```
                                ;15746  =0      ;-------------------------------;ALU <Z>
                                ;15747  SKPST1: STATE_K[ZERO],                  ;SOMETHING TO DO. INITIALIZE STATE
                                ;15748          BEN/ROR,                        ;BRANCH ON BYTE OFFSET OF SRC ADDR
U 060C, 0000,023C,1980,F800,1404,67E2   ;15749          J/SKPST2                        ;
                                ;15750
                                ;15751          ;1------------------------------;ALU = 0
U 060D, 0018,0038,1980,FA80,0000,0A25   ;15752          J/SKPLOCEXIT,R[R0]_K[ZERO]      ;LENGTH = 0
                                ;15753
                                ;15754  =010    ;-------------------------------;LA <1:0>
                                ;15755  SKPST2:                                 ;LA<1:0> = 00
                                ;15756          Q_Q-K[.4] CLK.UBCC,             ;SEE IF A LWD'S WORTH
U 07E2, 0019,2000,11C0,F800,0010,0A16   ;15757          J/SKPALIGNED                    ;
                                ;15758
                                ;15759          ;011-----------------------------;LA<1:0> = 01
                                ;15760          D[BYTE]_CACHE,                  ;READ 1 BYTE
                                ;15761          STATE_STATE.AN.SKPLONG,         ;NOTE IT'S BYTE
                                ;15762          LC_RC[T2],                      ;LATCH COMPARE CHAR
                                ;15763          INTRPT.STROBE,                  ;INTERRUPTS PENDING?
U 07E3, 0000,803C,1180,4110,5404,47F2   ;15764          J/SKPBYTES                      ;
                                ;15765
                                ;15766          ;110-----------------------------;LA<1:0> = 10
                                ;15767          D[BYTE]_CACHE,                  ;READ 1 BYTE
                                ;15768          STATE_STATE.AN.SKPLONG,         ;NOTE IT'S BYTE
                                ;15769          LC_RC[T2],                      ;LATCH COMPARE CHAR
                                ;15770          INTRPT.STROBE,                  ;INTERRUPTS PENDING?
U 07E6, 0000,803C,1180,4110,5404,47F2   ;15771          J/SKPBYTES                      ;
                                ;15772
                                ;15773          ;111-----------------------------;LA<1:0> = 11
                                ;15774          D[BYTE]_CACHE,                  ;READ 1 BYTE
                                ;15775          STATE_STATE.AN.SKPLONG,         ;NOTE IT'S BYTE
                                ;15776          LC_RC[T2],                      ;LATCH COMPARE CHAR
                                ;15777          INTRPT.STROBE,                  ;INTERRUPTS PENDING?
U 07E7, 0000,803C,1180,4110,5404,47F2   ;15778          J/SKPBYTES                      ;
                                ;15779
                                ;15780  =010    ;-------------------------------;LA <1:0>
                                ;15781  SKPBYTES:                               ;LA<1:0> = 00
                                ;15782          ALU_D.XOR.LC,CLK.UBCC,DT/BYTE,  ;COMPARE BYTES
                                ;15783          BEN/INTERRUPT,                  ;BRANCH IF AN INTERRUPT PENDING
U 07F2, 0011,8E20,0180,F800,0010,0826   ;15784          J/SKPBYTES1                     ;
                                ;15785
                                ;15786          ;011-----------------------------;LA<1:0> = 01
                                ;15787          ALU_D.XOR.LC,CLK.UBCC,DT/BYTE,  ;COMPARE BYTES
                                ;15788          BEN/INTERRUPT,                  ;BRANCH IF AN INTERRUPT PENDING
U 07F3, 0011,8E20,0180,F800,0010,0826   ;15789          J/SKPBYTES1                     ;
                                ;15790
                                ;15791          ;110-----------------------------;LA<1:0> = 10
                                ;15792          ALU_D.XOR.LC,CLK.UBCC,DT/BYTE,  ;COMPARE BYTES.
                                ;15793          BEN/INTERRUPT,                  ;BRANCH IF AN INTERRUPT PENDING
U 07F6, 0011,8E20,0180,F800,001C,0846   ;15794          J/SKPLONG1                      ;NEXT COMPARE WILL BE OF LONGWORDS
                                ;15795                                          ;BECAUSE LA<1:0>=10, SO VA=11
                                ;15796          ;111-----------------------------;LA<1:0> = 11
                                ;15797          ALU_D.XOR.LC,CLK.UBCC,DT/BYTE,  ;COMPARE BYTES
                                ;15798          BEN/INTERRUPT,                  ;BRANCH IF AN INTERRUPT PENDING
U 07F7, 0011,8E20,0180,F800,0010,0826   ;15799          J/SKPBYTES1                     ;
```

```
                                ;15800  =110    ;--------------------------------;INTERRUPT PENDING?
                                ;15801  SKPBYTES1:                               ;NO INTERRUPT PENDING
                                ;15802          Q_Q-K[.1],CLK.UBCC,              ;DECREMENT COUNTER
                                ;15803          LAB_R[R1],                       ;LATCH SRC ADDR
U 0826, 0019,3B00,05C0,FA08,0010,06E9   ;15804          BEN/ALU,J/SKPBYTES2              ;CHARS =?
                                ;15805
                                ;15806          ;111-----------------------------;
U 0827, 0000,003C,1980,F800,0104,6A26   ;15807          J/SKPFPD,FE_K[ZERO]              ;INTERRUPT PENDING. SET FLAG
                                ;15808
                                ;15809  =1001   ;--------------------------------;ALU <Z> + IR <0>
                                ;15810  SKPBYTES2:                               ;LOCC. UNEQUAL.
                                ;15811          VA_LA+K[.1],R[R1]_LA+K[.1],      ;INCR DEST ADDR.
U 06E9, 0018,0114,0580,FA88,0200,0628   ;15812          Z?,J/SKPBYTES3                   ;MORE TO DO?
                                ;15813
                                ;15814          ;1011----------------------------;SKPC. UNEQUAL.
                                ;15815          R[R0]_Q+K[.1],                   ;BYTES DON'T MATCH
U 06EB, 0019,2014,0580,FA80,0000,0A25   ;15816          J/SKPLOCEXIT                     ;ALL DONE
                                ;15817
                                ;15818          ;1101----------------------------;LOCC. EQUAL.
                                ;15819          R[R0]_Q+K[.1],                   ;BYTES MATCH. RESET COUNT
U 06ED, 0019,2014,0580,FA80,0000,0A25   ;15820          J/SKPLOCEXIT                     ;ALL DONE
                                ;15821
                                ;15822          ;1111----------------------------;SKPC. EQUAL.
                                ;15823          VA_LA+K[.1],R[R1]_LA+K[.1],      ;BYTES MATCHED. INCREMENT ADDR
U 06EF, 0018,0114,0580,FA88,0200,0628   ;15824          Z?,J/SKPBYTES3                   ;MORE TO DO?
                                ;15825
                                ;15826  =0     ;--------------------------------;ALU <Z>
                                ;15827  SKPBYTES3:                               ;THIS IS THE ENTRY POINT FOR TERMINAL
                                ;15828                                           ;BYTES AS WELL AS PART OF THE BYTE LOOP
                                ;15829          D[BYTE]_CACHE,                   ;ALU NE 0 SO MORE TO DO
                                ;15820          J/SKPLOCEXIT                     ;ALL DONE
                                ;15821
                                ;15822          ;1111----------------------------;SKPC. EQUAL.
                                ;15823          VA_LA+K[.1],R[R1]_LA+K[.1],      ;BYTES MATCHED. INCREMENT ADDR
U 06EF, 0018,0114,0580,FA88,0200,0628   ;15824          Z?,J/SKPBYTES3                   ;MORE TO DO?
                                ;15825
                                ;15826  =0     ;--------------------------------;ALU <Z>
                                ;15827  SKPBYTES3:                               ;THIS IS THE ENTRY POINT FOR TERMINAL
                                ;15828                                           ;BYTES AS WELL AS PART OF THE BYTE LOOP
                                ;15829          D[BYTE]_CACHE,                   ;ALU NE 0 SO MORE TO DO
                                ;15830          STATE_STATE.AN.SKPLONG,          ;READ ANOTHER BYTE
                                ;15831          LC_RC[.2],                       ;LATCH COMPARE CHAR
                                ;15832          INTRP_PROBE,                     ;CHECK FOR AN INTERRUPT
U 0628, 0000,823C,1180,4110,5404,47F2   ;15833          J/SKPE[.5],BEN/ROR               ;BRANCH ON BYTE OFFSET OF ADDR
                                ;15834
                                ;15835          ;1--------------------------------;
U 0629, 0003,003C,0180,FA80,0000,0A25   ;15836          J/SKPLOC[.5],R[R0]_0             ;COUNTER = 0. ALL DONE.
                                ;15837
                                ;15838  =110    ;--------------------------------;INTERRUPT?
                                ;15839  SKPLONG1:
                                ;15840          Q_Q-K[.4]-1,                     ;LAST BYTE READ WAS AT BYTE 3
                                ;15841          CLK.UBCC,                        ;OF LONGWORD. NEXT WILL BE AT LONGWORD
                                ;15842          LAB_R[R1],                       ;BOUNDARY. SEE IF A LWD LEFT TO READ
U 0846, 0019,3B08,11C0,FA08,0010,06F9   ;15843          BEN/ALU,J/SKPLONG2               ;MATCH?
                                ;15844
                                ;15845          ;111-----------------------------;
U 0847, 0000,003C,1980,F800,0104,6A26   ;15846          J/SKPFPD,FE_K[ZERO]              ;GO SERVICE INTERRUPT
```

```
                                    ;15847  =1001   ;-------------------------------;Z BIT + IR <0>
                                    ;15848  SKPLONG2:
                                    ;15849          VA_LA+K[.1],R[R1]_LA+K[.1],     ;LOCC + UNEQUAL. INCREMENT SRC ADDR
U 06F9, 0018,0014,0580,FA88,0200,0A16 ;15850        J/SKPALIGNED                    ;CONTINUE WITH LWD SEARCH
                                    ;15851
                                    ;15852          ;1011-------------------------;
                                    ;15853          R[R0]_Q+5,                      ;SKPC + UNEQUAL.
U 06FB, 0019,2010,1180,FA80,0000,0A25 ;15854        J/SKPLOCEXIT                    ;ALL DONE. MISMATCH WHILE IN BYTE SEARCH
                                    ;15855
                                    ;15856          ;1101-------------------------;
                                    ;15857          R[R0]_Q+5,                      ;LOCC + EQUAL.
U 06FD, 0019,2010,1180,FA80,0000,0A25 ;15858        J/SKPLOCEXIT                    ;ALL DONE. MATCH IN BYTE SEARCH
                                    ;15859
                                    ;15860          ;1111-------------------------;
U 06FF, 0018,0014,0580,FA88,0200,0A16 ;15861        VA_LA+K[.1],R[R1]_LA+K[.1]      ;SKPC + EQUAL. INCREMENT SRC ADDR
                                    ;15862                                          ;CONTINUE SEARCH BY LWDS
```

```
                                        ;15863  .TOC    ""       Character string      : SKPC/LOCC LONGWORD OPERATIONS''
                                        ;15864
                                        ;15865  ;HAVE HIT FIRST LONGWORD BOUNDARY. MAKE A LONGWORD OF COMPARE CHARACTER
                                        ;15866  ;SO COMPARISONS CAN BE DONE A LONGWORD AT A TIME.
                                        ;15867
                                        ;15868          ;--------------------------------;
                                        ;15869  SKPALIGNED:
                                        ;15870          R[R0]_Q,                        ;SAVE CURRENT COUNT
                                        ;15871          SC_K[.FFF8],                    ;PREPARE TO MAKE LWD OF COMP CHAR
U 0A16, 0001,233C,7180,FA80,0084,65E9   ;15872          C3T?                            ;<4 CHARS LEFT?
                                        ;15873
                                        ;15874  =01     ;--------------------------------;ALU <C>
                                        ;15875          Q_Q+K[.4],CLK.UBCC,             ;<4 BYTES TO READ. RESET COUNTER
U 05E9, 0019,2014,11C0,F800,0010,0A18   ;15876          J7SKPALIGNED1                   ;SEE IF ANY LEFT AT ALL
                                        ;15877
                                        ;15878          ;11--------------------------------;
                                        ;15879          Q_RC[T2],D_RC[T2],              ;AT LEAST 4 BYTES TO READ
U 05EB, 0810,0038,01C0,F910,0000,0A19   ;15880          J7SKPALIGNED2                   ;PREPARE TO MAKE A LWD OF CMP CHAR
                                        ;15881
                                        ;15882          ;--------------------------------;
                                        ;15883  SKPALIGNED1:
U 0A18, 0000,013C,0180,F800,0000,0628   ;15884          Z?,J/SKPBYTES3                  ;SEE IF COUNT > 0. ASSUMES LA STILL
                                        ;15885                                          ;SET TO REFLECT UNINCREMENTED ADDRESS
```

```
                                ;15886          ;-------------------------------;
                                ;15887  SKPALIGNED2:
U 0A19, 0B00,003C,0180,F800,0000,0A1A   ;15888          D_D.SWAP                        ;CHAR IN BYTE 3 OF D + BYTE 0 OF Q
                                ;15889
                                ;15890          ;-------------------------------;
U 0A1A, 0D00,003C,0180,F800,0000,0A1C   ;15891          D_DAL.SC                        ;CHAR IN BYTES 2,3 OF D
                                ;15892
                                ;15893          ;-------------------------------;
U 0A1C, 0B00,003C,01E0,F800,0000,0A1D   ;15894          D_D.SWAP,Q_D                    ;CHAR IN D 0,1 + Q 2,3
                                ;15895
                                ;15896          ;-------------------------------;
U 0A1D, 001D,0030,0180,F990,0000,0A20   ;15897          RC[T2]_D.OR.Q                   ;RC[T2] NOW HAS LONGWORD OF COMPARE CHAR
                                ;15898
                                ;15899          ;-------------------------------;
                                ;15900          Q_R[R0],                        ;COUNT IN Q
                                ;15901          INTRPT.STROBE,                  ;TEST FOR INTERRUPTS
U 0A20, 0000,003C,01C0,FA00,4000,032F   ;15902          J/SKPLOOP2                      ;
                                ;15903
                                ;15904  =110    ;-------------------------------;INTERRUPT?
                                ;15905  SKPLONGLOOP:
                                ;15906          D_D.XOR.LC,CLK.UBCC,            ;COMPARE LWDS
                                ;15907          SC_K[ZERO],                     ;ASSUME MISS AT BYTE 0
                                ;15908          LAB_R[R1],                      ;LATCH SRC ADDR
U 0856, 0811,0320,1980,FA08,0094,6671   ;15909          C31?,J/SKPLOOP3                 ;BRANCH ON # BYTES LEFT
                                ;15910
                                ;15911          ;111----------------------------;
U 0857, 0000,003C,1980,F800,0104,6A26   ;15912          J/SKPFPD,FE_K[ZERO]             ;GO SERVICE THE INTERRUPT
                                ;15913
                                ;15914  =01     ;-------------------------------;ALU <C>
                                ;15915  SKPLOOP3:
                                ;15916          Q_Q+K[.4],CLK.UBCC,             ;< 4 BYTES LEFT. RESET + TEST COUNTER.
                                ;15917          BEN/ALU,                        ;A MATCH?
U 0671, 0019,3B14,11C0,F800,0010,03A9   ;15918          J/SKPLASTBYTES                  ;GET OUT OF LWD LOOP
                                ;15919
                                ;15920          ;11-----------------------------;
                                ;15921          VA_LA+K[.4],R[R1]_LA+K[.4],     ;> 3 BYTES LEFT TO READ
                                ;15922          BEN/ALU,                        ;A MATCH?
U 0673, 0018,1B14,1180,FA88,0200,0329   ;15923          J/SKPLOOP1                      ;CONTINUE IN LWD LOOP
                                ;15924
                                ;15925  ; ************************************************
                                ;15926  ; * Patch no. 048, PCS 0673 trapped to WCS 1176 *
                                ;15927  ; ************************************************
```

```
                                ;15928  ;WHEN CALLED, R[R1] POINTING AT BYTE 0 OF NEXT LONGWORD, I.E. THE LWD
                                ;15929  ;AFTER THE ONE FOR THE COMPARE OF THIS BRANCH.
                                ;15930  ;Q IS 8 BYTES LESS THAN COUNT AT BYTE 0 OF LWD BEING COMPARED.
                                ;15931
                                ;15932  =01001  ;-------------------------------;RETURN1F, IR <0> + ALU Z
                                ;15933  SKPLOOP1:
                                ;15934          CALL,J/LOCEQLONG,BEN/D.BYTES,   ;LOCC, ALU NE 0. SEE IF ANY BYTES MATCH.
U 0329, 0819,1825,C180,F800,0000,05CC   ;15935          D_D.ANDNOT.K[.FFFF]             ;CLEAR LOW WORD SO IF MISMATCH IS
                                ;15936                                          ;IN BYTES 2 OR 3, ONLY NEED A 4-WAY
                                ;15937                                          ;BEN TO CATCH IT
                                ;15938                                          ;LOCEQLONG RETURNS F IF NO MATCH,
                                ;15939                                          ;RETURNS 1F IF MATCH FOUND(ALL DONE)
                                ;15940          ;01011--------------------------;
                                ;15941  SKPLOOP4:                               ;SKPC, ALU NE 0
                                ;15942          R[R1] LA,                       ;KNOW THE SKPC INSTRUCTION
U 032B, 0000,003C,0180,FA88,0000,0A21   ;15943          J/SKPUNEQLONG                   ;WILL BE TERMINATED AT THIS LONGWORD
                                ;15944                                          ;SO SET ADDR TO BYTE 0 OF IT TO
                                ;15945                                          ;FIGURE OUT 1ST BYTE THAT MISMATCHED
                                ;15946          ;01101--------------------------;
                                ;15947          Q_Q+K[.8],                      ;LOCC, ALU = 0. WHOLE LWD OF = FOUND
U 032D, 0019,2014,01C0,F800,0000,033F   ;15948          J/LOCUNEQ                       ;
                                ;15949
                                ;15950          ;01111--------------------------;
                                ;15951  SKPLOOP2:                               ;SKPC, ALU = 0. CHAR FOUND. CONT SEARCH.
                                ;15952                                          ;LOCC. CHAR NOT FOUND RETURN FROM LOCEQLONG
                                ;15953          D[LONG]_CACHE,STATE_SKPLONG,    ;READ A LONGWORD OF SRC
                                ;15954          Q_Q-K[.4],CLK.UBCC,             ;DECREMENT + CHECK COUNTER
                                ;15955          LC_RC[T2],                      ;LATCH COMPARE CHAR
                                ;15956          BEN/INTERRUPT,                  ;INTERRUPT PENDING?
U 032F, 0019,2E00,11C0,4110,1414,6856   ;15957          J/SKPLONGLOOP                   ;
                                ;15958
                                ;15959  =11111  ;-------------------------------;RETURN1F
                                ;15960
                                ;15961  ;LOCC ONLY. A MATCH HAS BEEN FOUND AT BYTE INDICATED BY SC IN CURRENT LONGWORD.
                                ;15962  ;DECREMENT COUNT(Q) FROM START OF THIS LWD TO CORRECT BYTE.
                                ;15963  ;LA POINTING AT ADDR OF BYTE 0 OF CURRENT LWD.
                                ;15964  ;UPDATE REGISTERS AS PER SRM FOR MATCH.
                                ;15965
                                ;15966  LOCUNEQ:
                                ;15967          R[R0] Q-K[SC],                  ;DECREMENT COUNTER BY PRESET AMT
U 033F, 0019,2000,1D80,FA80,0000,0A2(   ;15968          J/SKPUNEQ5                      ;
```

```
                                ;15969  =01001  ;-------------------------------;RETURN1F, IR<0>, ALU Z
                                ;15970  SKPLASTBYTES:                           ;R1 POINTING AT BYTE 0 OF CURRENT LWD.
                                ;15971                                          ;Q VALUE FOR BYTE 0 OF NEXT LWD
                                ;15972          CALL,J/LOCEQLONG,               ;LOCC. SEE IF ANY BYTES MATCH
                                ;15973          D_D.ANDNOT.K[.FFFF],            ;CLEAR LOW WORD
U 03A9, 0819,1825,C180,F800,0000,05CC   ;15974          BEN/D.BYTES             ;
                                ;15975
                                ;15976          ;01011--------------------------;
                                ;15977          Q_Q-K[.4],                      ;SKPC. LAST LONGWORD DIDN'T MATCH
U 03AB, 0019,2000,11C0,F800,0000,0A21   ;15978          J7SKPUNEQLONG           ;DECREMENT Q SO COMPATIBLE WITH
                                ;15979                                          ;ENTRY FROM LONGLOOP
                                ;15980          ;01101--------------------------;
                                ;15981          Q_Q+K[.4],R[R0]_Q+K[.4],        ;LOCC. FOUND A LWD OF CHAR
U 03AD, 0019,2014,11C0,FA80,0000,0A25   ;15982          J7SKPLOCEXIT            ;ADDR POINTING AT BYTE 0 ALREADY
                                ;15983                                          ;SET COUNT TO INCLUDE WHOLE LWD
                                ;15984          ;01111--------------------------;
                                ;15985                                          ;LOCC. NO MATCH RETURN FROM LOCEQLONG
                                ;15986          VA_LA+K[.4],R[R1]_LA+K[.4],     ;SKPC. LAST LONGWORD MATCHED
U 03AF, 0018,0114,1180,FA88,0200,0628   ;15987          Z?,J/SKPBYTES3          ;CHECK ON TERMINAL BYTES
                                ;15988
                                ;15989  =11111  ;-------------------------------;RETURN1F FOR LOCEQLONG
                                ;15990  ;LOCC. LONGWORDS MATCHED RETURN. Q IS NOW POINTING AT BYTE 0 OF PREVIOUS
                                ;15991  ;LWD (BECAUSE OF THE Q_Q+8 AT LOCEQLONG). WANT IT TO BE BYTE 0 OF LWD
                                ;15992  ;IN WHICH THE MATCH OCCURRED, SO SUBTRACT 4 TO GET IT.
U 03BF, 0019,2000,11C0,F800,0000,033F   ;15993          Q_Q-K[.4],J/LOCUNEQ     ;DECREMENT COUNT TO BE EQUIV TO
                                ;15994                                          ;BYTE 0 OF CURRENT LWD
```

```
                                        ;15995  ;ALGORITHM:
                                        ;15996  ;       THIS ROUTINE SHARED BY LOCC LONGWORD + BYTE COMPARES, WHICH MAKES FOR
                                        ;15997  ;       SLIGHTLY UGLY COUNTER MACHINATIONS. FOR LWDS, Q IS EQUIVALENT TO
                                        ;15998  ;       BYTE 0 OF 2 LWDS PAST CURRENT LWD(I.E. THE LWD UNDER CONSIDERATION
                                        ;15999  ;       HERE). FOR BYTES, Q IS EQUIVALENT TO BYTE 0 OF 1 LWD PAST CURRENT LWD.
                                        ;16000  ;       IF A MATCH IS FOUND, ROUTINE RETURNS 1F WITH Q POINTING TO START OF
                                        ;16001  ;       CURRENT LWD + SC = BYTE POSITION OF 1ST MATCHING BYTES.
                                        ;16002  ;       IF NO MATCH, ROUTINE RETURNS F WITH REGISTERS SUITABLE TO CONT SEARCH
                                        ;16003
                                        ;16004  ;CALLING SEQUENCE:
                                        ;16005  ;       CALL,J/LOCEQLONG,BEN/D.BYTES
                                        ;16006
                                        ;16007  ;INPUTS:
                                        ;16008  ;       D<31:16>= 2 BYTES TO COMPARE, <15:0>=0
                                        ;16009  ;       Q = COUNT(SEE ABOVE FOR EXACT VALUE)
                                        ;16010  ;       SC = 0
                                        ;16011
                                        ;16012  ;OUTPUTS:
                                        ;16013  ;       SC = 0, 1, 2, OR 3 IF A MATCH; = 0 IF NO MATCH
                                        ;16014
                                        ;16015  ;RETURN:
                                        ;16016  ;       RETURNF IF NO MATCH; RETURN1F IF MATCH
                                        ;16017
                                        ;16018
                                        ;16019  =1100   ;-------------------------------;D.BYTES 1+0
                                        ;16020  LOCEQLONG:                              ;LOCC. DETERMINE IF A MATCH FOUND
                                        ;16021                                          ;IN CURRENT LONGWORD COMPARE
U 05CC, 0019,2016,01C0,F800,0000,001F   ;16022          Q_Q+K[.8],RETURN1F             ;BOTH BYTES = 0
                                        ;16023
                                        ;16024          ;1101-------------------------;
U 05CD, 0000,003C,0580,F800,0084,65CC   ;16025          SC_K[.1],J/LOCEQLONG           ;BYTE 1 = 0
                                        ;16026
                                        ;16027          ;1110-------------------------;
U 05CE, 0019,2016,01C0,F800,0000,001F   ;16028          Q_Q+K[.8],RETURN1F             ;BYTE 0 = 0
                                        ;16029
                                        ;16030          ;1111-------------------------;
U 05CF, 0000,183C,0980,F800,0084,6490   ;16031          BEN/D.BYTES,SC_K[.2]           ;BYTES 0 + 1 UNEQ 0. ASSUME IT'S 2
                                        ;16032
                                        ;16033  =00**   ;-------------------------------;D.BYTES 3 + 2
U 0490, 0019,2016,01C0,F800,0000,001F   ;16034          Q_Q+K[.8],RETURN1F             ;BYTES 2 + 3 = 0
                                        ;16035
                                        ;16036          ;01**-------------------------;
U 0494, 0000,003C,0D80,F800,0084,65CC   ;16037          SC_K[.3],J/LOCEQLONG           ;BYTE 3 = 0
                                        ;16038
                                        ;16039          ;10**-------------------------;
U 0498, 0019,2016,01C0,F800,0000,001F   ;16040          Q_Q+K[.8],RETURN1F             ;BYTE 2 = 0
                                        ;16041
                                        ;16042          ;11**-------------------------;
U 049C, 0000,003E,0180,F800,0000,000F   ;16043          RETURNF                        ;NO MATCH FOUND. CONTINUE READING.
                                        ;16044
                                        ;16045          ;-------------------------------;
```

```
                                ;16046  .TOC     ""      Character string      : SKPC - DETERMINE WHICH BYTE''
                                ;16047  ;SKPC. MISMATCH FOUND IN A LWD. DETERMINE WHICH BYTE OF LWD DIDN'T MATCH
                                ;16048  ;Q, THE COUNTER, IS 8 LESS THAN COUNT AT BYTE 0 OF CURRENT LWDS
                                ;16049
                                ;16050          ;-------------------------------;
                                ;16051  SKPUNEQLONG:
                                ;16052          D_D.ANDNOT.K[.FFFF],            ;CLEAR LOW WORD SO IF MISMATCH
U 0A21, 0819,1824,C180,F800,0000,062C   ;16053          BEN/D.BYTES                     ;IS IN BYTES 2 OR 3 ONLY A 2-WAY
                                ;16054                                          ;BEN IS NEEDED
                                ;16055  =1100   ;-------------------------------;D.BYTES 1 + 0
                                ;16056          Q_Q-K[.2],                      ;BYTES 0 + 1 BOTH =
                                ;16057          SC_K[.2],                       ;PRE-ASSUME IT'S BYTE 2
U 062C, 0019,3800,09C0,F8_0,0084,60B8   ;16058          D.B2?,J/SKPUNEQ3                ;DETERMINE IF MISMATCH IS BYTE 2 OR 3
                                ;16059
                                ;16060          ;1101-----------------------;
U 062D, 0000,003C,1980,F800,0084,60BC   ;16061          SC_K[ZERO],J/SKPUNEQ1           ;BYTE 0 UNEQUAL
                                ;16062
                                ;16063          ;1110-----------------------;
                                ;16064          SC_K[.1],Q_Q-K[.1],             ;
U 062E, 0019,2000,05C0,F800,0084,60BC   ;16065          J/SKPUNEQ1                      ;BYTE 1
                                ;16066
                                ;16067          ;1111-----------------------;
U 062F, 0000,003C,1980,F800,0084,60BC   ;16068          SC_K[ZERO],J/SKPUNEQ1           ;BYTES 0 + 1 UNEQUAL. BYTE 0 COMES 1ST
                                ;16069
                                ;16070  =10**   ;-------------------------------;D.BYTE 2
                                ;16071  SKPUNEQ3:
                                ;16072          Q_Q-K[.1],                      ;Q + SC ALREADY = 2.
U 0088, 0019,2000,05C0,F800,0080,C0BC   ;16073          SC_SC+1,J/SKPUNEQ1              ;JUST INCREMENT FOR BYTE 3
                                ;16074
                                ;16075          ;11**-----------------------;
                                ;16076  SKPUNEQ1:
U 00BC, 0019,2014,0180,FA80,0000,0A24   ;16077          R[R0]_Q+K[.8]                   ;UPDATE R0 TO BE WITHIN CURRENT LWD
                                ;16078
                                ;16079          ;-------------------------------;
                                ;16080  SKPUNEQ5:
U 0A24, 0018,0014,1D80,FA88,0000,0A25   ;16081          R[R1]_LA+K[SC],J/SKPLOCEXIT     ;SET COUNTER TO # BYTES AT TIME OF MISMATCH
                                ;16082
                                ;16083          ;-------------------------------;
                                ;16084
                                ;16085  ;FINAL REGISTER CONTENTS ACCRODING TO THE SRM:
                                ;16086  ;SKPC - ALL MATCHED, LOCC - NO MATCH:
                                ;16087  ;R0 = 0
                                ;16088  ;R1 = END OF STRING + 1
                                ;16089  ;SKPC - MISMATCH FOUND, LOCC - MATCH FOUND:
                                ;16090  ;R0 = # BYTES REMAINING IN STRING, INCL UNEQUAL(SKPC) OR EQUAL(LOCC)
                                ;16091  ;R1 = ADDRESS OF UNEQUAL(SKPC) OR EQUAL(LOCC) BYTE
                                ;16092
                                ;16093  SKPLOCEXIT:
                                ;16094          ALU_R[R0],SET.CC(WORD),         ;SET PSL CC ON COUNT
U 0A25, 0000,403C,0180,FA00,2070,05AE   ;16095          CLR.FPD,J/STRINGFINAL           ;CLEAR FPD BIT + EXIT
                                ;16096          ;-------------------------------;
```

```
                                    ;16097  .TOC    ''       Character string      : SKPC/LOCC FPD + RESTART''
                                    ;16098
                                    ;16099  ;SAVE ALL NECESSARY INFO IN R0 + R1
                                    ;16100  ;R0     BYTES 1-0 = LENGTH FROM Q
                                    ;16101  ;       BYTE 2 = PC DELTA
                                    ;16102  ;       BYTE 3 = COMP CHAR
                                    ;16103  ;STATE BIT 2 = 0 = BYTE
                                    ;16104  ;             = 1 = LONG
                                    ;16105  ;BIT 2 CHOSEN SO NO KMX CONFLICT ON COUNTER DECREMENT IN SAME INSTR
                                    ;16106  ;AND CAN BE USED WITH CALL,BAKUP.PC FOR FPD
                                    ;16107
                                    ;16108
                                    ;16109          ;-------------------------------;
                                    ;16110  SKPFPD: STATE2?,                        ;BRANCH ON BYTE/LONG
U 0A26, 0001,373C,0180,FA80,0000,0860 ;16111        R[R0]_Q                         ;ASSUME IT'S BYTE
                                    ;16112
                                    ;16113  =*000   ;-------------------------------;<2>FOR BEN/STATE, '0> FOR BAKUP.PC
                                    ;16114  SKPFPD1:                                ;<3> NEVER SET
                                    ;16115          Q_PC,                           ;PREPARE TO BACKUP THE PC
U 0860, 0014,0039,01C0,F800,0000,0EB8 ;16116        CALL,J/BAKUP.PC                 ;
                                    ;16117
                                    ;16118  =10     ;-------------------------------;
                                    ;16119          Q_R[T2],D_D.SWAP,SC_K[.FFF8],   ;Q<7:0> COMP CHAR,D<31:24> PC DELTA
U 0862, 0B10,0038,71C0,F910,0084,69E8 ;16120        J7FPDPACK1                      ;JOIN GENERAL PACKING ROUTINE
                                    ;16121
                                    ;16122  =100    ;-------------------------------;
U 0864, 0019,2014,0180,FA80,0000,0860 ;16123        R[R0]_Q+K[.8],J/SKPFPD1         ;BACKUP ADDR FOR LONGWORD CASE
                                    ;16124                                          ;4 FOR THIS LWD, 4 FOR NEXT LWD
                                    ;16125  =
                                    ;16126
                                    ;16127  0C1:    ;-------------------------------;
                                    ;16128  SKPRESTART:                             ;FAULT COMPLETED. RESET COUNT +
                                    ;16129                                          ;START FROM THE TOP.
                                    ;16130          D_R[R0],Q_R[R0],                ;UNPACK R0
                                    ;16131          SC_K[.FFF0],                    ;PREPARE FOR DAL.SC
U 00C1, 0800,003D,6DC0,FA00,0084,69F2 ;16132        CALL,J/FPDUNPACK                ;UNPACK PC DELTA + STATE
                                    ;16133
                                    ;16134  1C1:    ;-------------------------------;
                                    ;16135          R[T2]_D.OXT[BYTE],              ;D<7:0>=COMP CHAR
U 01C1, 0003,803C,0180,F990,0000,02F8 ;16136        J/SKPRES1                       ;CONTINUE
```

```
                                        ;16137  .TOC   ""      Character string      : SPANC, SCANC''
                                        ;16138
                                        ;16139  ;ALGORITHM:     SPANC (TIL UNEQUAL) 2B /SCANC (TIL EQUAL) 2A
                                        ;16140  ;THIS IS A VERY STRAIGHTFORWARD BYTE-WISE SEARCH.
                                        ;16141  ;AFTER OBTAINING ALL THE OPERANDS, A SRC BYTE IS READ(SCANEQ);
                                        ;16142  ;IT IS USED TO INDEX THE TABLE(SPANMORE); THE TABLE IS READ; THAT BYTE
                                        ;16143  ;IS ANDED WITH THE MASK CHARACTER AND THE SEARCH TERMINATES OR CONTINUES
                                        ;16144  ;AS A RESULT OF THAT AND OPERATION, DEPENDING ON THE OP-CODE:
                                        ;16145  ;SCANC CONTINUES IF NO MATCH, SPANC TERMINATEES IF NO MATCH
                                        ;16146
                                        ;16147  ;INPUTS:
                                        ;16148  ;Q      LENGTH(1ST OPERAND)
                                        ;16149  ;D      ADDRESS(2ND OPERAND)
                                        ;16150
                                        ;16151  ;REGISTER USAGE:
                                        ;16152  ;R0     LENGTH(DT/WORD)
                                        ;16153  ;R1     ADDR
                                        ;16154  ;R2     MASK CHAR
                                        ;16155  ;R3     TABLE ADDR
                                        ;16156
                                        ;16157  ;OUTPUTS:
                                        ;16158  ;R0     NUMBER OF BYTES REMAINING
                                        ;16159  ;R1     ADDRESS OF BYTE THAT TERMINATED SEARCH OR END OF STRING + 1
                                        ;16160  ;       IF NOT FOUND
                                        ;16161  ;R2     0
                                        ;16162  ;R3     TABLE ADDRESS
                                        ;16163
                                        ;16164  486:    ;-------------------------------;
                                        ;16165          R[T0]_Q.AND.K[.FFFF],           ;ARG 1 IS LENGTH
U 0486, 0019,2035,C180,F980,0000,047E   ;16166          CALL,J7ASPC                     ;GET ARG 3
                                        ;16167
                                        ;16168  4E6:    ;-------------------------------;
                                        ;16169          R[T1]_Q,                        ;ARG 2 IS ADDR
U 04E6, 0001,203D,0180,F988,0000,037E   ;16170          CALL,J7SPEC                     ;GET ARG 4(MASK)
                                        ;16171
                                        ;16172  4F6:    ;-------------------------------;
U 04F6, 0001,203C,0180,FA98,0000,0A29   ;16173          R[R3]_Q                         ;ARG 3 IS TBL ADDR
                                        ;16174
                                        ;16175          ;-------------------------------;
U 0A29, 0010,0038,01C0,F908,0000,0A2C   ;16176          Q_RC[T1]                        ;LOAD SRC ADDR
                                        ;16177
                                        ;16178          ;-------------------------------;
                                        ;16179          LC_RC[T0]&R1_Q,                 ;SAVE SRC ADDR
U 0A2C, 0001,203C,1980,FB80,1404,6A2D   ;16180          STATE_K[ZERO]                   ;INITIALIZE STATE REG
                                        ;16181
                                        ;16182          ;-------------------------------;
                                        ;16183          R[R0]_LC,                       ;SAVE LENGTH
U 0A2D, 0010,0038,3DF0,2E80,0000,0338   ;16184          Q_ID[PSL]                       ;LOAD PSL FOR CLRPSLCC
```

```
                                ;16185  =0****00 ;-------------------------------;
                                ;16186          CALL,J/CLRPSLCC,                ;CLEAR PSL CC
U 03 8, 0019,0035,4980,FA90,00C0,09E4   ;16187          R[R2]_D.AND.K[.FF]              ;ARG 4 IS MASK BYTE
                                ;16188
                                ;16189  =1****00 ;-------------------------------;
                                ;16190  SPANRES1:
                                ;16191          CALL,J/SETFPD,                  ;SET FPD BIT
                                ;16192          D_R[R0].AND.K[.FFFF],           ;CLEAR OUT HIGH WORD
U 0378, 0818,0035,C180,FA00,0010,0E16   ;16193          C[K.UBCC                        ;VERIFY COUNTER > 0
                                ;16194
                                ;16195          ;-------------------------------;
U 0379, 0000,003C,0180,F800,0000,06D8   ;16196          J/FPDPACK                       ;NO SPECIAL FPD SETUP REQD
                                ;16197
                                ;16198          ;-------------------------------;
U 037A, 0000,003C,0180,F800,0000,06D8   ;16199          J/FPDPACK                       ;NO SPECIAL FPD SETUP REQD
                                ;16200
                                ;16201          ;-------------------------------;
                                ;16202  SPANSTART:                              ;
                                ;16203          Q_D,                            ;COUNTER TO Q
                                ;16204          VA_R[R1],                       ;SET ADDR OF FIRST CHAR
U 037B, 0000,013C,01E0,FA08,0200,0634   ;16205          Z?,J/SPANUNEQ                   ;COUNTER > 0?
                                ;16206
                                ;16207  =0      ;-------------------------------;ALU <Z>
                                ;16208  SCANEQ:                                 ;MORE TO DO
                                ;16209  SPANUNEQ:                               ;MORE TO DO
                                ;16210          D[BYTE]_CACHE,                  ;READ SRC CHAR
                                ;16211          LAB_R[R3],                      ;PREPARE TO READ FROM TBL
U 0634, 0000,803C,0180,4218,0000,0A2E   ;16212          J/SPANMORE                      ;
                                ;16213
                                ;16214          ;1--------------------------------;ALL DONE
                                ;16215          R[R0]_0,SET.CC(LONG),           ;COUNTER = 0
                                ;16216          SGN/C[R.SD+SS,                  ;CLEAR 'WRITE REG' FLAG
U 0635, 0003,003C,0187,FA80,0070,09F1   ;16217          J/R2ZERO                        ;R1=START OF SRC
                                ;16218                                          ;R2=START OF TBL
                                ;16219          ;-------------------------------;
                                ;16220  SPANMORE:                               ;
                                ;16221          VA_LB+D.OXT,DT/BYTE,            ;SET VA TO ADDR WITHIN TABLE
                                ;16222                                          ;0 EXTEND TBL OFFSET THRU AMX
U 0A2E, 000F,8014,0180,F800,4200,0A30   ;16223          INTRPT.STROBE                   ;TIME TO CHECK ON INTERRUPTS
                                ;16224
                                ;16225          ;-------------------------------;
                                ;16226          D[BYTE]_CACHE,                  ;READ A BYTE FROM THE TABLE
                                ;16227          SC_R[R2],                       ;GET MASK CHARACTER
U 0A30, 0C00,8E3C,0180,4210,0082,0866   ;16228          BEN/INTERRUPT                   ;INTERRUPT PENDING?
                                ;16229
                                ;16230  =110    ;-------------------------------;INTERRUPT?
                                ;16231                                          ;NO INTERRUPTS PENDING
                                ;16232          ALU_D.AND.K[SC],DT/BYTE,        ;COMPARE IT WITH THE MASK CHAR
                                ;16233          CLK.UBCC,                       ;SEE IF =
                                ;16234          LAB_R[R1],                      ;LATCH SRC ADDR
U 0366, 0019,8034,1D80,FA08,0010,0A31   ;16235          J/SPANM1                        ;
                                ;16236
                                ;16237          ;111------------------------------;INTERRUPT IS PENDING
                                ;16238          FE_K[ZERO],                     ;IT'S AN INTERRUPT
U 0867, 0000,003C,1980,F800,0104,66D8   ;16239          J/FPDPACK                       ;
```

```
                                    ;16240  SPANM1: ;-------------------------------;
                                    ;16241          BEN/ALU,                        ;BRANCH ON COMPARISON.
                                    ;16242          Q_Q-K[.1],DT/WORD,              ;DECREMENT COUNTER
                                    ;16243          C[K.UBCC,                       ;SEE IF = 0
U 0A31, 0019,7B00,05C0,FA80,0010,0749 ;16244        R[R0]_Q-K[.1]                   ;SAVE A COPY OF THE COUNTER IN
                                    ;16245                                          ;CASE FAULTED
                                    ;16246  =1001   ;-------------------------------;ALU <Z>+ IR <0>
                                    ;16247                                          ;SCANC. AT LEAST 1 BIT FOUND. DONE
                                    ;16248          R[R0] Q+K[.1],SET.CC(LONG),     ;FOUND A MATCH. SET PSL CC<Z>
                                    ;16249          SGN/C[R.SD+SS,                  ;CLEAR 'WRITE REG' FLAG
U 0749, 0019,2014,0587,FA80,0070,09F1 ;16250        J/R2ZERO                        ;ZERO R2
                                    ;16251
                                    ;16252          ;1011-------------------------;SPANC. MASK BYTE MATCH. CONTINUE
                                    ;16253          VA_LA+K[.1],R[R1]_LA+K[.1],     ;INCREMENT SOURCE ADDR
U 074B, 0018,0114,0580,FA88,0200,0634 ;16254        Z?,J/SPANUNEQ                   ;MORE TO DO?
                                    ;16255
                                    ;16256          ;1101-------------------------;SCANC. NO BITS MATCH. CONTINUE
                                    ;16257          VA_LA+K[.1],R[R1]_LA+K[.1],     ;INCREMENT SOURCE ADDR
U 074D, 0018,0114,0580,FA88,0200,0634 ;16258        Z?,J/SCANEQ                     ;MORE TO DO?
                                    ;16259
                                    ;16260          ;1111-------------------------;SPANC. NO MATCH. DONE
                                    ;16261          R[R0] Q+K[.1],SET.CC(LONG),     ;SET COUNTER TO THIS MISMATCH
                                    ;16262          SGN/C[R.SD+SS,                  ;CLEAR 'WRITE REG' FLAG
U 074F, 0019,2014,0587,FA80,0070,09F1 ;16263        J/R2ZERO                        ;R2_0, R1=LOCN OF UNEQ
                                    ;16264
                                    ;16265          ;-------------------------------;
```

```
                                    ;16266  .TOC    ""       Character string      : SPANC/SCANC RESTART""
                                    ;16267
                                    ;16268  ;ON FPD RESTART, START WHOLE COMPARE SEQUENCE ALL OVER, THEREFORE
                                    ;16269  ;NO NEED FOR STATE INFO.
                                    ;16270  ;R1 IS NOT INCREMENTED FOR NEXT READ OF SRC UNTIL 1 FULL SEQUENCE
                                    ;16271  ;HAS BEEN SUCCESSFUL. THEREFORE, IT'S CURRENT VALUE IS THE CORRECT
                                    ;16272  ;ADDR TO RE-READ
                                    ;16273
                                    ;16274  4A:     ;-------------------------------;
                                    ;16275  SPANRESTART:                           ;
                                    ;16276          CALL,J/FPDUNPACK,              ;UNPACK R0(RESET PC + STATE)
                                    ;16277          D_R[R0],Q_R[R0],               ;
U 004A, 0800,003D,6DC0,FA00,0084,69F2 ;16278        SC_K[.FFF0]                    ;
                                    ;16279
                                    ;16280  14A:    ;-------------------------------;
U 014A, 0000,003C,0180,F800,0000,0378 ;16281        J/SPANRES1                     ;GO RESET FPD ADDR + RETRY OPERATION
```

```
;16282  .TOC    "       Character string      : CMPC3, CMPC5"
;16283
;16284  ;GLOBAL STRATEGY:
;16285  ;       THE CMPC ALGORITHM WILL COMPARE LONGWORDS AS LONG AS THE
;16286  ;       LONGER SOURCE IS LONGWORD ALIGNED AND THERE ARE AT LEAST 4 BYTES
;16287  ;       LEFT TO COMPARE, OTHERWISE IT WILL COMPARE BYTES.  THE CRITERIA
;16288  ;       FOR LONGWORD COMPARES ARE RE-EVALUATED EVERY ITERATION, SO A
;16289  ;       MAXIMUM OF 6 BYTE COMPARES WILL BE PERFORMED FOR CMPC3, AND
;16290  ;       10 FOR CMPC5, REGARDLESS OF STRING LENGTH.  THE COMPARE LOOP
;16291  ;       COMPARES (R0) BYTES OF THE LONG STRING AGAINST THE SHORT STRING
;16292  ;       OR FILLS, DEPENDING ON STATE BIT 5; WHEN R0 IS EXHAUSTED,
;16293  ;       R2 IS CHECKED TO SEE IF THE LOOP SHOULD BE RE-ENTERED FOR FILL
;16294  ;       COMPARISONS.
;16295  ;       BY ORGANIZING THE REGISTERS AS "SHORT-STRING, LONG STRING"
;16296  ;       INSTEAD OF "FIRST STRING, SECOND STRING" THE NUMBER
;16297  ;       OF BRANCHES IN THE CODE AND HENCE ITS SIZE ARE GREATLY
;16298  ;       REDUCED; THE PENALTY PAID IS SOMEWHAT MORE OVERHEAD SETTING
;16299  ;       UP AND CLEANING UP.
;16300
;16301  ;FAULT/INTERRUPT STRATEGY:
;16302
;16303  ;       TO MINIMIZE FAULT/INTERRUPT/RESTART CODE, THE GENERAL REGISTERS
;16304  ;       ARE UPDATED DURING THE COMPARE LOOP.  IF A FAULT OR INTERRUPT OCCURS,
;16305  ;       THE LOOP ITERATION WHICH FAULTED IS SCRATCHED EVEN IF VALID DATA
;16306  ;       WAS READ FROM ONE OR BOTH STRINGS; THE A AND B SCRATCHPAD LATCHES
;16307  ;       ALWAYS HOLD THE BEGINNING-OF-ITERATION VALUES OF ANY GENERAL
;16308  ;       REGISTERS CHANGED BY THE CURRENT ITERATION.
;16309
;16310  ;REGISTER USAGE:
;16311
;16312  ;   R0  HOLDS VARIOUS THINGS:
;16313  ;     1) UNTIL THE SHORTER STRING IS EXHAUSTED, IT HOLDS THE
;16314  ;               NUMBER OF CHARACTERS LEFT IN THE SHORTER STRING.
;16315  ;     2) ONCE THE SHORTER STRING IS EXHAUSTED, IT HOLDS THE
;16316  ;               NUMBER OF CHARACTERS LEFT IN THE LONGER STRING.
;16317  ;     3) IF THE CMPC INSTRUCTION IS INTERRUPTED OR FAULTED,
;16318  ;               IT HOLDS THE NUMBER DESCRIBED IN 1) OR 2) BYTE-SWAPPED
;16319  ;               IN ITS UPPER HALF, AND HOLDS THE HIGH 7 BITS OF THE
;16320  ;               STATE REGISTER IN BITS 14-8 AND PC-DELTA IN BITS 0-7.
;16321  ;   R1  HOLDS THE CURRENT ADDRESS WE ARE FETCHING FROM IN THE LONGER
;16322  ;       OF THE TWO STRINGS.
;16323  ;   R2  HOLDS TWO COPIES OF THE FILL CHARACTER IN ITS HIGH HALF,
;16324  ;       AND THE ABSOLUTE DIFFERENCE IN STRING LENGTHS IN ITS LOW HALF
;16325  ;       (ONCE THE SHORTER STRING IS EXHAUSTED, R2<15:0> IS ZEROED)
;16326  ;   R3  HOLDS THE ADDRESS WE ARE CURRENTLY FETCHING FROM IN THE
;16327  ;       SHORTER OF THE TWO STRINGS. R3 "FREEZES" AT THE BYTE FOLLOWING
;16328  ;       THE SHORTER STRING ONCE THE SHORTER STRING IS EXHAUSTED.
;16329  ;   STATE REGISTER BITS USED BY CMPC ARE:
;16330  ;       0       OFF=READING LONGER SRC, ON=READING SHORTER SRC.
;16331  ;               TESTED BY READ-FAULT LOGIC
;16332  ;       4       OFF=SRC1 ARGUMENT WAS LONGER, ON=SRC2 ARGUMENT WAS LONGER.
;16333  ;               USED TO RESHUFFLE THE REGISTERS AT THE END AND GET THE
;16334  ;               SENSE OF THE COMPARE CORRECTLY.
;16335  ;       5       ON=SHORTER STRING HAS BEEN EXHAUSTED. TESTED IN THE LOOP AND
;16336  ;               IN THE FINAL REGISTER RESHUFFLE CODE.
```

```
                                        ;16337  ;CMPC3 ARGUMENT FETCH - ENTER WITH LENGTH IN Q, ADDR 1 IN D
                                        ;16338
                                        ;16339  4C2:
                                        ;16340  CMPC3:  R[T1]_Q.OXT[WORD], CLK.UBCC,    ; SAVE COMMON STRING LENGTH
U 04C2, 0003,603D,0180,F988,0010,047E   ;16341          CALL[ASPC]                       ; AND GET SRC2 ADDRESS.
                                        ;16342
                                        ;16343          ;--------------------------------;
                                        ;16344  4E2:                                     ; RETURN FROM ASPC
U 04E2, 0001,003C,0180,FA98,0000,0A34   ;16345          R[R3]_D                          ; SAVE SRC2 ADDRESS
                                        ;16346
                                        ;16347          ;--------------------------------;
U 0A34, 0001,203C,0180,FA88,0000,0A36   ;16348          R[R1]_Q                          ; SAVE SRC1 ADDRESS
                                        ;16349
                                        ;16350          ;--------------------------------;
                                        ;16351          R[R2]_0, STATE_0(A),             ; ZERO STRING LENGTH DIFFERENCE,
U 0A36, 0003,003C,0180,FA90,148A,67B0   ;16352          SC_ALU, J/CMPC                   ; CLEAR STATE, JOIN COMMON CODE.
```

```
                                        ;16353  ;CMPC5 ARGUMENT FETCH - ENTER WITH LENGTH 1 IN Q, ADDR 1 IN D
                                        ;16354
                                        ;16355  483:    ;-------------------------------;
U 0483, 0003,603D,0180,F980,0000,037E   ;16356  CMPC5:  RC[TO]_Q.OXT[WORD], CALL[SPEC]  ; SAVE SRC1 LENGTH, GET FILL CHAR
                                        ;16357
                                        ;16358          ;-------------------------------;
                                        ;16359  493:    RC[T1]_Q, Q_D, D_D.SWAP,        ; SAVE SRC1 ADDRESS, START REP-
U 0493, 0B01,203D,71E0,F988,0084,60A2   ;16360          SC_K[.FF8], CAL[[MOVCCMPC5]     ; LICATING FILL CHAR, JOIN
                                        ;16361                                          ; COMMON CODE WITH MOVC5.
                                        ;16362          ;-------------------------------;
                                        ;16363
                                        ;16364  ;       ON RETURN FROM MOVCCMPC5, RC[TO]=SRC1 LENGTH,
                                        ;16365  ;       RC[T2]=Q=SRC2 LENGTH,R2<31:16>= 2 COPIES OF FILL CHAR,
                                        ;16366  ;       RC[T1]=SRC1 ADDR, D=SRC2 ADDR
                                        ;16367
                                        ;16368          ;-------------------------------;
                                        ;16369  0C93:                                   ; MOVCCMPC5 RETURNS[800]
U 0C93, 0001,003C,6580,FB80,1404,6A38   ;16370          LC_RC[TO]&R1_D, STATE_K[.10]    ; GET SRC1 LEN, R1 GETS SRC2ADDR,
                                        ;16371                                          ; SET "SRC2 LONGER" FLAG AS A GUESS
                                        ;16372          ;-------------------------------;
                                        ;16373          R[R2]_Q-LC, Q_Q-LC, DT/WORD,    ; GET LENGTH DIFF IN R2<15:0>
U 0A38, 0011,6000,01C0,FA90,0010,0A39   ;16374          CLK.UBCC
                                        ;16375
                                        ;16376          ;-------------------------------;
U 0A39, 0001,033C,0180,FA98,0000,065C   ;16377          R[R3]_D, C31?                   ; TEST WHICH IS LONGER
                                        ;16378
                                        ;16379  =0*     ;-------------------------------; BRANCH ON C31 (LC<=Q --> SRC2 IS BIGGER)
                                        ;16380          R[R0]_Q+LC, CLK.UBCC, D_0,      ; SRC1 BIGGER - R0 GETS SRC2LEN
U 065C, 0F11,2014,0580,FA80,0094,6A3A   ;16381          SC_K[.1], J/CMPCSRC1BIG         ; R1 GETS SRC1AD, R3 HAS SRC2AD
                                        ;16382
                                        ;16383          ;1*-------------------------------;
                                        ;16384          R[R0]_LC, CLK.UBCC, SC_K[.3],   ; SRC2 BIGGER - R0 GETS SRC1LEN
U 065E, 0010,0038,0D80,FA80,0094,67B0   ;16385          J/CMPC                          ; R3 GETS SRC1AD, R1 HAS SRC2AD
                                        ;16386
                                        ;16387          ;-------------------------------;
                                        ;16388  CMPCSRC1BIG:
                                        ;16389          R[R2]_D-Q, DT/WORD,             ; SRC1 BIGGER - NEGATE DIFF IN R2
U 0A3A, 001D,4000,1980,FA90,1404,67B0   ;16390          STATE_K[ZERO], J/CMPC           ; AND CLEAR "SRC2 LARGER" FLAG
                                        ;16391
                                        ;16392          ;-------------------------------;
```

```
                                        ;16393  ;       CMPC3 AND CMPC5 MERGE HERE. INTERRUPT RESTART ALSO COMES HERE
                                        ;16394  ;       TO SET UP FAULT ADDRESS, AND CMPCFILL COMES HERE TO RE-ENTER
                                        ;16395  ;       LOOP FOR FILLS.
                                        ;16396  ;       SC = NUMBER OF REGISTER WHICH GETS CONTENTS OF RC[T1],
                                        ;16397  ;       D = 4 COPIES OF FILL CHAR (NOT TRUE ON ENTRY FROM CMPC3/CMPC5)
                                        ;16398  ;       Z SET ON LOOP COUNT
                                        ;16399
                                        ;16400  =00     ;-------------------------------; CONSTRAINT BLOCK FOR CALL
                                        ;16401  CMPC:   LC_RC[T1], ID[T1]_D,            ; SAVE FILLS AND GET VALUE TO STORE
U 07B0, 0000,003D,C580,3D08,0000,07B3   ;16402          CALL[CMPCSETFPD]                ; THIS CALL IS JUST TO SAVE UPC
                                        ;16403
                                        ;16404  =10     ;-------------------------------; READ FAULT ENTRY - CMPC DOES NOT WRITE
                                        ;16405  CMPC.RDFAULT:
                                        ;16406          R[R0]_LB, D_LB, SD_NOT.SD,      ; RESTORE R0, SET FAULT FLAG,
U 07B2, 080C,1738,0183,FA80,0000,0658   ;16407          STATE0?, J/CMPCFPD              ; CHECK WHICH READ FAULTED
                                        ;16408
                                        ;16409          ;11-----------------------------;
                                        ;16410  CMPCSETFPD:
U 07B3, 0010,0038,81F0,2CE8,0000,0A3C   ;16411          R(SC)_LC, Q_ID[USTACK]          ; SC CAN HAVE THE FOLLOWING VALUES:
                                        ;16412                                          ; 1 OR 3 (LC HAS SRC1 ADDR)
                                        ;16413                                          ; 0 (LC=LENGTH FROM CMPCFILL/RST)
                                        ;16414          ;-------------------------------;
                                        ;16415          LAB_R[R1], D_Q,                 ; PRIME LATCHES FOR LOOP WHILE
U 0A3C, 0C00,013C,0180,FA08,0000,016C   ;16416          Z?, J/CMPCLPENTRY               ; CHECKING FOR NULL LOOP.
```

```
                                        ;16417  ;       THIS IS THE MAIN (AND ONLY) COMPARE LOOP. R0 COUNTS DOWN THE NUMBER
                                        ;16418  ;       OF CHARACTERS TO COMPARE. THE STRING ADDRESSES ARE IN R1 AND R3.
                                        ;16419  ;       IF STATE<5>=1, THE R1 STRING IS COMPARED AGAINST FILLS FROM ID[T4].
                                        ;16420  ;       AT THE END OF EACH ITERATION, SC HOLDS THE NUMBER OF BYTES
                                        ;16421  ;       COMPARED ON THAT ITERATION (1 OR 4), ALU<Z> IS SET ON R0,
                                        ;16422  ;       LA&LB CONTAIN R1.
                                        ;16423
                                        ;16424  =0      ;-------------------------------; ALU<Z> (R0 = 0)
                                        ;16425                                          ; (NOTE "SECOND READ" IS SET HERE)
                                        ;16426  CMPCLP:
                                        ;16427          R[R1]_LA+K[SC], VA_ALU, FE_SC,  ; UPDATE R1 FROM LAST ITERATION,
U 0640, 0B18,1A14,1D80,FA88,0300,0188   ;16428          D_D.SWAP, PSL.Z?, J/CMPCLP.1    ; LOAD VA, CHECK LAST COMPARISON.
                                        ;16429
                                        ;16430          ;1-------------------------------;
                                        ;16431          R[R1]_LA+K[SC], FE_SC,          ; COUNT RAN OUT - UPDATE R1,
U 0641, 0B18,1A14,1D80,FA88,0100,0169   ;16432          D_D.SWAP, PSL.Z?                ; CHECK LAST COMPARISON
                                        ;16433
                                        ;16434  =10*0   ;-------------------------------; BRANCH ON PSL<Z> (SRC1=SRC2),
                                        ;16435                                          ; ALSO ON ALU<Z> (INITIAL COUNT=0)
                                        ;16436  =10*1                                   ; (ALL THIS TO SHARE A WORD)
                                        ;16437          SC_SHF.VAL, ALU_PACK.FP,        ; D HAS SWAPPED XOR - GET NUMBER
                                        ;16438          Q_ALU.RIGHT2, D_Q, ID[T0]_D,    ; OF LEADING ZERO BITS IN D INTO
U 0169, 0C88,0038,C1C0,3C90,008C,6A43   ;16439          LA_RA[R2], J/CMPCNEQUAL         ; SC & Q<9:5>, SAVE OLD D & Q
                                        ;16440
                                        ;16441          ;11*0-----------------------;
                                        ;16442  CMPCLPENTRY:                            ; ENTER COMPARE LOOP HERE
                                        ;16443          ID[FPDA]_D, SET.FPD, LAB_R[R1], ; SET FAULT VECTOR AND F.P.D.,
                                        ;16444          VA_LA, SS_0&SD_0,               ; LOAD VA WITH R1,
U 016C, 0000,003C,B587,3E08,2600,018C   ;16445          J/CMPCLP.TA                     ; INIT FAULT/INT FLAG, ENTER LOOP
                                        ;16446
                                        ;16447          ;11*1-----------------------;
                                        ;16448          D_R[R2], Q_R[R2],               ; STRINGS EQUAL - EXIT LOOP,
                                        ;16449          N&Z_ALU.V&C_0, DT/WORD,         ; GET FILLS & LENGTH DIFF IN D&Q,
U 016D, 0800,403C,75C0,FA10,1454,2A3D   ;16450          STATE_STATE.OR.K[.20]           ; SET "SHORT STRING EXHAUSTED",
                                        ;16451                                          ; SET PSL<Z> ON R2<15:0>
                                        ;16452          ;-------------------------------;
                                        ;16453          R[R2]_D-Q, DT/WORD, D_D.SWAP,   ; ZERO OUT R2<15:0>, PUT FILLS
U CA3D, 0B1D,5A00,6580,FA90,0084,61A8   ;16454          SC_K[.10], PSL.Z?, J/CMPCFILL   ; IN HI Q & LOW D, TEST R2.
```

```
                                ;16455  =10**   ;-------------------------------; PSL<Z> (LAST COMPARE WAS EQUAL)
                                ;16456  CMPCLP,1:                               ; LAST COMPARE NOT EQUAL...
                                ;16457          SC_SHF.VAL, ALU_PACK.FP,        ; D HAS SWAPPED XOR - GET NUMBER
                                ;16458          Q_ALU.RIGHT2, D_Q, ID[T0]_D,    ; OF LEADING ZERO BITS IN D INTO
U 0188, 0C88,0038,C1C0,3C90,008C,6A43   ;16459          LA_RA[R2], J/CMPCNEQUAL         ; SC & Q<9:5>, SAVE OLD D & Q
                                ;16460
                                ;16461          ;11**---------------------------;
                                ;16462  CMPCLP.1A:                              ; LAST COMPARE WAS EQUAL...
                                ;16463          LAB_R[R0], ALU_LA.ANDNOT.K[.3], ; SET Z IF COUNT < 4,
                                ;16464          STATE_STATE.ANDNOT.K[.3],       ; CLEAR "SECOND READ" FLAG, TEST
U 018C, 0018,1524,0D80,FA00,1495,475B   ;16465          SC_FE, CLK.UBCC, ALU1-0?        ; IF R1 IS LONGWORD ALIGNED.
                                ;16466
                                ;16467  =1011   ;-------------------------------; ALU<1:0>=0 (LONG SRC IS ALIGNED)
                                ;16468          D[BYTE]_CACHE, INTRPT.STROBE,   ; NOT ALIGNED - READ A BYTE
U 075B, 0000,803C,0180,4000,4080,A645   ;16469          SC_SC-FE, J/CMPCLP.2            ; AND ENTER THE BYTE COMPARE FLOWS
                                ;16470
                                ;16471          ;1111---------------------------;
                                ;16472          D[LONG]_CACHE, R[R0]_LA-K[.4],  ; ALIGNED - READ A LONGWORD,
                                ;16473          CLK.UBCC, INTRPT.STROBE,        ; CHECK IF 4 BYTES LEFT TO COMPARE
U 075F, 0018,0100,1180,4280,4090,A644   ;16474          SC_SC-FE, Z?
                                ;16475
                                ;16476  =0       ;-------------------------------; ALU<Z> ( .GT. 3 BYTES LEFT TO COMPARE)
                                ;16477          ;*NOTE* - IF .LE. 3 BYTES TO COMPARE, THE BYTE WE WANT IS
                                ;16478          ; IN D<7:0>. SINCE THE LONGWORD READ WAS ALIGNED, WE HAVEN'T
                                ;16479          ; VIOLATED THE ARCHITECTURE BY OVERREADING.
                                ;15480          ;0-------------------------------;
                                ;16481          LA_RA[R3], VA_LA, Q_ID[T1],     ; LOAD VA WITH 2D STRING ADDR,
                                ;16482          STATE_STATE+1,                  ; GET FILLS IN Q, SET "SECOND RD",
U 0644, 0000,163C,C5F0,2C98,1600,C739   ;16483          STATE5?, J/CMPCLP.6             ; BRANCH ON WHETHER TO USE FILLS.
                                ;16484
                                ;16485  CMPCLP.2:
                                ;16486          ;1-------------------------------; COME HERE TO DO BYTE COMPARE..
                                ;16487          R[R0]_LA+MASK+1, CLK.UBCC,      ; DECR COUNT BY 1 (SC=0-->MASK=-2),
U 0645, 0000,0E10,C5F0,2E80,0010,0876   ;16488          Q_ID[T1], INT?                  ; GET FILLS IN Q & CHECK INTS
                                ;16489  =110
                                ;16490          ;-------------------------------; BRANCH ON INTERRUPT (PENDINC)
                                ;16491          LA_RA[R3], VA_LA, D_Q, Q_D,     ; LOAD 2D STRING ADDR IN VA,
                                ;16492          STATE_STATE+1,                  ; SET "SECOND READ",
U 0876, 0C00,163C,01E0,F898,1600,C6D9   ;16493          STATE5?, J/CMPCLP.3             ; TEST WHETHER TO USE FILLS
                                ;16494
                                ;16495          ;111----------------------------;
U 0877, 080C,0038,0180,FA80,0000,0733   ;16496          R[R0]_LB, D_LB, J/MOVCPACKST    ; INTERRUPT PENDING - GO AWAY.
```

```
                                    ;16497  =**01   ;-------------------------------; STATE<5> (SHORT STRING EXHAUSTED)
                                    ;16498  CMPCLP.3:
U 06D9, 0000,803C,0180,4000,C000,0A3E ;16499          D[BYTE]_CACHE ,J/CMPCLP.5       ; READ BYTE FROM THE SHORT STRING
                                    ;16500
                                    ;16501          ;**11-------------------------;
                                    ;16502  CMPCLP.4:
                                    ;16503          LAB_R[R1], SC_K[.1],           ; SET UP FOR NEXT ITERATION,
                                    ;16504          D_D.XOR.Q,N&Z_ALU.V&C_0,DT/BYTE,; COMPARE LOW BYTES OF D & Q,
U 06DB, 081D,8120,0580,FA08,00D4,6640 ;16505          Z?, J/CMPCLP                   ; TEST COUNT EXHAUSTED & LOOP.
                                    ;16506
                                    ;16507          ;-------------------------------;
                                    ;16508  CMPCLP.5:
U 0A3E, 0018,0014,0580,FA98,0000,06DB ;16509          R[R3]_LA+K[.1], J/CMPCLP.4     ; BUMP 2D STRING ADDR, GO COMPARE
                                    ;16510
                                    ;16511  =**01   ;-------------------------------; STATE<5> (SHORT STRING EXHAUSTED)
                                    ;16512  CMPCLP.6:
                                    ;16513          D[LONG]_CACHE, Q_D,            ; READ LONGWORD FROM 2D STRING
                                    ;16514          R[R3]_LA+K[.4],                ; UPDATE 2D STRING POINTER
U 0739, 0018,0E14,11E0,4298,0000,0886 ;16515          INT?, J/CMPCLP.7               ; TEST INTERRUPTS
                                    ;16516
                                    ;16517          ;**11-------------------------;
U 073B, 0C00,0E3C,01E0,F800,0000,0886 ;16518          D_Q, Q_D, INT?, J/CMPCLP.7     ; COMPARE AGAINST FILLS, TEST INTS
                                    ;16519
                                    ;16520          ;-------------------------------;
                                    ;16521  =110                                   ; BRANCH ON INTERRUPT (PENDING)
                                    ;16522  CMPCLP.7:
                                    ;16523          LAB_R[R1],SC_K[.4],            ; SET UP FOR NEXT ITERATION,
                                    ;16524          D_D.XOR.Q,N&Z_ALU.V&C_0,DT/LONG,; COMPARE LONGWORDS,
U 0886, 081D,0120,1180,FA08,00D4,6640 ;16525          Z?, J/CMPCLP                   ; TEST COUNT EXHAUSTED AND LOOP.
                                    ;16526
                                    ;16527          ;111--------------------------;
U 0887, 080C,0038,0180,FA80,0000,0659 ;16528          R[R0]_LB, D_LB, J/CMPCFPD1     ; INTERRUPT - THROW EVERYTHING OUT
                                    ;16529                                         ; EVEN THOUGH WE READ BOTH STRINGS
                                    ;16530                                         ;(NO OTHER PLACE TO CHECK
                                    ;16531                                         ; FOR INTERRUPTS IN LOOP!)
```

```
                                ;16532  ;       COMPARE LOOP EXITS
                                ;16533
                                ;16534  ;       COME HERE IF LOOP EXITED WITH NO DIFFERENCES FOUND - THIS MIGHT
                                ;16535  ;       MEAN WE ARE DONE, AND IT MIGHT MEAN THAT IT IS TIME TO COMPARE
                                ;16536  ;       THE EXCESS BYTES OF THE LONG STRING AGAINST FILLS.
                                ;16537  ;       ON ENTRY FROM COMPARE LOOP, Q=R2, PSL<Z> SET ON Q<15:0>,
                                ;16538  ;       SC=16., D=R2 BYTE-SWAPPED, AND R2<15:0> HAVE BEEN ZEROED.
                                ;16539
                                ;16540  =10**   ;-------------------------------; PSL<Z> (R2<15:0>=0 -->
                                ;16541                                          ; BOTH STRINGS EXHAUSTED)
                                ;16542  CMPCFILL:
                                ;16543          RC[T1]_Q.AND.K[.FFFF],          ; GET LENGTH DIFFERENCE IN RC[T1],
                                ;16544          CLK.UBCC, D_DAL.SC,             ; GET 4 FILL CHARS IN D,
U 01A8, 0D19,2034,C180,F988,0094,47B0   ;16545          SC_SC.ANDNOT.K[.FFFF], J/CMPC   ; ZERO SC SO CMPC WILL STORE
                                ;16546                                          ; RC[T1] IN R0, AND RE-ENTER LOOP.
                                ;16547          ;11**---------------------------;
                                ;16548          LAB_R1&RC[T0]_0, SC_ALU,        ; STRINGS ARE TRULY EQUAL -
U 01AC, 0003,003C,0180,FB00,0082,0A54   ;16549          J/CMPCFINIS                     ; CLEAN UP AND EXIT
                                ;16550
                                ;16551          ;-------------------------------;
                                ;16552  ;       COME HERE IF DIFFERENCE FOUND WITH D=COMPARAND FROM LONGER
                                ;16553  ;       STRING, ID[T0] = SWAPPED XOR OF COMPARANDS, SC & Q<9:5>= NUMBER
                                ;16554  ;       OF LEADING (HIGH-ORDER) ZERO BITS IN ID[T0], LA=R2,
                                ;16555  ;       AND FE = NUMBER OF BYTES IN EACH COMPARAND (1 OR 4).
                                ;16556  ;       STATE<3:0> = 1 ("SECOND READ" BIT SET)
                                ;16557
                                ;16558  ;       WHAT WE HAVE TO DO IS FIGURE OUT WHICH ONE OF THE 4 BYTES
                                ;16559  ;       IN THE COMPARANDS IS THE FIRST MISMATCHING BYTE, SET THE
                                ;16560  ;       COND CODES ON THE COMPARE OF THE TWO UNEQUAL BYTES, AND
                                ;16561  ;       BACK UP THE STRING POINTERS/COUNTERS TO THE MISMATCH POINT.
                                ;16562
                                ;16563  CMPCNEQUAL:
                                ;16564          R[R15]_Q.RIGHT.1,               ; R15<8:4> NOW HAS # BITS,
                                ;16565          D_D.SWAP, Q_ID[T0],             ; SWAP LONG COMPARAND, GET XOR IN Q
                                ;16566          STATE_STATE+FE,                 ; STATE<3:0> = # BYTES READ + 1
U 0A43, 0841,203C,C1F0,2EF8,1500,8A44   ;16567          FE_EA[0]                        ; AND SAVE A COPY IN FE
                                ;16568
                                ;16569          ;-------------------------------;
                                ;16570          Q_D.XOR.Q,                      ; RECONSTRUCT SHORT COMPARAND
U 0A44, 001D,0020,5DC0,F800,0084,4A45   ;16571          SC_SC.ANDNOT.K[.7]              ; BYTE-SWAPPED IN Q, SC GETS
                                ;16572                                          ; SHIFT COUNT TO BYTE-NORMALIZE
                                ;16573                                          ; COMPARANDS
                                ;16574          ;-------------------------------;
                                ;16575          D_DAL.SC,                       ; BYTE-NORMALIZE LONG COMPARAND
U 0A45, 0D18,0034,C180,F988,0000,0A4B   ;16576          RC[T1]_LA.AND.K[.FFFF]          ; GET LENGTH DIFF (OR 0) IN RC[T1]
                                ;16577
                                ;16578          ;-------------------------------;
                                ;16579          D_Q, Q_D, LC_RC[T1],            ; SWAP COMPARANDS
U 0A4B, 0C00,003C,09E0,F908,1404,AA4C   ;16580          STATE_STATE-R[.2]               ; STATE<3:0> = # BYTES READ - 1
                                ;16581
                                ;16582          ;-------------------------------;
                                ;16583          D_DAL.SC,                       ; BYTE-NORMALIZE SHORT COMPARAND
                                ;16584          SC&STATE_STATE-R[R15](EXP),     ; SC<1:0> = # BYTES TO BACK UP - 1
                                ;16585                                          ; THIS UNAVOIDABLY DESTROYS STATE!
U 0A4C, 0D00,163C,0180,FA78,1488,A792   ;16586          STATE4?                         ; CHECK WHICH WAY TO COMPARE BYTES.
```

```
                                        ;16587  =**10   ;-------------------------------; STATE<4> (SRC2 STRING LARGER)
                                        ;16588                                         ; NOW THAT COMPARANDS ARE BYTE-NORMALIZED,
                                        ;16589                                         ; A LWD SUBTRACT WILL PRODUCE RIGHT CC'S
                                        ;16590          ALU_D-Q-1, SET.CC(LONG),       ; COMPARE UNEQUAL BYTES
                                        ;16591          SC_SC.ANDNOT.K[.FFFC],         ; CLEAR JUNK FROM SC
U 0792, 001D,0008,F180,F880,00F4,4A4D   ;16592          LA_RA[R0], J/CMPCADJ           ; NOW GO ADJUST REGISTERS
                                        ;16593
                                        ;16594          ;**11-------------------------;
                                        ;16595          ALU_Q-D-1, SET.CC(LONG),       ; COMPARE UNEQUAL BYTES
                                        ;16596          SC_SC.ANDNOT.K[.FFFC],         ; CLEAR CRAP FROM SC
U 0793, 001D,2008,F180,F880,00F4,4A4D   ;16597          LA_RA[R0], J/CMPCADJ           ; NOW GO ADJUST REGISTERS
                                        ;16598
                                        ;16599          ;-------------------------------;
```

```
                                        ;16600          ;-------------------------------;
                                        ;16601  CMPCADJ:
                                        ;16602          R[R0]_LA+K[SC]+1, D_ALU,        ; ASSUMING SHORT STRING NOT GONE,
U 0A4D, 0818,0010,1D80,FA80,0080,CA53   ;16603          SC_SC+1                         ; SET SHORT STRING LENGTH TO
                                        ;16604                                          ; COUNT + NUMBER OF EXCESS BYTES.
                                        ;16605          ;-------------------------------;
U 0A53, 0011,0014,0180,FB00,1400,6A54   ;16606          LAB_R1&RC[TO]_D+LC, STATE_FE    ; COMPUTE LONG STRING LENGTH,
                                        ;16607                                          ; RESTORE STATE
                                        ;16608          ;-------------------------------;
                                        ;16609  CMPCFINIS:
                                        ;16610          LC_RC[TO]&R1_LA-K[SC], Q_ALU,   ; UPDATE LONG STRING ADDRESS
U 0A54, 0018,0000,1DC0,FB80,2000,0A55   ;16611          CLR.FPD                         ; BY NUMBER OF EXCESS BYTES
                                        ;16612
                                        ;16613          ;-------------------------------;
                                        ;16614          LA_RA[R3], PC&VA_PC, FLUSH.IB,  ; KILL OLD CONTENTS OF INST BUF,
                                        ;16615          STATE_STATE.ANDNOT.K[.20],      ; CLEAR "SHORT STR EXHAUSTED"
U 0A55, 2014,1638,7580,F899,5604,4799   ;16616          STATE5?                 ; WHILE WE TEST IT.
                                        ;16617
                                        ;16618  =**01   ;-------------------------------; STATE<5> (SHORT STRING EXHAUSTED)
                                        ;16619          R[R3]_LA-K[SC],PC_PC+1,LOAD.IB, ; SHORT STR NOT GONE - UPDATE
U 0799, 0018,1600,1D80,629C,0000,0648   ;16620          STATE4?, J/CMPCSETREGS  ; SHORT STRING POINTER BY EXCESS
                                        ;16621
                                        ;16622          ;**11-----------------------;
                                        ;16623          R[R0]_0, D_ALU, SC_ALU,         ; SHORT STRING EXHAUSTED -
                                        ;16624          PC_PC+1, LOAD.IB,               ;INHIBIT BUMP OF SHORT STRING PTR,
U 079B, 0803,163C,0180,6284,0082,0648   ;16625          STATE4?, J/CMPCSETREGS  ; ZERO SHORT STRING LENGTH
                                        ;16626
                                        ;16627  =***0   ;-------------------------------; STATE<4> (STRING2 LONGER THAN STRING1)
                                        ;16628  CMPCSETREGS:
U 0648, 0001,003C,1980,FA90,0084,6A59   ;16629          R[R2]_D, SC_K[ZERO], J/CMPCEXIT ; STRING1 LONGER - SWAP R0 & R2
                                        ;16630
                                        ;16631          ;***1-----------------------;
U 0649, 0018,0000,1D80,FA88,0000,0A58   ;16632          R[R1]_LA-K[SC]                  ; STRING2 LONGER - SWAP R1 & R3
                                        ;16633
                                        ;16634          ;-------------------------------;
U 0A58, 0001,203C,0980,FA98,0084,6A59   ;16635          R[R3]_Q, SC_K[.2]
                                        ;16636
                                        ;16637          ;-------------------------------;
                                        ;16638  CMPCEXIT:
U 0A59, 0010,0038,0180,F8E8,0000,0062   ;16639          R(SC)_LC, J/IRD                 ; STORE LONG LENGTH IN R0 OR R2,
                                        ;16640                                          ; AND EXIT
```

```
                                        ;16641  ;       CMPC3/CMPC5 FAULT AND INTERRUPT HANDLING CODE.
                                        ;16642
                                        ;16643  ;       COME HERE ON READ FAULTS AND INTERRUPTS.
