807 #  5219  5445  6279  6957  6975  6985  6992
SC_7                    1806 #  6277  6939  8050  8093  8133  8243
SC_8.                   1805 #
SC_9.                   1804 #  4837  4953
SC_EXP                  1785 #  2304  2319  5104  5376
SC_FE                   1788 #  2335  5463
SC_FE+S#                1787 #  3001  5245
SC_S#                   1789 #  2082  4018  4046  4083  4103  4123  4236  4333  4341  4837  4853
                        4864    4877  4893  4938  4953  5138  5185  5219  5445  5456  5460  5473
                        5479    5534  5535  6109  6277  6279  6286  6465  6476  6939  6957  6975
                        6985    6992  7204  7392  7424  7478  7494  8050  8093  8133  8243
SC_S#-FE                1786 #  3004  5247
SC_SC-1                 1774 #  7485
SC_SC-EXP               1781 #
SC_SC-EXP-1             1782 #  5100  5366
SC_SC-FE                1784 #
SC_SC-FE-1              1783 #  5105  5386
SC_SHIFT                1775 #  2992  5227
SC_SHIFT-1              1776 #  2285
SC_SHIFT-2              1777 #
SET APR ENABLES         1905 #  2073  6881  6966
SET AROV                1914 #  4031  4032  4056  4118  5244
SET FL NO DIVIDE        1917 #  5167  5179  5532
SET FOV                 1915 #
SET FPD                 1920 #  4131  4645  6204
SET HALT                1900 #  7641  7707
SET NO DIVIDE           1916 #  4238  4273  4304  4390  4897  5167  5179  5532
SET PDL OV              1923 #  3609  3665  3695  3699
SET P TO 36-S           4572 #  4631  5858  6661  6703
SET TRAP1               1924 #
SHIFT                   1772 #  2880  2892  2914  2919
SKIP AC REF             1968 #
SKIP AD.EQ.O            1952 #  2924  3239  3251  3342  3354  4021  4026  4228  4230  4234  4248
                        4293    4322  4352  4433  4511  4516  4521  4537  4544  4862  4891  5163
                        5189    5191  5264  5306  5330  5486  5537  5579  5624  5952  6005  6081
                        6084    6373  6410  6431  6554  7093  8069  8072  8147  8150
SKIP AD.LE.O            1953 #  3242  3254  3345  3357  4287  4921  5183  5507  6152  7059  7711
SKIP ADL.EQ.O           1964 #  2095  3887  3892  5206  5211  5325  5490  5493  5561  5564  5567
                        5570    5630  5728  5771  5910  5912  5994  6014  6198  6214  6268  6303
                        6306    6338  6802  6921  6928  6964  6970  7254  7255  7446  7448  7462
                        8037    8057  8095  8102  8105  8119  8124  8165  8183  8187  8192  8201
                        8205    8207  8218  8258  8277  8279  8302  8306  8320  8323  8325  8328
                        8331    8351  8361  8410  8428  8469  8516
SKIP ADL.LE.O           1955 #  7455
SKIP ADR.EQ.O           1965 #  2993  3499  3514  5015  6139  6212  6567  6844  6847  6867  6869
                        6872    6874  6876  7129  7131  7135  7137  7139  7141  7143  7145  7242
                        7295    7315  7387  7414  7988  8223  8240  8399  8414  8417  8422
SKIP CRYO               1960 #  3592  3635  3664  5175  5299  5301  5303  5305
SKIP CRY1               1961 #  3984  3996  5615  6029  6035  6048  6239  6250
SKIP CRY2               1962 #  5588  5648
```

Cross Reference Listing

SKIP DPO	1950 #	2305	2320	2479	2657	2664	2947	3019	3236	3248	3339	3351
	3681	3684	3689	3855	4003	4028	4050	4053	4108	4116	4217	4253
	4255	4260	4265	4269	4291	4308	4320	4327	4342	4361	4379	4431
	4438	4453	4894	5032	5056	5102	5106	5135	5165	5173	5207	5229
	5260	5367	5469	5516	5520	5531	5544	5779	5792	5809	5822	5829
	5847	5889	5891	5913	5930	5933	5939	5979	6071	6076	6113	6122
	6155	6172	6244	6294	6320	6511	6541	6583	6637	6756	6916	7048
	8060	8145	8319	8369								
SKIP DP18	1951 #	2284	2643	2650	5752	5999	6203	6292	6615	6616	7281	8372
	8382	8404										
SKIP EXECUTE	1969 #	7645										
SKIP FPD	1957 #	4070	4591	4596	4610	6103						
SKIP IF ACO	1949 #	2391	3295	4588								
SKIP IO LEGAL	1959 #	2341	3444	3447								
SKIP IRPT	1966 #	5036	5849	8001	8178	8347	8450	8481	8524			
SKIP JFCL	1963 #	3506										
SKIP KERNEL	1958 #	3437	3441	3442	3449	3558	6781					
SKIP -1MS	1967 #	5844										
SKIP-COMP DISP	1985 #	3277	3296	3297	5928							
SPEC MEM READ	1719 #	2317										
SPEC MEM WRITE	1720 #											
START NO TEST WRITE	1712 #	8131	8235									
START READ	1710 #	2135	2136	2138	2139	2158	2198	2204	2209	2215	2238	2242
	2271	2312	2323	2326	2337	2927	3233	3245	3360	3361	3364	3365
	3432	3433	3434	3439	3443	3446	3459	3473	3476	3501	3523	3543
	3616	3626	3660	3750	3900	3943	4117	4592	4601	4649	4653	4657
	4661	4997	5063	5743	5744	5768	5915	6165	6192	6265	6270	6326
	6446	6562	6708	6709	6710	6711	6786	6794	6913	7018	7022	7026
	7030	7034	7073	7077	7113	7149	7302	7420	7443	7464	7616	7651
	8087	8158	8345	8377	8391							
START WRITE	1711 #	2496	2692	2698	3312	3537	3540	3548	3589	3639	3644	3722
	3733	3879	3925	3935	4629	4635	4790	5018	5024	5040	5045	6358
	6404	7102	7106	7167	7267	7624	7639	7725	7732	7735	7738	7741
	7744	8539										
STATE_[]	1941 #	5008	5787	5813	5816	5838	5886	5892	5900	5931	5938	5957
	5985	6167	6194	6318	6375	6379	6439	7989	8487	8489	8491	8493
	8495	8497	8501									
STEP SC	1771 #	2092	2896	2898	2974	2976	2978	2982	3008	3015	3046	3048
	3050	3054	4149	4161	4167	4175	4185	4193	4448	4451	4491	4840
	4859	4959	5114	5116	5222	5246	5249	5252	5256	5420	5423	5430
	5448	6138	6282	6288	6483	6943	6959	6977	6994	7184	7197	7399
	7430	7498	8053	8107	8110	8136	8249					
SWEEP	1897 #	7192	7193	7196								
TAKE INTERRUPT	1992 #	8509										
TEST DISP	1984 #	3178	3184	7330								
TL []	1619 #	3887	3892	5325	5490	5493	5561	5564	5567	5570	5630	5728
	5771	5910	5912	5994	6014	6198	6214	6268	6303	6306	7446	7448
	7462	8037	8057	8095	8102	8105	8119	8124	8165	8183	8187	8192
	8201	8205	8207	8218	8258	8277	8279	8302	8306	8320	8323	8325
	8328	8331	8351	8361	8410	8428	8469	8516				
TR []	1618 #	3499	3514	6139	6212	6567	6867	6869	6872	6874	6876	7129
	7131	7135	7137	7139	7141	7143	7145	7387	7414	7988	8223	8240
	8399	8414	8417	8422								
TURN OFF PXCT	1894 #	2147	2150	2153	2165	2168	2171	2185				
TXXX TEST	1970 #	3192										

```
UNHALT                          1904 #  7644
UPDATE USER                     1927 #  3465    3469    3527
UUO                             1989 #  2342    3436    3445    3448    3450    3451    3452    3485    3490    3493    3513
                                3534    3547    3826    3828    3830    3832    3834    3836    3838    3840    6723    6806
                                6815    6816    6817    6821    6822    6823    6824    6829    6830    6831    6834    6835
                                6836    6837    6838    6839    6840    6841    7015    7037    7132    7591    7593    7595
                                7597    7599    7601    7603    7605
VMA                             1759 #
VMA EXTENDED                    1697 #
VMA PHYSICAL                    1694 #  3858    3866    3879    3899    3925    6787    6795    7251    7293    7636    7639
                                7722    7725    7732    7735    7738    7741    7744    8074    8082    8088    8159    8346
                                8376    8392    8511    8536    8539
VMA PHYSICAL READ               1696 #  7251    7293    8074    8082    8536
VMA PHYSICAL WRITE              1695 #  3858    3866    7636    7722    8511
VMA_[]                          1739 #  2135    2139    2157    2214    2241    2250    2337    2695    3233    3360    3365
                                3439    3475    3537    3548    3637    3642    3721    3749    4117    4592    4601    4652
                                4660    4996    5017    5023    5744    6270    6326    6404    6446    6709    6711    6912
                                7149    7167    7173    7251    7420    7464    7615    7623    7650    7751    8086    8272
                                8288    8344    8510
VMA_[] WITH FLAGS               1740 #  7523
VMA_[]+1                        1742 #  2136    2138    2183    2270    3245    3539    3941    5037    5042    5060    5915
                                6265    7076    7106
VMA_[]+XR                       1744 #  4648    4656    6708    6710
VMA_[]+[]                       1745 #  8073    8081    8157
VMA_[]-1                        1743 #  3434    3443    3446    7442
VMA_[].OR.[] WITH FL            1741 #  7382    7417
WORK[]                          1760 #  5947    6169    6368    6435    6513    6544    6547
WORK[]_.NOT.[]                  1670 #  5989
WORK[]_0                        1665 #  2084    7200    7201
WORK[]_1                        1666 #  2083
WORK[]_Q                        1663 #  4871    5919    5958
WORK[]_[]                       1667 #  2067    2075    2077    2078    4589    4848    4861    4881    4913    5737    5750
                                5753    5783    5845    5871    5872    5873    5888    5935    5971    6110    6112    6174
                                6187    6193    6392    6540    6729    6780    6882    6967    7020    7024    7028    7032
                                7036    7050    7061    7082    7085    7115    7116    7483    7489    7491    7631    7703
                                7718    7728    7986    7996    7997    7999    8000    8007    8017    8019    8021    8023
                                8027    8028    8152    8154    8174    8441
WORK[]_[] CLR LH                1668 #  3570    7449
WORK[]_[]-1                     1669 #  6168
WORK[]_[].AND.[]                1671 #  5800
WRITE TEST                      1709 #  2496    2692    2698    3312    3537    3540    3548    3589    3639    3644    3722
                                3733    3879    3925    3935    4629    4635    4790    5018    5024    5040    5045    6358
                                6404    7102    7106    7167    7267    7624    7639    7725    7732    7735    7738    7741
                                7744    8539
XR                              1758 #
[] LEFT_-1                      1560 #  2647
[] LEFT_0                       1558 #  2645
[] RIGHT_-1                     1561 #  2668
[] RIGHT_0                      1559 #  2666
[]+[]                           1425 #  2690    4128    4268
[]-#                            1427 #  5998    6291    8370    8380
[]-[]                           1426 #  4264    4320    4321    5014    5030    5054    5175    5183    5531    5913
[].AND.#                        1428 #  5206    5210    6337    6800    6926    6964    6970
[].AND.NOT.WORK[]               1672 #
[].AND.NOT.[]                   1431 #  4025
```

Cross Reference Listing


```
[].AND.Q                1429 #  4292
[].AND.WORK[]           1673 #  6553
[].AND.[]               1430 #  4053  6844  6847  7315
[].OR.[]                1432 #
[].XOR.#                1433 #  2095  7254  7255
[].XOR.[]               1434 #  7092
[]_#                    1436 #  2056  2060  2085  2086  2126  2652  2654  2659  2661  4437  4845
                        4868    4878  4882  4911  4954  5008  5755  5787  5813  5816  5838  5886
                        5892    5900  5931  5938  5957  5985  6167  6194  6318  6375  6379  6439
                        6851    6853  7176  7178  7189  7191  7270  7312  7989  8487  8489  8491
                        8493    8495  8497  8501
[]_#-[]                 1435 #
[]_(#-[])*2             1522 #  4949
[]_(-[])*.5             1523 #
[]_(-[]-.25)*.5 LONG    1524 #  4298  5525
[]_(-[]-.25)*2 LONG     1525 #  4375
[]_(AC[].AND.[])*.5     1520 #  4074  4076  5364  5455  5510
[]_(MEM.AND.[])*.5      1734 #  2331
[]_(Q+1)*.5             1521 #  5647
[]_([]+#)*2             1535 #
[]_([]+1)*2             1536 #  6457
[]_([]+[])*.5 LONG      1537 #  5394
[]_([]+[])*2 LONG       1538 #  4482
[]_([]+[]+.25)*.5 LO    1541 #
[]_([]-[])*.5 LONG      1539 #
[]_([]-[])*2 LONG       1540 #  4479
[]_([].AND.#)*.5        1527 #  4842  4849  7296
[]_([].AND.#)*2         1528 #
[]_([].AND.NOT.#)*.5    1529 #
[]_([].AND.NOT.#)*2     1530 #
[]_([].AND.[])*.5       1531 #  4068
[]_([].AND.[])*2        1532 #
[]_([].OR.#)*.5         1533 #
[]_([].OR.#)*2          1534 #
[]_+SIGN                1564 #  2329  2364  5107  5111  5136  5169  5230
[]_+SIGN*.5             1569 #  5368  5470  5521
[]_-1                   1437 #  4220  6204  7158
[]_-2                   1438 #
[]_-AC                  1453 #  4524
[]_-AC[]                1454 #  6083
[]_-Q                   1439 #
[]_-Q*.5                1441 #  5194
[]_-Q*2                 1440 #  4348
[]_-SIGN                1565 #  2327  2365  5108  5112  5137  5170  5231
[]_-SIGN*.5             1570 #  5370  5471  5523
[]_-WORK[]              1680 #  5811  6754  6768
[]_-[]                  1442 #  2484  2854  4417  4435  4442  4536  4540  4544  4546  5097  5172
                        5207    5316  5332  5361  5513  5593  5614  6080
[]_-[]*2                1444 #  5176
[]_-[]-.25              1443 #  5362
[]_.NOT.AC[]            1446 #  6082
[]_.NOT.AC              1445 #  2803  2813  4523
[]_.NOT.Q               1447 #  4808  5799
[]_.NOT.WORK[]          1679 #  5828  5856  5860  5996  6000  6772
[]_.NOT.[]              1448 #  2746  2783  2824  2834  3195  3990  4434  4538  4545  5266  5314
```

Cross Reference Listing

	5329	5331	5512	5522	5591	5780	5782	6334	7445	8360		
[]_O	1449 #	2068	2070	2073	2082	2101	2123	2307	2711	3178	3871	4122
	4218	4432	4712	4854	5000	5033	5057	5333	5506	5573	5589	5617
	5832	5904	5916	5927	5956	6077	6115	6149	6205	6209	6213	6216
	6271	6543	6938	6948	6979	7055	7130	7175	7243	7472	7517	7636
	8254	8481	8499	8500	8523							
[]_O XWD []	1451 #	2062	2065	2109	2145	2149	2152	2164	2167	2170	3850	3883
	3930	4294	4305	5695	5697	5699	5701	5703	5705	5707	5709	5711
	6407	7046	7225	7226	7227	7228	7229	7230	7231			
[]_O*.5 LONG	1450 #	4148										
[]_AC[]	1468 #	2088	2278	2688	5770	5788	5791	5821	5837	5909	5929	5961
	5976	6004	6007	6009	6026	6067	6069	6070	6102	6108	6135	6147
	6182	6325	6366	6403	6405	6509	6622	6629	6655	6657	6684	6741
	6749	6755										
[]_AC[]*.5	1470 #	4088										
[]_AC[]*2	1469 #	6016										
[]_AC[]-1	1465 #	6176	6195									
[]_AC[]-[]	1464 #	3995										
[]_AC[].AND.[]	1466 #											
[]_AC	1452 #	2264	2277	2495	2596	2598	2629	2635	2872	2889	3380	3624
	3657	3679	3735	3747	4046	4225	4616	4994	5100	5133	5165	5229
	5375	5773	5911	5973	6107	6109	6130	6267	7344	7626		
[]_AC*.5	1455 #	2895	2907	2910	4093	4308	5365	5467	5518			
[]_AC*.5 LONG	1456 #	2291	2997									
[]_AC*2	1457 #	4612										
[]_AC+1	1458 #	3394										
[]_AC+1000001	1459 #	3416	3586									
[]_AC+[]	1460 #	5006										
[]_AC-1	1461 #	3408										
[]_AC-[]	1462 #	3277	3970	4000								
[]_AC-[]-.25	1463 #	3997										
[]_AC.AND.MASK	1467 #	2875	2923									
[]_APR	1471 #	6818	6819	6871	6903	7064						
[]_CURRENT AC []	1472 #											
[]_EA	1475 #	2227										
[]_EA FROM []	1473 #	5741										
[]_EXP	1476 #	5307	5323	5653								
[]_FE	1477 #	2948	6470									
[]_FLAGS	1478 #	3535	3549	3861	3886	7256	8517					
[]_IO DATA	1729 #	7241	7289	7385	8009							
[]_MEM	1730 #	2176	2246	2260	2269	2273	2301	2318	3456	3480	3495	3520
	3524	3585	3630	3666	3752	3902	4664	5727	5746	6166	6409	6437
	6713	6728	6798	6915	7019	7023	7027	7031	7035	7075	7114	7252
	7294	7465	7510	7513	7619	7658	8090	8163	8349	8398		
[]_MEM THEN FETCH	1731 #	2312										
[]_MEM*.5	1732 #											
[]_MEM.AND.MASK	1733 #	4670										
[]_P	1479 #	6340	6486									
[]_PC WITH FLAGS	1480 #	3596	3713	3719	3915	4115	6779	7268	8518			
[]_Q	1481 #	4023	4031	4251	4306	4323	4329	4338	5424	5431	5451	5474
	5533	5535										
[]_Q*.5	1482 #	4086	5188	5193	5463							
[]_Q*2	1483 #											
[]_Q*2 LONG	1484 #	4334	4344									
[]_Q+1	1485 #	5003										

Cross Reference Listing


```
[]_RAM                  1486 #
[]_TIME                 1487 #  7089  7090  7091
[]_VMA                  1488 #  7479  7717  7983  7998  8442
[]_VMA FLAGS            1750 #  7237  7284  7376  7379  7415  7426  7432  7653
[]_VMA IO READ          1751 #  7237  7284  7376  7379  7653
[]_VMA IO WRITE         1752 #  7415  7426  7432
[]_WORK[]               1678 #  4866  4876  5069  5875  5876  5877  5878  5942  6153  6163  6196
                        6197    6218  6394  6430  6507  6627  6638  6745  6860  6894  6962  7006
                        7008    7010  7012  7014  7043  7052  7057  7063  7095  7099  7120  7452
                        7457    7705  7710  7714  7729  7747  7749  7750  7752  7987  8068  8071
                        8146    8149  8212  8214  8245  8285  8286  8287  8293  8452  8458  8467
                        8471    8483
[]_WORK[]+1             1681 #  5807  5840  5895  5993  6171  6634  6640
[]_XR                   1489 #  7455
[]_[]                   1490 #  2202  2926  3361  3364  3432  3433  3459  3486  3501  3560  3614
                        3732    3737  4017  4045  4069  4085  4215  4224  4239  4245  4274  4629
                        4937    5103  5104  5105  5143  5163  5171  5380  5381  5388  5389  5462
                        5504    5507  5549  5550  5789  5809  5855  5890  6150  6161  6169  6236
                        6336    6569  6660  6683  6702  6759  6825  6826  6969  6984  6991  6996
                        7094    7126  7272  7354  7356  7363  7388  8049  8127  8242  8266  8525
[]_[] SWAP              1491 #  2296  2476  2623  2625  2628  2631  2634  2637  3176  3181  3500
                        3677    3734  3748  3860  4710  4836  4852  4952  4995  5004  5068  6215
                        6272    6277  6278  6572  6895  7134  7147  7157  8172  8406
[]_[] XWD 0             1492 #  5970  6085  7505  8013  8015  8029
[]_[]*.5                1493 #  2059  2333  2334  2891  2912  2913  2917  3000  3003  4067  4124
                        4286    4314  4445  4446  4447  4453  4714  4790  4839  4858  5178  5186
                        5253    5302  5304  5320  5321  5340  5341  5346  5456  5489  5658  6279
                        6282    6286  6288  6368  6476  6483  6546  6975  6977  6985  6992  6994
                        7299    7317  7392  7398  8052  8062
[]_[]*.5 LONG           1494 #  2292  2978  3006  3050  4019  4047  4126  4147  4160  4288  4290
                        4302    4318  4333  4340  4392  4440  4940  5116  5139  5144  5219  5221
                        5251    5419  5422  5430  5433  5435  5449  5484  5539  5565  5568  5602
                        5604    5650
[]_[]*2                 1495 #  2057  2879  2902  2915  2918  2945  4022  4049  4105  4958  5173
                        5181    5256  6017  6563  6942  6957  6959  7244  7280  7424  7430  7521
                        8106    8109  8135  8248
[]_[]*2 LONG            1496 #  2898  2979  2982  2983  2984  3009  3013  3016  3051  3054  3055
                        4372    4452  4455  4457  4803  5298  5310  5318  5322  5396  5401  5403
                        5415    5417  5418  5447  5488  5492  5541  5558  5580  5597  5625
[]_[]*4                 1497 #  6273
[]_[]+#                 1498 #  3632  3661  6232  6241  7245  7290  8076  8084  8384  8394  8506
[]_[]+.25               1499 #  5654
[]_[]+0                 1500 #
[]_[]+1                 1501 #  2087  2091  2321  2324  3311  3521  3542  4635  5035  5059  5265
                        5766    5867  5885  5914  6057  6125  6137  6144  6164  6185  6199  6254
                        6317    6341  6435  6662  6704  7053  7519  8485  8535
[]_[]+1000001           1502 #  3725  3740  5011
[]_[]+AC                1503 #  3956  3986
[]_[]+AC[]              1504 #  3983  6018  6028  6031
[]_[]+Q                 1505 #  4346
[]_[]+RAM               1506 #  6248  6250
[]_[]+WORK[]            1674 #  4934  6189  6369  6445  6552  6560  7096  7469  7471  8098  8100
                        8168    8300
[]_[]+XR                1507 #  2196  2208  2223  2236  3471  5740  5743  7463
[]_[]+[]                1508 #  3852  3885  4099  4365  4368  4456  5117  5261  5344  5475  5830
```

	5862	6034	6045	6047	6484	6766	6782	6790	7047	7247	7282	7292
	7300	8374	8389									
[]_[]+[]+.25	1509 #											
[]_[]-#	1510 #											
[]_[]-1	1511 #	3327	5846	5950	5951	6062	6203	6240	6633	8475	8527	
[]_[]-1000001	1512 #											
[]_[]-AC	1513 #											
[]_[]-RAM	1514 #	6238	6243									
[]_[]-WORK[]	1683 #											
[]_[]-[]	1515 #	5825	6151	6762	7058	7182	7195					
[]_[]-[] REV	1516 #	5949										
[]_[].AND.AC	1542 #	2721	2844	3183	3206	7330						
[]_[].AND.NOT.#	1543 #	3528	3921	5638	6220	6588	6604	6861	6887	6972	7142	8230
	8267	8412										
[]_[].AND.NOT.AC	1545 #	2731	7360									
[]_[].AND.NOT.WORK[]	1676 #											
[]_[].AND.NOT.[]	1544 #	5399	6870	6873	7136	7146	7316					
[]_[].AND.Q	1546 #	4051	4723	4810								
[]_[].AND.WORK[]	1675 #	5948	8304									
[]_[].AND.[]	1547 #	4809	5326	5608	5610	5946	7159	8316				
[]_[].AND.#	1517 #	2329	2342	2364	3436	3445	3448	3450	3451	3452	3485	3490
	3493	3513	3534	3547	3826	3828	3830	3832	3834	3836	3838	3840
	3846	3936	4004	4961	5107	5111	5136	5169	5230	5981	6104	6280
	6283	6322	6453	6487	6490	6723	6733	6806	6815	6816	6817	6821
	6822	6823	6824	6829	6830	6831	6834	6835	6836	6837	6838	6839
	6840	6841	6863	6865	6897	6900	6918	6923	6930	6945	6963	6978
	6986	6995	7015	7037	7079	7132	7133	7394	7400	7591	7593	7595
	7597	7599	7601	7603	7605	7984	8041	8179	8251	8261	8444	
[]_[].AND.# CLR LH	1518 #	2949	6395	6433	6473	6618	6843	6846	6904	7233	8055	8125
	8176	8193	8274									
[]_[].AND.# CLR RH	1519 #	5730	5774	5974								
[]_[].EQV.AC	1548 #	2793										
[]_[].EQV.Q	1549 #	4029	4032	4054	4055							
[]_[].OR.#	1550 #	2058	2327	2365	3889	3894	4005	4358	4443	5108	5112	5137
	5170	5231	5426	5427	5434	5637	5986	6073	6117	6126	6131	6141
	6412	6416	6418	6479	6592	6596	6600	6608	6624	6965	6971	7065
	7140	7218	7219	7220	7221	7222	7223	7224	7654	7980	8034	8038
	8199	8202	8219	8226	8255	8259	8276	8278	8282	8314	8415	8419
	8424	8431	8436	8509								
[]_[].OR.AC	1551 #	2773	3201	7358								
[]_[].OR.FLAGS	1552 #											
[]_[].OR.WORK[]	1677 #	6760	6769	8129	8208							
[]_[].OR.[]	1553 #	2814	3991	4811	5732	5831	6061	6492	6864	6868	6875	6906
	6933	6950	6951	7138	7144	7286	8263	8447				