                                        ;16644  ;       ENTER AT CMPCFPD IF FAULT/INTERRUPT OCCURRED BEFORE SECOND READ,
                                        ;16645  ;       ENTER AT CMPCFPD1 IF IT OCCURRED DURING/AFTER SECOND READ.
                                        ;16646  ;       ON ENTRY D=R0, LA=OLD R3 IF SECOND READ.
                                        ;16647
                                        ;16648  =***0   ;-------------------------------; STATE<0> (2ND READ OF LOOP HAS OCCURRED)
                                        ;16649  CMPCFPD:
U 0658, 0000,003C,0180,F800,0000,0733   ;16650          J/MOVCPACKST                    ;THIS CAN BE BUMMED OUT
                                        ;16651          ;***1---------------------------;
                                        ;16652  CMPCFPD1:
U 0659, 0000,003C,0180,FA98,0000,0733   ;16653          R[R3]_LA, J/MOVCPACKST          ; RESTORE OLD VALUE OF R3
                                        ;16654
                                        ;16655          ;-------------------------------;
                                        ;16656
                                        ;16657  ;       CMPC3/CMPC5 FIRST PART DONE ROUTINE - ENTER HERE
                                        ;16658  ;       IF OPCODE ENCOUNTERED WHILE PSL<FPD> IS SET.
                                        ;16659
                                        ;16660  49:
                                        ;16661  CMPCRESTART:
U 0049, 0800,003C,0180,F880,1408,6A5A   ;16662          LA_RA[R0], D_LA, STATE_AMX.EXP  ; LOAD STATE FROM R0<14:7>,
                                        ;16663                                          ; D<7:0>=PCDELTA, D<31:16>=R0
                                        ;16664          ;-------------------------------;
                                        ;16665          PC&VA_D.OXT[BYTE]+PC, D_D.SWAP, ; ADD PC-DELTA TO PC, D<15:0> R0,
U 0A5A, 0B17,8014,6180,F801,1604,4A5C   ;16666          STATE_STATE.ANDNOT.K[.F]        ; CLEAR SOME GARBAGE FROM STATE
                                        ;16667
                                        ;16668          ;-------------------------------;
                                        ;16669          Q_D, D_R[R2],                   ; Q<15:0> R0, D<31:16> FILLS,
U 0A5C, 0800,003C,D1E0,FA10,1404,4A5D   ;16670          STATE_STATE.ANDNOT.K[.C0]       ; CLEAR REST OF JUNK FROM STATE
                                        ;16671
                                        ;16672          ;-------------------------------;
                                        ;16673          RC[T1]_Q.OXT[WORD], CLK.UBCC,   ; SAVE NEW CONTENTS OF R0,
U 0A5D, 0B03,603C,65E0,F988,0094,6A5E   ;16674          Q_D, D_D.SWAP, SC_K[.10]        ; PREPARE TO REPLICATE FILLS
                                        ;16675
                                        ;16676          ;-------------------------------;
U 0A5E, 0D00,003C,1980,F800,0084,67B0   ;16677          D_DAL.SC, SC_K[ZERO], J/CMPC    ; RE-ENTER MAIN LINE TO SET FPD.
```

```
                                        ;16678  .TOC     ""      Character string      : MATCHC""
                                        ;16679
                                        ;16680  ;ALGORITHM:
                                        ;16681  ;THE FIRST CHARACTER OF THE OBJECT IS COMPARED TO SEQUENTIAL
                                        ;16682  ;CHARACTERS IN THE SOURCE UNTIL THERE IS A MATCH.
                                        ;16683  ;THIS IS REFERRED TO AS THE 'OUTER LOOP''.
                                        ;16684  ;THEN THE ''INNER LOOP'' IS ENTERED TO COMPARE SUCCESSIVE CHARACTERS
                                        ;16685  ;(STARTING AT CHAR 2) OF THE OBJECT AGAINST THE SOURCE.
                                        ;16686  ;IF THE ENTIRE OBJECT STRING IS FOUND IN THE SOURCE STRING,
                                        ;16687  ;IT IS A MATCH.
                                        ;16688  ;IF THERE IS A MISMATCH IN THE INNER LOOP, THE COMPARISON CONTINUES
                                        ;16689  ;WITH THE NEXT CHARACTER OF THE SOURCE + THE 1ST CHARACTER
                                        ;16690  ;OF THE OBJECT.
                                        ;16691
                                        ;16692  ;INPUTS:
                                        ;16693  ;Q = OBJECT LENGTH(1ST OPERAND)
                                        ;16694  ;D = OBJECT ADDR(2ND OPERAND)
                                        ;16695
                                        ;16696  ;AFTER INITIALIZTION, REGISTER USAGE DURING EXECUTION IS:
                                        ;16697  ;R0      OBJECT LENGTH - 1
                                        ;16698  ;R1      OBJECT ADDR
                                        ;16699  ;R2      OBJECT LENGTH - SOURCE LENGTH -1
                                        ;16700  ;R3      SOURCE ADDR
                                        ;16701  ;RC0     INNER LOOP COUNTER. STARTS OUT = -R0
                                        ;16702  ;        USED TO COMPARE CHARACTERS 2 THRU END OF OBJECT
                                        ;16703
                                        ;16704  ;OUTPUTS:
                                        ;16705  ;R0 = 0 IF MATCH, OTHERWISE NUMBER OF BYTES REMAINING IN OBJ WHEN
                                        ;16706  ;        SRC EXHAUSTED
                                        ;16707  ;R1 = ADDR OF END OF OBJ + 1 IF MATCH, OTHERWISE ADDR OF NEXT BYTE
                                        ;16708  ;        OF OBJ
                                        ;16709  ;R2 = NUMBER OF BYTES REMAINING IN SRC IF MATCH, OTHERWISE 0
                                        ;16710  ;R3 = ADDR OF LAST BYTE MATCHED + 1, OR ADDR OF END OF SRC + 1
                                        ;16711
                                        ;16712  48A:    ;-------------------------------;
                                        ;16713          CALL,J/SPEC,                    ;GET SRC LENGTH(ARG 3)
U 048A, 0019,2035,C180,F980,0000,037E   ;16714          R[T0]_Q.AND.K[.FFFF]            ;SAVE OBJ LEN(ARG 1)
                                        ;16715
                                        ;16716  49A:    ;-------------------------------;
U 049A, 0001,203C,0180,F988,0000,0114   ;16717          R[T1]_Q                         ;SAVE OBJ ADDR(ARG 2)
                                        ;16718
                                        ;16719  =00*****;-------------------------------;
                                        ;16720          CALL,J/ASPC,                    ;GET SRC ADDR(ARG 4)
                                        ;16721          D_Q,                            ;COPY OBJ ADDR TO RETURNED IN Q
U 0114, 0C19,0035,C180,F990,0000,047E   ;16722          R[T2]_D.AND.K[.FFFF]            ;SAVE SRC LEN(ARG 3)
                                        ;16723
                                        ;16724  =11*****;-------------------------------;
                                        ;16725          R[R3]_D,                        ;SAVE SRC ADDR
U 0174, 0C01,003C,0180,FA98,0000,0410   ;16726          D_Q                             ;COPY OBJ ADDR
```

```
                                        ;16727  =0****00;--------------------------------;SUITABLE FOR CLRPSLCC + SETFPD
                                        ;16728          R[R1]_D,                        ;COPY OBJ ADDR
                                        ;16729          Q_ID[PSL],                      ;GET PSL FOR CLRPSLCC
U 0410, 0001,003D,3DF0,2E88,0000,09E4   ;16730          CALL,J/CLRPSLCC                 ;CLEAR PSL CC
                                        ;16731
                                        ;16732  =1****00;--------------------------------;
                                        ;16733          D_RC[T0],CLK.UBCC,              ;CHECK ON OBJ LEN
U 0450, 0810,0039,0180,F900,0010,0E16   ;16734          CALL,J/SETFPD                   ;SET FPD BIT IN PSL
                                        ;16735
                                        ;16736          ;--------------------------------;
U 0451, 0000,003C,0580,F800,0104,6A8D   ;16737          J/MATFPD, FE_K[.1]              ;FPD RESTART ADRR
                                        ;16738
                                        ;16739          ;--------------------------------;
U 0452, 0000,003C,0580,F800,0104,6A8D   ;16740          J/MATFPD, FE_K[.1]              ;FPD RESTART ADRR
                                        ;16741
                                        ;16742          ;--------------------------------;
                                        ;16743          Z?,                             ;IF OBJ LEN = 0, ALL DONE(MATCH)
                                        ;16744          Q_D,                            ;COPY OBJ LEN
U 0453, 0810,0138,01E0,F910,0010,066C   ;16745          D_RC[T2],CLK.UBCC               ;CHECK ON SRC LENGTH
                                        ;16746
                                        ;16747  =0       ;--------------------------------;ALU<Z>
                                        ;16748          D_Q-D-1,CLK.UBCC,               ;OBJ LEN-SRC LEN-1 = MAXIMUM NUMBER
                                        ;16749          R[R2]_Q-D-1,                    ;BYTES TO COMPARE IN OUTER LOOP
U 066C, 0810,2108,0180,FA90,0010,0678   ;16750          Z?,J/MAT1                       ;BRANCH ON SRC LEN = 0
                                        ;16751
                                        ;16752          ;1--------------------------------;ALU = 0
U 066D, 0001,003C,0180,FA90,0000,06AC   ;16753          J/MATDONEMATCH1,R[R2]_D         ;OBJ LEN = 0
                                        ;16754                                          ;R0 = UNDEF, R1 = START OBJ ADDR
                                        ;16755                                          ;R3 = START SRC ADDR
                                        ;16756  =0       ;--------------------------------;ALU<Z>
                                        ;16757  MAT1:    R[R0]_Q-K[.1],                  ;INNER LOOP COUNTER
U 0678, 0019,2300,0580,FA80,0000,07C1   ;16758          C31?,J/MATCHSTART               ;ANYTHING TO COMPARE?
                                        ;16759
                                        ;16760          ;1--------------------------------;ALU = 0
                                        ;16761          J/R2ZERO,R[R0]_Q,               ;SRC LEN = 0
U 0679, 0001,203C,0187,FA80,0000,09F1   ;16762          SGN/CLR.SD+SS                   ;CLEAR "WRITE REG" FLAG
                                        ;16763                                          ;R0 = OBJ LEN, R1 = START OBJ ADDR,
                                        ;16764                                          ;R2 = OL-SL-1, R3 = START SRC ADDR
                                        ;16765          ;--------------------------------;
```

```
                                        ;16766  .TOC    ""       Character string      : MATCHC OUTER LOOP""
                                        ;16767
                                        ;16768  ;ALL INITIALIZATION HAS BEEN DONE. PREPARE TO READ THE
                                        ;16769  ;FIRST CHARACTER OF THE OBJECT + THE NEXT(1ST IF AT BEGINNING)
                                        ;16770  .CHARACTER OF THE SOURCE
                                        ;16771  ;R0 = OBJ LEN -1
                                        ;16772  ;R1 = START OBJ ADDR
                                        ;16773  ;R2 = OBJ LEN - SRC LEN -1
                                        ;16774  ;R3 = START SRC ADDR
                                        ;16775
                                        ;16776  =01     ;--- ----------------------------;ALU<C>
                                        ;16777  MATCHSTART:                             ;
                                        ;16778          VA_R[R1],                       ;ADDR OF 1ST CHAR OF OBJ
U ^7C1, 0000,003C,0180,F^08,0200,0A6C   ;16779          J/MATOUTERLOOP                  ;
                                        ;16780
                                        ;16781          ;11---------------------------;ALU <C> = 1
                                        ;16782          Q_RC[T2],                       ;SRC LEN < OBJ LEN
U 07C3, 0010,0038,01C0,F910,0000,0A84   ;16783          J/MATDONENOMATCH2               ;R0 = OBJ LEN - 1, R1 = START OBJ ADDR
                                        ;16784                                          ;R2 = OL-SL-1, R3 = NEXT SRC ADDR
                                        ;16785          ;-------------------------------;
                                        ;16786  MATOUTERLOOP:                           ;
U 0A6C, 0000,803C,1980,4000,1404,6A6D   ;16787          D[BYTE]_CACHE,STATE_OUTER       ;READ 1ST CHAR OF OBJECT
                                        ;16788
                                        ;16789          ;-------------------------------;
                                        ;16790  MATOUT1:                                ;
                                        ;16791          VA_R[R3],                       ;READ NEXT(MAY BE 1ST) CHAR OF SRC
                                        ;16792          Q_D,                            ;COPY 1ST CHAR OF OBJ
U 0A6D, 0000,003C,01E0,FA18,4200,0A70   ;16793          INTRPT.STROBE                   ;CHECK FOR INTERRUPTS
                                        ;16794
                                        ;16795          ;-------------------------------;
                                        ;16796  MATOUT2:                                ;
                                        ;16797          D[BYTE]_CACHE,                  ;READ 1 BYTE
U 0A70, 0000,8E3C,0180,4000,00C0,0896   ;16798          BEN/INTERRUPT                   ;INTERRUPT PENDING?
                                        ;16799
                                        ;16800  =110    ;-------------------------------;INTERRUPT?
                                        ;16801          ALU_D.XOR.Q,CLK.UBCC,DT/BYTE,   ;COMPARE BYTE OF SRC WITH 1ST BYTE OF OBJ
U 0896, 001D,8020,0180,FA10,0010,0A71   ;16802          LAB_R[R2],J/MATOUT3             ;LATCH OUTER LOOP COUNTER
                                        ;16803
                                        ;16804          ;111--------------------------;YES, AN INTERRUPT
U ^897, 0000,003C,1980,F800,0104,6A8D   ;16805          J/MATFPD,FE_K[ZERO]             ;INTERRUPT PENDING
```

```
                                    ;16806  .PAGE
                                    ;16807
                                    ;16808          ;-------------------------------;
                                    ;16809  MATOUT3:                                ;
                                    ;16810          R[R2]_LA+K[.1],CLK.UBCC,        ;MORE TO DO?
U 0A71, 0018,0114,0580,FA90,0010,067C ;16811        Z?                              ;A MATCH?
                                    ;16812
                                    ;16813  =0      ;-------------------------------;ALU Z
                                    ;16814          LAB_R[3],                       ;NO MATCH
U 067C, 0000,013C,0180,FA18,0000,069C ;16815        J/MATUNEQ,Z?                    ;OUTER LOOP COUNTER = 0?
                                    ;16816
                                    ;16817          ;1-- ---------------------------;ALU = 0
                                    ;16818          J/MATEQ,                        ;A MATCH FOR 1ST CHAR OF OBJ FOUND
U 067D, 0800,003C,0180,FA00,0000,0A74 ;16819        D_R[R0]                         ;GET OBJ LEN - 1
                                    ;16820
                                    ;16821  =0      ;-------------------------------;ALU Z
                                    ;16822  MATUNEQ:                                ;
                                    ;16823          VA_LA+K[.1],R[R3]_LA+K[.1],     ;MORE TO DO. INCREMENT SRC ADD
U 069C, 0018,0014,0580,FA98,4200,0A70 ;16824        INTRPT.STROBE,J/MATOUT2         ;CHECK FOR INTERRUPTS
                                    ;16825
                                    ;16826          ;1-------------------------------;ALU = 0
                                    ;16827          Q_R[R0],                        ;NO MORE CHARACTERS TO TRY IN SRC
U 069D, 0000,003C,01C0,FA00,0000,0A80 ;16828        J/MATDONENOMATCH3               ;NO MATCH. R0 = OBJ LEN -1
                                    ;16829                                          ;R1 = START OBJ ADDR, R2 = 0,
                                    ;16830                                          ;R3 = NEXT SRC ADDR
                                    ;16831          ;-------------------------------;
```

```
                                ;16832  .TOC    ""      Character string      : MATCHC INNER LOOP""
                                ;16833
                                ;16834  MATEQ:  LAB_R1&RC[TO]_0-D,CLK.UBCC,     ;MAKE A DECREMENTING INNER LOOP COUNTER
U 0A74, 001F,2000,0180,FB00,0010,06A1   ;16835          J/MATIN1                ;
                                ;16836
                                ;16837  =0     ;-------------------------------;ALU Z
                                ;16838  MATIN2:                                 ;MISMATCH IN INNER LOOP.
                                ;16839          J/MATNOMATCH,                   ;BACK TO OUTER LOOP
U 06A0, 0000,003C,01C0,FA00,0000,0A7A   ;16840          Q_R[R0]                 ;LOAD ORIGINAL OBJ LEN - 1
                                ;16841
                                ;16842          ;1------------------------------;ALU = 0
                                ;16843  MATIN1: VA_LA+K[.1],R[R1]_LA+K[.1],     ;INCREMENT OBJ ADDR
U 06A1, 0018,0114,0580,FA88,0200,06A4   ;16844          Z?                      ;OBJ LEN = 0?
                                ;16845
                                ;16846  =0     ;-------------------------------;ALU Z
                                ;16847  MATIN5: D[BYTE]_CACHE,                  ;READ 1 BYTE OF OBJ
                                ;16848          LAB_R[R3],                      ;LATCH SRC ADDR
                                ;16849          STATE_INNEROBJ,                 ;
U 06A4, 0000,803C,0580,4218,1404,6A75   ;16850          J/MATIN3                ;
                                ;16851
                                ;16852          ;1------------------------------;ALU = 0
U 06A5, 0000,003C,0180,FA18,0000,0A89   ;16853          J/MATDONEMATCH,LAB_R[R3]        ;FOUND ALL OF OBJECT IN SRC
                                ;16854                                          ;R0 = OBJ LEN, R1 = LAST OBJ ADDR + 1
                                ;16855                                          ;R2 = CNTR, R3 = LAST SRC READ
                                ;16856          ;-------------------------------;
                                ;16857  MATIN3: R[R3]_LA+K[.1],VA_LA+K[.1],     ;INCREMENT SRC ADDR
                                ;16858          Q_D,                            ;COPY OBJ CHAR
U 0A75, 0018,0014,05E0,FA98,4200,0A78   ;16859          INTRPT.STROBE           ;CHECK FOR INTERRUPTS
                                ;16860
                                ;16861          ;-------------------------------;
                                ;16862          D[BYTE]_CACHE,                  ;READ 1 BYTE OF SRC
                                ;16863          STATE_INNERSRC,                 ;
U 0A78, 0000,8E3C,0D80,4000,1404,68B6   ;16864          BEN/!INTERRUPT          ;INTERRUPT PENDING?
                                ;16865
                                ;16866  =110    ;-------------------------------;INTERRUPT?
                                ;16867          ALU_Q.XOR.D,CLK.UBCC,DT/BYTE,   ;COMPARE BYTES
U 08B6, 001D,A020,0180,F900,0010,0A79   ;16868          LC_RC[TO],J/MATIN4      ;LATCH INNER LOOP COUNTER
                                ;16869
                                ;16870          ;111-----------------------------;
U 08B7, 0000,003C,1980,F800,0104,6A8D   ;16871          J/MATFPD,FE_K[ZERO]     ;GO SERVICE INTERRUPT
                                ;16872
                                ;16873          ;-------------------------------;
                                ;16874  MATIN4: LAB_R1&RC[TO]_0+LC+1,           ;INCREMENT INNER COUNTER
                                ;16875          CLK.UBCC,                       ;SEE IF IT'S NOW 0
                                ;16876          D_0+LC+1,                       ;SAVE -(# BYTES NOT EXAMINED IN
                                ;16877                                          ;INNER LOOP) IN CASE NO MATCH
U 0A79, 0813,0110,0180,FB00,0010,06A0   ;16878          Z?,J/MATIN2             ;A MATCH?
                                ;16879
                                ;16880          ;-------------------------------;
```

```
                                    ;16881  MATNOMATCH:
                                    ;16882  ;R0      ORIGINAL INNER LENGTH - 1
                                    ;16883  ;R1      LAST OBJ CHAR READ
                                    ;16884  ;R2      OBJLEN-SRCLEN-1= -(OUTER COUNTER) ORIGINALLY
                                    ;16885  ;R3      LAST SRC CHAR READ
                                    ;16886  ;RC 0    MINUS NUMBER BYTES LEFT IN INNER LOOP
                                    ;16887
U 0A7A, 081D,2014,01E0,FA08,0000,0A7C ;16888          D_Q+D, Q_D, LAB_R[R1]          ;ORIGNAL - CURRENT = # READ
                                    ;16889
                                    ;16890          ;-------------------------------;
                                    ;16891          R[R1]_LA-D,                    ;RESET OBJ ADDR TO START OF STRING
U 0A7C, 001C,2000,0180,FA88,0200,0A7D ;16892          VA_LA-D                        ;PREPARE TO RE-READ 1ST OBJ BYTE
                                    ;16893
                                    ;16894          ;-------------------------------;
U 0A7D, 0000,403C,0180,FA10,0010,0A7E ;16895          ALU_R[R2],CLK.UBCC,DT/WORD     ;SEE IF ANY LEFT TO DO IN OUTER LOOP
                                    ;16896                                         ;THIS IS DT/W FOR THE CASE WHERE
                                    ;16897                                         ;R2 = 0 UPON ENTERING INNER.
                                    ;16898                                         ;IF FPD, THEN R2_FFFF0000
                                    ;16899                                         ;AT RESTART(FROM ORNOT FFFF).
                                    ;16900          ;-------------------------------;
                                    ;16901          LAB_R[R3], D_D-K[.1],          ;DECREMENT SRC BY 1
U 0A7E, 0819,0100,0580,FA18,0000,06A8 ;16902          Z?                             ;BRANCH ON ANY LEFT IN OUTER
                                    ;16903
                                    ;16904  =0      ;-------------------------------;ALU Z
                                    ;16905          R[R3]_LA-D,                    ;RESET SRC ADDR SO READ NEXT BYTE
U 06A8, 001C,2000,0180,FA98,000C,0A6C ;16906          J/MATOUTERLOOP                 ;I.E. NEXT ONE IN OUTER LOOP CONTEXT
                                    ;16907
                                    ;16908          ;1-------------------------------;
U 06A9, 001F,0000,01C0,F800,0000,0A80 ;16909          Q_0-Q, J/MATDONENOMATCH3       ; Q SET UP TO BUMP R3 PAST SRC END
```

```
                                    ;16910  .TOC    ''       Character string      : MATCHC TERMINATION''
                                    ;16911
                                    ;16912  ;IF NO MATCH, EXIT WITH:
                                    ;16913  ;R 0    # BYTES LEFT IN OBJ WHEN SRC LEN = 0
                                    ;16914  ;R 1    ADDR OF NEXT BYTE OF OBJ TO EXAM WHEN SRC LEN = 0
                                    ;16915  ;R 2    0
                                    ;16916  ;R 3    END OF SRC + 1
                                    ;16917
                                    ;16918  MATDONENOMATCH3:                          ;ALU = 0
                                    ;16919          LAB_R[R3],Q_Q+K[.1],              ;Q = ORIGINAL OBJECT LENGTH
U 0A80, 0019,2014,05C0,FA18,0000,0A81 ;16920        J/MATDONENOMATCH                  ;
                                    ;16921
                                    ;16922          ;--------------------------------;
                                    ;16923  MATDONENOMATCH:
                                    ;16924          R[R3]_LA+Q,                       ;NOTHING LEFT TO LOOK AT IN OUTER
U 0A81, 001C,0014,0180,FA98,0000,0A85 ;16925        J/MATDONENOMATCH1                 ;R0 = OBJ LEN -1, R1 = START OBJ ADDR
                                    ;16926                                            ;R2 = 0, R3 = NEXT SRC ADDR
                                    ;16927          ;--------------------------------;
                                    ;16928  MATDONENOMATCH2:                          ;
U 0A84, 0000,003C,0180,FA18,0000,0A81 ;16929        LAB_R[R3],J/MATDONENOMATCH        ;
                                    ;16930
                                    ;16931          ;--------------------------------;
                                    ;16932  MATDONENOMATCH1:                          ;
U 0A85, 0000,003C,0180,FA00,0000,0A88 ;16933        LAB_R[R0]                         ;
                                    ;16934
                                    ;16935          ;--------------------------------;
                                    ;16936          R[R0]_LA+K[.1],                   ;ORIG OBJ LENGTH
                                    ;16937          SGN/C[R.SD+SS,                    ;CLEAR 'WRITE REG'' FLAG
U 0A88, 0018,0014,0587,FA80,0000,09F1 ;16938        J/R2ZERO                          ;
```

```
                                ;16939          ;-------------------------------;
                                ;16940  ;IF A MATCH, EXIT WITH:
                                ;16941  ;R 0    0
                                ;16942  ;R 1    END OF OBJ + 1
                                ;16943  ;R 2    # BYTES LEFT IN SRC
                                ;16944  ;R 3    LAST MATCHING BYTE ADDR + 1
                                ;16945
                                ;16946  MATDONEMATCH:
U 0A89, 0018,0014,0580,FA98,0000,0A8A   ;16947          R[R3]_LA+K[.1]                  ;POINT PAST END OF STRING
                                ;16948
                                ;16949          ;-------------------------------;
                                ;16950          D_R[R2],                        ;NEGATE R2(MAKE IT POSITIVE)
                                ;16951          Q_0,                            ;TO BE # BYTES SRC LEFT
U 0A8A, 0800,403C,01F8,FA10,0010,0A8C   ;16952          C[K.UBCC,DT/WORD                ;FOR THE CASE WHERE STRINGS ARE
                                ;16953                                          ;SAME LENGTH, IF A FPD WAS HANDLED,
                                ;16954                                          ;THEN R2_FFFF0000 INSTEAD OF 00000000
                                ;16955                                          ;SO DETECT THAT NOW
                                ;16956          ;-------------------------------;
                                ;16957          R[R2]_Q-D,                      ;EFFECT Q-D TO RESET COUNTER
U 0A8C, 001D,2100,0180,FA90,0000,06AC   ;16958          Z?                              ;SEE IF R2 = 0
                                ;16959
                                ;16960  =0      ;-------------------------------;ALU <Z>
                                ;16961  MATDONEMATCH1:
                                ;16962          R[R0]_0,N&Z_ALU.V&C_0,          ;SET R0 = 0, + SET PSL Z
U 06AC, 0003,003C,0180,FA80,2050,05AE   ;16963          CLR.FPD,J/STRINGFINAL           ;CLEAR FPD BIT IN PSL
                                ;16964
                                ;16965          ;1------------------------------;ALU = 0
                                ;16966          R[R2]_0,                        ;R2 = 0
U 06AD, 0003,003C,0180,FA90,0000,06AC   ;16967          J/MATDONEMATCH1                 ;
                                ;16968
                                ;16969          ;-------------------------------;
```

```
                                    ;16970  .TOC    ""      Character string      : MATCHC FPD + RESTART''
                                    ;16971
                                    ;16972          ;-------------------------------;
                                    ;16973  MATFPD:                                 ;
                                    ;16974          D_RC[TO],                       ;SAVE CURRENT INNERLOOP COUNTER
                                    ;16975          SC_K[.10],                      ;16(10) FOR DAL
U 0A8D, 0810,0038,65F8,F900,0084,6A8E ;16976        Q_0                             ;ROTATE IN 0S DURING SHIFT
                                    ;16977
                                    ;16978          ;-------------------------------;
                                    ;16979          D_DAL.SC,                       ;D<31:16>=RC(TO)
U 0A8E, 0D18,0034,C1C0,FA10,0000,0A90 ;16980        Q_R[R2].AND.K[.FFFF]            ;PRESERVE LOW WORD
                                    ;16981
                                    ;16982          ;-------------------------------;
                                    ;16983          R[R2] D.OR.Q,                   ;SAVE BOTH COUNTERS
U 0A90, 001D,0030,0180,FA90,0000,06D8 ;16984        J/FPDPACK                       ;GO GET STATE + PC DELTA
                                    ;16985
                                    ;16986          ;-------------------------------;
                                    ;16987
                                    ;16988  MATCHCUNSCRAMBLE:
                                    ;16989  ;RESTART CODE INITIALLY SHARED WITH EDITPC. ONCE HERE:
                                    ;16990  ;OUTER:          JUST RE-READ AT CURRENT ADDRS
                                    ;16991  ;INNEROBJ:       DECREMENT R3 THEN
                                    ;16992  ;INNERSRC:       RE-READ R1
                                    ;16993
                                    ;16994          D_DAL.SC,                       ;UNPACK R2
                                    ;16995          STATE_STATE.ANDNOT.K[.FFFC],    ;PROTECT AGAINST UNDEFINED BITS
U 0A91, 0D00,003C,F1C0,FA18,1404,4800 ;16996        Q_R[R3]                         ;LOAD SRC ADDR
                                    ;16997
                                    ;16998  =00     ;-------------------------------;RETURN3
                                    ;16999          D_Q,                            ;MOVE SRC ADDR TO D
                                    ;17000          RC[TO] D.ORNOT.K[.FFFF],        ;RESTORE -(# BYTES LEFT IN INNER LOOP)
U 0800, 0C19,001D,C180,F980,0000,0E16 ;17001        CALL,J/SETFPD                   ;SET FPD BIT
                                    ;17002
                                    ;17003          ;-------------------------------;
U 0801, 0000,003C,0180,F800,0000,0A8D ;17004        J/MATFPD                        ;SPECIFY FPD ADDR
                                    ;17005
                                    ;17006          ;-------------------------------;
U 0802, 0000,003C,0180,F800,0000,0A8D ;17007        J/MATFPD                        ;SPECIFY FPD ADDR
                                    ;17008
                                    ;17009          ;-------------------------------;
                                    ;17010          VA_R[R1],                       ;PREPARE TO REREAD OBJ
U 0803, 0000,173C,0180,FA03,0200,0808 ;17011        BEN/STATE3-0                    ;BRANCH ACCORDING TO WHERE INTERRUPTED
                                    ;17012
                                    ;17013  ; ************************************************
                                    ;17014  ; * Patch no. 004, PCS 0803 trapped to WCS 1143 *
                                    ;17015  ; ************************************************
```

```
                                    ;17016  =**00   ;-------------------------------;<1:0> OF STATE
                                    ;17017          D[BYTE] CACHE,                  ;RE-READ OUTER BYTE
                                    ;17018          STATE_OUTER,                    ;WAS IN OUTER LOOP
U 0808, 0000,803C,1980,4000,1404,6A6D ;17019        J/MATOUT1                       ;
                                    ;17020
                                    ;17021          ;**01---------------------------;
                                    ;17022          D[BYTE] CACHE,                  ;RE-READ OBJ BYTE
                                    ;17023          STATE_INNEROBJ,                 ;WAS READING OBJ IN INNER LOOP
                                    ;17024          LAB_R[R3],                      ;
U 0809, 0000,803C,0580,4218,1404,6A75 ;17025        J/MATIN3                        ;
                                    ;17026
                                    ;17027  =**11   ;-------------------------------;
                                    ;17028          R[R3] D-K[.1],                  ;DECREMENT SRC ADDR
U 080B, 0019,0000,0580,FA98,0000,06A4 ;17029        J/MATIN5                        ;
```

```
                                    ;17030  .TOC    ""      Character string      : MOVTC, MOVTUC""
                                    ;17031
                                    ;17032  ;       THIS ROUTINE MAKES HEAVY USE OF MOVC FLOWS IN ITS SETUP,
                                    ;17033  ;       FAULT/INTERRUPT HANDLING, AND RESTART OPERATIONS.  IN ORDER TO
                                    ;17034  ;       DO THIS IT WAS NECESSARY TO KEEP THE DESTINATION STRING
                                    ;17035  ;       LENGTH AND ADDRESS IN R2/R3 DURING THE MOVE, AND TRANSFER
                                    ;17036  ;       IT TO R4/R5 AT THE END. THIS WASTES A FEW CYCLES (ABOUT 5)
                                    ;17037  ;       BUT REDUCES THE CODE BY ABOJT 20 WORDS.
                                    ;17038  ;
                                    ;17039  ;       REGISTERS ARE SET UP JUST LIKE MOVC, EXCEPT THAT INSTEAD OF
                                    ;17040  ;       R2<31:16> CONTAINING TWO FILL CHARS, R2<31:24> CONTAINS ONE
                                    ;17041  ;       COPY OF THE FILL/ESCAPE CHAR AND R2<23:16>=0.
                                    ;17042  ;       R4, WHICH IS NOT USED AT ALL BY MOVC, IS USED TO STORE THE
                                    ;17043  ;       TRANSLATION TABLE ADDRESS.
                                    ;17044  ;
                                    ;17045  ;       STATE BIT ASSIGNMENTS ARE LIKEWISE THE SAME AS MOVC, EXCEPT
                                    ;17046  ;       THAT STATE<7> IS ON (TO INDICATE MOVE TRANSLATED) AND
                                    ;17047  ;       STATE<6> MEANS MERELY 'DESTINATION LARGER THAN SRC''
                                    ;17048  ;       WITHOUT THE IMPLIED MOVC MEANING OF 'NEED TO FILL DESTINATION''
                                    ;17049  ;       (MOVTC FILLS, MOVTUC DOESN'T).
                                    ;17050
                                    ;17051
                                    ;17052  ;MOVTC/MOVTUC ARGUMENT FETCH - BOTH COME HERE FROM C FORK.
                                    ;17053
                                    ;17054  3C3:
U 03C3, 0003,603D,0180,F980,0000,037E ;17055  MOVTC:  RC[TO]_Q.OXT[WORD], CALL[SPEC]  ; SAVE SRC LEN, GET FILL/ESC CHAR
                                    ;17056          ;-------------------------------;
U 03D3, 0001,203C,0180,F988,0000,0023 ;17057  3D3:    RC[T1]_Q                        ; SAVE SRC ADDR
                                    ;17058
                                    ;17059          ;-------------------------------;
                                    ;17060  =0****01*****                           ;CONSTRAINT FOR ASPC & MOVCCMPC5
U 0023, 0000,003D,0180,F800,0000,047E ;17061          CALL[ASPC]                      ;GET TABLE ADDR, Q GETS FILL
                                    ;17062
                                    ;17063          ;-------------------------------;
                                    ;17064  =0****11*****                           ;RETURN FROM ASPC
                                    ;17065          RC[T3]_D, D_Q, Q_0, SC_K[.18],  ; SAVE TBL ADR, SET UP TO ISOLATE
U 0063, 0C01,003D,7DF8,F998,0084,60A2 ;17066                  CAL[[MOVCCMPC5]         ;FILL IN D<31:24>, GO GET DST SPEC
                                    ;17067
                                    ;17068          ;-------------------------------;
                                    ;17069  =1****11*****                           ;RETURN FROM MOVCCMPC5
U 0863, 0000,003C,6580,F918,0104,6A92 ;17070          LC_RC[T3], FE_K[.10]            ;GET TABLE ADDR, SET FE='BACKWARDS'' FLAG
                                    ;17071
                                    ;17072          ;-------------------------------;
                                    ;17073  =       ;END OF DOUBLE CALL CONSTRAINT BLOCK
                                    ;17074
                                    ;17075          ;-------------------------------;
U 0A92, 0010,0038,0180,FAA0,0000,0C5E ;17076          R[R4]_LC, J/MOVC5SETUP          ;STORE TABLE ADDR & GO BACK TO MOVC
```

```
                                        ;17077  ;       BACK HERE FROM MOVC. REGISTERS SET UP AS FOLLOWS:
                                        ;17078  ;       R1 = SRC ADDR
                                        ;17079  ;       R2<31:24>=FILL/ESC, R2<23:16>=0, R2<15:0>=DESTLEN-SRCLEN-1
                                        ;17080  ;       R3 = DEST ADDR
                                        ;17081  ;       R4 = TRANSLATE TABLE ADDR
                                        ;17082  ;       D = "F80" (FAULT VECTOR ADDRESS)
                                        ;17083  ;       Q = 0
                                        ;17084  ;       LA, LB = R1
                                        ;17085  ;       LC = MIN(SRC LEN, DEST LEN)
                                        ;17086  ;       SC = FE = 10
                                        ;17087  ;       STATE<6> = (DEST LEN > SRC LEN)
                                        ;17088  ;       C31 SET IF SRC ADDR < DEST ADDR
                                        ;17089  ;       FIRST PART DONE SET BUT VECTOR NOT LOADED.
                                        ;17090
                                        ;17091  =1100   ;-------------------------------;C31 AND IR0 (BACKWARDS AND MOVTUC)
                                        ;17092  MOVTCWHATDIR:
U 063C, 0001,2028,B5C0,3C00,0000,0688   ;17093          ID[FPDA]_D, Q_NOT.Q, J/MOVTC.1  ;MOVTC FORWARDS - Q = -1
                                        ;17094
                                        ;17095          ;1101---------------------------;
                                        ;17096          ID[FPDA]_D, Q_LC, R[R5]_LC,     ;MOVTC BACKWARDS - Q = LOOPCT,
U 063D, 0010,0038,B5C0,3EA8,0100,A688   ;17097          FE_SC-FE, J/MOVTC.1             ;CLEAR BKWDS FLAG IN FE.
                                        ;17098
                                        ;17099          ;1110---------------------------;
U 063E, 0001,2028,B5C0,3C00,0000,0688   ;17100          ID[FPDA]_D, Q_NOT.Q, J/MOVTC.1  ;MOVTUC FORWARDS - Q = -1
                                        ;17101
                                        ;17102          ;1111---------------------------;
U 063F, 0001,2028,B5C0,3C00,0000,0688   ;17103          ID[FPDA]_D, Q_NOT.Q, J/MOVTC.1  ;MOVTUC BACKWARDS - DO IT FORWARDS.
```

```
                                        ;17104  =0*     ;-------------------------------;CALL CONSTRAINT FOR MOVCRBUMP
                                        ;17105  MOVTC.1:
                                        ;17106          LA_RA[R3], D_LC, CLK.UBCC,      ;GET LOOPCT IN D & SET Z ON IT,
                                        ;17107          STATE_STATE.OR.K[.80],          ;SET MOVTC FLAG IN STATE FOR RESTART
U 0688, 0810,0039,4180,F898,1414,25C4   ;17108          CALL[MOVCRBUMP]                 ;BUMP R1,R3 BY -1 OR LOOPCT
                                        ;17109
                                        ;17110          ;1*-----------------------------;
                                        ;17111          ID[TO]_D, D_R[R2], Q_0,          ;SAVE LOOPCT, GET FILL IN D<31:24>
U 068A, 0800,0D3C,C1F8,3E10,1400,432A   ;17112          STATE_STATE.ANDNOT.FE, Q31?     ;COND. CLEAR BKWD FLAG (MOVCRBUMP
                                        ;17113                                          ;SETS IT) AND TEST DIRECTION
                                        ;17114  =01*    ;-------------------------------;BRANCH ON Q31 (MOVING FORWARD)
                                        ;17115  MOVTCBKWD:
                                        ;17116          RC[TO]_NOT.Q, D_D.SWAP,         ;BACKWARD - INCR = -1
U 032A, 0B01,2028,0180,F980,0000,0A94   ;17117          J/MOVTCGO
                                        ;17118
                                        ;17119          :11*----------------------------;
                                        ;17120  MOVTCFW:
U 032E, 0B1F,0010,0180,F980,0000,0A94   ;17121          RC[TO]_Q+1, D_D.SWAP, J/MOVTCGO ;FORWARD - INCR = 1
                                        ;17122
                                        ;17123  MOVTCGO:;-------------------------------;
                                        ;17124          ALU_D.OXT[BYTE], SC_ALU,        ;GET FILL/ESC CHAR IN SC,
U 0A94, 0003,813C,C1F0,2D00,0082,06B4   ;17125          LC_RC[TO], Q_ID[TO], Z?         ;LC = INCR, Q = LOOPCT, ENTER LOOP
                                        ;17126  =0
                                        ;17127  MOVTCLP:;-------------------------------;BRANCH ON Z (LOOP COUNT EXHAUSTED)
                                        ;17128          LAB_R1&RC[T2]_Q-K[.1], Q_ALU,   ;DECREMENT LOOPCT & SAVE A COPY,
                                        ;17129          D_ALU, CLK.UBCC, STATE5?,       ;TEST IF FILLING.
U 06B4, 0819,3600,05C0,FB10,0010,064D   ;17130          J/MOVTCLP.1
                                        ;17131
                                        ;17132  ; ************************************************
                                        ;17133  ; * Patch no. 049, PCS 0684 trapped to WCS 1177 *
                                        ;17134  ; ************************************************
                                        ;17135
                                        ;17136          ;1-------------------------------;
                                        ;17137          LAB_R[R1], Q_K[.1], STATE4?,    ;LOOP OVER - SET UP TO ADJUST
U 06B5, 0018,1638,05C0,FA08,0000,06AE   ;17138          J/MOVTCLPDONE                   ;REGISTERS AND TEST FILL FLAG
                                        ;17139
                                        ;17140  =1101   ;-------------------------------;BRANCH ON STATE<5> (FILLING)
                                        ;17141  MOVTCLP.1:
                                        ;17142          R[R1]_LA+LC, VA_ALU, ID[TO]_D,  ;INCR SRC ADDR, LOAD VA,
U 064D, 0010,0014,C180,3E88,0200,0A95   ;17143          J/MOVTCLP.2                     ;SAVE LOOP COUNT.
                                        ;17144
                                        ;17145          ;1111---------------------------;
U 064F, 0818,0038,1D80,F898,0000,08C5   ;17146          D_K[SC], LA_RA[R3], J/MOVTCLP.3 ;GET FILL IN D, GO WRITE IT
                                        ;17147
                                        ;17148          ;-------------------------------;
                                        ;17149  MOVTCLP.2:
U 0A95, 0000,803C,01C0,40A0,0000,0A96   ;17150          D[BYTE]_CACHE, LA_RA[R4], Q_LA  ;READ SRC BYTE, GET TABLE ADR IN Q
                                        ;17151
                                        ;17152          ;-------------------------------;
                                        ;17153          VA_D.OXT[BYTE]+Q, Q_ID[TO],     ;INDEX INTO TABLE, RESTORE LOOPCT,
U 0A96, 001F,8014,C1F0,2C98,0200,0A98   ;17154          LA_RA[R3]
                                        ;17155
                                        ;17156          ;-------------------------------;
U 0A98, 0000,9B3C,0180,4000,0000,08C5   ;17157          D[BYTE]_CACHE, IR0?             ;READ TRANSLATED BYTE, BRANCH ON OPCODE
```

```
                                        ;17158  =*101   ;-------------------------------;BRANCH ON IRO (MOVTUC) (N ALWAYS 0)
                                        ;17159  MOVTCLP.3:
                                        ;17160          R[R3]_LA+LC, VA_ALU, INT?,      ;MOVTC - INCR DEST ADDR,
U 08C5, 0010,0E14,0180,FA98,0200,08E6   ;17161          J/MOVTCLP.5                     ;LOAD VA AND TEST FOR INTERRUPTS
                                        ;17162
                                        ;17163          ;*111-------------------------;
                                        ;17164          SC_D.OXT[BYTE].XOR.K[SC],       ;MOVTUC - COMPARE BYTE TO ESC CHAR
U 08C7, 001B,8E20,1D80,F800,0182,08D6   ;17165          FE_SC, INT?                     ;AND CHECK FOR INTERRUPTS
                                        ;17166
                                        ;17167  =110    ;-------------------------------;BRANCH ON INTERRUPTS (PENDING)
                                        ;17168          R[R3]_LA+LC, VA_ALU, SC_FE,     ;INCR DEST ADDR & LOAD VA,
U 08D6, 0010,1414,0180,FA98,0281,0811   ;17169          SC.GT.0?, J/MOVTCLP.4           ;RESTORE ESCAPE CHAR, TEST FOR MATCH
                                        ;17170
                                        ;17171          ;111--------------------------;
U 08D7, 0000,003C,0180,FA98,0000,0F82   ;17172          R[R3]_LA, J/MOVC.RDFAULT        ;INTERRUPT - LET MOVC HANDLE IT
                                        ;17173
                                        ;17174  =*01    ;-------------------------------;BRANCH ON SC .GT. 0 (NO MATCH)
                                        ;17175  MOVTCLP.4:
                                        ;17176          D_R[R2].ORNOT.K[.FFFF],         ;GET LENGTH DIFF IN D WITH HIGH
U 0811, 0818,001C,C180,FA10,0000,0AA2   ;17177          J/MOVTUCESCAPE                  ;ORDER 1'S, GO ADJUST REGS & EXIT
                                        ;17178
                                        ;17179          ;*11--------------------------;
U 0813, 0000,813C,0180,3000,0000,06B4   ;17180          CACHE_D[BYTE], Z?, J/MOVTCLP    ;NO MATCH - WRITE BYTE AND LOOP.
                                        ;17181
                                        ;17182  =110    ;-------------------------------;BRANCH ON INTERRUPTS (PENDING) (MOVTC)
                                        ;17183  MOVTCLP.5:
U 08E6, 0000,813C,0180,3000,0000,06B4   ;17184          CACHE_D[BYTE], Z?, J/MOVTCLP    ;NO INTERRUPT - WRITE BYTE & LOOP
                                        ;17185
                                        ;17186          ;111--------------------------;
U 08E7, 0000,003C,0180,FA98,0000,0F82   ;17187          R[R3]_LA, J/MOVC.RDFAULT        ;INTERRUPT - JOIN COMMON CODE
                                        ;17188
                                        ;17189          ;-------------------------------;
```

```
                                ;17190  .TOC    ''      Character string      : MOVTC/MOVTUC LOOP EXITS''
                                ;17191
                                ;17192  =1110   ;-------------------------------;BRANCH ON STATE<4> (MOVING BACKWARDS)
                                ;17193  MOVTCLPDONE:                            ; FORWARDS - BUMP R1&R3 BY 1
                                ;17194          LA_RA[R3], ALU_0+K[.1],         ;SET UP LATCHES FOR MOVCRBUMP,
U 06AE, 001B,0014,0580,F898,0010,068C   ;17195          CLR.UBCC, J/MOVTCUNBMP          ;CLEAR ALU CC'S
                                ;17196
                                ;17197          ;1111-------------------------;
U 06AF, 0000,003C,01C0,F8A8,0000,06AE   ;17198          LA_RA[R5], Q_LA, J/MOVTCLPDONE  ; BACKWARDS - BUMP BY LOOP CT
                                ;17199
                                ;17200  =0*     ;-------------------------------;CALL CONSTRAINT BLOCK FOR MOVCRBUMP
                                ;17201  MOVTCUNBMP:
                                ;17202          STATE_STATE.ANDNOT.K[.B0],      ;CLEAR OUT ALL BITS BUT 'DEST>SRC'',
U 068C, 0000,003D,9580,F800,1404,45C4   ;17203          CALL[MOVCRBUMP]                 ;INCR R1&R3 BY 1 OR LOOPCT
                                ;17204
                                ;17205          ;1*---------------------------;
                                ;17206          D_R[R2].ORNOT.K[.FFFF],         ;GET LENGTH DIFF WITH HIGH 1'S
U 068E, 0818,161C,C180,FA10,0000,0031   ;17207          STATE6?                 ;TEST IF DEST > SRC
                                ;17208
                                ;17209  =*0*1   ;-------------------------------;BRANCH ON STATE<6> (DEST > SRC)
                                ;17210                                          ; (STATE<4> SET BY MOVCRBUMP)
                                ;17211          R[R0]_NOT.D, D_0,               ;SAVE SRC EXCESS IN R0,
U 0031, 0F01,0028,0180,FA80,2000,0A99   ;17212          CLR.FPD, J/MOVTCEXIT            ;SWAP REGISTERS AND EXIT
                                ;17213
                                ;17214          ;*1*1-------------------------;
                                ;17215          LAB_R1&RC[T0]_D.0XT[WORD]+K[.1],;GET DEST EXCESS IN D & RC[T0],
U 0035, 081B,5B14,05F8,FB00,0000,0690   ;17216          D_ALU, Q_0, IR0?                ;CHECK MOVTC OR MOVTUC
                                ;17217
                                ;17218  =**0*   ;-------------------------------;IR1 (MOVTUC) (ALU CC'S ALL CLEAR)
                                ;17219          R[R2]_NOT.Q, DT/WORD,           ;SET R2<15:0> = FFFF, CLEAR
U 0690, 0001,6028,7580,FA90,1404,AAA1   ;17220          STATE_STATE-K[.20], J/MOVTCFILL ;'DEST>SRC'', SET 'FILLING''
                                ;17221
                                ;17222          ;**1*-------------------------;
U 0692, 0001,203C,0180,FA80,2000,0A99   ;17223          R[R0]_Q, CLR.FPD                ;NO S.C EXCESS, ZERO R0
                                ;17224
                                ;17225          ;-------------------------------;
                                ;17226  MOVTCEXIT:                              ;COMMON CLEANUP CODE, D= DEST LEN
U 0A99, 2014,0038,0180,F899,4200,0A9A   ;17227          LA_RA[R3], PC&VA_PC, FLUSH.IB   ;IB MAY BE BAD IF WE WERE FAULTED
                                ;17228                                          ;** FPD MUST BE CLEAR AT THIS POINT **
                                ;17229          ;-------------------------------;
U 0A9A, 0000,003C,0180,62AC,0000,0A9C   ;17230          R[R5]_LA, PC_PC+1, LOAD.IB      ;SET R5 = DEST ADDR, START FETHING
                                ;17231
                                ;17232          ;-------------------------------;
U 0A9C, 0000,003C,01C0,FA20,0000,0A9D   ;17233          Q_R[R4]
                                ;17234
                                ;17235          ;-------------------------------;
U 0A9D, 0001,203C,0180,FA98,0000,0A9E   ;17236          R[R3]_Q                         ;SET R3 = TABLE ADDRESS
                                ;17237
                                ;17238          ;-------------------------------;
```

```
                                    ;17239  MOVGETOUT:                                ;MOVC ENTERS HERE TO CLEAR REGS
U 0A9E, 0001,003C,0180,FAA0,0000,0AA0   ;17240          R[R4]_D                        ;STORE FINAL DEST LEN
                                    ;17241
                                    ;17242          ;------------------------------;
U 0AA0, 0003,003C,0180,FA90,0000,0062   ;17243          R[R2]_0, J/IRD                 ;ZERO R2 AND GO AWAY.
                                    ;17244
                                    ;17245          ;------------------------------;
                                    ;17246
                                    ;17247  ;       COME HERE FROM LOOP EXIT CODE IF WE NEED TO FILL (MOVTC ONLY)
                                    ;17248          ;------------------------------;
                                    ;17249  MOVTCFILL:
                                    ;17250          LC_RC[TO], Q_NOT.Q,            ;JUMP BACK TO MAIN LOOP FOR FILLS
U 0AA1, 0001,2028,65C0,F900,0104,6688   ;17251          FE_K[.10], J/MOVTC.1
                                    ;17252
                                    ;17253          ;------------------------------;
```

```
                                        ;17254  ;       MOVTUC COMES HERE WHEN TRANSLATED CHAR MATCHES ESCAPE CHAR
                                        ;17255  ;       D = DESTLEN-SRCLEN-1 IN <15:0>, FFFF IN <31:16>
                                        ;17256  ;       Q = NUMBER OF CHARS LEFT IN SRC - 1
                                        ;17257
                                        ;17258          ;-------------------------------;
                                        ;17259  MOVTUCESCAPE:
                                        ;17260          R[R0] Q+1, Q_ALU, SET.V,        ;R0 & Q GET CORRECOUNT CT,
U 0AA2, 001F,1610,01C0,FA80,2020,0079   ;17261          CLR.FPD, STATE6?                ;SET ESC FLAG, TEST DEST>SRC
                                        ;17262
                                        ;17263  =10*1   ;-------------------------------;BRANCH ON STATE<6>
                                        ;17264                                          ; (DESTLEN > SRCLEN) (FILL FLAG=0)
U 0079, 0C1D,2008,0180,FA80,0000,0A99   ;17265          R[R0]_Q-D-1, D_Q, J/MOVTCEXIT   ;R0=Q-(DESTL-SRCL-1)-1
                                        ;17266                                          ; = Q + SRCL - DESTL, R4 = Q
                                        ;17267          ;11*1-------------------------;
U 007D, 081F,4010,0180,F800,0000,0A99   ;17268          D_D.0XT[WORD]+Q+1, J/MOVTCEXIT  ;R4 = Q+(DESTL-SRCL-1)+1
                                        ;17269                                          ; = Q + DESTL - SRCL, R0 = Q
                                        ;17270          ;-------------------------------;
                                        ;17271
                                        ;17272
                                        ;17273  ;MOVTC/MOVTUC RESTART CODE INCLUDED IN MOVC
                                        ;17274  ;END OF MOVTC/MOVTUC
                                        ;17275
                                        ;17276  .LIST           ;Re-enable full listing
```

17276; This page intentionally left blank.

```
                                        ;17277  .TOC    "EDIT.MIC"
                                        ;17278  .TOC    "Revision 1.5"
                                        ;17279  ;        P. R. Guilbault
                                        ;17280
;17281  .NOBIN
;17282  .TOC    "       Revision History"
;17283
;17284  ; 01    Comment patch 86 that fixed FPD unpack problem.
;17285  ;       Comment patch 097 that fixed restart problem.
;17286  ;       Comment patch 098 that fixed restart problem.
;17287  ;       Add re-entry point labels for patch no. 098.
;17288  ; 00    Start of history
;17289
                                        ;17290  .BIN
                                        ;17291  .NOLIST         ;Disable listing of PCS code for quickie assemblies
```

```
;17292  .TOC    ""      Edit instruction      : ALGORITHM""
;17293
;17294  ;       EDITPC INTERPRETS THE PATTERN SEQUENCE AND PERFORMS
;17295  ;       THE REQUESTED OPERATION ON THE SOURCE, WRITING
;17296  ;       ANY OUTPUT TO THE DESTINATION.
;17297
;17298  ;INPUTS:
;17299  ;       Q = 1ST OPERAND, NAMELY LENGTH
;17300  ;       D = 2ND OPERAND, SRC ADDR
;17301
;17302  ;OUTPUTS:
;17303  ;(IF NO EXCEPTIONS)
;17304  ;       R0 = LENGTH
;17305  ;       R1 = START OF SRC
;17306  ;       R2 = 0
;17307  ;       R3 = ADDR OF EOSEND OPERATOR IN PATTERN
;17308  ;       R4 = 0
;17309  ;       R5 = END OF DEST + 1
;17310
;17311  ;DURING EXECUTION, THE INTERNAL REPRESENTATION OF REGISTERS:
;17312  ;R0     <7:0>LENGTH(IN NIBBLES)
;17313  ;       <15:8>ADJUST INPUT COUNTER
;17314  ;       <31:16> USED BY FPD FOR STATE + PC DELTA
;17315  ;R1     SRC ADDR
;17316  ;R2     <15:8> = SIGN, <7:0> = FILL,
;17317  ;       <31:15> = Q = CURRENT COUNTER AT FPD TIME
;17318  ;R3     PATTERN ADDR
;17319  ;R4     COPY OF ORIGINAL LENGTH(R0)
;17320  ;R5     DEST ADDR
;17321
;17322  ;FOR HANDLING INTERRUPTS + EXCEPTIONS, THE INTERPRETATION OF THE
;17323  ;STATE REGISTER IS:
;17324  ;IF STATE <6:4> = 0, THEN STATE <2:0>:
;17325  ;0000   FIRST READ. REREAD LENGTH OF STRING
;17326  ;0001   PATT1. READING 1ST BYTE OF PATTERN. REREAD R3
;17327  ;0010   PATT2. READING 2ND BYTE OF PATTERN. DECR R3 BY
;17328  ;        1 AND REGET PATTERN
;17329  ;0011   ADJUST INPUT
;17330  ;0100   (UNUSED. GENL DEST MODE)
;17331  ;0101   END FLOAT OR STORE SIGN
;17332  ;0110   INSERT (EQUIV. TO PATT2 + WRITE)
;17333  ;0111   FILL OR BLANK0
;17334  ;IF STATE <6:4> NON-0, THEN STATE <2:1>:
;17335  ;00     MOVE/FLOAT READ
;17336  ;01     FLOAT SNGL
;17337  ;10     MOVE/FLOAT WRITE
;17338  ;11     FLOAT DBL
;17339  ;BIT 3 UNUSED
```

```
;17340  ;STATE <6:4>:
;17341  ;000    NOT MOVE OR FLOAT OF ANY FLAVOR
;17342  ;001    MOVE
;17343  ;010    FLOAT
;17344  ;011    UNDEFINED
;17345  ;100    UNDEFINED
;17346  ;101    PRE-ZEROING PART OF MOVE
;17347  ;110    PRE-ZEROING PART OF FLOAT
;17348  ;111    UNDEFINED
;17349  ;STATE <7> = PREDEC CASE: ORIGINAL COUNT WAS ODD, SO R1 DECREMENTED
;17350  ;         SO INCREMENTATION IN LOOPS WORKS, BUT THIS INCREMENTATION
;17351  ;         HASN'T OCCURRED YET
;17352
;17353  ;LABELS OF INTEREST:
;17354  ;EDITFIRST       READ SIGN OF SOURCE STRING + INITIALIZE PSL COND CODES
;17355  ;EDITNEXT        EVERY TIME ANOTHER BYTE OF THE PATTERN IS NEEDED,
;17356  ;                INCREMENT ADDRESS + GET NEXT BYTE
;17357  ;EDPATT1RST      AFTER RESTARTING FROM AN INTERRUPT/EXCEPTION, 1 BYTE PATTERNS
;17358  ;                + BRIEF 2 BYTE PATTERNS REREAD PATTERN
;17359  ;EDZEROTOTHREE   PATTERN IS IN THE RANGE 0-3
;17360  ;EDFORTYTO47     PATTERN IS IN RANGE 40-47
;17361  ;EDV89A          PATTERN IS IN THE RANGE 81-8F,91-9F,A1-AF
;17362  ;ED'PATTNAME     FOR MOST PATTERNS, THE LABEL ED CONCATENATED
;17363  ;                WITH THE NAME OF THE PATTERN IS WHERE ITS CODE
;17364  ;                STARTS
;17365  ;EDMAYNEEDZEROS  FOR PATTERN=MOVE OR FLOAT, MAY NEED SOME INITIAL
;17366  ;                ZEROS/FILL BEFORE THE ACTUAL SRC DATA IS READ
;17367  ;EDMOVEORFLOAT   FOR PATTERN=MOVE OR FLOAT, IT IS NOW TIME TO
;17368  ;                READ SOME OF THE ACTUAL SRC DATA
;17369  ;EDFLOATRSTLEFT  MOVE/FLOAT NEEDS LEFT NIBBLE OF SRC BYTE
;17370  ;EDFLOATRIGHTNIB        MOVE/FLOAT NEEDS RIGHT NIBBLE OF SRC BYTE
;17371  ;EDFLOATNOTO     FLOAT ENCOUNTERED A SIGNIFICANT CHAR
;17372  ;EDFLOATEQ0      FLOAT ENCOUNTERED AN INSIGNIFICANT 0
;17373  ;EDMOVEWR2       FLOAT ENCOUNTERED A SIGNIFICANT 0 OR
;17374  ;                MOVE WANTS TO ASCII-IZE A CHARACTER
;17375  ;EDFLOATNOSIG    FLOAT FOUND 1ST SIGNIFICANT CHAR +
;17376  ;                NEEDS TO WRITE THE SIGN AS WELL AS THE CHAR
;17377  ;EDMOVEMORE      MOVE IS EVALUATING CHAR JUST READ
;17378  ;EDADJFINI       MAKE NEGATIVE WD COUNTER FOR ADJUST INPUT
;17379  ;EDADJINRIGHT    DETERMINE IS RIGHT NIBBLE = 0
;17380  ;EDADJINLEFT     DETERMINE IF LEFT NIBBLE = 0
;17381  ;EDITRS1         RESTART EDITPC AFTER AN INT/EXC. DETERMINE WHAT
;17382  ;                OPERATION WAS INTERRUPTED.
;17383  ;EDNOTMOVEORFLOAT        THE INTERRUPTED OPERATION WAS NOT MOVE OR FLOAT.
;17384  ;                FIGURE OUT WAHT IT WAS + RESUME IT
;17385  ;EDMVFLRDORWRITE         THE INTERRUPTED OPERATION WAS MOVE OR FLOAT.
;17386  ;                FIGURE OUT WHICH FLAVOR.
```

```
                                ;17387  .TOC    ""      Edit instruction      : EDITPC entry""
                                ;17388
                                ;17389          ;-------------------------------;
                                ;17390  3C6:    R[T0]_Q.AND.K[.FFFF],           ;SAVE LENGTH(ARG 1)
U 03C6, 0019,2035,C180,F980,0000,047E   ;17391          CALL,J7ASPC                     ;FETCH ARG 3
                                ;17392
                                ;17393          ;-------------------------------;
U 03E6, 0001,203C,0180,F988,0000,0190   ;17394  3E6:    R[T1]_Q                         ;SAVE ARG 2
                                ;17395
                                ;17396          ;-------------------------------;
                                ;17397  =00****0
                                ;17398          R[T2]_D,                        ;SAVE PATTERN ADDR(ARG 3)
U 0190, 0001,003D,0180,F990,0000,047E   ;17399          CALL,J7ASPC
                                ;17400
                                ;17401          ;-------------------------------;
                                ;17402  =11****0
U 01F0, 0019,0000,0580,FAA8,0000,01F1   ;17403          R[R5]_D-K[.1]                   ;DEST ADDR-1
                                ;17404
                                ;17405          ;-------------------------------;
U 01F1, 0000,003C,0180,F900,0000,0AA4   ;17406          LC_R[T0]                        ;LOAD LENGTH
                                ;17407
                                ;17408          ;-------------------------------;
                                ;17409          R[R0]_LC,                       ;SAVE LENGTH AS PASSED
U 0AA4, 0010,0038,01C0,FA20,0000,0AA5   ;17410          Q_LC                            ;
                                ;17411
                                ;17412          ;-------------------------------;
U 0AA5, 0001,203C,0180,FAA0,0000,0AA6   ;17413          R[R4]_Q                         ;LENGTH HERE ALSO FOR RESTORATION
                                ;17414                                          ;ON SUCCESSFUL FINISH
                                ;17415          ;-------------------------------;
                                ;17416          ALU_Q.ANDNOT.K[.1F],CLK.UBCC,   ;VERIFY LENGTH < 32
U 0AA6, 0019,2024,8D80,F908,0010,0AA8   ;17417          LC_R[T1]                        ;
                                ;17418
                                ;17419          ;-------------------------------;
                                ;17420          R[R1]_LC,                       ;SRC ADDR
U 0AA8, 0010,0138,0180,FA88,0000,06B8   ;17421          Z?                              ;BRANCH ON LEN > 31.
                                ;17422
                                ;17423  =0       ;-------------------------------;ALU <Z>
                                ;17424          R[R5]_D,                        ;LEN > 31
U 06B8, 0001,003C,0180,FAA8,0000,0B10   ;17425          J/EDMATGT31                     ;
                                ;17426
                                ;17427          ;1------------------------------;
                                ;17428          D_R[T2],                        ;PATTERN ADDRESS
U 06B9, 0810,0038,1980,F910,1404,65B8   ;17429          STATE_K[ZERO]                   ;INITIALIZE STATE TO 0
```

```
                                        ;17430  =0****00 ;-------------------------------;SUITABLE FOR RTN40 FOR CLRPSLCC
                                        ;17431                                          ;+ RETURN3 FOR SETFPD
                                        ;17432          Q_ID[PSL],                      ;PREPARE TO CLEAR PSL CC
U 05B8, 0000,003D,3DF0,2C00,0000,09E4   ;17433          CALL,J/CLRPSLCC                 ;
                                        ;17434
                                        ;17435  =1****00 ;-------------------------------;CLRPSLCC RETURNS 40
                                        ;17436          CALL,J/SETFPD,                  ;
U 05F8, 0019,0001,0580,FA98,0000,0E16   ;17437          R[R3]_D-K[.1]                   ;PATT ADDR-1
                                        ;17438
                                        ;17439          ;-------------------------------;
U 05F9, 0000,003C,0180,F800,0000,0B16   ;17440          J/EDITFPD                       ;
                                        ;17441
                                        ;17442          ;-------------------------------;
U 05FA, 0000,003C,0180,F800,0000,0B16   ;17443          J/EDITFPD                       ;
                                        ;17444
                                        ;17445          ;-------------------------------;
U 05FB, 0818,0038,7580,F800,0000,0AA9   ;17446          D_BLANK                         ;BLANK = 20(HEX)
                                        ;17447
                                        ;17448          ;-------------------------------;
```

```
                                    ;17449  .TOC    ''      Edit instruction      : SIGN EVALUATION''
                                    ;17450
                                    ;17451  ;BY NOW, ALL THE OPERANDS HAVE BEEN FETCHED. THEY ARE USED AS:
                                    ;17452  ;R0     LENGTH
                                    ;17453  ;R1     SRC ADDR
                                    ;17454  ;R3     PATTERN ADDR-1
                                    ;17455  ;R4     LENGTH
                                    ;17456  ;R5     DEST ADDR-1
                                    ;17457  ;RC 0   LENGTH
                                    ;17458  ;RC 1   SRC ADDR
                                    ;17459  ;RC 2   PATTERN ADDR
                                    ;17460  ;STATE  0
                                    ;17461  ;PSL    COND CODES ALL CLEAR, FPD SET
                                    ;17462  ;D      20(HEX)
                                    ;17463  ;THE SIGN OF THE SOURCE STRING MUST BE DETERMINED NEXT + THE DEFAULT
                                    ;17464  ;FILL + SIGN REGISTERS SET UP
                                    ;17465
                                    ;17466  EDITFIRST:
                                    ;17467          Q_R[R0].AND.K[.FF].RIGHT,       ;# NIBBLES/2 = # BYTES
U 0AA9, 0058,0034,49C0,FA00,0010,0AAA ;17468        C[K.LBCC                        ;SEE IF LENGTH = 0
                                    ;17469
                                    ;17470          ;-------------------------------;
                                    ;17471          LAB_R1&RC[T1]_ALU,              ;RC 1 = BLANK = DEFAULT FILL CHAR
                                    ;17472          ALU_D,                          ;
                                    ;17473          SWAPD,                          ;D<31:24>=BLANK
U 0AAA, 0B01,003C,0180,FB08,0104,6AAC ;17474        FE_K[.8]                        ;IF SIGN IS NEGATIVE, THIS
                                    ;17475                                          ;CONSTANT WILL BE USED TO SET PSLCC<N>
                                    ;17476          ;-------------------------------;
                                    ;17477          VA_LA+Q,                        ;WANT TO READ SIGN NIBBLE
                                    ;17478          SC_FE,                          ;SC_8 FOR ROTATING
                                    ;17479          Q_D,                            ;Q<31:24>=BLANK
U 0AAC, 001C,0114,01E0,F800,0281,06BC ;17480        Z?                              ;SRC LENGTH = 0?
                                    ;17481
                                    ;17482  =0      ;-------------------------------;ALU Z
                                    ;17483          D[BYTE]_CACHE,                  ;READ SIGN NIBBLE
                                    ;17484          STATE_FIRST,                    ;FIRST = 0 WHICH WILL ALSO
U 06BC, 0000,803C,1980,4000,1404,6AAD ;17485        J/EDSIGN                        ;BE USED TO KEEP PSLCC<N> OFF
                                    ;17486
                                    ;17487
                                    ;17488  ;STATE REGISTER USED HERE TO SET PSLCC<N+Z> ACCORDING TO SIGN NIBBLE.
                                    ;17489  ;STATE IS EITHER 0 IF POSITIVE OR 8 IF NEGATIVE NOW
                                    ;17490
                                    ;17491          ;1-------------------------------;
                                    ;17492  EDPLUSMINUS:                            ;EITHER SRC LEN = 0 OR STATE ALREADY
                                    ;17493                                          ;SET ACCORDING TO SIGN
                                    ;17494          STATE_STATE.OR.K[.4],           ;ALWAYS SET Z
U 06BD, 0000,003C,1180,F800,1404,2AAE ;17495        J/EDPM1                         ;
```

```
                                    ;17496  EDSIGN: ;--------------------------------;
                                    ;17497          D_K[.28],                       ;PRE-ASSUME SIGN IS NEG
U 0AAD, 0818,0F38,2D80,F800,0000,07EA ;17498        BEN/DECIMAL                     ;BRANCH ACCORDING TO SIGN
                                    ;17499                                          ;IN RIGHT NIB (NEG = 1,3,5,9,B,D).
                                    ;17500                                          ;Q<7:0> = BLANK
                                    ;17501  =*10    ;--------------------------------;D<3:0> = B OR D
                                    ;17502          D_BLANK,                        ;
U 07EA, 0818,0038,7580,F800,0000,06BD ;17503        J/EDPLUSMINUS                   ;USE STATE AS 0 FOR SAKE OF PSLCC<N>
                                    ;17504
                                    ;17505          ;*11------------------------------;
                                    ;17506          D_D+K[.4]+1,                    ;- IS 55 OCT = 2D HEX. USE 28+4+1
                                    ;17507          STATE_FE,                       ;STATE NEEDS TO BE 8 TO SET PSLCC<N>
U 07EB, 0819,0010,1180,F800,1400,66BD ;17508        J/EDPLUSMINUS                   ;
                                    ;17509
                                    ;17510  EDPM1:  ;--------------------------------;
                                    ;17511          RC[T2]_D,                       ;GET SIGN TO RC(2)<7:0> + TO D<15:8>
                                    ;17512          D_DAL.SC,                       ;BLANK(DEFAULT FILL CHAR) TO D<7:0>
                                    ;17513          Q_ID[PSL],                      ;PSLCC<Z+N> WILL BE SET MANUALLY
U 0AAE, 0D01,003C,3DF0,2D90,1480,0AB0 ;17514        SC_STATE                        ;GET PROPER PSLCC<N+Z> SETTING
                                    ;17515
                                    ;17516          ;--------------------------------;
                                    ;17517          D_Q.OR.K[SC],                   ;OR IN THE <N+Z> BIT
                                    ;17518          LAB_R[R0],                      ;CONSIDER EVEN/ODD OF COUNT
U 0AB0, 0819,2030,1DE0,FA00,0000,0AB1 ;17519        Q_D                             ;MOVE SIGN BYTE TO Q
                                    ;17520
                                    ;17521          ;--------------------------------;
                                    ;17522          R[R2]_Q,                        ;SAVE SIGN BYTE
                                    ;17523          ID[PSL]_D,                      ;REWRITE PSL
                                    ;17524          BEN/ROR,                        ;BRANCH ON COUNT EVEN/ODD
U 0AB1, 0001,223C,3D80,3E90,0000,01C4 ;17525        J/EDITNEXT                      ;
```

```
                                        ;17526  .TOC    ''       Edit instruction      : PATTERN DECODE''
                                        ;17527
                                        ;17528  ;IT IS TIME TO GET A BYTE FROM THE PATTERN + ACT ACCORDINGLY.
                                        ;17529  ;R3 NEEDS TO BE INCREMENTED TO GET NEXT BYTE.
                                        ;17530  ;ONLY STATE BIT STILL OF INTEREST IS <7>(PREDECREMENT).
                                        ;17531
                                        ;17532  =1*0    ;-------------------------------;LA<0>. KNOW PSL <C> OFF
                                        ;17533  EDITNEXT:
                                        ;17534          LAB_R[R3],                      ;
                                        ;17535          STATE_STATE.AN.NOTPREDEC,       ;MAINTAIN ONLY STATE<7>
U 01C4, 0000,003C,5980,FA18,1404,4AB5   ;17536          J/EDITN1                        ;
                                        ;17537
                                        ;17538          ;1*1----------------------------;LA<0> = 1 => COUNT IS ODD
                                        ;17539                                          ;GET HERE 1ST TIME ONLY
                                        ;17540          LAB_R[R1],                      ;NEED TO PRE-DECREMENT SRC ADDR
U 01C5, 0000,003C,4180,FA08,1404,6AB4   ;17541          STATE_PREDEC                    ;NOTE THIS IN STATE <7>
                                        ;17542
                                        ;17543          ;-------------------------------;
                                        ;17544          R[R1]_LA-K[.1],
U 0AB4, 0018,0000,0580,FA88,0000,01C4   ;17545          J/EDITNEXT                      ;
                                        ;17546
                                        ;17547          ;-------------------------------;
                                        ;17548  EDITN1: VA_LA+K[.1],R[R3]_LA+K[.1],     ;PREPARE TO READ NEXT BYTE OF PATTERN
U 0AB5, 0018,0014,0580,FA98,4200,0AB6   ;17549          INTRPT.STROBE                   ;CHECK ON INTERRUPTS PENDING
                                        ;17550
                                        ;17551          ;-------------------------------;
                                        ;17552  EDPATT1RST:                             ;
                                        ;17553          D[BYTE]_CACHE,                  ;READ 1 BYTE OF PATTERN
                                        ;17554          STATE_STATE.OR.PATT1,           ;SET STATE<0>
U 0AB6, 0000,8E3C,058C,4000,1404,28F6   ;17555          BEN/INTERRUPT                   ;INTERRUPT?
                                        ;17556
                                        ;17557  =110    ;-------------------------------;INTERRUPT
                                        ;17558          D_D.AND.K[.FF],                 ;EXTRACT THE BYTE SO ALU CC TESTS WORK
U 08F6, 0819,0034,4980,F800,0000,0AB8   ;17559          J/EDITNOINT                     ;
                                        ;17560
                                        ;17561          ;111----------------------------;
                                        ;17562          FE_K[ZERO],                     ;HANDLE AN INTERRUPT
U 08F7, 0000,003C,1980,F800,0104,6B16   ;17563          J/EDITFPD                       ;
                                        ;17564
                                        ;17565          ;-------------------------------;
                                        ;17566  EDITNOINT:                              ;
U 0AB8, 0019,0000,1180,F800,0010,0AB9   ;17567          ALU_D-K[.4],CLK.UBCC            ;CHECK FOR PATTERN < 4
                                        ;17568
                                        ;17569          ;-------------------------------;
                                        ;17570          ALU_D.ANDNOT.K[.3F],CLK.UBCC,   ;NEXT TEST IS > 3F
                                        ;17571          STATE_STATE.AN.STOO,            ;REALLY ONLY NEED TO CLEAR 'PATT1',
                                        ;17572          LAB_R[R5],                      ;BUT THAT CONSTANT'S NOT ACCESSIBLE
U 0AB9, 0019,1B24,5580,FA28,1414,476A   ;17573          BEN/ALU                         ;BRANCH ON 0-4 RANGE
```

```
                                        ;17574  ;KEEPING IN MIND THAT OF THE POSSIBLE RANGE 00-FF, ONLY A SMALL NUMBER
                                        ;17575  ;OF PATTERNS ARE VALID, DECODE THE PATTERN BYTE TO HANDLE THE MYRIAD
                                        ;17576  ;OF POSSIBILITIES(RESERVED OPERAND, ILLEGAL, OUT OF RANGE, ETC.)
                                        ;17577
                                        ;17578  =1010   ;-------------------------------;ALUCC Z+C
                                        ;17579          J/EDZEROTOTHREE,BEN/MUL,        ;IN RANGE 1-3. BRANCH ACCORDINGLY
                                        ;17580          Q_ID[PSL],DT/BYTE,              ;
U 076A, 0000,8C3C,3DF0,2E00,0010,0904   ;17581          A[U_R[R0],CLK.UBCC              ;TEST COUNT
                                        ;17582
                                        ;17583          ;1011-----------------------;
                                        ;17584          J/EDGREATERTHAN4,Z?,            ;BRANCH ON > 3F TEST
U 076B, 0019,0134,9580,F800,0010,06C8   ;17585          ALU_D.AND.K[.80],CLK.UBCC       ;TEST FOR > 50 BY BIT 6
                                        ;17586
                                        ;17587  =1111   ;-------------------------------;
                                        ;17588          J/ENDFLOAT2,                    ;PATTERN = 4 WHICH IS STORESIGN
U 076F, 0010,8038,01C0,F910,0000,0AC0   ;17589          Q_RC[T?],DT/BYTE                ;LOAD SIGN CHAR
                                        ;17590
                                        ;17591  =100    ;-------------------------------;D<1:0>
                                        ;17592  EDZEROTOTHREE:                          ;
                                        ;17593          J/EDEOEND,Z?,                   ;PATTERN = 0. SEE IF SRC ALL READ
                                        ;17594          LAB_R[R5],                      ;LATCH DEST ADDR
U 0904, 0000,013C,0180,FA28,0084,66DC   ;17595          SC_R[.8]                        ;IN CASE THIS IS PRE-MATURE(ABORT)
                                        ;17596
                                        ;17597          ;101----------------------------;
U 0905, 0000,023C,0180,F800,0000,0995   ;17598          J/EDENDFLOAT,BEN/ROR            ;PATTERN = 1. BRANCH ON SIGNIFICANCE
                                        ;17599
                                        ;17600          ;110----------------------------;
U 0906, 0819,2024,0580,F800,0000,0AC9   ;17601          J/EDSIG,D_Q.ANDNOT.PSWC         ;PATTERN = 2 = CLEAR SIGNIF
                                        ;17602
                                        ;17603          ;111----------------------------;
U 0907, 0819,2030,0580,F800,0000,0AC9   ;17604          J/EDSIG,D_Q.OR.PSWC             ;PATTERN = 3 = SET SIGNIF
```

```
                                    ;17605  =0      ;-------------------------------;ALUCC <Z>
                                    ;17606  EDGREATERTHAN4:                         ;PATTERN IS > 4
                                    ;17607          J/EDGREATERTHAN3F,Z?,           ;BRANCH ON > 50
                                    ;17608          LAB_R[R3],                      ;
                                    ;17609          ALU_D.AND.K[.40],               ;CONSIDER BIT 6
U 06C8, 0019,0134,3180,FA18,0010,06CC ;17610        CLK.UBCC                        ;TEST FOR < C0
                                    ;17611                                          ;KNOW BIT 7 IS SET
                                    ;17612          ;1------------------------------;
U 06C9, 0000,003C,0180,F800,0084,66DC ;17613        J/EDEOEND,SC_K[.8]              ;5 TO 3F
                                    ;17614
                                    ;17615  =0      ;-------------------------------;ALUCC <Z>
                                    ;17616  EDGREATERTHAN3F:                        ;PATTERN IS > 3F
                                    ;17617          J/EDGREATERTHAN4F,Z?,           ;BRANCH ON BIT 6(>BF TEST)
                                    ;17618                                          ;KNOW NOW THAT BIT 7 IS SET
                                    ;17619          Q_D.AND.K[.F],                  ;SAVE REPEAT COUNT IN Q
U 06CC, 0019,0134,61C0,F800,0010,06D0 ;17620        CLK.UBCC                        ;TEST FOR LOW NIBBLE = 0
                                    ;17621
                                    ;17622          ;1------------------------------;
                                    ;17623          J/EDFORTYTO4F,D3?,              ;TEST FOR >47
                                    ;17624          R[R3]_LA+K[.1],                 ;UPDATE DEST ADDR
                                    ;17625          VA_LA+K[.1],                    ;PREPARE TO READ NEXT BYTE OF PATTERN
U 06CD, 0018,1914,05F8,FA98,0200,0777 ;17626        Q_0                             ;
                                    ;17627
                                    ;17628  =0111   ;-------------------------------;D <3>, I.E. NIB =48-4F
                                    ;17629  EDFORTYTO4F:                            ;
                                    ;17630          J/EDFORTYTO47,D2-0?,            ;
                                    ;17631          LAB_R[R5],                      ;
U 0777, 0000,193C,0980,FA28,1404,2930 ;17632        STATE_STATE.OR.PATT2            ;SET STATE <1>. ALSO <7> MAY BE SET
                                    ;17633
                                    ;17634          ;1111-------------------------;
                                    ;17635          J/EDEOEND,LAB_R[R5],            ;48-4F(BIT 3 = 1)
U 077F, 0000,003C,0180,FA28,0084,66DC ;17636        SC_K[.8]                        ;
```

```
                                    ;17637  =*000   ;-----------------------------;D<2:0> (KNOW D IS 40-47)
                                    ;17638  EDFORTY1047:                          ;
                                    ;17639          J/EDLOADFILL,                 ;40. LOAD FILL
                                    ;17640          D[BYTE]_CACHE,                ;READ NEXT BYTE
U 0930, 0000,803C,1980,4000,0084,6AC1 ;17641        SC_K[ZERO]                    ;NO ROTATION REQD
                                    ;17642
                                    ;17643          ;*001-----------------------;
                                    ;17644          J/EDLOADSIGN,                 ;41. LOAD SIGN
                                    ;17645          D[BYTE]_CACHE,                ;READ NEXT BYTE
U 0931, 0000,803C,0180,4000,0084,6AC5 ;17646        SC_K[.8]                      ;PREPARE TO ROTATE SIGN CHAR
                                    ;17647
                                    ;17648          ;*010-----------------------;
                                    ;17649          J/EDLOADPLUS,BEN/PSL.CC,      ;42. LOADPLUS. CHECK SIGN.