[]_[].XOR.AC	1555 #	2763	3198									
[]_[].XOR.[]	1556 #	5612										
[]_[].XOR.#	1554 #	4312	8047									
.NOT.[]	1424 #	4229	7160	7235								
2T	1762 #	4308	4321	5106	5376	5948	6018	6082	6370	6507	6552	7099
	7456											
3T	1763 #	2095	2320	2330	2485	3018	3192	3277	3311	3327	3417	3507
	3591	3626	3634	3659	3663	3680	3685	3690	3727	3741	3957	3971
	4074	4076	4095	4129	4266	4270	4320	4326	4327	4328	4345	4349
	4370	4378	4459	4524	4541	4599	4614	4685	4703	4799	4843	4850
	4930	4946	4950	5012	5014	5031	5055	5100	5134	5175	5206	5207

Cross Reference Listing

		5210	5242	5245	5299	5301	5303	5305	5364	5365	5368	5370	5469
		5470	5471	5515	5519	5521	5523	5531	5614	5620	5778	5796	5798
		5846	5850	5913	6020	6034	6048	6080	6083	6168	6171	6203	6204
		6232	6238	6241	6250	6275	6289	6316	6336	6337	6459	6466	6510
		6569	6659	6666	6687	6701	6705	6708	6710	6746	6801	6915	6921
		6927	6964	6970	7216	7241	7245	7253	7254	7255	7290	7294	7296
		7463	8076	8084	8272	8289	8384	8395	8477	8507			
	4T	1764 #	2923	3394	3408	3983	3988	3995	3998	4000	4890	4895	5792
		5822	5999	6004	6029	6070	6243	6291	6431	6554	6755	7711	8004
		8069	8072	8147	8150	8372	8382						
	5T	1765 #	7485	7502									
	7-BIT DPB	4780 #	4784	4785	4786	4787	4788						
	7-BIT LDB	4682 #	4690	4691	4692	4693	4694						
(D)	MACRO%												
	AC	2031 #	2454	2455	2459	2460	2464	2465	2469	2470	2491	2507	2508
		2512	2513	2517	2518	2522	2523	2527	2528	2532	2533	2537	2538
		2542	2543	2549	2550	2554	2555	2559	2560	2564	2565	2569	2570
		2574	2575	2579	2580	2584	2585	2674	2704	2705	2714	2715	2724
		2725	2734	2735	2739	2740	2756	2757	2766	2767	2776	2777	2786
		2787	2796	2797	2806	2807	2817	2818	2827	2828	2837	2838	2847
		2848	3949	3950	3963	3964	4011	4012	4581	7976			
	AC DISP	2045 #	3423	6811	6812	7001							
	B	2033 #	2707	2717	2727	2737	2742	2759	2769	2779	2789	2799	2809
		2820	2830	2840	2850	3952	3966	4014					
	DAC	2035 #	2673	3978	3979	4036	4037	4062	4203	4204	4208	4209	4280
		5440	5501										
	DBL AC	2020 #	2682										
	DBL B	2036 #	4039	4206	4211								
	DBL FL-R	2026 #	5351	5352	5440	5501							
	DBL R	2019 #	2673	2674	3978	3979	4062	4280					
	FL-AC	2037 #	5077	5080	5081	5085	5088	5089	5122	5126	5127	5151	5155
		5156	5201	5202	5236	5237							
	FL-BOTH	2039 #	5079	5083	5087	5091	5124	5129	5153	5158			
	FL-I	2025 #	5081	5089	5127	5156							
	FL-MEM	2038 #	5078	5082	5086	5090	5123	5128	5152	5157			
	FL-R	2023 #	5077	5080	5085	5088	5122	5126	5151	5155	5236	5237	
	FL-RW	2024 #	5078	5079	5082	5083	5086	5087	5090	5091	5123	5124	5128
		5129	5152	5153	5157	5158							
	I	2011 #	2750	2862	3082	3083	3084	3085	3086	3087	3088	3089	3091
		3092	3100	3101	3102	3103	3104	3105	3106	3107	3118	3119	3120
		3121	3122	3123	3124	3125	3135	3136	3137	3138	3139	3140	3141
		3142	3257	3258	3259	3260	3261	3262	3263	3264	3369	3370	3371
		3372	3373	3374	3375	3376	3384	3385	3386	3387	3388	3389	3390
		3398	3399	3400	3401	3402	3403	3404	3411	3412	3423	3425	3576
		3578	3579	3706	3707	3708	3709	3759	3760	3761	3762	3763	3764
		3765	3766	3770	3771	3772	3773	3774	3775	3776	3777	3778	3779
		3780	3781	3782	3783	3784	3785	3786	3787	3788	3789	3790	3791
		3792	3793	3794	3795	3796	3797	3798	3799	3800	3801	3805	3806
		3807	3808	3812	3813	3814	3815	3816	3817	3818	3819	3820	3821
		3822	4990	5202	5666	5667	5668	5669	5670	5671	5672	5674	5675
		5676	5677	5679	5680	5681	5682	5684	5685	5686	5687	5688	5689
		5690	5691	5723	7534	7535	7536	7538	7539	7541	7542	7544	7545
		7546	7547	7548	7549	7550	7551	7553	7554	7555	7556	7557	7558
		7559	7560	7562	7563	7564	7565	7566	7567	7568	7569	7571	7572
		7573	7574	7575	7576	7577	7578	7580	7581	7582	7583	7584	7585

Cross Reference Listing

	7586	7587										
I-PF	2012 #	2455	2460	2465	2470	2508	2513	2518	2523	2528	2533	2538
	2543	2550	2555	2560	2565	2570	2575	2580	2585	2704	2705	2715
	2725	2735	2740	2757	2767	2777	2787	2796	2797	2807	2818	2828
	2838	2847	2848	2863	3383	3397	3673	3950	3964	4012	4037	4204
	4209											
IOT	2027 #	6811	6812	7001	7322	7323	7324	7325	7333	7334	7335	7336
	7347	7348	7349	7350	7610	7611	7976					
IR	2018 #	3577										
IW	2017 #	2706	2707	2798	2799	2849	2850					
M	2032 #	2456	2461	2466	2471	2509	2514	2519	2524	2529	2534	2539
	2544	2551	2556	2561	2566	2571	2576	2581	2586	2706	2716	2726
	2736	2741	2751	2752	2758	2768	2778	2788	2798	2808	2819	2829
	2839	2849	3951	3965	4013	4038	4205	4210	7001			
R	2013 #	2491	2749	3093	3094	3095	3096	3097	3098	3109	3110	3111
	3112	3113	3114	3115	3116	3127	3128	3129	3130	3131	3132	3133
	3134	3144	3145	3146	3147	3148	3149	3150	3151	3266	3267	3268
	3269	3270	3271	3272	3273	3283	3284	3285	3286	3287	3288	3289
	3290	3554	4581	4582	4583	4584	4585	5201				
R-PF	2014 #	2454	2459	2464	2469	2507	2512	2517	2522	2527	2532	2537
	2542	2549	2554	2559	2564	2569	2574	2579	2584	2714	2724	2734
	2739	2756	2766	2776	2786	2806	2817	2827	2837	3949	3963	4011
	4036	4203	4208									
ROUND	2040 #	5080	5081	5082	5083	5088	5089	5090	5091	5126	5127	5128
	5129	5155	5156	5157	5158	5201	5237					
RW	2016 #	2457	2462	2467	2472	2509	2510	2514	2515	2520	2525	2530
	2535	2540	2545	2551	2552	2556	2557	2562	2567	2572	2577	2582
	2587	2716	2717	2726	2727	2736	2737	2741	2742	2758	2759	2768
	2769	2778	2779	2788	2789	2808	2809	2819	2820	2829	2830	2839
	2840	3300	3301	3302	3303	3304	3305	3306	3307	3316	3317	3318
	3319	3320	3321	3322	3323	3951	3952	3965	3966	4013	4014	4038
	4039	4205	4206	4210	4211							
S	2034 #	2457	2462	2467	2472	2510	2515	2520	2525	2530	2535	2540
	2545	2552	2557	2562	2567	2572	2577	2582	2587			
SH	2021 #	2859	2860	2861								
SHC	2022 #	2864	2865									
SJC-	3217 #	3257	3266	3283	3300	3316	3369	3383	3397			
SJCA	3221 #	3261	3270	3287	3304	3320	3373	3387	3401			
SJCE	3219 #	3259	3268	3285	3302	3318	3371	3385	3399	5667		
SJCG	3224 #	3264	3273	3290	3307	3323	3376	3390	3404	5672		
SJCGE	3222 #	3262	3271	3288	3305	3321	3374	3388	3402	3411	5670	
SJCL	3218 #	3258	3267	3284	3301	3317	3370	3384	3398	3412	5666	
SJCLE	3220 #	3260	3269	3286	3303	3319	3372	3386	3400	5668		
SJCN	3223 #	3263	3272	3289	3306	3322	3375	3389	3403	5671		
TC-	3073 #	3118	3119	3127	3128							
TCA	3075 #	3122	3123	3131	3132							
TCE	3074 #	3120	3121	3129	3130							
TCN	3076 #	3124	3125	3133	3134							
TN-	3063 #											
TNA	3066 #	3086	3087	3095	3096							
TNE	3064 #	3084	3085	3093	3094	7324						
TNN	3067 #	3088	3089	3097	3098	7325						
TO-	3077 #	3135	3136	3144	3145							
TOA	3079 #	3139	3140	3148	3149							
TOE	3078 #	3137	3138	3146	3147							

Cross Reference Listing

```
        TON                     3080 #  3141    3142    3150    3151
        TZ-                     3069 #  3100    3101    3109    3110
        TZA                     3071 #  3104    3105    3113    3114
        TZE                     3070 #  3102    3103    3111    3112
        TZN                     3072 #  3106    3107    3115    3116
        W                       2015 #  2456    2461    2466    2471    2519    2524    2529    2534    2539    2544    2561
                                2566    2571    2576    2581    2586    2683    2751    2752
        W TEST                  2044 #  2491    4582    4584
        WORD-TNE                3065 #  7322
        WORD-TNN                3068 #  7323
(U) MEM                         908 #   2068    2087    2091    2135    2135    2135    2136    2136    2138    2138    2139
                                2139    2139    2157    2158    2175    2183    2198    2198    2204    2204    2209    2210
                                2214    2215    2228    2237    2238    2241    2242    2246    2250    2259    2268    2270
                                2271    2272    2300    2311    2312    2312    2322    2323    2325    2326    2330    2337
                                2337    2401    2433    2439    2444    2447    2476    2482    2496    2497    2596    2598
                                2631    2637    2645    2647    2666    2668    2691    2692    2694    2695    2696    2698
                                2711    2721    2731    2763    2773    2793    2803    2814    2824    2854    2927    2927
                                3233    3233    3233    3245    3245    3312    3313    3360    3360    3360    3361    3361
                                3364    3364    3365    3365    3365    3432    3432    3433    3433    3434    3434    3439
                                3439    3443    3443    3446    3446    3455    3459    3459    3472    3473    3475    3476
                                3479    3494    3501    3501    3516    3520    3522    3523    3524    3537    3537    3539
                                3539    3540    3543    3548    3548    3584    3588    3589    3601    3607    3615    3616
                                3625    3626    3629    3637    3639    3642    3644    3646    3652    3658    3660    3666
                                3721    3722    3723    3733    3749    3750    3751    3857    3858    3858    3858    3865
                                3866    3866    3866    3867    3869    3874    3879    3879    3879    3898    3899    3900
                                3901    3913    3917    3925    3925    3925    3934    3935    3939    3941    3942    3943
                                3957    3971    4023    4029    4052    4054    4117    4117    4117    4592    4592    4592
                                4601    4601    4629    4635    4636    4648    4649    4652    4653    4656    4657    4660
                                4661    4663    4669    4790    4791    4996    4997    5017    5018    5023    5024    5026
                                5037    5038    5040    5042    5043    5045    5049    5052    5060    5061    5063    5261
                                5265    5311    5333    5727    5741    5743    5743    5744    5744    5746    5767    5768
                                5915    5915    6164    6165    6166    6191    6192    6236    6242    6249    6265    6265
                                6270    6270    6326    6326    6358    6360    6404    6404    6406    6409    6437    6446
                                6446    6561    6562    6708    6708    6709    6709    6710    6710    6711    6711    6713
                                6727    6783    6786    6787    6791    6794    6795    6798    6912    6913    6914    7018
                                7019    7022    7023    7026    7027    7030    7031    7034    7035    7073    7074    7076
                                7077    7101    7102    7105    7106    7106    7106    7107    7113    7114    7149    7149
                                7149    7167    7167    7168    7173    7236    7240    7251    7251    7251    7251    7252
                                7252    7267    7288    7293    7293    7293    7294    7301    7302    7382    7382    7384
                                7417    7417    7418    7420    7420    7420    7442    7443    7464    7464    7465    7510
                                7513    7515    7523    7523    7525    7527    7615    7616    7618    7623    7624    7633