U 0932, 0000,1A3C,0180,F800,0084,6807 ;17650        SC_K[.8]                      ;PREPARE TO ROTATE SIGN CHAR
                                    ;17651
                                    ;17652          ;*011-----------------------;
                                    ;17653          J/EDLOADMINUS,BEN/PSL.CC,     ;43. LOADMINUS. CHECK SIGN
U 0933, 0000,1A3C,0180,F800,0084,6817 ;17654        SC_K[.8]                      ;PREPARE TO ROTATE SIGN CHAR
                                    ;17655
                                    ;17656          ;*100-----------------------;
                                    ;17657          J/EDINSERT,BEN/ROR,           ;44. INSERT. CHECK SIGNIFICANCE.
U 0934, 0810,8238,0180,F908,0000,09A5 ;17658        D_RC[T1],DT/BYTE              ;LOAD FILL CHAR, I.E.
                                    ;17659                                        ;ASSUME IT'S NO SIGNIFICANCE
                                    ;17660
                                    ;17661          ;*101-----------------------;
U 0935, 0000,803C,0180,4000,0000,0AE8 ;17662        J/EDBLANK0,D[BYTE]_CACHE      ;45. BLANK IF 0. READ NEXT BYTE
                                    ;17663
                                    ;17664          ;*110-----------------------;
                                    ;17665          J/EDREPLACESIGN,              ;46. REPLACE SIGN
                                    ;17666          D[BYTE]_CACHE,                ;READ NEXT BYTE
U 0936, 0010,8038,01C0,4108,0000,0AEA ;17667        Q_RC[T1]                      ;LOAD FILL
                                    ;17668
                                    ;17669          ;*111-----------------------;
U 0937, 0000,803C,0180,4000,0000,0AED ;17670        J/EDADJUSTINPUT,D[BYTE]_CACHE ;47. ADJUST INPUT. READ NEXT BYTE
```

```
                                    ;17671  =0      ;-------------------------------;ALUCC <Z>
                                    ;17672  EDGREATERTHAN4F:                        ;PATTERN IS >4F
                                    ;17673          J/EDEOEND,LAB_R[R5],            ;50 TO 7F OR >8F(BIT 6 = 1)
U 06D0, 0000,003C,0180,FA28,0084,66DC ;17674        SC_K[.8]                        ;
                                    ;17675
                                    ;17676          ;1-------------------------------;
                                    ;17677          J/EDGREATERTHAN7F,Z?,           ;BIT 6 = 0, SO PATTERN IS 8X,9X,AX OR BX
                                    ;17678                                          ;BRANCH ON LOW NIBBLE(X=0 IS ILLEGAL)
                                    ;17679          D_D.RIGHT2,                     ;SHIFT PATTERN SO CAN BRANCH ON HIGH NIBBLE
                                    ;17680          LAB_R[R0],                      ;WE'LL NEED COUNTER SHORTLY
U 06D1, 0201,213C,0180,FA00,0082,06D4 ;17681        SC_Q                            ;GET REPEAT COUNT
                                    ;17682
                                    ;17683  =0      ;-------------------------------;ALUCC <Z>
                                    ;17684  EDGREATERTHAN7F:                        ;PATTERN IS > 7F + < C0
                                    ;17685          J/EDV89A,BEN/D3-0,              ;BRANCH ON <5:4> OF PATTERN
                                    ;17686                                          ;WHICH IS NOW IN D<3:2>
                                    ;17687                                          ;(8,9, OR A NOW 0XX,4XX, OR 6XX)
                                    ;17688          LC_RC[T1],                      ;FILL CHAR REQD FOR FILL
U 06D4, 0800,593C,0180,F908,0010,0783 ;17689        D_[A,CLK.UBCC,DT/WORD           ;CHECK IF ANY INITIAL ZEROING REQD
                                    ;17690
                                    ;17691          ;1-------------------------------;
                                    ;17692          J/EDEOEND,LAB_R[R5],            ;80,90,A0,B0
U 06D5, 0000,003C,0180,FA28,0084,66DC ;17693        SC_K[.8]                        ;
```

```
                                        ;17694  ;THE PATTERN IS EITHER FILL(81-8F), MOVE(91-9F), OR FLOAT(A1-AF).
                                        ;17695  ;IN THE LATTER 2 CASES, IT IS TIME TO CONSIDER THE NEED FOR PRE-ZEROING
                                        ;17696  ;DUE TO A PRECEDING ADJUST INPUT.
                                        ;17697
                                        ;17698  =0011   ;-------------------------------;D3-0
                                        ;17699  EDV89A: J/EDFILL,                       ;PATTERN = 81-8F
                                        ;17700          STATE_STATE.OR.FILL,            ;STATE<2:0> 7
                                        ;17701          ALU_D,CLK.UBCC,DT/BYTE,         ;ALTHO WE KNOW THERE'S
                                        ;17702                                          ;SOME TO DO, NEED ALUCC<Z> = 1
                                        ;17703                                          ;FOR Z? TEST IN FILL LOOP
U 0783, 0001,803C,5D80,FA28,1414,2AE4   ;17704          LAB_R[R5]                       ;LATCH UP LAST DEST ADDR
                                        ;17705
                                        ;17706  ; ************************************************
                                        ;17707  ; * Patch no. 020, PCS 0783 trapped to WCS 1157 *
                                        ;17708  ; ************************************************
                                        ;17709
                                        ;17710          ;0111-------------------------;
                                        ;17711          J/EDMAYNEEDZEROS,ALU.N?,        ;PATTERN = 91-9F. IS RO(WORD) NEGATIVE?
                                        ;17712          STATE_STATE.OR.MOVE,            ;STATE_MOVE<4>+PRE-ZEROING<6>
                                        ;17713                                          ;PRE-ASSUME THERE'S ZEROING
                                        ;17714                                          ;STATE <3:0> = 0
                                        ;17715          LAB_R[R5],                      ;
U 0787, 0C00,1B3C,3580,FA28,1404,2797   ;17716          D_Q                             ;D_COUNT FOR BEN/D.BYTES TEST FOR ANY LEFT
                                        ;17717
                                        ;17718          ;1011-------------------------;
                                        ;17719          J/EDMAYNEEDZEROS,ALU.N?,        ;PATTERN = A1-AF = FLOAT
                                        ;17720                                          ;SEE IF RO NEGATIVE
                                        ;17721          STATE_STATE.OR.FLOAT,           ;STATE_FLOAT<5> + PRE-ZEROING<6>
                                        ;17722                                          ;PRE-ASSUME THERE'S ZEROING
                                        ;17723                                          ;STATE <3:0> = 0
                                        ;17724          LAB_R[R5],                      ;
U 078B, 0C00,1B3C,A580,FA28,1404,2797   ;17725          D_Q                             ;D_COUNT FOR BEN/D.BYTES FOR ANY LEFT
                                        ;17726
                                        ;17727          ;1111-------------------------;
                                        ;17728          J/EDEOEND,LAB_R[R5],            ;PATTERN = B1-BF
U 078F, 0000,003C,0180,FA28,0084,66DC   ;17729          SC_K[.8]                        ;
                                        ;17730
                                        ;17731  =0111   ;-------------------------------;ALUCC <N>
                                        ;17732  EDMAYNEEDZEROS:                         ;
                                        ;17733          STATE_STATE.AN.PREDECZERO,      ;ABOUT TO READ, SO STATE
                                        ;17734                                          ;PRE-DEC<7> + ZERO<6> WILL VANISH
                                        ;17735                                          ;ONLY STATE <5:4> CAN NOW BE SET
                                        ;17736          D_R[R0],                        ;LOAD SRC LENGTH FOR DECREMENT + LF/RT
                                        ;17737          SC_R[R0],                       ;FOR BEN/MUL AT FLOAT
                                        ;17738          CLK.UBCC,DT/BYTE,               ;CHECK ON LENGTH
                                        ;17739          ??,                             ;DID COUNTER FOR THIS PATTERN RUN OUT?
U 0797, 0800,813C,D180,FA00,1496,46F0   ;17740          J/EDMOVEORFLOAT                 ;
                                        ;17741
                                        ;17742          ;1111-------------------------;
                                        ;17743          VA_LA+K[.1],R[R5]_LA+K[.1],     ;THERE'S SOME INITIAL ZEROING REQD
                                        ;17744          BEN/ROR,                        ;CHECK PSL<C>FOR FILL OR 0 CHAR
U 079F, 0018,0214,0580,FAA8,0200,0945   ;17745          J/EDINITCHARS                   ;
```

```
                                        ;17746  =101    ;-------------------------------;PSL <C>
                                        ;17747  EDINITCHARS:                            ;
                                        ;17748          D_RC[T1],                       ;STORE FILL
                                        ;17749          INTRPT.STROBE,                  ;
                                        ;17750          BEN/D.BYTES,                    ;Q <7:0> = 0?
U 0945, 0810,1838,0180,F908,4000,07EE   ;17751          J/EDINIT1                       ;
                                        ;17752
                                        ;17753          ;111----------------------------;PSL <C> = 1
                                        ;17754          D_K[.30],                       ;ASCII 0
                                        ;17755          INTRPT.STROBE,                  ;
U 0947, 0818,1838,7980,F800,4000,07EE   ;17756          BEN/D.BYTES                     ;Q <7:0> = 0?
                                        ;17757
                                        ;17758  =1110   ;-------------------------------;D.BYTES <7:0>
                                        ;17759  EDINIT1:                                ;
                                        ;17760          J/EDITNEXT,                     ;REPEAT COUNT = 0
U 07EE, 0000,003C,0180,FAA8,0000,01C4   ;17761          R[R5]_LA                        ;RESET R5
                                        ;17762
                                        ;17763          ;1111---------------------------;
                                        ;17764          SC_K[.8],                       ;PREPARE FOR A MASK = 0010
                                        ;17765          CACHE_D[BYTE],                  ;WRITE 1 BYTE OF INIT CHAR
                                        ;17766          LAB_R[R0],                      ;
U 07EF, 0000,8E3C,0180,3200,0084,6966   ;17767          BEN/INTERRUPT                   ;
                                        ;17768
                                        ;17769  =110    ;-------------------------------;INTERRUPT?
                                        ;17770                                          ;NO INTERRUPT PENDING
                                        ;17771          R[R0] LA-MASK-1,                ;DECREMENT HIGH WORD BY 1
                                        ;17772          CLK.UBCC,DT/WORD,               ;SEE IF IT'S STILL NEG
U 0966, 0000,4008,0180,FA80,0010,0ABC   ;17773          J/EDINIT2                       ;
                                        ;17774
                                        ;17775          ;111----------------------------;
U 0967, 0000,003C,1980,F800,0104,6B16   ;17776          J/EDITFPD,FE_K[ZERO]            ;AN INTERRUPT IS PENDING
                                        ;17777
                                        ;17778          ;-------------------------------;
                                        ;17779  EDINIT2:                                ;
                                        ;17780          ALU.N?,                         ;MORE TO DO?
                                        ;17781          Q_Q-K[ZERO]-1,                  ;DECREMENT RUNNING COUNTER
                                        ;17782          D_Q-K[ZERO]-1,                  ;DUPLICATE IT IN D FOR BEN/D.BYTES(I.E. = 0)
                                        ;17783          CLK.UBCC,DT/BYTE,               ;CLOCK IT FOR END OF THIS PATTERN
                                        ;17784          LAB_R[R5],                      ;
U 0ABC, 0819,BB08,19C0,FA28,0010,0797   ;17785          J/EDMAYNEEDZEROS                ;
                                        ;17786
                                        ;17787          ;-------------------------------;
```

```
                                        ;17788  .TOC    ""      Edit instruction      : BRIEF PATTERNS""
                                        ;17789
                                        ;17790  =0      ;-------------------------------;ALUCC <Z>
                                        ;17791  EDEOEND:                                ;
                                        ;17792          J/EDREPEATTOOFEWABORT,          ;
U 06DC, 0018,0014,0580,FAA8,0000,0B0A   ;17793          R[R5]_LA+K[.1]                  ;DIDN'T LOOK AT ALL SPECIFIED
                                        ;17794
                                        ;17795          ;1------------------------------;
                                        ;17796          R[R5]_LA+K[.1],                 ;INCREMENT DEST ADD
U 06DD, 0018,1A14,0580,FAA8,0000,07FB   ;17797          BEN/PSL.CC                      ;SEE IF IT WAS -0
                                        ;17798
                                        ;17799  =1011   ;-------------------------------;PSLCC<Z>
                                        ;17800  EDITEND1:                               ;
                                        ;17801          R[R2]_0,D_0,                    ;NOT 0. GOOD FINISH
U 07FB, 0F03,003C,0180,FA90,0000,0AFE   ;17802          J/EDITDONE                      ;
                                        ;17803
                                        ;17804          ;1111----------------------------;
U 07FF, 0819,2024,0180,F800,0000,0ABD   ;17805          D_Q.ANDNOT.PSWN                 ;GUARANTEE N NOT SET
                                        ;17806
                                        ;17807          ;-------------------------------;
U 0ABD, 0000,003C,3DR0,3C00,C000,07FB   ;17808          ID[PSL]_D,J/EDITEND1            ;
                                        ;17809
                                        ;17810  =101    ;-------------------------------;PSL <C>(BEN/ROR)
                                        ;17811  EDENDFLOAT:                             ;
                                        ;17812          D_Q.OR.PSWC,                    ;SET SIGNIFICANCE
                                        ;17813          LAB_R[R5],                      ;
U 0995, 0819,2030,0580,FA28,0000,0ABE   ;17814          J/EDENDFLOAT1                   ;
                                        ;17815
                                        ;17816          ;111-----------------------------;
                                        ;17817          LAB_R[R3],                      ;
                                        ;17818          STATE_STATE.AN.NOTPREDEC,       ;MAINTAIN ONLY STATE<7>
U 0997, 0000,003C,598U,FA18,1404,4AB5   ;17819          J/EDITN1                        ;
                                        ;17820
                                        ;17821          ;-------------------------------;
                                        ;17822  EDENDFLOAT1:                            ;
                                        ;17823          ID[PSL]_D,                      ;REWRITE PSL
U 0ABE, 0010,0038,3DC0,3D10,0000,0AC0   ;17824          Q_RC[T2]                        ;LOAD SIGN CHAR
                                        ;17825
                                        ;17826          ;-------------------------------;
                                        ;17827  ENDFLOAT2:                              ;
                                        ;17828          VA_LA+K[.1],R[R5]_LA+K[.1],     ;
                                        ;17829          Q_0,D_Q,                        ;GET SIGN IN D<7:0>
                                        ;17830          STATE_STATE.OR.K[.1],           ;STATE = 5 = ENDFLOAT + STORESIGN
                                        ;17831                                          ;BIT 2 FROM INSERTST1
U 0AC0, 0C18,0014,05F8,FAA8,1604,2ACC   ;17832          J/EDINSERTST1                   ;GO WRITE THE SIGN
                                        ;17833
                                        ;17834          ;-------------------------------;
                                        ;17835  EDLOADFILL:                             ;
                                        ;17836          RC[T1]_D.AND.K[.FF],            ;SAVE FILL CHAR
U 0AC1, 0819,0034,4980,F988,0000,0AC4   ;17837          D_D.AND.K[.FF]                  ;FILL CHAR TO R2 ALSO
                                        ;17838
                                        ;17839          ;-------------------------------;
                                        ;17840          Q_R[R2].ANDNOT.K[.FF],          ;FILL CHAR SAVED IN R2 ALSO
U 0AC4, 0018,0024,49C0,FA10,0000,0AC8   ;17841          J/EDLOADPORM1                   ;
```

```
                                ;17842          ;--------------------------------;
                                ;17843  EDLOADSIGN:                              ;
                                ;17844  EDLOADPORM:                              ;
                                ;17845          RC[T2]_D.AND.K[.FF],            ;SAVE SIGN CHAR
                                ;17846          D_D.AND.K[.FF],                 ;GUARANTEE OTHER BYTES OF D = 0
U 0AC5, 0819,0034,49F8,F990,0000,0AC6   ;17847          Q_Q                             ;SHIFT IN OS ALSO
                                ;17848
                                ;17849          ;--------------------------------;
                                ;17850          Q_R[R2].AND.K[.FF],             ;Q<7:0>=FILL,
U 0AC6, 0D18,0034,49C0,FA10,0000,0AC8   ;17851          D_DAL.SC                        ;D<15:8>=SIGN
                                ;17852
                                ;17853          ;--------------------------------;
                                ;17854  EDLOADPORM1:                             ;
U 0AC8, 001D,2030,0180,FA90,0000,01C4   ;17855          R[R2]_Q.OR.D,J/EDITNEXT         ;R[R2]<15:8>=SIGN,<7:0>=FILL
                                ;17856
                                ;17857  =0111   ;--------------------------------;PSLCC<N>
                                ;17858  EDLOADPLUS:                             ;
U 0807, 0000,803C,0180,4000,0000,0AC5   ;17859          D[BYTE]_CACHE,J/EDLOADPORM      ;PSLCC<N> = 0
                                ;17860
                                ;17861          ;1111-----------------------------;
                                ;17862          LAB_R[R3],                      ;PSLCC<N> = 1
                                ;17863          STATE_STATE.AN.NOTPREDEC,       ;MAINTAIN ONLY STATE<7>
U 080F, 0000,003C,5980,FA18,1404,4AB5   ;17864          J/EDITN1                        ;
                                ;17865
                                ;17866  =0111   ;--------------------------------;PSLCC<N>
                                ;17867  EDLOADMINUS:                            ;
                                ;17868          LAB_R[R3],                      ;PSLCC<N> = 0
                                ;17869          STATE_STATE.AN.NOTPREDEC,       ;MAINTAIN ONLY STATE<7>
U 0817, 0000,003C,5980,FA18,1404,4AB5   ;17870          J/EDITN1                        ;
                                ;17871
                                ;17872          ;1111-----------------------------;
U 081F, 0000,803C,0180,4000,0000,0AC5   ;17873          D[BYTE]_CACHE,J/EDLOADPORM      ;PSLCC<N> = 1
                                ;17874
                                ;17875          ;--------------------------------;
U 0AC9, 0000,003C,3D80,3C00,0000,01C4   ;17876  EDSIG:  IDLPSL]_D,J/EDITNEXT            ;CLEAR + SET SIG
                                ;17877
                                ;17878  =101    ;--------------------------------;PSLCC<C>
                                ;17879  EDINSERT:                               ;
                                ;17880          VA_LA+K[.1],R[R5]_LA+K[.1],     ;INCREMENT DEST ADDR
                                ;17881          Q_Q,                            ;
U 09A5, 0018,0014,05F8,FAA8,0200,0ACC   ;17882          J/EDINSERTST1                   ;
                                ;17883
                                ;17884          ;111------------------------------;
U 09A7, 0000,803C,0180,4000,0000,09A5   ;17885          D[BYTE]_CACHE,J/EDINSERT        ;READ 1 BYTE
                                ;17886
                                ;17887          ;--------------------------------;
                                ;17888  EDINSERTST1:                            ;
                                ;17889          CACHE_D[BYTE],                  ;WRITE 1 BYTE(USED BY ENDFL + STORESIGN)
                                ;17890          STATE_STATE.OR.DEST,            ;SET STATE<2>
U 0ACC, 0000,803C,1180,3000,1404,21C4   ;17891          J/EDITNEXT                      ;
                                ;17892
                                ;17893          ;--------------------------------;
```

```
                                        ;17894  .TOC    ''      Edit instruction      : MOVE + FLOAT''
                                        ;17895
                                        ;17896  ;PATTERN = 91-9F, A1-AF
                                        ;17897  ;AT THIS POINT, EITHER:
                                        ;17898  ;NO INITIAL ZEROING REQD BEFORE THIS MOVE OR FLOAT OR
                                        ;17899  ;INITIAL ZEROING WAS JUST DONE + CAN DO THE 'REAL'' OPERATION
                                        ;17900
                                        ;17901  =0      ;-------------------------------;ALUCC <Z>
                                        ;17902  EDMOVEORFLOAT:                          ;
                                        ;17903          LAB_R[R1],                      ;
                                        ;17904          SC_R[.FFFC],                    ;-4 IN CASE LEFT NIBBLE
U 06F0, 0000,0C3C,F180,FA08,0084,69B2   ;17905          BEN/MUL,J/EDFLOATM1             ;BRANCH ON NIBBLE POSITION(D) +
                                        ;17906                                          ;ANY LEFT (SC = 0)
                                        ;17907          ;1-------------------------------;
                                        ;17908          LAB_R[R3],                      ;THIS PATTERN ENDED AFTER ZEROING
                                        ;17909          STATE_STATE.AN.NOTPREDEC,       ;MAINTAIN ONLY STATE<7>
U 06F1, 0000,003C,5980,FA18,1404,4AB5   ;17910          J/EDITN1                        ;
                                        ;17911
                                        ;17912  =010    ;-------------------------------;BEN/MUL. SC + D<0> USED
                                        ;17913  EDFLOATM1:                              ;
                                        ;17914          LAB_R[R3],                      ;SC = Q = 0
                                        ;17915          STATE_STATE.AN.NOTPREDEC,       ;MAINTAIN ONLY STATE<7>
U 09B2, 0000,003C,5980,FA18,1404,4AB5   ;17916          J/EDITN1                        ;
                                        ;17917
                                        ;17918          ;011------------------------------;
                                        ;17919          LAB_R[R3],                      ;SC = Q = 0
                                        ;17920          STATE_STATE.AN.NOTPREDEC,       ;MAINTAIN ONLY STATE<7>
U 09B3, 0000,003C,5980,FA18,1404,4AB5   ;17921          J/EDITN1                        ;
                                        ;17922
                                        ;17923          ;110------------------------------;
                                        ;17924          J/EDFLOATRIGHTNIB,              ;D<0> = 0 = RIGHT NIB
U 09B6, 0000,013C,0180,F800,0200,072C   ;17925          VA_LA,Z?                        ;
                                        ;17926
                                        ;17927          ;111------------------------------;
                                        ;17928          VA_LA+K[.1],R[R1]_LA+K[.1],      ;D<0> = 1 = LEFT NIB
                                        ;17929          INTRPT.STROBE,                  ;
U 09B7, 0018,0114,0580,FA88,4200,06F4   ;17930          Z?                              ;
```

```
                                        ;17931  =0      ;------------------------------ --;ALUCC <Z>
                                        ;17932  EDFLOATRSTLEFT:                           ;
                                        ;17933          D[BYTE]_CACHE,                    ;READ 1 BYTE
                                        ;17934          STATE_STATE.AN.DESTDBL,           ;CLEAR STATE<2:1>
                                        ;17935          BEN/INTERRUPT,                    ;
U 06F4, 0000,8E3C,D580,4000,1404,49C6   ;17936          J/EDFLOATLEFT1                    ;
                                        ;17937
                                        ;17938          ;1-------------------------------;
U 06F5, 0000,003C,0180,FA88,0000,072D   ;17939          R[R1]_LA,J/EDNOMORE               ;SRC LEN = 0
                                        ;17940
                                        ;17941  =110    ;-------------------------------;INTERRUPT
                                        ;17942  EDFLOATLEFT1:                             ;
U 09C6, 0D00,003C,0180,F800,0000,0ACD   ;17943          D_DAL.SC,J/EDFLOATAMBI            ;GET LEFT NIBBLE TO <3:0>
                                        ;17944
                                        ;17945          ;111------------------------------;
U 09C7, 0000,003C,1980,F800,0104,6B16   ;17946          J/EDITFPD,FE_K[ZERO]              ;
                                        ;17947
                                        ;17948  =0      ;-------------------------------;ALUCC <Z>
                                        ;17949  EDFLOATRIGHTNIB:                          ;
                                        ;17950          D[BYTE]_CACHE,                    ;
                                        ;17951          STATE_STATE.AN.DESTDBL,           ;CLEAR STATE<2:1>
U 072C, 0000,803C,D580,4000,1404,4ACD   ;17952          J/EDFLOATAMBI                     ;
                                        ;17953
                                        ;17954          ;1-------------------------------;
                                        ;17955  EDNOMORE:                                 ;
                                        ;17956          R[R0]_0-1,                        ;
U 072D, 001B,0000,0580,FA80,0000,0B06   ;17957          J/EDREPEATTOOMANYABORT            ;
```

```
                                ;17958          ;--------------------------------;
                                ;17959  EDFLOATAMBI:                             ;
                                ;17960          D_D.AND.K[.F],CLK.UBCC,         ;EXTRACT RIGHT NIBBLE
                                ;17961          LAB_R[R5],                      ;DEST ADDR
U 0ACD, 0819,1634,6180,FA28,0010,0819   ;17962          BEN/STATE?-4                    ;SEPARATE MOVE + FLOAT
                                ;17963
                                ;17964  =**01   ;--------------------------------;STATE <5> = MOVE OR FLOAT
                                ;17965          J/EDMOVEMORE,                   ;MOVE PATTERN
                                ;17966          Z?,                             ;NIB = 0?
                                ;17967          VA_LA+K[.1],                    ;PREPARE TO WRITE NEXT DEST LOCN
U 0819, 0018,0114,0580,FAA8,0200,0744   ;17968          R[R5]_LA+K[.1]                  ;INCREMENT DEST ADDR
                                ;17969
                                ;17970          ;**11---------------------------;
                                ;17971          Z?,                             ;FLOAT PATTERN
                                ;17972          VA_LA+K[.1],                    ;PREPARE TO WRITE NEXT DEST LOCN
U 081B, 0018,0114,0580,FAA8,0200,0740   ;17973          R[R5]_LA+K[.1]                  ;INCREMENT DEST ADDR
                                ;17974
                                ;17975  =0       ;--------------------------------;ALUCC <Z>
                                ;17976          J/EDFLOATNOT0,BEN/ROR,          ;
                                ;17977          STATE_STATE.OR.DEST,            ;NEXT OPERATION WILL BE A WRITE
                                ;17978                                          ;SET STATE<2>
U 0740, 0001,823C,118C,F998,1404,2A05   ;17979          RC[T3]_D,DT/BYTE                ;CHAR NOT 0. SAVE IT.
                                ;17980
                                ;17981          ;1------------------------------;
                                ;17982          J/EDFLOATEQ0,BEN/ROR,           ;CHAR = 0. TEST SIGNIFICANCE
U 0741, 0000,023C,1180,F800,1404,2A35   ;17983          STATE_STATE.OR.DEST             ;NEXT OPERATION WILL BE A WRITE
                                ;17984                                          ;SET STATE<2>
                                ;17985  =101    ;--------------------------------;PSLCC <C>
                                ;17986  EDFLOATNOT0:                            ;
                                ;17987          D_RC[T2],DT/BYTE,               ;NOT 0 + 1ST SIGNIF CHAR
                                ;17988                                          ;MUST STORE SIGN, SET SIGNIF, STORE CHAR
                                ;17989          SC_PSLADDR,                     ;PREPARE TO R/W PSL
U 0A05, 0810,8038,6180,F910,0084,6ACE   ;17990          J/EDFLOATNOSIG                  ;
                                ;17991
                                ;17992          ;111----------------------------;
                                ;17993          D_D.OR.ASCII,DT/BYTE,           ;NOT 0 + ALREADY SIGNIF
U 0A07, 0819,8030,7980,F800,0000,0AD6   ;17994          J/EDFLOATSIG                    ;MAKE IT AN ASCII CHAR
                                ;17995
                                ;17996  =101    ;--------------------------------;PSLCC <C>
                                ;17997  EDFLOATEQ0:                             ;
                                ;17998          D_RC[T1],DT/BYTE,               ;AN INSIGNIF 0. STORE FILL CHAR
U 0A35, 0810,8038,0180,F908,0000,0AD6   ;17999          J/EDFLOATSIG                    ;
                                ;18000
                                ;18001          ;111----------------------------;
                                ;18002  EDMOVEWR2:                              ;
                                ;18003          D_D.OR.ASCII,DT/BYTE,           ;A MEANINGFUL 0
U 0A37, 0819,8030,7980,F800,0000,0AD6   ;18004          J/EDFLOATSIG                    ;
```

```
                                ;18005          ;-------------------------------;
                                ;18006  EDFLOATNOSIG:                           ;
                                ;18007          CACHE_D[BYTE],                  ;STORE SIGN CHAR FOR NOT 0
U 0ACE, 0000,803C,0980,3000,1404 ;18008         STATE_STATE-K[.2]               ;NOTE THAT THIS IS A SPECIAL CASE(CLEAR <2>)
                                ;18009                                          ;OF MOVE/FLOAT WRITE, NAMELY, COUNTER
                                ;18010                                          ;DOESN'T NEED TO BE RESET AFTER CPD
                                ;18011                                          ;+ 1ST SIGNIF DIGIT CASE
                                ;18012          ;-------------------------------;
                                ;18013          Q_ID(SC),                       ;
U 0AD0, 0C00,003C,01F0,2400,0000,0AD1 ;18014    D_Q                             ;COPY COUNT
                                ;18015
                                ;18016          ;-------------------------------;
                                ;18017          D_Q.ANDNOT.PSWZ,                ;CLEAR PSLCC<Z>
                                ;18018          Q_D,                            ;RESTORE COUNT
U 0AD1, 0819,2024,11E0,FA28,0000,0AD2 ;18019    LAB_R[R5]                       ;
                                ;18020
                                ;18021          ;-------------------------------;
U 0AD2, 0819,0030,0580,F800,0000,0AD4 ;18022    D_D.OR.PSWC                     ;SET PSLCC<C>
                                ;18023
                                ;18024          ;-------------------------------;
                                ;18025          ID(SC)_D,                       ;SAVE PSL
                                ;18026          VA_LA+K[.1],                    ;PREPARE TO WRITE CHAR
U 0AD4, 0018,0014,0580,36A8,0200,0AD5 ;18027    R[R5]_LA+K[.1]                  ;INCREMENT DEST ADDR
                                ;18028
                                ;18029          ;-------------------------------;
                                ;18030          D_RC[T3],DT/BYTE,               ;REGET CHAR
                                ;18031          STATE_STATE.OR.DESTDBL,         ;SPECIAL CASE:2 WRITES/READ
U 0AD5, 0810,8038,D580,F918,1404,2A37 ;18032    J/EDMOVEWR2                     ;SET <2:1>
                                ;18033
                                ;18034          ;-------------------------------;
                                ;18035  EDFLOATSIG:                             ;
                                ;18036          CACHE_D[BYTE],                  ;
                                ;18037          Q_Q-K[.1],                      ;Q IS REALLY A NIBBLE.
                                ;18038          SC_Q-K[.1],                     ;SC USED FOR BEN/MUL FOR COUNT
U 0AD6, 0019,A000,05C0,3200,0082,0AD8 ;18039    LAB_R[R0]                       ;
                                ;18040
                                ;18041          ;-------------------------------;
                                ;18042          R[R0]_LA-K[.1],DT/BYTE,CLK.UBCC,;DECREMENT SRC COUNT
                                ;18043          D_LA-K[.1],                     ;PREPARE FOR LEFT/RT
U 0AD8, 0818,8000,0580,FA80,0010,06F0 ;18044    J/EDMOVEORFLOAT                 ;
```

```
                                        ;18045  =0      ;-------------------------------;ALUCC<Z>
                                        ;18046  EDMOVEMORE:                             ;
                                        ;18047          J/EDMOVESIGNIF,                 ;DIGIT NOT = 0
                                        ;18048          RC[T3]_Q,                       ;SAVE COUNT
U 0744, 0001,203C,1180,F998,14          ;18049          STATE_STATE.OR.DEST             ;NOTE A WRITE IS COMING UP SOON
                                        ;18050                                          ;SET STATE<2>
                                        ;18051          ;1------------------------------;
                                        ;18052          J/EDFLOATEQ0,                   ;DIGIT = 0
                                        ;18053          BEN/ROR,                        ;
U 0745, 0000,023C,1180,F800,1404,2A35   ;18054          STATE_STATE.OR.DEST             ;NOTE A WRITE IS COMING UP SOON
                                        ;18055                                          ;SET STATE<2>
                                        ;18056          ;-------------------------------;
                                        ;18057  EDMOVESIGNIF:                           ;
U 0AD9, 0000,003C,3DF0,2C00,0000,0ADA   ;18058          Q_ID[PSL]                       ;READ PSL
                                        ;18059
                                        ;18060          ;-------------------------------;
                                        ;18061          D_0.OR.PSWC,                    ;SET PSLCC<C>
U 0ADA, 0819,2030,05E0,F800,0000,0AE0   ;18062          Q_D                             ;COUNT CHAR
                                        ;18063
                                        ;18064          ;-------------------------------;
U 0AE0, 0819,0024,1180,F800,0000,0AE1   ;18065          D_D.ANDNOT.PSWZ                 ;GUARANTEE PSL Z BIT = 0
                                        ;18066
                                        ;18067          ;-------------------------------;
                                        ;18068          ID[PSL]_D,                      ;WRITE PSL
                                        ;18069          D_Q,                            ;RESTORE CHAR
                                        ;18070          Q_RC[T3],                       ;RESTORE COUNT
U 0AE1, 0C10,0038,3DC0,3D18,0000,0A37   ;18071          J/EDMOVEWR2                     ;+ WRITE IT
                                        ;18072
                                        ;18073          ;-------------------------------;
```

```
                                        ;18074  .TOC    ''      Edit instruction      : OTHER PATTERNS''
                                        ;18075
U OAE4, 0810,8038,0180,F800,0000,0AE6   ;18076  EDFILL: D_LC,DT/BYTE                    ;81-8F
                                        ;18077
                                        ;18078          ;-------------------------------;
                                        ;18079  EDFILL1:                               ;
                                        ;18080          VA_LA+K[.1],R[R5]_LA+K[.1],     ;INCR. DEST. ADDR
U OAE6, 0018,0114,0580,FAA8,0200,075C   ;18081          Z?                              ;
                                        ;18082
                                        ;18083  =0      ;-------------------------------;ALUCC <Z>
                                        ;18084  EDFILL2:                               ;
                                        ;18085          CACHE_D[BYTE],                  ;WRITE 1 BYTE OF FILL
                                        ;18086          LAB_R[R5],                      ;
                                        ;18087          Q_Q-K[.1],CLK.UBCC,             ;DECREMENT COUNT
U 075C, 0019,A000,05C0,3228,0010,0AE6   ;18088          J/EDFILL1                       ;
                                        ;18089
                                        ;18090          ;1------------------------------;ALUCC <Z> = 1
U 075D, 0000,003C,0180,FAA8,0000,01C4   ;18091          J/EDITNEXT,R[R5]_LA             ;R[R5] = LAST BYTE WRITTEN
                                        ;18092
                                        ;18093          ;-------------------------------;
                                        ;18094  EDBLANK0:                              ;
                                        ;18095          D_D.AND.K[.FF],                 ;GUARANTEE BYTES 3-1 OF LEN = 0
U OAE8, 0819,1A34,4980,F800,0C00,082B   ;18096          BEN/PSL.CC                      ;TEST PSLCC<Z>
                                        ;18097
                                        ;18098  =1011   ;-------------------------------;PSLCC<Z>
U 082B, 0000,183C,0180,F800,0000,0780   ;18099          J/EDBLANKONOT0,D.B0?            ;PSLCC<Z>=0
                                        ;18100
                                        ;18101          ;1111----------------------------;
                                        ;18102          D.B0?,                          ;PSLCC<Z>=1
                                        ;18103          D_LA-D,                         ;AMOUNT TO BACKUP BY
U 082F, 081C,3800,01E0,F800,0000,076C   ;18104          Q_D                             ;MOVE COUNTER TO Q
                                        ;18105
                                        ;18106  =***0   ;-------------------------------;D BYTES 3 - 1 = 0
                                        ;18107          R[R5]_LA+K[.1],                 ;LEN = 0
U 076C, 0018,0014,0580,FAA8,0000,0491   ;18108          J/EDUNPREDICTABLE               ;
                                        ;18109
                                        ;18110          ;***1----------------------------;
U 076D, 0019,0014,0580,FAA8,0200,0AE9   ;18111          VA_D+K[.1],R[R5]_D+K[.1]        ;BACK UP DEST ADDR
                                        ;18112
                                        ;18113          ;-------------------------------;
                                        ;18114  EDFILLRST:                             ;
                                        ;18115          D_RC[T1],DT/BYTE,               ;FILL CHAR
                                        ;18116          STATE_STATE.OR.FILL,            ;FILL = BLANK0 = STATE<2:0> = 7
U OAE9, 0810,8038,5D80,F908,1404,275C   ;18117          J/EDFILL2                       ;
                                        ;18118
                                        ;18119  =***0   ;-------------------------------;D.BYTE 0. KNOW BYTES 3-1 = 0
                                        ;18120  EDBLANKONOTC:                          ;
                                        ;18121          R[R5]_LA+K[.1],                 ;LEN = 0
U 0780, 0018,0014,0580,FAA8,0000,0491   ;18122          J/EDUNPREDICTABLE               ;
                                        ;18123
                                        ;18124          ;***1----------------------------;
                                        ;18125          LAB_R[R3],                      ;
                                        ;18126          STATE_STATE.AN.NOTPREDEC,       ;MAINTAIN ONLY STATE<7>
U 0781, 0000,003C,5980,FA18,1404,4AB5   ;18127          J/EDITN1                        ;
```

```
                                ;18128  EDREPLACESIGN:                          ;PATTERN = 46
U 0AEA, 0819,0034,4980,F800,0000,0AEC   ;18129          D_D.AND.K[.FF]          ;CLEAR D BYTES 3-1 FOR NARROWER BEN
                                ;18130
                                ;18131          ;-----------------------------;
                                ;18132          Q_D-K[.1],              ;Q = NEXT PATTERN BYTE(POSITION)-1
                                ;18133          D_Q,                    ;D<7:0> = FILL CHAR
U 0AEC, 0C19,1800,05C0,F800,0000,0784   ;18134          D.BÓ?                   ;COUNT = 0?
                                ;18135
                                ;18136  =***0   ;-----------------------------;D.BYTE 0
                                ;18137          J/EDUNPREDICTABLE,      ;
U 0784, 0018,0014,0580,FAA8,0000,0491   ;18138          R[R5]_LA+K[.1]          ;INCREMENT DEST ADDR
                                ;18139
                                ;18140          ;***1-----------------------;
                                ;18141          VA_LA-Q,                ;DEST ADDR-POSITION
U 0785, 001C,1A00,0180,F800,0200,0833   ;18142          BEN/PSL.CC              ;ONLY N+Z CASE OF INTEREST
                                ;18143
                                ;18144  =0011   ;-----------------------------;PSLCC<N+Z>
                                ;18145          LAB_R[R3],              ;
                                ;18146          STATE_STATE.AN.NOTPREDEC,       ;MAINTAIN ONLY STATE<7>
U 0833, 0000,003C,5980,FA18,1404,4AB5   ;18147          J/EDITN1                ;
                                ;18148
                                ;18149          ;0111-----------------------;
                                ;18150          LAB_R[R3],              ;
                                ;18151          STATE_STATE.AN.NOTPREDEC,       ;MAINTAIN ONLY STATE<7>
U 0837, 0000,003C,5980,FA18,1404,4AB5   ;18152          J/EDITN1                ;
                                ;18153
                                ;18154  ; ************************************************
                                ;18155  ; * Patch no. 041, PCS 0837 trapped to WCS 1175 *
                                ;18156  ; ************************************************
                                ;18157
                                ;18158          ;1011-----------------------;
                                ;18159          LAB_R[R3],              ;
                                ;18160          STATE_STATE.AN.NOTPREDEC,       ;MAINTAIN ONLY STATE<7>
U 083B, 0000,003C,5980,FA18,1404,4AB5   ;18161          J/EDITN1                ;
                                ;18162
                                ;18163          ;1111-----------------------;
                                ;18164          Q_0,                    ;N+Z = 1 = -0
                                ;18165          STATE_STATE.OR.PATT2,           ;STATE<1> SETTING IS REDUNDANT
                                ;18166          CACHE_D[BYTE],          ;REGET PATTERN BYTE + RESTART IF FAULTED
U 083F, 0000,803C,09F8,3000,1404,21C4   ;18167          J/EDITNEXT              ;
                                ;18168
                                ;18169          ;-----------------------------;
```

```
                                    ;18170  .TOC    ""      Edit instruction      : ADJUST INPUT''
                                    ;18171
                                    ;18172  EDADJUSTINPUT:                                ;PATTERN = 47
U 0AED, 0819,0034,4980,F800,0000,0AEE   ;18173          D_D.AND.K[.FF]                    ;CLEAN OUT BYTES 3-1
                                    ;18174
                                    ;18175          ;--------------------------------;
                                    ;18176          D.B0?,                            ;ADJ LEN = 0?
                                    ;18177          ALU_D.ANDNOT.K[.1F],              ;SAVE ONLY <4:0>
U 0AEE, 0019,1824,8D80,F800,0010,078C   ;18178          CLK.UBCC                          ;CHECK ON ADJ LEN > 31
                                    ;18179
                                    ;18180  =***0   ;--------------------------------;D.BYTE 0
                                    ;18181  EDUP1:  R[R5]_LA+K[.1],                   ;ADJ LEN = 0
U 078C, 0018,0014,0580,FAA8,0000,0491   ;18182          J/EDUNPREDICTABLE                 ;
                                    ;18183
                                    ;18184          ;***1-----------------------------;
                                    ;18185          Q_R[R0].AND.K[.FF],               ;Q = SRC LEN
                                    ;18186          C[K.UBCC,                         ;CHECK IF SRC LEN = 0
U 078D, 0018,0134,49C0,FA00,0010,0794   ;18187          J/EDADJINNOTO,Z?                  ;BRANCH ON ADJ > 31
                                    ;18188
                                    ;18189  =0      ;--------------------------------;ALUCC <Z>
                                    ;18190  EDADJINNOTO:                              ;
U 0794, 0000,003C,0180,FA28,0000,078C   ;18191          J/EDUP1,LAB_R[R5]                 ;ADJ LEN > 31
                                    ;18192
                                    ;18193          ;1---------------------------------;
                                    ;18194          Q_Q-D,DT/BYTE,CLK.UBCC,           ;SRC LEN - ADJ LEN
                                    ;18195          D_Q,                              ;D = R[R0] = SRC LENGTH LEFT
                                    ;18196          SC_K[.FFE8],                      ;PREPARE FOR RT 24 SHIFT
U 0795, 0C1D,A100,69C0,F800,0094,679C   ;18197          Z?                                ;BRANCH ON SRC LEN = 0
                                    ;18198
                                    ;18199  =0      ;--------------------------------;ALUCC <Z>
                                    ;18200                                            ;SRC LEN NOT 0
                                    ;18201          BEN/ALU,                          ;BRANCH ON SRC LEN - REQ LEN
                                    ;18202          D_D.AND.K[.1].                    ;ONLY NEED BIT 0 FOR TEST
U 079C, 0819,1B34,0580,F800,0000,084A   ;18203          J/EDADJIN1                        ;
                                    ;18204
                                    ;18205          ;1---------------------------------;SRC LEN = 0
                                    ;18206          Q_Q.AND.K[.FF],                   ;BY DEFN., IF REQD LEN
U 079D, 0019,2034,49C0,F800,0000,0AF0   ;18207          J/EDADJINFINI                     ;> 0 + SRC LEN = 0, SRC LEN < REQD LEN
```

```
                                        ;18208  =1010   ;-------------------------------;ALUCC <Z,C>
                                        ;18209  EDADJIN1:                               ;SRC<REQD
U 084A, 0019,2034,49C0,F800,0000,0AF0   ;18210          Q_Q.AND.K[.FF],J/EDADJINFINI   ;SRC LEN < REQ LEN
                                        ;18211
                                        ;18212          ;1011-------------------------;SRC>REQD
                                        ;18213  EDADJIN2:                               ;
                                        ;18214          VA_R[R1],                      ;ASSUME IT'S RIGHT NIBBLE
                                        ;18215          INTRPT.STROBE,                 ;
                                        ;18216          STATE_STATE.AN.PREDECZERO,     ;ABOUT TO READ, SO NO LONGER PREDEC
                                        ;18217                                         ;(PREDEC=C0, MAINTAIN <6:0>)
U 084B, 0000,193C,D180,FA08,5604,47B4   ;18218          D0?,J/EDADJIN3                 ;BRANCH ACC TO BIT 0 OF COUNT
                                        ;18219
                                        ;18220          ;-------------------------------;
                                        ;18221  ;1110                                  ;ILLEGAL COMBINATION
                                        ;18222  =1111   ;-------------------------------;SRC = REQD
                                        ;18223          LAB_R[R3],                     ;
                                        ;18224          STATE_STATE.AN.NOTPREDEC,      ;MAINTAIN ONLY STATE<7>
U 084F, 0000,003C,5980,FA18,1404,4AB5   ;18225          J/EDITN1                       ;
                                        ;18226
                                        ;18227          ;-------------------------------;
                                        ;18228  EDADJINFINI:                           ;
U 0AF0, 0D00,003C,0180,FA00,0000,0AF1   ;18229          D_DAL.SC,LAB_R[R0]             ;GET COUNT IN Q<7:0> TO D<15:8>
                                        ;18230
                                        ;18231          ;-------------------------------;
U 0AF1, 001C,2030,0180,FA80,0000,01C4   ;18232          R[R0]_LA.OR.D,J/EDITNEXT       ;
                                        ;18233
                                        ;18234  =***0   ;-------------------------------;D BIT 0. <3:1> = 0
                                        ;18235  EDADJIN3:                               ;
                                        ;18236          D[BYTE]_CACHE,                 ;READ A BYTE
                                        ;18237          STATE_STATE.OR.ADJINP,         ;NOTE IT'S ADJUST INPUT(STATE<1:0>=3)
                                        ;18238          LAB_R[R0],                     ;
                                        ;18239          BEN/INTERRUPT,                 ;INTERRUPT PENDING?
U 07B4, 0000,8E3C,0D80,4200,1404,2A46   ;18240          J/EDADJIN5                     ;
                                        ;18241
                                        ;18242          ;***1-------------------------;
                                        ;18243          J/EDADJIN3,INTRPT.STROBE,      ;
                                        ;18244          VA_LA+K[.1],                   ;PREPARE TO READ NEXT SRC BYTE
U 07B5, 0018,0014,0580,FA88,4200,07B4   ;18245          R[R1]_LA+K[.1]                 ;INCRMENT SRC ADDR
                                        ;18246
                                        ;18247  =0      ;-------------------------------;ALUCC <Z>
                                        ;18248  EDADJIN4:                               ;
                                        ;18249          D[BYTE]_CACHE,                 ;READ A BYTE
                                        ;18250          STATE_STATE.OR.ADJINP,         ;NOTE IT'S ADJUST INPUT(STATE<1:0>=3)
                                        ;18251          LAB_R[R0],                     ;
                                        ;18252          BEN/INTERRUPT,                 ;INTERRUPT PENDING?
U 07C4, 0000,8E3C,0D80,4200,1404,2A46   ;18253          J/EDADJIN5                     ;
                                        ;18254
                                        ;18255          ;1----------------------------;
                                        ;18256          J/EDITNEXT,                    ;REQUESTED COUNT ALL DONE
U 07C5, 0000,003C,0180,FA88,0000,01C4   ;18257          R[R1]_LA                       ;DON'T INCREMENT SRC ADDR
```

```
                                ;18258  =110    ;--------------------------------;INTERRUPT
                                ;18259  EDADJIN5:                               ;
                                ;18260          R[R0]_LA-K[.1],DT/BYTE,         ;DECREMENT SRC LENGTH
                                ;18261          SC_K[.1],                       ;TO EASE CONSTRAINT ON BEN/MUL
                                ;18262          BEN/ROR,                        ;LA <0> TO DETERMINE IF LEFT OR RT NIB
U 0A46, 0018,8200,0580,FA80,0084,6A56   ;18263          J/EDADJNOINT                    ;
                                ;18264
                                ;18265          ;111----------------------------;
U 0A47, 0000,003C,1980,F800,0104,6B16   ;18266          J/EDITFPD,FE_K[ZERO]            ;
                                ;18267
                                ;18268  =110    ;--------------------------------;LA<0>
                                ;18269  EDADJNOINT:                             ;
                                ;18270          D_D.AND.K[.F],CLK.UBCC,         ;EXTRACT RIGHT NIBBLE
U 0A56, 0819,0034,6180,F800,0010,0AF4   ;18271          J/EDADJINRIGHT                  ;
                                ;18272
                                ;18273          ;111----------------------------;
                                ;18274          D_D.AND.K[.F0],                 ;EXTRACT LEFT NIB
                                ;18275          C[K.UBCC,                       ;SEE IF IT'S 0
U 0A57, 0819,0034,CD80,F800,0010,0AF6   ;18276          J/EDADJINLEFT                   ;
                                ;18277          ;--------------------------------;
                                ;18278  EDADJINRIGHT:                           ;
                                ;18279          Z?,                             ;BRANCH ON RIGHT NIB = 0
                                ;18280          Q_Q-K[.1],                      ;DECREMENT REQUESTED LENGTH
                                ;18281          D_Q-K[.1],                      ;A COPY OF DECREMENTED LENGTH
                                ;18282          SC_SC-K[.1],                    ;SC_0 FOR BEN/MUL
                                ;18283          CLR.UBCC,                       ;SEE IF IT HIT 0
U 0AF4, 0819,2100,05C0,FA08,0094,A7E0   ;18284          LAB_R[R1]                       ;LATCH SRC ADDR
                                ;18285
                                ;18286  =0      ;--------------------------------;ALU <Z>
                                ;18287          J/EDADJINNEQ0,BEN/MUL,          ;NIB NOT 0
U 07E0, 0600,0C3C,0180,F800,0000,0822   ;18288          D_D.RIGHT                       ;COUNTER/2
                                ;18289
                                ;18290          ;1------------------------------;
                                ;18291          VA_LA+K[.1],R[R1]_LA+K[.1],     ;INCREMENT SRC ADDR
                                ;18292          INTRPT.STROBE,                  ;CHECK ON INTERRUPTS
U 07E1, 0018,0114,0580,FA88,4200,07C4   ;18293          J/EDADJIN4,Z?                   ;BRANCH ON REQ LEN DONE
                                ;18294
                                ;18295          ;--------------------------------;
                                ;18296  EDADJINLEFT:                            ;
                                ;18297          Q_Q-K[.1],                      ;DECREMENT REQ LENGTH
                                ;18298          D_Q-K[.1],                      ;COPY THE DECREMENTED LENGTH
                                ;18299          C[K.UBCC,                       ;SEE IF IT'S 0
                                ;18300          LAB_R[R1],                      ;LATCH SRC ADDR
U 0AF6, 0819,2100,05C0,FA08,0010,07EC   ;18301          Z?                              ;BRANCH ON NIB = 0
                                ;18302
                                ;18303  =0      ;--------------------------------;ALU <>
                                ;18304          J/EDADJINNEQ0,                  ;NIB NOT = 0
U 07EC, 0600,003C,0180,F800,0000,0822   ;18305          D_D.RIGHT                       ;COUNTER/2 (COUNT VIA BYTES)
                                ;18306
                                ;18307          ;1------------------------------;
                                ;18308          VA_LA,INTRPT.STROBE,            ;NIB = 0
U 07ED, 0000,013C,0180,F800,4200,07C4   ;18309          J/EDADJIN4,Z?                   ;BRANCH ON REQ LEN = 0
```

```
                                ;18310  =*10    ;--------------------------------;D<0>. KNOW SC = 0
                                ;18311  EDADJINNEQ0:                             ;
                                ;18312          R[R1]_LA+D,                      ;ADVANCE SRC ADDR
                                ;18313          D_Q,                             ;COPY OF DECREMENTED COUNTER
                                ;18314          Q_ID[PSL],                       ;
U 0822, 0C1C,2014,3DF0,2E88,0000,0AF8   ;18315          J/EDADJINPSL                     ;
                                ;18316
                                ;18317  ;FOR THE CASE WHERE THE 1ST SIGNIF DIGIT IS A RIGHT NIB,
                                ;18318  ;+ THE ADDR IS TO BE LEFT POINTING AT THE RIGHT NIB
                                ;18319  ;BECAUSE THE NEXT READ IS OF THE LEFT NIB + WILL INCREMENT THE ADDR
                                ;18320
                                ;18321          ;*11-----------------------------;
                                ;18322          R[R1]_LA+D+1,                    ;ADVANCE SRC ADDR
                                ;18323          D_Q,                             ;COPY OF DECREMENTED COUNTER
                                ;18324          Q_ID[PSL],                       ;
U 0823, 0C1C,2010,3DF0,2E88,0000,0AF8   ;18325          J/EDADJINPSL                     ;
                                ;18326
                                ;18327          ;--------------------------------;
                                ;18328  EDADJINPSL:                              ;
                                ;18329          D_Q.ANDNOT.PSWZ,                 ;CLEAR Z
U 0AF8, 0819,2024,11E0,F800,0000,0AF9   ;18330          Q_D                              ;RESTORE COUNTER
                                ;18331
                                ;18332          ;--------------------------------;
                                ;18333          D_D.OR.PSWV,                     ;SET V
U 0AF9, 0819,0030,0980,FA00,0000,0AFC   ;18334          LAB_R[R0]                        ;
                                ;18335
                                ;18336          ;--------------------------------;
                                ;18337          ID[PSL]_D,                       ;WRITE BACK THE PSL
                                ;18338          R[R0]_LA-Q,DT/BYTE,              ;REDUCE COUNTER BY ADDL AMOUNT
U 0AFC, 001C,8000,3D80,3E80,0000,01C4   ;18339          J/EDITNEXT                       ;
                                ;18340
                                ;18341          ;--------------------------------;
```

```
                                        ;18342  .TOC    ""      Edit instruction      : TERMINATION""
                                        ;18343
                                        ;18344  EDITDONE:                                ;END OPERATOR ENCOUNTERED
                                        ;18345                                           ;ALL SRC USED.
                                        ;18346          Q_R[R4],                         ;ORIGINAL SRC LENGTH
U 0AFE, 0000,003C,59C0,FA20,1404,4B00   ;18347          STATE_STATE.AN.6TO4              ;RETAIN ONLY BIT 7
                                        ;18348
                                        ;18349          ;--------------------------------;
U 0B00, 0001,203C,01B0,FA80,0000,0B02   ;18350          Q_Q.RIGHT,R[R0]_Q                ;R0 RESTORED. Q = LEN/2
                                        ;18351
                                        ;18352          ;--------------------------------;
                                        ;18353          LAB_R[R1],                       ;CURRENT SRC ADDR
U 0B02, 0000,163C,0180,FA08,0000,0460   ;18354          BEN/STATE7-4                     ;MAY NEED TO UN-PREDEC SRC ADDR
                                        ;18355
                                        ;18356  =0***   ;--------------------------------;STATE<7>
                                        ;18357          R[R1]_LA-Q,                      ;REGEN ORIG SRC ADDR
U 0460, 001C,0000,0180,FA88,0000,0B04   ;18358          J/EDITD1                         ;
                                        ;18359
                                        ;18360          ;1***-----------------------------;PRE-DEC TO CORRECT
U 0468, 0018,0014,0580,FA88,0000,0B04   ;18361          R[R1]_LA+K[.1]                   ;PRE-DEC + 1 = AS WAS
                                        ;18362
                                        ;18363          ;--------------------------------;
                                        ;18364  EDITD1: R[R4]_D,                         ;R4_0
                                        ;18365          Q_ID[CES],                       ;READ CES REGISTER
U 0B04, 0001,1A3C,31F0,2EA0,2000,085D   ;18366          BEN/PSL.CC,CLR.FPD               ;CLEAR FPD BIT
                                        ;18367
                                        ;18368  =1101   ;--------------------------------;PSLCC <V>
                                        ;18369  EDEXIT:                                  ;
                                        ;18370          PC&VA_PC,                        ;
U 085D, 2014,0038,0180,F801,4200,00AB   ;18371          FLUSH.IB,J/IB.FILL               ;
                                        ;18372
                                        ;18373          ;1111-----------------------------;
U 085F, 0819,2030,6580,F800,0000,0B05   ;18374          D_Q.OR.K[.10]                    ;SET INTEGER OVERFLOW IN CES
                                        ;18375
                                        ;18376          ;--------------------------------;
U 0B05, 0000,003C,3180,3C00,0000,085D   ;18377          ID[CES]_D,J/EDEXIT               ;PSL<V> =1. CAN BE TRAPPED AT IRD
```

```
                                    ;18378          ;-------------------------------;
                                    ;18379  EDREPEATTOOMANYABORT:                   ;MOVE OR FLOAT EXHAUSTED
U 0B06, 0000,003C,0180,FA28,0000,0B08 ;18380          LAB_R[R5]                     ;SRC LENGTH BEFORE ALL DONE
                                    ;18381
                                    ;18382          ;-------------------------------;
                                    ;18383          R[R5]_LA+K[.1],               ;ADDR OF NEXT DEST BYTE
U 0B08, 0018,0014,0580,FAA8,2400,0106 ;18384          SET.FPD,J/RSVOPR              ;GUARANTEE FPD IS SET
                                    ;18385
                                    ;18386          ;-------------------------------;
                                    ;18387
                                    ;18388  ;EOEND ENCOUNTERED WHEN SRC LENGTH NOT = 0
                                    ;18389  ;OR AN UNDEFINED OPERATOR ENCOUNTERED
                                    ;18390  ;NEED TO STRAIGHTEN OUT R0 + CHECK ON PRE-DEC
                                    ;18391
                                    ;18392  EDREPEATTOOFEWABORT:                   ;
                                    ;18393          D_R[R0].AND.K[.FFFF],         ;CURRENTLY <15:8>=ADJ INP NEG COUNTER
U 0B0A, 0818,0034,C1C0,FA00,0000,0B0C ;18394          Q_R[R0].AND.K[.FFFF]          ;<7:0> = SRC LEN
                                    ;18395                                        ;NEED BYTE 1 IN BYTE 2 POSITION
                                    ;18396          ;-------------------------------;
                                    ;18397          D_D.SWAP,                     ;D<31:24>=SRC LEN,<23:16>=ADJ INP CNTR
                                    ;18398                                        ;D<15:0>=0
                                    ;18399          ALU_Q.SXT[WORD].ANDNOT.K[.FF], ;Q<31:16>=SIGN EXT,
U 0B0C, 0B1A,6024,49C0,F800,0000,0B0D ;18400          Q_ALU                         ;Q<15:7>=ADJ INP CNTR
                                    ;18401                                        ;Q<7:0> = 0
                                    ;18402          ;-------------------------------;
                                    ;18403          D_DAL.SC,                     ;D<31:24>=SXT, <23:16>=ADJ INP CNTR,
                                    ;18404                                        ;D<15:7>= 0, D<7:0>=SC LEN
                                    ;18405          LAB_R[R1],                    ;MAY NEED + 1
U 0B0D, 0D00,003C,5980,FA08,1404,4B0E ;18406          STATE_STATE.AN.6TO4           ;CLEAR OUT ALL BUT <7>
                                    ;18407
                                    ;18408  ; ************************************************
                                    ;18409  ; * Patch no. 097, PCS 0B0D trapped to WCS 11A5 *
                                    ;18410  ; ************************************************
                                    ;18411
                                    ;18412          ;-------------------------------;
                                    ;18413          BEN/STATE7-4,                 ;SEE IF R1 NEEDS UN-PRE-DECREMENT
U 0B0E, 0001,163C,0180,FA80,0000,0491 ;18414          R[R0]_D                       ;
                                    ;18415
                                    ;18416  =0***   ;-------------------------------;STATE<7>
                                    ;18417  EDUNPREDICTABLE:                      ;ALL BETS OFF
U 0491, 0000,003C,0180,F800,2400,0106 ;18418          SET.FPD,J/RSVOPR              ;
                                    ;18419
                                    ;18420          ;1***-------------------------;R1 NEEDS INCREMENT
                                    ;18421          R[R1]_LA+K[.1],               ;
U 0499, 0018,0014,0580,FA88,2400,0106 ;18422          SET.FPD,J/RSVOPR              ;
```

```
                                    ;18423  EDMATGT31:                          ;R0 = LENGTH OF STRING
                                    ;18424                                      ;R1 = START OF SRC
                                    ;18425                                      ;R5 = START OF DEST
U 0B10, 0010,0038,01C0,F910,0000,0B11   ;18426          Q_RC[T2]               ;LOAD PATTERN ADDR
                                    ;18427
                                    ;18428          ;---------------------------;
U 0B11, 0001,203C,0180,FA98,0000,0B14   ;18429          R[R3]_Q                ;LEAVE START OF SRC IN R3
                                    ;18430
                                    ;18431          ;---------------------------;
U 0B14, 0003,003C,0180,FAA0,0000,0B15   ;18432          R[R4]_0                ;
                                    ;18433
                                    ;18434          ;---------------------------;
                                    ;18435          R[R2]_0,                ;
U 0B15, 0003,003C,0180,FA90,2400,0106   ;18436          SET.FPD,J/RSVOPR       ;REQ LEN > 31
                                    ;18437
                                    ;18438          ;---------------------------;
```

```
                                        ;18439  .TOC    ""      Edit instruction      : FPD + RESTART""
                                        ;18440
                                        ;18441  ;ALGORITHM:
                                        ;18442  ;       AFTER HANDLING AN INTERRUPT OR EXCEPTION, EDITPC
                                        ;18443  ;       IS TO RESUME. ALL INFO ABOUT WHERE TO DO SO IS MAINTAINED
                                        ;18444  ;       IN THE STATE REGISTER. IT IS ALLOCATED AS FOLLOWS:
                                        ;18445  ;IF STATE <6:4> = 0, THEN STATE <2:0> ALLOCATED AS:
                                        ;18446  ;0000   FIRST   REREAD LENGTH OF STRING
                                        ;18447  ;0001   PATTERN 1       READING 1ST BYTE OF PATTERN. REREAD R3
                                        ;18448  ;0010   PATTERN 2       READING 2ND BYTE OF PATTERN. DECR RA 3 BY
                                        ;18449  ;                       1 AND REGET PATTERN
                                        ;18450  ;0011   ADJUST INPUT
                                        ;18451  ;0100   (UNUSED. GENL DEST MODE)
                                        ;18452  ;0101   END FLOAT OR STORE SIGN
                                        ;18453  ;0110   INSERT (EQUIV. TO PATT2 + WRITE)
                                        ;18454  ;0111   FILL OR BLANK0
                                        ;18455  ;BIT 3 UNUSED
                                        ;18456  ;IF STATE<6:4> NOT 0, THEN STATE<2:1> ALLOCATED AS:
                                        ;18457  ;00     MOVE/FLOAT READ
                                        ;18458  ;01     FLOAT SNGL
                                        ;18459  ;10     MOVE/FLOAT WRITE
                                        ;18460  ;11     FLOAT DBL
                                        ;18461  ;STATE <6:4> ALLOCATED AS:
                                        ;18462  ;000    NOT MOVE OR FLOAT OF ANY FLAVOR
                                        ;18463  ;001    MOVE
                                        ;18464  ;010    FLOAT
                                        ;18465  ;011    UNDEFINED
                                        ;18466  ;100    UNDEFINED
                                        ;18467  ;101    PRE-ZEROING PART OF MOVE
                                        ;18468  ;110    PRE-ZEROING PART OF FLOAT
                                        ;18469  ;111    UNDEFINED
                                        ;18470  ;STATE <7> = PREDEC CASE: ORIGINAL COUNT WAS ODD, SO R1 DECREMENTED
                                        ;18471  ;        SO INCREMENTATION IN LOOPS WORKS, BUT THIS INCREMENTATION
                                        ;18472  ;        HASN'T OCCURRED YET
                                        ;18473  ;FOR UNDEFINED STATE COMBINATIONS, J/RSVOPR.
                                        ;18474
                                        ;18475  EDITFPD:                                ;CALLED WITH Q = COUNT(OR 0 IF IRRELEVANT)
U 0B16, 0F00,003C,7180,F800,0084,6B18   ;18476          SC_K[.FFF8],D_0                 ;SC_-8
                                        ;18477
                                        ;18478          ;-------------------------------;
                                        ;18479          D_DAL.SC,                       ;ROTATE SO COUNT FROM Q<7:0> TO  D <31:24>
U 0B18, 0D18,0034,C1C0,FA10,0000,0B19   ;18480          Q_R[R2].AND.K[.FFFF]            ;SIGN<15:8> + FILL<7:0> IN R2
                                        ;18481
                                        ;18482          ;-------------------------------;
                                        ;18483          R[R2] Q.OR.D,SC_STATE,          ;OR COUNT INTO HI BYTE OF R2
U 0B19, 001D,2030,0180,FA90,1480,06D8   ;18484          J/FPDPACK                       ;GO PUT PC DELTA + STATE INTO R0 HIGH
                                        ;18485
                                        ;18486          ;-------------------------------;
```

```
                                    ;18487  41:
                                    ;18488  ;THIS ADDR SHARED BY CRC(OP-CODE = 0B), MATCHC(39), + EDITPC(38)
                                    ;18489  CRCRESTART:                                ;
                                    ;18490  MATCHCRESTART:                            ;
                                    ;18491  EDITRESTART:                              ;
                                    ;18492          CALL,J/FPDUNPACK,                 ;RESET PC + UNPACK R0
                                    ;18493          D_R[R0],Q_R[R0],                  ;
U 0041, 0800,003D,6DC0,FA00,0084,69F2;18494          SC_K[.FFF0]                       ;-16(10)
                                    ;18495
                                    ;18496          ;---------------------------------;
                                    ;18497  141:    ALU_0+K[.1]+1,                    ;GUARANTEE ALU Z,N,C CLEAR
U 0141, 001B,0010,0580,F800,0010,0B1C;18498          CLK.UBCC                          ;
                                    ;18499
                                    ;18500          ;---------------------------------;
                                    ;18501          SC_0,                             ;START STATE TOWARDS HOME
                                    ;18502          FE_K[.FFF0],                      ;-16 WILL BE USED BY EDIT RESTART
U 0B1C, 0001,1B3C,6D80,F800,0186,66E8;18503          BEN/ALU                           ;IR<0>
                                    ;18504
                                    ;18505  =**0*   ;---------------------------------;IR<0>. KNOW ALU CC Z,N,C = 0
                                    ;18506          J/EDITRS1,                        ;EDITPC
                                    ;18507          STATE_SC.VIA.KMX,                 ;RESTORE STATE
U 06E8, 0800,003C,1DC0,FA10,1404,6B1D;18508          D_R[R2],Q_R[R2]                   ;NEXT UNPACK R2
                                    ;18509
                                    ;18510          ;**1*-----------------------------;
                                    ;18511          BEN/IR2-1,STATE_SC.VIA.KMX,       ;IT'S MATCHC OR CRC
U 06EA, 0800,093C,1DC0,FA10,1404,6274;18512          D_R[R2],Q_R[R2]                   ;
                                    ;18513
                                    ;18514  =1*0    ;---------------------------------;IR<1>. IR<2> = 0 FOR BOTH
                                    ;18515          J/MATCHCUNSCRAMBLE,               ;MATCHC(39) IT WAS
                                    ;18516          SC_FE,                            ;A -16
U 0274, 0019,201C,C180,FA90,0081,0A91;18517          R[R2]_Q.ORNOT.K[.FFFF]            ;REMAKE IT A NEGATIVE LWD
                                    ;18518
                                    ;18519          ;1*1------------------------------;
U 0275, 0800,003C,0180,FA00,0000,1010;18520          J/CRCUNSCRAMBLE,D_R[R0]           ;CRC(B) IT WAS
```

```
                                ;18521          ;---------------------------------;
                                ;18522  EDITRS1:                                   ;
                                ;18523          RC[T1]_Q.AND.K[.FF],            ;SAVE FILL CHAR
                                ;18524          SC_FE,                          ;GET THE -16 FOR SHIFTING
U 0B1D, 0B19,2034,4980,F988,0081,0B1E   ;18525          SWAPD                           ;NOW D<7:0>= COUNTER
                                ;18526                                          ;<23:16> = SIGN
                                ;18527                                          ;<31:24> = FILL
                                ;18528
                                ;18529  ; ************************************************
                                ;18530  ; * Patch no. 098, PCS 0B1D trapped to WCS 11A6 *
                                ;18531  ; ************************************************
                                ;18532
                                ;18533  ED.PA.98.A:
                                ;18534          ;---------------------------------;
                                ;18535          D_DAL.SC,                       ;D<7:0>=SIGN BYTE
                                ;18536          Q_D.AND.K[.FF],                 ;RESTORE COUNTER
U 0B1E, 0D19,0034,49C0,FA10,0000,0B22   ;18537          LAB_R[R2]                       ;NEED TO CLEAR R2 <31:16>
                                ;18538
                                ;18539  ED.PA.98:
                                ;18540          ;---------------------------------;
                                ;18541          RC[T2]_D.AND.K[.FF],            ;SIGN RESET
U 0B22, 0C19,0034,4980,F990,0000,0840   ;18542          D_Q                             ;MOVE COUNTER SO PRESERVED
                                ;18543                                          ;ACROSS SETFPD
                                ;18544          ;---------------------------------;
```

```
                                    ;18545  =00                                    ;RETURN3
                                    ;18546          CALL,J/SETFPD,                  ;RESET FPD ADDR
U 0840, 0018,0035,C180,FA90,0000,0E16 ;18547        R[R2]_LA.AND.K[.FFFF]           ;RESET TO FILL + SIGN
                                    ;18548
                                    ;18549          ;-------------------------------;
U 0841, 0000,003C,0180,F800,0000,0B16 ;18550        J/EDITFPD                       ;
                                    ;18551
                                    ;18552          ;-------------------------------;
U 0842, 0000,003C,0180,F800,0000,0B16 ;18553        J/EDITFPD                       ;
                                    ;18554
                                    ;18555          ;-------------------------------;
                                    ;18556          BEN/STATE7-4,                   ;CONSIDER <6:4>
                                    ;18557          STATE_STATE.AND.NOT.K[.8],              ;GUARANTEE STATE<3> CLEAR
                                    ;18558          Q_D,                            ;RESTORE COUNTER TO Q
                                    ;18559          D_R[R5],                        ;DEST ADDR
U 0843, 0800,163C,01E0,FA28,1604,4868 ;18560        VA_R[R5]
                                    ;18561
                                    ;18562  =1000   ;-------------------------------;STATE <6:4>. NOT MOVE OR FLOAT
                                    ;18563          LAB_R[R3],                      ;PATTERN ADDR
                                    ;18564          BEN/STATE3-0,                   ;CONSIDER STATE<2:0>
U 0868, 0000,173C,0180,FA18,0000,0A60 ;18565        J/EDNOTMOVEORFLOAT              ;
                                    ;18566
                                    ;18567          ;1001-----------------------------;MOVE NOT PRE-ZEROING
                                    ;18568          LAB_R[R0],                      ;
                                    ;18569          D_D-K[.1],                      ;IF THIS IS A WRITE, NEED TO
                                    ;18570                                          ;DECREMENT ADDR
                                    ;18571          BEN/STATE3-0,                   ;DETERMINE IF A READ OR WRITE
U 0869, 0819,1700,0580,FA00,0000,0879 ;18572        J/EDMVFLRDORWRITE               ;
                                    ;18573
                                    ;18574          ;1010-----------------------------;FLOAT NOT PRE-ZEROING
                                    ;18575          LAB_R[R0],                      ;
                                    ;18576          D_D-K[.1],                      ;IF THIS IS A WRITE, NEED TO
                                    ;18577                                          ;DECREMENT ADDR
                                    ;18578          BEN/STATE3-0,                   ;
U 086A, 0819,1700,0580,FA00,0000,0879 ;18579        J/EDMVFLRDORWRITE               ;
                                    ;18580
                                    ;18581          ;1011-----------------------------;UNDEFINED
U 086B, 0000,003C,0180,F800,0000,0106 ;18582        J/RSVOPR                        ;
                                    ;18583
                                    ;18584          ;1100-----------------------------;UNDEFINED
U 086C, 0000,003C,0180,F800,0000,0106 ;18585        J/RSVOPR                        ;
                                    ;18586
                                    ;18587          ;1101-----------------------------;MOVE PRE-ZEROING
                                    ;18588          BEN/ROR,                        ;BRANCH ON FILL OR ZERO
                                    ;18589          D_Q,                            ;KEEP COUNT IN D+Q
U 086D, 0C00,023C,0180,F800,0000,0945 ;18590        J/EDINITCHARS                   ;
                                    ;18591
                                    ;18592          ;1110-----------------------------;FLOAT PRE-ZEROING
                                    ;18593          BEN/ROR,                        ;BRANCH ON FILL OR ZERO
                                    ;18594          D_Q,                            ;KEEP COUNT IN D+Q
U 086E, 0C00,023C,0180,F800,0000,0945 ;18595        J/EDINITCHARS                   ;
                                    ;18596
                                    ;18597          ;1111-----------------------------;UNDEFINED
U 086F, 0000,003C,0180,F800,0000,0106 ;18598        J/RSVOPR                        ;
```

```
                                    ;18599  =*000
                                    ;18600  EDNOTMOVEORFLOAT:
                                    ;18601          ;-------------------------------;STATE2-0
                                    ;18602          J/EDITFIRST,                    ;
U 0A60, 0818,0038,7580,F800,0000,0AA9;18603         D_BLANK                         ;BLANK EXPECTED IN D
                                    ;18604
                                    ;18605  EDSTATE1:
                                    ;18606          ;*001---------------------------;
                                    ;18607          J/EDPATT1RST,                   ;STATE = 1
                                    ;18608          Q_0,                            ;GENERAL CLEANUP
                                    ;18609          INTRPT.STROBE,                  ;
U 0A61, 0000,003C,01F8,FA18,4200,0AB6;18610         VA_R[R3]                        ;
                                    ;18611
                                    ;18612  EDSTATE2:
                                    ;18613          ;*010---------------------------;
                                    ;18614          VA_LA-K[.1],                    ;STATE = 2
                                    ;18615          R[R3]_LA-K[.1],                 ;
                                    ;18616          Q_0,                            ;GENERAL CLEANUP
                                    ;18617          INTRPT.STROBE,                  ;
U 0A62, 0018,0000,05F8,FA98,4200,0AB6;18618         J/EDPATT1RST                    ;
                                    ;18619
                                    ;18620  ; ************************************************
                                    ;18621  ; * Patch no. 086, PCS 0A62 trapped to WCS 119F *
                                    ;18622  ; ************************************************
                                    ;18623
                                    ;18624          ;*011---------------------------;
                                    ;18625          D_0,                            ;STATE = 3 = ADJINP.
U 0A63, 0F00,003C,0180,F800,0000,084B;18626         J/EDADJIN2                      ;FORCE RIGHT AT LEFT/RIGHT TEST
                                    ;18627                                          ;I.E. DON'T RE-INCREMENT SRC ADDR
                                    ;18628          ;*100---------------------------;
                                    ;18629          R[R5]_D-K[.1],                  ;STATE = 4
U 0A64, 0019,0000,0580,FAA8,0000,0A61;18630         J/EDSTATE1                      ;
                                    ;18631
                                    ;18632          ;*101---------------------------;
                                    ;18633          D_RC[T2],                       ;STATE = 5 = STORE SIGN + ENDFLOAT
                                    ;18634          Q_0,                            ;CLEAN-UP
U 0A65, 0810,0038,01F8,F910,0000,0ACC;18635         J/EDINSERTST1                   ;RETRY SIGN CHAR
                                    ;18636
                                    ;18637          ;*110---------------------------;
                                    ;18638          R[R5]_D-K[.1],                  ;STATE = 6 = INSERT
U 0A66, 0019,0000,0580,FAA8,0000,0A62;18639         J/EDSTATE2                      ;
                                    ;18640
                                    ;18641          ;*111---------------------------;
U 0A67, 0019,2014,05C0,F800,0000,0AE9;18642         Q_Q+K[.1],J/EDFILLRST           ;RESET COUNTER
```

```
                                        ;18643  =1001   ;-------------------------------;IF MOVE OR FLOAT THEN ONLY
                                        ;18644  EDMVFLRDORWRITE:                        ;STATE <2:1> OF <3:0> CAN BE SET
                                        ;18645                                          ;MOVE OR FLOAT READ
                                        ;18646          SC_K[.FFFC],                    ;PREPARE FOR LEFT NIB CASE
U 0879, 0000,023C,F180,F800,0084,6A76   ;18647          BEN/ROR,J/EDRSTFL1              ;BRANCH ACC TO LEFT/RT NIB
                                        ;18648                                          ;KEEP LA UNALTERED(R0) THRU THIS INSTRUCTION
                                        ;18649          ;1011-------------------------;FLOATSNGL CASE
                                        ;18650  EDMVFLDADDRRESTORE:                     ;
U 087B, 0001,003C,0180,FAA8,0000,0879   ;18651          R[R5]_D,J/EDMVFLRDORWRITE       ;SAVE DECREMENTED DEST ADDR
                                        ;18652
                                        ;18653          ;1101-------------------------;MOVE OR FLOAT WRITE
                                        ;18654  EDMVFLCNTRRESET:                       ;
                                        ;18655          Q_Q+K[.1],                      ;RESET COUNTER
U 087D, 0019,2014,05C0,F800,0000,087B   ;18656          J/EDMVFLDADDRRESTORE            ;
                                        ;18657
                                        ;18658          ;1111-------------------------;FLOATDBL CASE
                                        ;18659          Q_Q+K[.1],                      ;RESET COUNTER
                                        ;18660                                          ;PSL <C> NOW SET, SO SIGN WON'T
U 087F, 0019,2014,05C0,F800,0000,087B   ;18661          J/EDMVFLDADDRRESTORE            ;GET RE-WRITTEN AFTER RE-READ OF CHAR
                                        ;18662
                                        ;18663  =110    ;-------------------------------;LA<0>
                                        ;18664  EDRSTFL1:                               ;
                                        ;18665          J/EDFLOATRIGHTNIB,              ;
U 0A76, 0000,003C,0180,FA08,0200,072C   ;18666          VA_R[R1]                        ;
                                        ;18667
                                        ;18668          ;111--------------------------;LA <0> = 1
                                        ;18669          J/EDFLOATRSTLEFT,               ;
                                        ;18670          VA_R[R1],                       ;
U 0A77, 0000,003C,0180,FA08,4200,06F4   ;18671          INTRPT.STROBE                   ;
                                        ;18672
                                        ;18673  .LIST           ;Re-enable full listing
```

```
                                        ;18674  .TOC    "DECMAL.MIC"
                                        ;18675  .TOC    "Revision 2.9"
                                        ;18676  ;        R. J. Avarbock, P. R. Guilbault
                                        ;18677
;18678  .NOBIN
;18679  .TOC    "       Revision History"
;18680
;18681  ; 02    Remove absolute jumps.
;18682  ;       Add Patch no. 093 to fix (ADD,SUB)P(4,6) V bit problem.
;18683  ;       Add Patch no. 092 to fix CVTPT V bit problem.
;18684  ;       Add Patch no. 090 to fix CVTTP int. problem.
;18685  ;       Add Patch no. 089 to fix (ADD,SUB)P(4,6) V bit bug.
;18686  ;       Change macro names that deal with conditions codes.
;18687  ; 01    Add Patch no. 074 to fix MULP bug.
;18688  ; 00    Create this file by merging MOVP.MIC, CONV.MIC, ADDP.MIC, MULP.MIC, DIVP.MIC, ASHP.MIC, and SUB.MIC
;18689  ;       Start of history
;18690
                                        ;18691  .BIN
                                        ;18692  .NOLIST          ;Disable listing of PCS code for quickie assemblies
```

```
;18693  .TOC     ""      Decimal string        : MOVP""
;18694
;18695  ;ALGORITHM:
;18696  ;       1. STARTING AT 'MOVP.INIT', THE LAST SPECIFIER IS EVALUATED,
;18697  ;       AND THE REGISTERS ARE INITIALIZED. FIRST PART DONE-FLAG
;18698  ;       IS SET ('BCD.FPD').
;18699
;18700  ;       2. USING THE 'READ-BCD'-SUBROUTINE, DATA IS READ FROM
;18701  ;       SOURCE-STRING, A LONGWORD AT THE TIME ('MVP.0').
;18702
;18703  ;       3. USING THE 'WRITE-BCD'-SUBROUTINE, THE DATA IS WRITTEN
;18704  ;       INTO THE DST-STRING ('MVP.1'), AND THE REGISTERS ARE UPDATED.
;18705
;18706  ;       4. WHEN THE LENGTH REACHES ZERO, (REMEMBER THAT
;18707  ;       THE TWO STRINGS HAVE SAME LENGTH), THE CONDITION CODES ARE SET,
;18708  ;       AND THE GENERAL REGISTERS ARE LOADED ('FINIO').
;18709  ;       THIS ROUTINE IS MORE EXTENSIVE THAN MIGHT BE EXPECTED,
;18710  ;       SINCE IT IS USED BY ALMOST ALL THE PACKED DECIMAL INSTRUCTIONS
;18711  ;       TO SET CONDITION CODES.
;18712
;18713  ;STORAGE:
;18714  ;       R0 CONTAINS NEGATIVE SRC-LENGTH
;18715  ;       R1 CONTAINS SRC-ADDRESS
;18716  ;       R3 CONTAINS DST-ADDRESS
;18717
;18718  ;       SAVE LENGTH IN RC1
;18719  ;       SAVE SRC-ADDRESS IN ID[T0]
;18720  ;       SAVE DST-ADDRESS IN ID[T1]
;18721
;18722  ;       INSTRUCTION DEPENDENT ALU-FUNCTION IS "A-1"
;18723  ;       INSTRUCTION DEPENDENT CLOCKING OF PSL CONDITION CODES IS:
;18724  ;               Z_ALU=0L[DT], N_ALU SIGN[UDT], V_0, C_C
;18725
;18726  ;STATE-REGISTER:
;18727          ;-------------------------------------------------------------;
;18728          ;INTRPT ;       ;       ;       ;       ;1.TIME ;SIGN   ;       ;
;18729          ;       ;       ;       ;       ;       ;WRITE  ;       ;       ;
;18730          ;       ;       ;       ;       ;       ;       ;       ;       ;
;18731          ;       ;       ;       ;       ;       ;       ;       ;       ;
;18732          ;-------------------------------------------------------------;
```

```
                                    ;18733          ;ENTER HERE FROM D-FORK WITH LENGTH IN Q, SRC-ADDRESS IN D
                                    ;18734  484:    ;-------------------------------;
                                    ;18735  MOVP.INIT:
                                    ;18736          ALU_Q.OXT[WORD],                ;  ISOLATE SRC-LENGTH
                                    ;18737          RC[T1]_ALU,                     ;  SAVE LENGTH
U 0484, 0803,603C,C180,3D88,0000,0290 ;18738        ID[T0]_D,D_ALU                  ;  SAVE SRC-ADDRESS
                                    ;18739          ;-------------------------------;
                                    ;18740  =00*****
                                    ;18741          SC_K[.30],                      ;  ID-ADDRESS FOR LATER USE
                                    ;18742          ALU_NOT.D,R[R15]_ALU,           ;  SAVE NEGATIVE LENGTH
                                    ;18743          DK/RIGHT,                       ;  DIVIDE  LENGTH BY 2
U 0290, 06C1,0029,7980,FAF8,0084,647E ;18744        CALL,J/ASPC                     ;  EVALUATE DST-ADDRESS
                                    ;18745          ;-------------------------------;
                                    ;18746  =11*****                               ;  REENTER HERE AFTER A FAULT
                                    ;18747  MVP.I0: STATE_FE,                       ;  USE FE TO CLEAR STATE
                                    ;18748          ALU_D+Q+1,R[R3]_ALU,            ;  GENERATE DST ADDRESS
U 02F0, 001D,0010,C580,3E98,1400,6495 ;18749        ID[T1]_D,J/MVP.I1               ;  SAVE INITIAL ADDRESS IN T1
                                    ;18750  =;END   ;-------------------------------;
                                    ;18751  =C***
                                    ;18752  MVP.I1: STATE_STATE-FE,                 ;  CLEAR STATE-REGISTER
                                    ;18753          ALU_Q.AND.K[.FFF0],             ;  TEST ILLEGAL BITS OF LENGTH
                                    ;18754          SET.CC(LONG),                   ;  CLOCK Z-BIT
                                    ;18755          LAB_R[R15],                     ;  GET LENGTH
                                    ;18756          Q_ID(SC),D_Q,                   ;  GET SRC-ADDRESS
U 0495, 0C19,2035,6DF0,2678,1470,AB24 ;18757        CALL,J/BCD.FPD.00               ;  SET 1. PART DONE
                                    ;18758          ;-------------------------------;
                                    ;18759  =1***   ID[FPDA]_D,                     ;  LOAD RETURN-ADDRESS FOR FPD
                                    ;18760          ALU_LA,Q_ALU.RIGHT,             ;  GET LENGTH
U 049D, 0040,803C,B5C0,3C00,0010,0850 ;18761        CLK.UBCC,BYTE
                                    ;18762  =;END   ;-------------------------------;
```

```
                                ;18763  =00
                                ;18764          ;00--------------------------------;
                                ;18765  MVP.0:  LA_RA[R1],                      ;  GET SRC-ADDRESS
                                ;18766          ALU_Q+K[.3],D_ALU,              ;  UPDATE LENGTH
                                ;18767          CLK.UBCC,BYTE,
U 0850, 0819,BB15,0D80,F888,0010,0AF7   ;18768          ALU?,CALL,J/READ0               ;  CALL BCD-READ-SUBROUTINE
                                ;18769          ;--------------------------------;
                                ;18770  FINI0:  ;START OF ROUTINE TO RESET REGISTERS AND SET PSL-CC
                                ;18771          ;01--------------------------------;
                                ;18772          N_AMX.Z_TST,                    ;  CLEAR N-BIT
                                ;18773          ALU_0(A),R[R0]_ALU,             ;  CLEAR R0
                                ;18774          Q_ID[T0],STATE3-0?,             ;  GET SRC-ADDRESS
U 0851, 0003,173C,C1F0,2E80,0030,088D   ;18775          J7FINI1
                                ;18776          ;11--------------------------------;
                                ;18777  =11     R[R1]_LA-K[.4],                 ;  UPDATE SRC-ADDRESS
U 0853, 0018,1700,1180,FA88,0000,00DA   ;18778          STATE3-0?                       ;  TEST FOR 1. TIME THROUGH
                                ;18779  =;END   ;--------------------------------;
                                ;18780  =101*   ;BRANCH ON 1. WRITE BIT
                                ;18781          ;101*------------------------------;
                                ;18782          D_D.ANDNOT.K[.F],               ;  STRIP OFF SIGN-NIBBLE
U 00DA, 0819,0F24,6180,F800,0000,085A   ;18783          BCDSGN?,J/MVP.FIRST             ;  TEST DECIMAL SIGN
                                ;18784          ;111*------------------------------;
                                ;18785          LA_RA[R3],                      ;  GET DST-ADDRESS
U 00DE, 000C,0038,01C0,F898,0010,0058   ;18786          Q_LB,CLK.UBCC                   ;  CLOCK LENGTH
                                ;18787  =;END   ;--------------------------------;
                                ;18788  =10*****
                                ;18789  MVP.1:  INTRPT.STROBE,                  ;  STROBE FOR LATER TESTING
U 0058, 0019,2015,0180,FA80,4000,0C6B   ;18790          R[R0]_Q+K[.8],CALL,J/WRITE      ;  UPDATE LENGTH, WRITE DATA
                                ;18791          ;--------------------------------;
                                ;18792  =11*****
                                ;18793          R[R3]_LA-K[.4],                 ;  UPDATE DST-ADDRESS
                                ;18794          STATE_STATE.OR.K[.4],           ;  SET 1. TIME BIT OF STATE
U 0078, 0018,0E00,1180,FA98,1404,2A86   ;18795          BEN/INTERRUPT                   ;  TEST FOR PENDING INTERRUPTS
                                ;18796  =;END   ;--------------------------------;
                                ;18797  =110    ;BRANCH ON INTERRUPT REQUEST
                                ;18798          ;110-------------------------------;
                                ;18799          ALU_R[R0],                      ;  GET LENGTH
                                ;18800          Q_ALU.RIGHT,                    ;  DIVIDE IT BY 2
U 0A86, 0040,803C,01C0,FA00,0010,0850   ;18801          C[K.UBCC,BYTE,J/MVP.0           ;  LOOP BACK TO READ NEXT LONGWORD
                                ;18802          ;111-------------------------------;
                                ;18803          STATE_K[.80],                   ;  SET INTERRUPT-BIT OF STATE
U 0A87, 0000,003C,4180,F800,1404,6033   ;18804          J/SAVE.BCD                      ;  SAVE CONTEXT AND TAKE INTERRRUPT
                                ;18805  =;END   ;--------------------------------;
```

```
                                ;18806          ;ENTER HERE AFTER READING FIRST LONGWORD OF SRC-STRING
                                ;18807          ;-------------------------------;
                                ;18808  =10      ;BRANCH ON BCDSGN
                                ;18809          ;10-------------------------------;
                                ;18810  MVP.FIRST:
                                ;18811          LA_RA[R3],Q_LB,CLK.UBCC,        ;  GET DST-ADDRESS, CLOCK LENGTH
U 085A, 000C,0038,01C0,F898,0010,0058   ;18812          J/MVP.1
                                ;18813          ;11-------------------------------;
                                ;18814          STATE_STATE.OR.K[.2],           ;  SET SIGN-BIT
                                ;18815          LA_RA[R3],Q_LB,CLK.UBCC,        ;  GET DST-ADDRESS, CLOCK LENGTH
U 085B, 000C,0038,09C0,F898,1414,2058   ;18816          J/MVP.1
                                ;18817  =;END   ;-------------------------------;
                                ;18818
                                ;18819
                                ;18820          ;ROUTINE TO SET FIRST PART DONE FLAG'' AND GENERATE THE
                                ;18821          ;ADDRESS 33 FOR ID[FPDA], AS WELL AS TEST THE
                                ;18822          ;PSL Z-BIT TO CHECK FOR ILLEGAL LENGTHS.
                                ;18823          ;THIS ROUTINE IS USED BY MOST DECIMAL STRING INSTRUCTIONS.
                                ;18824  BCD.FPD.00:
                                ;18825          ALU_D+Q+1,R[R1]_ALU,            ;  LOAD HIGH SRC-ADDRESS
U 0B24, 001D,1A10,0180,FA88,0000,00F9   ;18826          PSL.CC?,J/BCD.FPD               ;  TEST LENGTHS
                                ;18827          ;-------------------------------;
                                ;18828  BCD.FPD.0:
                                ;18829          ALU_D+Q+1,R[R1]_ALU,            ;  LOAD HIGH SRC-ADDRESS
U 0B25, 001D,1A10,C180,3E88,0000,00F9   ;18830          ID[T0]_D,PSL.CC?                ;  SAVE LOW ADDRESS, TEST LENGTHS
                                ;18831          ;-------------------------------;
                                ;18832  =10*1   ;BRANCH ON PSL Z-BIT, (V-BIT=0)
                                ;18833          ;10*1-----------------------------;
                                ;18834  BCD.FPD:
U 00F9, 0000,003C,0180,F800,0000,0106   ;18835          J/RSVOPR                        ;  ILLEGAL LENGTH
                                ;18836          ;11*1-----------------------------;
                                ;18837          ALU_K[.19],D_ALU.LEFT,SI/MUL~,  ;  GENERATE ADDRESS '33''
U 00FD, 0838,003A,BB80,F800,2400,0008   ;18838          SET.FPD,RETURN8                 ;  SET FPD-BIT OF PSL
                                ;18839  =;END   ;-------------------------------;
```

```
                                ;18840          ;ROUTINE TO SET PSL CONDITION CODES, AND RESET GENERAL REGISTERS
                                ;18841          ;EXPECTS R0 TO HAVE BEEN RESET, Q=SRC-ADDRESS
                                ;18842          ;THIS ROUTINE IS USED BY MOST DECIMAL STRING INSTRUCTIONS.
                                ;18843
                                ;18844  =1101    ;BRANCH ON SIGN-BIT OF STATE
                                ;18845          ;1101----------------------------;
                                ;18846  FINI1:  R[R1]_Q,                        ;  R1 GETS SRC-ADDRESS
                                ;18847          Q_ID[T1],STATE7-4?,             ;  TEST FOR OVERFLOW
U 088D, 0001,363C,C5F0,2E88,0000,0A93   ;18848          J7FINI3
                                ;18849          ;1111----------------------------;
                                ;18850  FINI2:  R[R1]_Q,                        ;  R1 GETS SRC-ADDRESS
                                ;18851          Q_ID[T1],                       ;  GET DST-ADDRESS
U 088F, 0001,3A3C,C5F0,2E83,0000,089B   ;18852          PSL.CC?,J/FINI16                ;  TEST PSL Z-BIT
                                ;18853  =;END    ;------------------------------;
                                ;18854  =011    ;BRANCH ON OVERFLOW-BIT OF STATE
                                ;18855          ;011-----------------------------;
U 0A93, 0001,203C,0180,FA98,0000,0B29   ;18856  FINI3:  R[R3]_Q,J/FINI8                  ;  RESET R3
                                ;18857          ;111-----------------------------;
                                ;18858  FINI4:  R[R3]_Q,                        ;  R3 GETS DST-ADDRESS
U 0A97, 0001,203C,31F0,2E98,0020,0B26   ;18859          Q_ID[CES],SET.V                 ;  OVERFLOW, SET V-BIT OF PSL
                                ;18860  =;END    ;------------------------------;
                                ;18861  FINI5:  SET.V,                          ;  SET V-BIT ON OVERFLOW
U 0B26, 0819,2030,A580,F800,0020,0B28   ;18862          ALU_Q.OR.K[.60],D_ALU           ;  GENERATE TRAP-VALUE
                                ;18863          ;------------------------------;
                                ;18864  FINI6:  ID[CES]_D,                      ;  LOAD TRAP-CODE
                                ;18865          ALU_0(A),R[R2]_ALU,             ;  CLEAR R2
U 0B28, 0003,003C,3180,3E90,2000,0B2C   ;18866          CLR.FPD,J/FINI15                ;  RESET FPD-BIT IN PSL
                                ;18867          ;------------------------------;
                                ;18868  FINI8:  ALU_0(A),R[R2]_ALU,             ;  CLEAR R2
U 0B29, 0003,003C,0180,FA90,2000,0B2C   ;18869          CLR.FPD,J/FINI15                ;  RESET FPD-BIT OF PSL
                                ;18870          ;------------------------------;
                                ;18871  FINI15: FLUSH.IB,PC&VA_PC,               ;  GET READY FOR NEXT INSTRUCTION
U 0B2C, 2014,0038,0180,F801,4200,00AB   ;18872          J/IB.FILL
                                ;18873          ;------------------------------;
```

```
                                ;18874          ;ENTER HERE IF SIGN-BIT OF STATE IS SET
                                ;18875  =1011   ;BRANCH ON PSL Z-BIT
                                ;18876          ;1011-----------------------------;
                                ;18877  FINI16: ALU_K[.80],N&Z_ALU,BYTE,          ;  SET N-BIT
U 089B, 0018,9638,4180,F800,0060,0A93   ;18878          STATE7-4?,J/FINI3          ;  TEST FOR OVERFLOW
                                ;18879          ;1111-----------------------------;
                                ;18880  FINI17: ALU_R[T1],D_ALU.RIGHT,          ;  RESULT IS -0, SO GET DST-LENGTH
U 089F, 0850,1638,0180,F908,0000,0AA3   ;18881          STATE7-4?                  ;  AND TEST FOR OVERFLOW
                                ;18882  =;END   ;-----------------------------;
                                ;18883  =011    ;BRANCH ON OVERFLOW-BIT
                                ;18884          ;011----------------------------;
                                ;18885  FINI18: ALU_D+Q,VAK/LOAD,                ;  NO OVERFLOW, LOAD DST-ADDRESS
U 0AA3, 001D,1614,2180,F800,0200,0872   ;18886          K[.14],STATE7-4?,J/FINI20  ;  TEST FOR PACKED-TO-NEMERIC CONV.
                                ;18887          ;111----------------------------;
                                ;18888  FINI19: R[R3]_Q,                         ;  R3 GETS DST-ADDRESS
                                ;18889          Q_ID[ES],SET.V,                 ;  SET V-BIT
U 0AA7, 0001,203C,31F0,2E98,0020,0B26   ;18890          J/FINI5
                                ;18891  =;END   ;-----------------------------;
                                ;18892  =10     ;BRANCH ON BIT 4 OF STATE
                                ;18893          ;10-----------------------------;
U 0872, 0818,0038,8580,F800,0000,0B2D   ;18894  FINI20: D_K[.C],J/FINI21           ;  D GETS +0
                                ;18895          ;T1-----------------------------;
                                ;18896          ALU_0+K[.14]+1,D_ALU.LEFT,      ;  PACKED-TO-NUMERIC CONVERSION
U 0873, 083B,0910,2380,F800,0000,06EC   ;18897          SI/MUL-,IR2-1?             ;  GENERATE CONSTANT .2B
                                ;18898  =;END   ;-----------------------------;
                                ;18899  =0*     ;BRANCH ON BIT 2 OF OP-CODE
                                ;18900          ;0*-----------------------------;
U 06EC, 0001,203C,0180,F800,0200,0B2D   ;18901          VA_Q,J/FINI21              ;  PACKED-TO-SEPARATE
                                ;18902          ;1*-----------------------------;
U 06EE, 0001,203C,0180,FA98,0000,0B29   ;18903          R[R3]_Q,J/FINI8            ;  PACKED TO TRAILING, DON'T CHANGE
                                ;18904  =;END   ;-----------------------------;
U 0B2D, 0000,803C,0180,3000,0000,0A93   ;18905  FINI21: CACHE_D[BYTE],J/FINI3      ;  WRITE +0
                                ;18906          ;-----------------------------;
```

```
;18907  .TOC    ""      Decimal string        : CMPP3, CMPP4""
;18908
;18909  ;COMPARE PACKED BCD-STRINGS
;18910  ;ALGORITHM:
;18911  ;       1. FIRST THE LAST SPECIFIERS ARE EVALUATED, THE LENGTHS ARE CHECKED,
;18912  ;       AND FIRST PART DONE FLAG IS SET.
;18913  ;       THE TWO INSTRUCTIONS, CMPP3 AND CMPP4, HAVE INSTRUCTION-FLOWS
;18914  ;       THAT MERGE AT ''CMP.I''.
;18915
;18916  ;       2. THE LENGTHS OF THE TWO DECIMAL STRINGS ARE COMPARED (''CMP40'').
;18917  ;       IF THEY ARE NOT EQUAL, THE ROUTINE READS LEADING BYTES OF
;18918  ;       THE LONGEST STRING (''CMP41'' OR ''CMP42''), UNTIL THEY BECOME
;18919  ;       EQUAL, OR A NON-ZERO BYTE IS FOUND.
;18920
;18921  ;       3. IF THE LENGTHS ARE EQUAL, WE ENTER THE MAIN LOOP (''CMP3LP'').
;18922  ;       THE LEADING BYTE OF EACH STRING IS READ AND COMPARED.
;18923  ;       IF THEY ARE EQUAL, THE NEXT PAIR OF BYTES ARE READ,
;18924  ;       AND SO ON.
;18925
;18926  ;       4.IF ONE STRING CONTAINS A LARGER DIGIT, THE SIGN-NIBBLE OF THAT
;18927  ;       STRING IS THEN READ AND USED TO DETERMINE THE RESULT OF
;18928  ;       THE COMPARISON.
;18929
;18930  ;STORAGE:
;18931  ;       SC=# OF BYTES IN STRING 1,FE=# BYTES IN STRING 2,
;18932  ;       R15=SRC1-ADDRESS-1,R3=SRC2-ADDRESS-1
;18933  ;       RC0=1.LENGTH/2
;18934  ;       R0=2.LENGTH/2
;18935  ;       ID[T0]=SRC1-ADDRESS,ID[T1]=SRC2-ADDRESS
;18936
;18937  ;       INSTRUCTION DEPENDENT ALU-FUNCTION IS ''A-1''.
;18938  ;       MNEMONICS ARE CMPP3 AND CMPP4.
;18939  ;       OP-CODES ARE 35 AND 37
;18940
;18941  ;STATE-REGISTER:
;18942          ;-------------------------------------------------------------
;18943          ;INTRPT ;       ;       ;       ;NON-0  ;       ;0      ;       ;
;18944          ;       ;       ;       ;       ;STRING ;       ;       ;       ;
;18945          ;       ;       ;       ;       ;       ;       ;       ;       ;
;18946          ;       ;       ;       ;       ;       ;       ;       ;       ;
;18947          ;-------------------------------------------------------------
```

```
                                    ;18948          ;ENTER HERE FROM D-FORK WITH L IN Q,A1 IN D, IN CMPP3-INSTRUCTION
                                    ;18949          ;-------------------------------;
                                    ;18950  489:    ALU_0(A),N&Z_ALU.V&C_0,       ;  SET PSL Z-BIT
U 0489, 0C03,003C,C1B0,3C00,0050,0B2E ;18951        Q/K/RIGHT,ID[T0]_D,D_Q        ;  DIVIDE LENGTH BY 2, STORE ADDRESS
                                    ;18952          ;-------------------------------;
                                    ;18953          ALU_Q.OXT[BYTE],SC_ALU,       ;  STORE LENGTH IN SC, DUPLICATE IT IN D
                                    ;18954          Q_ID[T0],                     ;  RETRIEVE ADDRESS
U 0B2E, 0003,A03C,C1F0,2D80,0082,00B3 ;18955        R[T0]_ALU,J/CMP.I             ;  SAVELENGTH IN RC 0, JOIN CMPP4
                                    ;18956          ;-------------------------------;
                                    ;18957
                                    ;18958          ;ENTER HERE FROM D-FORK WITH L1 IN Q,A1 IN D, IN CMPP4-INSTRUCTION
                                    ;18959          ;-------------------------------;
                                    ;18960  485:    ALU_Q.ANDNOT.K[.1F],          ;  MASK OUT 5 LOW BITS
U 0485, 0019,6024,8DB0,F800,0050,00A3 ;18961        N&Z_ALU.V&C_0,WORD,Q/K/RIGHT  ;  CLOCK PSL Z-BIT, DIVIDE LENGTH BY 2
                                    ;18962          ;-------------------------------;
                                    ;18963  =C10**1*
                                    ;18964          ALU_Q.OXT[BYTE],              ;  ISOLATE 1. LENGTH
                                    ;18965          SC_ALU,R[T0]_ALU,             ;  STORE FIRST LENGTH IN SC
                                    ;18966          ID[T0]_D,                     ;  SAVE ADDRESS IN T0
U 00A3, 0003,A03D,C180,3D80,0082,037E ;18967        CALL,J/SPEC                   ;  EVALUATE L2
                                    ;18968          ;-------------------------------;
                                    ;18969  =011**1*
                                    ;18970  CMP.I:  ALU_Q-K[.1],R[R15]_ALU,LONG,  ;  STORE HIGH SRC-ADDRESS
U 00B3, 0019,2001,0580,FAF8,0180,C47E ;18971        SC_SC+1,FEK/LOAD,CALL,J/ASPC  ;  INCREMENT SRC1-LENGTH
                                    ;18972          ;-------------------------------;
                                    ;18973  =111**1*
                                    ;18974          ALU_Q.ANDNOT.K[.1F],          ;  STRIP OFF LOW 5 BITS
                                    ;18975          N_AMX,Z_TST,WORD,Q/K/RIGHT,   ;  DIVIDE LENTGTH BY 2
U 00F3, 0019,6024,8D90,F800,0030,0B34 ;18976        J/CMP4IT
                                    ;18977  =;END   ;-------------------------------;
                                    ;18978  CMP4I1: ALU_D[INST.DEP]Q,             ;  SUBTRACT 1 FROM D
U 0B34, 001D,000C,C580,3E98,0000,05C1 ;18979        R[R3]_ALU,ID[T1]_D            ;  INITIALIZE SRC2-ADDRESS
                                    ;18980  =0***   ;0***-------------------------;
                                    ;18981          ALU_Q.OXT[BYTE]+K[.1],SC_ALU, ;  INITIALIZE SRC2-LENGTH
                                    ;18982          R[T1]_ALU,                    ;  SAVE SRC2-LENGTH
                                    ;18983          PSL.CC?,                      ;  TEST FOR LEGAL LENGTHS
U 05C1, 001B,BA15,0580,F988,0082,00F9 ;18984        CALL,J/BCD.FPD
                                    ;18985          ;1***-------------------------;
                                    ;18986          ID[FPDA]_D,                   ;  LOAD 33
U 05C9, 0000,003C,B580,3C00,0090,EB35 ;18987        SC_NABS(SC-FE),CLK.UBCC       ;  GET DIFFERENCE IN LENGTHS
                                    ;18988  =;END   ;-------------------------------;
                                    ;18989          STATE_K[ZERO],                ;  INITIALIZE STATE-REG.
                                    ;18990          D_0,LAB_R[R15],               ;  GET 1. ADDRESS
U 0B35, 0F00,123C,1980,FA78,1404,68A3 ;18991        EALU?,J/CMP40                 ;  COMPARE LENGTHS
                                    ;18992          ;-------------------------------;
```

```
                                ;18993  =0011    ;BRANCH ON EALU Z AND N-BITS
                                ;18994           ;0011----------------------------;
                                ;18995  CMP40:   SC_SC-K[.1],                     ;  ADJUST 2. LENGTH
                                ;18996           LAB_R[R3],LONG,                  ;  ADDRESS OF 2. STRING
U 08A3, 0F00,003C,0580,FA18,0084,A8DB   ;18997           D_0,J/CMP420             ;  2. STRING IS LONGER
                                ;18998           ;0111----------------------------;
                                ;18999           FE_SC-K[.1],SC_FE,
                                ;19000           VA_LA+K[.1],R[R15]_ALU,LONG,     ;  LOAD 1. SRC-ADDRESS
                                ;19001           D_0,Q_0,
                                ;19002           INTRPT.STROBE,                   ;  STROBE INTERRUPTS
U 08A7, 0F18,0014,05F8,FAF8,4385,A8EE   ;19003           J/CMP3L10                ;  SAME LENGTH, ENTER MAIN LOOP
                                ;19004           ;1011----------------------------;
                                ;19005           VA_LA+K[.1],R[R15]_ALU,LONG,     ;  LOAD 1. SRC-ADDRESS
U 08AB, 0F18,0014,0580,FAF8,0300,88AE   ;19006           FE_SC+FE,D_0,J/CMP41     ;  1. STRING IS LONGER
                                ;19007  =;END    ;----------------------------;
                                ;19008  =1110    ;BRANCH ON LOW BYTE OF D .NE. 0
                                ;19009           ;1110----------------------------;
                                ;19010  CMP41:   D[BYTE]_CACHE,                   ;  READ LEADING BYTE OF STRING 1
                                ;19011           SC_SC+1,CLK.UBCC,                ;  INCREMENT DIFFERENCE IN LENGTHS
                                ;19012           LAB_R[R15],                      ;  GET 1. ADDRESS
U 08AE, 0000,923C,0180,4278,0090,C8BB   ;19013           EALU?,J/CMP410           ;  TEST DIFFERNCE
                                ;19014           ;1111----------------------------;
                                ;19015           Q_ID[TO],LC_RC[TO],              ;  GET SRC-ADDRESS AND LENGTH
U 08AF, 0000,003C,C1F0,2D00,0000,0B3A   ;19016           J/CMPSGN1                ;  NON-ZERO BYTE
                                ;19017  =;END    ;----------------------------;
                                ;19018  =1011    ;BRANCH ON EALU Z-BIT
                                ;19019           ;1011----------------------------;
                                ;19020  CMP410:  VA_LA+K[.1],R[R15]_ALU,LONG,     ;  LOAD AND UPDATE 1. ADDRESS
U 08BB, 0018,1814,0580,FAF8,0200,08AE   ;19021           D.B0?,J/CMP41            ;  TEST BYTE
                                ;19022           ;1111----------------------------;
                                ;19023           ALU_D.OXT[LONG]-K[ZERO],         ;  CLOCK BYTE JUST READ
                                ;19024           CLK.UBCC,
                                ;19025           SC_FE,
U 08BF, 001B,0000,1980,FA18,0091,0B36   ;19026           LAB_R[R3],LONG,J/CMP3L2  ;  STRINGS ARE OF EQUAL LENGTH
                                ;19027  =;END    ;----------------------------;
```

```
                                ;19028  =1110   ;BRANCH ON LOW BYTE OF D
                                ;19029          ;1110----------------------------;
                                ;19030  CMP42:  D[BYTE]_CACHE,                   ;  READ BYTE FROM STRING 2
                                ;19031          LAB_R[R3],Q_0,                  ;  GET ADDRESS READY
                                ;19032          SC_FE,FE_SC,                    ;
U 08CE, 0000,923C,01F8,4218,0181,08DB   ;19033          EA[U?,J/CMP420                  ;  TEST DIFFERENCE IN LENGTHS
                                ;19034          ;1111----------------------------;
U 08CF, 0000,003C,0180,F800,0080,AB38   ;19035          SC_SC-FE                        ;  NON-ZERO BYTE
                                ;19036  =;END   ;------------------------------;
                                ;19037          VA_LA+K[SC],                    ;  LOAD ADDRESS OF SIGN-BYTE
U 0B38, 0018,0014,1D80,F800,0200,036A   ;19038          J/CMPSGN21                      ;  CHECK SIGN-BYTE
                                ;19039          ;------------------------------;
                                ;19040  =1011   ;BRANCH ON EALU Z-BIT
                                ;19041          ;1011----------------------------;
                                ;19042  CMP420: VA_LA+K[.1],R[R3]_ALU,LONG,     ;  LOAD AND UPDATE 2. ADDRESS
                                ;19043          FE_SC+1,CLK.UBCC,SC_FE,         ;  INCREMENT COUNT
U 08DB, 0018,1814,0580,FA98,0391,C8CE   ;19044          D.B0?,J/CMP42                   ;  TEST BYTE FOR 0
                                ;19045          ;1111----------------------------;
                                ;19046          R[R3]_LA-K[.1],SC_FE,           ;  UPDATE 2. ADDRESS
U 08DF, 0018,0000,0580,FA98,0081,08FF   ;19047          J/CMP3L4                        ;  STRINGS ARE OF EQUAL LENGTH
                                ;19048
                                ;19049  ; ************************************************
                                ;19050  ; * Patch no. 063, PCS 08DF trapped to WCS 1189 *
                                ;19051  ; ************************************************
                                ;19052
                                ;19053  =;END
```

```
                                ;19054          ;COMPARE TWO DECIMAL STRINGS WITH SAME LENGTH
                                ;19055          ;R15 HAS ADDRESS FOR STRING 1
                                ;19056          ;R2 HAS ADDRESS FOR STRING 2
                                ;19057          ;SC HAS LENGTH
                                ;19058
                                ;19059          ;---------------------------------;
                                ;19060  =101    ;BRANCH ON SC .GT. 0
                                ;19061          ;---------------------------------;
                                ;19062  CMP3LP: ;MAIN LOOP FOR COMPARING TWO STRINGS OF EQUAL LENGTH
                                ;19063          ;101-------------------------------;
                                ;19064          ALU_D.OXT[BYTE].ANDNOT.K[.F],     ;  STRIP OFF SIGN-NIBBLE
                                ;19065          SC_ALU,D_Q,Q_D,
U 0AE5, 0C1B,8024,61E0,F800,0082,0B45   ;19066          J/CP3SB                           ;  THIS IS SIGN-BYTE
                                ;19067          ;111-------------------------------;
                                ;19068  CMP3L1: VA_LA+K[.1],R[R15]_ALU,           ;  UPDATE AND LOAD ADDRESS 1
U 0AE7, 0018,1814,0580,FAF8,4200,08EE   ;19069          INTRPT.STROBE,D.B0?               ;  TEST FOR 0 STRING
                                ;19070  =;END   ;---------------------------------;
                                ;19071  =1110   ;BRANCH ON LOW BYTE OF D NE 0
                                ;19072          ;1110------------------------------;
                                ;19073  CMP3L10:
                                ;19074          ALU_D.OXT[BYTE]-Q,CLK.UBCC,       ;  COMPARE THE TWO BYTES
                                ;19075          D[BYTE]_CACHE,LAB_R[R3],          ;  READ NEXT BYTE
U 08EE, 001F,8E00,0180,4218,0010,0B36   ;19076          BEN/INTERRUPT,J/CMP3L2            ;  TEST FOR INTERRUPT
                                ;19077          ;1111------------------------------;
                                ;19078          ALU_D.OXT[BYTE]-Q,CLK.UBCC,       ;  COMPARE THE TWO BYTES
                                ;19079          D[BYTE]_CACHE,LAB_R[R3],          ;  READ NEXT BYTE
                                ;19080          STATE_STATE.OR.K[.8],             ;  SET NON-ZERO BIT OF STATE
U 08EF, 001F,8E00,0180,4218,1414,2B36   ;19081          BEN/INTERRUPT,J/CMP3L2            ;  TEST FOR INTERRUPT
                                ;19082  =;END   ;---------------------------------;
                                ;19083  =110    ;BRANCH ON INTERRUPT REQUEST
                                ;19084          ;110-------------------------------;
                                ;19085  CMP3L2: VA_LA+K[.1],R[R3]_ALU,LONG,       ;  LOAD AND UPDATE ADDRESS 2
                                ;19086          SC_SC-K[.1],FEK/LOAD,             ;  UPDATE COUNT
U 0B36, 0018,1B14,0580,FA98,0384,A8FA   ;19087          ALU?,J/CMP3L3                     ;  COMPARE THE BYTES
                                ;19088          ;111-------------------------------;
                                ;19089          STATE_K[.80],                     ;  SET INTERRUPT-BIT OF STATE
U 0B37, 0000,003C,4180,F800,1404,6033   ;19090          J/SAVE.BCD                        ;  SAVE CONTEXT AND TAKE INTERRRUPT
                                ;19091  =;END   ;---------------------------------;
                                ;19092  =1010   ;BRANCH ON ALU Z AND C-BITS
                                ;19093          ;1010------------------------------;
                                ;19094  CMP3L3: Q_ID[TO],LC_RC[TO],               ;  GET SRC1-LENGTH AND ADDRESS
U 08FA, 0000,003C,C1F0,2D00,0000,0B3A   ;19095          J/CMPSGN1                         ;  STRING1 > STRING2, CHECK SIGN
                                ;19096          ;1011------------------------------;
                                ;19097          VA_LA+K[SC]+1,                    ;  LOAD SRC2 SIGN-ADDRESS
U 08FB, 0018,0010,1D80,F800,0200,036A   ;19098          J/CMPSGN21                        ;  STRING2 > STRING 1, CHECK SIGN
                                ;19099  =1111   ;1111------------------------------;
                                ;19100  CMP3L4: Q_D.OXT[BYTE],                    ;  SAVE 1. BYTE IN Q
                                ;19101          D[BYTE]_CACHE,                    ;  READ 2. BYTE
                                ;19102          LAB_R[RT5],                       ;  GET 1. ADDRESS
U 08FF, 0003,943C,01C0,4278,0000,0AE5   ;19103          SC?,J/CMP3LP                      ;  TEST LENGTH, LOOP BACK
                                ;19104  =;END   ;---------------------------------;
```

```
                                    ;19105  CMPSGN1:        ;COME HERE WHEN STRING 1 IS  GT. STRING 2
                                    ;19106          ;LC HAS SRC1-LENGTH, Q HAS SRC1-ADDRESS
                                    ;19107          ;-------------------------------;
                                    ;19108          ALU_Q+LC,VAK/LOAD,
U 0B3A, 0011,2014,0180,F898,0200,0B3C ;19109        LA_RA[R3],LONG                  ; LOAD SIGN-BYTE ADDRESS
                                    ;19110          ;-------------------------------;
                                    ;19111          ALU_K[.1],N&Z_ALU.V&C_0,        ; CLEAR CONDITION-CODES
U 0B3C, 0018,8038,0580,4000,0050,0B3E ;19112        D[BYTE]_CACHE                   ; READ SIGN
                                    ;19113          ;-------------------------------;
                                    ;19114  CMPSGN1.1:
                                    ;19115          R[R1]_Q,Q_ID[T1],               ; RESET R1
U 0B3E, 0001,2F3C,C5F0,2E88,0000,0882 ;19116        BCDSGN?                         ; TEST SIGN OF STRING 1
                                    ;19117          ;-------------------------------;
                                    ;19118  =10     ;BRANCH ON SIGN-NIBBLE
                                    ;19119          ;10-----------------------------;
                                    ;19120  CMPSGN1.2:
                                    ;19121          ALU_0(A),R[R0]_ALU,             ; RESET R0 WITH 0
U 0B82, 0003,003C,0180,FA80,0000,0A93 ;19122        J/FINI3                         ; JOIN FINISH-ROUTINE
                                    ;19123          ;11-----------------------------;
                                    ;19124          ALU_K[.80],N&Z_ALU.V&C_0,       ; STRING 2 IS GREATER
U 0B83, 0018,8038,4180,F800,0050,0B82 ;19125        BYTE,J/CMPSGN1.2                ; SET N, CLEAR Z,C,V
                                    ;19126  =;END   ;-------------------------------;
                                    ;19127
                                    ;19128          ;ENTER HERE IF SRC2>SRC1
                                    ;19129          ;-------------------------------;
                                    ;19130  CMPSGN2.2:
                                    ;19131          ALU_0(A),R[R0]_ALU,             ; RESET R0
U 0B44, 0003,0F3C,C1F0,2E80,0000,088A ;19132        Q_ID[T0],BCDSGN?                ; GET SRC1-ADDRESS, TEST SIGN
                                    ;19133          ;-------------------------------;
                                    ;19134  =10     ;BRANCH ON SIGN-NIBBLE
                                    ;19135          ;10-----------------------------;
                                    ;19136  CMPSGN2.3:
                                    ;19137          R[R1]_Q,Q_ID[T1],               ; RESET R1, GET SRC2-ADDRESS
U 088A, 0001,203C,C5F0,2E88,0000,0A93 ;19138        J/FINI3                         ; STRING2 IS GREATER, FINISHED
                                    ;19139          ;11-----------------------------;
                                    ;19140          ALU_K[.1],N&Z_ALU.V&C_0,        ; STRING1 IS GREATER
U 088B, 0018,0038,0580,F800,0050,088A ;19141        J/CMPSGN2.3
                                    ;19142  =;END   ;-------------------------------;
```

```
                                ;19143  CP3SB:  ;ENTER HERE AFTER REACHING SIGN-BYTES
                                ;19144          ;D HAS 2. STRING SIGN-BYTE, Q HAS 1. STRING SIGN-BYTE
                                ;19145          ;-------------------------------;
                                ;19146          FE_SC,
                                ;19147          ALU_D.AND.K[.F0],CLK.UBCC,      ;  STRIP OFF SIGN-NIBBLE
U 0B45, 0C19,0034,CDE0,F800,0192,0B46   ;19148          SC_ALU,D_Q,Q_D                  ;  D GETS 2. STRING SIGN-BYTE
                                ;19149          ;-------------------------------;
                                ;19150          EALU_SC-FE,CLK.UBCC,            ;  COMPARE THE HIGH NIBBLES
U 0B46, 0000,013C,0187,F800,0010,A7F0   ;19151          SGN/CLR.SD+SS,Z?                ;  TEST FOR ZERO-STRING
                                ;19152          ;-------------------------------;
                                ;19153  =0      ;BRANCH ON ALU Z-BIT
                                ;19154          ;0------------------------------;
                                ;19155          ALU_K[.8],N&Z_ALU.V&C_0,        ;  CLEAR CONDITION CODES
                                ;19156          STATE_STATE.OR.K[.8],           ;  SET NOT-ALL-0-BIT OF STATE
U 07F0, 0018,1238,0180,F800,1454,2362   ;19157          EALU?,J/CMP.SGN.00              ;  TEST THE DIFFERENCE
                                ;19158          ;1------------------------------;
                                ;19159          ALU_K[.1],N&Z_ALU.V&C_0,        ;  CLEAR CONDITION CODES
U 07F1, 0018,1238,0580,F800,0050,0362   ;19160          EALU?,J/CMP.SGN.00              ;  TEST THE DIFFERENCE
                                ;19161  =;END   ;-------------------------------;
                                ;19162  =001*   ;BRANCH ON EALU Z AND N-BITS
                                ;19163          ;001*----------------------------;
                                ;19164  CMP.SGN.00:                             ;  SRC1-STRING IS GREATER
                                ;19165          Q_ID[TO],                       ;  GET SRC1-ADDRESS
U 0362, 0C00,003C,C1F0,2C00,0000,0B3E   ;19166          D_Q,J/CMPSGN1.1
                                ;19167          ;011*----------------------------;
U 0366, 0000,173C,0180,F800,0000,0025   ;19168          STATE3-0?,J/CMP.SGN.0           ;  TEST FOR NON-ZERO STRINGS
                                ;19169          ;101*----------------------------;
                                ;19170  CMPSGN21:
                                ;19171          ALU_K[.80],N&Z_ALU.V&C_0,BYTE,  ;  SRC2-STRING IS GREATER
U 036A, 0018,8038,4180,4000,0050,0B44   ;19172          D[BYTE]_CACHE,J/CMPSGN2.2        ;  READ SRC2-SIGN-BYTE
                                ;19173  =;END   ;-------------------------------;
                                ;19174  =01*1   ;BRANCH ON NOT-ALL-0-BIT OF STA
                                ;19175          ;01*1----------------------------;
                                ;19176  CMP.SGN.0:                              ;  BOTH STRINGS ARE 0
                                ;19177          ALU_0(A),N&Z_ALU.V&C_0,         ;  SET Z-BIT FOR EQUAL STRINGS
                                ;19178          R[R0]_ALU,Q_ID[TO],             ;  RESET R0, GET SRC1-ADDRESS
U 0025, 0003,003C,C1F0,2E80,0050,088D   ;19179          J/FINI1                         ;  SIGNS DON'T MATTER
                                ;19180          ;11*1----------------------------;
                                ;19181          ALU_0(A),N&Z_ALU.V&C_0,         ;  GUESS THAT THEY ARE EQUAL
                                ;19182          R[R0]_ALU,D_Q,Q_ID[TO],         ;  RESET R0, GET SRC1-ADDRESS
U 002D, 0C03,0F3C,C1F0,2E80,0050,0892   ;19183          BCDSGN?,J/CMP.SGN               ;  TEST SIGN OF SRC2
                                ;19184  =;END   ;-------------------------------;
```

```
                                        ;19185          ;ENTER HERE IF ALL BYTES ARE EQUAL EXCEPT SIGN-BYTES.
                                        ;19186  =10     ;BRANCH ON SIGN-NIBBLE OF 2. STRING
                                        ;19187          ;10------------------------------;
                                        ;19188  CMP.SGN:
                                        ;19189          R[R1] Q,Q_ID[T1],               ;  RESET R1,  GET SRC2-BYTE
U 0892, 0001,2F3C,C5F0,2E88,0000,08B2   ;19190          BCDSGN?,J7CMP.SGN.1             ;  2. STRING IS POSITIVE
                                        ;19191          ;11------------------------------;
                                        ;19192          R[R1] Q,Q_ID[T1],               ;  RESET R1,  GET SRC2-BYTE
U 0893, 0001,2F3C,C5F0,2E88,0000,08C2   ;19193          BCDSGN?,J7CMP.SGN.2             ;  2. STRING IS NEGATIVE
                                        ;19194  =;END   ;------------------------------;
                                        ;19195  =10     ;BRANCH ON SIGN-NIBBLE OF 1.STRING
                                        ;19196          ;10------------------------------;
                                        ;19197  CMP.SGN.1:
                                        ;19198          ALU Q,R[R3]_ALU,                ;  RESET R3
U 08B2, 0001,203C,0180,FA98,0000,0B29   ;19199          J/FINI8                         ;  FINISHED
                                        ;19200          ;11------------------------------;
                                        ;19201          ALU K[.80],N&Z_ALU.V&C_0,BYTE,  ;  1. STRING IS NEGATIVE
U 08B3, 0018,8038,4180,F800,0050,08B2   ;19202          J/CMP.SGN.1
                                        ;19203  =;END   ;------------------------------;
                                        ;19204  =10     ;BRANCH ON SIGN-NIBBLE OF 1.STRING
                                        ;19205          ;10------------------------------;
                                        ;19206  CMP.SGN.2:
                                        ;19207          ALU K[.1],N&Z_ALU.V&C_0,        ;  1.STRING IS GREATER THAN 2. STRING
U 08C2, 0018,0038,0580,F800,0050,08B2   ;19208          J/CMP.SGN.1
                                        ;19209          ;11------------------------------;
                                        ;19210  CMPFIN: ALU Q,R[R3]_ALU,                ;  RESET R3
U 08C3, 0001,203C,0180,FA98,0000,0B29   ;19211          J/FINI8
                                        ;19212  =;END   ;------------------------------;
```

```
;19213  .TOC    ""      Decimal string        : CVTLP''
;19214
;19215  ;CONVERT LONGWORD TO PACKED STRING
;19216  ;       ROUTINE FOR CONVERTING A LONGWORD INTO A PACKED,BCD,
;19217  ;       NUMERIC STRING.
;19218  ;ALGORITHM:
;19219  ;       1. STARTING AT 'L2P.INIT', THE LAST SPECIFIER IS EVALUATED,
;19220  ;       AND THE LENGTH IS CHECKED,AND FIRST PART DONE FLAG
;19221  ;       IS SET ('L2P00').
;19222
;19223  ;       2. BEFORE ENTERING THE LOOP, THE LONGWORD IS LEFT
;19224  ;       ADJUSTED ('L2P1') AND NEGATED IF NEGATIVE ('L2P10').
;19225
;19226  ;       3. THE ACTUAL LOOP STARTS AT 'L2PL', BUT WE ENTER AT 'L2PB'.
;19227  ;       THE ALGORITHM USED, IS A STEP-WISE, LEFT-TO-RIGHT,
;19228  ;       EVALUATION OF THE EXPRESSION:
;19229
;19230  ;       S= (...(A[N]*2+A[N-1])*2+A[N-2])*2+...+A[1])*2+A[0]
;19231
;19232  ;       WHERE A[N],A[N-1],...,A[0] ARE THE BITS IN
;19233  ;       SOURCE-LONGWORD.
;19234  ;       DURING EACH PASS THROUGH THE LOOP,
;19235  ;       THE NEXT BIT IS READ FROM THE SRC-LONGWORD ('L2PB'),
;19236  ;       THE STRING IS MULTIPLIED BY 2 (DECIMAL)('L2PL5'),
;19237  ;       AND THE BIT IS ADDED IN ('L2PL30').
;19238  ;       IF THE STRING DOES NOT FIT IN A SINGLE LONGWORD (8 DIGITS),
;19239  ;       ANOTHER REGISTER IS USED TO STORE THE UPPER DIGITS,
;19240  ;       AND HAS TO BE DOUBLED DURING EACH PASS ('L2PL5').
;19241
;19242  ;       4. WHEN THE STRING IS COMPLETE ('L2PL2'), THE STRING IS WRITTEN INTO
;19243  ;       MEMORY ('L2P.WRITE'), USING 'BCD-WRITE'- ROUTINE.
;19244
;19245  ;       5. FINALLY THE CONDITION CODES ARE SET,AND THE
;19246  ;       REGISTERS ARE LOADED WITH 0 OR ADDRESSES ('L2P.F2').
;19247
;19248  ;       6. IF A MEMORY-FAULT OR INTERRUPT OCCURS AFTER 1. PART DONE HAS
;19249  ;       BEEN SET, THE 'BCD.SAVE'-ROUTINE IS USED TO SAVE CONTEXT.
;19250  ;       ON RESTART, THE 'BCD.RESTORE'-ROUTINE IS USED TO RESTORE
;19251  ;       THE INITIAL CONTEXT, AND THE INSTRUCTION IS RESTARTED AT 'L2P00'.
```

```
;19252  ;STORAGE:
;19253  ;       UP TO TWO LONGWORDS MAY BE NEEDED FOR DESTINATION,AND
;19254  ;       R[R2],AND R[R3] ARE USED FOR STORAGE WHILE IT IS BEING
;19255  ;       GENERATED.
;19256  ;       ID[T4] IS  USED FOR STORING THE BINARY SRC-DATA,
;19257  ;       R[R1] CONTAINS DST-ADDRESS+1 (HIGH END OF STRING)
;19258  ;       R[R15] CONTAINS DST-LENGTH (NEGATIVE LENGTH-1)
;19259  ;       RC[T1] HAS  INITIAL DST-LENGTH
;19260  ;       RC[T5] HAS ORIGINAL LONGWORD.
;19261  ;       RC[T7] WILL CONTAIN ANY OVERFLOW DATA.
;19262  ;       SC CONTAINS SHIFT-COUNT.
;19263
;19264  ;       OP-CODE IS 'F9'
;19265  ;       MNEMONIC IS 'CVTLP'
;19266  ;       INSTRUCTION-FORMAT IS:
;19267  ;       opcode  src.rl, dstlen.rw, dstaddr.ab
;19268
;19269  ;       INST. DEPENDENT ALU FUNCTION IS 'A-B-PSL.BORROW'
;19270  ;       INST. DEPENDENT CLOCKING OF CC IS: Z_Z,N_N,V_0,C_ALU CARRY[UDT]
;19271
;19272          ;STATE-REGISTER
;19273          ;-------------------------------------------------------------
;19274          ;INTR:  ;OVFL.  ;       ;       ;       ;1.TIM: ;SGN:   ;       ;
;19275          ;       ;       ;       ;       ;       ;0=1.   ;1=NEG  ;       ;
;19276          ;       ;       ;       ;       ;       ;1=>1.  ;0=POS  ;       ;
;19277          ;       ;       ;       ;       ;       ;       ;       ;       ;
;19278          ;-------------------------------------------------------------
```

```
                                ;19279          ;ENTER HERE FROM C-FORK WITH LONGWORD IN Q,LENGTH IN D
                                ;19280          ;-------------------------------;
                                ;19281  342:
                                ;19282  L2P.INIT:       ;EVALUATE SPECIFIERS AND INITIALIZE REGISTERS. SET FPD.
                                ;19283          STATE_K[ZERO],                  ;  INITIALIZE STATE-REGISTER
                                ;19284          ALU_D.OXT[WORD],D_ALU,          ;  ISOLATE DST-LENGTH
U 0342, 0803,403C,1980,F988,1404,6052   ;19285          RC[T1]_ALU                      ;  SAVE IT IN RC[T1]
                                ;19286          ;-------------------------------;
                                ;19287  =10*0***
                                ;19288          ALU_Q,RC[T5]_ALU,               ;  SAVE LONGWORD IN RC5
U 0052, 0601,203D,0180,F9A8,0000,047E   ;19289          DK/RIGHT,CAL[,J/ASPC            ;  DIVIDE LENGTH BY 2, EVALUATE DST-ADDR
                                ;19290          ;-------------------------------;
                                ;19291  =11*0***                                ;  REENTER HERE AFTER A FAULT
                                ;19292  L2P00:  ALU_Q.AND.K[.FFF0],             ;  CLOCK EXCESS LENGTH-BITS
                                ;19293          N&Z_ALU.V&C_0,R[R3]_ALU,        ;  CLEAR R3 (LOW WORD ONLY)
U 0072, 0019,2035,6D80,FA98,0050,0B25   ;19294          CAL[,J/BCD.FPD.0                ;  CALL SUBROUTINE TO SET
                                ;19295                                          ;  FPD-FLAG AND GENERATE FPD-ADDR.
                                ;19296                                          ;  ALSO TEST FOR ILLEGAL LENGTH
                                ;19297          ;-------------------------------;
                                ;19298  =11*1***
                                ;19299          ID[FPDA]_D,                     ;  WRITE FAULT-ADDRESS(33) IN ID-REG.
U 007A, 0810,0038,B580,3D28,0000,0B4C   ;19300          D_RC[T5],J/L2P0                 ;  GET ORIGINAL LONGWORD
                                ;19301  =;END    ;-------------------------------;
                                ;19302  L2P0:   SC_K[.20],                      ;  LOAD INITIAL BIT-COUNT (32.)
                                ;19303          ALU_K[.20],                     ;  USE 10 TO INITIALIZE DST-STRING
                                ;19304          R[R2]_ALU.RIGHT,                ;  LEAVING LOW NIBBLE FOR SIGN
U 0B4C, 0058,0D38,7580,FA90,0084,6B54   ;19305          BEN/SIGNS                       ;  TEST SIGN-BIT AND D-REGISTER
                                ;19306  =;END    ;-------------------------------;
                                ;19307  =100    ;BRANCH ON D<31> AND D NE 0
                                ;19308          ;100----------------------------;
                                ;19309          ALU_0(A),R[R2]_ALU,             ;  LONGWORD IS ZERO
                                ;19310          N&Z_ALU.V&C_0,                  ;  SET Z-BIT
U 0B54, 0003,003C,0180,FA90,0050,08C9   ;19311          J/L2PL2                         ;  FINISHED , NO CALCULATIONS NECESSARY
                                ;19312  =10     ;110----------------------------;
                                ;19313  L2P1:   SC_SC-SHF.VAL,                  ;  UPDATE BIT-COUNT (LEFT ADJUSTED)
                                ;19314          ALU_0(A),N&Z_ALU.V&C_0,         ;  SET Z-BIT
                                ;19315          D_DAL.NORM,                     ;  NORMALIZE D
U 0B56, 0E03,003C,0180,F800,00DC,AB4D   ;19316          J/L2P2                          ;  ENTER MAIN LOOP
                                ;19317          ;111----------------------------;
                                ;19318  L2P10:  D_0-D,                          ;  NEGATE LONGWORD
                                ;19319          STATE_STATE.OR.K[.2],           ;  SET MINUS BIT OF STATE
U 0B57, 081F,2000,0980,F800,1404,2B56   ;19320          J/L2PT                          ;  JOIN POSITIVE PATH
                                ;19321  =;END    ;-------------------------------;
                                ;19322  L2P2:   ALU_D,Q_ALU.LEFT,               ;  GET 2. BIT OF LONGWORD
                                ;19323          INTRPT.STROBE,                  ;  STROBE FOR INTERRUPTS
U 0B4D, 0021,003C,01C0,F800,4000,019C   ;19324          J/L2PB                          ;  START LOOPING
                                ;19325          ;-------------------------------;
```

```
                                    ;19326          ;LOOP WHICH ADDS DECIMAL STRING IN R2(AND R3 IF
                                    ;19327          ;NECESSARY) TO ITSELF, AND ADDS IN HIGH ORDER BIT OF LONGWORD IN T4.
                                    ;19328          ;ENTER LOOP AT L2PB, WITH LONGWORD IN Q,
                                    ;19329          ;AND BIT-COUNT OF LONGWORD IN SC.
                                    ;19330          ;-------------------------------;
                                    ;19331  =10*0   ;BRANCH ON PSL Z AND C (V=0)
                                    ;19332          ;10*0-------------------------------;
                                    ;19333  L2PL:   ALU_LA+LC,D_ALU,                    ; ADD 6'S TO DATA
U 0198, 0810,0014,0180,F800,0000,0B55 ;19334        J/L2PL5                             ; DOUBLE PRECISION, NO CARRY
                                    ;19335          ;10*1-------------------------------;
                                    ;19336          ALU_LA+LC,D_ALU,                    ; ADD 6'S TO DATA
U 0199, 0810,0014,0180,F800,0000,0B55 ;19337        J/L2PL5                             ; DOUBLE PREC. ,CARRY
                                    ;19338          ;11*0-------------------------------;
                                    ;19339  L2PB:   ALU_0+Q,LAB_R[R2],LONG,             ; SHIFT DATA LEFT AND STORE IT
                                    ;19340          D_ALU.LEFT,                         ; IN D-REGISTER
                                    ;19341          Q_DEC.CON,                          ; LOAD ALL 6'S IN Q
                                    ;19342          C[K.UBCC,                           ; CLOCK HIGH ORDER BIT
                                    ;19343          SC_SC-K[.1],                        ; DECREMENT COUNT
                                    ;19344          GEN/INTERRUPT,                      ; TEST FOR INTERRUPT
U 019C, 083F,0E14,05D0,FA10,0094,AB66 ;19345        J/L2PL1
                                    ;19346          ;11*1-------------------------------;
                                    ;19347          ALU_0+K[.1],R[R3]_ALU,              ; START 2. HALF OF STRING
U 019D, 0C1B,0014,05D0,FA98,0050,0B4E ;19348        D_Q,Q_DEC.CON,N&Z_ALU.V&C_0         ; CLEAR Z-BIT TO SIGNAL DOUBLE PREC.
                                    ;19349  =;END   ;-------------------------------;
                                    ;19350          ALU_Q,RC[T7]_ALU,                   ; STORE ALL 6'S IN RC[T7]
U 0B4E, 0001,203C,01E0,F988,0000,019C ;19351        Q_D,J/L2PB                          ; RESTORE DATA TO Q, REJOIN LOOP
                                    ;19352          ;-------------------------------;
                                    ;19353  =110    ;BRANCH ON INTERRUPT-PENDING
                                    ;19354          ;110--------------------------------;
                                    ;19355  L2PL1:  ID[T4]_D,                           ; SAVE LONGWORD
                                    ;19356          Q_LA+Q,                             ; ADD 6'S
                                    ;19357          LC_RC[T7],                          ; GET DECIMAL CONSTANT READY
                                    ;19358          SC?,                                ; TEST COUNT
U 0866, 001C,1414,D1C0,3D38,0000,08C9 ;19359        J/L2PL2
                                    ;19360          ;111--------------------------------;
U 0867, 0000,003C,4180,F800,1404,6033 ;19361        STATE_K[.80],J/SAVE.BCD             ; SET INTERRUPT-BIT, SAVE CONTEXT
                                    ;19362          ;-------------------------------;
```

```
                                    ;19363          ;CONTINUATION OF BASIC LOOP, CONVERTING A LONGWORD
                                    ;19364          ;TO A PACKED DECIMAL STRING.
                                    ;19365          ;-------------------------------;
                                    ;19366  =01     ;BRANCH ON SC GT 0 (SC<9:8>=0)
                                    ;19367          ;01-------------------------------;
                                    ;19368  L2PL2:  ALU_0(A),LC_RC[T1],             ;  GET DST-LENGTH
                                    ;19369          N&Z_ALU.V&C_0,                  ;  CLEAR N-BIT, SET Z-BIT
U 08C9, 0003,003C,0180,F908,0050,0B5D   ;19370          J/L2P.WRITE                     ;  FINISHED, WRITE DST-STRING
                                    ;19371          ;11-------------------------------;
                                    ;19372          D_Q+LB,LA_RA[R3],               ;  ADD DATA TO ITSELF
                                    ;19373          Q_DEC.CON,                      ;  DECIMAL CONSTANT FOR ADJUSTMENT
                                    ;19374          SET.CC(LONG),                   ;  CLOCK PSL-CARRY-BIT
                                    ;19375          K[.10],                         ;  CONSTANT FOR NEXT INSTRUCTION
                                    ;19376          ALU.N?,                         ;  TEST HIGH BIT OF BINARY SRC
U 08CB, 080D,3B14,65D0,F898,0070,0436   ;19377          J/L2PL3
                                    ;19378  =;END   ;-------------------------------;
                                    ;19379  =011*   ;BRANCH ON ALU N-BIT (C31 IS CLEAR)
                                    ;19380          ;011*-----------------------------;
                                    ;19381  L2PL3:  R[R2] D-Q,LONG,                 ;  DECIMAL ADJUST AND STORE SUM
                                    ;19382          Q_ID[T4],                       ;  RETRIEVE LONGWORD
                                    ;19383          PSL.CC?,                        ;  TEST PSL Z AND C-BITS
                                    ;19384          INTRPT.STROBE,                  ;  STROBE FOR INTERRUPTS
U 0436, 001D,1A00,D1F0,2E90,4000,0198   ;19385          J/L2PL
                                    ;19386          ;111*-----------------------------;
                                    ;19387  L2PL30: Q_Q-K[.10],                     ;  ADD IN NEW BIT
U 043E, 0019,2000,65C0,F800,0000,0436   ;19388          J/L2PL3                         ;  AFTER SIGN-NIBBLE
                                    ;19389  =;END   ;-------------------------------;
                                    ;19390  L2PL5:  ;ROUTINE TO DOUBLE SECOND HALF OF STRING
                                    ;19391          ;-------------------------------;
U 0B55, 081C,202C,01D0,F800,0000,0B5C   ;19392          D_LA+D+PSL.C,Q_DEC.CON          ;  ADD STRING TO ITSELF WITH CARRY
                                    ;19393          ;-------------------------------;
                                    ;19394          R[R3] D-Q,LONG,                 ;  DECIMAL ADJUST, STORE RESULT
                                    ;19395          Q_ID[T4],                       ;  GET SOURCE LONGWORD READY
U 0B5C, 001D,0000,D1F0,2E98,0000,019C   ;19396          J/L2PB                          ;  LOOP BACK
                                    ;19397          ;-------------------------------;
```

```
                                ;19398  L2P.WRITE:
                                ;19399          ;ROUTINE WHICH WRITES DATA IN R[R2] AND R[R3] INTO DST-STRING.
                                ;19400          ;-------------------------------;
                                ;19401          STATE_STATE.ANDNOT.K[.F0],      ;  CLEAR UPPER BITS OF STATE
                                ;19402          ALU_0-LC-1,R[R15]_ALU,          ;  GET DST-LENGTH
U 0B5D, 0013,0008,CDC0,FAF8,1414,4B64   ;19403          CLK.UBCC,Q_ALU
                                ;19404          ;-------------------------------;
U 0B64, 0800,003C,0180,FA10,0000,0B65   ;19405  L2P.W0: D_R[R2]                  ;  GET FIRST LONGWORD
                                ;19406          ;-------------------------------;
U 0B65, 0000,003C,01C0,FA78,0010,029C   ;19407  L2P.W1: Q_R[P   ,CLK.UBCC,LONG    ;  GET LENGTH
                                ;19408          ;-------------------------------;
                                ;19409  =00*****
                                ;19410  L2P.    LA_RA[R1],                      ;  GET DST-ADDRESS
                                ;19411          ALU_Q+K[.8],                    ;  INCREMENT DST-LENGTH
                                ;19412          SHF7ALU.DT,LONG,
                                ;194            SC_ALU,QK/SHF,                  ;  STORE IN SC TO MAKE MASK
                                ;194            CLR.UBCC,
                                ;19415          SIGNS?,                         ;  TEST FOR END OF STRING AND ZERO
U 029C, 0079,2D15,01C0,F888,0092,0E59   ;19416          CALL,J/WRITE1                   ;  GET ADDRESS, WRITE DATA
                                ;19417          ;01*****-----------------------;
                                ;19418  L2P.F2: ALU_0(A),R[R0]_ALU,              ;  END OF DST-STRING, CLEAR R0
                                ;19419          N_AMX.Z_TST,                    ;  CLEAR N-BIT
                                ;19420          D_0,Q_ID[T0],                   ;  GET DST-LENGTH
U 02BC, 0F03,0D3C,C1F0,2E80,0030,01B4   ;19421          SIGNS?,J/L2P.F3                 ;  TEST FOR OVERFLOW
                                ;19422          ;-------------------------------;
                                ;19423  =11*****
U 02FC, 0018,0000,1180,FA88,0000,0B6B   ;19424          R[R1]_LA-K[.4],LONG             ;  UPDATE ADDRESS
                                ;19425  =;END   ;-------------------------------;
                                ;19426          Q_LB,                           ;  GET LENGTH
                                ;19427          LA_RA[R3],                      ;  GET NEXT DATA
U 0B6B, 000C,0038,11C0,F898,1404,2B6C   ;19428          STATE_STATE.OR.K[.4]            ;  CLEAR 1. TIME FLAG
                                ;19429          ;-------------------------------;
U 0B6C, 0019,2014,01C0,FAF8,0000,0B6E   ;19430          R[R15]_Q+K[.8],LONG,Q_ALU       ;  UPDATE LENGTH
                                ;19431          ;-------------------------------;
                                ;19432          D_LA.AND.K[.FFFF],
U 0B6E, 0818,0034,C180,FA90,0000,0B73   ;19433          R[R2]_ALU,LONG                  ;  GET NEW DATA
                                ;19434          ;-------------------------------;
                                ;19435          R[R3]_0,LONG,
U 0B73, 0003,003C,0180,FA98,0000,0B65   ;19436          J/L2P.W1                        ;  NO MORE DATA AFTER THIS WRITE
                                ;19437          ;-------------------------------;
                                ;19438  =10*     ;BRANCH ON D NE 0 (D<31>=0)
                                ;19439          ;10*----------------------------;
                                ;19440  L2P.F3: ID[T1]_D,STATE3-0?,             ;  CLEAR T1, TEST SIGN-BIT,
U 01B4, 0000,173C,C580,3C00,0000,088D   ;19441          J/FINIT                         ;  JOIN FINISH-ROUTINE
                                ;19442
                                ;19443  ; ************************************************
                                ;19444  ; * Patch no. 002, PCS 01B4 trapped to WCS 1141 *
                                ;19445  ; ************************************************
                                ;19446
                                ;19447          ;11*----------------------------;
                                ;19448          STATE_STATE.OR.K[.40],          ;  SET OVERFLOW BIT
U 01B6, 0000,003C,3180,F800,1404,21B4   ;19449          J/L2P.F3
                                ;19450  =;END   ;-------------------------------;
```

```
;19451  .TOC    ""      Decimal string        : CVTPL""
;19452
;19453  ;CONVERT PACKED STRING TO LONGWORD
;19454  ;ROUTINE WHICH CONVERTS A PACKED STRING INTO A LONGWORD.
;19455  ;
;19456  ;ALGORITHM:
;19457  ;       1. FIRST THE LAST SPECIFIER IS EVALUATED, AND
;19458  ;       THE LENGTH IS TESTED, AND FIRST PART DONE FLAG IS SET.
;19459  ;       THE SS-FLIP-FLOP IS USED TO REMEMBER WHETHER THE
;19460  ;       DESTINATION IS TO BE STORED IN MEMORY OR A REGISTER.
;19461
;19462  ;       2. WE PROCEED TO LOOK FOR A LEADING NON-ZERO DIGIT
;19463  ;       IN THE SRC-STRING, LOOPING ON LENGTH AND NON-ZERO DATA ('P2L').
;19464
;19465  ;       3. ONCE WE FIND A NON-ZERO BYTE, WE ENTER THE MAIN LOOP.
;19466  ;       THE PROCESS WE USE, IS A STEP BY STEP EVALUATION OF THE
;19467  ;       EXPRESSION:
;19468
;19469  ;       S= (...(A[N]*10+A[N-1])*100+A[N-2]*10+A[N-3])*100+...+
;19470  ;                       +A[4]*10+A[3])*100+A[2]*10+A[1])*10+A[0]
;19471
;19472  ;       WHERE A[N],...,A[0] ARE THE DIGITS IN THE DECIMAL STRING.
;19473  ;       ON EACH PASS THROUGH THE LOOP, WE READ TWO MORE DIGITS FROM
;19474  ;       THE SRC-STRING, MULTIPLY ONE OF THEM BY 10. (BINARY),
;19475  ;       ADD THE OTHER DIGIT IN, AND ADD THE RESULT TO THE RUNNING SUM.
;19476  ;       THE SUM IS THEN MULTIPLIED BY ONE HUNDRED, AS THE
;19477  ;       NEXT TWO DIGITS ARE READ IN ('P2LL1').
;19478
;19479  ;       4. FINALLY WE REACH THE SIGN-BYTE ('P2LS'), THE LAST DIGIT
;19480  ;       IS ADDED IN ('P2LS1'), WE SET THE CONDITION CODES,
;19481  ;       AND LOAD THE GENERAL REGISTERS.
;19482
;19483          ;SC CONTAINS SRC-LENGTH,
;19484          ;R1 GETS SRC-ADDRESS
;19485          ;ORIGINAL SRC-LENGTH IS SAVED IN RC0
;19486          ;SRC-ADDRESS IS SAVED IN T0
;19487          ;DST-ADDSRESS IS SAVED IN T1
;19488          ;SS-FLIP-FLOP DETERMINES WHETHER DST IS IN MEMORY OR GEN. REG.
;19489          ;STATE-REGISTER:
;19490          ;-------------------------------------------------------------
;19491          ;INTRPT ;OVFLOW ;       ;       ;       ;       ;PACKED ;       ;
;19492          ;       ;       ;       ;       ;       ;       ;DECIML ;       ;
;19493          ;       ;       ;       ;       ;       ;       ;SIGN   ;       ;
;19494          ;       ;       ;       ;       ;       ;       ;       ;       ;
;19495          ;-------------------------------------------------------------
```

```
                                ;19496          ;THE FIRST ROUTINE READS LEADING ZERO'S IN THE SRC-STRING,
                                ;19497          ;AND TESTS FOR OVERFLOW BY COMPARING NUMBER OF
                                ;19498          ;NON-ZERO BYTES WITH 6, AND IF THEY ARE EQUAL, COMPARING
                                ;19499          ;THE LEADING BYTE WITH 02.
                                ;19500
                                ;19501
                                ;19502          ;ENTER HERE FROM C-FORK WITH SRC-LENGTH IN Q,SRC-ADDRESS IN D
                                ;19503          ;-------------------------------;
                                ;19504  340:    ALU_Q.OXT[WORD].ANDNOT.K[.1F],  ;  ISOLATE HIGH BITS OF LENGTH
                                ;19505          RC[T1]_ALU,                     ;  CLEAR RC1
                                ;19506          N&Z_ALU.V&C_0,                  ;  CLOCK IT
                                ;19507          QK/RIGHT,SGN/CLR.SD+SS,         ;  DIVIDE LENGTH BY 2
U 0340, 001B,6024,8DB7,F988,1450,60B0   ;19508  STATE_FE
                                ;19509          ;-------------------------------;
                                ;19510  =01*0**0
                                ;19511          ALU_Q.OXT[BYTE],                ;  ISOLATE SRC-LENGTH
                                ;19512          SC_ALU,                         ;  LOAD SRC-LENGTH/2 IN SC
                                ;19513          RC[T0]_ALU,                     ;  SAVE IT IN RC0 AS WELL
                                ;19514          SGN/NOT.SD,                     ;  SET SD
                                ;19515          STATE_STATE-FE,                 ;  CLEAR STATE-REGISTER
U 00B0, 0003,A03D,C183,3D80,1482,A47E   ;19516  ID[T0]_D,CALL,J/ASPC             ;  SAVE SRC-ADDRESS IN ID[T0]
                                ;19517          ;-------------------------------;
                                ;19518  =11*0**0
                                ;19519  P2L00:  R[R3]_D,ID[T1]_D,               ;  SAVE ADDRESS IN R3 AND T1
U 00F0, 0001,1A3D,C580,3E98,0000,00F9   ;19520  PSL.CC?,CALL,J/BCD.FPD          ;  SET FPD, TEST FOR LEGAL LENGTH
                                ;19521          ;-------------------------------;
U 00F1, 0800,0038,0180,F800,1C00,0B74   ;19522  D_RLOG,J/P2L02                   ;  GET REGISTER NUMBER OFF RLOG
                                ;19523                                          ;  (BUT IT IS IN UPPER BITS, DUMMY!)
                                ;19524          ;-------------------------------;
                                ;19525  =11*1**0
                                ;19526  P2L0:   ID[FPDA]_D,                     ;  SAVE MEMORY-FAULT ADDRESS (33)
                                ;19527          ALU_Q,VAR/LOAD,R[R1]_ALU,       ;  LOAD SRC-ADDRESS
U 00F8, 0001,203C,B580,3E88,0300,090E   ;19528  FE_SC,J/P2L1
                                ;19529  =;END   ;-------------------------------;
                                ;19530  P2L02:  ALU_K[.FF],                     ;  REMEMBER ACROSS FAULTS
                                ;19531          RC[T1]_ALU,SGN/SS.FROM.SD,      ;  SET REGISTER FLAG
U 0B74, 0018,0038,4982,F988,0000,00F0   ;19532  J/P2L00
                                ;19533          ;-------------------------------;
```

```
                                ;19534  =101    ;BRANCH ON SC GT 0
                                ;19535          ;101----------------------------;
                                ;19536  P2L:    SC_FE,                          ; GET ORIGINAL LENGTH
                                ;19537          ALU_D.OXT[BYTE].AND.K[.FFF0],   ; STRIP OFF SIGN-NIBBLE
                                ;19538          D_A[U,RIGHT2,                   ; DIVIDE BY 4
                                ;19539          L[_RC[T2],BCDSGN?,              ; GET SUM,TEST SIGN
U 0885, 089B,8F34,6D80,F910,0081,08D2   ;19540  J/P2LS                          ; SIGN-BYTE, SKIP TO SIGN-ROUTINE.
                                ;19541          ;111----------------------------; SRC-LENGTH IS >=0
                                ;19542          ALU_LA+K[.1],VAK/LOAD,          ; LOAD SRC-ADDRESS
                                ;19543          R[RT]_ALU,LONG,D.B0?,           ; TEST SOURCE-DATA
U 0887, 0018,1814,0580,FA88,0200,090E   ;19544  J/P2LT                          ; LOAD SRC-ADDRESS,TEST FOR NON-ZERO D
                                ;19545  =;END   ;----------------------------;
                                ;19546  =1110   ;BRANCH ON LOW BYTE OF D
                                ;19547          ;1110---------------------------; LOOK FOR LEADING NON-ZERO BYTE
                                ;19548  P2L1:   D[BYTE]_CACHE,                  ; READ NEXT BYTE
                                ;19549          SC_SC-K[.1],                    ; UPDATE SRC-LENGTH
                                ;19550          ALU_K[.1],
                                ;19551          LAB_R1&RC[T2]_ALU.RIGHT2,       ; LOAD ADDRESS IN LA AND CLEAR SUM
U 090E, 0098,9438,05F8,4310,0084,AB85   ;19552  Q_0,SC?,J/P2L                    ; TEST SRC-LENGTH
                                ;19553          ;1111---------------------------; D HAS NON-ZERO BYTE
                                ;19554          EALU_SC-K[.4],CLK.UBCC,         ; CLOCK OVERFLOW LIMIT
U 090F, 0000,003C,3180,FA88,0014,AB75   ;19555  ALU_LA,R[R1]_ALU,J/P2L2          ; ADJUST SRC-ADDRESS
                                ;19556  =;END   ;----------------------------;
                                ;19557  P2L2:   ALU_D-K[.3],CLK.UBCC,BYTE,
U 0B75, 0019,9200,0D80,F800,0010,0913   ;19558  EALU?                           ; TEST LENGTH OF NONZERO STRING
                                ;19559          ;----------------------------;
                                ;19560  =0011   ;BRANCH ON EALU N- AND Z-BIT
                                ;19561          ;0011---------------------------;
U 0913, 0000,003C,3180,F800,1404,291B   ;19562  STATE_STATE.OR.K[.40],J/P2LL      ; SET OVERFLOW-BIT
                                ;19563          ;0111---------------------------; ON THE BOUNDARY MAYBE OVERFLOW
                                ;19564          ALU_D.OXT[BYTE].AND.K[.FFF0],   ; ISOLATE HIGH NIBBLE
                                ;19565          Q_A[U,RIGHT2,                   ; SHIFTED RIGHT TWICE
U 0917, 009B,9B34,6DC0,F800,0000,0175   ;19566  ALU.N?,J/P2L3                    ; TEST HIGH DIGIT
                                ;19567
                                ;19568  ; ***************************************************
                                ;19569  ; * Patch no. 013, PCS 0917 trapped to WCS 114D *
                                ;19570  ; ***************************************************
                                ;19571
                                ;19572          ;1011---------------------------; NO OVERFLOW,START MAIN LOOP
                                ;19573  P2LL:   ALU_D.OXT[BYTE].AND.K[.FFF0],   ; ISOLATE HIGH NIBBLE
                                ;19574          Q_A[U,RIGHT2,                   ; SHIFTED RIGHT TWICE
U 091B, 009B,8034,6DC0,F800,0000,017D   ;19575  J/P2LL0
                                ;19576  =;END   ;----------------------------;
                                ;19577  =01*1   ;BRANCH ON ALU N-BIT (IR<0>=0)
                                ;19578          ;01*1---------------------------;
                                  19579  P2L3:   STATE_STATE.OR.K[.40],         ; SET OVERFLOW-BIT OF STATE
                                ;19580          ALU_D.OXT[BYTE]-Q,              ; SUBTRACT 4*HIGH NIBBLE
                                ;19581          D_A[U,QK/RIGHT,LC_RC[T2],       ; STORE RESULT IN D, GET SUM IN LC
U 0175, 081F,8000,31B0,F910,1404,2B7C   ;19582  J/P2LL2
                                ;19583          ;11*1---------------------------;
                                ;19584  P2LL0:  ALU_D.OXT[BYTE]-Q,              ; SUBTRACT 4*HIGH NIBBLE
                                ;19585          D_A[U,QK/RIGHT,LC_RC[T2],       ; STORE RESULT IN D, GET SUM IN LC
U 017D, 081F,8000,01B0,F910,0000,0B7C   ;19586  J/P2LL2
                                ;19587  =;END   ;----------------------------;
```

```
                                ;19588          ;CONTINUATION OF
                                ;19589          ;MAIN LOOP FOR CONVERTING PACKED STRING TO LONGWORD.