                                7636    7636    7636    7638    7639    7639    7639    7650    7651    7657    7706    7722
                                7722    7722    7723    7724    7725    7725    7725    7726    7727    7730    7731    7732
                                7732    7732    7733    7734    7735    7735    7735    7736    7737    7738    7738    7738
                                7739    7740    7741    7741    7741    7742    7743    7744    7744    7744    7745    7746
                                7748    7751    8033    8073    8074    8074    8074    8081    8082    8082    8082    8086
                                8087    8088    8089    8131    8157    8158    8159    8162    8235    8264    8272    8288
                                8344    8345    8346    8348    8375    8376    8377    8390    8391    8392    8397    8456
                                8462    8510    8511    8511    8511    8512    8519    8526    8532    8534    8536    8536
                                8536    8539    8539    8539
(U) MICROCODE OPTIONS           1182 #
        OPT                     1183 #  6854
(U) MICROCODE VERSION           1184 #
        UCV                     1185 #  6855
(D) MODE                        1245 #
```

```
(U) MULTI PREC          913 #   4299    4351    4366    4369    4376    4490    5362    5395    5515    5526
(U) MULTI SHIFT         915 #   2880    2892    2914    2919    4716    4721    4797    4805
(U) PHYSICAL           1082 #   3858    3866    3879    3899    3925    6787    6795    7179    7180    7183    7192
                       7193     7196    7237    7251    7284    7293    7376    7379    7415    7426    7432    7636
                       7639     7653    7722    7725    7732    7735    7738    7741    7744    8074    8082    8088
                       8159     8346    8376    8392    8511    8536    8539
(U) PI.CLR             1141 #   7129
(U) PI.CO1             1130 #
(U) PI.CO2             1131 #
(U) PI.DIR             1140 #   7135
(U) PI.IP1             1122 #   7312
(U) PI.IP2             1123 #
(U) PI.IP3             1124 #
(U) PI.IP4             1125 #
(U) PI.IP5             1126 #
(U) PI.IP6             1127 #
(U) PI.IP7             1128 #
(U) PI.MBZ             1139 #   7131
(U) PI.ON              1129 #   7140    7142
(U) PI.REQ             1142 #   7137
(U) PI.SC1             1147 #
(U) PI.SC2             1148 #
(U) PI.SC3             1149 #
(U) PI.SC4             1150 #
(U) PI.SC5             1151 #
(U) PI.SC6             1152 #
(U) PI.SC7             1153 #
(U) PI.TCF             1144 #   7145
(U) PI.TCN             1143 #   7143
(U) PI.TSF             1145 #   7141
(U) PI.TSN             1146 #   7139
(U) PI.ZER             1121 #
(U) PXCT               1086 #
        BIS-DST-EA     1092 #   6702    6704    6710    6711    6713
        BIS-SRC-EA     1090 #
        CURRENT        1087 #   2135    2136    2138    2139    2158    2198    2204    2312    2337    2927    3233
                       3245     3360    3361    3364    3365    3432    3433    3457    3459    3481    3496    3501
                       3616     4117    4592    4601    7149    7420    7461    7651
        D1             1089 #   2228    2270    2322    2325    2693    2697    3638    3643    5019    5025    5039
                       5044
        D2             1093 #   3590    3627    3659    4650    4654    4998    5062    6708    6709
        E1             1088 #   2237    2242    2248    3477    3563
        E2             1091 #   4599    4618    4658    4662    4666    5734    5743    5744    5746    6665
(U) RAMADR              625 #
        AC#            626 #   2264    2277    2291    2428    2495    2596    2598    2629    2635    2721    2731
                       2763     2773    2793    2803    2813    2844    2872    2875    2889    2895    2907    2910
                       2923     2985    2997    3018    3183    3198    3201    3206    3207    3277    3296    3336
                       3339     3342    3345    3348    3351    3354    3357    3380    3394    3408    3416    3586
                       3617     3619    3624    3657    3679    3682    3687    3715    3735    3747    3956    3970
                       3986     3997    4000    4017    4018    4044    4046    4093    4108    4215    4216    4225
                       4306     4308    4350    4352    4523    4524    4528    4606    4612    4616    4892    4964
                       4994     5006    5009    5071    5100    5133    5163    5165    5228    5229    5257    5263
                       5365     5369    5371    5375    5467    5518    5619    5620    5773    5864    5901    5903
                       5911     5937    5973    6075    6107    6109    6121    6130    6200    6267    6421    6494
                       6746     6770    7330    7341    7344    7358    7360    7529    7626    7990
```

Cross Reference Listing

AC*#	627 #	2080	2081	2088	2278	2290	2396	2679	2688	2929	2950	2994
	3024	3983	3995	4074	4076	4078	4086	4088	4095	4097	4109	4110
	4111	4112	4113	4114	4226	4310	4332	4336	4339	4353	4354	4356
	4359	4378	4380	4382	4383	4509	4510	4513	4514	4515	4518	4519
	4520	4526	4527	5364	5455	5508	5510	5516	5575	5583	5621	5628
	5770	5778	5788	5791	5821	5824	5837	5844	5851	5861	5899	5909
	5929	5944	5961	5976	5978	6004	6007	6009	6016	6017	6018	6020
	6026	6028	6031	6037	6039	6067	6069	6070	6074	6081	6082	6083
	6087	6088	6095	6097	6102	6108	6124	6135	6147	6156	6162	6173
	6176	6178	6182	6195	6201	6202	6325	6356	6361	6366	6403	6405
	6509	6510	6622	6629	6636	6655	6657	6664	6684	6705	6741	6743
	6749	6751	6755	6758	6763							
RAM	630 #	2090	2092	6238	6243	6248	6250					
VMA	629 #	2068	2176	2246	2260	2269	2273	2301	2312	2318	2331	2402
	2434	2440	2445	2448	2498	2694	3313	3456	3480	3495	3516	3520
	3524	3585	3602	3608	3630	3647	3653	3666	3724	3752	3868	3870
	3875	3902	3914	3918	3940	4636	4664	4670	4791	5027	5050	5053
	5727	5746	6166	6360	6406	6409	6437	6713	6728	6798	6915	7019
	7023	7027	7031	7035	7075	7105	7107	7114	7168	7236	7241	7252
	7289	7294	7385	7418	7465	7510	7513	7515	7525	7527	7619	7633
	7638	7658	7706	7723	7724	7726	7727	7730	7731	7733	7734	7736
	7737	7739	7740	7742	7743	7745	7746	7748	8009	8033	8090	8163
	8349	8398	8456	8462	8513	8520	8532	8534				
XR#	628 #	2186	2196	2208	2223	2236	2252	3461	3463	3471	4643	4646
	4648	4656	5738	5740	5743	6708	6710	6714	7454	7455	7463	
#	631 #	2067	2075	2077	2078	2083	2084	3570	4589	4848	4861	4866
	4871	4875	4876	4881	4887	4913	4926	4934	4939	4945	5069	5737
	5750	5753	5783	5800	5807	5811	5828	5840	5845	5856	5860	5871
	5872	5873	5875	5876	5877	5878	5888	5895	5919	5935	5942	5947
	5948	5955	5958	5971	5989	5993	5996	6000	6110	6112	6153	6163
	6168	6169	6171	6174	6187	6189	6193	6196	6197	6218	6368	6369
	6392	6394	6430	6435	6445	6507	6513	6540	6544	6547	6552	6553
	6560	6627	6634	6638	6640	6729	6745	6754	6760	6768	6769	6772
	6780	6860	6882	6894	6962	6967	7006	7008	7010	7012	7014	7020
	7024	7028	7032	7036	7043	7050	7052	7057	7061	7063	7082	7085
	7095	7096	7099	7115	7116	7120	7200	7201	7449	7452	7457	7469
	7471	7483	7489	7491	7631	7703	7705	7710	7714	7718	7728	7729
	7747	7749	7750	7752	7986	7987	7996	7997	7999	8000	8007	8017
	8019	8021	8023	8027	8028	8068	8071	8098	8100	8129	8146	8149
	8152	8154	8168	8174	8208	8212	8214	8245	8285	8286	8287	8293
	8300	8304	8441	8452	8458	8467	8471	8483				
(D) READ	1256 #	2454	2455	2457	2459	2460	2462	2464	2465	2467	2469	2470
	2472	2491	2507	2508	2509	2510	2512	2513	2514	2515	2517	2518
	2520	2522	2523	2525	2527	2528	2530	2532	2533	2535	2537	2538
	2540	2542	2543	2545	2549	2550	2551	2552	2554	2555	2556	2557
	2559	2560	2562	2564	2565	2567	2569	2570	2572	2574	2575	2577
	2579	2580	2582	2584	2585	2587	2673	2674	2704	2705	2714	2715
	2716	2717	2724	2725	2726	2727	2734	2735	2736	2737	2739	2740
	2741	2742	2749	2756	2757	2758	2759	2766	2767	2768	2769	2776
	2777	2778	2779	2786	2787	2788	2789	2796	2797	2806	2807	2808
	2809	2817	2818	2819	2820	2827	2828	2829	2830	2837	2838	2839
	2840	2847	2848	2859	2860	2861	2863	2864	2865	3093	3094	3095
	3096	3097	3098	3109	3110	3111	3112	3113	3114	3115	3116	3127
	3128	3129	3130	3131	3132	3133	3134	3144	3145	3146	3147	3148
	3149	3150	3151	3266	3267	3268	3269	3270	3271	3272	3273	3283

Cross Reference Listing

		3284	3285	3286	3287	3288	3289	3290	3300	3301	3302	3303	3304
		3305	3306	3307	3316	3317	3318	3319	3320	3321	3322	3323	3383
		3397	3554	3577	3673	3949	3950	3951	3952	3963	3964	3965	3966
		3978	3979	4011	4012	4013	4014	4036	4037	4038	4039	4062	4203
		4204	4205	4206	4208	4209	4210	4211	4280	4581	4582	4583	4584
		4585	5077	5078	5079	5080	5081	5082	5083	5085	5086	5087	5088
		5089	5090	5091	5122	5123	5124	5126	5127	5128	5129	5151	5152
		5153	5155	5156	5157	5158	5201	5236	5237	5351	5352	5440	5501
		5723											
(U)	READ CYCLE	1074 #	2135	2136	2138	2139	2158	2198	2204	2209	2215	2238	2242
		2271	2312	2323	2326	2337	2927	3233	3245	3360	3361	3364	3365
		3432	3433	3434	3439	3443	3446	3459	3473	3476	3501	3523	3543
		3616	3626	3660	3750	3900	3943	4117	4592	4601	4649	4653	4657
		4661	4997	5063	5743	5744	5768	5915	6165	6192	6265	6270	6326
		6446	6562	6708	6709	6710	6711	6786	6794	6913	7018	7022	7026
		7030	7034	7073	7077	7113	7149	7237	7251	7284	7293	7302	7376
		7379	7420	7443	7464	7616	7651	7653	8074	8082	8087	8158	8268
		8345	8377	8391	8536								
(D)	ROUND	1244 #	5080	5081	5082	5083	5088	5089	5090	5091	5126	5127	5128
		5129	5155	5156	5157	5158	5201	5237					
(U)	RSRC	559 #											
	AB	561 #	3570	7449									
	AQ	560 #											
	DO	567 #	2666	5970	6085	7237	7284	7376	7379	7415	7426	7432	7505
		7653	8013	8015	8029								
	DA	565 #	2062	2065	2098	2109	2145	2149	2152	2155	2164	2167	2170
		2173	2424	2949	3487	3850	3883	3930	4294	4305	4699	5695	5697
		5699	5701	5703	5705	5707	5709	5711	6395	6407	6433	6473	6618
		6843	6846	6904	7046	7089	7090	7091	7225	7226	7227	7228	7229
		7230	7231	7233	7266	7298	8055	8125	8176	8193	8274	8466	
	DQ	566 #											
	OA	564 #	2227	2668	3596	3713	3719	3915	4115	5368	5370	5470	5471
		5521	5523	5741	6779	7268	8518						
	OB	563 #	2645	2647									
	OQ	562 #	3535	3549	3861	3886	5730	5774	5974	7256	8517		
(U)	S#	949 #	2082	2285	2304	2304	2319	2319	2335	2873	2876	2880	2890
		2892	2908	2911	2914	2919	2931	2946	2948	2975	2992	3001	3004
		3047	4018	4046	4083	4103	4123	4236	4333	4341	4603	4685	4708
		4713	4715	4717	4720	4783	4784	4785	4786	4787	4788	4796	4800
		4802	4804	4837	4853	4864	4877	4893	4938	4953	5104	5106	5138
		5143	5145	5171	5172	5185	5186	5208	5212	5214	5215	5219	5227
		5242	5245	5247	5298	5302	5304	5307	5310	5318	5320	5321	5322
		5323	5341	5346	5376	5387	5402	5404	5445	5456	5460	5463	5473
		5479	5485	5496	5534	5535	5545	5559	5566	5569	5581	5598	5603
		5605	5626	5651	5653	5659	5795	5796	5798	5964	6109	6277	6279
		6286	6335	6340	6342	6465	6467	6469	6470	6476	6486	6669	6939
		6957	6975	6985	6992	7204	7392	7424	7478	7494	8050	8093	8133
		8243											
(U)	SCAD	926 #											
	A	934 #	2082	2931	4018	4046	4083	4103	4123	4236	4333	4341	4599
		4613	4690	4691	4692	4693	4694	4799	4837	4853	4864	4877	4893
		4938	4953	5138	5185	5208	5212	5214	5215	5219	5445	5456	5460
		5473	5479	5534	5535	6109	6277	6279	6286	6335	6465	6466	6476
		6666	6687	6706	6939	6957	6975	6985	6992	7204	7392	7424	7478
		7494	8050	8093	8133	8243							

Cross Reference Listing


```
        A*2                 927 #   6458
        A+B                 931 #   2285    2304    2304    2319    2319    2876    2880    2890    2892    2911    2914
                            2919    2946    2948    2992    3001    4715    4720    4784    4785    4786    4787    4788
                            4796    4804    5104    5106    5134    5143    5145    5186    5227    5229    5241    5245
                            5298    5302    5304    5307    5310    5318    5320    5321    5322    5323    5341    5346
                            5376    5387    5402    5404    5468    5485    5496    5559    5566    5569    5581    5598
                            5603    5605    5626    5651    5653    5659    5795    5796    5798    6336    6340    6468
                            6469    6470    6486    6774
        A-1                 933 #   2092    2896    2898    2974    2976    2978    2982    3008    3015    3046    3048
                            3050    3054    4149    4161    4167    4175    4185    4193    4448    4451    4491    4840
                            4859    4959    5114    5116    5222    5246    5249    5252    5256    5420    5423    5430
                            5448    6138    6282    6288    6483    6943    6959    6977    6994    7184    7197    7399
                            7430    7485    7498    8053    8107    8110    8136    8249
        A-B                 930 #   2873    2908    2975    3004    3047    4591    4596    4610    4631    4685    4708
                            4713    4717    4783    4802    5165    5171    5172    5247    5519    5545    5848    5858
                            6658    6661    6701    6703
        A-B-1               929 #   5100    5105    5366    5386
        A.AND.B             932 #   4603    4800    5964    6342    6467    6669
        A.OR.B              928 #   2335    5463
(U) SCADA                   935 #
        BYTE1               939 #   4591    4596    4599    4610    4690    4799    5848    6336    6466    6658    6666
                            6701    6706    6774
        BYTE2               940 #   4691
        BYTE3               941 #   4692
        BYTE4               942 #   4693
        BYTE5               943 #   4614    4694    6459    6688
        PTR44               938 #   4631    5858    6661    6703
        S#                  937 #   2082    2285    2304    2304    2319    2319    2335    2873    2876    2880    2890
                            2892    2908    2911    2914    2919    2931    2946    2948    2975    2992    3001    3004
                            3047    4018    4046    4083    4103    4123    4236    4333    4341    4603    4685    4708
                            4713    4715    4717    4720    4783    4784    4785    4786    4787    4788    4796    4800
                            4802    4804    4837    4853    4864    4877    4893    4938    4953    5104    5106    5138
                            5143    5145    5171    5172    5185    5186    5208    5212    5214    5215    5219    5227
                            5241    5245    5247    5298    5302    5304    5307    5310    5318    5320    5321    5322
                            5323    5341    5346    5376    5387    5402    5404    5445    5456    5460    5463    5473
                            5479    5485    5496    5534    5535    5545    5559    5566    5569    5581    5598    5603
                            5605    5626    5651    5653    5659    5795    5796    5798    5964    6109    6277    6279
                            6286    6335    6340    6342    6465    6467    6469    6470    6476    6486    6669    6939
                            6957    6975    6985    6992    7204    7392    7424    7478    7494    8050    8093    8133
                            8243
        SC                  936 #   2092    2896    2898    2974    2976    2978    2982    3008    3015    3046    3048
                            3050    3054    4149    4161    4167    4175    4185    4193    4448    4451    4491    4840
                            4859    4959    5100    5105    5114    5116    5134    5165    5222    5229    5246    5249
                            5252    5256    5366    5386    5420    5423    5430    5448    5468    5519    6138    6282
                            6288    6468    6483    6943    6959    6977    6994    7184    7197    7399    7430    7485
                            7498    8053    8107    8110    8136    8249
(U) SCADB                   944 #
        EXP                 946 #   2304    2304    2319    2319    5100    5104    5106    5134    5165    5229    5366
                            5376    5387    5468    5519
        FE                  945 #   2335    2873    2876    2880    2890    2892    2908    2911    2914    2919    2946
                            2948    3001    3004    4603    4685    4708    4713    4715    4720    4784    4785    4786
                            4787    4788    4796    4800    4802    4804    5105    5143    5145    5171    5172    5186
                            5242    5245    5247    5298    5302    5304    5307    5310    5318    5320    5321    5322
                            5323    5341    5346    5386    5402    5404    5463    5485    5496    5545    5559    5566
                            5569    5581    5598    5603    5605    5626    5651    5653    5659    5796    5798    5964
```

Cross Reference Listing

	6336	6340	6342	6467	6468	6469	6470	6486	6669			
SHIFT	947 #	2285	2975	2992	3047	5227						
SIZE	948 #	4591	4596	4610	4631	4717	4783	5795	5848	5858	6658	6661
	6701	6703	6774									
(U) SETFOV	1030 #	5167	5179	5532								
(U) SETFPD	1036 #	4131	4645	6204								
(U) SETNDV	1031 #	4238	4273	4304	4390	4897	5167	5179	5532			
(U) SETOV	1028 #	4031	4032	4056	4118	4238	4273	4304	4390	4897	5167	5179
	5244	5532										
(U) SHSTYLE	773 #											
ASHC	778 #	2898	3007	3010	3014	3017	4167	4175	4185	4193	4299	4303
	4372	4376	4392	5116	5219	5221	5251	5395	5397	5401	5403	5419
	5423	5430	5433	5435	5447	5488	5492	5527	5541	5559	5565	5568
	5581	5598	5602	5604	5626	5651						
DIV	780 #	4448	4451	4452	4495	4499	5298	5310	5318	5322	5415	5417
	5418											
LSHC	779 #	2978	2982	4480	4483							
NORM	774 #	2332	2892	4126	4160	4333	4489	5449	5539			
ONES	776 #	4719	4722	4795	4798	5798						
ROT	777 #	2914	2919									
ROTC	781 #	3050	3054									
ZERO	775 #											
(U) SKIP	851 #											
ACO	859 #	2391	3295	4588								
ADEQO	871 #	2924	3239	3251	3342	3354	4021	4026	4228	4230	4234	4248
	4293	4322	4352	4433	4511	4516	4521	4537	4544	4862	4891	5163
	5189	5191	5264	5306	5330	5486	5537	5579	5624	5952	6005	6081
	6084	6373	6410	6431	6554	7093	8069	8072	8147	8150		
ADLEQO	855 #	2095	3887	3892	5206	5211	5325	5490	5493	5561	5564	5567
	5570	5630	5728	5771	5910	5912	5994	6014	6198	6214	6268	6303
	6306	6338	6802	6921	6928	6964	6970	7254	7255	7446	7448	7462
	8037	8057	8095	8102	8105	8119	8124	8165	8183	8187	8192	8201
	8205	8207	8218	8258	8277	8279	8302	8306	8320	8323	8325	8328
	8331	8351	8361	8410	8428	8469	8516					
ADREQO	856 #	2993	3499	3514	5015	6139	6212	6567	6844	6847	6867	6869
	6872	6874	6876	7129	7131	7135	7137	7139	7141	7143	7145	7242
	7295	7315	7387	7414	7988	8223	8240	8399	8414	8417	8422	
CRY0	854 #	3592	3635	3664	5175	5299	5301	5303	5305			
CRY1	867 #	3984	3996	5615	6029	6035	6048	6239	6250			
CRY2	862 #	5588	5648									
DPO	863 #	2305	2320	2479	2657	2664	2947	3019	3236	3248	3339	3351
	3681	3684	3689	3855	4003	4028	4050	4053	4108	4116	4217	4253
	4255	4260	4265	4269	4291	4308	4320	4327	4342	4361	4379	4431
	4438	4453	4894	5032	5056	5102	5106	5135	5165	5173	5207	5229
	5260	5367	5469	5516	5520	5531	5544	5779	5792	5809	5822	5829
	5847	5889	5891	5913	5930	5933	5939	5979	6071	6076	6113	6122
	6155	6172	6244	6294	6320	6511	6541	6583	6637	6756	6916	7048
	8060	8145	8319	8369								
DP18	864 #	2284	2643	2650	5752	5999	6203	6292	6615	6616	7281	8372
	8382	8404										
EXECUTE	873 #	7645										
FPD	858 #	4070	4591	4596	4610	6103						
INT	860 #	5036	5849	8001	8178	8347	8450	8481	8524			
IOLGL	852 #	2341	3444	3447								
IOT	865 #											

Cross Reference Listing


```
        JFCL                    866 #   3506
        KERNEL                  857 #   3437    3441    3442    3449    3558    6781
        LE                      861 #   3242    3254    3345    3357    4287    4921    5183    5507    6152    7059    7711
        LLE                     853 #   7455
        SC                      872 #   2092    2896    2898    2974    2976    2978    2982    2999    3008    3015    3046
                                3048    3050    3054    4149    4161    4167    4175    4185    4193    4448    4451    4491
                                4840    4859    4959    5114    5116    5222    5246    5249    5252    5256    5369    5371
                                5420    5423    5430    5448    6138    6282    6288    6474    6483    6943    6959    6977
                                6994    7184    7197    7399    7430    7498    8053    8107    8110    8136    8249
        TRAP CYCLE              869 #   8468
        TXXX                    868 #   3192
        -1 MS                   876 #   5028    5844    7097    8173
        -CONTINUE               875 #   7647
        -IO BUSY                874 #   7480    7486    7495    7503
(U) SPEC                        737 #   2476    2482    2596    2598    2631    2637    2645    2647    2666    2668    2711
                                2721    2731    2763    2773    2793    2803    2814    2824    2854    4023    4029    4052
                                4054    8264    8526
        APR EN                  751 #   2073    6881    6966
        APR FLAGS               749 #   6886    6890    7067
        ASHOV                   758 #   2898    3014    3017
        CLR IO BUSY             741 #   7373    7411
        CLR IO LATCH            740 #   7458    7466    7470    7473    7482    7488    7497    7502
        CLRCLK                  739 #   7044    7098    8213    8453
        CLRCSH                  750 #   7179    7180    7183
        EXPTST                  759 #   5307    5323    5053
        FLAGS                   760 #   2124    2125    2485    3311    3327    3394    3408    3464    3465    3468    3469
                                3505    3526    3527    3597    3609    3665    3695    3699    3714    3720    3904    3905
                                3906    3957    3971    3988    3998    4001    4031    4032    4056    4106    4118    4128
                                4131    4238    4273    4304    4390    4539    4541    4606    4623    4645    4897    5167
                                5179    5244    5532    6204    6205    6803    7257    7272    8472