                                ;19590          ;R[T2] HAS CURRENT SUM
                                ;19591          ;SC HAS SRC-LENGTH
                                ;19592
                                ;19593          ;--------------------------------;
                                ;19594  P2LL2:   ALU_D-Q,D_ALU,LA_RA[R1],        ;  SUBTRACT 2*HIGH NIBBLE, GET ADDRESS
U 087C, 081D,0000,0580,F888,0094,AB7D   ;19595          SC_SC-K[.T],CLK.0BCC        ;  UPDATE SRC-LENGTH
                                ;19596          ;--------------------------------;
                                ;19597          ALU_D+LC,SHF/ALU.DT,            ;  ADD INTO SUM
                                ;19598          QK/SHF,DK/SHF,                  ;  STORE RESULT IN Q,D,AND R[T2]
U 087D, 0871,0014,01C0,F990,0000,0883   ;19599          R[T2]_ALU.LEFT2             ;  SHIFT LEFT TWICE
                                ;19600
                                ;19601  ; ***********************************************
                                ;19602  ; * Patch no. 056, PCS 087D trapped to WCS 1182 *
                                ;19603  ; ***********************************************
                                ;19604
                                ;19605          ;--------------------------------;
                                ;19606          INTRFT.STROBE,                  ;  STROBE INTERRUPTS
                                ;19607          ALU_LA+K[.1],                   ;  INCREMENT SRC-ADDRESS
                                ;19608          R[RT]_ALU,LONG,                 ;  UPDATE IT
                                ;19609          VAK/LOAD,                       ;  LOAD SRC-ADDRESS
U 0883, 0518,1214,0580,FA88,4200,0927   ;19610          DK/LEFT,BEN/EALU            ;  TEST SRC-LENGTH FOR SIGN-BYTE
                                ;19611          ;--------------------------------;
                                ;19612  =0111   ;BRANCH ON EALU N-BIT
                                ;19613          ;0111-----------------------------;
                                ;19614          ALU_D+Q,                        ;  SHIFT LEFT 3 TIMES
                                ;19615          D_ALU.LEFT3,                    ;  ADD AND MULTIPLY RESULT BY 8
                                ;19616          LC_RC[T2],                      ;  GET PARTIAL SUM
U 0927, 08BD,0E14,0180,F910,0000,0B96   ;19617          BEN/INTERRUPT,J/P2LL1       ;  TEST FOR PENDING INTERRUPTS
                                ;19618          ;1111-----------------------------;
                                ;19619          D[BYTE]_CACHE,ALU_Q,            ;  READ SIGN-BYTE
                                ;19620          RC[T2]_ALU.LEFT,                ;  STORE SUM IN RC[T2]
J 092F, 0021,A03C,01B0,4190,0000,0B85   ;19621          QK/RIGHT,J/P2L              ;  SHIFT THE DATA
                                ;19622  =;END    ;--------------------------------;
                                ;19623  =110    ;BRANCH ON INTERRUPT REQUEST
                                ;19624          ;110------------------------------;
                                ;19625  P2LL1:   Q_D+LC,RC[T2]_ALU,              ;  SAVE SUM*100 IN RC[T2]
U 0B96, 0011,8014,01C0,4190,0000,091B   ;19626          D[BYTE]_CACHE,J/P2LL        ;  GET NEXT DATA
                                ;19627          ;111------------------------------;
                                ;19628          STATE_STATE.OR.K[.80],          ;  SET INTERRUPT-BIT OF STATE
U 0B97, 0000,003C,4180,F800,1404,2033   ;19629          J/SAVE.BCD                  ;  SAVE CONTEXT OF INSTRUCTION
                                ;19630  =;END    ;--------------------------------;
                                ;19631          ;ENTER HERE AFTER READING SIGN-BYTE
                                ;19632  =10     ;BRANCH ON DECIMAL SIGN
                                ;19633          ;10-------------------------------;
                                ;19634  P2LS:    ALU_Q+LC,Q_ALU,                 ;  SET PLUS-BIT,GET SUM*10
U 08D2, 0211,2014,01C0,F800,0000,0B84   ;19635          DK/RIGHT2,J/P2LS1
                                ;19636          ;11-------------------------------;
                                ;19637          ALU_Q+LC,Q_ALU,                 ;  GENERATE SUM*10
                                ;19638          STATE_STATE.OR.K[.2],           ;  SET MINUS-BIT
U 08D3, 0211,2014,09C0,F800,1404,2B84   ;19639          DK/RIGHT2,J/P2LS1           ;  SHIFT LAST NIBBLE RIGHT
                                ;19640  =;END    ;--------------------------------;
```

```
                                ;19641  P2LS1:  ALU_D+Q,D_ALU,
U 0B84, 081D,0014,C5F0,2C00,0000,0B86   ;19642          Q_ID[T1]
                                ;19643          ;-------------------------------;
                                ;19644          EALU_K[.1],CLK.UBCC,            ;  CLEAR EALU CC
                                ;19645          ALU_0,VAK/LOAD,SC_ALU,          ;  LOAD DST-ADDRESS
U 0B86, 0001,.,73C,05F8,F800,0296,66F8  ;19646          Q_0,STATE3-0?                   ;  TEST SIGN-BIT
                                ;19647
                                ;19648  ; ************************************************
                                ;19649  ; * Patch no. 014, PCS 0B86 trapped to WCS 114E *
                                ;19650  ; ************************************************
                                ;19651          ;-------------------------------;
                                ;19652  =0*     ;BRANCH ON SIGN-BIT
                                ;19653          ;0*-----------------------------;
                                ;19654  P2LF2:  ALU_D.XOR.Q,Q_ALU,              ;  XOR THE HIGH BITS
U 06F8, 001D,0020,01C0,F800,0000,0B8B   ;19655          J/P2LF3                         ;  TO TEST FOR OVERFLOW
                                ;19656          ;1*-----------------------------;
                                ;19657          ALU_0-D,D_ALU,                  ;  NEGATE LONGWORD
                                ;19658          QK/RIGHT,SI/MUL-,               ;  SET HIGH BIT OF Q
U 06FA, 081F,2000,03B0,F800,0000,06F8   ;19659          J/P2LF2                         ;  WOOPS-FORGOT -0 CASE!!!
                                ;19660  =;END   ;-------------------------------;
                                ;19661  P2LF3:  ALU_0(A),R[R0]_ALU,             ;  CLEAR R0
U 0B8B, 0003,0D3C,C1F0,2E80,0000,0BC3   ;19662          Q_ID[T0],Q31?                   ;  GET SRC-ADDRESS, TEST Q
                                ;19663          ;-------------------------------;
                                ;19664  =011    ;BRANCH ON Q<31>
                                ;19665          ;011----------------------------;
                                ;19666          ALU_0(A),R[R2]_ALU,             ;  CLEAR R2
U 0BC3, 0003,003C,0180,FA90,0000,0B8C   ;19667          J/P2LF5
                                ;19668          ;111----------------------------;
                                ;19669          STATE_STATE.OR.K[.40],          ;  SET OVERFLOW-BIT OF STATE
                                ;19670          ALU_0(A),R[R2]_ALU,             ;  CLEAR R2
U 0BC7, 0003,003C,3180,FA90,1404,2B8C   ;19671          J/P2LF5
                                ;19672  =;END   ;-------------------------------;
                                ;19673  P2LF5:  ALU_0,R[R1]_ALU,                ;  LOAD SRC-ADDRESS
U 0B8C, 0001,203C,31F0,2E88,0000,0B8E   ;19674          Q_ID[CES]
                                ;19675          ;-------------------------------;
                                ;19676          ALU_0(A),R[R3]_ALU,             ;  CLEAR R0
U 0B8E, 0003,123C,0180,FA98,0000,08E2   ;19677          EALU?,J/P2LF7                   ;  TEST FOR REGISTER OR MEMORY
                                ;19678          ;-------------------------------;
```

```
                                        ;19679  =10      ;BRANCH ON SIGN SRC
                                        ;19680          ;10-----------------------------;
                                        ;19681  P2LF7:   CACHE_D[LONG],                  ;  WRITE DST-LONGWORD
                                        ;19682           ALU_D,N&Z_ALU.V&C_0,
U 08E2, 0001,003C,6180,3000,00D4,68E3   ;19683           SC_R[.F]                        ;  MAKE NEXT STATE HARMLESS
                                        ;19684                                           ;  NEED NEXT STATE TO CLEAR FPD
                                        ;19685          ;11-----------------------------;
                                        ;19686           ALU_D,R(SC)_ALU,N&Z_ALU.V&C_0,  ;  LOAD IN GEN. REGISTER, CLOCK PSL-CC
U 08E3, 0001,163C,0180,F8E8,2050,0130   ;19687           CLR.FPD,STATE7-4?,J/P2LF8       ;  RESET FPD-BIT OF PSL, TEST OVERFLOW
                                        ;19688          ;-----------------------------;
                                        ;19689  =0**    ;BRANCH ON OVERFLOW-BIT
                                        ;19690          ;0**--------------------------;
                                        ;19691  P2LF8:   FLUSH.IB,PC&VA_PC,              ;  GET READY FOR NEXT INSTRUCTION
U 0130, 2014,0038,0180,F801,4200,00AB   ;19692           J/IB.FILL
                                        ;19693          ;1**--------------------------;
U 0134, 0819,2030,6580,F800,0000,0B94   ;19694           ALU_Q.OR.K[.10],D_ALU           ;  INTEGER OVERFLOW, LOAD TRAP-VALUE
                                        ;19695  =;END    ;-----------------------------;
                                        ;19696           SET.V,                          ;  SET V-BIT
U 0B94, 0000,003C,3180,3C00,0020,0130   ;19697           ID[CES]_D,J/P2LF8               ;  WRITE CONTROL-REGISTER
                                        ;19698           ;-----------------------------;
```

```
;19699  .TOC    ""      Decimal string        : CVTPS""
;19700
;19701  ;CONVERT A PACKED STRING TO LEADING SEPARATE NUMERIC STRING
;19702  ;ALGORITHM:
;19703  ;       1. FIRST THE SPECIFIERS ARE EVALUATED AND REGISTERS INITIALIZED.
;19704  ;       SOME OF THIS CODE IS SHARED WITH THAT OF CVTPT-INSTRUCTION.
;19705  ;       ROUTINE STARTS AT "CVTPS.INIT".
;19706
;19707  ;       2. THE BASIC LOOP OF THE INSTRUCTION ("P2NL"),
;19708  ;       READS A PACKED BCD-BYTE FROM THE SOURCE STRING (STARTING AT THE
;19709  ;       TRAILING END), SPLITS IT INTO TWO ZONED BYTES, AND WRITES THE
;19710  ;       RESULTING WORD  INTO THE DST-STRING.
;19711  ;       TWO SLIGHTLY DIFFERENT PATHS ARE TAKEN THROUGH THE LOOP,
;19712  ;       DEPENDING ON WHETHER THE WORD RESULTING
;19713  ;       FROM A PACKED BCD-BYTE IS WORD ALIGNED ("P2NL1") OR NOT ("P2NL3").
;19714
;19715  ;       3.AFTER REACHING THE END OF BOTH SRC AND DST-STRINGS, THE
;19716  ;       "P2N.FIN"-ROUTINE IS EXECUTED TO SET CONDITION-CODES AND
;19717  ;       CLEAN UP THE GENERAL REGISTERS.
;19718
;19719  ;       4.IN CASE OF INTERRUPTS OR MEMORY-FAULTS, THE INITIAL STATE
;19720  ;       OF THE OPERANDS ARE SAVED IN GENERAL REGISTERS,
;19721  ;       AND THE INSTRUCTION IS RESTARTED AT "P2T.I00".
;19722  ;STORAGE:
;19723  ;       RC[T0] HAS SRC-LENGTH
;19724  ;       RC[T1] HAS DST-LENGTH-1
;19725  ;       RC[T2] HAS LEFT-OVER DIGIT
;19726  ;       RC[T3] HAS OVERFLOW DATA
;19727  ;       R1 HAS HIGH SRC-ADDRESS
;19728  ;       R3 HAS HIGH DST-ADDRESS
;19729  ;       ID[T0] HAS SRC-ADDRESS
;19730  ;       ID[T1] HAS DST-ADDRESS
;19731  ;       SRC-LENGTH AND DST-LENGTH ARE KEPT IN SC AND FE
;19732
;19733          ;INST.DEP. ALU FUNCTION IS "A-1"
;19734          ;OPCODE IS "08"
;19735          ;MNEMONIC IS "CVTPS"
;19736          ;THE SEQUENCE OF OPERANDS IS:
;19737          ;opcode  srclen.rw, srcaddr.ab, dstlen.rw, dstaddr.ab
;19738
;19739          ;STATE-REGISTER:
;19740          ;-----------------------------------------------------------------
;19741          ;INTRPT ;OVFLOW ;       ;ALIGN  ;       ;       ;SIGN   ;???    ;
;19742          ;       ;       ;       ;       ;       ;       ;       ;       ;
;19743          ;       ;       ;       ;       ;       ;       ;       ;       ;
;19744          ;       ;       ;       ;       ;       ;       ;       ;       ;
;19745          ;-----------------------------------------------------------------
```

```
                                ;19746          ;ENTER HERE FROM D-FORK WITH SRC-LENGTH IN Q,SRC-ADDRESS IN D
                                ;19747          ;-------------------------------;
                                ;19748  4C5:
                                ;19749  CVTPS.INIT:
                                ;19750          ALU_Q.0XT[WORD],                ;  ISOLATE SRC-LENGTH
                                ;19751          RC[T0]_ALU.RIGHT,               ;  SAVE IT IN RC 0
                                ;19752          Q_ALU.RIGHT,ID[T0]_D,           ;  SAVE SRC-ADDRESS
U 04C5, 0043,603C,C1C7,3D80,0000,0B95   ;19753          SGN/CLR.SD+SS                   ;  CLEAR SS FOR LATER BRANCHING
                                ;19754          ;-------------------------------;
                                ;19755          STATE_K[ZERO],                  ;  INITIALIZE STATE-REGISTER
                                ;19756          ALU_D+Q,R[R15]_ALU,             ;  GENERATE HIGH SRC-ADDRESS
U 0B95, 0C1D,0014,1980,FAF8,1404,60A7   ;19757          D_Q,J/P2T.IO                    ;  JOIN PACKED TO TRAILING ROUTINE
                                ;19758          ;-------------------------------;
                                ;19759          ;ENTER HERE FROM CVTPT-INITIALIZATION-ROUTINE,
                                ;19760          ;UPPER NIBBLE OF SIGN-BYTE IN D AND Q.
                                ;19761          ;LA HAS DST-ADDRESS
                                ;19762          ;-------------------------------;
                                ;19763  =10      ;BRANCH ON SIGN-NIBBLE
                                ;19764          ;10-------------------------------;
                                ;19765  P2S2:    STATE_K[ZERO],                  ;  CLEAR STATE-REGISTER
                                ;19766          VA_LA,                          ;  LOAD DST-ADDRESS
                                ;19767          QK/LEFT2,DK/RIGHT2,             ;  SHIFT DATA RIGHT AND LEFT
U 08EA, 0200,033C,1988,F800,1604,66FC   ;19768          C31?,J/P2S3                     ;  TEST DST-LENGTH
                                ;19769          ;11-------------------------------;
                                ;19770          STATE_STATE.OR.K[.2],           ;  SET MINUS-BIT
                                ;19771          QK/LEFT2,DK/RIGHT2,             ;  SHIFT NIBBLE RIGHT AND LEFT
U 08EB, 0200,033C,0988,F800,1604,26FC   ;19772          VA_LA,C31?                      ;  LOAD DST-ADDRESS, TEST DST-LENGTH
                                ;19773  =;END    ;-------------------------------;
                                ;19774  =0*      ;BRANCH ON ALU C31
                                ;19775  P2S3:    ;0*-------------------------------;
                                ;19776          ALU_D,LAB_R1&RC[T3]_ALU,        ;  SAVE OVERFLOW IN RC3
                                ;19777          EALU_FE,CLK.UBCC,               ;  CLOCK SRC-LENGTH
U 06FC, 0001,003C,0180,FB18,0010,6BF6   ;19778          J/P2NL
                                ;19779          ;1*-------------------------------;
                                ;19780          EALU_FE,CLK.UBCC,               ;  CLOCK SRC-LENGTH
                                ;19781          ALU_D.OR.K[.C0],D_ALU.RIGHT2,   ;  MAKE DATA ZONED WHILE SHIFTING
U 06FE, 0899,0230,D188,F800,0010,6BE6   ;19782          QK/LEFT2,ROR?,J/P2NI1
                                ;19783  =;END    ;-------------------------------;
```

```
;19784  .TOC    ""      Decimal string        : CVTPT''
;19785
;19786  ;CONVERT A PACKED STRING TO A TRAILING NUMERIC STRING
;19787  ;ALGORITHM:
;19788  ;       1. FIRST THE SPECIFIERS ARE EVALUATED AND REGISTERS INITIALIZED.
;19789  ;       SOME OF THIS CODE IS SHARED WITH THAT OF CVTPS-INSTRUCTION.
;19790  ;       ROUTINE STARTS AT 'CVTPT.INIT''.
;19791  ;       ONE OF THE OPERANDS IS A TABLE-ADDRESS. THIS IS ADDED TO
;19792  ;       THE SIGN-BYTE OF THE SRC-STRING TO FORM A POINTER
;19793  ;       INTO A TABLE OF SIGN-BYTES, TO GET THE SIGN-BYTE FOR THE
;19794  ;       DST-STRING('P2NI').
;19795
;19796  ;       2. THE BASIC LOOP OF THE INSTRUCTION ('P2NL''),
;19797  ;       READS A PACKED BCD-BYTE FROM THE SOURCE STRING (STARTING AT THE
;19798  ;       TRAILING END), SPLITS IT INTO TWO ZONED BYTES, AND WRITES THE
;19799  ;       RESULTING WORD  INTO THE DST-STRING.
;19800  ;       TWO SLIGHTLY DIFFERENT PATHS ARE TAKEN THROUGH THE LOOP,
;19801  ;       DEPENDING ON WHETHER THE WORD RESULTING
;19802  ;       FROM A PACKED BCD-BYTE IS WORD ALIGNED ('P2NL1'') OR NOT ('P2NL3'').
;19803
;19804  ;       3.AFTER REACHING THE END OF BOTH SRC AND DST-STRINGS, THE
;19805  ;       'P2N.FIN''-ROUTINE IS EXECUTED TO SET CONDITION-CODES AND
;19806  ;       CLEAN UP THE GENERAL REGISTERS.
;19807
;19808  ;       4.IN CASE OF INTERRUPTS OR MEMORY-FAULTS, THE INITIAL STATE
;19809  ;       OF THE OPERANDS ARE SAVED IN GENERAL REGISTERS,
;19810  ;       AND THE INSTRUCTION IS RESTARTED AT 'P2T.I00''.
;19811
;19812  ;STORAGE:
;19813  ;       R1=SRC-ADDRESS (HIGH END OF STRING)
;19814  ;       R3=DST-ADDRESS  (HIGH)
;19815  ;       FE AND SC CONTAIN THE TWO LENGTHS, INITIALLY
;19816  ;       FE=SRC-LENGTH,SC=DST-LENGTH+1
;19817  ;       RC[T2] STORES LEFTOVER BYTES BETWEEN PASSES THROUGH LOOP.
;19818  ;       RC[T3] IS USED TO STORE OVERFLOW DATA.
;19819  ;       RC[T5] IS USED TO STORE TABLE-ADDRESS
;19820
;19821  ;OP-CODE IS '24''
;19822  ;INSTRUCTION DEPENDENT ALU-FUNCTION IS 'A-1''
;19823  ;INSTRUCTION FORMAT:
;19824  ;opcode  srclen.rw, srcaddr.ab, tbladdr.ab, dstlen.rw, dstaddr.ab
;19825
;19826  ;STATE-REGISTER IS USED FOR STATUS.
;19827          ;STATE-REGISTER:
;19828          ;-----------------------------------------------------------
;19829          ;INTRPT ;OVFLOW ;       ;ALIGN  ;       ;       ;SIGN   ;???    ;
;19830          ;       ;       ;       ;       ;       ;       ;       ;       ;
;19831          ;       ;       ;       ;       ;       ;       ;       ;       ;
;19832          ;       ;       ;       ;       ;       ;       ;       ;       ;
;19833          ;-----------------------------------------------------------
```

```
                                ;19834          ;ENTER HERE FROM D-FORK WITH SRC-LENGTH IN Q, SRC-ADDRESS IN D
                                ;19835          ;-------------------------------;
                                ;19836  442:    ALU_Q.OXT[WORD],RC[TO]_ALU.RIGHT,
                                ;19837          Q_ALU.RIGHT,ID[TO]_D,           ;  SAVE SRC-ADDRESS
U 0442, 0043,603C,C1C7,3D80,0000,048C   ;19838          SGN/CLR.SD+SS
                                ;19839          ;-------------------------------;
                                ;19840  =00*****
                                ;19841          STATE_K[.1],                    ;  CLEAR STATE-REGISTER
                                ;19842          ALU_D+Q,R[R15]_ALU,             ;  GENERATE HIG SRC-ADDRESS
U 048C, 0C1D,0015,0580,FAF8,1404,647E   ;19843          D_Q,CALL,J/ASPC                 ;  EVALUATE TABLE-ADDRESS
                                ;19844          ;-------------------------------;
                                ;19845  =11*****
                                ;19846          ALU_D,RC[T5]_ALU,D_Q,           ;  SAVE TABLE-ADDRESS
U 04EC, 0C01,003C,0180,F9A8,0000,00A7   ;19847          J/P2T.I0
                                ;19848  =;END   ;-------------------------------;
                                ;19849  =010**1*
                                ;19850  P2T.I0: ALU_D.AND.K[.FFF0],
                                ;19851          RC[T3]_ALU,                     ;  CLEAR OVERFLOW-REGISTER
                                ;19852          N&Z_ALU.V&C_0,                  ;  CLOCK LENGTH
U 00A7, 0C19,0035,6D80,F998,0050,037E   ;19853          D_Q,CALL,J/SPEC                 ;  EVALUATE DST-LENGTH
                                ;19854          ;-------------------------------;
                                ;19855  =011**1*
                                ;19856          ALU_Q-K[.1],SC_ALU,             ;  SAVE SRC-LENGTH-1 IN SC
U 00B7, 0019,2001,0580,F800,0082,047E   ;19857          CALL,J/ASPC                     ;  EVALUATE DST-ADDRESS
                                ;19858          ;-------------------------------;
                                ;19859  =111**1*
                                ;19860  P2T.I00:        ;REENTER HERE AFTER A FAULT
                                ;19861          ;-------------------------------;
                                ;19862          ALU_Q.ANDNOT.K[.1F],            ;  STRIP OFF LENGTH
                                ;19863          N_MUX.Z_TST,WORD,               ;  CLOCK EXTRA BITS
                                ;19864          LAB_R[RT5],                     ;  GET HIGH ADDRESS
U 00F7, 0019,6024,8D87,FA78,0030,0622   ;19865          SGN/CLR.SD+SS,J/P2NI.01         ;  CLEAR SS FOR BRANCHING
                                ;19866  =;END   ;-------------------------------;
```

```
                                        ;19867  =0***
                                        ;19868  P2NI.01:
                                        ;19869          FE_SC,ALU_Q.OXT[BYTE]-K[.1],    ;  SAVE DST-LENGTH IN RC1
                                        ;19870          CLR.UBCC,                       ;  CLOCK IT INTO ALU CC
                                        ;19871          Q_ALU,SC_ALU,RC[T1]_ALU,
U 0622, 001B,A901,05C0,F988,0192,0748   ;19872          IR2-1?,CALL,J/SET.FPD.P2N       ;  SET FIRST PART DONE
                                        ;19873          ;1***---------------------------;
                                        ;19874          ALU_LA,VAK/LOAD,                ;  LOAD SRC-ADDRESS
                                        ;19875          LC_RC[T5]&R1_ALU,               ;  GET TABLE-ADDRESS
                                        ;19876          ID[FPDA]_D,                     ;  LOAD .33 IN FPDA (RESTART ADDRESS)
U 062A, 0000,003C,B580,3FA8,0280,CB98   ;19877          SC_SC+1,J/P2NI                  ;  INCREMENT DST-LENGTH
                                        ;19878  =;END   ;---------------------------;
                                        ;19879  =0*     ;BRANCH ON BIT 2 OF OPCODE (IR<1>=0)
                                        ;19880          ;0*---------------------------;
                                        ;19881  SET.FPD.P2N:
                                        ;19882          ALU_D+Q+1,R[R3]_ALU,ID[T1]_D,   ;  LOAD DST-ADDRESS, SAVE IT
U 0748, 001D,1A10,C580,3E98,0080,C0F9   ;19883          SC_SC+1,PSL.CC?,J/BCD.FPD       ;  TEST FOR ILLEGAL LENGTHS
                                        ;19884          ;1*---------------------------;
                                        ;19885          ALU_D+Q,R[R3]_ALU,ID[T1]_D,     ;  INITIALIZE DST-ADDRESS
U 074A, 001D,1A14,C580,3E98,0080,C0F9   ;19886          SC_SC+1,PSL.CC?,J/BCD.FPD       ;  SET FIRST PART DONE
                                        ;19887  =;END   ;---------------------------;
                                        ;19888  P2NI:   D[BYTE]_CACHE,LAB_R[R3],        ;  READ SIGN-BYTE
U 0B98, 0000,893C,0180,4218,0000,074C   ;19889          IR2-1?                          ;  TEST FOR SEPARATE OR TRAILING
                                        ;19890          ;---------------------------;
                                        ;19891  =0*     ;BRANCH ON BIT 2 OF OP-CODE (IR<1>=0)
                                        ;19892          ;0*---------------------------;
                                        ;19893          ALU_D,AND.K[.F0],N_AMX.Z_TST,   ;  CLOCK HIGH NIBBLE
                                        ;19894          LC_RC[T1],                      ;  GET DST-LENGTH
                                        ;19895          D_ALU,Q_ALU,BCDSGN?,            ;  TEST SIGN OF SOURCE
U 074C, 0819,0F34,CDC0,F908,0030,08EA   ;19896          J/P2S2                          ;  LEADING SEPARATE STRING
                                        ;19897          ;1*---------------------------;
                                        ;19898          SC_K[.8],                       ;  FOR LATER SHIFTING
                                        ;19899          ALU_D.OXT[BYTE]+LC,VAK/LOAD,    ;  LOAD INDEXED TABLE-ADDRESS
U 074E, 0013,8F14,0180,F800,0284,68F2   ;19900          BCDSGN?                         ;  TEST SRC-SIGN
                                        ;19901  =;END   ;---------------------------;
                                        ;19902
                                        ;19903  ; ************************************************
                                        ;19904  ; * Patch no. 092, PCS 074E trapped to WCS 119C *
                                        ;19905  ; ************************************************
```

```
                                ;19906          ;-------------------------------;
                                ;19907  =10     ;BRANCH ON SIGN-NIBBLE
                                ;19908          ;10-----------------------------;
                                ;19909          ALU_D.OXT[BYTE].AND.K[.FFF0],   ; STRIP OFF SIGN-NIBBLE
                                ;19910          N_ARX.Z_TST,                    ; CLOCK Z-BIT OF PSL
                                ;19911          LC_RC[TT],                      ; GET DST-LENGTH
                                ;19912          D[BYTE]_CACHE,Q_0,              ; READ TABLE-ENTRY
U 08F2, 001B,8034,6DF8,4108,0030,0B99   ;19913  J/P2T0
                                ;19914          ;11-----------------------------;
                                ;19915          STATE_STATE+1,                  ; SET MINUS-BIT OF STATE
                                ;19916          ALU_D.OXT[BYTE].AND.K[.FFF0],   ; STRIP OFF SIGN-NIBBLE
                                ;19917          N_ARX.Z_TST,                    ; CLOCK Z-BIT OF PSL
                                ;19918          LC_RC[TT],                      ; GET DST-LENGTH
U 08F3, 001B,8034,6DF8,4108,1430,CB99   ;19919  D[BYTE]_CACHE,Q_0                ; READ TABLE-ENTRY
                                ;19920  =;END    ;-------------------------------;
                                ;19921  P2T0:   D_DAL.SC,Q_D,                   ; STORE RESULT IN D AND Q
                                ;19922          ALU_0+LC+1,SC_ALU,              ; STORE DST-LENGTH IN SC
U 0B99, 0D13,0010,01E0,F800,0092,0B9A   ;19923  CLK.UBCC
                                ;19924          ;-------------------------------;
                                ;19925          D_Q,Q_D,                        ; D GETS DATA IN BYTE 0
                                ;19926          SC_SC+1,                        ; INCREMENT DST-COUNT
                                ;19927          ALU_R[R3],VAK/LOAD,             ; LOAD DST-ADDRESS
U 0B9A, 0C00,013C,01E0,FA18,0280,C7FC   ;19928  Z?,J/P2NI0                      ; TEST DST-LENGTH
                                ;19929          ;-------------------------------;
                                ;19930  =0      ;BRANCH ON ALU Z-BIT
                                ;19931          ;0------------------------------;
                                ;19932  P2NI0:  EALU_FE,CLK.UBCC,               ; CLOCK SRC-COUNT
U 07FC, 0000,023C,0180,F800,0010,6BE6   ;19933  ROR?,J/P2NI1                    ; TEST DST-ADDRESS FOR WORD ALIGNMENT
                                ;19934          ;1------------------------------;
                                ;19935  P2TI5:  ALU_0(A),LAB_R1&RC[T2]_ALU,     ; NO LEFT-OVER DIGIT
                                ;19936          EALU_FE,CLK.UBCC,               ; CLOCK SRC-LENGTH
U 07FD, 0003,003C,0180,FB10,0010,6BF6   ;19937  J/P2NL                          ; JUMP TO MAIN-LOOP
                                ;19938  =;END    ;-------------------------------;
                                ;19939  =110    ;BRANCH ON LOW BIT OF DST-ADDRESS (+1)
                                ;19940          ;110----------------------------;
                                ;19941  P2NI1:  CACHE_D[BYTE],                  ; WRITE FIRST BYTE
                                ;19942          LAB_R[R1],                      ; GET SRC-ADDRESS
U 0BE6, 0000,803C,0580,3208,0084,ABF6   ;19943  SC_SC-K[.1],J/P2NL               ; NO LEFT-OVER NIBBLE
                                ;19944          ;1T1----------------------------;
                                ;19945          ALU_0+LB+1,R[R3]_ALU,LONG,      ; ADJUST DST-ADDRESS
                                ;19946          STATE_STATE.OR.K[.10],          ; SET UNALIGNMENT-BIT
U 0BE7, 000F,0910,6580,FA98,1404,27C0   ;19947  IR2-1?                          ; TEST BIT 2 OF OP-CODE
                                ;19948  =;END    ;-------------------------------;
                                ;19949  =0*     ;BRANCH ON BIT 2 OF OP-CODE
                                ;19950          ;0*-----------------------------;
                                ;19951          ALU_Q.OXT[WORD].OR.K[.3030],    ; LEADING SEPARATE NUMERIC
                                ;19952          LAB_R1&RC[T2]_ALU,
U 07C0, 001B,6030,C980,FB10,0000,0BF6   ;19953  J/P2NL
                                ;19954          ;1*-----------------------------;
                                ;19955          ALU_Q.OXT[WORD].OR.K[.30],      ; MAKE IT ZONED
                                ;19956          LAB_R1&RC[T2]_ALU,              ; SAVE SIGN-BYTE IN RC[T2]
U 07C2, 001B,6030,7980,FB10,0000,0BF6   ;19957  J/P2NL                          ; JOIN MAIN LOOP
                                ;19958  =;END    ;-------------------------------;
```

```
                                ;19959  ;MAIN LOOP FOR CONVERTING PACKED TO NUMERIC.
                                ;19960  ;IT READS A BYTE AND WRITES A WORD IN EACH PASS,
                                ;19961  ;EXCEPT POSSIBLE IN PASSES AFTER DST-LENGTH HAS REACHED 0.
                                ;19962  ;RC[T2] IS USED TO STORE LEFT-OVER BYTE IF DST-ADDRESS IS NOT
                                ;19963  ;ALIGNED. RC[T3] HAS OVERFLOW DATA.
                                ;19964  ;STATE DETERMINES WHETHER DST-ADDRESS WAS WORD-ALIGNED,
                                ;19965  ;AND HAS A SIGN-BIT
                                ;19966  ;EALU CC REFLECTS SRC-LENGTH-2, WHICH IS STORED IN FE.
                                ;19967  ;SC HAS DST-LENGTH+1(IN BYTES).
                                ;19968
                                ;19969          ;-------------------------------;
                                ;19970  =110    ;BRANCH ON INTERRUPT PENDING
                                ;19971          ;110------------------------------;
                                ;19972  P2NL:   ALU_LA-K[.1],R[R1]_ALU,
                                ;19973          VAX7LOAD,                       ;  LOAD SRC-ADDRESS
                                ;19974          STATE_STATE.OR.K[.1],           ;  CLR 1.TIME FLAG
                                ;19975          EALU?,                          ;  TEST SRC-LENGTH
U 08F6, 0018,1200,0580,FA88,1604,2496   ;19976          J/P2NL00
                                ;19977          ;111-----------------------------;
U 08F7, 0000,003C,4180,F800,1404,6033   ;19978          STATE_K[.80],J/SAVE.BCD         ;  SET INTERRUPT-BIT, SAVE CONTEXT
                                ;19979  =;END   ;-------------------------------;
                                ;19980  =011*   ;BRANCH ON EALU N-BIT (SS IS 0)
                                ;19981          ;011*----------------------------;
                                ;19982  P2NL00: D[BYTE]_CACHE,                  ;  READ NEXT SRC-BYTE
                                ;19983          FE_SC-K[.2],CLK.UBCC,           ;  UPDATE DST-LENGTH, CLOCK IT
U 0496, 0000,963C,0980,4000,0195,A2AC   ;19984          SC_FE,BEN/STATE7-4,J/P2NL0      ;  TEST ALIGNMENT
                                ;19985          ;111*----------------------------;
                                ;19986          D_0,FE_SC-K[.2],                ;  UPDATE DST-LENGTH
                                ;19987          ALU_LA,R[R1]_ALU,LONG,          ;  RESTORE SRC-ADDRESS
                                ;19988          SC_FE,                          ;  NO MORE INPUT
U 049E, 0F00,163C,0980,FA88,0195,A2AC   ;19989          CLR.UBCC,BEN/STATE7-4           ;  UPDATE DST-LENGTH, TEST ALIGNMENT
                                ;19990  =;END   ;-------------------------------;
                                ;19991  =1*0    ;BRANCH ON ALIGNMENT-BIT OF STATE (BIT 4)
                                ;19992          ;1*0-----------------------------;
                                ;19993  P2NL0:  ALU_D.AND.K[.F],N_AMX.Z_TST,    ;  CLOCK Z-BIT
                                ;19994          Q_ALU.LEFT2,                    ;  SHIFT LOW NIBBLE LEFT
U 02AC, 0679,0034,61C0,F800,0030,0B9C   ;19995          DR/RIGHT,J/P2NL1                ;  SHIFT HIGH NIBBLE RIGHT
                                ;19996          ;1*1-----------------------------;
                                ;19997          ALU_D.OXT[BYTE].AND.K[.FFF0],
                                ;19998          Q_ALU,LA_RA[R3],                ;  SAVE HIGH NIBBLE
U 02AD, 001B,8034,6DC0,F898,0030,0BA5   ;19999          N_AMX.Z_TST,J/P2NL3             ;  CLOCK Z-BIT
                                ;20000  =;END   ;-------------------------------;
```

```
                                ;20001  P2NL1:  ;COME HERE IF DST-ADDRESS IS WORD-ALIGNED
                                ;20002          ;-------------------------------;
                                ;20003          QK/LEFT2,
                                ;20004          ALU_D.OXT[BYTE].AND.K[.7E],
                                ;20005          DK/SHF,                         ;  SHIFT LOW NIBBLE LEFT
                                ;20006          SHF/RIGHT2,N_AMX.Z_TST,
                                ;20007          LA_RA[R3],                      ;  GET DST-ADDRESS
U 0B9C, 089B,9234,F988,F898,0030,05D2   ;20008  BYTE,BEN/EALU                    ;  TEST DST-LENGTH
                                ;20009          ;-------------------------------;
                                ;20010  =001*   ;BRANCH ON EALU Z AND N-BITS
                                ;20011          ;001*-----------------------------;
                                ;20012          ALU_LA-K[.2],LONG,              ;  UPDATE ADDRESS
                                ;20013          VAK/LOAD,R[R3]_ALU,             ;  LOAD DST-ADDRESS
                                ;20014          QK/LEFT2,DK/RIGHT,              ;  KEEP SHIFTING LOW NIBBLE
U 05D2, 0618,0000,0988,FA98,0200,0B9D   ;20015  J/P2NL2
                                ;20016          ;011*-----------------------------;
                                ;20017          FE_SC-K[.1],SC_FE,CLK.UBCC,     ;  UPDATE SRC-LENGTH
                                ;20018          ALU_LA-K[.1],
                                ;20019          VAK/LOAD,R[R3]_ALU,             ;  LOAD DST-ADDRESS
U 05D6, 0018,0000,0590,FA98,0395,ABAA   ;20020  QK/RIGHT2,J/P2NL10
                                ;20021
                                ;20022  ; ************************************************
                                ;20023  ; * Patch no. 057, PCS 05D6 trapped to WCS 1183 *
                                ;20024  ; ************************************************
                                ;20025
                                ;20026          ;101*-----------------------------;
                                ;20027          FE_SC-K[.1],SC_FE,CLK.UBCC,     ;  UPDATE SRC-LENGTH
                                ;20028          LC_RC[T3],                      ;  GET PREVIOUS OVERFLOW
U 05DA, 081D,0030,0580,F918,0195,ABA6   ;20029  ALU_D.OR.Q,D_ALU,J/P2NL33        ;  ALL GOES TO OVERFLOW
                                ;20030  =;END   ;-------------------------------;
                                ;20031  P2NL2:  ALU_D.OR.K[.3030],
U 0B9D, 0819,0030,C988,F800,0000,0B9E   ;20032  D_ALU,QK/LEFT2                   ;  MAKE ZONED DIGITS
                                ;20033          ;-------------------------------;
                                ;20034          INTRPT.STROBE,
U 0B9E, 081D,0030,0180,F888,4000,0BA4   ;20035  ALU_D.OR.Q,D_ALU,LA_RA[R1]        ;  ASSEMBLE THE WORD
                                ;20036          ;-------------------------------;
                                ;20037          CACHE_D[WORD],FE_SC-K[.1],      ;  WRITE WORD
                                ;20038          SC_FE,CLK.UBCC,                 ;  UPDATE SRC-LENGTH
                                ;20039          BEN/INTERRUPT,
U 0BA4, 0000,4E3C,0580,3000,0195,ABF6   ;20040  J/P2NL                          ;  LOOP BACK
                                ;20041          ;-------------------------------;
```

```
                                ;20042  P2NL3:  ;COME HERE IF DST-ADDRESS IS NOT ALIGNED WITH SRC
                                ;20043          ;-------------------------------;
                                ;20044          ALU_D.OXT[BYTE].AND.K[.F],
                                ;20045          N_AMX.Z_TST,D_ALU,              ;  GET LOW NIBBLE IN D
                                ;20046          BYTE,LC_RC[T2],                 ;  GET PREVIOUS NIBBLE
U 0BA5, 081B,9234,6180,F910,0030,05E2   ;20047          BEN/EALU                        ;  TEST DST-LENGTH
                                ;20048          ;-------------------------------;
                                ;20049  =001*   ;BRANCH ON EALU Z AND N-BITS (SS IS CLEAR)
                                ;20050          ;001*---------------------------;
                                ;20051          ALU_LA-K[.2],VAK/LOAD,
                                ;20052          R[R3]_ALU,LONG,                 ;  LOAD AND UPDATE DST-ADDRESS
U 05E2, 0018,0000,0988,FA98,0200,0BA8   ;20053          QK/LEFT2,J/P2NL4                ;  SHIFT HIGH NIBBLE
                                ;20054          ;011*---------------------------;
                                ;20055          FE_SC-K[.1],SC_FE,CLK.UBCC,     ;  UPDATE SRC-LENGTH
U 05E6, 001D,0030,05C0,F800,0195,ABAC   ;20056          ALU_D.OR.Q,Q_ALU,J/P2NL30       ;  CURRENT DATA IS OVERFLOW
                                ;20057          ;101*---------------------------;
                                ;20058          FE_SC-K[.1],SC_FE,CLK.UBCC,     ;  UPDATE SRC-LENGTH
                                ;20059          ALU_D.OR.Q,
U 05EA, 081D,0030,0580,F918,0195,ABA6   ;20060          D_ALU,LC_RC[T3]                 ;  GET OVERFLOW
                                ;20061  =;END   ;-------------------------------;
                                ;20062  P2NL33: ALU_D.OR.LC,CLK.UBCC,
                                ;20063          LAB_R1&RC[T3]_ALU,EALU?,        ;  SAVE OVERFLOW IN RC[T1]
U 0BA6, 0011,1230,0180,FB18,0010,0526   ;20064          J/P2NL34                        ;  TEST SRC-LENGTH
                                ;20065          ;-------------------------------;
                                ;20066  P2NL4:  ALU_D.OR.LC,D_ALU,QK/LEFT2,     ;  ASSEMBLE WORD
                                ;20067          LA_RA[R1],                      ;  GET SRC-ADDRESS READY
                                ;20068          FE_SC-K[.1],CLK.UBCC,           ;  DECREMENT SRC-LENGTH
                                ;20069          SC_FE,
U 0BA8, 0811,0030,0588,F888,4195,ABA9   ;20070          INTRPT STROBE
                                ;20071          ;-------------------------------;
                                ;20072          CACHE_D[WORD],                  ;  WRITE WORD
                                ;20073          ALU_Q.OR.K[.3030],RC[T2]_ALU,   ;  SAVE LEFT-OVER NIBBLE
                                ;20074          BEN/INTERRUPT,                  ;  TEST FOR PENDING INTERRUPTS
U 0BA9, 0019,6E30,C980,3190,0000,0BF6   ;20075          J/P2NL
                                ;20076          ;-------------------------------;
```

```
                                    ;20077  ;THIS PAGE HAS CLEANUP ROUTINES FOR P2N-INSTRUCTION.
                                    ;20078
                                    ;20079  P2NL10: ;DST-LENGTH IS 0,DST-ADDRESS IS ALIGNED,WRITE ONE MORE BYTE
                                    ;20080          ;-------------------------------;
                                    ;20081          ALU_Q.OR.K[.C0],
                                    ;20082          D_ALU.RIGHT2,                   ;  MAKE DATA ZONED AS WE SHIFT
                                    ;20083          Q_D,
U 0BAA, 0899,2030,D1E0,F800,4000,0BAB ;20084          INTRPT.STROBE
                                    ;20085          ;-------------------------------;
                                    ;20086  P2NL11: CACHE_D[BYTE],                  ;  WRITE BYTE, LOOP BACK
                                    ;20087          ALU_Q,SC_K[ZERO],               ;  CLEAR DST-LENGTH
                                    ;20088          LAB_R1&RC[T3]_ALU,              ;  GET SRC-ADDRESS AND OVERFLOW
U 0BAB, 0001,AE3C,1980,3318,0084,6BF6 ;20089          BEN/INTERRUPT,J/P2NL
                                    ;20090          ;-------------------------------;
                                    ;20091  P2NL30: ;DST-LENGTH IS 0,DST-ADDRESS IS NOT ALIGNED,WRITE ONE MORE BYTE
                                    ;20092          ;-------------------------------;
U 0BAC, 0810,0038,7180,F910,0084,6BAD ;20093          ALU_RC[T2],D_ALU,SC_K[.FFF8]    ;  GET PREVIOUS NIBBLE
                                    ;20094          ;-------------------------------;
                                    ;20095          ALU_LA-K[.1],
                                    ;20096          VAK/LOAD,R[R3]_ALU,             ;  UPDATE AND LOAD DST-ADDRESS
U 0BAD, 0D18,0000,0580,FA98,0200,0BAE ;20097          D_DAL.SC                        ;  SHIFT Q INTO PLACE
                                    ;20098          ;-------------------------------;
                                    ;20099          ALU_0(A),RC[T2]_ALU,            ;  CLEAR LEFT-OVER NIBBLE
                                    ;20100          INTRPT.STROBE,
U 0BAE, 0003,003C,0180,F990,4000,0BAB ;20101          J/P2NL11
                                    ;20102          ;-------------------------------;
                                    ;20103  =011*   ;BRANCH ON EALU N-BIT (SS IS CLEAR)
                                    ;20104          ;011*---------------------------;
                                    ;20105  P2NL34: ALU_LA-K[.1],R[R1]_ALU,
                                    ;20106          VAK/LOAD,                       ;  LOAD SRC-ADDRESS
                                    ;20107          STATE_STATE.OR.K[.1],           ;  CLR 1.TIME FLAG
U 0526, 0018,0000,0580,FA88,1604,2496 ;20108          J/P2NL00
                                    ;20109          ;111*---------------------------;
                                    ;20110          ALU_RC[T2],                     ;  GET LEFT-OVER WORD
                                    ;20111          Q_ALU,STATE7-4?,                ;  TEST ALIGNMENT
U 052E, 0010,1638,01C0,F910,0000,02F4 ;20112          J/P2N.FIN                       ;  FINISH UP
                                    ;20113  =;END   ;-------------------------------;
```

```
                                ;20114          ;--------------------------------;
                                ;20115  =1*0    ;BRANCH ON ALIGNMENT-BIT OF STATE (BIT 4).
                                ;20116          ;1*0-----------------------------;
                                ;20117  P2N.FIN:
                                ;20118          ALU_RC[T3],Q_ID[T1],
U 02F4, 0010,0038,C5F0,2D18,0010,0BB1 ;20119    CLK.UBCC,J/P2NL35               ; CLOCK OVERFLOW
                                ;20120          ;1*1-----------------------------;
                                ;20121          ALU_Q.ANDNOT.K[.3030],          ; ISOLATE DIGIT FROM ZONE
U 02F5, 0019,2024,C9C0,F800,0000,0BB0 ;20122    Q_ALU
                                ;20123  =;END   ;--------------------------------;
                                ;20124          LC_RC[T3],                      ; GET PREVIOUS OVERFLOW-DATA
                                ;20125          ALU_Q.OR.LC,CLK.UBCC,WORD,      ; CLOCK OVERFLOW
U 0BB0, 0011,6030,C5F0,2D18,0010,0BB1 ;20126    Q_ID[T1]                        ; GET DST-SIGN-ADDRESS
                                ;20127          ;--------------------------------;
                                ;20128  P2NL35: VA_Q,Q_ID[T0],                  ; LOAD IT IN VA, GET SRC-ADDRESS
U 0BB1, 0001,213C,C1F0,2C00,0200,0804 ;20129    Z?                              ; TEST FOR OVERFLOW
                                ;20130          ;--------------------------------;
                                ;20131  =0      ;BRANCH ON ALU Z-BIT
                                ;20132          ;0-----------------------------;
U 0804, 0000,003C,3180,F800,1404,2805 ;20133    STATE_STATE.OR.K[.40]           ; SET OVERFLOW-BIT OF STATE
                                ;20134          ;1-----------------------------;
                                ;20135          ALU_0(A),R[R0]_ALU,N_VMX.Z_TST, ; CLEAR R0, CLEAR PSL-N-BIT
U 0805, 0003,093C,2180,FA80,0030,07D0 ;20136    K[.T4],IR2-1?                   ; SEPARATE OR TRAILING NUMERIC?
                                ;20137  =;END   ;--------------------------------;
                                ;20138  =0*     ;BRANCH ON BIT 2 OF OP-CODE
                                ;20139          ;0*----------------------------;
                                ;20140          ALU_0+K[.14]+1,
                                ;20141          D_ALU.LEFT,SI/MUL-,             ; GENERATE CONSTANT .2B
U 07D0, 083B,1710,2380,F800,0000,091D ;20142    STATE3-0?,J/P2NL38              ; TEST SIGN-BIT
                                ;20143          ;1*----------------------------;
                                ;20144          STATE_STATE.OR.K[.10],          ; USE THIS BIT IN FINISH-ROUTINE
U 07D2, 0000,173C,6580,F800,1404,288D ;20145    STATE3-0?,J/FINI1               ; TEST SIGN-BIT
                                ;20146  =;END   ;--------------------------------;
                                ;20147  =1101   ;BRANCH ON SIGN-BIT
                                ;20148          ;1101--------------------------;
                                ;20149  P2NL38: STATE_STATE.OR.K[.10],          ; USE THIS BIT IN FINISH-ROUTINE
U 091D, 0000,973C,6580,3000,1404,288D ;20150    CACHE_D[BYTE],STATE3-0?,J/FINI1 ; WRITE SIGN-BYTE, TEST SIGN-BIT
                                ;20151          ;1111--------------------------;
U 091F, 0819,0014,0980,F800,0000,091D ;20152    ALU_D+K[.2],D_ALU,J/P2NL38      ; GENERATE .2D
                                ;20153  =;END   ;--------------------------------;
```

```
;20154  .TOC     ""      Decimal string        : CVTTP""
;20155
;20156  ;CONVERT TRAILING NUMERIC STRING TO PACKED
;20157  ;ALGORITHM:
;20158  ;       1. STARTING AT "CVTTP.INIT" THE SPECIFIERS ARE EVALUATED
;20159  ;       AND THE REGISTERS INITIALIZED. FIRST PART DONE IS SET.
;20160  ;       THE SIGN-BYTE IS ADDED TO A TABLE-ADDRESS, TO GET ADDRESS OF
;20161  ;       DEST-SIGN-BYTE ("T2P.I1").
;20162
;20163  ;       2. THE MAIN LOOP CONSISTS OF TWO PARTS.
;20164  ;       FIRST BYTES ARE READ FROM THE SRC-STRING, AND PACKED INTO A
;20165  ;       A LONGWORD ("T2P.L0"), UNTIL THE LONGWORD IS COMPLETE OR THE SRC-STRING
;20166  ;       IS EXHAUSTED. THEN THE LONGWORD IS WRITTEN INTO THE
;20167  ;       DST-STRING ("T2P.LONG0"), AND THE FIRST STEP IS REPEATED.
;20168  ;       WHILE READING ZONED BYTES FROM THE SRC-STRING, THEY ARE
;20169  ;       CHECKED FOR CORRECT FORMAT, AND A RESERVED OPERAND FAULT IS TAKEN IF
;20170  ;       THEY ARE NOT IN THE RANGE 30-39.
;20171
;20172  ;       3. FINALLY, AFTER REACHING THE END OF BOTH STRINGS, ("T2P.FIN1"),
;20173  ;       THE CONDITION CODES ARE SET AND THE GENERAL REGISTERS ARE LOADED.
;20174  ;       NOTE THAT EVEN THOUGH THE SIGN-BYTE COMES OUT OF THE TABLE,
;20175  ;       WE MAY CHANGE A -0 TO A +0, AND THE PREFERRED SIGNS (C OR D)
;20176  ;       ARE ALWAYS GENERATED.
;20177
;20178  ;       IF AN INTERRUPT OR MEMORY FAULT OCCUR, THE ORIGINAL
;20179  ;       OPERANDS ARE SAVED IN GENERAL REGISTERS ("BCD.SAVE"),
;20180  ;       AND THE INSTRUCTION IS RESTARTED AT "T2P.I1".
;20181
;20182  ;       THE OP-CODE IS 26
;20183  ;       THE SEQUENCE OF OPERANDS IS:
;20184  ;       opcode  srclen.rw, srcaddr.ab, tbladdr.ab, dstlen.rw, dstaddr.ab
;20185
;20186  ;STORAGE:
;20187  ;       SRC-LENGTH IS STORED IN RC6, AND SAVED IN RC0
;20188  ;       SRC-ADDRESS IS SAVED IN T0, STORED IN R1
;20189  ;       DST-LENGTH IS SAVED IN RC1,STORED IN R2
;20190  ;       DST-ADDRESS IS SAVED IN T1,STORED IN R3
;20191  ;       TABLE-ADDRESS IS SAVED IN RC5
;20192
;20193  ;STATE-REGISTER:
;20194  ;-------------------------------------------------------------------;
;20195  ;INTRPT ;OVFLOW ;       ;END OF ;       ;1.WRIT ;SIGN   ;       ;
;20196  ;       ;       ;       ;SRC    ;       ;       ;       ;       ;
;20197  ;       ;       ;       ;       ;       ;       ;       ;       ;
;20198  ;       ;       ;       ;       ;       ;       ;       ;       ;
;20199  ;-------------------------------------------------------------------;
;20200
```

```
                                ;20201  443:
                                ;20202          ;ENTER HERE FROM D-FORK WITH SRC-LENGTH IN Q, DST-ADDRESS IN D
                                ;20203          ;--------------------------------;
                                ;20204  CVTPT.INIT:
                                ;20205          ALU_Q.AND.K[.FFE0],             ;  TEST SRC-LENGTH
U 0443, 0019,2034,A180,F800,0050,0588   ;20206          N&Z_ALU.V&C_0                   ;  DIVIDE IT BY 2
                                ;20207          ;--------------------------------;
                                ;20208  =00*****
                                ;20209          ALU_Q.OXT[BYTE],CLK.UBCC,       ;  CLOCK LENGTH FOR LATER USE
                                ;20210          RC[T0]_ALU,SC_ALU,              ;  SAVE SRC-LENGTH
                                ;20211          D_ALU,ID[T0]_D,                 ;  SAVE SRC-ADDRESS
U 0588, 0803,A03D,C180,3D80,0092,047E   ;20212          CALL,J/ASPC                     ;  EVALUATE TABLE-ADDRESS
                                ;20213          ;--------------------------------;
                                ;20214  =11*****
                                ;20215          SC_SC-K[.1],
                                ;20216          ALU_D,RC[T5]_ALU,               ;  STORE ADDRESS IN RC5
U 05E8, 0001,003C,0580,F9A8,0084,A282   ;20217          J/T2P.I0
                                ;20218          ;--------------------------------;
                                ;20219  =000**1*
                                ;20220  T2P.I0: ALU_0-Q,RC[T6]_ALU,             ;  INITIALIZE RC6 WITH NEGATIVE LENGTH
U 0282, 001F,0001,0180,F9B0,0000,037E   ;20221          CAL[,J/SPEC                     ;  EVALUATE DST-LENGTH
                                ;20222          ;--------------------------------;
                                ;20223  =001**1*
                                ;20224          ALU_D.AND.K[.FFE0],             ;  STRIP OFF LENGTH-BITS
U 0292, 0019,0035,A180,F800,0030,047E   ;20225          N_AMX.Z_TST,CALL,J/ASPC         ;  CLOCK EXTRA BITS INTO PSL-Z-BIT
                                ;20226          ;--------------------------------;
                                ;20227  =111**1*
                                ;20228          ALU_Q.OXT[WORD],Q_ALU,          ;  SAVE DST-LENGTH IN RC1
U 02F2, 0003,603C,C5C0,3D88,0000,0BB2   ;20229          RC[T1]_ALU,ID[T1]_D,J/T2P.I1    ;  SAVE DST-LENGTH AND ADDRESS
                                ;20230  =;END   ;--------------------------------;
```

```
                                ;20231          ;REENTER HERE AFTER A FAULT OR INTERRUPT
                                ;20232          ;-------------------------------;
                                ;20233  T2P.I1: FE_K[.14],                      ;  SC GETS SRC-LENGTH-1,FE GETS 20.
                                ;20234          ALU_0-Q-1,R[R2]_ALU,            ;  INITIALIZE DST-LENGTH
U 0BB2, 001F,0008,21B0,FA90,0104,6631   ;20235          QK/RIGHT                        ;  DIVIDE LENGTH BY 2
                                ;20236
                                ;20237  ; ***********************************************
                                ;20238  ; * Patch no. 090, PCS 0BB2 trapped to WCS 119A *
                                ;20239  ; ***********************************************
                                ;20240
                                ;20241  =0***   ;-------------------------------;
                                ;20242  T2P.X1: ALU_D+Q+1,R[R3]_ALU,            ;  INITIALIZE R3 TO HIGH DST-ADDRESS
                                ;20243          Q_ID[T0],                       ;  GET SRC-ADDRESS
U 0631, 001D,1A11,C1F0,2E98,0000,00F9   ;20244          PSL.CC?,CALL,J/BCD.FPD
                                ;20245          ;-------------------------------;
U 0639, 0000,003C,B580,3C00,0000,0BB3   ;20246          ID[FPDA]_D                      ;  STORE MEMORY-FAULT ADDRESS (33)
                                ;20247          ;-------------------------------;
                                ;20248          ALU_Q+K[SC],R[R1]_ALU,          ;  GENEARATE HIGH SRC-ADDRESS
U 0BB3, 0019,2114,1D80,FA88,0200,080C   ;20249          VAK/LOAD,Z?                     ;  LOAD ADDRESS IN VA, TEST LENGTH
                                ;20250          ;-------------------------------;
                                ;20251  =0      ;BRANCH ON ALU Z-BIT
                                ;20252          ;-------------------------------;
                                ;20253          Q_RC[T5],                       ;  GET TABLE-ADDRESS
                                ;20254          SC_K[.A0],                      ;  BUILD CONSTANT AA
U 080C, 0010,8038,25C0,4128,0084,680D   ;20255          D[BYTE]_CACHE                   ;  READ TRAILING BYTE
                                ;20256          ;-------------------------------;
                                ;20257          ALU_D.OXT[BYTE]+Q,VAK/LOAD,     ;  LOAD INDEXED ADDRESS
                                ;20258          SC_SC.OR.K[.A],
                                ;20259          LC_RC[T6],Q_0,                  ;  GET SRC-LENGTH
U 080D, 001F,8114,F5F8,F930,0284,2814   ;20260          Z?                              ;  TEST SRC-LENGTH AGAIN
                                ;20261          ;-------------------------------;
                                ;20262  =0      ;BRANCH ON ALU Z-BIT
                                ;20263          ;-------------------------------;
                                ;20264          ALU_Q+LC+1,
                                ;20265          RC[T6]_ALU,SGN/LOAD.SS,
                                ;20266          STATE_K[.1],                    ;  INITIALIZE STATE-REGISTER
U 0814, 0011,A010,0581,41B0,1404,6BB4   ;20267          D[BYTE]_CACHE,J/T2P.2
                                ;20268          ;-------------------------------;
                                ;20269          STATE_K[.10],
                                ;20270          Q_R[R2],CLK.UBCC,
U 0815, 0F00,003C,65C0,FA10,1414,6197   ;20271          D_0,J/T2P.LONG1                 ;  WRITE 0 IN DST-STRING
                                ;20272          ;-------------------------------;
```

```
                                ;20273  T2P.2:  ALU_D.OXT[BYTE]-K[SC],
U 0BB4, 001B,8000,1DD0,F800,0081,0BB5   ;20274          SC_FE,Q_DEC.CON                 ;  SC GETS 20.
                                ;20275          ;-------------------------------;
                                ;20276          ALU_Q.XOR.K[.60],               ;  TEST DECIMAL CONSTANT
                                ;20277          CLK.UBCC,BYTE,
U 0BB5, 0019,AF20,A580,F930,0010,0902   ;20278          LC_RC[T6],BCDSGN?               ;  TEST DECIMAL SIGN
                                ;20279          ;-------------------------------;
                                ;20280  =10     ;BRANCH ON BCD-SIGN-NIBBLE
                                ;20281          ;-------------------------------;
                                ;20282  T2P.3:  ALU_D.AND.K[.F0],D_ALU,         ;  STRIP OFF SIGN-NIBBLE
U 0902, 0819,0134,CD80,F800,0000,081C   ;20283          Z?,J/T2P.4
                                ;20284          ;-------------------------------;
                                ;20285          STATE_STATE+1,
                                ;20286          ALU_D.AND.K[.F0],D_ALU,         ;  STRIP OFF SIGN-NIBBLE
U 0903, 0819,0134,CD80,F800,1400,C81C   ;20287          Z?,J/T2P.4
                                ;20288          ;-------------------------------;
                                ;20289  =0      ;BRANCH ON ALU Z-BIT
                                ;20290          ;-------------------------------;
U 081C, 0000,003C,0180,F800,0000,0106   ;20291  T2P.4:  J/RSVOPR                 ;  INVALID SIGN-BYTE
                                ;20292          ;-------------------------------;
                                ;20293  T2P.5:  FE_SC.OR.K[.C],                 ;  FE GETS 28., SC KEEPS 20.
                                ;20294          ALU_Q+LC+1,SGN/LOAD.SS,
                                ;20295          LAB_R1&RC[T6]_ALU,
                                ;20296          D_D.SWAP,                       ;  SWAP SIGN-BYTE INTO HIGH BYTE
U 081D, 0B13,1210,8581,FB30,0104,293E   ;20297          EALU?,J/T2P.6                   ;  BRANCH ON SIGN OF SRC-LENGTH
                                ;20298          ;-------------------------------;
                                ;20299  =1110   ;BRANCH ON SGN SRC
                                ;20300          ;-------------------------------;
                                ;20301  T2P.6:  STATE_STATE.OR.K[.10],          ;  SET END-OF-SRC-BIT OF STATE
U 093E, 0000,003C,65C0,FA10,1414,2197   ;20302          Q_R[R2],CLK.UBCC,J/T2P.LONG1    ;  GET DST-LENGTH
                                ;20303
                                ;20304  ; ************************************************
                                ;20305  ; * Patch no. 040, PCS 093E trapped to WCS 1174 *
                                ;20306  ; ************************************************
                                ;20307          ;-------------------------------;
                                ;20308  T2P.7:  VA_LA-K[.1],                    ;  UPDATE AND LOAD SRC-ADDRESS
                                ;20309          LC_RC[T6]&R1_ALU,               ;  GET SRC-LENGTH
U 093F, 0018,0000,05F8,FBB0,0200,0908   ;20310          Q_0,J/T2P.L10
                                ;20311          ;-------------------------------;
```

```
                                    ;20312          ;MAIN LOOP FOR READING 8 BYTES FROM SRC-STRING
                                    ;20313          ;EXPECTS RC6=-SRC-LENGTH
                                    ;20314          ;EXPECTS LA=R1=SRC-ADDRESS+1
                                    ;20315          ;EXPECTS FE=28,SC=28-4*(#OF BYTES READ)
                                    ;20316          ;-------------------------------;
                                    ;20317  =1100   ;BRANCH ON SC NE 0 AND SS-BIT
                                    ;20318          ;-------------------------------;
                                    ;20319  T2P.L0: D_DAL.SC,                       ;  SHIFT DATA INTO PLACE
                                    ;20320          Q_R[R2],CLK.UBCC,
                                    ;20321          STATE_STATE.OR.K[.10],          ;  SET END-OF-STRING-BIT
U 094C, 0D00,0F3C,65C0,FA10,1414,2195 ;20322        ZONED?,J/T2P.LONG0              ;  TEST LAST BYTE
                                    ;20323          ;-------------------------------;
                                    ;20324          D_DAL.SC,                       ;  SHIFT DATA INTO PLACE
                                    ;20325          Q_R[R2],CLK.UBCC,
U 094D, 0D00,0F3C,01C0,FA10,0010,0195 ;20326        ZONED?,J/T2P.LONG0              ;  TEST LAST BYTE
                                    ;20327          ;-------------------------------;
                                    ;20328          D_DAL.SC,
                                    ;20329          SC_K[.FFFC],Q_0,
U 094E, 0D00,0F3C,F1F8,F800,0084,6921 ;20330        ZONED?,J/T2P.END.OF.SRC         ;  END OF STRING, NOT LONG
                                    ;20331          ;-------------------------------;
                                    ;20332  T2P.L00:        VA_LA-K[.1],            ;  UPDATE ADDRESS
                                    ;20333          LC_RC[T6]&R1_ALU,
                                    ;20334          FE_SC,SC_FE,                    ;  SWAP SC(28.) WITH FF(BYTE-COUNT)
                                    ;20335          D_DAL.SC,Q_0,                   ;  SHIFT DATA INTO PLACE
U 094F, 0D18,0F00,05F8,FBB0,0381,0909 ;20336        ZONED?,J/T2P.L1                 ;  TEST FOR LEGAL ZONED BYTE
                                    ;20337          ;-------------------------------;
                                    ;20338  =01     ;BRANCH ON LEGAL ZONED BYTE
                                    ;20339          ;-------------------------------;
U 0909, 0000,003C,0180,F800,0000,0106 ;20340  T2P.L1: J/RSVOPR                       ;  ILLEGAL ZONED BYTE
                                    ;20341          ;-------------------------------;
                                    ;20342  T2P.L10:
                                    ;20343          ALU_Q+LC+1,SGN/LOAD.SS,         ;  LOAD ALU<15> INTO SS
                                    ;20344          LAB_R1&RC[T6]_ALU,              ;  GET SRC-ADDRESS, UPDATE  COUNT
                                    ;20345          Q_D,D[BYTE]_CACHE,              ;  READ NEXT SRC-BYTE
                                    ;20346          FE_SC-K[.4],CLK.UBCC,           ;  UPDATE AND CLOCK BYTE-COUNT
                                    ;20347          SC_FE,                          ;  MOVE SHIFT-COUNT(=28) INTO SC
U 090B, 0011,B210,11E1,4330,0195,A94C ;20348        EALU?,J/T2P.L0
                                    ;20349          ;-------------------------------;
```

```
                                ;20350  =00***01
                                ;20351          ;---------------------------------;
                                ;20352  T2P.LONG0:      ;ENTER HERE AFTER ASSEMBLING A COMPLETE LONGWORD
                                ;20353          ;THIS ROUTINE WRITES LONGWORD IN D INTO DST-STRING
                                ;20354          ;---------------------------------;
U 0195, 0000,003C,0180,F800,0000,0106   ;20355          J/RSVOPR                        ;  ILLEGAL BYTE
                                ;20356          ;---------------------------------;
                                ;20357  =00***11
                                ;20358  T2P.LONG1:
                                ;20359          LA_RA[R3],                      ;  GET DST-ADDRESS
                                ;20360          ALU_Q+K[.8],                    ;  INCREMENT LENGTH
                                ;20361          SHF7ALU.DT.LONG,                ;  DIVIDE BY 4 TO MAKE MASK
                                ;20362          QK/SHF,SC_ALU,                  ;  LOAD MASK IN SC
                                ;20363          CLK.UBCC,SIGNS?,                ;  TEST LENGTH AND DATA
                                ;20364          INTRPT.STROBE,                  ;  STROBE FOR INTERRUPTS
U 0197, 0079,2D15,01C0,F898,4092,0E59   ;20365          CALL,J/WRITE1                   ;  CALL WRITE-BCD ROUTINE
                                ;20366          ;---------------------------------;
                                ;20367  =01***11
                                ;20368          Q_ID[T0],                       ;  GET SRC-ADDRESS
                                ;20369          ALU_0(A),R[R0]_ALU,             ;  CLEAR R0
                                ;20370          N_ARX.Z_TST,                    ;  CLEAR PSL N-BIT
                                ;20371          STATE7-4?,                      ;  TEST FOR END OF SRC-STRING
U 01B7, 0003,163C,C1F0,2E80,0030,035C   ;20372          J/T2P.FIN1
                                ;20373          ;---------------------------------;
                                ;20374  =11***11
                                ;20375          STATE_STATE.OR.K[.4],           ;  SET 1.WRITE BIT
                                ;20376          R[R3]_LA-K[.4],                 ;  UPDATE DST-ADDRESS
U 01F7, 0018,0E00,1180,FA98,1404,2C06   ;20377          BEN/INTERRUPT,J/T2P.FIN2        ;  TEST FOR INTERRUPT REQUESTS
                                ;20378          ;---------------------------------;
                                ;20379  =1*0    ;BRANCH ON END-OF-SCR-BIT OF STATE
                                ;20380          ;---------------------------------;
                                ;20381  T2P.LONG3:
                                ;20382          R[R2]_Q+K[.8],                  ;  UPDATE DST-LENGTH
                                ;20383          SC_SC+1,FEK/LOAD,               ;  LOAD SC AND FE WITH 28
U 033C, 0F19,2014,01F8,FA90,0180,C93F   ;20384          Q_0,D_0,J/T2P.7
                                ;20385          ;---------------------------------;
                                ;20386          R[R2]_Q+K[.8],                  ;  UPDATE DST-LENGTH
U 033D, 0F19,2014,01F8,FA90,0000,093E   ;20387          Q_0,D_0,J/T2P.6                 ;  NO MORE INPUT-DATA
                                ;20388          ;---------------------------------;
```

```
                                    ;20389          ;ENTER HERE IF LESS THAN A LONGWORD REMAINS OF SRC-STRING.
                                    ;20390  =01     ;BRANCH ON LEGAL ZONED BYTE
                                    ;20391          ;-------------------------------;
                                    ;20392  T2P.END.OF.SRC:
U 0921, 0000,003C,0180,F800,0000,0106 ;20393        J/RSVOPR
                                    ;20394          ;-------------------------------;
U 0923, 0000,003C,0180,F800,0080,EBB6 ;20395        SC_NABS(SC-FE)                  ;  GET READY TO RIGHT-ADJUST DATA
                                    ;20396          ;-------------------------------;
                                    ;20397          STATE_STATE.OR.K[.10],          ;  SET END-OF-SRC-BIT OF STATE
                                    ;20398          Q_R[R2],CLK.UBCC,               ;  GET DST-LENGTH
U 0BB6, 0D00,003C,65C0,FA10,1414,2197 ;20399        D_DAL.SC,J/T2P.LONG1            ;  RIGHT-ADJUST DATA
                                    ;20400          ;-------------------------------;
                                    ;20401
                                    ;20402          ;ENTER HERE IF DST-STRING IS FULL
                                    ;20403  =1*0    ;BRANCH ON END-OF-SRC-BIT OF STATE
                                    ;20404          ;-------------------------------;
                                    ;20405  T2P.FIN1:
                                    ;20406          Q_LB,SC_K[.1B],LA_RA[R1],       ;  GET SRC-ADDRESS
U 035C, 000C,1638,EDC0,F888,0084,633C ;20407        STATE7-4?,J/T2P.LONG3           ;  TEST END-OF-SRC-BIT OF STATE
                                    ;20408          ;-------------------------------;
                                    ;20409          STATE_STATE.ANDNOT.K[.30],      ;  CLEAR BITS 4 AND 5
U 035D, 0000,173C,7980,F800,1404,488D ;20410        STATE3-0?,J/FINI1               ;  TEST SIGN-BIT OF STATE
                                    ;20411          ;-------------------------------;
                                    ;20412  =110    ;BRANCH ON INTERRUPT REQUEST
                                    ;20413          ;-------------------------------;
                                    ;20414  T2P.FIN2:
                                    ;20415          Q_LB,SC_K[.1B],LA_RA[R1],       ;  GET SRC-ADDRESS
U 0C06, 000C,1638,EDC0,F888,0084,633C ;20416        STATE7-4?,J/T2P.LONG3           ;  TEST END-OF-SRC-BIT
                                    ;20417          ;-------------------------------;
                                    ;20418          STATE_K[.80],                   ;  SET INTERRUPT-BIT OF STATE
U 0C07, 0000,003C,4180,F800,1404,6033 ;20419        J/SAVE.BCD                      ;  SAVE CONTEXT
                                    ;20420          ;-------------------------------;
```

```
;20421  .TOC    ""      Decimal string        : CVTSP''
;20422
;20423  ;CONVERT LEADING SEPARATE NUMERIC TO PACKED
;20424  ;ALGORITHM:
;20425  ;       1. STARTING AT ''CVTSP.INIT'' THE SPECIFIERS ARE EVALUATED
;20426  ;       AND THE REGISTERS INITIALIZED. FIRST PART DONE IS SET.
;20427  ;       THE SIGN-BYTE IS READ AND DECODED ( I.E 2B OR 20 FOR +,
;20428  ;       AND 2D FOR -) (''S2P.2'').
;20429  ;       THE REST OF THE INSTRUCTION USES CODE SHARED WITH THE
;20430  ;       CVTTP-INSTRUCTION.
;20431
;20432  ;       2. THE MAIN LOOP CONSISTS OF TWO PARTS.
;20433  ;       FIRST BYTES ARE READ FROM THE SRC-STRING, AND PACKED INTO A
;20434  ;       A LONGWORD (''T2P.L0''), UNTIL THE LONGWORD IS COMPLETE OR THE SRC-STRING
;20435  ;       IS EXHAUSTED. THEN THE LONGWORD IS WRITTEN INTO THE
;20436  ;       DST-STRING (''T2P.LONG0''), AND THE FIRST STEP IS REPEATED.
;20437  ;       WHILE READING ZONED BYTES FROM THE SRC-STRING, THEY ARE
;20438  ;       CHECKED FOR CORRECT FORMAT, AND A RESERVED OPERAND FAULT IS TAKEN IF
;20439  ;       THEY ARE NOT IN THE RANGE 30-39.
;20440
;20441  ;       3. FINALLY, AFTER REACHING THE END OF BOTH STRINGS, (''T2P.FIN1''),
;20442  ;       THE CONDITION CODES ARE SET AND THE GENERAL REGISTERS ARE LOADED.
;20443
;20444  ;       IF AN INTERRUPT OR MEMORY FAULT OCCUR, THE ORIGINAL
;20445  ;       OPERANDS ARE SAVED IN GENERAL REGISTERS (''BCD.SAVE''),
;20446  ;       AND THE INSTRUCTION IS RESTARTED AT ''T2P.I1''.
;20447
;20448
;20449  ;STORAGE:
;20450  ;       T0 HAS LOW SRC-ADDRESS,
;20451  ;       T1 HAS LOW DST-ADDRESS,
;20452  ;       RC0 HAS LOW SRC-LENGTH
;20453  ;       RC1 HAS DST-LENGTH
;20454  ;       R1 GETS HIGH SRC-ADDRESS
;20455  ;       R2 GETS NEGATIVE DST-LENGTH
;20456  ;       R3 GETS HIGH DST-ADDRESS
;20457  ;       RC6 GETS NEGATIVE SRC-LENGTH
;20458
;20459  ;       OP-CODE IS 09
;20460  ;       MNEMONIC IS CVTSP
;20461  ;       THERE ARE NO INSTRUCTION DEPENDENT OPERATIONS.
;20462  ;       STATE-REGISTER:
;20463  ;-------------------------------------------------------------------;
;20464  ;INTRPT ;OVFLOW ;       ;END OF ;       ;1.     ;SIGN   ;       ;
;20465  ;       ;       ;       ;SRC    ;       ;WRITE  ;       ;       ;
;20466  ;       ;       ;       ;       ;       ;       ;       ;       ;
;20467  ;       ;       ;       ;       ;       ;       ;       ;       ;
;20468  ;-------------------------------------------------------------------;
;20469
```

```
                                ;20470          ;--------------------------------;
                                ;20471  4C1:
                                ;20472  CVTSP.INIT:
                                ;20473          ALU_Q.ANDNOT.K[.1F],            ;  TEST SRC-LENGTH
U 04C1, 0019,6024,8D80,F800,0050,0283   ;20474          N&Z_ALU.V&C_0,WORD
                                ;20475          ;--------------------------------;
                                ;20476  =000**1*
                                ;20477          ALU_Q.OXT[BYTE],                ;  ISOLATE LENGTH
                                ;20478          RC[T0]_ALU,SC_ALU,              ;  SAVE LENGTH IN SC AND RC0
                                ;20479          ID[T0]_D,                       ;  SAVE SRC-ADDRESS
U 0283, 0003,A03D,C180,3D80,0082,037E   ;20480          CALL,J7SPEC                     ;  EVALUATE DST-LENGTH
                                ;20481          ;--------------------------------;
                                ;20482  =001**1*
                                ;20483          ALU_D.AND.K[.FFE0],             ;  TEST DST-LENGTH
                                ;20484          N_AMX.Z_TST,                    ;  "AND" RESULT INTO Z-BIT
U 0293, 0019,0035,A180,F800,0030,047E   ;20485          CALL,J/ASPC                     ;  EVALUATE DST-ADDRESS
                                ;20486          ;--------------------------------;
                                ;20487  =111**1*
                                ;20488          ALU_Q.OXT[BYTE],RC[T1]_ALU,     ;  SAVE DST-LENGTH
                                ;20489          Q_ALU,                          ;  NEED Q 0-EXTENDED
U 02F3, 0003,A03C,C5C0,3D88,0000,0A28   ;20490          ID[T1]_D,J/S2P.1                ;  SAVE DST-ADDRESS
                                ;20491  =;END ;--------------------------------;
                                ;20492  ;S2P.1 IS NOW PART OF RESTART-ROUTINE
                                ;20493  ;S2P.1: STATE_K[ZERO],QK/RIGHT,         ;  INITIALIZE STATE, DIVIDE LENGTH BY 2
                                ;20494  ;       J/S2P.10
                                ;20495          ;--------------------------------;
                                ;20496  S2P.10: ALU_D+Q+1,R[R3]_ALU,             ;  GENERATE HIGH DST-ADDRESS
U 0BB8, 001D,0010,C1F0,2E98,0000,0BB9   ;20497          Q_ID[T0]                        ;  GET SRC-ADDRESS
                                ;20498          ;--------------------------------;
                                ;20499          VA_Q,                           ;  LOAD LOW SRC-ADDRESS
                                ;20500          FE_K[.18],                      ;  FE GETS 24.