        INHCRY18                755 #   3417    3587    3634    3663    3683    3688    4929    5007
        LDACBLK                 761 #   2070    6936    6953
        LDINST                  762 #   2177    3562    6799    7659
        LDPAGE                  742 #   7174    8280
        LDPI                    757 #   3515    7160    7235    7712
        LDPXCT                  744 #   3567
        LOADIR                  756 #   5733
        LOADXR                  747 #   2248    3457    3481    3496    4599    4618    4666    5734    5746    6665    6702
                                6704    6713    7461
        MEMCLR                  752 #   2068    3516    7236    7633    7706    8033    8456    8462
        NICOND                  743 #   2135    2136    2138    2139    2199    2205    2429    2950    2996    3024    3233
                                3245    3336    3360    3361    3364    3365    3432    3433    3459    3501    3618    3694
                                3701    4117    4381    4384    4592    5257    5263    5622    7149    7420    7990
        PREV                    746 #   7617    7625
        PXCT OFF                754 #   2147    2150    2153    2165    2168    2171    2185
        SWEEP                   753 #   7192    7193    7196
        WAIT                    745 #   2317
        #                       738 #   7641    7644    7707
(U) STATE                       952 #
        BLT                     954 #   5008
        COMP-DST                961 #   5938
        CVTDB                   960 #   5985
        DST                     957 #   5900
        DSTF                    959 #   5838
        EDIT-DST                963 #   5957    6167    6194    6413    6439    8489    8493
```

Cross Reference Listing

```
        EDIT-S+D        964 #   6379
        EDIT-SRC        962 #   5931    6318    6375    8487    8495    8497    8501
        MAP             955 #   7989
        SIMPLE          953 #
        SRC             956 #   5787    5816    5886    8491
        SRC+DST         958 #   5813    5892
(U) SWITCH%
        FULL            377     1202    1204    2079    2099    2104    2108    4065    4133    4136    4283    4393
                        4396    4464    4529    4815    4966    4970    6960    6968
        SIM             374     2117    2122    2129
(U) T                   889 #
        2T              892 #   4308    4321    5106    5376    5948    6018    6082    6370    6507    6552    7099
                        7456
        3T              893 #   2095    2320    2330    2485    3018    3192    3277    3311    3327    3417    3507
                        3591    3626    3634    3659    3663    3680    3685    3690    3727    3741    3957    3971
                        4074    4076    4095    4129    4266    4270    4320    4326    4327    4328    4345    4349
                        4370    4378    4459    4524    4541    4599    4614    4685    4703    4799    4843    4850
                        4930    4946    4950    5012    5014    5031    5055    5100    5134    5175    5206    5207
                        5210    5242    5245    5299    5301    5303    5305    5364    5365    5368    5370    5469
                        5470    5471    5515    5519    5521    5523    5531    5614    5620    5778    5796    5798
                        5846    5850    5913    6020    6034    6048    6080    6083    6168    6171    6203    6204
                        6232    6238    6241    6250    6275    6289    6316    6336    6337    6459    6466    6510
                        6569    6659    6666    6687    6701    6705    6708    6710    6746    6801    6915    6921
                        6927    6964    6970    7216    7241    7245    7253    7254    7255    7290    7294    7296
                        7463    8076    8084    8272    8289    8384    8395    8477    8507
        4T              894 #   2923    3394    3408    3983    3988    3995    3998    4000    4890    4895    5792
                        5822    5999    6004    6029    6070    6243    6291    6431    6554    6755    7711    8004
                        8069    8072    8147    8150    8372    8382
        5T              895 #   7485    7502
(D) TEST                1257 #  2456    2456    2457    2457    2461    2461    2462    2462    2466    2466    2467
                        2467    2471    2471    2472    2472    2491    2509    2509    2510    2510    2514    2514
                        2515    2515    2519    2519    2520    2520    2524    2524    2525    2525    2529    2529
                        2530    2530    2534    2534    2535    2535    2539    2539    2540    2540    2544    2544
                        2545    2545    2551    2551    2552    2552    2556    2556    2557    2557    2561    2561
                        2562    2562    2566    2566    2567    2567    2571    2571    2572    2572    2576    2576
                        2577    2577    2581    2581    2582    2582    2586    2586    2587    2587    2683    2706
                        2706    2707    2707    2716    2716    2717    2717    2726    2726    2727    2727    2736
                        2736    2737    2737    2741    2741    2742    2742    2751    2751    2752    2752    2758
                        2758    2759    2759    2768    2768    2769    2769    2778    2778    2779    2779    2788
                        2788    2789    2789    2798    2798    2799    2799    2808    2808    2809    2809    2819
                        2819    2820    2820    2829    2829    2830    2830    2839    2839    2840    2840    2849
                        2849    2850    2850    3300    3301    3302    3303    3304    3305    3306    3307    3316
                        3317    3318    3319    3320    3321    3322    3323    3951    3951    3952    3952    3965
                        3965    3966    3966    4013    4013    4014    4014    4038    4038    4039    4039    4205
                        4205    4206    4206    4210    4210    4211    4211    4582    4584    5078    5078    5079
                        5079    5082    5082    5083    5083    5086    5086    5087    5087    5090    5090    5091
                        5091    5123    5123    5124    5124    5128    5128    5129    5129    5152    5152    5153
                        5153    5157    5157    5158    5158    7001
(U) TRAP1               1045 #  4031    4032    4056    4118    4238    4273    4304    4390    4897    5167    5179
                        5244    5532
(U) TRAP2               1044 #  3609    3665    3695    3699
(U) VECTOR CYCLE        1116 #  7285
(D) VMA                 1259 #  2455    2460    2465    2470    2508    2513    2518    2523    2528    2533    2538
                        2543    2550    2555    2560    2565    2570    2575    2580    2585    2704    2705    2715
                        2725    2735    2740    2757    2767    2777    2787    2796    2797    2807    2818    2828
```

Cross Reference Listing

	2838	2847	2848	2859	2860	2861	2863	2864	2865	3383	3397	3423
	3673	3950	3964	4012	4037	4204	4209					
(U) WAIT	1105 #	2135	2135	2136	2136	2138	2138	2139	2139	2158	2158	2175
	2198	2198	2204	2204	2209	2215	2228	2238	2242	2246	2259	2268
	2271	2272	2300	2311	2312	2312	2323	2326	2330	2337	2337	2401
	2433	2439	2444	2447	2496	2496	2497	2692	2692	2694	2698	2698
	2927	2927	3233	3233	3245	3245	3312	3312	3313	3360	3360	3361
	3361	3364	3364	3365	3365	3432	3432	3433	3433	3434	3439	3443
	3446	3455	3459	3459	3473	3476	3479	3494	3501	3501	3520	3523
	3524	3537	3537	3540	3540	3543	3548	3548	3584	3589	3589	3601
	3607	3616	3616	3626	3629	3639	3639	3644	3644	3646	3652	3660
	3666	3722	3722	3723	3733	3733	3750	3751	3858	3866	3867	3869
	3874	3879	3879	3900	3901	3913	3917	3925	3925	3935	3935	3939
	3943	4117	4117	4592	4592	4601	4601	4629	4629	4635	4635	4636
	4649	4653	4657	4661	4663	4669	4790	4790	4791	4997	5018	5018
	5024	5024	5026	5040	5040	5045	5045	5049	5052	5063	5727	5743
	5744	5746	5768	5915	6165	6166	6192	6265	6270	6326	6358	6358
	6360	6404	6404	6406	6409	6437	6446	6562	6708	6709	6710	6711
	6713	6727	6783	6786	6791	6794	6798	6913	6914	7018	7019	7022
	7023	7026	7027	7030	7031	7034	7035	7073	7074	7077	7102	7102
	7105	7106	7106	7107	7113	7114	7149	7149	7167	7167	7168	7240
	7251	7252	7267	7267	7288	7293	7294	7302	7382	7384	7417	7418
	7420	7420	7443	7464	7465	7510	7513	7515	7523	7525	7527	7616
	7618	7624	7624	7636	7638	7639	7639	7651	7651	7657	7722	7723
	7724	7725	7725	7726	7727	7730	7731	7732	7732	7733	7734	7735
	7735	7736	7737	7738	7738	7739	7740	7741	7741	7742	7743	7744
	7744	7745	7746	7748	8074	8082	8087	8089	8131	8158	8162	8235
	8290	8345	8348	8377	8391	8397	8511	8512	8519	8532	8534	8536
	8539	8539										
(U) WORK	968 #											
ACO	1009 #											
AC1	1010 #											
AC2	1011 #											
AC3	1012 #											
ADJBPW	987 #	4913	4934									
ADJP	982 #	4848	4875									
ADJPTR	984 #	4589	4866	4928								
ADJQ1	985 #	4871	4889									
ADJR2	986 #	4881	4948									
ADJS	983 #	4861	4876	4939								
APR	989 #	2075	6860	6882	6894	6962	6967	7987				
BADWO	969 #	8027										
BADW1	970 #	8028										
BDH	999 #	6112	6187	6196	6218							
BDL	1000 #	6110	6153	6193	6197							
CBR	979 #	7008	7024	8300								
CMS	996 #	5935	5947	5948	5958							
CSTM	980 #	7012	7032	8304								
DDIV SGN	1013 #											
DECHI	1018 #	2084	2086	6241								
DECLO	1017 #	2083	2085	6232								
DIV	972 #											
DVSOR H	1014 #											
DVSOR L	1015 #											
EO	991 #	5737	6163	6435	6445							

;
Cross Reference Listing


```
        E1                      992 #   5750    5753    6189    6368    6369    6544    6547    6552    6560
        FILL                    995 #   5919    5942    5955    6430    6507    6513    6627    6729
        FSIG                    997 #   6392    6394
        HSBADR                  988 #   2067    7014    7036    7705    7710
        MSK                     994 #   5800    5971    6553
        MUL                     971 #
        PERIOD                  1005 #  7063    7115    7120
        PTA.E                   1021 #  7201    8068    8146    8152
        PTA.U                   1022 #  7200    8071    8149    8154
        PUR                     981 #   7010    7028    8129    8208
        SBR                     978 #   7006    7020    8098    8100    8168
        SLEN                    993 #   5783    5807    5811    5828    5840    5856    5860    5888    5895    5989    5993
                                5996    6000    6168    6169    6171    6174    6540    6634    6638    6640    6754    6768
                                6772
        SV.ARX                  975 #   5872    5877    7631    7703    7714    7729    7752    8000    8285    8483
        SV.AR                   974 #   5845    5875    7728    7749    7996    8293
        SV.AR1                  1024 #  8174    8214
        SV.BRX                  977 #   5069    5873    5878    6745    6760    6769    7997    8287
        SV.BR                   976 #   5871    5876    8007    8017    8019    8021    8023    8286    8441
        SV.VMA                  973 #   7483    7489    7491    7718    7747    7750    7986    7999    8245    8458    8467
        TIMEO                   1003 #  2077    7052    7085    7095
        TIME1                   1004 #  2078    7043    7050    7082    7096    7099    8212    8452
        TRAPPC                  1023 #  6780    8471
        TTG                     1006 #  7057    7061    7116
        YSAVE                   1020 #  3570    7449    7452    7457    7469    7471
(D) WRITE                       1260 #
(U) WRITE CYCLE                 1076 #  2476    2482    2496    2596    2598    2631    2637    2645    2647    2666    2668
                                2692    2698    2711    2721    2731    2763    2773    2793    2803    2814    2824    2854
                                3312    3537    3540    3548    3589    3639    3644    3722    3733    3858    3866    3879
                                3925    3935    3957    3971    4023    4029    4052    4054    4629    4635    4790    5018
                                5024    5040    5045    5261    5265    5311    5333    6358    6404    7102    7106    7167
                                7267    7415    7426    7432    7624    7636    7639    7722    7725    7732    7735    7738
                                7741    7744    8131    8206    8218    8235    8264    8269    8429    8511    8526    8539
(U) WRITE TEST                  1075 #  2476    2482    2496    2596    2598    2631    2637    2645    2647    2666    2668
                                2692    2698    2711    2721    2731    2763    2773    2793    2803    2814    2824    2854
                                3312    3537    3540    3548    3589    3639    3644    3722    3733    3858    3866    3879
                                3925    3935    3957    3971    4023    4029    4052    4054    4629    4635    4790    5018
                                5024    5040    5045    5261    5265    5311    5333    6358    6404    7102    7106    7167
                                7237    7251    7267    7284    7293    7376    7379    7415    7426    7432    7624    7636
                                7639    7653    7722    7725    7732    7735    7738    7741    7744    8037    8074    8082
                                8206    8264    8270    8511    8526    8536    8539
(U) WRU CYCLE                   1112 #  7238
(U) #                           923 #   2056    2058    2060    2062    2065    2095    2109    2145    2149    2152    2164
                                2167    2170    2327    2329    2342    2364    2365    2652    2654    2659    2661    2678
                                2949    3020    3022    3436    3445    3448    3450    3451    3452    3485    3490    3493
                                3499    3513    3514    3534    3547    3633    3662    3725    3740    3826    3828    3830
                                3832    3834    3836    3838    3840    3847    3850    3883    3888    3890    3893    3895
                                3922    3930    3937    4004    4005    4257    4294    4295    4305    4313    4358    4411
                                4437    4444    4535    4700    4844    4846    4851    4869    4873    4879    4883    4912
                                4917    4951    4955    4962    5011    5107    5108    5111    5112    5136    5137    5141
                                5169    5170    5206    5210    5230    5231    5368    5370    5450    5470    5471    5490
                                5493    5521    5523    5695    5697    5699    5701    5703    5705    5707    5709    5711
                                5728    5731    5755    5771    5775    5910    5912    5970    5975    5982    5987    5994
                                5998    6014    6056    6072    6073    6079    6085    6105    6118    6120    6127    6132
                                6139    6142    6198    6212    6214    6221    6246    6252    6268    6280    6283    6291
```

Cross Reference Listing

6303	6306	6323	6338	6396	6407	6416	6418	6434	6453	6474	6479
6488	6491	6567	6589	6593	6597	6601	6605	6609	6619	6625	6723
6734	6801	6806	6815	6816	6817	6821	6822	6823	6824	6829	6830
6831	6834	6835	6836	6837	6838	6839	6840	6841	6843	6846	6852
6862	6863	6866	6867	6869	6872	6874	6876	6888	6898	6901	6905
6919	6924	6927	6931	6946	6963	6964	6965	6970	6971	6972	6978
6987	6995	7015	7037	7046	7065	7080	7132	7133	7176	7178	7189
7191	7218	7219	7220	7221	7222	7223	7224	7225	7226	7227	7228
7229	7230	7231	7234	7245	7254	7255	7270	7290	7297	7387	7395
7401	7414	7446	7448	7462	7505	7591	7593	7595	7597	7599	7601
7603	7605	7641	7644	7655	7707	7981	7985	7988	8013	8015	8029
8039	8042	8048	8056	8058	8077	8085	8096	8103	8119	8126	8166
8177	8180	8188	8194	8200	8203	8220	8224	8227	8241	8252	8256
8260	8262	8275	8276	8278	8283	8303	8315	8321	8326	8332	8371
8381	8385	8395	8400	8414	8417	8420	8423	8432	8437	8445	8507
8516											

;
; Location / Line Number Index

; Dcode Loc'n	0	1	2	3	4	5	6	7
D 0000	3812	5666	5667	5668	5669	5670	5671	5672
D 0010	5674	5675	5676	5677	5679	5680	5681	5682
D 0020	5684	5685	5686	5687	5688	5689	5690	5691
D 0030	3759	3760	3761	3762	3763	3764	3765	3766
D 0040	3770	3771	3772	3773	3774	3775	3776	3777
D 0050	3778	3779	3780	3781	3782	3783	3784	3785
D 0060	3786	3787	3788	3789	3790	3791	3792	3793
D 0070	3794	3795	3796	3797	3798	3799	3800	3801
D 0100	3805	3806	3807	3808	3813	3673	3814	3815
D 0110	5351	5352	5440	5501	3978	3979	4062	4280
D 0120	2673	2674	5236	5723	2682	2683	5237	5201
D 0130	3816	3817	5202	4581	4582	4583	4584	4585
D 0140	5077	3818	5078	5079	5080	5081	5082	5083
D 0150	5085	3819	5086	5087	5088	5089	5090	5091
D 0160	5122	3820	5123	5124	5126	5127	5128	5129
D 0170	5151	3821	5152	5153	5155	5156	5157	5158
D 0200	2454	2455	2456	2457	2459	2460	2461	2462
D 0210	2464	2465	2466	2467	2469	2470	2471	2472
D 0220	4011	4012	4013	4014	4036	4037	4038	4039
D 0230	4203	4204	4205	4206	4208	4209	4210	4211
D 0240	2859	2860	2861	2862	2863	2864	2865	3822
D 0250	2491	4990	3411	3412	3423	3425	3554	7976
D 0260	3576	3577	3578	3579	3706	3707	3708	3709
D 0270	3949	3950	3951	3952	3963	3964	3965	3966
D 0300	3257	3258	3259	3260	3261	3262	3263	3264
D 0310	3266	3267	3268	3269	3270	3271	3272	3273
D 0320	3369	3370	3371	3372	3373	3374	3375	3376
D 0330	3283	3284	3285	3286	3287	3288	3289	3290
D 0340	3383	3384	3385	3386	3387	3388	3389	3390
D 0350	3300	3301	3302	3303	3304	3305	3306	3307
D 0360	3397	3398	3399	3400	3401	3402	3403	3404
D 0370	3316	3317	3318	3319	3320	3321	3322	3323
D 0400	2704	2705	2706	2707	2714	2715	2716	2717
D 0410	2724	2725	2726	2727	2734	2735	2736	2737
D 0420	2739	2740	2741	2742	2749	2750	2751	2752
D 0430	2756	2757	2758	2759	2766	2767	2768	2769
D 0440	2776	2777	2778	2779	2786	2787	2788	2789
D 0450	2796	2797	2798	2799	2806	2807	2808	2809
D 0460	2817	2818	2819	2820	2827	2828	2829	2830
D 0470	2837	2838	2839	2840	2847	2848	2849	2850
D 0500	2507	2508	2509	2510	2512	2513	2514	2515
D 0510	2517	2518	2519	2520	2522	2523	2524	2525
D 0520	2527	2528	2529	2530	2532	2533	2534	2535
D 0530	2537	2538	2539	2540	2542	2543	2544	2545
D 0540	2549	2550	2551	2552	2554	2555	2556	2557
D 0550	2559	2560	2561	2562	2564	2565	2566	2567
D 0560	2569	2570	2571	2572	2574	2575	2576	2577
D 0570	2579	2580	2581	2582	2584	2585	2586	2587

;
; Dcode Loc'n Location / Line Number Index

Dcode Loc'n	0	1	2	3	4	5	6	7
D 0600	3082	3083	3084	3085	3086	3087	3088	3089
D 0610	3091	3092	3093	3094	3095	3096	3097	3098
D 0620	3100	3101	3102	3103	3104	3105	3106	3107
D 0630	3109	3110	3111	3112	3113	3114	3115	3116
D 0640	3118	3119	3120	3121	3122	3123	3124	3125
D 0650	3127	3128	3129	3130	3131	3132	3133	3134
D 0660	3135	3136	3137	3138	3139	3140	3141	3142
D 0670	3144	3145	3146	3147	3148	3149	3150	3151
D 0700	6811	6812	7001	7534	7610	7611	7535	7536
D 0710	7322	7323	7333	7334	7347	7348	7538	7539
D 0720	7324	7325	7335	7336	7349	7350	7541	7542
D 0730	7544	7545	7546	7547	7548	7549	7550	7551
D 0740	7553	7554	7555	7556	7557	7558	7559	7560
D 0750	7562	7563	7564	7565	7566	7567	7568	7569
D 0760	7571	7572	7573	7574	7575	7576	7577	7578
D 0770	7580	7581	7582	7583	7584	7585	7586	7587

;
; Location / Line Number Index

Ucode Loc'n	0	1	2	3	4	5	6	7
U 0000	2056:	7704:	2057	2065:	7645:	7647:	6349=	6350=
U 0010	2089=	7705:	2090=	2058	3192=	3195=	3198=	3201=
U 0020	3872=	4020=	3877=	3880=	7706=	4021=	7707=	3883=
U 0030	3466=	4048=	3470=	4389=	3474=	4049=	3478=	4390=
U 0040	2261:	2265:	2269:	2277:	2287:	2290:	2297:	2301:
U 0050	2314:	2320:	2341:	2059	4311=	4313=	4347=	4348=
U 0060	3538=	2060	4941=	5260=	3541=	3543=	4948=	7378=
U 0070	4651=	2063	4655=	5261=	4659=	4315=	4662=	7379=
U 0100	2067	2148=	2151=	2154=	7632=	2155=	2068	2159=
U 0110	2070	2166=	2169=	2172=	7634=	2173=	7636=	2178=
U 0120	4075=	4077=	4168=	4172=	4078=	2073	4176=	4180=
U 0130	3736=	2075	5506=	5508=	3739=	3743=	2077	6857:
U 0140	2098=	2101=	4186=	4190=	5185=	5186=	4194=	4198=
U 0150	3458=	2109=	3459=	5416=	5188=	5190=	5192=	5193=
U 0160	4231=	4234=	2078	5140=	4237=	4238=	4240=	5142=
U 0170	5740=	4084=	5742=	5417=	5743=	4085=	5744=	5194=
U 0200	5599=	2199=	5601=	2205=	5603=	2211=	5605=	2216=
U 0210	5607=	2224=	2080	2228=	2081	2238=	5608=	2242=
U 0220	3616=	3618=	3620=	2082	2094=	2095=	2083	7414=
U 0230	6708=	6660=	6709=	6661=	6710=	2084	6711=	7415=
U 0240	4590=	4591=	2085	5487=	2086	4592=	6393=	6394=
U 0250	3233=	3236=	3239=	3242=	3245=	3248=	3251=	3254=
U 0260	2138=	2139=	5340=	5488=	4619=	4622=	5341=	4623=
U 0270	3336=	3339=	3342=	3345=	3348=	3351=	3354=	3357=
U 0300	3493=	3497=	5345=	3499=	7146=	7147=	5346=	3500=
U 0310	3914=	2087	3916=	3919=	3868=	7149=	3870=	3501=
U 0320	3513=	3514=	5649=	2091	2092	3515=	5651=	5325=
U 0330	4256=	4258=	6508=	4261=	2125	3517=	6509=	5327=
U 0340	4687=	4690=	4691=	2127	4692=	4693=	2186	4694=
U 0350	4629=	4630=	4631=	4632=	4333=	4373=	4334=	4374=
U 0360	4783=	4784=	4785=	2248	4786=	4787=	2253	4788=
U 0370	4413=	2271	2306=	2308=	4414=	4415=	4416=	4417=
U 0400	3935:	2274	2323=	2326=	2342=	2343=	2279	5343=
U 0410	4094=	4096=	2364=	2365=	4097=	2291	2293	5344=
U 0420	5298=	5299=	5300=	5301=	5302=	5303=	5304=	5305=
U 0430	5306=	2328	2329	5399=	2446=	2449=	5307=	5400=
U 0440	5318=	6136=	5319=	6138=	5320=	2332	5321=	6139=
U 0450	5322=	2333	2335	5433=	2898=	2902=	5323=	5434=
U 0460	5993=	5995=	5653=	5657=	5997=	5999=	5660=	2338
U 0470	7269=	2499	5423=	5425=	7271=	7273=	5426=	5427=
U 0500	5808=	5810=	2928=	2929=	5812=	5814=	2629	5816=
U 0510	6194=	2635	2694	5546=	2947=	2948=	6195=	5547=
U 0520	5560=	5561=	5563=	5564=	5566=	5567=	5569=	5570=
U 0530	5572=	2699	2814	6633=	2978=	2979=	5573=	6635=
U 0540	5889=	5891=	5893=	4104=	2982=	2983=	5896=	4106=
U 0550	4709=	4343=	4710=	4344=	2995=	2996=	5751=	5752=
U 0560	6027=	6030=	6186=	4127=	2880	6031=	6187=	4129=
U 0570	4912=	4914=	2893	4918=	5787=	5788=	5790=	5793=

;
; Location / Line Number Index

; Ucode Loc'n	0	1	2	3	4	5	6	7
U 0600	6033=	6036=	8111=	8112=	4605=	6037=	4607=	8113=
U 0610	5611=	2896	5613=	5615=	3002=	3004=	5985=	5988=