U 0BB9, 0C01,203C,7D80,F800,0304,6632   ;20501          D_Q
                                ;20502          ;--------------------------------;
                                ;20503  =0***   ALU_D+K[SC]+1,                  ;  GENERATE HIGH SRC-ADDRESS
                                ;20504          LC_RC[T1]&R1_ALU,
U 0632, 0019,1A11,1D80,FB88,0000,00F9   ;20505          CALL,PSL.CC?,J/BCD.FPD          ;  SET FPD, TEST FOR LEGAL LENGTHS
                                ;20506          ;--------------------------------;
                                ;20507  S2P.2:  ALU_0-LC-1,R[R2]_ALU,           ;  INITIALIZE DST-LENGTH
U 063A, 0F13,0008,B580,[illegible],0000,0BBA   ;20508          ID[FPDA]_D,D_0                  ;  LOAD FPD-ADDRESS
                                ;20509          ;--------------------------------;
```

```
                                    ;20510          ;--------------------------------;
                                    ;20511          D[BYTE]_CACHE,                   ;  READ SIGN-BYTE
                                    ;20512          ALU_D-K[SC],RC[T6]_ALU,          ;  D HAD 0
                                    ;20513          SGN/LOAD.SS,                     ;  LOAD SS WITH SIGN
U 0BBA, 0019,8000,1D81,41B0,0081,0BBB ;20514        SC_FE                            ;  SC GETS 24.
                                    ;20515          ;--------------------------------;
                                    ;20516          ALU_D.XOR.K[.20],                ;  COMPARE IT WITH SPACE
                                    ;20517          D_ALU,CLK.UBCC,
U 0BBB, 0819,8020,7580,F930,0010,0BBC ;20518        BYTE,LC_RC[T6]                   ;  GET NEGATIVE SRC-LENGTH
                                    ;20519          ;--------------------------------;
                                    ;20520          ALU_D.XOR.K[.D],D_ALU,           ;  COMPARE SIGN-BYTE WITH ^X 2D
U 0BBC, 0819,8120,8980,F800,0010,0820 ;20521        CLK.UBCC,BYTE,Z?                 ;  TEST PREVIOUS COMPARISON
                                    ;20522          ;--------------------------------;
                                    ;20523  =0      ;BRANCH ON ALU Z-BIT
                                    ;20524          ;--------------------------------;
                                    ;20525          ALU_D.XOR.K[.6],                 ;  COMPARE WITH ^X 2B
U 0820, 0019,8120,D580,F800,0010,0824 ;20526        CLK.UBCC,BYTE,Z?,J/S2P.5         ;  TEST PREVIOUS COMPARISON
                                    ;20527          ;--------------------------------;
                                    ;20528  S2P.7:  FE_SC+K[.4],                     ;  FE GETS 28., SC HAS 24.
                                    ;20529          ALU_0+LC+1,SGN/LOAD.SS,          ;  UPDATE NEGATIVE SRC-LENGTH
                                    ;20530          LAB_R1&RC[T6]_ALU,               ;  GET SRC-ADDRESS
U 0821, 0F13,1210,1181,FB30,0104,893E ;20531        D_0,EALU?,J/T2P.6
                                    ;20532          ;--------------------------------;
                                    ;20533  =0      ;BRANCH ON ALU Z-BIT
                                    ;20534          ;--------------------------------;
U 0824, 0F00,013C,0180,F800,0000,081C ;20535  S2P.5:  D_0,Z?,J/T2P.4                   ;  TEST LAST COMPARISON
                                    ;20536          ;--------------------------------;
                                    ;20537          STATE_K[.2],                     ;  NEGATIVE STRING
U 0825, 0000,003C,0980,F800,1404,6821 ;20538        J/S2P.7
                                    ;20539          ;--------------------------------;
```

```
;20540  .TOC    ""      Decimal string        : ADDP4, ADDP6, SUBP4, SUBP6''
;20541
;20542  ;ROUTINE TO ADD OR SUBTRACT TWO PACKED STRINGS, WITH 4 OR 6 OPERANDS.
;20543
;20544  ;ALGORITHM:
;20545  ;       1. THE SPECIFIERS ARE EVALUATED, AND THE REGISTERS ARE INITIALIZED
;20546  ;       STARTING AT ''ADS.IN''. FIRST PART DONE-FLAG IS SET (''AS16'').
;20547  ;       NOTE THAT THE CODE IS SHARED BETWEEN THE FOUR INSTRUCTIONS,
;20548  ;       AND ONLY WHEN NECESSARY ARE DIFFERENT PATHS TAKEN BY
;20549  ;       BRANCHING ON IR<0> (4 OR 6 OPERANDS) AND STATE<3> (ADD/SUBTRACT).
;20550  ;
;20551  ;       2. THE MAIN LOOP STARTS AT ''ADS.EN'', AND CONSISTS OF
;20552  ;       SEVERAL STEPS:
;20553  ;               A. READ LONGWORD OF 1. STRING (''AS0''),
;20554  ;               USING ''READ-BCD''-SUBROUTINE.
;20555  ;               B. READ LONGWORD OF 2. STRING (''AS1''),
;20556  ;               USING ''READ-BCD''-ROUTINE OR ''READ-BCD-WITH
;20557  ;               WRITE-CHECK''-SUBROUTINE, DEPENDING ON
;20558  ;               NUMBER OF OPERANDS.
;20559  ;               C. IF THIS IS FIRST PASS THRU THE LOOP, THE SIGN-NIBBLES
;20560  ;               ARE TESTED, AND USED TO DETERMINE WHETHER AN ADD OR
;20561  ;               SUBTRACT SHOULD BE DONE (''FIRST.ADDSUB'').
;20562  ;               D. THEN THE ACTUAL ADD (''ADD1'') OR SUBTRACT (''SUB1'')
;20563  ;               TAKES PLACE.
;20564  ;               E. THE RESULTING LONGWORD IS WRITTEN INTO THE
;20565  ;               DEST-STRING, USING ''WRITE-BCD''-SUBROUTINE (''AS3'').
;20566  ;               F. ALL THE REGISTERS ARE UPDATED, I.E. ADDRESSES
;20567  ;               ARE DECREMENTED, AND LENGTHS ARE INCREASED (''AS4'').
;20568  ;               G. A TEST IS MADE FOR OVERFLOW (''AS8'').
;20569  ;               THIS TEST IS QUITE COMPLEX BECAUSE WE MAY BE DOING
;20570  ;               THE SUBTRACTION THE WRONG WAY, (I.E. SUBTRACTING
;20571  ;               A LARGER NUMBER FROM A SMALLER ONE) IN WHICH CASE THERE
;20572  ;               WOULD BE NO OVERFLOW IF LEFT-OVER DIGITS ARE ALL 9'S.
;20573  ;
;20574  ;       3. AFTER REACHING THE END OF ALL STRINGS,
;20575  ;       WE LOAD THE GENERAL REGISTERS AND SET THE CONDITION
;20576  ;       CODES (''ASF1'').
;20577  ;       A CHECK IS MADE TO SEE IF THERE IS A BORROW
;20578  ;       OUT OF THE LAST DIGIT, IN WHICH CASE THE DEST-STRING
;20579  ;       NEEDS TO BE NEGATED (''NEGATE'').
;20580  ;
;20581  ;       4. IF A FAULT OR INTERRUPT HAPPENS DURING THIS INSTRUCTION,
;20582  ;       THE CURRENT STATE OF THE OPERANDS ARE BACKED UP IN THE
;20583  ;       GENERAL REGISTERS (''ADS.MEMORY.FAULT'').
;20584  ;       THE INSTRUCTION IS RESTARTED AT ''ADS.EN''.
```

```
                                        ;20585
                                        ;20586  ;STORAGE:
                                        ;20587  ;       R0-R5 ARE USED TO HOLD LENGTHS AND ADDRESS.
                                        ;20588  ;       RC7 IS USED FOR OVERFLOW.
                                        ;20589  ;       RC6 IS USED TO HOLD FIRST OPERAND,WHILE READING SECOND
                                        ;20590  ;       DST-LENGTH IS SAVED IN R15 IN CASE STRING HAS TO BE NEGATED
                                        ;20591  ;       OR SIGN CHANGED
                                        ;20592
                                        ;20593  ;MNEMONICS AND OPCODES FOR THE 4 INSTRUCTIONS ARE:
                                        ;20594  ;       ADDP4,20
                                        ;20595  ;       ADDP6,21
                                        ;20596  ;       SUBP4,22
                                        ;20597  ;       SUBP6,23
                                        ;20598  ;       CCK/INST.DEP CLOCKS THE C-BIT IN THE PSL FROM ALU-CARRY,
                                        ;20599  ;       CLEARS V, LEAVES Z AND C UNCHANGED
                                        ;20600  ;       ALU/INST.DEP IS "A-B-BORROW"
                                        ;20601
                                        ;20602  STATE-REGISTER:
                                        ;20603  ;-------------------------------------------------------------
                                        ;20604  ;INTR:  ;OVFL:  ;ALL-9  ;NEGATE ;ADD/   ;1.TIM  ;SGN:   ;CARRY: ;
                                        ;20605  ;       ;       ;0=9'S  ;       ;  SUB  ;       ;       ;       ;
                                        ;20606  ;       ;       ;1=     ;       ;       ;       ;       ;       ;
                                        ;20607  ;       ;       ;       ;       ;       ;       ;       ;       ;
                                        ;20608  ;-------------------------------------------------------------
                                        ;20609
                                        ;20610  3CA:
                                        ;20611  ADS.IN: ;ENTER HERE FROM C-FORK, WITH LENGTH IN Q,ADDRESS IN D
                                        ;20612          ;-------------------------------;
                                        ;20613          ALU_Q,OXT[WORD],RC[T0]_ALU,     ;  SAVE LENGTH IN RC 0
                                        ;20614          ID[T0]_D,D_ALU,                 ;  SAVE ADDRESS IN ID[T0]
U 03CA, 0803,603D,C180,3D80,0000,037E   ;20615          CALL,J7SPEC                     ;  EVALUATE 2. LENGTH
                                        ;20616          ;-------------------------------;
                                        ;20617  3DA:
                                        ;20618          ALU_Q,AND,K[.FFE0],             ;  STRIP OFF HIGH BITS
U 03DA, 0019,2035,A180,F800,0050,047E   ;20619          N&Z_ALU,V&C_0,CALL,J/ASPC       ;  EVALUATE 2. ADDRESS
                                        ;20620          ;-------------------------------;
                                        ;20621  3FA:
                                        ;20622          ALU_Q,OXT[WORD],RC[T1]_ALU,     ;  SAVE 2. LENGTH
                                        ;20623          ID[T1]_D,                       ;  SAVE 2. ADDRESS
U 03FA, 0803,7B3C,C580,3D88,0000,018D   ;20624          D_ALU,IRQ?,J/ADS.I1             ;  2 OR 3 OPERANDS?
                                        ;20625          ;-------------------------------;
```

```
                                    ;20626  =01101  ;BRANCH ON LOW BIT OF OP-CODE
                                    ;20627          ;--------------------------------;
                                    ;20628  ADS.I1: STATE_K[ZERO],                  ;  CLEAR STATE-REGISTER
                                    ;20629          ALU_0-D-1,R[R15]_ALU,LONG,      ;  STORE NEGATIVE LENGTH
U 018D, 001F,2008,1980,FAF8,1404,6BBE ;20630        J/ASI5
                                    ;20631          ;--------------------------------;
                                    ;20632  =01111  ALU_D.AND.K[.FFE0],             ;  TEST 2. LENGTH
                                    ;20633          N_AMX.Z_TST,                    ;  AND IT INTO Z-BIT
U 018F, 0019,0035,A180,F800,0030,037E ;20634        CALL,J/SPEC                     ;  EVALUATE 3. LENGTH
                                    ;20635          ;--------------------------------;
                                    ;20636  =11111  ALU_D.OXT[WORD],                ;  0-EXTEND 3. LENGTH
                                    ;20637          RC[T3]_ALU,D_ALU,               ;  STORE IT IN RC3 AND D
U 019F, 0803,403C,1980,F998,1404,6590 ;20638        STATE_K[ZERO]                   ;  CLEAR STATE-REGISTER
                                    ;20639  =;END   ;--------------------------------;
                                    ;20640  =00*****
                                    ;20641          ALU_0-D-1,R[R15]_ALU,           ;  SAVE DST-LENGTH IN R15
                                    ;20642          DK/RIGHT,                       ;  DIVIDE DST-LENGTH BY 2
                                    ;20643          ID[T4]_D,                       ;  SAVE LENGTH
U 0590, 061F,2009,D180,3EF8,0000,047E ;20644        CALL,J7ASPC                     ;  EVALUATE DST-ADDRESS
                                    ;20645          ;-------------- ----------------;
                                    ;20646  =11*****
                                    ;20647          ALU_D+Q+1,R[R5]_ALU,            ;  GENERATE HIGH DST-ADDRESS
U 05F0, 001D,0010,D1F0,2EA8,0000,0BBD ;20648        Q_ID[T4],J/ADS.I2               ;  RETRIEVE DST-LENGTH
                                    ;20649  =;END   ;--------------------------------;
                                    ;20650  ADS.I2: ALU_0-Q-1,D_Q,R[R4]_ALU,        ;  INITIALIZE R4 WITH DST-LENGTH
U 0BBD, 0C1F,0008,0180,FAA0,0000,0BBE ;20651        J/ASI5                          ;  RETRIEVE LENGTH
                                    ;20652          ;--------------------------------;
```

```
                                    ;20653  ASI5:   ;--------------------------------;
                                    ;20654          ALU_D.AND.K[.FFE0],             ;  CLOCK DST-LENGTH
                                    ;20655          N_MUX.Z_TST,                    ;  AND IT INTO Z-BIT
U 0BBE, 0019,0034,A180,F900,0030,0BC0 ;20656        LC_RC[T0]                       ;  GET 1. SRC-LENGTH
                                    ;20657
                                    ;20658          ;--------------------------------;
                                    ;20659          ALU_LC,D_ALU,                   ;  D GETS 1. SRC-LENGTH
U 0BC0, 0810,0038,C1F0,2C00,0000,0BC1 ;20660        Q_ID[T0]                        ;  Q GETS 1. ADDRESS
                                    ;20661
                                    ;20662          ;--------------------------------;
                                    ;20663          ALU_0-D-1,R[R0]_ALU,LONG,       ;  STORE LENGTH IN R2
U 0BC1, 061F,2008,0180,FA80,0000,0BC2 ;20664        DK/RIGHT                        ;  DIVIDE BY 2 TO GET BYTE-COUNT
                                    ;20665
                                    ;20666          ;--------------------------------;
                                    ;20667          ALU_D+Q+1,                      ;  GENERATE HIGH SRC-ADDRESS
                                    ;20668          LC_RC[T1]&R1_ALU,               ;  GET 2. LENGTH, STORE 1. ADDRESS
U 0BC2, 001D,0010,C5F0,2F88,0000,0BC4 ;20669        Q_ID[T1]                        ;  GET 2. ADDRESS
                                    ;20670
                                    ;20671          ;--------------------------------;
                                    ;20672          ALU_K[.9],D_ALU.LEFT,
U 0BC4, 0838,1A38,DB80,F800,0000,02E8 ;20673        SI/MUL~,PSL.CC?                 ;  GENERATE ID-BUS- ADDRESS .13
                                    ;20674
                                    ;20675  =10**   ;10**-----------------------------;  BRANCH ON PSL<Z> C AND V  ARE CLEAR
U 02E8, 0000,003C,0180,F800,0000,0106 ;20676        J/RSVOPR                        ;  ILLEGAL STRING-LENGTHS
                                    ;20677
                                    ;20678          ;11**--------------------------;
                                    ;20679  ASI6:   ALU_0-LC-1,R[R2]_ALU,
                                    ;20680          ID[FPDA]_D,                     ;  LOAD FPDA WITH ADDRESS 13
U 02EC, 0013,0008,B580,3E90,2400,0BC5 ;20681        SET.FPD                         ;  SET FPD-BIT OF PSL
                                    ;20682
                                    ;20683  =;END   ;--------------------------------;
U 0BC5, 0C50,0038,01C0,F800,0000,0BC6 ;20684        D_Q,ALU_LC,Q_ALU.RIGHT          ;  GET 2. LENGTH
                                    ;20685
                                    ;20686          ;--------------------------------;
                                    ;20687          ALU_D+Q+1,R[R3]_ALU,            ;  GENERATE HIGH 2. ADDRESS
U 0BC6, [illegible],0010,0180,FA98,0000,02AB ;20688 J/ADS.EN                        ;  ENTER MAIN LOOP
```

```
                                ;20689  =0**10  ;0**10-------------------------;  BRANCH ON SS FLIP FLOP
                                ;20690  ADDSUB: ALU_R[R0],                      ;  FINISHED, SET CONDITION-CODES
                                ;20691          Q_ALU.RIGHT,SI/ASHR,            ;  GET LENGTH
U 02AA, 0040,003D,18C0,FA00,0084,6BCE   ;20692          SC_K[ZERO],CALL,J/REG.ADJ       ;  CLEAR R0, LOAD R1 WITH ADDRESS
                                ;20693
                                ;20694          ;0**11-------------------------;
                                ;20695  ADS.EN: ALU_R[R0],                      ;  ENTER LOOP HERE
                                ;20696          D_ALU.RIGHT,BYTE,CLK.UBCC,      ;  GET SRC-LENGTH
U 02AB, 0840,803C,0180,FA00,0010,092A   ;20697          J/AS0
                                ;20698
                                ;20699  =1**10  ;1**10-------------------------;
                                ;20700          ALU_R[R2],Q_ALU.RIGHT,          ;  PART OF FINISH-ROUTINE
U 02BA, 0040,003C,0CC0,FA10,0084,64CD   ;20701          SI/ASHR,SC_K[.3],J/ASF1         ;  GET READY TO RESET REGISTERS
                                ;20702
                                ;20703  =;END
                                ;20704  =10     ;10-------------------------;
                                ;20705  AS0:    LA_RA[R1],                      ;  GET 1. ADDRESS
                                ;20706          ALU_D+K[.3],                    ;  INCREMENT LENGTH
                                ;20707          D_ALU,CLK.UBCC,BYTE,            ;  CLOCK IT
U 092A, 0819,9B15,0D80,F888,0010,0AF7   ;20708          ALU?,CALL,J/READ0               ;  READ A LONG-WORD
                                ;20709
                                ;20710          ;11-------------------------;
                                ;20711          ID[TO]_D,                       ;  SAVE DATA IN ID[TO]
                                ;20712          ALU_R[R2],Q_ALU.RIGHT,          ;  GET 2. SRC-LENGTH
U 092B, 0040,803C,C1C0,3E10,0010,093A   ;20713          CLK.UBCC,BYTE                   ;
                                ;20714
                                ;20715  =;END
                                ;20716  =10     ;10-------------------------;
                                ;20717  AS1:    LA_RA[R3],ALU_Q+K[.3],          ;  GET ADDRESS, TEST LENGTH
                                ;20718          D_ALU,CLK.UBCC,BYTE,            ;
U 093A, 0819,9B15,0D80,F898,0010,0AF5   ;20719          ALU?,CALL,J/READ00              ;  READ A LONGWORD OF 2.OPERAND
                                ;20720
                                ;20721          ;11-------------------------;
                                ;20722  AS2:    FE_K[.8],                       ;  FOR USE IF 1. TIME
                                ;20723          ALU_0+Q,QK/DEC.CON,             ;
                                ;20724          STATE3-0?,                      ;  BRANCH ON 1.TIME,ADD/SUB
U 093B, 001F,1714,01D0,F800,0104,6953   ;20725          J/AS20                          ;
```

```
                                    ;20726  =0011   ;0011----------------------------;  BR ON 1TIME & A/S BITS/STATE
                                    ;20727  AS20:    Q_ID[TO],STATE_FE,                ;  GET FIRST OPERAND, INITIALIZE STATE
                                    ;20728           ALU_0(A),N&Z_ALU.V&C_0,           ;  SET Z-BIT, CLEAR C-BIT
                                    ;20729           IR2=1?,                           ;  TEST FOR ADD/SUB
U 0953, 0003,093C,C1F0,2C00,1450,6942 ;20730         J/FIRST.ADDSUB
                                    ;20731
                                    ;20732           ;0111----------------------------;
                                    ;20733  AS21:    ALU_D+Q,D_ALU,LONG,               ;  ADD THE 6'S TO OPERAND
U 0957, 081D,0014,C1F0,2C00,0000,0BC8 ;20734         Q_ID[TO],J/ADD1                   ;  GET FIRST OPERAND
                                    ;20735
                                    ;20736           ;1011----------------------------;
                                    ;20737           Q_ID[TO],STATE_FE,                ;  GET FIRST OPERAND, INITIALIZE STATE
                                    ;20738           ALU_0(A),N&Z_ALU.V&C_0,           ;  SET Z-BIT, CLEAR C-BIT
                                    ;20739           IR2=1?,                           ;  TEST FOR ADD/SUB
U 095B, 0003,093C,C1F0,2C00,1450,6942 ;20740         J/FIRST.ADDSUB                    ;
                                    ;20741
                                    ;20742           ;1111----------------------------;
                                    ;20743           Q_ID[TO],                         ;  GET 1. OPERAND
U 095F, 0000,003C,C1F0,2C00,0000,0BC9 ;20744         J/SUB1                            ;
                                    ;20745  =;END
```

```
                                    ;20746  ADD1:   ALU_D+Q+PSL.C,                  ;  ADD THE TWO OPERANDS
                                    ;20747          D_ALU,QK/DEC.CON,               ;  GENERATE DECIMAL CONSTANT
                                    ;20748          SET.CC(LONG),                   ;  LOAD PSL CARRY BIT FROM ALU RESULT
U 0BC8, 081D,1B2C,01D0,F800,0070,096D ;20749          IR0?,J/ADD2                     ;  TEST FOR 4 OR 6 OPERANDS
                                    ;20750          ;-------------------------------;
                                    ;20751  SUB1:   ALU_D[INST.DEP]Q,               ;  INST-DEPENDENT SUBTRACT WITH BORROW
                                    ;20752          D_ALU,QK/DEC.CON,               ;  D GETS RESULT, Q GETS 6'S
                                    ;20753          SET.CC(LONG),                   ;  CLOCK CARRY-BIT
U 0BC9, 081D,1B0C,01D0,F800,0070,096D ;20754          IR0?,J/ADD2                     ;  TEST FOR 4 OR 6 OPERANDS
                                    ;20755          ;-------------------------------;
                                    ;20756  =1101   ;BRANCH ON LOW BIT OF OP-CODE
                                    ;20757          ;1101-----------------------------;
U 096D, 081D,0000,0180,F800,0000,0140 ;20758  ADD2:   ALU_D-Q,D_ALU,J/AS3              ;  DECIMAL ADJUST RESULT
                                    ;20759
                                    ;20760  ; ************************************************
                                    ;20761  ; * Patch no. 025, PCS 096D trapped to WCS 115C *
                                    ;20762  ; ************************************************
                                    ;20763
                                    ;20764          ;1111-----------------------------;
                                    ;20765          ALU_D-Q,D_ALU,                  ;  DECIMAL ADJUST RESULT
U 096F, 081D,0000,0180,FA20,0000,0BCA ;20766          LAB_R[R4]                       ;  6 OPERANDS
                                    ;20767  =;END
                                    ;20768
                                    ;20769  ; ************************************************
                                    ;20770  ; * Patch no. 069, PCS 096F trapped to WCS 118F *
                                    ;20771  ; ************************************************
                                    ;20772
                                    ;20773  DC.PA.69:
                                    ;20774          ;-------------------------------;
U 0BCA, 0000,003C,0180,F8A8,0000,0140 ;20775          LA_RA[R5],J/AS3                 ;  SAVE A CYCLE BY MERGING THIS
                                    ;20776          ;-------------------------------;
                                    ;20777  =100****
                                    ;20778          ;-------------------------------;
                                    ;20779  AS3:    STATE_STATE.ANDNOT.K[.10],      ;  CLEAR NEGATE-BIT OF STATE
                                    ;20780          INTRPT.STROBE,                  ;  STOBE INTERRUPTS FOR LATER TEST
                                    ;20781          ALU_LB,Q_ALU,BYTE,              ;  GET DST-LENGTH
U 0140, 000C,8039,65C0,F800,5414,4C6B ;20782          CLK.UBCC,CALL,J/WRITE           ;  WRITE DST-LONGWORD
                                    ;20783          ;-------------------------------;
```

```
                                    ;20784          ;RETURN HERE AFTER WRITING D IN DST-STRING
                                    ;20785          ;-------------------------------;
                                    ;20786  =110****
                                    ;20787  AS4:    SC_K[ZERO],LAB_R[R0],Q_LB,      ;  GET FIRST LENGTH
                                    ;20788          D_0,                           ;  CLEAR LENGTH-SUM
                                    ;20789          BEN/INTERRUPT,                 ;  TEST FOR INTERRUPT REQUESTS
U 0160, 0F0C,0E39,19C0,FA00,0084,6C26 ;20790        CALL,J/UPDATE                  ;  UPDATE R0 AND R1
                                    ;20791          ;-------------------------------;
                                    ;20792  =111****
                                    ;20793          ALU_LA-K[.4],SC_SC+1,          ;  UPDATE SRC.1-ADDRESS
U 0170, 0018,0000,1180,F8E8,0080,C446 ;20794        R(SC)_ALU,LONG
                                    ;20795  =;END
                                    ;20796  =0****
                                    ;20797  DC.PA.25:
                                    ;20798          ;-------------------------------;
                                    ;20799          LAB_R[R2],Q_LB,                ;  UPDATE R2 AND R3
U 0446, 000C,0C39,01C0,FA10,0000,0C26 ;20800        CALL,J/UPDATE
                                    ;20801          ;1****--------------------------;
                                    ;20802          R[R3]_LA-K[.4],                ;  SRC.2-ADDRESS
                                    ;20803          STATE_STATE.OR.K[.4],          ;  SET 1.TIME BIT
U 0456, 0018,1B00,1180,FA98,1404,21AD ;20804        IR0?                           ;  TEST 2/3-OPERANDS
                                    ;20805  =;END   ;-------------------------------;
                                    ;20806  =01101  ;BRANCH ON LOW BIT OF OP-CODE
                                    ;20807          ;01101----------------------------;
U 01AD, 0000,173C,0180,F800,0000,0977 ;20808        STATE3-0?,J/AS8                ;  TEST FOR ADD/SUB
                                    ;20809          ;01111----------------------------;
                                    ;20810          LAB_R[R4],Q_LB,SC_SC+1,        ;  UPDATE DST LENGTH
U 01AF, 000C,0039,01C0,FA20,0080,CC26 ;20811        CALL,J/UPDATE
                                    ;20812  =11111  ;-------------------------------;
                                    ;20813          R[R5]_LA-K[.4],LONG,           ;  UPDATE DST ADDRESS
U 01BF, 0018,1700,1180,FAA8,0000,0977 ;20814        STATE3-0?,J/AS8                ;  TEST FOR ADD/SUB
                                    ;20815  =;END   ;-------------------------------;
```

```
                                    ;20816  =0111    ;0111----------------------------;  BRANCH ON ADD/SUB-BIT OF STATE
U 0977, 0000,123C,0180,F800,0000,02AA ;20817  AS8:     EALU?,J/ADDSUB                  ;  TEST FOR END OF ALL OPERANDS
                                    ;20818
                                    ;20819          ;1111----------------------------;
                                    ;20820          LC_RC[T7],                      ;  GET OVERFLOW DATA
                                    ;20821          ALU_0+Q,Q_DEC.CON,              ;  GET READY FOR OVERFLOW TEST
U 097F, 001F,1B14,01D0,F938,0081,0983 ;20822          SC_FE,ALU?                      ;  TEST FOR END OF DST
                                    ;20823  =;END
                                    ;20824
                                    ;20825  =0011   ;0011----------------------------;  BR ON ALU<N&Z>, ON DST LEN AFT UPDATE
                                    ;20826          Q_Q+LC,                         ;  CHECKING FOR ALL 9'S IN RC 7
U 0983, 0011,2214,01C0,F800,0000,0C15 ;20827          RŌR?,J/AS9                      ;  TEST FOR BORROW
                                    ;20828
                                    ;20829          ;0111----------------------------;
U 0987, 0000,123C,0180,F800,0000,02AA ;20830          EALU?,J/ADDSUB                  ;  NO OVERFOLW YET
                                    ;20831
                                    ;20832          ;1011----------------------------;
U 098B, 0000,123C,0180,F800,0000,02AA ;20833          EALU?,J/ADDSUB                  ;  GO TO BEGINNING OF LOOP
                                    ;20834  =;END
                                    ;20835
                                    ;20836  =101    ;101-----------------------------;  BRANCH ON PSL C-BIT
                                    ;20837  AS9:    ALU_Q-MASK-1,Q_DEC.CON,         ;  GET MASK BIT FOR ADDING TO OVERFLOW
                                    ;20838          EALŪ_K[.1],                     ;  CLEAR EALU CC
U 0C15, 0001,2008,05D0,F800,0014,6BCC ;20839          CLK.ŪBCC,J/CHECK.9
                                    ;20840
                                    ;20841          ;111-----------------------------;
                                    ;20842          STATE_STATE.OR.K[.20],          ;  SET NOT-ALL-9-BIT
U 0C17, 0000,123C,7580,F800,1404,22AA ;20843          EALU?,J/ADDSUB
                                    ;20844  =;END
                                    ;20845  CHECK.9:;-----------------------------;
U 0BCC, 081D,0300,0180,F800,0000,07F8 ;20846          ALU_D-Q,D_ALU,C31?              ;  DECIMAL ADJUST(NO NEED), TEST CARRY
                                    ;20847
                                    ;20848  =0*      ;0*-----------------------------;  BRANCH ON ALU C31
                                    ;20849          STATE_STATE.OR.K[.20],          ;  NO LONGER ALL 9'S
U 07F8, 0000,123C,7580,F800,1404,22AA ;20850          EALU?,J/ADDSUB
                                    ;20851
                                    ;20852          ;1*-----------------------------;
U 07FA, 0000,123C,0180,F800,0000,02AA ;20853          EALU?,J/ADDSUB                  ;  RESTART LOOP
                                    ;20854  =;END
                                    ;20855
                                    ;20856  ; ************************************************
                                    ;20857  ; * Patch no. 089, PCS 07FA trapped to WCS 1199 *
                                    ;20858  ; ************************************************
```

```
                                ;20859          ;ENTER HERE AFTER READING FIRST TWO OPERANDS
                                ;20860          ;ADD/SUB-BIT IS SET TO  REFLECT OP-CODE
                                ;20861
                                ;20862          ;--------------------------------;
                                ;20863  =10      ;BRANCH ON ADD/SUB BIT OF OP-CODE
                                ;20864          ;10------------------------------;
                                ;20865  FIRST.ADDSUB:
                                ;20866          ALU_D.ANDNOT.K[.F],             ;  STRIP OFF SIGN-NIBBLE
                                ;20867          D_Q,Q_ALU,
                                ;20868          STATE_STATE.ANDNOT.FE,          ;  CLEAR ADD/SUB-BIT OF STATE
U 0942, 0C19,0F24,61C0,F800,1400,494A   ;20869  BCDSGN?,J/FIRST.0
                                ;20870          ;11------------------------------;
                                ;20871          ALU_D.ANDNOT.K[.F],             ;  STRIP OFF SIGN-NIBBLE
                                ;20872          D_Q,Q_ALU,                      ;  SWAP THE OPERANDS
U 0943, 0C19,0F24,61C0,F800,0000,094A   ;20873  BCDSGN?,J/FIRST.0
                                ;20874  =;END   ;--------------------------------;
                                ;20875  =10      ;BRANCH ON BCD-SIGN
                                ;20876          ;10------------------------------;
                                ;20877  FIRST.0:
                                ;20878          ALU_D.ANDNOT.K[.F],D_ALU,       ;  STRIP SIGN-NIBBLE
U 094A, 0819,0F24,6180,F800,0000,096A   ;20879  BCDSGN?,J/P2                    ;  POSITIVE
                                ;20880          ;11------------------------------;
                                ;20881          STATE_STATE.OR.K[.2],           ;  NEGATIVE
U 094B, 0000,0F3C,0980,F800,1404,2962   ;20882  BCDSGN?                         ;  TEST THE OTHER SIGN
                                ;20883  =;END   ;--------------------------------;
                                ;20884  =10      ;BRANCH ON BCD-SIGN
                                ;20885          ;10------------------------------;
                                ;20886          ALU_D.ANDNOT.K[.F],D_ALU,       ;  STRIP SIGN-NIBBLE
U 0962, 0819,0024,6180,F800,1400,896A   ;20887  STATE_STATE+FE,J/P2              ;  COMPLEMENT ADD/SUB
                                ;20888          ;11------------------------------;
                                ;20889          ALU_D.ANDNOT.K[.F],D_ALU,       ;  STRIP SIGN-NIBBLE
U 0963, 0819,0024,6180,F800,0000,096A   ;20890  J/P2
                                ;20891  =;END   ;--------------------------------;
                                ;20892  =011*    ;BRANCH ON ADD/SUB-BIT OF STATE
                                ;20893          ;011*----------------------------;
                                ;20894  FIR1:    ALU_D+Q,D_ALU,Q_ID[T0],         ;  START ADD-OPERATION
U 0556, 081D,0014,C1F0,2C00,0000,0BC8   ;20895  J/ADD1
                                ;20896          ;111*----------------------------;
                                ;20897          ALU_Q+MASK,                     ;  FORCES A CARRY
                                ;20898          SET_CC(LONG),                   ;  SET C-BIT
                                ;20899          Q_ID[T0],                       ;  GET 1. OPERAND
U 055E, 0001,2014,C1F0,2C00,0070,0BC9   ;20900  J/SUB1
                                ;20901  =;END   ;--------------------------------;
                                ;20902  =10      ;BRANCH ON BCD-SIGN
                                ;20903          ;10------------------------------;
                                ;20904  P2:      ID[T0]_D,D_Q,                   ;  SAVE STRIPPED OPERAND IN T0
                                ;20905          ALU_0+Q,Q_DEC.CON,              ;  GET READY FOR ADD
                                ;20906          SC_FE,
U 096A, 0C1F,1714,C1D0,3C00,0081,0556   ;20907  STATE3-0?,J/FIR1                 ;  TEST ADD/SUB BIT
                                ;20908          ;11------------------------------;
U 096B, 0000,003C,0180,F800,1400,896A   ;20909  STATE_STATE+FE,J/P2              ;  COMPLEMENT ADD/SUB
                                ;20910  =;END   ;--------------------------------;
                                ;20911
                                ;20912
```

```
                                    ;20913
                                    ;20914          ;ROUTINE TO UPDATE POINTERS IN REGISTERS ADDRESSED BY SC
                                    ;20915          ;UPDATES CARRY-BIT OF STATE FROM THAT OF THE PSL-CARRY
                                    ;20916          ;------------------------------;
                                    ;20917  =110     ;BRANCH ON INTERRUPT REQUEST BIT (ADD/SUBTRACT ONLY)
                                    ;20918          ;110-----------------------------;
                                    ;20919  UPDATE: ALU_Q+K[.8],R(SC)_ALU,LONG,     ;  UPDATE LENGTH
                                    ;20920          Q_ALU,SC_SC+1,CLK.UBCC,         ;  POINT TO ADDRESS
                                    ;20921          SGN/LOAD.SS,ROR?,               ;  TEST CARRY-BIT
U 0C26, 0019,2214,01C1,F8E8,0090,CC35   ;20922          J/UPDATE.1
                                    ;20923          ;111-----------------------------;
                                    ;20924          STATE_STATE.OR.K[.80],          ;  SET INTERRUPT-BIT OF STATE
U 0C27, 0000,003C,4180,F800,1404,2013   ;20925          J/ADS.MEMORY.FAULT              ;  JOIN FAULT-ROUTINE
                                    ;20926  =;END    ;------------------------------;
                                    ;20927  =101     ;BRANCH ON C-BIT OF PSL
                                    ;20928          ;101-----------------------------;
                                    ;20929  UPDATE.1:
                                    ;20930          STATE_STATE.ANDNOT.K[.1],       ;  CLEAR CARRY-BIT OF STATE
                                    ;20931          LAB_R(SC),                      ;  GET ADDRESS
                                    ;20932          D_D.OR.Q,                       ;  TEST FOR NEGATIVE
                                    ;20933          SGN/LOAD.SS,                    ;  LOAD SS WITH ALU<15>
U 0C35, 081D,0032,0581,F868,1404,4010   ;20934          RETURN10
                                    ;20935          ;111-----------------------------;
                                    ;20936          STATE_STATE.OR.K[.1],           ;  SET CARRY-BIT OF STATE
                                    ;20937          LAB_R(SC),                      ;  GET ADDRESS
                                    ;20938          D_D.OR.Q,                       ;  TEST FOR NEGATIVE
                                    ;20939          SGN/LOAD.SS,                    ;  LOAD SS WITH ALU<15>
U 0C37, 081D,0032,0581,F868,1404,2010   ;20940          RETURN10
                                    ;20941  =;END    ;------------------------------;
```

```
                                    ;20942  REG.ADJUST:
                                    ;20943          ;ROUTINE WHICH USES LENGTH IN REGISTER(SC) TO
                                    ;20944          ;RESET ADDRESS IN REGISTER(SC+1).
                                    ;20945          ;-------------------------------;
                                    ;20946          LAB_R(SC),ALU/A,AMX/LA,         ;  GET LENGTH
U 0BCD, 0040,003C,00C0,F868,0000,0BCE ;20947        Q_ALU.RIGHT,SI/ASHR             ;  SIGN-EXTEND, DIVIDE BY 2
                                    ;20948          ;-------------------------------;
                                    ;20949  REG.ADJ:        ALU_0(A),R(SC)_ALU,
U 0BCE, 0003,003C,0180,F8E8,0080,CBD0 ;20950        LONG,SC_SC+1                    ;  CLEAR LENGTH, GET ADDRESS
                                    ;20951          ;-------------------------------;
                                    ;20952  REG.AD: LAB_R(SC),
U 0BD0, 0002,A03C,01C0,F868,0000,0BD1 ;20953        ALU_Q.SXT[BYTE],Q_ALU           ;  SIGN-EXTEND LENGTH
                                    ;20954          ;-------------------------------;
                                    ;20955          ALU_LA+Q,R(SC)_ALU,LONG,        ;  SAVE STRING-ADDRESS
                                    ;20956          Q_ALU,
U 0BD1, 001C,0016,01C0,F8E8,0000,0010 ;20957        RETURN10
                                    ;20958          ;-------------------------------;
                                    ;20959
                                    ;20960
                                    ;20961  SGN.CHANGE:     ;ROUTINE TO CONVERT A -0 STRING TO A +0 STRING.
                                    ;20962          ;EXPECTS LENGTH TO BE IN R15, ADDRESS IN R3 OR R5,
                                    ;20963          ;DEPENDING ON LOW BIT OF OPCODE.
                                    ;20964          ;-------------------------------;
                                    ;20965          ALU_R[R15].ORNOT.K[.FF],
                                    ;20966          Q_ALU.RIGHT,SI/ASHR,            ;  GET LENGTH, DIVIDE BY 2
U 0BD2, 0058,1B1C,48C0,FA78,0000,098D ;20967        IR0?                            ;  TEST 2/3-OPERANDS
                                    ;20968          ;-------------------------------;
                                    ;20969  =1101   ;BRANCH ON LOW BIT OF OPERAND
                                    ;20970          ;-------------------------------;
                                    ;20971  SGN.C1: LA_RA[R3],                      ;  GET ADDRESS
                                    ;20972          D_R[.C],                        ;  GET DATA
U 098D, 0818,0038,8580,F898,0000,0BD3 ;20973        J7SGN.C2
                                    ;20974          ;-------------------------------;
                                    ;20975  SGN.C10:
                                    ;20976          LA_RA[R5],                      ;  GET 3-OPERAND DST-ADDRESS
U 098F, 0818,0038,8580,F8A8,0000,0BD3 ;20977        D_R[.C]                         ;  GET DATA
                                    ;20978          ;-------------------------------;
U 0BD3, 001C,0008,0180,F800,0200,0BD4 ;20979  SGN.C2: ALU_LA-Q-1,VAK/LOAD            ;  GET ADDRESS OF SIGN-BYTE
                                    ;20980
                                    ;20981  DC.PA.79:
                                    ;20982          ;-------------------------------;
                                    ;20983          ALU_K[ZERO],N&Z_ALU.V&C_0,      ;  SET Z-BIT, CLEAR N-BIT,C,V
U 0BD4, 0018,8038,1980,3000,0050,0B29 ;20984        CACHE_D[BYTE],J7FINI8           ;  WRITE +0
                                    ;20985          ;-------------------------------;
```

```
                                ;20986  ;ROUTINE WHICH NEGATES STRING. POINTED TO BY R2 OR R4, WITH LENGTH IN R15.
                                ;20987  ;USED WHEN A SUBTRACT RESULTS IN A BORROW OUT OF THE MOST SIGNIFICANT DIGIT.
                                ;20988          ;-------------------------------;
                                ;20989  =101    ;BRANCH ON ALL 9'S BIT OF STATE
                                ;20990          ;101----------------------------;
                                ;20991  NEGATE0:
                                ;20992          STATE_STATE.ANDNOT.K[.40],      ;  SET NEGATE-BIT OF STATE
                                ;20993          ALU_R[R15],D_ALU.RIGHT,SI/ASHR, ;  DIVIDE LENGTH BY 2
                                ;20994          CLK.UBCC,LONG,IR0?,             ;  TEST FOR 4 OR 6 OPERANDS
U 0C45, 0840,1B3C,3080,FA78,1414,499D   ;20995          J/NEGA0
                                ;20996          ;111----------------------------;
                                ;20997  NEGATE: STATE_STATE.OR.K[.10],          ;  SET NEGATE-BIT OF STATE
                                ;20998          ALU_R[R15],D_ALU.RIGHT,SI/ASHR, ;  DIVIDE LENGTH BY 2
                                ;20999          CLK.UBCC,LONG,IR0?,             ;  TEST FOR 4 OR 6 OPERANDS
U 0C47, 0840,1B3C,6480,FA78,1414,299D   ;21000          J/NEGA0
                                ;21001
                                ;21002  ; ************************************************
                                ;21003  ; * Patch no. 093, PCS 0C47 trapped to WCS 119D *
                                ;21004  ; ************************************************
                                ;21005
                                ;21006  =;END   ;-------------------------------;
                                ;21007  =1101   ;BRANCH ON 2/3-OPERAND BIT OF OP-CODE
                                ;21008          ;1101---------------------------;
U 099D, 0000,003C,0180,F898,0000,0970   ;21009  NEGA0:  LA_RA[R3],J/NEGA1          ;  4 OPERANDS
                                ;21010          ;1T11---------------------------;
U 099F, 0000,003C,0180,F8A8,0000,0970   ;21011          LA_RA[R5]               ;  6 OPERANDS
                                ;21012  =;END   ;-------------------------------;
                                ;21013  =00
                                ;21014  NEGA1:  ALU_D+K[.3],                    ;  INCREMENT LENGTH
                                ;21015          D_ALU,CLK.UBCC,BYTE,
                                ;21016          ALU?,                           ;  TEST LENGTH
U 0970, 0819,9B15,0D80,F800,0010,0B17   ;21017          CALL,J/READOW           ;  READ WITH WRITE-CHECK
                                ;21018          ;01-----------------------------;
                                ;21019          STATE_STATE.ANDNOT.K[.4],       ;  CLEAR 1.TIME BIT OF STATE
                                ;21020          ALU_LB,Q_ALU.RIGHT,SI/ASHR,     ;
U 0971, 004C,1B38,10C0,F800,1404,4979   ;21021          IR0?,J/NEGA5            ;  TEST OPC-CODE FOR 4/6 OPERANDS
                                ;21022
                                ;21023          ;11-----------------------------;
                                ;21024  =11     ALU_D.ANDNOT.K[.F],Q_ALU,       ;  STRIP OFF SIGN-NIBBLE
U 0973, 0019,1724,61C0,F800,0000,09AB   ;21025          STATE3-0?               ;  TEST 1.TIME BIT
                                ;21026  =;END   ;-------------------------------;
                                ;21027  =1011   ;BRANCH ON 1. TIME BIT OF STATE
                                ;21028          ;1011---------------------------;
                                ;21029          ALU_0-Q,D_ALU,                  ;  NEGATE DATA (BINARY)
                                ;21030          Q_DEC.CON,SET.CC(LONG),         ;  CLOCK C-BIT (OR BORROW, RATHER)
U 09AB, 081F,0000,01D0,F800,0070,0BD5   ;21031          J/NEG.ADJ               ;  TEST ALL 9'S BIT OF STATE
                                ;21032          ;1111---------------------------;
                                ;21033          ALU_0[INST.DEP]D,D_ALU,         ;  NEGATE BINARY WITH BORROW
                                ;21034          SET.CC(LONG),                   ;  GET BORROW IF ANY
U 09AF, 081F,200C,01D0,F800,0070,0BD5   ;21035          Q_DEC.CON               ;  GENERATE DECIMAL CONSTANT
                                ;21036
                                ;21037  NEG.ADJ:;-------------------------------;
                                ;21038          ALU_D-Q,D_ALU,                  ;  DECIMAL ADJUST
U 0BD5, 081D,0000,0180,F800,0000,0148   ;21039          J/NEGA10                ;
```

```
                                    ;21040  =10*****
                                    ;21041  NEGA10: Q_LB,CLK.UBCC,                  ;  CLOCK LENGTH
U 0148, 000C,0039,01C0,F800,0010,0C6B ;21042            CALL,J/WRITE                ;  WRITE OUT RESULT
                                    ;21043          ;---------------------------;
                                    ;21044  =11*****
                                    ;21045          Q_LB,                       ;  GET LENGTH
                                    ;21046          STATE_STATE.ANDNOT.K[.1],   ;  CLEAR CARRY-BIT OF STATE
U 0168, 000C,0238,05C0,F800,1404,4C75 ;21047        ROR?                        ;  TEST PSL-CARRY-BIT
                                    ;21048  =;END   ;---------------------------;
                                    ;21049  =101    ;BRANCH ON PSL-CARRY-BIT
                                    ;21050          ;101------------------------;
                                    ;21051  NEGA2:  ALU_Q+K[.8],R[R15]_ALU,     ;  UPDATE DST-LENGTH
U 0C75, 0019,3B14,0180,FAF8,0000,09BD ;21052        IR0?,J/NEGA3                ;  TEST FOR 4/6 OPERANDS
                                    ;21053          ;111------------------------;
                                    ;21054          ALU_Q+K[.8],R[R15]_ALU,     ;  UPDATE DST-LENGTH
                                    ;21055          STATE_STATE+1,              ;  SET CARRY-BIT OF STATE
U 0C77, 0019,3B14,0180,FAF8,1400,C9BD ;21056        IR0?,J/NEGA3                ;  TEST FOR 4/6 OPERANDS
                                    ;21057  =;END   ;---------------------------;
                                    ;21058  =1101   ;BRANCH ON LOW BIT OF OPCODE
                                    ;21059          ;1101-----------------------;
                                    ;21060  NEGA3:  ALU_LA-K[.4],R[R3]_ALU,LONG, ;  UPDATE ADDRESS
                                    ;21061          STATE_STATE.OR.K[.4],       ;  SET 1.TIM FLAG
U 09BD, 0018,1600,1180,FA98,1404,2C45 ;21062        STATE7-4?,J/NEGATE0         ;  TEST ALL 9'S BIT
                                    ;21063          ;1111-----------------------;
                                    ;21064          ALU_LA-K[.4],R[R5]_ALU,LONG, ;  UPDATE ADDRESS (3-OPERANDS)
                                    ;21065          STATE_STATE.OR.K[.4],       ;  SET 1.TIM FLAG
U 09BF, 0018,1600,1180,FAA8,1404,2C45 ;21066        STATE7-4?,J/NEGATE0         ;  TEST ALL 9'S BIT
                                    ;21067  =;END   ;---------------------------;
                                    ;21068  =01     ;BRANCH ON LOW BIT OF OP-CODE
                                    ;21069          ;01-------------------------;
                                    ;21070  NEGA5:  ALU_LA+Q,R[R3]_ALU,         ;  LOAD R3 WITH LOW DEST-ADDRESS
U 0979, 001C,1714,0180,FA98,0000,0261 ;21071        STATE3-0?,J/ASF4            ;  TEST ADD/SUB AND SIGN-BIT
                                    ;21072          ;11-------------------------;
                                    ;21073          ALU_LA+Q,R[R5]_ALU,         ;  LOAD R5 WITH LOW DEST-ADDRESS
U 097B, 001C,1714,0180,FAA8,0000,0261 ;21074        STATE3-0?,J/ASF4            ;  TEST ADD/SUB AND SIGN-BIT
                                    ;21075  =;END   ;---------------------------;
```

```
                                ;21076
                                ;21077  =0****
                                ;21078  ASF1:   ALU_0(A),R[R2]_ALU,LONG,          ;  CLEAR R2
                                ;21079          STATE_STATE.ANDNOT.K[.4],         ;  CLEAR 1. TIME BIT
U 04CD, 0003,003D,1180,FA90,1404,4BD0   ;21080          CALL,J/REG.AD                     ;  RESET R3
                                ;21081          ;---------------------------------;
U 04DD, 0000,1B3C,0180,F800,0000,03ED   ;21082          IR0?                              ;  TEST FOR 4 OR 6 OPERANDS
                                ;21083          ;---------------------------------;
                                ;21084  =01101  ;BRANCH ON LOW BIT OF OPCODE
                                ;21085  ASF1.X: ;01101-----------------------------;
                                ;21086          LAB_R[R15],ALU_LA,D_ALU.RIGHT,    ;  GET DST-LENGTH
                                ;21087          SET.CC(LONG),SI/ASHR,
U 03ED, 0840,173C,0080,FA78,0070,0260   ;21088          STATE3-0?,J/ASF3                  ;  TEST ADD/SUB-BIT
                                ;21089          ;01111-----------------------------;
                                ;21090          ALU_R[R4],Q_ALU.RIGHT,SI/ASHR,    ;  GET DST-LENGTH
                                ;21091          SC_R[.4],
U 03EF, 0040,003D,10C0,FA20,0084,6BCE   ;21092          CALL,J/REG.ADJ                    ;  RESET R5
                                ;21093  =11111  ;---------------------------------;
                                ;21094          LAB_R[R15],ALU_LA,D_ALU.RIGHT,    ;  GET DST-REGISTER
                                ;21095          SI/ASHR,SET.CC(LONG),             ;  CLEAR CARRY
U 03FF, 0840,173C,0080,FA78,0070,0260   ;21096          STATE3-0?,J/ASF3                  ;  TEST ADD/SUB-BIT
                                ;21097  =;END   ;---------------------------------;
                                ;21098  =0*00   ;8-WAY BRANCH ON ADD/SUB-,SIGN-,AND CARRY-BITS OF STATE
                                ;21099          ;0*00-----------------------------;
                                ;21100  ASF3:   ALU_0(A),N_AMX.Z_TST,             ;  ADD,POSITIVE,NO CARRY
U 0260, 0003,163C,0180,F800,0030,0CA3   ;21101          STATE7-4?,J/ASF7                  ;  TEST OVERFLOW
                                ;21102          ;0*01-----------------------------;
                                ;21103  ASF4:   STATE_STATE.OR.K[.40],            ;  ADD,POSITIVE,CARRY
                                ;21104          ALU_0(A),N_AMX.Z_TST,
U 0261, 0003,003C,3180,F800,1434,2CA7   ;21105          J/ASF8
                                ;21106          ;0*10-----------------------------;
U 0262, 0000,1A3C,0180,F800,0000,09CB   ;21107          PSL.CC?,J/ASF6                    ;  ADD,NEGATIVE,NO CARRY
                                ;21108          ;0*11-----------------------------;
                                ;21109          STATE_STATE.OR.K[.40],            ;  ADD,NEGATIVE,CARRY
U 0263, 0000,1A3C,3180,F800,1404,29CB   ;21110          PSL.CC?,J/ASF6                    ;  TEST Z-BIT
                                ;21111          ;1*00-----------------------------;
                                ;21112          STATE_STATE.OR.K[.2],             ;  SUBTRACT,POSITIVE,BORROW
                                ;21113          ALU_Q-D,R(SC)_ALU,                ;  GET STARTING ADDRESS
                                ;21114          SET.CC(LONG),                     ;  SET CARRY
U 0268, 001D,2000,0980,F8E8,1474,2C47   ;21115          J/NEGATE
                                ;21116          ;1*01-----------------------------;
                                ;21117          ALU_0(A),N_AMX.Z_TST,             ;  SUBTRACT,POSITIVE,NO BORROW
U 0269, 0003,163C,0180,F800,0030,0CA3   ;21118          STATE7-4?,J/ASF7
                                ;21119          ;1*10-----------------------------;
                                ;21120          STATE_STATE.ANDNOT.K[.2],         ;  SUBTRACT,NEGATIVE,BORROW
                                ;21121          ALU_Q-D,R(SC)_ALU,                ;  GET STARTING ADDRESS
                                ;21122          SET.CC(LONG),                     ;  SET CARRY
U 026A, 001D,2000,0980,F8E8,1474,4C47   ;21123          J/NEGATE
                                ;21124          ;1*11-----------------------------;
U 026B, 0000,1A3C,0180,F800,0000,09CB   ;21125          PSL.CC?,J/ASF6                    ;  SUBTRACT,NEGATIVE,NO BORROW
                                ;21126  =;END   ;---------------------------------;
```

```
                                    ;21127
                                    ;-------------------------------;
                                    ;21128  =1011   ;BRANCH ON PSL-Z-BIT
                                    ;21129          ;1011-----------------------;
                                    ;21130  ASF6:   ALU_K[.80],N&Z_ALU.V&C_0,BYTE,  ;  SET PSL-N-BIT
U 09CB, 0018,9638,4180,F800,0050,0CA3 ;21131        STATE7-4?,J/ASF7
                                    ;21132          ;1111-----------------------;
U 09CF, 0000,163C,0180,F800,0000,0C83 ;21133        STATE7-4?
                                    ;21134  =;END   ;-------------------------------;
                                    ;21135  =011    ;BRANCH ON OVERFLOW-BIT OF STATE
                                    ;21136          ;011------------------------;
                                    ;21137          ALU_0(A),N&Z_ALU.V&C_0,         ;  CLEAR N,C,V-BITS OF PSL
U 0C83, 0003,003D,0180,F800,0050,0BD2 ;21138        CAL[,J/SGN.CHANGE               ;  MAKE STRING +0 RATHER THAN -0
                                    ;21139          ;111------------------------;
                                    ;21140          ALU_0(A),N_AMX.Z_TST,           ;  CLEAR N-BIT OF PSL
U 0C87, 0003,003C,0180,F800,2030,0CA7 ;21141        CLR_FPD,J/ASF8
                                    ;21142  =;END   ;-------------------------------;
                                    ;21143  =011    ;BRANCH ON OVERFLOW-BIT OF STATE
                                    ;21144          ;011------------------------;
                                    ;21145  ASF7:   ALU_0+Q,SET.CC(LONG),           ;  CLEAR PSL CARRY-BIT
U 0CA3, 001F,0014,0180,F800,2070,0B2C ;21146        CLR_FPD,J/FINI15                ;  JOIN COMMON FINISH-ROUTINE
                                    ;21147
                                    ;21148          ;111------------------------;
                                    ;21149  ASF8:   Q_ID[CES],ALU_0+Q,              ;  CLEAR CARRY-BIT
U 0CA7, 001F,0014,31F0,2C00,0070,0B26 ;21150        SET.CC(LONG),J/FINI5            ;  LOAD TRAP-VALUE
                                    ;21151  =;END   ;-------------------------------;
```

```
;21152  .TOC    ''      Decimal string        : MULP''
;21153
;21154  ;MULTIPLY BCD-STRINGS
;21155  ;       ROUTINE TO MULTIPLY MULTIPLIER-STRING WITH MULTIPLICAND-STRING,
;21156  ;       STORING THE RESULT IN PRODUCT-STRING.
;21157
;21158  ; ALGORITHM:
;21159  ;       1. FIRST THE SPECIFIERS ARE EVALUATED AND STORED, VARIOUS REGISTERS
;21160  ;       ARE INITIALIZED,(ROUTINE 'MULP.INIT'')
;21161
;21162  ;       2.THEN THE MULTIPLIER IS READ IN ITS ENTIRETY, AND STORED IN
;21163  ;       TEMPORARY REGISTERS RC<0-5>, 3 BYTES PR REGISTER (USING
;21164  ;       'LOAD.MULTIPLIER''-ROUTINE).
;21165
;21166  ;       3. THE PRODUCT IS INITIALIZED TO 0 ('MULSGN''-ROUTINE).
;21167  ;       ACTUALLY, THE FIRST LONGWORD IS NOT CLEARED.
;21168
;21169  ;       4. STARTING AT THE LEAST SIGNIFICANT DIGIT, A BYTE IS READ FROM
;21170  ;       THE MULTIPLICAND STRING ('MULR.1'').
;21171
;21172  ;       5. EACH RC-REGISTER IS MULTIPLIED BY
;21173  ;       THE PAIR OF DIGITS FROM MULTIPLICAND ('MULM'').
;21174
;21175  ;       6. THE RESULT IS ADDED TO THE PARTIAL PRODUCT-STRING,
;21176  ;       (USING 'MURAW''-ROUTINE).
;21177
;21178  ;       7. STEPS 4, 5 AND 6 ARE REPEATED UNTIL THE MULTIPLICAND-STRING
;21179  ;       IS EXHAUSTED, AT WHICH POINT THE GENERAL REGISTERS
;21180  ;       ARE RESET, AND THE CONDITION CODES ARE DETERMINED ('MUL.FIN'').
;21181
;21182  ;       8. IF AN INTERRUPT OR MEMORY-FAULT OCCURS, THE CURRENT STATE
;21183  ;       OF THE INSTRUCTION IS SAVED IN GENERAL REGISTERS,
;21184  ;       ('MULT.MEMORY.FAULT''), AND THE INSTRUCTION RESUMES WHERE
;21185  ;       IT LEFT OFF BY RESTORING THE REGISTERS ('MULP.DIVP.RESTORE'')
;21186
;21187  ;       OP-CODE IS ''25''
;21188  ;       MNEMONIC IS 'MULP''
;21189  ;       INSTRUCTION FORMAT IS:
;21190  ;        opcode mulrlen.rw, mulraddr.ab, muldlen.rw,
;21191  ;                        muldaddr.ab, prodlen.rw, prodaddr.ab
;21192  ;       INSTRUCTION DEPENDENT ALU FUNCTION IS ''A-B-PSL.BORROW''
;21193  ;       INSTRUCTION DEPENDENT CC-CLOCKING IS  Z_Z,N_N,V_0,C_ALU CARRY[UDT]
```

```
;21194
;21195  ; STORAGE ALLOCATION FOR MULTIPLY-INSTRUCTION.
;21196  ;       RC0-RC5 ARE USED TO STORE MULTIPLICAND, 3 BYTES EACH.
;21197  ;       RC6=RC-COUNTER
;21198  ;       RC7=LAST WRITTEN LONGWORD DURING FINISH, NEW PRODUCT DURING READS
;21199  ;       R0=HIGH NIBBLE OF MULTIPLIER-BYTE
;21200  ;       R1=ABSOLUTE PRODUCT-LENGTH, INIT. TO -LENGTH-1
;21201  ;       R2=MULTIPLICAND-LENGTH, INIT. TO LENGTH/2
;21202  ;       R3=MULTIPLICAND-ADDRESS, INIT. TO HIGH ADDRESS
;21203  ;       R4=PRODUCT-LENGTH DURING EACH PASS, INIT. TO -LENGTH-1
;21204  ;       R5=PRODUCT-ADDRESS DURING EACH PASS, INIT. TO HIGH ADDRESS + 1
;21205  ;       ID[T0]=NEXT DIGIT OF MULTIPLICAND
;21206  ;       ID[T2]=66666666 (DECIMAL CONSTANT) DURING MULTIPLICATION
;21207  ;       ID[T3]=ABSOLUTE PRODUCT-ADDRESS, INIT. TO HIGH ADDRESS+2
;21208  ;       ID[T4]=ABSOLUTE PRODUCT-LENGTH, INITIALIZED TO -LENGTH-1
;21209  ;       ID[T6]=MULTIPLIER-ADDRESS, INIT. TO LOW ADDRESS
;21210  ;       ID[T7]=MULTIPLIER-LENGTH, INIT. TO LENGTH
;21211  ;       ID[T8]=HIGH LIMIT FOR RC-COUNT
;21212  ;       STATE-REGISTER IS USED FOR STATUS
;21213  ;       FE=CURRENT DIGIT
;21214  ;       SC,FE,R15,D,Q ARE SCRATCH-REGISTERS
;21215  ;       STATE-REGISTER BIT-ALLOCATION:
;21216          ;-------------------------------------------------------------------
;21217          ;INTRPT ;OVFLOW ;       ;       ;1.READ ;1. WR  ;SIGN   ;HI/LO  ;
;21218          ;       ;       ;       ;       ;       ;       ;       ;DIGIT  ;
;21219          ;       ;       ;       ;       ;       ;       ;       ;       ;
;21220          ;       ;       ;       ;       ;       ;       ;       ;       ;
;21221          ;-------------------------------------------------------------------
;21222
```

```
                                ;21223  3CB:
                                ;21224  MULP.INIT:
                                ;21225          ;ENTER HERE FROM DP2 WITH M'PLIER-LENGTH IN Q,
                                ;21226          ;M'PLIER ADDRESS IN D.
                                ;21227          ;-------------------------------;
                                ;21228          ALU_Q.OXT[WORD],D_ALU,          ;  CLEAR HIGH WORD
                                ;21229          ID[T6]_D,                       ;  SAVE MULTIPLIER-ADDRESS
U 03CB, 0803,603D,D980,3C00,0000,037E   ;21230          CALL,J7SPEC                     ;  EVALUATE M'PLICAND-LENGTH
                                ;21231          ;-------------------------------;
                                ;21232  3DB:    ALU_Q.AND.K[.FFE0],             ;  GET HIGH BITS OF LENGTH
                                ;21233          RC[T6]_ALU,                     ;  CLEAR OUT RC6
                                ;21234          D_Q,Q_D,
U 03DB, 0C19,2034,A1E0,F9B0,0050,0131   ;21235          NEZ_ALU.V&C_0,LONG              ;  SET Z-BIT(UNLESS ERROR)
                                ;21236          ;-------------------------------;
                                ;21237  =01*****
                                ;21238          ID[T7]_D,D_ALU.RIGHT,           ;  STORE MULTIPLIER-LENGTH
                                ;21239          ALU_Q.OXT[WORD],                ;  CLEAR HIGH WORD OF LENGTH
                                ;21240          RC[T0]_ALU.RIGHT,               ;  SAVE BYTE-COUNT
U 0131, 0843,603D,DD80,3D80,0000,047E   ;21241          CALL,J7ASPC                     ;  EVALUATE M'PLICAND ADDRESS
                                ;21242          ;-------------------------------;
                                ;21243  =11*****
                                ;21244          ALU_Q.AND.K[.FFF0],             ;  HIGH BITS OF MULTIPLICAND-LENGTH
U 0171, 0019,2034,6D80,F800,0030,00AA   ;21245          N_OR_X.Z_TST                    ;  OR IT INTO Z-BIT
                                ;21246  =;END   ;-------------------------------;
                                ;21247  =010**1*
                                ;21248  MUL.I1: STATE_K[ZERO],                  ;  INITIALIZE STATE-REGISTER
                                ;21249          ALU_Q.OXT[BYTE]+D,              ;  GENERATE HIGH ADDRESS
                                ;21250          RC[T1]_ALU,                     ;  SAVE HIGH ADDRESS
U 00AA, 001F,A015,1980,F988,1404,637E   ;21251          CALL,J7SPEC                     ;  EVALUATE PRODUCT-LENGTH
                                ;21252          ;-------------------------------;
                                ;21253  =011**1*
                                ;21254          ALU_D.AND.K[.FFE0],             ;  HIGH BITS OF PRODUCT-LENGTH
                                ;21255          N_OR_X.Z_TST,                   ;  OR INTO PSL Z-BIT
U 00BA, 0019,0035,A180,F800,0030,047E   ;21256          CALL,J/ASPC                     ;  EVALUATE PRODUCT-ADDRESS
                                ;21257          ;-------------------------------;
                                ;21258  =111**1*
                                ;21259          ALU_Q,SC_ALU,                   ;  SAVE PRODUCT-LENGTH IN SC
                                ;21260          QK/RIGHT,                       ;  DIVIDE IT BY 2
U 00FA, 0001,3A3C,01B0,F800,0082,0328   ;21261          PSL.CC?,J/MUL.I2                ;  TEST FOR LEGAL LENGTHS
                                ;21262  =;END   ;-------------------------------;
```

```
                                    ;21263          ;--------------------------------;
                                    ;21264  =10**   ;BRANCH ON PSL Z-BIT
                                    ;21265          ;--------------------------------;
U 0328, 0000,003C,0180,F800,0000,0106 ;21266  MUL.I2: J/RSVOPR                       ;  LENGTHS OUT OF RANGE
                                    ;21267          ;--------------------------------;
                                    ;21268          FE_K[.34],                      ;  USE IT FOR ADDRESS LATER
                                    ;21269          LC_RC[T1],                      ;  RETRIEVE HIGH MULTIPLICAND-ADDRESS
                                    ;21270          ALU_Q.OXT[BYTE]+D+1,            ;  GENERATE HIGH PRODUCT-ADDRESS
U 032C, 081F,A010,2980,F908,0104,6BD6 ;21271        D_ALU
                                    ;21272          ;--------------------------------;
                                    ;21273          ALU_LC,R[R3]_ALU,               ;  STORE MULTIPLICAND-ADDRESS
U 0BD6, 0010,0038,CD80,3E98,0000,0BDD ;21274        ID[T3]_D                        ;  STORE PRODUCT-ADDRESS
                                    ;21275          ;--------------------------------;
                                    ;21276          ALU_0-K[SC]-1,                  ;  NEGATIVE PRODUCT-LENGTH
U 0BDD, 001B,0008,1DC0,FB80,0000,0BDE ;21277        LC_RC[T0]&R1_ALU,Q_ALU          ;  R1 GET PRODUCT-LENGTH
                                    ;21278          ;--------------------------------;
                                    ;21279          SC_FE,D_Q,
U 0BDE, 0C19,0000,1180,FAA8,0081,0BE0 ;21280        ALU_D-K[.4],R[R5]_ALU,LONG      ;  INITIALIZE R5 WITH DST-ADDR-4
                                     21281          ;--------------------------------;
                                     21282          ALU_Q+K[.8],R[R4]_ALU,LONG,     ;  INITIALIZE R4 WITH DST-LENGTH+8
U 0BE0, 0019,2014,0180,36A0,0000,0585 ;21283        ID(SC)_D,J/MUL.MUL              ;  STORE LENGTH IN T4
                                    ;21284          ;--------------------------------;
```

```
                                ;21285  =00
                                ;21286  LOAD.MULTIPLIER:        ;ROUTINE WHICH READS MULTIPLIER AND LOADS IT IN
                                ;21287          ;CONSECUTIVE REGISTERS OF RC.
                                ;21288          ;EXPECTS ID[T7]=M'PLIER-LENGTH=Q
                                ;21289          ;ID[T6]=M'PLIER-ADDRESS,
                                ;21290          ;RETURNS RC-LIMIT IN ID[T8] (1 THRU 6)
                                ;21291          ;USES RC7,R15 AS SCRATCH TO HOLD LENGTH AND ADDRESS.
                                ;21292          ;EXPECTS Q TO HAVE SRC-LENGTH
                                ;21293          ;00-------------------------------;
                                ;21294          ALU_Q.OXT[BYTE],                ;  GET M'PLIER-LENGTH
                                ;21295          DK/SHF,CLK.UBCC,
                                ;21296          RC[T7]_ALU.RIGHT,Q_ID[T6],      ;  GET SRC-ADDRESS
U 0990, 0843,A03D,D9F0,2DB8,0010,0BE2   ;21297          CALL,J/MULT.SETFPD              ;  SET FIRST PART DONE-FLAG
                                ;21298          ;01-------------------------------;
                                ;21299  MULT.MEMORY.FAULT:                      ;  FAULT-ROUTINE STARTS HERE
                                ;21300          Q_ID[T6],                       ;  GET MULTIPLIER-ADDRESS
                                ;21301          ALU_R[R1],SC_ALU,               ;  SAVE PRODUCT-LENGTH IN SC
U 0991, 0000,003C,D9F0,2E08,0082,0CA9   ;21302          J/MULT.SAVE
                                ;21303          ;10-------------------------------;
                                ;21304          Q_ID[T6],                       ;  GET MULTIPLIER-ADDRESS
                                ;21305          ALU_R[R1],SC_ALU,               ;  SAVE PRODUCT-LENGTH IN SC
U 0992, 0000,003C,D9F0,2E08,0082,0CA9   ;21306          J/MULT.SAVE                     ;  ROUTINE TO SAVE CONTEXT OF MULP-INST
                                ;21307          ;11-------------------------------;
U 0993, 0C00,003C,01E0,F800,0180,ABE1   ;21308          D_Q,Q_D,SC_SC-FE,FEK/LOAD       ;  CLEAR SC AND FE
                                ;21309  =;END   ;-------------------------------;
U 0BE1, 0C00,003C,B580,3C00,0000,0BE3   ;21310          ID[FPDA]_D,D_Q,J/PL.LL          ;  LOAD FPD-ADDRESS
                                ;21311          ;-------------------------------;
                                ;21312  MULT.SETFPD:
                                ;21313          R[R15]_D+Q+1,FE_SC,             ;  GET HIGH ADDRESS
U 0BE2, 001D,0012,81F0,2EF8,2500,0003   ;21314          SET.FPD,Q_ID[USTACK],RETURN3    ;  GET FPD-ADDRESS FROM U-STACK
                                ;21315          ;-------------------------------;
                                ;21316  PL.LL:   SC_FE,
                                ;21317          Q_D-K[.2],CLK.UBCC,             ;  COMPARE LENGTH WITH 2
U 0BE3, 0019,9B00,09C0,F800,0091,09D7   ;21318          BYTE,ALU?                       ;  TEST LENGTH
                                ;21319          ;-------------------------------;
                                ;21320  =0111   ;BRANCH ON ALU N-BIT
                                ;21321          ;0111-----------------------------;
                                ;21322          Q_D-K[.1],                      ;  ADJUST LENGTH
                                ;21323          LAB_R[R15],                     ;  GET ADDRESS
U 09D7, 0019,1B00,05C0,FA78,0000,09E7   ;21324          BEN/ALU,J/PL.LL1                ;  LOAD STANDARD COUNT
                                ;21325          ;1111-----------------------------;
                                ;21326          D_RC[T0],                       ;  GET SIGN-BYTE
U 09DF, 0810,1738,0180,F900,0000,09F3   ;21327          STATE3-0?,J/PL.LL3              ;  SHOULD WE CHECK SIGNS?
                                ;21328  =;END   ;-------------------------------;
                                ;21329  =0111   ;BRANCH ON ALU N-BIT
                                ;21330          ;0111-----------------------------;
U 09E7, 0F00,003C,0180,F800,0000,099A   ;21331  PL.LL1: D_0,J/PL.LL2                  ;  READ 3 BYTES
                                ;21332          ;1111-----------------------------;
U 09EF, 0819,2020,0580,F800,0000,099A   ;21333          ALU_Q.XOR.K[.1],D_ALU           ;  READ LESS THAN 3 BYTES
                                ;21334  =;END   ;-------------------------------;
```

```
                                    ;21335  =10     ;10----------------------------;
                                    ;21336  PL.LL2: VA_LA-K[.1],                    ;  GET ADDRESS READY
                                    ;21337          CALL,
U 099A, 0018,0C01,0580,F800,0200,0E24 ;21338        MUL?,J/READ2                    ;  CALL READ-BCD-ROUTINE
                                    ;21339          ;11----------------------------;
                                    ;21340          SC_FE,                          ;  GET RC-POINTER
U 099B, 0018,0000,0D80,FAF8,0081,0BE4 ;21341        R[R15]_LA-K[.3]                 ;  UPDATE ADDRESS
                                    ;21342  =;END   ;----------------------------;
U 0BE4, 0001,003C,0180,F838,0000,0BE5 ;21343        ALU_D,RC(SC)_ALU                ;  STORE DATA IN RC
                                    ;21344          ;----------------------------;
U 0BE5, 0810,0038,0180,F938,0100,CBE8 ;21345        D_RC[T7],FE_SC+1                ;  GET LENGTH, INCREMENT RC-POINTER
                                    ;21346          ;----------------------------;
                                    ;21347          ALU_D-K[.3],
                                    ;21348          RC[T7]_ALU,D_ALU,               ;  UPDATE LENGTH
U 0BE8, 0819,8000,0D80,F9B8,0010,0BE3 ;21349        CLK.UBCC,BYTE,J/PL.LL           ;  LOOP BACK TO READ ANOTHER 3 BYTES
                                    ;21350          ;----------------------------;
                                    ;21351  =0011   ;BRANCH ON 1.READ AND 1. WRITE BIT OF STATE
                                    ;21352          ;0011--------------------------;
                                    ;21353  PL.LL3: ALU_D.ANDNOT.K[.F],RC[T0]_ALU,  ;  CLEAR SIGN-NIBBLE
U 09F3, 0019,0F24,6180,F980,0000,09A2 ;21354        BCDSGN?,J/PL.LL4                ;  TEST SIGN OF MULTIPLIER
                                    ;21355          ;0111--------------------------;
                                    ;21356          ALU_D.ANDNOT.K[.F],RC[T0]_ALU,  ;  THIS IS RESTART,
U 09F7, 0019,0024,6180,F980,0000,09A2 ;21357        J/PL.LL4                        ;  SIGNS HAVE BEEN CALCULATED
                                    ;21358          ;1011--------------------------;
                                    ;21359          ALU_D.ANDNOT.K[.F],RC[T0]_ALU,  ;  WHAT A WASTE!
U 09FB, 0019,0024,6180,F980,0000,09A2 ;21360        J/PL.LL4
                                    ;21361          ;1111--------------------------;
                                    ;21362          ALU_D.ANDNOT.K[.F],RC[T0]_ALU,
U 09FF, 0019,0024,6180,F980,0000,09A2 ;21363        J/PL.LL4
                                    ;21364  =;END   ;----------------------------;
                                    ;21365  PL.EX:  Q_R[R2],CLK.UBCC,BYTE,          ;  GET MULTIPLICAND-LENGTH
U 0BE9, 0F00,803E,E1C0,3E10,0010,0010 ;21366        ID[T8]_D,D_0,RETURN10           ;  SAVE RC-LIMIT (1 TO 6)
                                    ;21367          ;----------------------------;
                                    ;21368  =10     ;BRANCH ON DECIMAL SIGN-NIBBLE
                                    ;21369          ;10----------------------------;
U 09A2, 0818,0038,1D80,F800,0000,0BE9 ;21370  PL.LL4: D_K[SC],J/PL.EX                ;  GET NO. OF RC-REGISTERS USED
                                    ;21371          ;11----------------------------;
                                    ;21372          STATE_STATE.OR.K[.2],           ;  SET SIGN-BIT
U 09A3, 0000,003C,0980,F800,1404,29A2 ;21373        J/PL.LL4
                                    ;21374  =;END   ;----------------------------;
```

```
                                ;21375  =0****
                                ;21376  MUL.MUL:        ;START MULTIPLICATION-ROUTINE HERE
                                ;21377                  ;-------------------------------;
                                ;21378          ALU_LC,R[R2] ALU,Q_ID[T7],      ;  STORE MULTIPLICAND-LENGTH
U 0585, 0010,0039,DDF0,2E90,0000,0990   ;21379          CAL[,J/LOAD.MULTIPLIER          ;  LOAD MULTIPLIER INTO RC
                                ;21380                  ;-------------------------------;
                                ;21381          ALU_Q+Q+1,Q_ALU,                ;  ADJUST MULTIPLICAND LENGTH
U 0595, 001F,0010,01C0,F800,0000,0828   ;21382          J/MULR.1
                                ;21383  =;END           ;-------------------------------;
                                ;21384          ;ROUTINE TO READ BYTE FROM MULTIPLICAND.
                                ;21385          ;EXPECTS R2=MULTIPLICAND LENGTH
                                ;21386          ;R3=MULTIPLICAND ADDRESS
                                ;21387          ;USES 1. TIME BIT TO TEST FOR SIGN-BYTE
                                ;21388          ;-------------------------------;
                                ;21389  =0       ;BRANCH ON ALU Z-BIT
                                ;21390          ;0------------------------------;
                                ;21391  MULR.1: VA_R[R3],DK/RIGHT2,             ;  LOAD MULTIPLICAND ADDRESS
                                ;21392          STATE_STATE.ANDNOT.K[.1],       ;  CLEAR HIGH DIGIT BIT
U 0828, 0200,003C,0580,FA18,1604,4BEA   ;21393          J/MULR1
                                ;21394          ;1------------------------------;
                                ;21395          R[R0]_0,DK/RIGHT2,              ;  NO MORE DIGITS
                                ;21396          STATE_STATE.ANDNOT.K[.1],       ;  CLEAR HIGH DIGIT BIT
U 0829, 0203,013C,0580,FA80,1404,482C   ;21397          Z?,J/MULR5                      ;  TEST LAST NIBBLE
                                ;21398  =;END           ;-------------------------------;
                                ;21399  MULR1:  D[BYTE]_CACHE,                  ;  READ NEXT BYTE
U 0BEA, 0019,A000,05E0,4290,0000,0BEB   ;21400          R[R2]_Q-K[.1],Q_D               ;  UPDATE LENGTH
                                ;21401                  ;-------------------------------;
U 0BEB, 0C18,0000,05E0,FA98,0000,0BEC   ;21402          R[R3]_LA-K[.1],Q_D,D_Q          ;  DECREMENT MULTIPLICAND-ADDRESS
                                ;21403                  ;-------------------------------;
                                ;21404          ALU_Q.OXT[BYTE],                ;  ISOLATE NEW DIGITS
                                ;21405          R[R0]_ALU,ID[T0]_D,             ;  STORE BYTE IN R0
U 0BEC, 0C03,B73C,C1E0,3E80,0000,0566   ;21406          Q_D,D_Q,STATE3-0?               ;  TEST FOR 1. TIME READ
                                ;21407                  ;-------------------------------;
                                ;21408  =011*   ;BRANCH ON 1. TIME BIT OF STATE
                                ;21409          ;011*----------------------------;
                                ;21410  MULSGN0:
                                ;21411          Q_R[R4],CLK.UBCC,BYTE,          ;  GET PRODUCT-LENGTH
U 0566, 0000,8F3C,01C0,FA20,0010,0312   ;21412          B[DSGN?,J/MULSGN                ;  CHECK SIGN-NIBBLE OF MULTIPLICAND
                                ;21413          ;111*----------------------------;
                                ;21414          ALU_Q,SC_ALU,                   ;  GET PREVIOUS NIBBLE
U 056E, 0001,203C,0180,FA08,0082,0BEE   ;21415          LAB_R[R1],J/MULPUP              ;  GET PRODUCT LENGTH
                                ;21416  =;END           ;-------------------------------;
                                ;21417  =0       ;BRANCH ON ALU Z-BIT
                                ;21418          ;0------------------------------;
                                ;21419  MULR5:  ALU_D,SC_ALU,ID[T0]_D,          ;  SAVE PREVIOUS NIBBLE IN SC AND T0
U 082C, 0001,003C,C180,3E08,0082,0BEE   ;21420          LAB_R[R1],J/MULPUP              ;  GET PRODUCT-LENGTH
                                ;21421          ;1------------------------------;
                                ;21422          ALU_D,SC_ALU,ID[T0]_D,          ;  SAVE PREVIOUS NIBBLE IN SC AND T0
U 082D, 0001,173C,C180,3E18,0082,01EA   ;21423          LAB_R[R3],STATE3-0?,J/MUL.FIN   ;  GET MULTIPLICAND-ADDR., TEST 1. TIME
                                ;21424  =;END           ;-------------------------------;
```

```
                                    ;21425  MULPUP: ;ENTER HERE AFTER READING A PAIR OF DIGITS FROM MULTIPLICAND.
                                    ;21426          ;UPDATE ABSOLUTE PRODUCT PARAMETERS.
                                    ;21427          ;DURING EACH PASS THROUGH THE PRODUCT STRING,
                                    ;21428          ;USE R4 AND R5 AS POINTERS, AND INITIALIZE THEM BY
                                    ;21429          ;R1 AND ID[T3].
                                    ;21430
                                    ;21431          ;-------------------------------;
                                    ;21432          Q_ID[T3],                       ;  GET PRODUCT ADDRESS
U 0BEE, 0000,003C,CDF0,2EA0,0000,0BF0 ;21433        ALU_LA,R[R4]_ALU,LONG           ;  GET PRODUCT-LENGTH
                                    ;21434          ;-------------------------------;
                                    ;21435          ALU_Q-K[.1],                    ;  UPDATE ADDRESS
U 0BF0, 0819,2000,0580,F800,0000,0BF1 ;21436        D_ALU
                                    ;21437          ;-------------------------------;
                                    ;21438          ID[T3]_D,                       ;  SAVE ADDRESS
U 0BF1, 0001,203C,CD80,3EA8,0000,0BF2 ;21439        R[R5]_Q,LONG                    ;  SAVE LENGTH
                                    ;21440          ;-------------------------------;
U 0BF2, 0018,0014,0980,FA88,0000,0BF3 ;21441        ALU_LA+K[.2],R[R1]_ALU          ;  UPDATE LENGTH
                                    ;21442          ;-------------------------------;
                                    ;21443          INTRPT.STROBE,                  ;  STROBE FOR INTERRUPTS
U 0BF3, 0018,0038,0580,F980,4000,0BF4 ;21444        ALU_K[.1],RC[T6]_ALU            ;  INITIALIZE DIVISOR-COUNT
                                    ;21445          ;-------------------------------;
                                    ;21446          STATE_STATE.ANDNOT.K[.10],      ;  CLEAR CARRY-BIT OF STATE
                                    ;21447          ALU_R[T0],                      ;  GET 1. DIGIT
                                    ;21448          D_ALU.LEFT2,
U 0BF4, 0870,0E38,65F8,F900,1404,4CE6 ;21449        Q_0,BEN/INTERRUPT,J/MULM03      ;  TEST FOR INTERRUPTS
                                    ;21450          ;-------------------------------;
                                    ;21451
```

```
                                ;21452          ;ROUTINE TO SET CONDITION-CODES FOR MULP-INSTRUCTION
                                ;21453  =0101*  ;BRANCH ON 1. TIME BIT OF STATE
                                ;21454  MUL.FIN:
                                ;21455          ;0101*-------------------------;
U 01EA, 0000,003C,0180,FA08,0000,0BEE   ;21456          LAB_R[R1],J/MULPUP      ;  SAVE PREVIOUS NIBBLE IN SC
                                ;21457          ;0111*-------------------------;
                                ;21458  =0111*  ALU_0+LB+1,R[R3]_ALU,     ;  UPDATE R3
U 01EE, 000F,0011,1180,FA98,0084,6BCD   ;21459          SC_R[.4],CALL,J/REG.ADJUST      ;  ADJUST REGISTERS 4 AND 5
                                ;21460
                                ;21461  ; ************************************************
                                ;21462  ; * Patch no. 066, PCS 01EE trapped to WCS 118C *
                                ;21463  ; ************************************************
                                ;21464
                                ;21465          ;1111*-------------------------;
U 01FE, 0010,0038,0180,F938,0030,0BF5   ;21466  =1111*  ALU_RC[T7],N_AMX.Z_TST      ;  TEST LAST LONGWORD FOR 0
                                ;21467  =;END   ;-------------------------;
                                ;21468          ALU_0(A),R[R0]_ALU,LONG,        ;  CLEAR R0
                                ;21469          N_AMX.Z_TST,                    ;  CLEAR N-BIT
                                ;21470          Q_ID[T6],                       ;  GET MULTIPLICAND-ADDRESS
U 0BF5, 0003,173C,D9F0,2E80,0030,012C   ;21471          STATE3-0?                       ;  TEST SIGN-BIT
                                ;21472          ;-------------------------;
                                ;21473  =110*
                                ;21474  MULP.ECO.4:
                                ;21475          ;110*-------------------------;BRANCH ON SIGN-BIT OF STATE
                                ;21476          R[R1]_Q,LONG,                   ;  LOAD IT IN R1
U 012C, 0001,363C,0180,FA88,0000,035A   ;21477          STATE7-4?,J/MUL.F.PLUS          ;  POSITIVE, TEST OVERFLOW-BIT
                                ;21478          ;111*-------------------------;
                                ;21479          R[R1]_Q,LONG,                   ;  LOAD IT IN R1
U 012E, 0001,3A3C,0180,FA88,0000,0A0B   ;21480          PSL.CC?                         ;  TEST Z-BIT
                                ;21481  =;END   ;-------------------------;
                                ;21482  =1011   ;BRANCH ON PSL Z-BIT
                                ;21483          ;1011-------------------------;
                                ;21484          ALU_K[.80],N&Z_ALU.V&C_0,BYTE,  ;  SET N-BIT
U 0A0B, 0018,9638,4180,F800,0050,035A   ;21485          STATE7-4?,J/MUL.F.PLUS          ;  TEST FOR OVERFLOW
                                ;21486          ;1111-------------------------;
                                ;21487          ALU_0(A),R[R2]_ALU,             ;  CLEAR R2
U 0A0F, 0003,163C,D1F0,2E90,0000,033A   ;21488          Q_ID[T4],STATE7-4?              ;  GET DST-LENGTH, TEST OVERFLOW
                                ;21489  =;END   ;-------------------------;
                                ;21490  =01*    ;BRANCH ON OVERFLOW-BIT OF STATE
                                ;21491          ;01*--------------------------;
                                ;21492  MUL.F2: STATE_K[.1],                    ;  FOR RESTART
                                ;21493          ALU_Q.SXT[BYTE],Q_ALU.RIGHT,
U 033A, 0042,A03C,04C0,F800,1404,698F   ;21494          SI/ASHR,J/SGN.C10               ;  CHANGE SIGN
                                ;21495          ;11*--------------------------;
U 033E, 0000,003C,31F0,2C00,0020,0B26   ;21496          SET.V,Q_ID[CES],J/FINI5         ;  GET ID[CES]
                                ;21497  =;END   ;-------------------------;
                                ;21498  =01*    ;BRANCH ON OVERFLOW-BIT OF STATE
                                ;21499          ;01*--------------------------;
                                ;21500  MUL.F.PLUS:
                                ;21501          ALU_0(A),R[R2]_ALU,             ;  CLEAR R2
U 035A, 0003,003C,0180,FA90,2000,0B2C   ;21502          CLR.FPD,J/FINI5                 ;  CLEAR FIRST PART DONE-FLAG
                                ;21503          ;11*--------------------------;
                                ;21504          ALU_0(A),R[R2]_ALU,             ;  CLEAR R2
U 035E, 0003,003C,31F0,2E90,0020,0B26   ;21505          SET.V,Q_ID[CES],J/FINI5         ;  GET ID[CES]
                                ;21506  =;END
```

```
                                    ;21507          ;ROUTINE WHICH SETS THE SIGN-BIT,DEPENDING ON
                                    ;21508          ;LOW NIBBLE OF D, AND SIGN-NIBBLE IN RC0.
                                    ;21509          ;IT CLEARS OUT SIGN-NIBBLE IN RC0, AND CLEARS OUT PRODUCT-STRING.
                                    ;21510          ;THE SIGN-NIBBLE IN THE PRODUCT WILL BE GENERATED BY THE
                                    ;21511          ;WRITE-ROUTINE, WHEN IT SEES THE 1.TIME BIT SET.
                                    ;21512          ;-------------------------------;
                                    ;21513  =00***10        ;BRANCH ON BCD-SIGN
                                    ;21514          ;-------------------------------;
                                    ;21515  MULSGN: STATE_STATE.OR.K[.4],           ;  SET 1. WRITE-BIT
                                    ;21516          ALU_0(A),N&Z_ALU.V&C_0,         ;  SET Z-BIT
U C312, 0F03,003D,1180,F8A8,1454,2C6B   ;21517          LA_RA[R5],D_0,CALL,J7WRITE      ;  WRITE 0
                                    ;21518          ;-------------------------------;
                                    ;21519          STATE_STATE+K[.2],              ;  COMPLEMENT SIGN-BIT
                                    ;21520          Q_R[R4],CLK.UBCC,               ;  START OF LOOP TO CLEAR OUT
U 0313, 0000,003C,09C0,FA20,1414,8312   ;21521          J7MULSGN
                                    ;21522          ;-------------------------------;
                                    ;21523  =01***10
                                    ;21524          STATE_STATE+K[.4],              ;  CLEAR 1.WRITE, SET 1. READ
                                    ;21525          D_R[R2],CLK.UBCC,BYTE,          ;  GET MULTIPLICAND-LENGTH
U 0332, 0800,803C,1180,FA10,1414,8BFD   ;21526          J7MULM00                        ;  FINISHED, READ ANOTHER BYTE
                                    ;21527          ;-------------------------------;
                                    ;21528  =11***10
U 0372, 0019,2014,0180,FAA0,0000,0BF8   ;21529          R[R4]_Q+K[.8]                   ;  UPDATE LENGTH
                                    ;21530  =;END   ;-------------------------------;
                                    ;21531          R[R5]_LA-K[.4],D_0,
U 0BF8, 0F18,0000,1180,FAA8,0000,0566   ;21532          J/MULSGN0                       ;  UPDATE ADDRESS
                                    ;21533          ;-------------------------------;
```

```
                                    ;21534  MULM:   ;ROUTINE WHICH MULTIPLIES NEXT RC-REGISTER BY D.GITS IN
                                    ;21535          ;ID[T0] AND R0 (LOW NIBBLE). LEAVES RESULT IN RC7.
                                    ;21536          ;CHECKS FOR LAST RC-REGISTER.
                                    ;21537          ;R15 IS USED FOR SCRATCH.
                                    ;21538          ;ID[T8] HAS UPPER LIMIT FOR RC-POINTER
                                    ;21539          ;-------------------------------;
                                    ;21540          ALU_RC[T6],SC_ALU,              ;  GET RC-POINTER
U 0BF9, 0810,0038,E1F0,2D30,0082,0BFA ;21541        D_RC[T6],Q_ID[T8]               ;  GET RC-LIMIT
                                    ;21542          ;-------------------------------;
                                    ;21543          ALU_D-Q,CLK.UBCC,               ;  COMPARE WITH UPPER LIMIT
U 0BFA, 001D,8000,0180,F800,0010,0BFC ;21544        BYTE,J/MULM0
                                    ;21545          ;-------------------------------;
                                    ;21546  MULM0:  LC_RC(SC),SC_SC+1,              ;  GET DATA, INCREMENT POINTER
                                    ;21547          ALU_LC,D_ALU.LEFT2,             ;  START MULTIPLYING BY 10.
                                    ;21548          Q_ID[T0],                       ;  GET FIRST DIGIT IN M'CAND
U 0BFC, 0870,0138,C1F0,2C30,0080,C830 ;21549        Z?,J/MULM01                     ;  TEST FOR END OF M'PLIER
                                    ;21550          ;-------------------------------;
                                    ;21551  =0      ;BRANCH ON ALU Z-BIT
                                    ;21552          ;0------------------------------;
                                    ;21553  MULM01: ALU_K[SC],RC[T6]_ALU,           ;  UPDATE RC-POINTER
U 0830, 0018,0038,1D80,F9B0,4000,0BFE ;21554        INTRPT.STROBE,J/MULM02          ;  STROBE INTERRUPTS
                                    ;21555          ;1------------------------------;
U 0831, 0800,803C,0180,FA10,0010,0BFD ;21556        D_R[R2],CLK.UBCC,BYTE           ;  GET M'CAND-LENGTH
                                    ;21557  =:END   ;-------------------------------;
                                    ;21558  MULM00: ALU_R[R0].AND.K[.F0],           ;  GET HIGH DIGIT OF M'CAND
                                    ;21559          Q_D,D_ALU.RIGHT2,               ;  SHIFT IT RIGHT TWICE
U 0BFD, 0898,0134,CDE0,FA00,0010,0828 ;21560        CLK.UBCC,Z?,J/MULR.1            ;  READ ANOTHER BYTE OF M'CAND
                                    ;21561          ;-------------------------------;
                                    ;21562  MULM02: STATE_STATE.ANDNOT.K[.1],        ;  CLEAR HIGH-DIGIT BIT
U 0BFE, 0001,2E3C,05F8,F800,1486,4CE6 ;21563        SC_Q,Q_0,BEN/INTERRUPT          ;  TEST FOR PENDING INTERRUPTS
                                    ;21564          ;-------------------------------;
                                    ;21565  =110    ;BRANCH ON INTERRUPT REQUEST
                                    ;21566          ;110----------------------------;
                                    ;21567  MULM03: EALU_SC-K[.4],DK/LEFT2,         ;  COMPARE FOR ADD OR SUBTRACT
                                    ;21568          ALU_0+Q,R[R15]_ALU,LONG,        ;  CLEAR R15 (Q=0)
                                    ;21569          CLK.UBCC,Q_DEC.CON,             ;  D GETS 10.*OPERAND
U 0CE6, 011F,0C14,11D0,FAF8,0014,AD13 ;21570        SC.NE.0?,J/MULM1                ;  TEST DIGIT FOR 0
                                    ;21571          ;111----------------------------;
                                    ;21572  MUL.INTERRUPT:
                                    ;21573          STATE_STATE.OR.K[.80],          ;  SET INTERRUPT-BIT OF STATE
U 0CE7, 0000,003C,4180,F800,1404,2991 ;21574        J/MULT.MEMORY.FAULT
                                    ;21575  =:END
                                    ;21576
                                    ;21577  ; ************************************************
                                    ;21578  ; * Patch no. 026, PCS 0CE7 trapped to WCS 115D *
                                    ;21579  ; ************************************************
```

```
                                    ;21580          ;CONTINUATION OF LOOP TO MULTIPLY LONGWORD FROM DIVISOR WITH
                                    ;21581          ;PAIR OF DIGITS FROM MULTIPLICAND.
                                    ;21582
                                    ;21583  =011    ;BRANCH ON SC NE 0
                                    ;21584          ;011-------------------------------;
                                    ;21585  MULM1:  ALU_0+LC,D_ALU,                   ;  GET RC-REGISTER TO BE MULTIPLIED
                                    ;21586          LAB_R[R15],Q_DEC.CON,             ;  PRODUCT GETS 0, Q GETS 6'S
U 0D13, 0813,0014,01D0,FA78,0000,0D33   ;21587          J/MULA1
                                    ;21588          ;111-------------------------------;
                                    ;21589          Q_D,D_Q,LAB_R[R15],               ;  R15 HAS PARTIAL PRODUCT
                                    ;21590          SC_SC-K[.1],CLK.UBCC,             ;  ADJUST DIGIT, CLOCK IT
U 0D17, 0C00,123C,05E0,FA78,0094,AA13   ;21591          BEN/EALU,J/MULM2                  ;  TEST LOW DIGIT FOR >=4
                                    ;21592  =;END   ;-------------------------------;
                                    ;21593  =011    ;BRANCH ON SC NE 0
                                    ;21594          ;011-------------------------------;
                                    ;21595  MULTWO: ALU_LA,R[R15]_ALU,D_ALU,          ;  GET PARTIAL PRODEUCT
                                    ;21596          STATE_STATE.OR.K[.1],             ;  SET HIGH DIGIT-BIT
                                    ;21597          Q_0,                              ;  FOR NEXT INSTRUCTION
                                    ;21598          STATE3-0?,                        ;  TEST 1. TIME BIT
U 0D23, 0800,173C,05F8,FAF8,1404,2A3B   ;21599          J/MURAW                           ;  R15 HAS PRODUCT
                                    ;21600          ;111-------------------------------;
                                    ;21601          SC_SC-K[.1],CLK.UBCC,             ;  ADJUST SC IN CASE IT IS 1
                                    ;21602          ALU_D,SHF/ALU.DT,LONG,            ;  MULTIPLY BY 4
                                    ;21603          D_Q,QK/SHF,LAB_R[R15],            ;  D GETS 6'S, Q HAS MULTIPLIER*10.
U 0D27, 0C61,123C,05C0,FA78,0094,AA13   ;21604          EALU?,J/MULM2                     ;  TEST FOR DIGIT >= 4
                                    ;21605  =;END   ;-------------------------------;
```

```
                                    ;21606          ;STILL PART OF LOOP TO MULTIPLY MULTIPLIER LONGWORD BY PAIR
                                    ;21607          ;OF DIGITS FROM MULTIPLICAND.
                                    ;21608
                                    ;21609  =0011   ;BRANCH ON EALU N-BIT AND Z-BIT
                                    ;21610          ;0011-----------------------------;
                                    ;21611  MULM2:  D_Q,Q_LC,
                                    ;21612          R[R15]_ALU,LONG,                    ;  INITIALIZE R15 WITH LONGWORD
U 0A13, 0C10,0038,C9C0,3EF8,0000,0A2F ;21613        ID[T2]_D,J/MULSU1                   ;  DO A SUBTRACT (DIGIT > 4)
                                    ;21614          ;0111-----------------------------;
                                    ;21615          ID[T2]_D,                           ;  SAVE ALL 6,S IN T2
                                    ;21616          R[R15]_ALU,ALU_0+LC,D_ALU,
U 0A17, 0813,0C14,C9D0,3EF8,0000,0D33 ;21617        Q_DEC.CON,SC.NE.0?,J/MULA1          ;  ADD IF DIGIT IS 4
                                    ;21618          ;1011-----------------------------;
                                    ;21619          ID[T2]_D,                           ;  SAVE ALL 6,S IN T2
                                    ;21620          R[R15]_ALU,ALU_0+LC,D_ALU,
U 0A1B, 0813,0C14,C9D0,3EF8,0000,0D33 ;21621        Q_DEC.CON,SC.NE.0?,J/MULA1          ;  ADD IF DIGIT IS 1,2,OR 3
                                    ;21622  =;END   ;-----------------------------;
                                    ;21623  =011    ;BRANCH ON SC NE 0
                                    ;21624          ;011------------------------------;
                                    ;21625  MULA1:  LA_RA[R0],ALU_LA.AND.K[.F],
                                    ;21626          SC_ALU,STATE_STATE+1,               ;  GET NEXT DIGIT, SET HIGH DIGIT-BIT
                                    ;21627          DK/LEFT2,                           ;  SHIFT LEFT, IN CASE OF HIGH DIGIT
U 0D33, 0118,1734,6180,F880,1482,CA1E ;21628        STATE3-0?,J/MULSU0                  ;  FINISHED, D HAS PRODUCT
                                    ;21629          ;111------------------------------;
U 0D37, 081D,0014,0580,F800,0084,AC0G ;21630        D_D+Q,SC_SC-K[.1]                   ;  ADD IN THE 6'S,DECREMENT DIGIT
                                    ;21631  =;END   ;-----------------------------;
U 0C00, 0811,0014,01D0,F800,0000,0C01 ;21632        D_D+LC,Q_DEC.CON                    ;  ADD OPERANDS
                                    ;21633          ;-----------------------------;
                                    ;21634          D_D-Q,Q_ID[T2],                     ;  DECIMAL ADJUST, GET 6'S
                                    ;21635          R[R15]_ALU,LONG,                    ;  SAVE IT IN R15
U 0C01, 081D,0C00,C9F0,2EF8,0000,0D33 ;21636        SC.NE.0?,J/MULA1                    ;  TEST DIGIT
                                    ;21637          ;-----------------------------;
                                    ;21638  =0111   ;BRANCH ON EALU N-BIT
                                    ;21639          ;0111-----------------------------;
                                    ;21640  MULSUB: LA_RA[R0],ALU_LA.AND.K[.F],         ;  GET NEXT DIGIT
                                    ;21641          SC_ALU,STATE_STATE+1,               ;  SET HIGH DIGIT-BIT
                                    ;21642          DK/LEFT2,STATE3-0?,                 ;  SHIFT PREVIOUS PRODUCT
U 0A27, 0118,1734,6180,F880,1482,CA1E ;21643        J/MULSU0
                                    ;21644          ;1111-----------------------------;
                                    ;21645  MULSU1: EALU_SC-K[.8],CLK.UBCC,             ;  COMPARE COUNT WITH 8
U 0A2F, 0811,0000,01D0,F800,0014,AC02 ;21646        D_D-LC,Q_DEC.CON                    ;  SUBTRACT OPERANDS
                                    ;21647  =;END   ;-----------------------------;
                                    ;21648          SC_SC+1,                            ;  SUBTRACT OPERANDS
                                    ;21649          ALU_D-Q,R[R15]_ALU,D_ALU,           ;  DECIMAL ADJUST
                                    ;21650          Q_ID[T2],                           ;  GET ALL 6'S
U 0C02, 081D,1200,C9F0,2EF8,0080,CA27 ;21651        EALU?,J/MULSUB                      ;  TEST FOR DIGIT=0
                                    ;21652          ;-----------------------------;
```

```
                                    ;21653  =1110    ;BRANCH ON HIGH/LOW-BIT OF STATE
                                    ;21654          ;1110----------------------------;
                                    ;21655  MULSU0: EALU_SC-K[.4],CLK.UBCC,          ;  COMPARE DIGIT WITH 4
                                    ;21656          ALU_0+LC,                        ;  LC HAS LONGWORD OF MULTIPLIER
                                    ;21657          D_ALU.LEFT2,                     ;  D GETS MULTIPLIER*4
                                    ;21658          Q_DEC.CON,LAB_R[R15],            ;  GET PREVIOUS PRODUCT
U 0A1E, 0873,0C14,11D0,FA78,0014,AD23   ;21659          SC.NE.0?,J/MULTWO                ;  DO THE HIGH DIGIT AS WELL
                                    ;21660          ;1111----------------------------;
                                    ;21661          ALU_Q+LB,Q_ALU,                  ;  ADD THIS PRODUCT TO PREVIOUS ONE
                                    ;21662          DK/LEFT2,                        ;  D HAS NOW BEEN MULTIPLD. BY 10
U 0A1F, 010D,2014,05C0,F800,1404,AC03   ;21663          STATE_STATE-K[.1]                ;  COMPENSATE FOR PREVIOUS ADD
                                    ;21664  =;END   ;----------------------------;
                                    ;21665          STATE_STATE.OR.K[.1],            ;  SET HIGH NIBBLE BIT
U 0C03, 081D,0014,05D0,F800,1404,2C04   ;21666          D_D+Q,Q_DEC.CON                  ;  ACTUAL ADD
                                    ;21667          ;----------------------------;
                                    ;21668          ALU_D-Q,                         ;  DECIMAL ADJUST
                                    ;21669          R[RT5]_ALU,D_ALU,Q_0,            ;  STORE IN R15
U 0C04, 081D,1700,01F8,FAF8,0000,0A3B   ;21670          STATE3-0?,J/MURAW                ;  ADD IT INTO PRODUCT-STRING
                                    ;21671          ;----------------------------;
```

```
                                ;21672  ;ROUTINE TO READ A LONGWORD FROM PRODUCT, ADD IT TO D (DECIMAL ADD),
                                ;21673  ;AND WRITE OUT THE RESULT IN THE SAME LONGWORD-LOCATION. IT USES ADDRESS
                                ;21674  ;AND LENGTH IN R5 AND R4, AND UPDATES EACH OF THEM BY 3 AFTER THE WRITE.
                                ;21675
                                ;21676  =1011
                                ;21677  MURAW:  ;1011----------------------------;  BRANCH ON 1. TIME BIT OF STATE
U 0A3B, 081D,0000,0180,FA20,0000,0152   ;21678          LAB_R[R4],D_D-Q,J/MURW3          ;  GET PRODUCT-LENGTH
                                ;21679
                                ;21680          ;1111--------------------------;
                                ;21681          ALU_RL.4],D_ALU.RIGHT,           ;  GET PRODUCT-LENGTH
                                ;21682          SI/ASHR,CLK.UBCC,BYTE,           ;  GET LENGTH
U 0A3F, 0840,803C,0080,FA20,0010,09BA   ;21683          J/MURW0                          ;
                                ;21684
                                ;21685  =10
                                ;21686  MURW0:  ;10-----------------------------;
                                ;21687          LA_RA[R5],                       ;  PRODUCT-ADDRESS
                                ;21688          ALU_D+K[.3],D_ALU,               ;  CLOCK LENGTH
                                ;21689          CLK.UBCC,BYTE,                   ;
U 09BA, 0819,9B15,0D80,F8A8,0010,0B17   ;21690          ALU?,CALL,J/READ0W               ;  READ-SUBROUTINE (W/WRITE-CHECK)
                                ;21691
                                ;21692          ;11----------------------------;
                                ;21693          D_D+Q,LAB_R[R15],                ;  GET CURRENT PRODUCT IN R15
U 09BB, 081D,0214,4980,FA78,0000,0D45   ;21694          K[.FF],ROR?                      ;  ADD IN 0'S, TEST FOR CARRY
                                ;21695
                                ;21696  =101    ;101----------------------------;  BRANCH ON PSL CARRY-BIT
                                ;21697          D_D+LB,Q_DEC.CON,                ;  ADD INTO PARTIAL PRODUCT
U 0D45, 080D,0014,01D0,F800,0070,0A3B   ;21698          SET.CC(LONG),J/MURAW             ;  CLOCK PSL-CARRY
                                ;21699
                                ;21700          ;111---------------------------;
U 0D47, 0018,0010,49C0,F800,0000,0C05   ;21701          ALU_LA+K[.FF]+1,Q_ALU            ;  ADD CARRY INTO PARIAL PRODUCT
                                ;21702
                                ;21703  ; ************************************************
                                ;21704  ; * Patch no. 074, PCS 0D47 trapped to WCS 1192 *
                                ;21705  ; ************************************************
                                ;21706
                                ;21707          ;------ ----------------------;
                                ;21708          D_D+Q,Q_DEC.CON,                 ;  DECIMAL ADD, GET 6'S FOR ADJUSTMENT
U 0C05, 081D,0014,01D0,F800,0070,0A3B   ;21709          SET.CC(LONG),J/MURAW             ;  CLOCK C-BIT
                                ;21710
                                ;21711  =10*****
                                ;21712  MURW3:  ;10*****-------------------------;
                                ;21713          Q_LB,LA_RA[R5],                  ;  GET PRODUCT-LENGTH AND PRODUCT-ADDR
                                ;21714          CLK.UBCC,BYTE,                   ;  CLOCK LENGTH
U 0152, 000C,8039,01C0,F8A8,0010,0C79   ;21715          CALL,J/WRITE.MUL                 ;  WRITE SUBROUTINE
                                ;21716
                                ;21717          ;11*****-----------------------;
U 0172, 0018,0000,0D80,FAA8,0000,0C08   ;21718          R[R5]_LA-K[.3]                   ;  UPDATE ADDRESS
                                ;21719
                                ;21720          ;---------------------------;
U 0C08, 0018,0038,D5C0,FA20,0000,0C09   ;21721          LAB_R[R4],Q_K[.6]                ;  GET PRODUCT-LENGTH
                                ;21722
                                ;21723          ;------------------ ----------;
                                ;21724          STATE_STATE.OR.K[.4],            ;  SET 1. WRITE-BIT
                                ;21725          ALU_LA+Q,R[R4]_ALU.LONG,         ;  UPDATE PRODUCT-LENGTH
U 0C09, 001C,0014,1180,FAA0,1404,2BF9   ;21726          J/MULM                           ;
```

```
;21727  .TOC    ''      Decimal string        : DIVP''
;21728
;21729  ;DECIMAL DIVIDE
;21730  ;ALGORITHM:
;21731  ;       1. FIRST THE SPECIFIERS ARE EVALUATED ('DIVP.INIT'),
;21732  ;       AND STORED IN VARIOUS REGISTERS.
;21733
;21734  ;       2. USING THE SUBROUTINE 'DIVDR'' FIRST-PART-DONE-FLAG
;21735  ;       IS SET ('DIVFPD''), AND THE DIVISOR IS READ IN ITS
;21736  ;       ENTIRETY AND STORED IN RC-REGISTERS 0-3. THE DIVISOR
;21737  ;       IS LEFT-ADJUSTED, SO THAT THE HIGH NIBBLE OF RC0
;21738  ;       CONTAINS THE FIRST NON-ZERO DIGIT.
;21739  ;       IN THE PROCESS, THE DIVISOR IS CHECKED FOR ZERO-NESS ('DIVERR'').
;21740
;21741  ;       3. USING THE SUBROUTINE 'DIVND'', WE READ THE
;21742  ;       DIVIDEND IN ITS ENTIRETY, AND STORE IT IN ID-REGISTERS T0-T3.
;21743  ;       IT IS STORED ON THE STACK AS WELL, USING THE FOUR FIRST LONGWORDS.
;21744  ;       IN CASE OF A MEMORY FAULT, STEP 2 AND SOMETIMES STEP 3 IS REPEATED.
;21745
;21746  ;       4. THE ROUTINE 'DIVC1'' CONTROLS THE EXECUTION OF THE MAIN LOOP.
;21747  ;       FIRST THE LENGTHS OF THE 3 OPERANDS ARE COMPARED, ('DIVC10''),
;21748  ;       AND A DECISION IS MADE AS TO WHETHER WE GENERATE A LEADING 0,
;21749  ;       ('DIVC11''), AN OVERFLOW DIGIT ('DIVC4''), OR A REAL DIGIT ('DIVC2'').
;21750
;21751  ;       5. THE DIGIT IS CALCULATED USING A RESTORING ALGORITHM,
;21752  ;       I.E. BY REPEATED SUBTRACTION OF THE DIVISOR FROM THE UPPER PORTION
;21753  ;       OF THE DIVIDEND, ('DVSUB''), UNTIL A BORROW RESULTS FROM THE MOST
;21754  ;       SIGNIFICANT DIGIT, AT WHICH POINT IT IS ADDED BACK IN ONCE ('DVADO'').
;21755
;21756  ;       6. AFTER FINDING THE QUOTIENT DIGIT, WE SHIFT THE DIVIDEND ONE
;21757  ;       DIGIT LEFT AND STORE IT BOTH IN THE ID-BUS REGISTERS AND
;21758  ;       ON THE STACK. ('DIVST'').
;21759
;21760  ;       7. FINALLY, THE ROUTINE 'DIVSAV'' TAKES THE DIGIT JUST GENERATED
;21761  ;       IN RC5 AND EITHER SHIFTS IT INTO THE HIGH NIBBLE, OR WRITES THE
;21762  ;       BYTE CONTAINING IT INTO THE QUOTIENT-STRING ('DIVS01'').
;21763  ;       IF THIS IS THE SIGN-BYTE, THE REGISTERS ARE RESET, AND WE CLOCK THE
;21764  ;       CONDITION CODES ('DIVFIN'').
;21765
;21766  ;       8. IN CASE OF A MEMORY FAULT OR INTERRUPT, THE CURRENT STATE
;21767  ;       OF THE INSTRUCTION IS SAVED IN GENERAL REGISTER R0-R6,
;21768  ;       AND THE INSTRUCTION RESUMES WHERE IT LEFT OFF.
;21769  ;       THE DIVISOR IS READ BAC IN, AND THE DIVIDEND IS RECOVERED FROM
;21770  ;       THE STACK ('DIV.R4'').
```

```
;21771  ;STORAGE ALLOCATION:
;21772  ;       RC0,RC1,RC2,RC3 ARE USED TO STORE DIVISOR
;21773  ;       ID(T0),ID(T1),ID(T2),ID(T3) ARE USED TO STORE DIVIDEND
;21774  ;       STACK IS USED TO SAVE DIVIDEND IN CASE OF INTERRUPTS
;21775  ;       ID[T5] HAS # OF NON-ZERO BYTES IN DIVISOR
;21776  ;       ID[T6] HAS LOW DIVISOR-ADDRESS
;21777  ;       ID[T7] HAS DIVISOR-LENGTH/2
;21778  ;       ID[T8] HAS DIVIDEND-ADDRESS
;21779  ;       ID[T9] HAS ORIGINAL QUOTIENT LENGTH
;21780  ;       R0,R1 ARE USED FOR SCRATCH DURING PROCESSING
;21781  ;       R2 HAS DIVIDEND LENGTH.OR.K[.1]
;21782  ;       R3 HAS DIVIDEND ADDRESS,LOW
;21783  ;       R4 HAS CURRENT QUOTIENT LENGTH ,INITIALLY LENGTH
;21784  ;       R5 HAS QUOTIENT ADDRESS, INITIALLY LOW ADDRESS
;21785  ;       R15 HAS LEADING DIGIT OF DIVIDEND
;21786  ;       RC4 HAS # OF RC-REGISTERS USED TO STORE DIVISOR
;21787  ;       RC5 HAS CURRENT DIGIT OF QUOTIENT
;21788
;21789  ;       STATE-REGISTER:
;21790          ;-------------------------------------------------------------
;21791          ;INTRPT ;OVFLOW ;END     ;1.PART ;DIV.   ;QUOT.  ;DIVR.  ;0-NIB  ;
;21792          ;       ;       ;OF      ;OPER.S ;INTRPT ;SIGN   ;SIGN   ;IN     ;
;21793          ;       ;       ;INSTRU  ;READ   ;       ;       ;       ;DIVISR ;
;21794          ;       ;       ;        ;       ;       ;       ;       ;       ;
;21795          ;-------------------------------------------------------------
;21796
;21797  ;       OPCODE IS "27"
;21798  ;       MNEMONIC IS "DIVP"
;21799  ;       INSTRUCTION DEPENDENT ALU FUNCTION IS "A-B-PSL.BORROW"
;21800  ;       INSTRUCTION DEPENDENT CC-CLOCKING IS: Z_Z,N_N,V_0,C_ALU CARRY[UDT]
```

```
                                    ;21801  3CD:
                                    ;21802          ;ENTER HERE FROM C-FORK WITH DIVISOR-LENGTH IN Q,
                                    ;21803          ;AND DIVISOR-ADDRESS IN D.
                                    ;21804          ;THIS ROUTINE EVALUATES SPECIFIERS AND INITIALIZES REGISTERS.
                                    ;21805          ;-------------------------------;
                                    ;21806  DIVP.INIT:
                                    ;21807          ID[T6]_D,                       ;  SAVE DIVISOR-ADDRESS IN T6
                                    ;21808          ALU_Q.OXT[WORD],                ;  ISOLATE LENGTH
                                    ;21809          D_ALU.RIGHT,                    ;  DIVIDE LENGTH BY 2
U 03CD, 0843,603C,D980,3C00,1400,6126 ;21810        STATE_FE                        ;  USE TO CLEAR STATE
                                    ;21811          ;-------------------------------;
                                    ;21812  =010**1*
                                    ;21813          STATE_STATE-FE,                 ;  CLEAR STATE-REGISTER
U 0126, 0000,003D,DD80,3C00,1400,A37E ;21814        ID[T7]_D,CALL,J/SPEC            ;  SAVE LENGTH/2 IN T7
                                    ;21815          ;-------------------------------;
                                    ;21816  =011**1*
                                    ;21817          ALU_Q.AND.K[.FFF0],             ;  MASK OUT LOW 4 BITS
                                    ;21818          N&Z_ALU.V&C_0,LONG,             ;  CLOCK Z-BIT
U 0136, 0019,2035,6D80,F9A8,0050,047E ;21819        RC[T5]_ALU,CALL,J/ASPC          ;  CLEAR RC5,GET DIVIDEND-ADDRESS
                                    ;21820          ;-------------------------------;
                                    ;21821  =111**1*
                                    ;21822          ID[T8]_D,                       ;  SAVE DIVIDEND-ADDRESS IN T8
                                    ;21823          ALU_Q.OXT[WORD],                ;  SAVE DIVIDEND LENGTH IN RC0
                                    ;21824          RC[T0]_ALU.RIGHT,               ;  SHIFT BACK LATER
U 0176, 0043,603C,E180,3D80,0000,01A2 ;21825        J/DIV.I1
                                    ;21826  =;END   ;-------------------------------;
                                    ;21827  =010**1*
                                    ;21828  DIV.I1: ALU_Q.AND.K[.FFE0],             ;  MASK OUT THE ILLEGAL BITS
                                    ;21829          N_ARX.Z_TST,LONG,               ;  "OR" RESULT INTO Z-BIT
U 01A2, 0019,2035,A180,F800,0030,037E ;21830        CALL,J/SPEC                     ;  EVALUATE QUOTIENT-LENGTH
                                    ;21831          ;-------------------------------;
                                    ;21832  =011**1*
                                    ;21833          ALU_D.AND.K[.FFE0],R[R15]_ALU,  ;  CLEAR R15
                                    ;21834          N_ARX.Z_TST,WORD,               ;  CLOCK QUOTIENT LENGTH
U 01B2, 0019,4035,A180,FAF8,0030,047E ;21835        CALL,J/ASPC                     ;  EVALUATE QUOTIENT-ADDRESS
                                    ;21836          ;-------------------------------;
                                    ;21837  =111**1*
                                    ;21838          ALU_Q.AND.K[.1F],Q_ALU,         ;  ISOLATE QUOTIENT-LENGTH
                                    ;21839          R[R4]_ALU,LONG,                 ;  SAVE IT IN R4
U 01F2, 0019,3A34,8DC0,FAA0,0000,039? ;21840        PSL.CC?,J/DIV.I2                ;  TEST FOR ILLEGAL LENGTHS
                                    ;21841  =;END   ;-------------------------------;
```

```
                                    ;21842          ;-------------------------------;
                                    ;21843  =10**   ;BRANCH ON PSL Z-BIT
                                    ;21844          ;10**---------------------------;
U 0399, 0000,003C,0180,F800,0000,0106 ;21845  DIV.I2: J/RSVOPR                       ;  ILLEGAL LENGTHS
                                    ;21846          ;11**---------------------------;
                                    ;21847          ALU_D,R[R5]_ALU,LONG,           ;  STORE QUOTIENT-ADDRESS
U 039D, 0C01,003C,E1F0,2EA8,0000,0C0A ;21848          D_Q,Q_ID[T8]                    ;  GET DIVIDEND-ADDRESS
                                    ;21849  =;END   ;-------------------------------;
                                    ;21850          ALU_Q,R[R3]_ALU,LONG,           ;  STORE DIVIDEND-ADDRESS IN R3
U 0C0A, 0F01,203C,E580,3E98,0000,0C0C ;21851          ID[T9]_D,D_0                    ;  SAVE LENGTH IN T9
                                    ;21852          ;-------------------------------;
U 0C0C, 001F,2000,0180,F900,0070,01A0 ;21853          LC_RC[T0],ALU_0-D,SET.CC(LONG)  ;  SET C-BIT
                                    ;21854          ;-------------------------------;
                                    ;21855  =00**0
                                    ;21856          ALU_LC,R[R2]_ALU.LEFT,          ;  INITIALIZE DIVIDEND-LENGTH
                                    ;21857          SI/MUL-,D_Q,
U 01A0, 0C30,0039,03F8,FA90,0000,09C0 ;21858          Q_0,CALL,J/DIVDR                ;  ROUTINE TO LOAD DIVISOR IN RC
                                    ;21859          ;-------------------------------;
                                    ;21860  =10**0
                                    ;21861  DIVC0:  STATE_STATE.ANDNOT.K[.C],       ;  CLEAR SIGN-BIT
                                    ;21862          VA_R[R3],                       ;  LOAD DIVIDEND-ADDRESS
U 01B0, 0000,003D,8580,FA18,1604,4C21 ;21863          CA[L,J/DIVND                    ;  READ DIVIDEND INTO ID AND STACK
                                    ;21864          ;-------------------------------;
```

```
                                ;21865          ;ROUTINE WHICH CONTROLS THE EXECUTION OF THE DIVIDE-INSTRUCTION
                                ;21866          ;CALLS THE NECESSARY SUBROUTINES, TO READ AND LOAD DATA,
                                ;21867          ;CALCULATE NEW DIGITS, AND UPDATE TEMPORARY STRINGS.
                                ;21868          ;R2 HAS DIVIDEND LENGTH.OR.K[.1]
                                ;21869          ;ID[T5] HAS DIVISOR-LENGTH
                                ;21870  =11**0
U 01B8, 0000,003C,3180,F800,1404,21B9   ;21871  DIVC01: STATE_STATE.OR.K[.40]          ;  SET OVERFLOW-BIT OF STATE
                                ;21872          ;-----------------------------;
                                ;21873  DIVC1:  ALU_R[R2],D_ALU,                ;  GET DIVIDEND LENGTH
                                ;21874          Q_ID[T5],                       ;  GET DIVISOR-LENGTH
U 01B9, 0800,003C,D5F0,2E10,0000,0C0D   ;21875          J/DIVC10
                                ;21876  =;END   ;-----------------------------;
                                ;21877  DIVC10: STATE_STATE.OR.K[.10],          ;  SET 1.PART FLAG
                                ;21878          INTRPT.STROBE,                  ;  STROBE FOR INTERRUPTS
                                ;21879          ALU_D-Q,D_ALU,                  ;  DIVIDEND.LENGTH-DIVISOR.LENGTH
U 0C0D, 081D,0000,6580,F8A0,5404,2C0E   ;21880          LA_RA[R4]                       ;  GET PRODUCT-LENGTH
                                ;21881          ;-----------------------------;
                                ;21882          ALU_LA-D,
                                ;21883          D_ALU,CLK.UBCC,BYTE,            ;  CLOCK THE DIFFERENCE
U 0C0E, 081C,AE00,0180,F800,0010,0D56   ;21884          BEN/INTERRUPT                   ;  TEST FOR INTERRUPT REQUESTS
                                ;21885          ;-----------------------------;
                                ;21886  =110    ;BRANCH ON INTERRUPT REQUEST
                                ;21887          ;110--------------------------;
                                ;21888          STATE_STATE.ANDNOT.K[.88],      ;  CLEAR INTERRUPT-BITS OF STATE
                                ;21889          SC_RC[T4],Q_RC[T4],             ;  GET RC-COUNT
U 0D56, 0010,1B38,C5C0,F920,1486,4423   ;21890          BEN/ALU,J/DIVC11                ;  TEST THE DIFFERENCE
                                ;21891          ;111--------------------------;
                                ;21892          STATE_STATE.OR.K[.88],          ;  SET INTERRUPT-BIT, AND DIV.INTR.
U 0D57, 0000,003C,3580,F800,1404,29C1   ;21893          J/DIV.MEMORY.FAULT              ;  JOIN MEMORY FAULT ROUTINE
                                ;21894  =;END   ;-----------------------------;
                                ;21895  =00011  ;BRANCH ON ALU Z AND N-BITS
                                ;21896          ;00011--------------------------;
U 0423, 0000,003C,01C0,FA20,0000,0437   ;21897  DIVC11: Q_R[R4],J/DIVC2                 ;  RC5 IS 0
                                ;21898          ;00111--------------------------;
                                ;21899          FE_SC,ALU_Q.OR.K[.30],          ;  GET ID-BUS POINTER
                                ;21900          CLR.UBCC,SC_ALU,                ;  STORE IT IN SC
U 0427, 0019,2031,7980,F800,0192,0D97   ;21901          CALL,J/DVSUB                    ;  CALCULATE AND WRITE QUOTIENT DATA
                                ;21902          ;01011--------------------------;
                                ;21903          FE_SC,ALU_Q.OR.K[.30],          ;  GET ID-BUS POINTER
                                ;21904          CLR.UBCC,SC_ALU,                ;  STORE IT IN SC
U 042B, 0019,2031,7980,F800,0192,0D97   ;21905          CALL,J/DVSUB                    ;  CALCULATE OVERFLOW DIGITS
                                ;21906  =10111  ;10111--------------------------;
                                ;21907  DIVC2:  R[R4]_LA-K[.1],CLK.UBCC,BYTE,   ;  UPDATE DST-LENGTH
U 0437, 0C18,8000,0580,FAA0,0010,0C42   ;21908          D_Q,J/DIVSAV                    ;  WRITE QUOTIENT-DIGIT
                                ;21909          ;11011--------------------------;
                                ;21910  =11011  ALU_RC[T5],                     ;  GET OVERFLOW-DIGIT
                                ;21911          CLK.UBCC,BYTE,                  ;  CLOCK DIGIT
U 043B, 0010,8038,0180,F928,0010,0C10   ;21912          J/DIVC4
                                ;21913  =;END   ;-----------------------------;
                                ;21914  DIVC4:  RC[T5]_0,Z?,                    ;  CLEAR IT,TEST IT
U 0C10, 0003,013C,0180,F9A8,0000,01B8   ;21915          J/DIVC01                        ;  FOR OVERFLOW
                                ;21916          ;-----------------------------;
```

```
                                ;21917          ;ROUTINE WHICH READS DIVISOR AND STORES IT IN RC
                                ;21918          ;LEFT ADJUSTED, SO THAT HIGH NIBBLE OF RC0 IS NON-ZERO.
                                ;21919          ;EXPECTS DIVISOR-LENGTH/2 IN T7, ADDRESS IN T6
                                ;21920          ;RC4 HAS # OF REGISTERS USED TO STORE THE DIVISOR (0-3)
                                ;21921          ;R1 IS USED TO STORE DIVISOR-ADDRESS,INITIALIZED TO LOW ADDRESS-1
                                ;21922          ;RC6 IS USED FOR NON-ZERO DIVISOR-LENGTH
                                ;21923          ;RETURNS # OF NON-ZERO DIGITS-1 IN T5
                                ;21924  =00
U 09C0, 0000,003D,D9F0,2C00,0000,0C12   ;21925  DIVDR:  Q_ID[T6],CALL,J/DIVFPD          ;  Q GETS DIVISOR-LENGTH
                                ;21926          ;01-------------------------------;
                                ;21927  DIV.MEMORY.FAULT:                           ;  ENTER HERE ON FAULTS AND INTERRUPTS
                                ;21928          ALU_R[R15].AND.K[.F],D_ALU,         ;  SAVE NIBBLE FROM R15
U 09C1, 0818,0034,6180,FA78,0000,0CBA   ;21929  J/DIV.SAVE
                                ;21930          ;10-------------------------------;
                                ;21931          ALU_R[R15].AND.K[.F],D_ALU,         ;  SAVE NIBBLE FROM R15
U 09C2, 0818,0034,6180,FA78,0000,0CBA   ;21932  J/DIV.SAVE
                                ;21933          ;11-------------------------------;
U 09C3, 0C00,003C,DDF0,2C00,0000,0C11   ;21934          D_Q,Q_ID[T7]                ;  GET DIVISOR-LENGTH
                                ;21935  =;END   ;-------------------------------;
                                ;21936          SC_Q,ID[FPDA]_D,                    ;  STORE RESTART ADDRESS
U 0C11, 0F01,203C,B580,3C00,0082,0A4E   ;21937  D_0,J/DIVD0
                                ;21938          ;-------------------------------;
                                ;21939  DIVFPD: ALU_Q,R[R1]_ALU,VAK/LOAD,           ;  LOAD DIVISOR-ADDRESS
U 0C12, 0001,203E,81F0,2E88,2600,0003   ;21940  SET.FPD,Q_ID[USTACK],RETURN3          ;  LOAD FAULT ADDRESS, SET 1. PART DONE
                                ;21941          ;-------------------------------;
                                ;21942  =011   ;BRANCH ON SC NE 0
                                ;21943          ;011------------------------------;
                                ;21944  DIVR:   ALU_D.AND.K[.F0],                   ;  STRIP OFF SIGN-NIBBLE
                                ;21945          D_ALU,CLK.UBCC,
U 0D63, 0819,0F34,CD80,F800,0010,09DA   ;21946  BCDSGN?,J/DIVD6                     ;  SIGN-BYTE, CHECK FOR 0 DIVISOR
                                ;21947          ;111------------------------------;
                                ;21948          ALU_0+LB+1,R[R1]_ALU,LONG,          ;  LOAD DIVISOR-ADDRESS
                                ;21949          VAK/LOAD,
                                ;21950          STATE_STATE.ANDNOT.K[.3],           ;  CLEAR 0-NIBBLE BIT
U 0D67, 000F,1810,0D80,FA88,1604,4A4E   ;21951  D.B0?,J/DIVD0                       ;  TEST D FOR 0
                                ;21952  =;END   ;-------------------------------;
```

```
                                ;21953  =1110   ;BRANCH ON LOW BYTE OF D NE. 0
                                ;21954          ;1110----------------------------;
                                ;21955  DIVD0:  D[BYTE]_CACHE,ALU_K[.1],        ;  READ NEXT BYTE FROM DIVISOR
                                ;21956          LAB_R1&RC[T4]_ALU.RIGHT2,       ;  GET ADDRESS, CLEAR RC4
                                ;21957          SC_SC-K[.1],                    ;  DECREMENT COUNT
U 0A4E, 0098,8C38,0580,4320,0084,AD63   ;21958          SC.NE.0?,J/DIVR                 ;  TEST LENGTH
                                ;21959          ;1111----------------------------;
                                ;21960          ALU_D.ANDNOT.K[.F],
U 0A4F, 0019,8024,6180,F800,0010,0C13   ;21961          CLK.UBCC,BYTE                   ;  CLOCK HIGH NIBBLE
                                ;21962  =;END   ;------------------------------;
                                ;21963          ALU_0+K[SC]+1,                  ;  ADJUST LENGTH
                                ;21964          Q_D,D_ALU.LEFT,                 ;  MAKE IT NIBBLE-COUNT AGAIN
U 0C13, 083B,0110,1DE0,F800,0000,0834   ;21965          Z?                              ;  TEST HIGH NIBBLE
                                ;21966          ;------------------------------;
                                ;21967  =0      ;BRANCH ON ALU Z-BIT
                                ;21968          ;0----------------------------;
                                ;21969  DIVD01: ID[T5]_D,                       ;  SAVE LENGTH IN ID[T5]
                                ;21970          ALU_0-D,Q_ALU.RIGHT,D_Q,        ;  KLUDGE TO INITIALIZE LENGTH
U 0834, 0C5F,2000,D4C0,3C00,0000,0C14   ;21971          SI/ASHR,J/DIVD02                ;  IN RC6
                                ;21972          ;1----------------------------;
                                ;21973          ALU_D-K[.1],D_ALU,              ;  ADJUST NIBBLE-COUNT FOR LEADING 0
                                ;21974          STATE_STATE.OR.K[.1],           ;  REMEMBER TO LEFT-ADJUST LATER
U 0835, 0819,0000,0580,F800,1404,2834   ;21975          J/DIVD01
                                ;21976  =;END   ;------------------------------;
                                ;21977  DIVD02: SC_K[.18],FE_K[.18],            ;  SET UP COUNTERS FOR LOOP
                                ;21978          ALU_0+Q+1,RC[T6]_ALU,           ;  NEGATIVE BYTE-COUNT
U 0C14, 001F,0010,7DF8,F9B0,0194,6C16   ;21979          Q_0,CLK.UBCC
                                ;21980          ;------------------------------;
                                ;21981  DIVD03: SC_FE,FE_SC,
                                ;21982          LC_RC[T6],D_DAL.SC,             ;  LOAD NEGATIVE COUNT IN LC
U 0C16, 0D00,003C,0180,F930,0181,0A5B   ;21983          J/DIVD1                         ;  LEFT-ADJUST THE BYTE WE ALREADY READ
                                ;21984          ;------------------------------;
```

```
                                    ;21985          ;RC[T6] HAS NEGATIVE COUNT
                                    ;21986          ;ENTER HERE AFTER FINDING NON-ZERO BYTE
                                    ;21987          ;-------------------------------;
                                    ;21988  =1011   ;BRANCH ON EALU Z-BIT
                                    ;21989          ;1011-----------------------------;
                                    ;21990  DIVD1:  D[BYTE]_CACHE,                  ;  READ BYTE FROM DIVISOR
                                    ;21991          FE_SC-K[.8],                    ;  UPDATE COUNT
                                    ;21992          ALU_Q+LC+1,                     ;  UPDATE DIVISOR-LENGTH
                                    ;21993          LAB_R1&RC[T6]_ALU,              ;  GET DIVISOR ADDRESS, STORE LENGTH
                                    ;21994          SC_FE,Q_D,
U 0A5B, 0011,A110,01E0,4330,0195,A838 ;21995        CLR.UBC[,Z?,J/DIVD2             ;  TEST DIVISOR-LENGTH
                                    ;21996          ;1111---------------------------;
                                    ;21997                                          ;  LONGWORD IS COMPLETE, STORE IT
                                    ;21998          SC_RC[T4],D_D.SWAP,             ;  GET RC-POINTER, PUT DATA IN
U 0A5F, 0B10,0038,0180,F920,0082,0C18 ;21999        J/DIVD3                         ;  ARITHMETIC ORDER
                                    ;22000  =;END    ;-------------------------------;
                                    ;22001  =0      ;BRANCH ON ALU Z-BIT
                                    ;22002          ;0------------------------------;
                                    ;22003  DIVD2:  D_DAL.SC,                       ;  SHIFT IN NEW BYTE
                                    ;22004          A[U_0+LB+1,                     ;  INCREMENT ADDRESS
                                    ;22005          LC_RC[T6]&R1_ALU,               ;  GET LENGTH, STORE ADDRESS
                                    ;22006          VAR/LOAD,Q_0,                   ;  LOAD DIVISOR-ADDRESS
                                    ;22007          SC_FE,FE_SC,
U 0838, 0D0F,1210,01F8,FBB0,0381,0A5B ;22008        EA[U?,J/DIVD1                   ;  TEST FOR COMPLETE LONGWORD
                                    ;22009          ;1--------------------------------;
                                    ;22010          D_D.ANDNOT.K[.F],               ;  SIGN-BYTE
U 0839, 0819,0F24,6180,F800,0000,09D2 ;22011        BCDSGN?,J/DIVD4                 ;  STRIP SIGN-NIBBLE, TEST SIGN
                                    ;22012  =;END    ;-------------------------------;
                                    ;22013  DIVD3:  ALU_D,RC(SC)_ALU,               ;  STORE LONGWORD IN RC-REGISTER
U 0C18, 0F01,003C,0180,F838,0081,0C19 ;22014        D_0,SC_FE
                                    ;22015          ;-------------------------------;
                                    ;22016          RC[T4]_0+LC+1,                  ;  INCREMENT RC-POINTER
                                    ;22017          FE_SC+K[.8],                    ;  UPDATE SHIFT-CONSTANT
U 0C19, 0F13,0010,01F8,F9A0,0104,8C16 ;22018        D_0,Q_0,J/DIVD03                ;  FOR LATER LEFT ADJUSTMENT
                                    ;22019          ;-------------------------------;
                                    ;22020  =10     ;BRANCH ON BCD-SIGN
                                    ;22021          ;10-----------------------------;
                                    ;22022  DIVD4:  STATE_STATE.ANDNOT.K[.2],       ;  CLEAR DIVISOR-SIGN
                                    ;22023          D_DAL.SC,                       ;  MERGE WITH OLD DATA
U 09D2, 0D00,003C,0980,F800,1404,4C1A ;22024        J7DIVD31
                                    ;22025          ;11-----------------------------;
                                    ;22026          STATE_STATE.OR.K[.2],           ;  SET DIVISOR-SIGN
U 09D3, 0D00,003C,0980,F800,1404,2C1A ;22027        D_DAL.SC                        ;  SHIFT DATA INTO PLACE
                                    ;22028  =;END    ;-------------------------------;
                                    ;22029  DIVD31: D_D.SWAP,                       ;  PUT LONGWORD IN ARITHMETIC ORDER
                                    ;22030          SC_FE,FE_SC,
U 0C1A, 0B00,003C,01F8,F800,0181,0C1C ;22031        Q_0                             ;  GET READY FOR SHIFT
                                    ;22032          ;-------------------------------;
                                    ;22033          D_DAL.SC,SC_RC[T4],             ;  LEFT ADJUST, GET RC-POINTER
U 0C1C, 0D10,1738,ED80,F920,0082,0A6E ;22034        K[.1B],STATE3-0?                ;  CHECK ODD NIBBLE BOUNDARY
                                    ;22035          ;-------------------------------;
```

```
                                ;22036  =1110   ;BRANCH ON 0-NIBBLE-BIT
                                ;22037          ;1110----------;
                                ;22038          ALU_D,RC(SC)_ALU,               ;  STORE LAST LONGWORD IN RC
U 0A6E, 0001,003E,0180,F838,0000,0010   ;22039  RETURN10
                                ;22040          ;1111----------;
                                ;22041          FE_SC,ALU_0-K[.18]-1,SC_ALU,    ;  GENERATE CONSTANT -28. FOR SHIFTING
U 0A6F, 0F1B,0008,EDE0,F800,0192,0C1D   ;22042  CLR.UBCC,Q_D,D_0
                                ;22043  =;END   ;----------;
                                ;22044  DIVD32: D_DAL.SC,                       ;  SHIFT ONE NIBBLE
U 0C1D, 0D00,123C,0180,F800,0181,0A7B   ;22045  SC_FE,FE_SC,EALU?                ;  FINISHED ?
                                ;22046          ;----------;
                                ;22047  =1011   ;BRANCH ON EALU Z-BIT
                                ;22048          ;1011----------;
                                ;22049          ALU_D,RC(SC)_ALU,               ;  STORE RESULT OF SHIFT IN RC
U 0A7B, 0001,003C,0580,F838,0094,AC1E   ;22050  SC_SC-K[.1],CLK.UBCC,J/DIVD33    ;  DECREMENT COUNT TO GET PREVIOUS REG
                                ;22051          ;1111----------;
U 0A7F, 0001,003E,0180,F838,0000,0010   ;22052  ALU_D,RC(SC)_ALU,RETURN10         ;  LOAD LAST REGISTER, RETURN
                                ;22053  =;END   ;----------;
                                ;22054  DIVD33: D_Q,ALU_RC(SC),Q_ALU,           ;  GET NEXT RC-REGISTER
U 0C1E, 0C10,0038,01C0,F830,0181,0C1D   ;22055  SC_FE,FE_SC,J/DIVD32             ;  KEEP LOOPING
                                ;22056          ;----------;
                                ;22057          ;  ENTER HERE IF SIGN-BYTE IS FIRST NON-ZERO BYTE
                                ;22058  =10     ;BRANCH ON BCD-SGN
                                ;22059          ;10----------;
                                ;22060  DIVD6:
                                ;22061          STATE_STATE.ANDNOT.K[.2],       ;  CLEAR DIVISOR-SIGN-BIT
                                ;22062          RC[T4]_0,D_D.SWAP,              ;  LEFT-ADJUST BY SWAPPING
U 09DA, 0B03,003C,0980,F9A0,1404,4C20   ;22063  J/DIVD61
                                ;22064          ;11----------;
                                ;22065          STATE_STATE.OR.K[.2],           ;  SET DIVISOR SIGN-BIT
                                ;22066          RC[T4]_0,D_D.SWAP,              ;  LEFT-ADJUST BY SWAPPING
U 09DB, 0B03,003C,0980,F9A0,1404,2C20   ;22067  J/DIVD61
                                ;22068  =;END   ;----------;
                                ;22069  DIVD61: RC[T0]_D,D_0,Q_ID[CES],         ;  STORE SINGLE NIBBLE, GET CES-REGISTER
U 0C20, 0F01,013C,31F0,2D80,0000,083C   ;22070  Z?                              ;  TEST SINGLE NIBBLE
                                ;22071          ;----------;
                                ;22072  =0      ;BRANCH ON ALU Z-BIT
                                ;22073          ;0----------;
U 083C, 0000,003E,D580,3C00,0000,0010   ;22074  ID[T5]_D,RETURN10                ;  OK-NON-ZERO DIGIT
                                ;22075          ;1----------;
                                ;22076  DIVERR: ALU_Q.OR.K[.40],D_ALU,          ;  DECIMAL DIVIDE BY 0
U 083D, 0819,2030,3180,F800,0000,0828   ;22077  J/FINI6                         ;  TRAP CODE IS 4
                                ;22078  =;END   ;----------;
```

```
                                    ;22079  DIVND:  ;ROUTINE WHICH READS DIVIDEND AND LOADS IT INTO
                                    ;22080          ;ID-REGISTERS AND STACK.
                                    ;22081          ;R1 IS USED TO STORE DIVIDEND-ADDRESS ,INITIALLY LOW ADDRESS
                                    ;22082          ;RC6 IS USED FOR DIVIDEND-LENGTH DURING INSTRUCTION
                                    ;22083          ;RC7 IS USED FOR # OF LONGWORDS ON STACK (30-33), INITIALLY 30
                                    ;22084          ;R2 HAS DIVIDEND-LENGTH (# OF NIBBLES-1)
                                    ;22085          ;R[SP] HAS ORIGINAL STACK-POINTER-16. (R14)
                                    ;22086          ;DATA IS WRITTEN ON STACK IN ARITHMETIC ORDER.
                                    ;22087          ;-------------------------------;
                                    ;22088          SC_K[.18],FE_K[.18],
U 0C21, 0840,003C,7D80,F890,0184,6C22 ;22089        LA_RA[R2],ALU_LA,D_ALU.RIGHT    ;  DIVIDE LENGTH BY 2
                                    ;22090          ;-------------------------------;
                                    ;22091          RC[6]_0-D,SC_SC+K[.8],          ;  INITIALIZE COUNTER
U 0C22, 001F,2000,0180,F9B0,0094,8C23 ;22092        CLK.UBCC                        ;  WITH DIVND.-LENGTH
                                    ;22093          ;-------------------------------;
                                    ;22094          ALU_LB,VAK/LOAD,Q_0,
U 0C23, 000C,0038,01F8,FBB0,0200,0A8B ;22095        LC_RC[T6]&R1_ALU                ;  LOAD DIV. ADDRESS
                                    ;22096          ;-------------------------------;
                                    ;22097  =1011   ;BRANCH ON EALU Z-BIT
                                    ;22098          ;1011-----------------------------;
                                    ;22099  DIVN3:  D[BYTE]_CACHE,Q_D,              ;  READ BYTE OF DIVIDEND
                                    ;22100          FE_SC-K[.8],SC_FE,              ;  UPDATE COUNTER
                                    ;22101          ALU_Q+LC+1,LAB_R1&RC[T6]_ALU,   ;  INCREMENT COUNTER (Q=0)
                                    ;22102          CLK.UBCC,
U 0A8B, 0011,A110,01E0,4330,0195,A844 ;22103        Z?,J/DIVN4                      ;  TEST FOR END OF STRING
                                    ;22104          ;1111-----------------------------;
                                    ;22105          ALU_R[R15].OR.K[.30],           ;  LONGWORD COMPLETE
                                    ;22106          SC_ALU,Q_ALU,D_D.SWAP,          ;  GET DATA IN ARITHMETIC ORDER
U 0A8F, 0B18,0030,79C0,FA78,0082,0C24 ;22107        J/DIVN5                         ;  STORE DATA IN ID-REGISTER
                                    ;22108  =:END   ;-------------------------------;
                                    ;22109  =0      ;BRANCH ON ALU Z-BIT
                                    ;22110          ;0-----------------------------;
                                    ;22111  DIVN4:  D_DAL.SC,Q_0,                   ;  SHIFT NEW BYTE INTO LONGWORD
                                    ;22112          ALU_0+LB+1,                     ;  INCREMENT ADDRESS
                                    ;22113          LC_RC[T6]&R1_ALU,               ;  GET LENGTH, STORE ADDRESS
                                    ;22114          SC_FE,FE_SC,
                                    ;22115          VAR/LOAD,                       ;  LOAD DIVIDEND-ADDRESS
U 0844, 0D0F,1210,01F8,FBB0,0381,0A8B ;22116        BEN/EALU,J/DIVN3                ;  TEST FOR A COMPLETE LONGWORD
                                    ;22117          ;1-----------------------------;
                                    ;22118          D_D.ANDNOT.K[.F],               ;  THIS IS SIGN-BYTE
U 0845, 0819,0F24,6180,F800,0000,09E2 ;22119        BCDSGN?,J/DIVN8                 ;  TEST SIGN-NIBBLE
                                    ;22120  =:END   ;-------------------------------;
```

```
                                ;22121          ;ENTER HERE TO STORE LONGWORD ON STACK AND IN ID-REGISTER
                                ;22122  DIVN5:  LAB_R[SP],
                                ;22123          ALU_Q.ORNOT.K[.3],              ;  GET NEGATIVE COUNT
                                ;22124          SHF7ALU.DT,LONG,                ;  GENERATE STACK ADDRESS
                                ;22125          QK/SHF,
U 0C24, 0079,201C,0DC0,3670,0000,0C25   ;22126          ID(SC)_D                        ;  STORE IN ID-REGISTER
                                ;22127          ;-------------------------------;
                                ;22128          VA_Q+LB,                        ;  LOAD STACK-ADDRESS
U 0C25, 000D,2014,0180,F888,0200,0C28   ;22129          LA_RA[R1]
                                ;22130          ;-------------------------------;
                                ;22131          CACHE_D[LONG],                  ;  WRITE LONGWORD ON STACK
                                ;22132          ALU_Q+K[SC]+1,                  ;  INCREMENT ID-POINTER
                                ;22133          R[RT5]_ALU,                     ;  STORE IT IN R15
U 0C28, 001B,0010,1DF8,32F8,0081,0C29   ;22134          SC_FE,Q_0
                                ;22135          ;-------------------------------;
                                ;22136          ALU_LA,LC_RC[T6],               ;  GET DIVIDEND-LENGTH
                                ;22137          VAK7LOAD,SC_SC+K[.8],           ;  RE-INITIALIZE COUNTER
U 0C29, 0000,003C,0180,F930,0284,8A8B   ;22138          J/DIVN3                         ;  REENTER LOOP
                                ;22139          ;-------------------------------;
                                ;22140  =10     ;BRANCH ON BCD-SIGN
                                ;22141          ;10-------------------------------;
                                ;22142  DIVN8:  STATE_STATE+K[.2],              ;  DUPLICATE DIVISOR-SIGN
U 09E2, 0D00,003C,0980,F800,1404,8C2A   ;22143          D_DAL.SC,J/DIVN9                ;  SHIFT DATA IN
                                ;22144          ;T1-------------------------------;
                                ;22145          STATE_STATE+K[.6],              ;  COMPLEMENT DIVISOR SIGN-BIT
U 09E3, 0D00,003 ,D580,F800,1404,8C2A   ;22146          D_DAL.SC
                                ;22147  =:END   ;-------------------------------;
                                ;22148  DIVN9:  D_D.SWAP,                       ;  GET DATA IN ARITHMETIC ORDER
U 0C2A, 0800,003C,01F8,F800,0181,0C2C   ;22149          SC_FE,FE_SC,Q_0
                                ;22150          ;-------------------------------;
                                ;22151          D_DAL.SC,ALU_R[R15].OR.K[.30],  ;  LEFT ADJUST, GET ID-POINTER
U 0C2C, 0D18,0030,79C0,FA78,0082,0C2D   ;22152          SC_ALU,Q_ALU
                                ;22153          ;-------------------------------;
                                ;22154          LAB_R[SP],                      ;  GET STACK-POINTER
                                ;22155          ALU_Q.ORNOT.K[.3],
                                ;22156          SHF7ALU.DT,LONG,                ;  GENERATE STACK-ADDRESS
                                ;22157          QK/SHF,
U 0C2D, 0079,201C,0DC0,3670,0081,0C2E   ;22158          ID(SC)_D,SC_FE
                                ;22159          ;-------------------------------;
U 0C2E, 001C,0014,0180,F800,0200,0C30   ;22160          VA_LA+Q                         ;  LOAD STACK-ADDRESS
                                ;22161          ;-------------------------------;
                                ;22162          ALU_Q(A),R[R15]_ALU,            ;  INITIALIZE R15
                                ;22163          CACHE_D[LONG],                  ;  WRITE LAST LONGWORD ON STACK
U 0C30, 0003,003C,6580,32F8,0084,6C31   ;22164          SC_K[.10]                       ;  CONSTANT TO UPDATE STACK POINTER
                                ;22165          ;-------------------------------;
                                ;22166          ALU_LA-K[SC],R[SP]_ALU,         ;  RESERVE 16 BYTES ON STACK
U 0C31, 0018,0002,1D80,FAF0,0000,0009   ;22167          RETURN9
                                ;22168          ;-------------------------------;
```

```
                                ;22169          ;ROUTINE WHICH GENERATES A DIGIT OF QUOTIENT DEPENDING ON
                                ;22170          ;RC AND ID AND R15 (HIGH NIBBLE OF DIVIDEND).
                                ;22171          ;SAVES DIGIT IN RC5
                                ;22172          ;RC[T4] HAS NUMBER OF LONGWORDS USED IN RC (0-3),
                                ;22173          ;FE STORES THIS NUMBER DURING THE ROUTINE.
                                ;22174
                                ;22175          ;-------------------------------;
                                ;22176  DVSUB0: ALU_Q[INST.DEP]D,D_ALU,          ;  SUBTRACT WITH BORROW
                                ;22177          Q_DEC.CON,                      ;  STORE DECIMAL CONSTANT
                                ;22178          SC_SC-K[.2],                    ;  ADJUST RC ADDRESS POINTER
                                ;22179          LC_RC[T5],                      ;  GET CURRENT DIGIT
U 0C32, 081D,320C,09D0,F928,00F4,AA9B   ;22180          SET.CC(LONG),BEN/EALU           ;  TEST POINTER
                                ;22181          ;-------------------------------;
                                ;22182  =1011   ;BRANCH ON EALU Z-BIT
                                ;22183          ;1011-----------------------------;
                                ;22184  DVSUB01:
                                ;22185          D_D-Q,                          ;  DECIMAL ADJUST FOR SUBTRACTION
                                ;22186          LC_RC(SC),                      ;  GET LONGWORD FROM DIVISOR
                                ;22187          Q_ID(SC),                       ;  AND LONGWORD FROM DIVIDEND
U 0A9B, 081D,0000,01F0,2430,0080,CC33   ;22188          SC_SC+1,J/DVSUB1                ;  READJUST ADDRESS
                                ;22189          ;1111-----------------------------;
                                ;22190          SC_SC+1,                        ;  ADJUST ADDRESS
U 0A9F, 081D,0200,0180,FA78,0080,CD75   ;22191          D_D-Q,LAB_R[R15],ROR?           ;  DECIMAL ADJUST, TEST FOR BORROW
                                ;22192  =;END
                                ;22193
                                ;22194  ; ************************************************
                                ;22195  ; * Patch no. 072, PCS 0A9F trapped to WCS 1191 *
                                ;22196  ; ************************************************
                                ;22197
                                ;22198          ;-------------------------------;
                                ;22199  =101    ;BRANCH ON PSL C-BIT
                                ;22200          ;101------------------------------;
                                ;22201          R[R15]_LA-K[.1],                ;  BORROW-SO TRY LEFT-OVER DIGIT
U 0D75, 0018,8000,0580,FAF8,0070,0D77   ;22202          SET.CC(BYTE)                    ;  CLOCK PSL C-BIT
                                ;22203          ;111------------------------------;
                                ;22204          ID(SC)_D,RC[T5]_0+LC+1,         ;  NO BORROW-INCREMENT QUOTIENT
U 0D77, 0013,0210,0180,35A8,0080,8D95   ;22205          SC_SC+FE,ROR?,J7DVSUB2          ;  TRY AGAIN
                                ;22206  =;END   ;-------------------------------;
                                ;22207  DVSUB1: ID(SC)_D,EALU_SC+K[.DFCF],      ;  KEEP LOOPING
U 0C33, 0810,0038,AD80,3400,0014,8C32   ;22208          D_LC,CLK.UBCC,J/DVSUB0
                                ;22209          ;-------------------------------;
```

```
                                ;22210  =101     ;BRANCH ON  PSL C-BIT
                                ;22211          ;101-------------------------------;
                                ;22212  DVSUB2: ALU_0+LC,RC[T5]_ALU,            ;  STORE NEW DIGIT
U 0D95, 0013,0014,01D0,F9A8,0000,0C36   ;22213          Q_DEC.CON,J/DVAD        ;  GENERATE ALL 6'S
                                ;22214          ;111-------------------------------;
                                ;22215  DVSUB:  LC_RC(SC),ALU_LC,               ;  GET DIVISOR LONGWORD
                                ;22216          D_ALU,Q_ID(SC),                 ;  GET DIVIDEND LONGWORD
U 0D97, 0810,0038,01F0,2430,0010,6C34   ;22217          EALU_FE,CLK.UBCC
                                ;22218  =;END   ;-------------------------------;
                                ;22219          ALU_Q[INST.DEP]D,D_ALU,         ;  SUBTRACT WITH BORROW
                                ;22220          Q_DEC.CON,SC_SC-K[.1],          ;  STORE DECIMAL CONSTANT
                                ;22221          LC_RC[T5],                      ;  GET DIGITS
                                ;22222          SET.CC(LONG),BEN/EALU,          ;  TEST ADDRESS POINTER
U 0C34, 081D,320C,05D0,F928,00F4,AA9B   ;22223          J/DVSUB01
                                ;22224          ;-------------------------------;
                                ;22225  DVAD:   LC_RC(SC),Q_ID(SC),             ;  GET OPERANDS FOR RESTORING PORTION
U 0C36, 0C00,003C,11F0,2430,01C4,6C38   ;22226          D_0,FE_K[.4]            ;  D GETS 6'S
                                ;22227          ;-------------------------------;
                                ;22228  DVAD0:  D_D+Q,                          ;  ADD 6'S TO DIVIDEND LONGWORD
                                ;22229          EALU_SC-K[.30],                 ;  COMPARE SC WITH LOW LIMIT
U 0C38, 081D,0014,7980,FA70,0014,AC39   ;22230          CLK.UBCC,LAB_R[SP]      ;  GET STACK POINTER READY FOR STORING
                                ;22231          ;-------------------------------;
                                ;22232          D_D+LC+PSL.C,Q_DEC.CON,         ;  ADD WITH CARRY
U 0C39, 0811,002C,05D0,F800,00F4,AC3A   ;22233          SC_SC-K[.1],SET.CC(LONG)        ;  CLOCK CARRY
                                ;22234          ;-------------------------------;
                                ;22235          D_D-Q,                          ;  DECIMAL ADJUST
                                ;22236          LC_RC(SC),Q_ID(SC),             ;  GET NEXT OPERANDS
U 0C3A, 081D,1200,01F0,2430,0080,CAAB   ;22237          SC_SC+1,BEN/EALU        ;  TEST POINTER
                                ;22238          ;-------------------------------;
                                ;22239  =1011   ;BRANCH ON EALU Z-BIT
                                ;22240          ;-------------------------------;
                                ;22241          ALU_0+Q,D_ALU,Q_DEC.CON,
U 0AAB, 081F,0014,05D0,3400,0084,AC38   ;22242          ID(SC)_D,SC_SC-K[.1],J/DVAD0    ;  KEEP ADDING
                                ;22243          ;-------------------------------;
                                ;22244          ALU_LB,VA_ALU,LA_RA[R2],
                                ;22245          FE_SC+1,SC_FE,Q_D,
U 0AAF, 000C,0038,01E0,F890,0381,CC3B   ;22246          J/DIVST
                                ;22247          ;-------------------------------;
```

```
                                ;22248  DIVST:  ;ROUTINE WHICH READS DIVIDEND -STRING OFF
                                ;22249          ;ID-BUS, SHIFTS IT LEFT, AND WRITES THE RESULT ON THE
                                ;22250          ;STACK AS WELL AS BACK ON ID-BUS.
                                ;22251          ;DECREMENTS DIVIDEND-LENGTH BY 1.