U 0620	6044=	6045=	6049=	5456=	3008=	3010=	6050=	5460=
U 0630	5837=	5839=	2912	5841=	3015=	3017=	6192=	6193=
U 0640	6173=	6175=	3021=	3023=	3050=	3051=	2914	6176=
U 0650	5855=	5857=	5859=	5860=	6277=	6278=	6279=	6280=
U 0660	4870=	6292=	4871=	6295=	4880=	6297=	4881=	6299=
U 0670	5102=	6301=	5103=	6304=	5244=	6307=	5245=	6309=
U 0700	6370=	2915	6374=	6376=	6377=	6380=	2917	6382=
U 0710	6317=	6319=	6321=	6324=	6325=	6327=	6546=	6547=
U 0720	5246=	6584=	5247=	6586=	5534=	6590=	5535=	6594=
U 0730	5461=	6598=	5473=	6602=	5462=	6606=	5474=	6610=
U 0740	6415=	6417=	3054=	3056=	3296=	3297=	6419=	2920
U 0750	5899=	5900=	2931	5901=	6623=	6626=	6628=	6630=
U 0760	6440=	6441=	3360=	3361=	3364=	3365=	6442=	2949
U 0770	7217=	7218=	7219=	7220=	7221=	7222=	7223=	7224=
U 1000	8094=	8097=	8099=	8101=	3485=	3486=	2950	8103=
U 1010	3490=	3491=	5480=	7359=	3534=	3535=	5485=	7361=
U 1020	5932=	5934=	5935=	2976	3547=	3548=	2984	7387=
U 1030	5941=	5943=	5944=	2985	3564=	3568=	2999	7388=
U 1040	8148=	8151=	8153=	8155=	3604=	3608=	3019	8159=
U 1050	6008=	6010=	6011=	3024	5538=	7456=	5539=	7458=
U 1060	8005=	8007=	5729=	8011=	3048	8013=	5731=	8015=
U 1070	8017=	8019=	8021=	8023=	6542=	6543=	6544=	3206
U 1100	8482=	8484=	8486=	8488=	8490=	8492=	8494=	8496=
U 1110	8498=	8499=	8500=	8502=	7078=	7081=	3207	7083=
U 1120	7100=	7102=	7103=	3312	7484=	7487=	7490=	7492=
U 1130	8164=	8167=	8169=	8173=	8192=	8195=	3313	8200=
U 1140	8454=	8455=	8456=	3461	3640=	3644=	3665=	3668=
U 1150	3686=	3691=	3694=	3696=	3700=	3701=	3853=	3855=
U 1160	3859=	3860=	3863=	3866=	3891=	3893=	3896=	3900=
U 1170	3989=	3990=	3999=	4001=	4004=	4005=	4022=	4023=
U 1200	4028=	4029=	4031=	4032=	4052=	4053=	4054=	4056=
U 1210	4109=	4110=	4117=	4118=	4219=	4221=	4246=	4249=
U 1220	4267=	4271=	4273=	4275=	4289=	4291=	4293=	4294=
U 1230	4303=	4304=	4319=	4320=	4322=	4323=	4326=	4327=
U 1240	4328=	4329=	4337=	4338=	4353=	4354=	4363=	4364=
U 1250	4381=	4382=	4432=	4433=	4434=	4435=	4441=	4442=
U 1260	4451=	4452=	4455=	4456=	4480=	4483=	4497=	4501=
U 1270	4513=	4514=	4518=	4519=	4523=	4524=	4539=	4541=
U 1300	4545=	4546=	4841=	4844=	4860=	4862=	4865=	4866=
U 1310	4896=	4897=	4931=	4933=	4960=	4963=	4999=	5001=
U 1320	5010=	5013=	5020=	5025=	5029=	5032=	5034=	5036=
U 1330	5041=	5046=	5058=	5059=	5107=	5108=	5111=	5112=
U 1340	5116=	5117=	5136=	5137=	5166=	5167=	5169=	5170=
U 1350	5175=	5176=	5178=	5179=	5182=	5183=	5207=	5208=
U 1360	5211=	5212=	5214=	5215=	5222=	5223=	5230=	5231=
U 1370	5252=	5253=	5256=	5257=	5263=	5264=	5265=	5267=

; Location / Line Number Index

; Ucode Loc'n	0	1	2	3	4	5	6	7
U 1400	2135:	2136:	2476:	2479:	2482:	2486:	2496:	2596:
U 1410	2598:	2623:	2625:	2628:	2631:	2634:	2637:	2643:
U 1420	2645:	2647:	2650:	2657:	2652:	2654:	2659:	2661:
U 1430	2664:	3311:	2666:	2668:	2678:	8400	2679:	3327:
U 1440	3380:	2711:	2721:	2731:	2746:	2763:	2773:	2783:
U 1450	2793:	2803:	2813:	2824:	2834:	2844:	2854:	3984:
U 1460	7340:	7344:	2924:	7341:	2974:	2975:	2993:	5727:
U 1470	3046:	3047:	3176:	3178:	3181:	3184:	3277:	3295:
U 1500	8414	8438	8441	8448	2392:	2397:	2403:	8450
U 1510	8459	8462	8472	8511	2424:	2429:	2435:	2441:
U 1520	3432:	3433:	3435:	3436:	3437:	3440:	3441:	3442:
U 1530	3444:	3445:	3447:	3448:	3449:	3450:	3451:	3452:
U 1540	3508:	3558:	3408:	3585:	3598:	3627:	3660:	3418:
U 1550	3713:	3678:	3720:	7982:	3733:	3747:	3848:	3930:
U 1560	3957:	3971:	8521	8524	8537	2688:	4067:	2693:
U 1570	5135:	4044:	8539		5163:		5097:	5100:
U 1600	4215:	4224:						
U 1610	4588:	3394:	2874:	2877:	7329:	3996:	5206:	7330:
U 1620	4596:	5227:	2888:	2890:	4600:	4601:	5243:	4285:
U 1630	4610:	5445:	2909:	2911:	4612:	5361:	5504:	5364:
U 1640	4994:	4017:			7355:			7357:
U 1650	7591:	7593:	7595:	7597:	7599:	7601:	7603:	7605:
U 1660	3838:	3826:	3828:	3830:	3832:	3840:	3834:	3836:
U 1670								
U 1700	6852:	6815:	6816:	6817:	6860:	6894:	6818:	6819:
U 1710	6821:	6822:	6823:	6824:	7129:	7126:	6825:	6826:
U 1720	6829:	6984:	7174:	6913:	6957:	6975:	6830:	6831:
U 1730	6834:	6835:	6836:	6837:	6838:	6839:	6840:	6841:
U 1740	5695:	5697:	5699:	5701:	5703:		5705:	5707:
U 1750	5709:	5711:			7617:	7625:		
U 1760	7006:	7008:	7010:	7012:	7089:	7121:	7014:	7015:
U 1770	7018:	7022:	7026:	7030:	7073:	7113:	7034:	7037:
U 2000	5310=	5311=	5314=	5316=	5329=	5330=	5331=	5332=
U 2010	5369=	5371=	5374=	5376=	5380=	5381=	5388=	5389=
U 2020	5430=	5431=	5448=	5449=	5470=	5471=	5491=	5492=
U 2030	5494=	5496=	5511=	5512=	5521=	5522=	5530=	5531=
U 2040	5532=	5533=	5549=	5550=	5582=	5583=	5587=	5588=
U 2050	5592=	5594=	5619=	5620=	5627=	5628=	5637=	5638=
U 2060	5754=	5756=	5772=	5910=	5776=	5779=	5773=	5911=
U 2070	5781=	5782=	5824=	5826=	5798=	5874=	5799=	5875=
U 2100	5830=	5831=	5845=	5847=	5850=	5852=	5863=	5865=
U 2110	5912=	6269=	5914=	5915=	5913=	6270=	5918=	5920=
U 2120	5928=	5929=	5955=	5956=	5962=	5965=	5980=	5983=
U 2130	5977=	6111=	5978=	6113=	6001=	6005=	6015=	6016=
U 2140	6019=	6021=	6055=	6056=	6068=	6069=	6072=	6073=
U 2150	6076=	6077=	6082=	6084=	6085=	6087=	6095=	6096=
U 2160	6106=	6107=	6116=	6118=	6120=	6237=	6122=	6239=
U 2170	6124=	6125=	6130=	6133=	6143=	6144=	6148=	6149=

;
; Location / Line Number Index

Ucode Loc'n	0	1	2	3	4	5	6	7
U 2200	6154=	6155=	6157=	6161=	6199=	6200=	6204=	6205=
U 2210	6211=	6212=	6213=	6214=	6215=	6216=	6233=	6234=
U 2220	6240=	6241=	6247=	6248=	6253=	6255=	6274=	6275=
U 2230	6282=	6283=	6288=	6289=	6315=	6316=	6340=	6343=
U 2240	6357=	6358=	6387=	6388=	6367=	6690=	6368=	6691=
U 2250	6408=	6410=	6432=	6434=	6436=	6437=	6454=	6456=
U 2260	6477=	6481=	6483=	6485=	6512=	6513=	6566=	6567=
U 2270	6571=	6573=	6617=	6619=	6639=	6641=	6656=	6657=
U 2300	6666=	6670=	6685=	6688=	6702=	7054=	6703=	7056=
U 2310	6742=	6743=	6750=	6751=	6758=	6759=	6761=	6762=
U 2320	6788=	6796=	6805=	6806=	6868=	6869=	6870=	6871=
U 2330	6873=	6874=	6875=	6876=	6878=	6879=	6922=	6925=
U 2340	6932=	6937=	6944=	6947=	6959=	6962=	6965=	6966=
U 2350	6971=	6972=	6977=	6978=	6985=	6988=	6994=	6995=
U 2360	7051=	7052=	7062=	7063=	7094=	7095=	7130=	7131=
U 2370	7132=	7133=	7136=	7137=	7138=	7139=	7140=	7141=
U 2400	7142=	7143=	7144=	7145=	7177=	7179=	7185=	7186=
U 2410	7190=	7192=	7198=	7200=	7239=	7242=	7243=	7244=
U 2420	7255=	7256=	7266=	7267=	7283=	7285=	7287=	7289=
U 2430	7291=	7293=	7297=	7298=	7312=	7313=	7316=	7317=
U 2440	7374=	7375=	7383=	7386=	7393=	7395=	7399=	7401=
U 2450	7412=	7413=	7419=	7420=	7425=	7427=	7431=	7433=
U 2460	7444=	7445=	7448=	7450=	7453=	7454=	7463=	7464=
U 2470	7470=	7471=	7498=	7499=	7504=	7505=	7640=	7641=
U 2500	7652=	7653=	7656=	7660=	7713=	7715=	7989=	7990=
U 2510	7723=	7726=	7724=	7727=	7728=	7730=	7729=	7731=
U 2520	7733=	7736=	7734=	7737=	7739=	7742=	7740=	7743=
U 2530	7745=	7747=	7746=	7748=	8040=	8043=	8054=	8056=
U 2540	8061=	8063=	8070=	8072=	8075=	8078=	8083=	8085=
U 2550	8105=	8107=	8118=	8132=	8124=	8126=	8119=	8133=
U 2560	8128=	8213=	8129=	8215=	8137=	8145=	8175=	8178=
U 2570	8182=	8183=	8186=	8236=	8204=	8206=	8188=	8241=
U 2600	8207=	8209=	8222=	8224=	8229=	8233=	8244=	8246=
U 2610	8250=	8253=	8260=	8262=	8278=	8280=	8283=	8284=
U 2620	8301=	3482	8305=	8306=	8303=	3487	8322=	8323=
U 2630	8327=	8328=	8333=	8334=	8337=	8338=	8350=	8351=
U 2640	8359=	8360=	8362=	8363=	8373=	8378=	8383=	8386=
U 2650	8393=	8396=	8405=	8407=	8410=	8413=	8416=	8417=
U 2660	8421=	8423=	8427=	8429=	8433=	8434=	8443=	8446=
U 2670	8466=	8468=	8470=	8471=	8474=	8475=	8477=	3520
U 2700	8508=	8509=	8514=	8516=	8517=	8518=	8526=	8527=
U 2710	8533=	8534=	3523	3525	3527	3529	3549	3571
U 2720	3593	3611	3630	3635	3649	3654	3664	3681
U 2730	3716	3722	3724	3728	3734	3748	3750	3753
U 2740	3850	3885	3886	3888	3902	3907	3923	3926
U 2750	3938	3940	3944	3992	4003	4018	4026	4045
U 2760	4046	4050	4068	4070	4087	4088	4099	4108
U 2770	4111	4112	4113	4114	4116	4122	4123	4124

Location / Line Number Index

Ucode Loc'n	0	1	2	3	4	5	6	7
U 3000	4131	4147	4150	4162	4217	4225	4226	4228
U 3010	4251	4253	4287	4296	4297	4300	4305	4306
U 3020	4308	4332	4335	4336	4339	4345	4351	4352
U 3030	4356	4358	4359	4361	4366	4367	4369	4370
U 3040	4376	4379	4384	4392	4431	4437	4438	4444
U 3050	4445	4446	4447	4448	4453	4457	4458	4461
U 3060	4491	4509	4511	4516	4521	4526	4527	4528
U 3070	4535	4537	4544	4615	4616	4635	4636	4644
U 3100	4646	4667	4671	4704	4713	4716	4717	4718
U 3110	4721	4722	4723	4790	4791	4794	4797	4798
U 3120	4799	4800	4802	4805	4808	4809	4810	4812
U 3130	4837	4847	4848	4851	4853	4855	4874	4875
U 3140	4876	4877	4883	4891	4921	4935	4938	4939
U 3150	4951	4953	4956	4965	4995	5002	5003	5005
U 3160	5007	5008	5015	5028	5051	5053	5056	5064
U 3170	5068	5070	5072	5104	5105	5106	5114	5138
U 3200	5143	5146	5171	5172	5173	5196	5219	5228
U 3210	5229	5249	5333	5362	5367	5382	5386	5387
U 3220	5390	5395	5397	5402	5404	5405	5418	5420
U 3230	5435	5450	5451	5455	5463	5469	5475	5489
U 3240	5490	5493	5513	5517	5520	5523	5524	5527
U 3250	5542	5576	5579	5589	5617	5622	5624	5630
U 3260	5733	5735	5737	5738	5746	5784	5795	5796
U 3270	5801	5822	5828	5829	5832	5844	5861	5868
U 3300	5871	5872	5873	5876	5877	5879	5885	5886
U 3310	5903	5904	5930	5937	5938	5939	5947	5948
U 3320	5949	5950	5951	5952	5957	5958	5971	5973
U 3330	5975	5989	6014	6017	6040	6058	6061	6063
U 3340	6071	6074	6079	6081	6088	6097	6108	6109
U 3350	6128	6150	6152	6162	6163	6165	6166	6167
U 3360	6168	6169	6172	6178	6183	6190	6196	6197
U 3370	6198	6201	6202	6203	6218	6222	6242	6244
U 3400	6249	6250	6271	6272	6286	6335	6336	6338
U 3410	6360	6361	6397	6403	6404	6405	6406	6424
U 3420	6445	6447	6457	6460	6465	6466	6467	6468
U 3430	6469	6471	6474	6486	6489	6491	6493	6494
U 3440	6511	6552	6555	6560	6562	6637	6659	6662
U 3450	6664	6683	6701	6704	6706	6713	6714	6728
U 3460	6730	6735	6745	6746	6754	6756	6764	6766
U 3470	6768	6769	6770	6772	6775	6779	6781	6799
U 3500	6802	6843	6844	6846	6847	6862	6863	6864
U 3510	6866	6867	6872	6881	6883	6886	6888	6891
U 3520	6896	6899	6902	6903	6905	6907	6916	6928
U 3530	6940	6949	6950	6954	6963	6964	6967	6969
U 3540	6970	6981	6991	6992	6996	7019	7020	7023
U 3550	7024	7027	7028	7031	7032	7035	6723:	6724:
U 3560	7036	7044	7046	7047	7048	7057	7059	7064
U 3570	7065	7068	7075	7086	7090	7091	7093	7097

;
; Location / Line Number Index

Ucode Loc'n	0	1	2	3	4	5	6	7
U 3600	7105	7106	7107	7114	7115	7117	7134	7135
U 3610	7157	7158	7159	7161	7167	7168	7175	7180
U 3620	7193	7202	7205	7225	7226	7227	7228	7229
U 3630	7230	7231	7234	7235	7236	7246	7248	7251
U 3640	7253	7254	7258	7281	7295	7299	7302	7315
U 3650	7364	7417	7446	7461	7462	7466	7473	7480
U 3660	7495	7511	7513	7515	7517	7519	7521	7523
U 3670	7525	7527	7529	7620	7627	7638	7711	7717
U 3700	7719	7722	7725	7732	7735	7738	7741	7744
U 3710	7749	7750	7751	7753	7983	7985	7986	7987
U 3720	7988	7997	7998	7999	8001	8027	8028	8029
U 3730	8033	8036	8037	8048	8050	8058	8088	8090
U 3740	5909:	6266:	5970:	6103:	5769:	8201	8218	8254
U 3750	8257	6267:	8258	8264	5770:	8266	8271	8273
U 3760	8275	8276	8277	8285	8286	8287	8290	8294
U 3770	8315	8317	8319	8347	8361	8369	8398	7996:

No errors detected
End of microcode assembly
311 pages of listing
Used 58.68 seconds, 117 pages of core
  Symbol table: 30P
  Text strings: 9P
  Loc'n assignment: 18P
  Cross reference: 52P