                                ;22252          ;R14 (STACK-POINTER) POINTS TO LOW ADDRESS
                                ;22253          ;RA2 HAS DIVIDEND LENGTH
                                ;22254          ;RA1 IS USED FOR SCRATCH TO KEEP DIV.LENGTH WHILE USED FOR COUNTER
                                ;22255          ;RA15 GETS LEFT-OVER DIGIT
                                ;22256
                                ;22257
                                ;22258          ;-- ------------------------------;
                                ;22259          ALU_LA-K[.8],R[R1]_ALU,         ;  UPDATE DIVIDEND LENGTH
                                ;22260          CLK.UBCC,
                                ;22261          D_DAL.SC,                       ;  SHIFT IN NIBBLE
U 0C3B, 0D18,0000,0180,FA88,0191,0C3C   ;22262          SC_FE,FE_SC
                                ;22263          ;------------------------------;
                                ;22264          ALU_D.AND.K[.F],R[R15]_ALU,     ;  STORE HIGH NIBBLE IN R15
U 0C3C, 0C19,0034,61F0,26F8,0000,0C3E   ;22265          D_Q,Q_ID(SC)                    ;  GET NEXT LONGWORD
                                ;22266          ;------------------------------;
                                ;22267          ALU_LA-K[.1],                   ;  UPDATE REAL DIVIDEND LENGTH BY 1
U 0C3E, 0018,0000,0580,FA90,0185,AC40   ;22268          R[R2]_ALU,FE_SC-K[.1],SC_FE     ;  FE GETS .30, SC GETS 4
                                ;22269          ;------------------------------;
                                ;22270  DIVST2: D_DAL.SC,                       ;  SHIFT DATA INTO D
                                ;22271          SC_FE,FE_SC,
U 0C40, 0D00,1B3C,0180,F800,0181,0AB7   ;22272          ALU?                            ;  TEST LENGTH
                                ;22273          ;------------------------------;
                                ;22274  =0111   ;BRANCH ON ALU-N-BIT
                                ;22275          ;0111----------------------------;
                                ;22276  DIVST3: ID(SC)_D,LA_RA[R1],             ;  STORE SHIFTED RESULT
                                ;22277          SC_SC+R[.2],                    ;  SC NOW POINTS TO
U 0AB7, 0000,1B3C,0980,3488,0084,8AC7   ;22278          ALU?,J/DIVST4                   ;  ID-REGISTER WITH NEXT HIGH NIBBLE
                                ;22279          ;1111----------------------------;
U 0ABF, 0819,0024,6180,F800,0000,0AB7   ;22280          D_D.ANDNOT.K[.F],J/DIVST3       ;  LOW NIBBLE WAS NO GOOD
                                ;22281  =;END   ;------------------------------;
                                ;22282  =0111   ;BRANCH ON ALU N-BIT
                                ;22283          ;0111----------------------------;
                                ;22284  DIVST4: CACHE_D[LONG],                  ;  STORE IT ON THE STACK
                                ;22285          R[R1]_LA-K[.8],CLK.UBCC,        ;  UPDATE LENGTH
U 0AC7, 0018,0000,0180,3288,0010,0C41   ;22286          J/DIVST5
                                ;22287          ;1111----------------------------;
                                ;22288          CACHE_D[LONG],                  ;  WRITE LAST LONGWORD
U 0ACF, 0000,003E,01C0,3220,0000,0010   ;22289          Q_R[R4],RETURN10                ;  GET DST-LENGTH
                                ;22290  =;END   ;------------------------------;
                                ;22291  DIVST5: Q_ID(SC),D_Q,FE_SC-K[.1],       ;  GET NEXT REGISTER
                                ;22292          LA_RA[R1],
                                ;22293          VA_VA+4,                        ;  UPDATE STACK ADDRESS
U 0C41, 0C00,003C,05F0,248B,0185,AC40   ;22294          SC_FE,J/DIVST2
                                ;22295          ;------------------------------;
                                ;22296
```

```
                                    ;22297  DIVSAV: ;THIS ROUTINE TAKES THE DIGIT JUST GENERATED IN RC5
                                    ;22298          ;AND EITHER SHIFTS IT INTO THE HIGH NIBBLE,
                                    ;22299          ;OR WRITES THE BYTE CONTAINING IT IN THE QUOTIENT-BYTE
                                    ;22300          ;POINTED TO BY R5.
                                    ;22301          ;THIS CHOICE DEPENDS ON WHETHER R4 (THE QUOTIENT-LENGTH)
                                    ;22302          ;IS ODD OR EVEN.
                                    ;22303          ;IF R4=0, THE SIGN-NIBBLE IS ADDED TO RC5,
                                    ;22304          ;THE SIGN-BYTE IS WRITTEN, AND THE ROUTINE EXITS VIA
                                    ;22305          ;DIVFIN, TO SET PSL CONDITION CODES.
                                    ;22306          ;D AND Q ARE CLOBBERED, BUT SC AND FE ARE NOT USED.
                                    ;22307
                                    ;22308          ;-------------------------------;
                                    ;22309          STATE_STATE.ANDNOT.K[.9],       ;  CLEAR BITS JUST IN CASE
                                    ;22310          ALU_R[T5],Q_ALU.LEFT,           ;  ENTER HERE AFTER A FAULT
U 0C42, 0030,9B38,D9C0,F928,1434,4AD3 ;22311        N_AMX.Z_TST,BYTE,ALU?
                                    ;22312          ;-------------------------------;
                                    ;22313  =0011   ;BRANCH ON ALU Z AND N-BITS
                                    ;22314          ;0011---------------------------;
                                    ;22315          VA_R[R5],D_Q,MUL?,              ;  LOAD QUOTIENT-ADDRESS
U 0AD3, 0C00,0C3C,0180,FA28,0200,0DD6 ;22316        J/DIVS0                         ;  TEST ODD/EVEN LENGTH
                                    ;22317          ;0111---------------------------;
                                    ;22318          STATE_STATE.OR.K[.20],          ;  SET END-OF-INSTRUCTION BIT IN STATE
                                    ;22319          ALU_0+Q,SHF/LEFT,
                                    ;22320          SET_CC(LONG),                   ;  CLEAR C-BIT
U 0AD7, 083F,0014,7580,F800,1474,2C46 ;22321        DK/SHF,J/DIVS2
                                    ;22322          ;1011---------------------------;
                                    ;22323          STATE_STATE.OR.K[.20],          ;  SET END-OF-INSTRUCTION BIT IN STATE
                                    ;22324          N&Z_ALU.V&C_0,                  ;  SET Z-BIT
                                    ;22325          ALU_0(A),RC[T5]_ALU,
U 0ADB, 0F03,003C,7580,F9A8,1454,2C46 ;22326        D_0,J/DIVS2                     ;  NULL-STRING
                                    ;22327  =;END   ;-------------------------------;
                                    ;22328  =110    ;BRANCH ON LOW BIT OF D
                                    ;22329          ;110----------------------------;
                                    ;22330  DIVS0:  ALU_LA+K[.1],R[R5]_ALU,         ;  INCREMENT QUOTIENT ADDRESS
U 0DD6, 0618,0014,0580,FAA8,0000,0C43 ;22331        DK/RIGHT,J/DIVS01
                                    ;22332          ;111----------------------------;
                                    ;22333          ALU_Q,                          ;  NOT A WHOLE BYTE YET
U 0DD7, 00A1,203C,0180,F9A8,0000,01B9 ;22334        RC[T5]_ALU.LEFT3,J/DIVC1        ;  STORE DIGIT, SHIFTED LEFT
                                    ;22335  =;END   ;-------------------------------;
                                    ;22336  DIVS01: CACHE_D[BYTE],                  ;  STORE BYTE IN QUOTIENT STRING
U 0C43, 0000,803C,01C0,3210,0000,0C44 ;22337        ALU_R[R2],Q_ALU                 ;  GET DIVIDEND-LENGTH
                                    ;22338          ;-------------------------------;
                                    ;22339          D_Q,Q_ID[T5],                   ;  GET DIVISOR-LENGTH
                                    ;22340          A[U_0(A),RC[T5]_ALU,            ;  CLEAR QUOTIENT-BYTE
U 0C44, 0C03,003C,D5F0,2DA8,0000,0C0D ;22341        J/DIVC10
```

```
                                ;22342
                                ;22343  DIVS2:  VA_R[R5],Q_ID[T9],                ;  GET QUOTIENT ADDRESS
U 0C46, 0100,173C,E5F0,2E28,0200,03AA ;22344          DK/LEFT2,STATE3-0?          ;  TEST SIGN-BIT
                                ;22345
                                ;22346  =01*    ;BRANCH ON SIGN-BIT OF STATE-REGISTER
                                ;22347          ;01*-------------------------------;
U 03AA, 0819,0030,8580,F800,0000,0DE7 ;22348          D_D.OR.K[.C],J/DIVS4        ;  POSITIVE
                                ;22349          ;T1*-------------------------------;
                                ;22350          D_D.OR.K[.D],
U 03AE, 0819,1A30,8980,F800,0000,0AEB ;22351          PSL.CC?,J/DIVS3             ;  NEGATIVE
                                ;22352  =;END   ;-------------------------------;
                                ;22353  =1011   ;BRANCH ON PSL Z-BIT
                                ;22354          ;1011-----------------------------;
                                ;22355  DIVS3:  CACHE_D[BYTE],QK/RIGHT,           ;  WRITE NEGATIVE SIGN-BYTE
                                ;22356          ALU_K[.80],N&Z_ALU.V&C_0,       ;  SET PSL N-BIT
U 0AEB, 0018,8038,41B0,3000,0050,0C48 ;22357          J/DIVFIN
                                ;22358          ;1111-----------------------------;
U 0AEF, 0000,163C,0180,F800,0000,0DE3 ;22359          STATE7-4?                   ;  TEST FOR OVERFLOW
                                ;22360  =;END   ;-------------------------------;
                                ;22361  =011    ;BRANCH ON OVERFLOW-BIT OF STATE REGISTER
                                ;22362          ;011------------------------------;
U 0DE3, 0818,0038,8580,F800,0000,0DE7 ;22363  DIVS30: D_K[.C]                      ;  WRITE PLUS ZERO
                                ;22364          ;T11------------------------------;
                                ;22365  DIVS4:  CACHE_D[BYTE],                   ;  WRITE LAST BYTE
U 0DE7, 0000,803C,01B0,3000,0000,0C48 ;22366          QK/RIGHT,J/DIVFIN           ;  DIVIDE LENGTH BY 2
                                ;22367  =;END   ;-------------------------------;
                                ;22368  DIVFIN: ;ROUTINE WHICH FINISHES UP THE DIVIDE-INSTRUCTION
                                ;22369          ;RESETS THE STACK-POINTER, SETS THE CONDITION-CODES
                                ;22370          ;-------------------------------;
                                ;22371          ALU_LA-Q,R[R5]_ALU,LONG,        ;  RESET R5 WITH QUOTIENT ADDRESS
U 0C48, 001C,0000,D9F0,2EA8,0000,0C49 ;22372          Q_ID[T6]                    ;  GET DIVISOR ADDRESS
                                ;22373          ;-------------------------------;
U 0C49, 0003,003C,0180,FA30,0000,0C4A ;22374          ALU_0(A),R[R0]_ALU          ;  CLEAR R0
                                ;22375          ;-------------------------------;
U 0C4A, 0001,203C,0180,FA88,0000,0C4C ;22376          R[R1]_Q                     ;  R1 GETS DIVISOR ADDRESS
                                ;22377
                                ;22378  DIVP.JUNK.EXIT:
                                ;22379          ;-------------------------------;
U 0C4C, 0000,003C,6580,FA70,0000,0C4D ;22380          LAB_R[SP],K[.10]            ;  GET READY TO POP STACK
                                ;22381          ;-------------------------------;
U 0C4D, 0018,1614,6580,FAF0,0000,0DF3 ;22382          R[SP]_LA+K[.10],STATE7-4?   ;  RESTORE STACK POINTER
                                ;22383          ;-------------------------------;
                                ;22384  =011    ;BRANCH ON OVERFLOW-BIT OF STATE
                                ;22385          ;011------------------------------;
U 0DF3, 0003,003C,0180,FAA0,0000,0B29 ;22386          ALU_0(A),R[R4]_ALU,J/FINI8  ;  CLEAR R4
                                ;22387          ;111------------------------------;
                                ;22388          ALU_0(A),R[R4]_ALU,             ;  CLEAR R4
U 0DF7, 0003,003C,31F0,2EA0,0000,0B26 ;22389          Q_ID[CES],J/FINI5           ;  LOAD TRAP-VALUE
                                ;22390  =;END   ;-------------------------------;
```

```
;22391  .TOC    ""      Decimal string        : ASHP''
;22392
;22393  ;ARITMETIC SHIFT OF PACKED BCD
;22394  ;ALGORITHM:
;22395  ;       1. THE MICRO-CODE STARTS BY EVALUATING THE SPECIFIERS
;22396  ;       AND INITIALIZING REGISTERS (''ASHP.INIT'').
;22397  ;       FIRST PART DONE FLAG IS SET (''ASH.I3'').
;22398
;22399  ;       2. THE MAIN LOOP STARTS AT ''ASHP.E'', AND BEGINS BY READING
;22400  ;       A LONGWORD FROM THE SOURCE-STRING (''ASHP.E1'').
;22401
;22402  ;       3. THE PROGRAM USES TWO DIFFERENT PATHS THROUGH THE LOOP,
;22403  ;       DEPENDING ON WHETHER IT IS DOING A RIGHT SHIFT (''NEG.CNT''),
;22404  ;       OR A LEFT SHIFT (''POS.CNT'').
;22405
;22406  ;       4. ON RIGHT SHIFTS, THE ROUNDING OPERAND IS ADDED TO THE MOST
;22407  ;       SIGNIFICANT DISCARDED DIGIT (''NEG.4''), AND RESULTING CARRIES
;22408  ;       MAY PROPAGATE THROUGH THE STRING (''NEG.3'').
;22409
;22410  ;       5. ON LEFT SHIFTS, 0'S ARE SHIFTED INTO THE LEAST SIGNIFICANT
;22411  ;       END OF THE STRING (''POS.2'').
;22412
;22413  ;       6. IN ANY CASE, THE NEWLY READ DATA IS SHIFTED TOGETHER WITH
;22414  ;       PREVIOUS DATA (STORED IN RC2), AND THE RESULT IS
;22415  ;       WRITTEN INTO THE DEST-STRING (''ASHP.WRITE'').
;22416
;22417  ;       7. STEPS 2,3,4,5,6 ARE REPEATED UNTIL WE REACH THE
;22418  ;       END OF BOTH STRINGS, AT WHICH TIME WE LOAD THE GENERAL REGISTERS
;22419  ;       AND SET THE CONDITION-CODES (''ASHP.FIN'').
;22420
;22421  ;       8. IN CASE OF INTERRUPTS OR MEMORY-FAULTS, THE INITIAL STATE
;22422  ;       OF THE INSTRUCTION IS SAVED (''BCD.SAVE''), AND THE INSTRUCTION
;22423  ;       IS RESTARTED (''RESTART.ASHP'').
;22424
;22425
;22426  ;       OP-CODE IS ''F8''
;22427  ;       CCK/INST.DEP IS DEFINED AS: Z_Z,N_N,V_0,C_ALU CARRY[UDT]
;22428  ;       INST.DEP ALU-FUNCTION IS A-B-PSL.BORROW
;22429
;22430  ;       THIS ROUTINE CALLS THE SUBROUTINES:
;22431  ;       READ-BCD,WRITE-BCD,SPEC,ASPC
```

```
;22432  ;STORAGE-ALLOCATION:
;22433  ;       R0 HAS SRC-LENGTH ,INITIALLY -LENGTH-1
;22434  ;       R1 HAS SRC-ADDRESS, INITIALLY HIGH ADDRESS+1
;22435  ;       R2 HAS DST-LENGTH, INITIALLY -LENGTH-1
;22436  ;       R3 HAS DST-ADDRESS, INITIALLY HIGH ADDRESS+1
;22437  ;       RC0 HAS SRC-LENGTH
;22438  ;       RC1 HAS DST-LENGTH
;22439  ;       RC2 HAS LEFT-OVER DATA DURING RIGHT SHIFTS
;22440  ;       RC4 HAS ORIGINAL SHIFT-COUNT*4
;22441  ;       RC5 HAS SHIFT-COUNT*4 IN LOW BYTE, ROUNDING-OPERAND IN HIGH BYTE
;22442  ;       ID[T0] HAS SRC-ADDRESS
;22443  ;       ID[T1] HAS LOW DST-ADDRESS
;22444  ;       ID[T2] HAS ROUNDING OPERAND
;22445
;22446  ;STATE-REGISTER:
;22447  ;-------------------------------------------------------------------;
;22448  ;INTRPT ;OVFLOW ;1.TIME ;COUNT  ;CARRY  ;1.TIME ;SIGN   ;        ;
;22449  ;       ;       ;READ   ;DIRECT ;       ;WRITE  ;       ;        ;
;22450  ;       ;       ;       ;       ;       ;       ;       ;        ;
;22451  ;       ;       ;       ;       ;       ;       ;       ;        ;
;22452  ;-------------------------------------------------------------------;
```

```
                                ;22453  3CC:    ;ENTER HERE FROM C-FORK WITH Q=COUNT,D=SOURCE-LENGTH
                                ;22454  ASHP.INIT:
                                ;22455          ;-------------------------------;
                                ;22456          ALU_D.OXT[WORD],RC[T0]_ALU,     ;  SAVE SRC-LENGTH IN RC0
                                ;22457          D_ALU,QK/LEFT2,                 ;  MULTIPLY COUNT BY 4
U 03CC, 0803,403C,018F,F980,0000,0602   ;22458          SGN/CLR.SD+SS                   ;  CLEAR SS FOR LATER BRANCHING
                                ;22459  =000**1*
                                ;22460          ;000**1*-------------------------;
                                ;22461          ALU_D.AND.K[.FFE0],             ;  ISLOATE SRC-LENGTH
                                ;22462          RC[T2]_ALU,                     ;  CLEAR FOR STORAGE-REGISTER
U 0602, 0C19,0035,A180,F990,0050,047E   ;22463          N&Z_ALU.V&C_0,D_Q,CALL,J/ASPC   ;  CLOCK LENGTH, EVALUATE SRC-ADDRESS
                                ;22464  =110**1*
                                ;22465  ASH.I1: ;110**1*-------------------------;
                                ;22466          ID[T0]_D,                       ;  SAVE SRC-ADDRESS IN T0
                                ;22467          ALU_Q.OXT[WORD],RC[T4]_ALU,     ;  INITIALIZE RC4 WITH 4*SHIFT-COUNT
                                ;22468          D_ALU,SC_ALU,                   ;  LOAD IT IN SC AS WELL
U 0662, 0803,603D,C180,3DA0,0082,037E   ;22469          CALL,J/SPEC                     ;  EVALUATE ROUNDING OPERAND
                                ;22470  =111**1*
                                ;22471          ;111**1*-------------------------;
U 0672, 0819,0034,6180,F800,0000,0C4E   ;22472          ALU_D.AND.K[.F],D_ALU           ;  ISOLATE LOW NIBBLE
                                ;22473  =;END
                                ;22474
                                ;22475  ; ************************************************
                                ;22476  ; * Patch no. 059, PCS 0672 trapped to WCS 1185 *
                                ;22477  ; ************************************************
                                ;22478
                                ;22479          ;-------------------------------;
                                ;22480          D_D.SWAP,ID[T2]_D,              ;  SAVE ROUNDING OPERAND IN T2
U 0C4E, 0800,003C,C980,3C00,0010,01C6   ;22481          EALU_SC,CLK.UBCC,J/ASHP.I2      ;  CLOCK DIRECTION OF SHIFT
                                ;22482
                                ;22483  =0*100**1*
                                ;22484  =0*111**1*
                                ;22485  ASHP.REEN.0:
                                ;22486          ;0*111**1*-----------------------;
                                ;22487          ALU_Q.OXT[WORD],RC[T4]_ALU,     ;  INITIALIZE RC4 WITH SHIFT-COUNT
                                ;22488          EALU_SC,CLK.UBCC,               ;  CLOCK DIRECTION OF SHIFT
U 00F6, 0003,603D,C980,3DA0,0010,0C99   ;22489          ID[T2]_D,CALL,J/RESTORE.BCD     ;  SAVE ROUNDING OPERAND
                                ;22490  =1*100**1*
                                ;22491  ASHP.I2:
                                ;22492          ;1*100**1*-----------------------;
                                ;22493          ALU_D.OR.Q,RC[T5]_ALU,          ;  SAVE BOTH COUNT AND ROUNDING IN RC5
U 01C6, 001D,0031,0180,F9A8,0000,037E   ;22494          CALL,J/SPEC                     ;  EVALUATE DST-LENGTH
                                ;22495  =1*101**1*
                                ;22496  DC.PA.59:
                                ;22497          ;1*101**1*-----------------------;
                                ;22498          ALU_D.OXT[WORD],RC[T1]_ALU,     ;  STORE  DST-LENGTH IN RC1
U 01D6, 0803,403D,0180,F988,0000,047E   ;22499          D_ALU,CALL,J/ASPC               ;  EVALUATE DST-ADDRESS
                                ;22500  =1*111**1*
                                ;22501  ASHP.REEN:
                                ;22502          ;1*111**1*-----------------------;
                                ;22503          ALU_0-Q-1,R[R2]_ALU,            ;  STORE NEGATIVE DST-LENGTH IN R2
                                ;22504          QK/RIGHT,ID[T1]_D,              ;  SAVE DST-ADDRESS IN T1
U 01F6, 001F,1208,C5B0,3E90,0000,05F6   ;22505          EALU?,J/ASH.I10                 ;  TEST DIRECTION OF SHIFT
                                ;22506  =
```

```
                                    ;22507  =011*   ;BRANCH ON EALU N-BIT (SS=0)
                                    ;22508          ;011*-------------------------;
                                    ;22509  ASH.I10:
                                    ;22510          STATE_K[ZERO],                  ;  CLEAR STATE-REGISTER
                                    ;22511          ALU_D+Q+1,R[R3]_ALU,            ;  GENERATE HIGH DST-ADDRESS
U 05F6, 001D,0010,1980,FA98,1404,66A2 ;22512        J/ASH.I3
                                    ;22513          ;111*-------------------------;
                                    ;22514          STATE_K[.10],                   ;  SET LEFT-SHIFT-BIT IN STATE
                                    ;22515          ALU_D+Q+1,R[R3]_ALU,            ;  INITIALIZE R3 WITH HIGH DST-ADDRESS
U 05FE, 001D,0010,6580,FA98,1404,66A2 ;22516        J/ASH.I3
                                    ;22517  =;END
                                    ;22518  =0***   ;0***-------------------------;
                                    ;22519  ASH.I3: ALU_RC[T0],D_ALU.RIGHT,         ;  GET SRC-LENGTH, DIVIDE BY 2
U 06A2, 0850,0039,C1F0,2D00,0000,0B24 ;22520        Q_ID[T0],CALL,J/BCD.FPD.00      ;  GET SRC-ADDRESS, SET FPD
                                    ;22521          ;T***-------------------------;
                                    ;22522          ALU_0-LC-1,R[R0]_ALU,           ;  INITIALIZE R0 WITH NEG. SRC-LENGTH
U 06AA, 0013,0008,B580,3E80,0000,0036 ;22523        ID[FPDA]_D,J/ASHP.E             ;  LOAD 1.PART DONE RETURN ADDRESS (33)
                                    ;22524  =;END   ;-------------------------;
                                    ;22525          ;ENTER HERE IN ORDER TO READ NEXT LONGWORD FROM SRC-STRING
                                    ;22526          ;EXPECTS D TO HAVE NEGATIVE SRC-LENGTH, REFLECTED IN ALU CC
                                    ;22527  =10     ;10------ -------------------------;
                                    ;22528  ASHP.E1:
                                    ;22529          LA_RA[R1],                      ;  GET SRC-ADDRESS
                                    ;22530          ALU_D+K[.3],                    ;  INCREMENT LENGTH
                                    ;22531          D_ALU,CLK.UBCC,BYTE,
                                    ;22532          INTRPT.STROBE,                  ;  STROBE INTERRUPTS
U 09EA, 0819,9B15,0D80,F888,4010,0AF7 ;22533        ALU.N?,CALL,J/READ0             ;  TEST LENGTH,READ SRC-STRING
                                    ;22534  =11     ;11-------------------------;
                                    ;22535          ALU_LA-K[.4],                   ;  UPDATE SRC-ADDRESS
                                    ;22536          LC_RC[T4]&R1_ALU,               ;  GET SHIFT-COUNT
U 09EB, 0018,0E00,1180,FBA0,0000,0DFE ;22537        BEN/INTERRUPT                   ;  TEST FOR PENDING INTERRUPTS
                                    ;22538  =;END   ;-------------------------;
                                    ;22539  =110    ;BRANCH ON INTERRUPT-REQUEST
                                    ;22540          ;110-------------------------;
                                    ;22541          FE_K[.20],                      ;  LOAD 1. READ BIT TEMPORARILY IN FE
                                    ;22542          LAB_R[R0],ALU_LC,SC_ALU,        ;  STORE COUNT IN SC
U 0DFE, 0010,0038,7580,FA00,0186,6C50 ;22543        J/ASHP.E2
                                    ;22544          ;111-------------------------;
                                    ;22545          STATE_K[.80],                   ;  SET INTERRUPT-BIT OF STATE
U 0DFF, 0000,003C,4180,F800,1404,6033 ;22546        J/SAVE.BCD                      ;  ROUTINE TO SAVE CONTEXT
                                    ;22547  =;END   ;-------------------------;
```

```
                                    ;22548  ASHP.E2:       R[R0]_LA+K[.8],          ;  UPDATE SRC-LENGTH
                                    ;22549          EALU_SC+FE,CLR.UBCC,           ;  CLOCK SHIFT-COUNT
U 0C50, 0018,1614,0180,FA80,0010,8E04   ;22550          STATE7-4?                      ;  TEST 1.TIME AND COUNT-SIGN
                                    ;22551          ;-----------------------------;
                                    ;22552  =100     ;BRANCH ON 1. READ AND POS/NEG-BITS OF STATE
                                    ;22553          ;100-------------------------;
                                    ;22554          STATE_STATE.OR.FE,             ;  SET 1. READ BIT
                                    ;22555          D_D.ANDNOT.K[.F],              ;  STRIP OFF SIGN-NIBBLE
                                    ;22556          RC[T2]_ALU,Q_0,                ;  SAVE DATA IN RC[T2]
U 0E04, 0819,0F24,61F8,F990,1400,2A22   ;22557          BCDSGN?,J/FIRST.POS            ;  TEST DECIMAL SIGN
                                    ;22558          ;101-------------------------;
                                    ;22559          STATE_STATE.OR.FE,             ;  SET 1. READ BIT
                                    ;22560          ALU_D.ANDNOT.K[.F],            ;  STRIP OFF SIGN-NIBBLE
                                    ;22561          RC[T2]_ALU,                    ;  SAVE DATA
U 0E05, 0019,0F24,6180,F990,1400,2A02   ;22562          BCDSGN?,J/FIRST.NEG            ;  TEST DECIMAL SIGN
                                    ;22563          ;110-------------------------;
                                    ;22564  POS.CNT:       STATE_STATE.OR.FE,       ;  SET 1. READ BIT (RECALL THAT FE=20)
U 0E06, 0010,1438,01C0,F910,1400,2026   ;22565          Q_RC[T2],SC?,J/POS.1           ;  GET LEFT-OVER DATA, TEST COUNT
                                    ;22566          ;111-------------------------;
                                    ;22567          STATE_STATE.OR.FE,             ;  SET 1. READ BIT
                                    ;22568          Q_D,D_RC[T2],                  ;  GET LEFT-OVER DATA(INITIALLY 0)
U 0E07, 0810,1238,01E0,F910,1400,22A6   ;22569          EALU?,J/NEG.CNT                ;  SHIFT-COUNT IS NEGATIVE, TEST IT
                                    ;22570  =;END    ;-----------------------------;
                                    ;22571          ;ENTER HERE IF COUNT IS NON-NEGATIVE, NOT SIGN-BYTE
                                    ;22572          ;-----------------------------;
                                    ;22573  =0*110   ;BRANCH ON SC<9-5> NE 0
                                    ;22574          ;0*110-----------------------;
                                    ;22575  POS.1:   RC[T2]_D,                      ;  STORE NEWLY READ DATA IN RC2
                                    ;22576          D_DAL.SC,                      ;  SHIFT CURRENT DATA INTO PLACE
U 0026, 0D01,003D,0180,F990,0000,0C52   ;22577          CALL,J/ASHP.WRITE              ;  WRITE RESULT IN DST-STRING
                                    ;22578          ;0*111-----------------------;
                                    ;22579          SC_SC-K[.20],D_0,              ;  STILL WRITING TRAILING 0'S
U 0027, 0F01,003D,7580,F990,0084,AC52   ;22580          RC[T2]_D,CALL,J/ASHP.WRITE
                                    ;22581          ;1*110-----------------------;
                                    ;22582  ASHP.E:  ALU_R[R0],D_ALU.RIGHT,         ;  GET SRC-LENGTH
U 0036, 0840,803C,0180,FA00,0010,09EA   ;22583          CLK.UBCC,BYTE,J/ASHP.E1        ;  READ NEXT LONGWORD
                                    ;22584          ;1*111-----------------------;
U 0037, 0010,0038,0180,F920,0082,0C51   ;22585          SC_RC[T4]                      ;  RETRIEVE SHIFT-COUNT
                                    ;22586  =;END    ;-----------------------------;
                                    ;22587  POS.2:   Q_0,D_RC[T2],                  ;  INSERT TRAILING 0'S
U 0C51, 0810,1438,01F8,F910,0000,0026   ;22588          SC?,J/POS.1                    ;  TEST COUNT
```

```
                                    ;22589  ASHP.WRITE:
                                    ;22590          ;-------------------------------;
U 0C52, 0018,1738,1D80,F9A0,0000,015A ;22591        RC[T4]_K[SC],STATE3-0?          ; SAVE SHIFT-COUNT, TEST 1.TIME-BIT
                                    ;22592          ;-------------------------------;
                                    ;22593  =101*   ;BRANCH ON 1.TIME-BIT OF STATE
                                    ;22594          ;101*-----------------------------;
U 015A, 0819,0024,6180,F800,0000,015E ;22595  ASH.W0: D_D.ANDNOT.K[.F]               ; CLEAR SIGN-NIBBLE
                                    ;22596          ;T11*-----------------------------;
U 015E, 0000,803C,01C0,FA10,0010,0594 ;22597        Q_R[R2],CLK.UBCC,BYTE           ; GET DST-LENGTH
                                    ;22598  =;END   ;-------------------------------;
                                    ;22599  =00*****
                                    ;22600  ASH.W1: LA_RA[R3],                      ; GET DST-ADDRESS
                                    ;22601          ALU_Q+K[.8],                    ; INCREMENT NIBBLE-COUNT
                                    ;22602          SC_ALU,                         ; STORE IN SC
                                    ;22603          SHF/ALU.DT,LONG,                ; MULTIPLY BY 4
                                    ;22604          QK/SHF,                         ; LOAD IN SC
                                    ;22605          CLK.UBCC,
                                    ;22606          SIGNS?,                         ; TEST OVERFLOW AND END OF STRING
U 0594, 0079,2D15,01C0,F898,0092,0E59 ;22607        CALL,J/WRITE1                   ; CALL 'WRITE-BCD'-SUBROUTINE
                                    ;22608          ;01*****-------------------------;
                                    ;22609          ALU_R[R0],D_ALU.RIGHT,
                                    ;22610          CLK.UBCC,BYTE,                  ; CLOCK SRC-LENGTH
                                    ;22611          Q_ID[T0],                       ; GET SRC-ADDRESS
U 05B4, 0840,823C,C1F0,2E00,0010,0E15 ;22612        ROR?,J/ASHP.FIN                 ; TEST FOR CARRY
                                    ;22613          ;-------------------------------;
                                    ;22614  =11*****
                                    ;22615          STATE_STATE.OR.K[.4],            ; SET 1. WRITE BIT OF STATE
                                    ;22616          R[R3]_LA-K[.4],                 ; UPDATE DST-ADDRESS
U 05F4, 0018,0200,1180,FA98,1404,2E0D ;22617        ROR?                            ; TEST PSL CARRY BIT
                                    ;22618  =;END   ;-------------------------------;
                                    ;22619  =101    ;BRANCH ON PSL C-BIT
                                    ;22620          ;101------------------------------;
                                    ;22621          STATE_STATE.ANDNOT.K[.8],       ; CLEAR CARRY-BIT OF STATE
                                    ;22622          R[R2]_Q+K[.8],                  ; UPDATE DST-LENGTH
U 0E0D, 0019,2016,0180,FA90,1404,4010 ;22623        RETURN10
                                    ;22624          ;111------------------------------;
                                    ;22625          STATE_STATE.OR.K[.8],           ; SET CARRY-BIT OF STATE
U 0E0F, 0019,2016,0180,FA90,1404,2010 ;22626        R[R2]_Q+K[.8],RETURN10          ; UPDATE DST-LENGTH
                                    ;22627  =;END   ;-------------------------------;
```

```
                                        ;22628          ;ENTER HERE IF COUNT WAS ORIGINALLY NEGATIVE
                                        ;22629          ;Q HAS LONG-WORD JUST READ, D HAS PREVIOUS LONGWORD
                                        ;22630          ;THE NEW DATA GETS SHIFTED TOGETHER WITH THE OLD,
                                        ;22631          ;AND THE COUNT GETS INCREMENTED.
                                        ;22632          ;-------------------------------;
                                        ;22633  =0011*  ;BRANCH ON EALU N-BIT (SS=0)
                                        ;22634          ;0011*---------------------------;
                                        ;22635  NEG.CNT:
                                        ;22636          D_DAL.SC,ALU_0+Q,RC[T2]_ALU,    ;  SAVE NEW DATA IN RC2
                                        ;22637          Q_DEC.CON,                      ;  GET READY FOR DECIMAL ADD
U 02A6, 0D1F,1715,01D0,F990,0000,0642   ;22638          CALL,STATE3-0?,J/NEG.2
                                        ;22639          ;0111*---------------------------;
                                        ;22640          RC[T2]_Q,SC_SC+K[.20],          ;  INCREMENT SHIFT-COUNT
U 02AE, 0001,203C,7580,F990,0084,8C53   ;22641          J/NEG.5
                                        ;22642          ;1011*---------------------------;
                                        ;22643  =1011*  ALU_R[R0],D_ALU.RIGHT,          ;  GET SRC-LENGTH
U 02B6, 0840,803C,0180,FA00,0010,09EA   ;22644          CLK.UBCC,BYTE,J/ASHP.E1
                                        ;22645  =;END   ;-------------------------------;
U 0C53, 0018,0038,1D80,F9A0,0000,0036   ;22646  NEG.5:  RC[T4]_K[SC],J/ASHP.E           ;  STORE SHIFT-COUNT
                                        ;22647          ;-------------------------------;
                                        ;22648  =001*   ;BRANCH ON CARRY AND FIRST WRITE BITS OF STATE
                                        ;22649          ;001*---------------------------;
                                        ;22650  NEG.2:  ALU_D+Q,D_ALU,                  ;  ADD 6'S TO STRING
U 0642, 081D,0014,C9F0,2C00,0000,0C54   ;22651          Q_ID[T2],J/NEG.4                ;  GET ROUNDING OPERAND
                                        ;22652          ;011*---------------------------;
U 0646, 0000,803C,01C0,FA10,0010,0594   ;22653  NEG.20: Q_R[R2],CLK.UBCC,BYTE,J/ASH.W1  ;  GET DST-LENGTH, WRITE-ROUTINE
                                        ;22654          ;101*---------------------------;
                                        ;22655  NEG.3:  D_D-Q,                          ;  DECIMAL ADJUST
U 064A, 081D,1700,0180,F800,0000,015A   ;22656          STATE3-0?,J/ASH.W0              ;  TEST FIRST TIME BIT
                                        ;22657          ;111*---------------------------;
                                        ;22658          ALU_D+Q+1,D_ALU,                ;  ADD 6'S AND 1 FOR CARRY
                                        ;22659          SET.CC(LONG),                   ;  CLOCK PSL C-BIT
U 064E, 081D,0010,01D0,F800,0070,064A   ;22660          Q_DEC.CON,J/NEG.3               ;  Q GETS DECIMAL ADJUSTMENT
                                        ;22661  =;END   ;-------------------------------;
                                        ;22662  NEG.4:  ALU_Q.OXT[BYTE]+D,D_ALU,        ;  ADD ROUNDING OPERAND
                                        ;22663          Q_DEC.CON,                      ;  Q GETS DECIMAL ADJUSTMENT
U 0C54, 081F,A014,01D0,F800,0070,064A   ;22664          SET.CC(BYTE),J/NEG.3
                                        ;22665
                                        ;22666  ; ************************************************
                                        ;22667  ; * Patch no. 039, PCS 0C54 trapped to WCS 1173 *
                                        ;22668  ; ************************************************
```

```
                                    ;22669          ;ENTER AFTER READING FIRST LONGWORD, TO TEST FOR SIGN.
                                    ;22670          ;-------------------------------;
                                    ;22671  =10     ;BRANCH ON SIGN NIBBLE
                                    ;22672          ;10-------------------------------;
                                    ;22673  FIRST.NEG:
                                    ;22674          ALU_R[R0],D_ALU.RIGHT,          ;  GET SRC-LENGTH
U 0A02, 0840,803C,0180,FA00,0010,09EA ;22675        CLK.UBCC,BYTE,J/ASHP.E1         ;  READ NEXT LONGWORD
                                    ;22676          ;11-------------------------------;
                                    ;22677          STATE_STATE.OR.K[.2],           ;  SET MINUS SIGN-BIT OF STATE
                                    ;22678          ALU_R[R0],D_ALU.RIGHT,          ;  GET SRC-LENGTH
U 0A03, 0840,803C,0980,FA00,1414,29EA ;22679        CLK.UBCC,BYTE,J/ASHP.E1         ;  READ NEXT LONGWORD
                                    ;22680  =;END   ;-------------------------------;
                                    ;22681  =10     ;BRANCH ON SIGN-NIBBLE
                                    ;22682          ;10-------------------------------;
                                    ;22683  FIRST.POS:
U 0A22, 0000,143C,0180,F800,0000,0026 ;22684        SC?,J/POS.1                     ;  TEST SHIFT-COUNT
                                    ;22685          ;11-------------------------------;
                                    ;22686          STATE_STATE.OR.K[.2],           ;  SET SIGN-BIT TO NEGATIVE
U 0A23, 0000,143C,0980,F800,1404,2026 ;22687        SC?,J/POS.1                     ;  TEST SHIFT-COUNT
                                    ;22688  =;END   ;-------------------------------;
                                    ;22689
                                    ;22690          ;ENTER HERE AFTER REACHING END OF DST-STRING
                                    ;22691  =101    ;BRANCH ON PSL CARRY BIT
                                    ;22692          ;101------------------------------;
                                    ;22693  ASHP.FIN:
U 0E15, 0818,1B38,8580,F800,0000,01B5 ;22694        D_K[.C],ALU?,J/ASH.F2           ;  TEST SRC-LENGTH
                                    ;22695          ;T11------------------------------;
                                    ;22696          ALU_0+Q,SET.CC(LONG),           ;  CLEAR C-BIT
U 0E17, 001F,0014,3180,F800,1474,21B5 ;22697        STATE_STATE.OR.K[.40]           ;  SET OVERFLOW-BIT OF STATE
                                    ;22698  =;END   ;-------------------------------;
                                    ;22699  =01*1   ;BRANCH ON ALU N-BIT
                                    ;22700          ;01*1-----------------------------;
                                    ;22701  ASH.F2: ALU_0(A),R[R0]_ALU,             ;  CLEAR R0
                                    ;22702          N_MUX.Z_TST,                    ;  CLEAR N-BIT
                                    ;22703          STATE_STATE.ANDNOT.K[.30],      ;  USE THESE BITS IN FINISH-ROUTINE
U 01B5, 0003,173C,7980,FA80,1434,488D ;22704        STATE3-0?,J/FINI1               ;  TEST SIGN-BIT
                                    ;22705          ;11*1-----------------------------;
U 01BD, 0010,003A,0180,F920,0082,0010 ;22706        SC_RC[T4],RETURN10              ;  GET SHIFT-COUNT
                                    ;22707  =;END   ;-------------------------------;
```

```
                                    ;22708  .TOC    ""      Decimal string        : BCD-READ SUBROUTINE""
                                    ;22709
                                    ;22710          ;SUBROUTINE WHICH READS FROM 0 TO 4 BYTES OF DATA
                                    ;22711          ;FROM MEMORY,STARTING IN ADDRESS DETERMINED BY LA,
                                    ;22712          ;USING ~COUNT/2 IN D.
                                    ;22713          ;CONDITION CODES REFLECT COUNT.
                                    ;22714          ;RETURNS DATA IN ALGEBRAIC ORDER, FILLED OUT WITH 0'S,
                                    ;22715          ;IN D-REGISTER.
                                    ;22716          ;RETURN IS MADE WITH SC=FE,Q=DECIMAL CONSTANT=66666666
                                    ;22717          ;ENTER AT READ0 IF YOU WANT A STRAIGHT READ
                                    ;22718          ;ENTER AT READ00 IF YOU WANT A READ/W WRITE CHECK
                                    ;22719          ;(DEPENDING ON LOW BIT OF OP-CODE).
                                    ;22720          ;-------------------------------;   ***READ SUBROUTINE***
                                    ;22721  =0101   ;BRANCH ON N-BIT OF ALU AND IR<0>
                                    ;22722          ;0101-----------------------------;
                                    ;22723  READ00: D_0,SC_FE,ALU_D+K[ZERO],
U 0AF5, 0F19,0016,19D0,F800,0081,0001 ;22724        Q_DEC.CON,RETURN1               ;   END OF INPUT-STRING
                                    ;22725          ;0111-----------------------------;
                                    ;22726  READ0:  D_0,SC_FE,ALU_D+K[ZERO],
U 0AF7, 0F19,0016,19D0,F800,0081,0001 ;22727        Q_DEC.CON,RETURN1               ;   END OF INPUT-STRING
                                    ;22728          ;T101-----------------------------;
                                    ;22729          ALU_LA-K[.4],                   ;   GET ADDRESS
U 0AFD, 0018,1B00,1180,F800,0200,0B27 ;22730        VAK7LOAD,ALU.N?,J/READ1W        ;   TEST FOR WHOLE LONG-WORD
                                    ;22731                                          ;   SWITCH TO READ-W-WRITE-CHK ROUTINE
                                    ;22732          ;1111-----------------------------;
                                    ;22733          ALU_LA-K[.4],                   ;   GET ADDRESS
U 0AFF, 0018,1B00,1180,F800,0200,0B07 ;22734        VAK7LOAD,ALU.N?                 ;   TEST FOR WHOLE LONG-WORD
                                    ;22735  =;END   ;-------------------------------;
                                    ;22736  =0111   ;BRANCH ON N-BIT OF ALU
                                    ;22737          ;0111-----------------------------;
                                    ;22738  READ1:  VA_LA-K[.1],                    ;   LOAD ADDRESS WITH GUESS
U 0B07, 0018,0C00,0580,F800,0200,0E24 ;22739        MUL?,J/READ2                    ;   LESS THAN A LONGWORD LEFT
                                    ;22740          ;1111-----------------------------;
                                    ;22741          D[BYTE]_CACHE,                  ;   READ BYTE(MAY BE WHOLE LONGWORD)
U 0B0F, 0000,823C,0180,4000,0000,0E1A ;22742        ROR?                            ;   BRANCH ON ADDRESS
                                    ;22743  =;END   ;-------------------------------;
                                    ;22744  =010    ;BRANCH ON LOW 2 BITS OF LA
                                    ;22745          ;010------------------------------;
                                    ;22746  R4:     D_D.SWAP,SC_FE,                 ;   GET DATA IN ARITHMETIC ORDER
                                    ;22747          ALU_0+Q,Q_DEC.CON,              ;   LOAD ALL 6'S
U 0E1A, 0B1F,0016,01D0,F800,0081,0003 ;22748        RETURN3                         ;   GOT IT ALREADY
                                    ;22749          ;011------------------------------;
                                    ;22750          D_D.SWAP,                       ;   PUT DATA IN ALGEBRAIC ORDER
                                    ;22751          ALU_LA-K[.1],                   ;   CHANGE ADDRESS
U 0E1B, 0B18,0000,0580,F800,0200,0C55 ;22752        VAK7LOAD,J/R401                 ;   ADDRESS NOT ALIGNED
                                    ;22753          ;110------------------------------;
U 0E1E, 0B00,C03C,0180,F800,0200,0C58 ;22754        D_D.SWAP,VA_LA,J/R410           ;   GET READY FOR NEXT READ
                                    ;22755          ;T11------------------------------;
                                    ;22756          Q_D,                            ;   SAVE FIRST BYTE IN Q
                                    ;22757          ALU_LA-K[.3],                   ;   CHANGE ADDRESS
U 0E1F, 0018,0000,0DE0,F800,0200,0C5B ;22758        VAK7LOAD,J/R411
                                    ;22759  =;END   ;-------------------------------;
```

```
                                        ;22760  R401:   ALU_D.ANDNOT.K[.FF],Q_ALU,      ;  SAVE 3 HIGH BYTES IN Q
U 0C55, 0019,8024,49C0,4000,0000,0C56   ;22761          D[BYTE]_CACHE                    ;  READ ANOTHER IN D
                                        ;22762          ;--------------------------------;
                                        ;22763  R4010:  D_D.OXT[BYTE]+Q,SC_FE,           ;  ASSEMBLE BYTE
U 0C56, 081F,8016,01D0,F800,0081,0003   ;22764          Q_DEC.CON,RETURN3
                                        ;22765          ;--------------------------------;
                                        ;22766  R410:   ALU_D.ANDNOT.K[.FFFF],Q_ALU,    ;  SAVE FIRST WORD IN Q
U 0C58, 0019,4024,C1C0,4000,0000,0C59   ;22767          D[WORD]_CACHE                    ;  GET NEXT WORD
                                        ;22768          ;--------------------------------;
U 0C59, 0B00,003C,0180,F800,0000,0C5A   ;22769  R4100:  D_D.SWAP                         ;  IN ALGEBRAIC ORDER
                                        ;22770          ;--------------------------------;
                                        ;22771          D_D.OXT[WORD]+Q,                 ;  "OR" LOW WORD OF D WITH HIGH OF Q
U 0C5A, 081F,4016,01D0,F800,0081,0003   ;22772          Q_DEC.CON,SC_FE,RETURN3          ;  ASSEMBLE THE DATA
                                        ;22773          ;--------------------------------;
                                        ;22774  R411:   D[BYTE]_CACHE,
U 0C5B, 0000,803C,7180,4000,0084,6E27   ;22775          SC_K[.FFF8],J/R2?                ;  GET REST OF DATA
                                        ;22776          ;--------------------------------;
```

```
                                ;22777          ;-------------------------------;
                                ;22778  =100     ;BRANCH ON 2 LOW BITS OF D
                                ;22779          ;100-----------------------------;
                                ;22780  READ2:   ALU_LA-K[.3],VAK/LOAD,          ; READ 3 BYTES
U 0E24, 0018,0200,0D80,F800,0200,0E2A ;22781    ROR?,J/R3                       ; TEST LOW BITS OF LA
                                ;22782          ;101-----------------------------;
                                ;22783          ALU_LA-K[.2],VAK/LOAD,          ; READ 2 BYTES
U 0E25, 0018,0200,0980,F800,0200,0E36 ;22784    ROR?,J/R2                       ; BRANCH ON ADDRESS-BITS
                                ;22785          ;110-----------------------------;
U 0E26, 0000,803C,0180,4000,0000,0C5C ;22786    D[BYTE]_CACHE                   ; READ 1 BYTE
                                ;22787  =;END    ;-------------------------------;
                                ;22788  READ20:  D_D.OXT[BYTE]+K[ZERO],SC_FE,
U 0C5C, 081B,8016,19D0,F800,0081,0003 ;22789    Q_DEC.CON,RETURN3               ; CLEAR UPPER 3 BYTES, RETURN
                                ;22790          ;-------------------------------;
                                ;22791  =010     ;BRANCH ON LOW TWO BITS OF LA
                                ;22792          ;010-----------------------------;
                                ;22793  R3:      D[BYTE]_CACHE,Q_0,              ; READ FIRST BYTE
U 0E2A, 0000,803C,71F8,4000,0084,6E27 ;22794    SC_K[.FFF8],J/R21               ; GET READY TO SHIFT DATA
                                ;22795          ;011-----------------------------;
U 0E2B, 0018,0008,1180,F800,0200,05A0 ;22796    VA_LA-K[.4]-1,J/R300            ; READ 3 BYTES
                                ;22797          ;110-----------------------------;
U 0E2E, 0000,803C,0180,4000,0000,0C5D ;22798    D[BYTE]_CACHE,J/R301            ; READ FIRST OF 3 BYTES
                                ;22799          ;111-----------------------------;
                                ;22800          D[BYTE]_CACHE,Q_0,              ; READ FIRST BYTE
U 0E2F, 0000,803C,71F8,4000,0084,6E27 ;22801    SC_K[.FFF8],J/R21               ; GET READY FOR SHIFT
                                ;22802  =;END    ;-------------------------------;
U 0C5D, 001E,0000,0980,F800,0200,0C60 ;22803  R301:   VA_LA-K[.2]                ; NEW ADDRESS
                                ;22804          ;-------------------------------;
                                ;22805          D[BYTE]_CACHE,                  ; READ NEXT BYTE
                                ;22806          Q_D.OXT[BYTE],SC_K[.FFF0],      ; ISOLATE PREVIOUS BYTE
U 0C60, 0003,803C,6DC0,4000,0084,6E27 ;22807    J/R21
                                ;22808          ;-------------------------------;
                                ;22809  =0****
U 05A0, 0000,803D,0180,4000,0000,0C61 ;22810  R300:   D[BYTE]_CACHE,CALL,J/SWAP.D ; 1-INSTRUCTION SUBROUTINE
                                ;22811          ;1****---------------------------;
                                ;22812          Q_D.OXT[WORD],                  ; ISOLATE PREVIOUS DATA
                                ;22813          D[WORD]_CACHE,                  ; READ NEXT WORD
U 05B0, 0003,403C,69C0,4000,0084,6C62 ;22814    SC_K[.FFE8],J/R4111             ; GET READY FOR SHIFT
                                ;22815  =;END    ;-------------------------------;
U 0C61, 0B00,003E,0180,F800,0200,0010 ;22816  SWAP.D: D_D.SWAP,VA_LA,RETURN10     ; SUBROUTINE
                                ;22817          ;-------------------------------;
```

```
                                        ;22818  =110    ;BRANCH ON LOW BIT OF LA
                                        ;22819          ;110----------------------------;
                                        ;22820  R2:     D[BYTE]_CACHE,Q_0,              ;  READ NEXT BYTE
U 0E36, 0000,803C,6DF8,4000,0084,6E27   ;22821          SC_K[.FFF0],J/R21               ;  SHIFT IT
                                        ;22822          ;1T1----------------------------;
                                        ;22823          D[BYTE]_CACHE,Q_0,              ;  READ NEXT BYTE
U 0E37, 0000,823C,6DF8,4000,0084,6E23   ;22824          SC_K[.FFF0],ROR?,J/R20          ;  GET READY TO SHIFT, TEST ADDRESS
                                        ;22825  =:END   ;----------------------------;
                                        ;22826  =011    ;BRANCH ON BIT 1 OF LA
                                        ;22827          ;011----------------------------;
                                        ;22828  R20:    VA_LA,                          ;  CHANGE ADDRESS
U 0E23, 0000,003C,6980,F800,0284,6C63   ;22829          SC_K[.FFE8],J/R211
                                        ;22830          ;1T1----------------------------;
U 0E27, 0B00,003C,0180,F800,0000,0C62   ;22831  R21:    D_D.SWAP                        ;  ARITHMETIC ORDER
                                        ;22832  =:END   ;----------------------------;
                                        ;22833  R4111:  D_DAL.SC,SC_FE,                 ;  SHIFT DATA INTO PLACE,RIGHT ADJUSTED
                                        ;22834          ALU_0+Q,
U 0C62, 0D1F,0016,01D0,F800,0081,0003   ;22835          Q_DEC.CON,RETURN3               ;  RETURN WITH DECIMAL CONSTANT
                                        ;22836          ;----------------------------;
                                        ;22837  R211:   D[BYTE]_CACHE,
U 0C63, 0003,803C,01C0,4000,0000,0C62   ;22838          Q_D.OXT[BYTE],J/R4111
                                        ;22839          ;----------------------------;
                                        ;22840
```

```
                                    ;22841  .TOC    "      Decimal string        : BCD-READ-WITH-WRITE-CHECK SUBROUTINE"
                                    ;22842
                                    ;22843          ;ROUTINE WHICH READS FROM 0 TO 4 BYTES OF DATA
                                    ;22844          ;FROM MEMORY,STARTING IN ADDRESS DETERMINED BY LA,
                                    ;22845          ;USING -COUNT/2 IN D.
                                    ;22846          ;CONDITION CODES REFLECT COUNT.
                                    ;22847          ;RETURNS DATA IN ALGEBRAIC ORDER, FILLED OUT WITH 0'S,
                                    ;22848          ;IN D-REGISTER.
                                    ;22849          ;RETURN IS MADE WITH SC=FE,Q=DECIMAL CONSTANT=66666666
                                    ;22850          ;-------------------------------;   ***READ SUBROUTINE***
                                    ;22851  =0111   ;BRANCH ON N-BIT OF ALU
                                    ;22852          ;0111---------------------------;
                                    ;22853  READ0W: D_0,SC_FE,ALU_D+K[ZERO],
U 0B17, 0F19,0016,19D0,F800,0081,0001 ;22854        Q_DEC.CON,RETURN1               ;  END OF INPUT-STRING
                                    ;22855          ;1111---------------------------;
                                    ;22856          ALU_LA-K[.4],                   ;  GET ADDRESS
U 0B1F, 0018,1B00,1180,F800,0200,0B27 ;22857        VAK7LOAD,ALU.N?                 ;  TEST FOR WHOLE LONG-WORD
                                    ;22858  =;END   ;-------------------------------;
                                    ;22859  =0111   ;BRANCH ON N-BIT OF ALU
                                    ;22860          ;0111---------------------------;
                                    ;22861  READ1W: VA_LA-K[.1],
U 0B27, 0018,0C00,0580,F800,0200,0E44 ;22862        MUL?,J/READ2W                   ;  LESS THAN A LONGWORD LEFT
                                    ;22863          ;1111---------------------------;
                                    ;22864          D[BYTE]_CACHE.WCHK,             ;  READ BYTE(MAY BE WHOLE LONGWORD)
U 0B2F, 0000,823C,0180,5000,0000,0E3A ;22865        ROR?                            ;  BRANCH ON ADDRESS
                                    ;22866  =;END   ;-------------------------------;
                                    ;22867  =010    ;BRANCH ON LOW 2 BITS OF LA
                                    ;22868          ;010----------------------------;
                                    ;22869  R4W:    D_D.SWAP,SC_FE,                 ;  PUT DATA IN ARITHMETIC ORDER
                                    ;22870          ALU_0+Q,Q_DEC.CON,              ;  LOAD ALL 6'S
U 0E3A, 0B1F,0016,01D0,F800,0081,0003 ;22871        RETURN3                         ;  GOT IT ALREADY
                                    ;22872          ;011----------------------------;
                                    ;22873          D_D.SWAP,                       ;  PUT DATA IN ALGEBRAIC ORDER
                                    ;22874          ALU_LA-K[.1],                   ;  CHANGE ADDRESS
U 0E3B, 0B18,0000,0580,F800,0200,0C64 ;22875        VAK7LOAD,J/R401W                ;  ADDRESS NOT ALIGNED
                                    ;22876          ;110----------------------------;
U 0E3E, 0B00,003C,0180,F800,0200,0C65 ;22877        D_D.SWAP,VA_LA,J/R410W          ;  PUT DATA IN ARITHMETIC ORDER
                                    ;22878          ;111----------------------------;
                                    ;22879          Q_D,                            ;  SAVE FIRST BYTE IN Q
                                    ;22880          ALU_LA-K[.3],                   ;  CHANGE ADDRESS
U 0E3F, 0018,0000,0DE0,F800,0200,0C66 ;22881        VAK7LOAD,J/R411W
                                    ;22882  =;END   ;-------------------------------;
```

```
                                    ;22883  R401W:  ALU_D.ANDNOT.K[.FF],Q_ALU,      ;  SAVE 3 HIGH BYTES IN Q
                                    ;22884          D[BYTE]_CACHE.WCHK,            ;  READ ANOTHER IN D
U 0C64, 0019,8024,49C0,5000,0000,0C56 ;22885        J/R4010
                                    ;22886          ;-----------------------------;
                                    ;22887  R410W:  ALU_D.ANDNOT.K[.FFFF],Q_ALU,   ;  SAVE FIRST WORD IN Q
                                    ;22888          D[WORD]_CACHE.WCHK,            ;  GET NEXT WORD
U 0C65, 0019,4024,C1C0,5000,0000,0C59 ;22889        J/R4100
                                    ;22890          ;-----------------------------;
                                    ;22891  R411W:  D[BYTE]_CACHE.WCHK,
U 0C66, 0000,803C,7180,5000,0084,6E47 ;22892        SC_K[.FFF8],J/R21W             ;  GET REST OF DATA
                                    ;22893          ;-----------------------------;
                                    ;22894  =100    ;BRANCH ON 2 LOW BITS OF D
                                    ;22895          ;100-------------------------;
                                    ;22896  READ2W: ALU_LA-K[.3],VAK/LOAD,         ;  READ 3 BYTES
U 0E44, 0018,0200,0D80,F800,0200,0E4A ;22897        ROR?,J/R3W                     ;  TEST LOW BITS OF LA
                                    ;22898          ;101-------------------------;
                                    ;22899          ALU_LA-K[.2],VAK/LOAD,         ;  READ 2 BYTES
U 0E45, 0018,0200,0980,F800,0200,0E56 ;22900        ROR?,J/R2W                     ;  BRANCH ON ADDRESS-BITS
                                    ;22901          ;110-------------------------;
                                    ;22902          D[BYTE]_CACHE.WCHK,            ;  READ 1 BYTE
U 0E46, 0000,803C,0180,5000,0000,0C5C ;22903        J/READ20
                                    ;22904  =;END    ;-----------------------------;
                                    ;22905  =010    ;BRANCH ON LOW TWO BITS OF LA
                                    ;22906          ;010-------------------------;
                                    ;22907  R3W:    D[BYTE]_CACHE.WCHK,Q_0,        ;  READ FIRST OF 3 BYTES
U 0E4A, 0000,803C,71F8,5000,0084,6E47 ;22908        SC_K[.FFF8],J/R21W             ;  GET READY TO SHIFT
                                    ;22909          ;011-------------------------;
U 0E4B, 0018,0008,1180,F800,0200,0660 ;22910        VA_LA-K[.4]-1,J/R300W          ;  READ 3 BYTES
                                    ;22911          ;110-------------------------;
U 0E4E, 0000,803C,0180,5000,0000,0C68 ;22912        D[BYTE]_CACHE.WCHK,J/R301W     ;  READ FIRST BYTE
                                    ;22913          ;111-------------------------;
                                    ;22914          D[BYTE]_CACHE.WCHK,Q_0,        ;  READ FIRST BYTE
U 0E4F, 0000,803C,71F8,5000,0084,6E47 ;22915        SC_K[.FFF8],J/R21W             ;  GET READY TO SHIFT
                                    ;22916  =;END    ;-----------------------------;
U 0C68, 0018,0000,0980,F800,0200,0C69 ;22917  R301W: VA_LA-K[.2]                     ;  ADJUST ADDRESS FOR NEXT BYTE
                                    ;22918          ;-----------------------------;
                                    ;22919          D[BYTE]_CACHE.WCHK,            ;  READ NEXT BYTE
                                    ;22920          Q_D.OXT[BYTE],SC_K[.FFF0],     ;  ISOLATE THE FIRST BYTE
U 0C69, 0003,803C,6DC0,5000,0084,6E47 ;22921        J/R21W
                                    ;22922          ;0****-----------------------;
                                    ;22923  =0****
                                    ;22924  R300W:  D[BYTE]_CACHE.WCHK,            ;  READ NEXT BYTE
U 0660, 0000,803D,0180,5000,0000,0C61 ;22925        CALL,J/SWAP.D                  ;  SUBROUITINE: D_D.SWAP,VA_LA
                                    ;22926          ;1****-----------------------;
                                    ;22927          Q_D.OXT[WORD],                 ;  ISOLATE PREVIOUS DATA
                                    ;22928          D[WORD]_CACHE.WCHK,            ;  READ NEW WORD
U 0670, 0003,403C,69C0,5000,0084,6C62 ;22929        SC_K[.FFE8],J/R4111            ;  GET READY TO SHIFT
                                    ;22930  =;END    ;-----------------------------;
```

```
                                    ;22931  =110     ;BRANCH ON LOW BIT OF LA
                                    ;22932          ;110-------------------------------;
                                    ;22933  R2W:    D[BYTE]_CACHE.WCHK,Q_0,          ;  READ FIRST OF TWO BYTES
U 0E56, 0000,803C,6DF8,5000,0084,6E47 ;22934        SC_K[.FFF0],J/R21W              ;  GET READY TO SHIFT
                                    ;22935          ;1T1-------------------------------;
                                    ;22936          D[BYTE]_CACHE.WCHK,Q_0,          ;  READ FIRST BYTE
U 0E57, 0000,823C,6DF8,5000,0084,6E43 ;22937        SC_K[.FFF0],ROR?,J/R20W         ;  TEST ADDRESS ALIGNMENT
                                    ;22938  =;END   ;-------------------------------;
                                    ;22939  =011    ;BRANCH ON BIT 1 OF LA
                                    ;22940          ;011-------------------------------;
                                    ;22941  R20W:   VA_LA,                          ;  ADJUST ADDRESS
U 0E43, 0000,003C,6980,F800,0284,6C6A ;22942        SC_K[.FFE8],J/R211W             ;  GET SHIFT CONSTANT
                                    ;22943          ;1T1-------------------------------;
U 0E47, 0B00,003C,0180,F800,0000,0C62 ;22944  R21W:  D_D.SWAP,J/R4111                 ;  GET DATA IN ARITHMETIC ORDER
                                    ;22945  =;END   ;-------------------------------;
                                    ;22946  R211W:  D[BYTE]_CACHE.WCHK,             ;  READ NEXT BYTE
U 0C6A, 0003,803C,01C0,5000,C000,0C62 ;22947        Q_D.0XT[BYTE],J/R4111           ;  ISOLATE PREVIOUS DATA
                                    ;22948          ;-------------------------------;
                                    ;22949
```

```
                                    ;22950  .TOC     ""      Decimal string        : BCD-WRITE SUBROUTINE''
                                    ;22951
                                    ;22952          ;SUBROUTINE WHICH WRITES FROM 0 TO 4 BYTES OF
                                    ;22953          ;DATA IN D, DEPENDING ON COUNT IN LB AND Q,
                                    ;22954          ;STARTING IN ADDRESS GIVEN BY LA.
                                    ;22955          ;CONDITION CODES REFLECT COUNT.
                                    ;22956          ;THE Z-BIT OF THE PSL IS UPDATED.
                                    ;22957          ;RC[T7] WILL HAVE ANY OVERFLOW-DATA.
                                    ;22958          ;STATE-REGISTER IS USED TO SIGNAL FIRST TIME THROUGH.
                                    ;22959          ;Q,D,SC,LA,LB,RC[T7],LC, ARE USED.
                                    ;22960          ;STATE-REGISTER:
                                    ;22961  ;-------------------------------------------------------------
                                    ;22962  ;       ;OVFL:  ;       ;       ;       ;1.TIM: ;SGN:   ;       ;
                                    ;22963  ;       ;0=NO   ;       ;       ;       ;0=1.   ;0=POS  ;       ;
                                    ;22964  ;       ;1=YES  ;       ;       ;       ;1=>1.  ;1=NEG  ;       ;
                                    ;22965  ;       ;       ;       ;       ;       ;       ;       ;       ;
                                    ;22966  ;-------------------------------------------------------------
                                    ;22967          ;-------------------------------;
                                    ;22968  WRITE:  ALU_Q+K[.8],                    ;  INCREMENT NIBBLE-COUNT
                                    ;22969          SHF7ALU.DT,LONG,                ;  SHIFT IT LEFT TWICE
                                    ;22970          SC_ALU,                         ;  SAVE IT FOR MASK
                                    ;22971          QK7SHF,
                                    ;22972          CLK.UBCC,                       ;  CLOCK NEW CC
U 0C6B, 0079,2D14,01C0,F938,0092,0E59 ;22973        LC_RC[T7],SIGNS?                ;  RC 7 HAS OVERFLOW DATA
                                    ;22974          ;-------------------------------;
                                    ;22975  =001    ;BRANCH ON Q31 AND D NE 0
                                    ;22976          ;001----------------------------;
                                    ;22977  WRITE1: ALU_D.OR.LC,                    ;  ADD IN NEW OVERFLOW DATA
                                    ;22978          RC[T7]_ALU,                     ;  STORE IT IN RC 7
U 0E59, 0011,0032,0180,F988,0000,0020 ;22979        RETURN20                        ;  ALREADY POSITIVE-END OF DST
                                    ;22980          ;011----------------------------;
                                    ;22981          ALU_D.OR.LC,                    ;  SAVE OVERFLOW DATA
                                    ;22982          RC[T7]_ALU,
                                    ;22983          STATE_STATE.OR.K[.40],          ;  SET OVERFLOW-BIT
U 0E5B, 0011,0032,3180,F988,1404,2020 ;22984        RETURN20                        ;  ALREADY POSITIVE-END OF DST
                                    ;22985          ;101----------------------------;
U 0E5D, 001F,1400,0180,F800,0082,0E65 ;22986        ALU_0-Q,SC_ALU,SC?,J/WRI0       ;  BRANCH ON NEW Q
                                    ;22987          ;111----------------------------;
U 0E5F, 001F,1400,0180,F800,0082,0E65 ;22988        ALU_0-Q,SC_ALU,SC?              ;  BRANCH ON NEW Q
                                    ;22989  =;END   ;-------------------------------;
                                    ;22990  =101    ;BRANCH ON SC GT 0
                                    ;22991          ;101----------------------------;
                                    ;22992  WRI0:   Q_D,ALU_D,N_AMX.Z_TST,          ;  CLOCK Z-BIT
                                    ;22993          SC_K[ZERO],FEK/LOAD,
U 0E65, 0001,173C,19E0,F800,01B4,6B39 ;22994        STATE3-0?,J/WR101               ;  WRITE WHOLE WORD
                                    ;22995          ;111----------------------------;
                                    ;22996          ALU_0+MASK+1,Q_ALU,SC_SC+K[.4], ;  LOAD MASK IN Q
U 0E67, 0003,0010,11C0,F80C,0084,8C6C ;22997        J/WR1
                                    ;22998  =;END   ;-------------------------------;
```

```
                                    ;22999  WR1:    ALU_D.ANDNOT.Q,N_AMX.Z_TST,     ;  GET DATA, CLOCK Z-BIT
                                    ;23000          Q_D,D_ALU,                      ;  MASK OUT LOW PART
U 0C6C, 081D,0024,11E0,F800,0134,AC6D ;23001        FE_SC-K[.4]                     ;  SAVE MASK IN FE
                                    ;23002          ;-------------------------------;
                                    ;23003  WR10:   ALU_Q-D,RC[T7]_ALU,CLK.UBCC,    ;  GET OVERFLOW
                                    ;23004          FE_SC.ANDNOT.K[.4],             ;  MAKE SHIFT-COUNT
                                    ;23005          SC_FE,
U 0C6D, 001D,3700,1180,F9B8,0195,4B39 ;23006        STATE3-0?                       ;  TEST DECIMAL SIGN-BIT OF STATE
                                    ;23007          ;-------------------------------;
                                    ;23008  =1001   ;BRANCH ON STATE 1.TIME AND SIGN-BITS
                                    ;23009          ;1001-----------------------------;
U 0B39, 0819,0030,8580,F800,0000,0B3F ;23010  WR101:  D_D.OR.K[.C],J/WR2              ;  1. TIME , POSITIVE
                                    ;23011          ;T011-----------------------------;
U 0B3B, 0819,0030,8980,F800,0000,0B3F ;23012        D_D.OR.K[.D],J/WR2              ;  1.TIME, NEGATIVE
                                    ;23013          ;T101-----------------------------;
                                    ;23014          SC_FE,FE_SC,                    ;  GET SHIFT-VALUE
                                    ;23015          ALU_LB,RC[PTE.PA]_ALU.RIGHT,    ;  SHIFT LENGTH
U 0B3D, 0B4C,0C38,01F8,F9D8,0181,0E6B ;23016        Q_0,D_D.SWAP,MUL?,J/WR3         ;  BRANCH ON DST-LENGTH
                                    ;23017          ;T111-----------------------------;
                                    ;23018  WR2:    SC_FE,FE_SC,                    ;  GET SHIFT-VALUE
                                    ;23019          ALU_LB,RC[PTE.PA]_ALU.RIGHT,    ;  SHIFT LENGTH
U 0B3F, 0B4C,0C38,01F8,F9D8,0181,0E6B ;23020        Q_0,D_D.SWAP,SC.NE.0?           ;  BRANCH ON DST-LENGTH
                                    ;23021  =;END   ;-------------------------------;
                                    ;23022  =011    ;BRANCH ON SC NE 0
                                    ;23023          ;011------------------------------;
                                    ;23024  WR3:    ALU_LA-K[.4],VAK/LOAD,          ;  LOAD ADDRESS
U 0E6B, 0018,0200,1180,F800,0200,0E82 ;23025        ROR?,J/W400                     ;  WRITE ALL THE DATA
                                    ;23026          ;111------------------------------;
                                    ;23027          ALU_RC[PTE.PA],Q_0,             ;  PASS LENGTH THRU ALU
U 0E6F, 0D10,0138,01F8,F958,0000,0848 ;23028        D_DAL.SC,Z?,J/WR4               ;  RIGHT-ADJUST THE DATA
                                    ;23029  =;END   ;-------------------------------;
                                    ;23030  =0      ;BRANCH ON ALU Z-BIT
                                    ;23031          ;0------------------------------;
                                    ;23032  WR4:    STATE_STATE.OR.K[.40],          ;  SET OVERFLOW-BIT
                                    ;23033          ALU_LA-Q-1,VAK/LOAD,            ;  GENERATE ADDRESS (Q=0)
U 0848, 001C,1508,3180,F800,1604,2ADC ;23034        BEN/ALU1-0,J/W4
                                    ;23035          ;1------------------------------;
                                    ;23036  WR5:    ALU_LA-Q-1,VAK/LOAD,            ;  GENERATE ADDRESS (Q=0)
U 0849, 001C,1508,0180,F800,0200,0ADC ;23037        BEN/ALU1-0                      ;  LOAD-ADDRESS, TEST LENGTH
                                    ;23038  =;END   ;-------------------------------;
```

```
                                    ;23039  ;THE ACTUAL WRITING IS DONE ON THIS PAGE.
                                    ;23040
                                    ;23041
                                    ;23042          ;-------------------------------;
                                    ;23043  =1100   ;BRANCH ON LOW TWO BITS OF ALU
                                    ;23044          ;1100---------------------------;
                                    ;23045  W4:     ALU_LA-K[.4],VAK/LOAD,
U 0ADC, 0018,0200,1180,F800,0200,0E82 ;23046        ROR?,J/W400                     ;  WRITE 4 BYTES
                                    ;23047          ;1101---------------------------;
U 0ADD, 0018,0200,0D80,F800,0200,0E7E ;23048        VA_LA-K[.3],ROR?,J/W300         ;  WRITE 3 BYTES
                                    ;23049          ;1110---------------------------;
U 0ADE, 0018,0200,0980,F800,0200,0E76 ;23050        VA_LA-K[.2],ROR?,J/W200         ;  WRITE 2 BYTES
                                    ;23051          ;1111---------------------------;
                                    ;23052  W100:   CACHE_D[BYTE],                  ;  WRITE LAST BYTE
U 0ADF, 000C,803A,01C0,3000,0000,0060 ;23053        ALU_LB,Q_ALU,RETURN60           ;  GET LENGTH IN Q
                                    ;23054  =;END   ;-------------------------------;
                                    ;23055  =110    ;BRANCH ON LOW BIT OF ADDRESS
                                    ;23056          ;110----------------------------;
                                    ;23057  W200:   CACHE_D[WORD],                  ;  WRITES LAST WORD
U 0E76, 000C,403A,01C0,3000,0000,0060 ;23058        ALU_LB,Q_ALU,RETURN60           ;  GET LENGTH IN Q
                                    ;23059          ;111----------------------------;
U 0E77, 0000,803C,7180,3000,0084,6C70 ;23060        CACHE_D[BYTE],SC_K[.FFF8]       ;  LOAD -8 FOR SHIFTING
                                    ;23061  =;END   ;-------------------------------;
U 0C70, 0D18,0000,0580,F800,0200,0ADF ;23062        VA_LA-K[.1],D_DAL.SC,J/W100     ;  WRITE TWO BYTES
                                    ;23063          ;-------------------------------;
                                    ;23064  =110    ;BRANCH ON LOW BIT OF LA
                                    ;23065          ;110----------------------------;
                                    ;23066  W300:   CACHE_D[BYTE],SC_K[.FFF8],      ;  WRITE FIRST OF 3 BYTES
U 0E7E, 0000,803C,7180,3000,0084,6C78 ;23067        J/W410
                                    ;23068          ;111----------------------------;
                                    ;23069          CACHE_D[WORD],SC_K[.FFF0],      ;  WRITE FIRST OF 3 BYTES
U 0E7F, 0000,403C,6D80,3000,0084,6C76 ;23070        J/W401
                                    ;23071  =;END   ;-------------------------------;
```

```
                                    ;23072  =010    ;BRANCH ON LOW 2 BITS OF LA
                                    ;23073          ;010----------------------------;
                                    ;23074  W400:   CACHE_D[LONG],                  ;  WRITE IT ALL
U 0E82, 000C,003A,01C0,3000,0000,0060 ;23075        ALU_LB,Q_ALU,RETURN60           ;  GET LENGTH IN Q
                                    ;23076          ;011----------------------------;
                                    ;23077          CACHE_D[WORD],SC_K[.FFF0],
U 0E83, 0000,403C,6D80,3000,0084,6C73 ;23078        J/W401                          ;  WRITE FIRST WORD
                                    ;23079          ;110----------------------------;
                                    ;23080          CACHE_D[WORD],SC_K[.FFF0],
U 0E86, 0000,403C,6D80,3000,0084,6C78 ;23081        J/W410                          ;  WRITE FIRST WORD
                                    ;23082          ;111----------------------------;
                                    ;23083          CACHE_D[BYTE],SC_K[.FFF8],
U 0E87, 0000,803C,7180,3000,0084,6C71 ;23084        J/W411                          ;  WRITE FIRST BYTE
                                    ;23085  =;END   ;----------------------------;
                                    ;23086  W411:   ALU_LA-K[.3],                   ;  ADJUST ADDRESS
U 0C71, 0D18,0000,0D80,F800,0200,0C72 ;23087        VAK7LOAD,D_DAL.SC               ;  SHIFT DATA INTO D FOR WRITING
                                    ;23088          ;----------------------------;
                                    ;23089          CACHE_D[WORD],SC_K[.FFF0],      ;  WRITE NEXT WORD,  GET SHIFT COUNT
U 0C72, 0000,403C,6D80,3000,0084,6C76 ;23090        J/W401T
                                    ;23091          ;----------------------------;
                                    ;23092  W401:   ALU_LA-K[.2],VAK/LOAD,
U 0C73, 0D18,0000,0980,F800,0200,0C74 ;23093        D_DAL.SC                        ;  LOAD ADDRESS, RIGHT ADJUST
                                    ;23094          ;----------------------------;
U 0C74, 0000,803C,7180,3000,0084,6C76 ;23095        CACHE_D[BYTE],SC_K[.FFF8]       ;  WRITE BYTE
                                    ;23096          ;----------------------------;
                                    ;23097  W4011:  ALU_LA-K[.1],VAK/LOAD,          ;  LOAD ADDRESS FOR NEXT BYTE
                                    ;23098          D_DAL.SC,                       ;  SHIFT DATA INTO D
U 0C76, 0D18,0000,0580,F800,0200,0ADF ;23099        J7W100
                                    ;23100          ;----------------------------;
                                    ;23101  W410:   ALU_LA-K[.2],VAK/LOAD,          ;  LOAD ADDRESS
                                    ;23102          D_DAL.SC,DT/LONG,               ;  RIGHT ADJUST DATA
U 0C78, 0D18,0000,0980,F800,0200,0E76 ;23103        J7W200                          ;
                                    ;23104          ;----------------------------;
```

```
                                ;23105          ;THE MULP-INSTRUCTION ENTERS THE BCD-WRITE ROUTINE HERE
                                ;23106  WRITE.MUL:
                                ;23107          ALU_Q+K[.8],                    ;  INCREMENT NIBBLE-COUNT
                                ;23108          SHF7ALU.DT,LONG,                ;  SHIFT IT LEFT TWICE
                                ;23109          SC_ALU,
                                ;23110          QK7SHF,
                                ;23111          CLK.UBCC,                       ;  CLOCK NEW CC
U 0C79, 0079,2D14,01C0,F938,0092,0E89   ;23112          LC_RC[T7],SIGNS?
                                ;23113          ;-------------------------------;
                                ;23114  =001    ;BRANCH ON Q31 AND D NE 0
                                ;23115          ;001------------------------------;
                                ;23116          ALU_0(A),RC[T7]_ALU,LONG,       ;  SAVE LAST WRITTEN DATA
U 0E89, 0003,003E,0180,F988,0000,0020   ;23117          RETURN20                        ;  ALREADY POSITIVE-END OF DST
                                ;23118          ;011------------------------------;
                                ;23119          ALU_0(A),RC[T7]_ALU,LONG,       ;  SAVE LAST WRITTEN DATA
                                ;23120          STATE_STATE.OR.K[.40],          ;  SET OVERFLOW-BIT
U 0E8B, 0003,003E,3180,F988,1404,2020   ;23121          RETURN20                        ;  ALREADY POSITIVE-END OF DST
                                ;23122          ;101------------------------------;
U 0E8D, 001F,1400,0180,F800,0082,0E95   ;23123          ALU_0-Q,SC_ALU,SC?,J/WRIO.MUL   ;  BRANCH ON NEW Q
                                ;23124          ;11T------------------------------;
U 0E8F, 001F,1400,0180,F800,0082,0E95   ;23125          ALU_0-Q,SC_ALU,SC?              ;  BRANCH ON NEW Q
                                ;23126  =;END   ;-------------------------------;
                                ;23127  =101    ;BRANCH ON SC GT 0
                                ;23128          ;101------------------------------;
                                ;23129  WRIO.MUL:
                                ;23130          Q_D,ALU_D,N_AMX.Z_TST,BYTE,     ;  CLOCK Z-BIT
                                ;23131          SC_K[ZERO],FEK/LOAD,
U 0E95, 0001,973C,19E0,F800,01B4,6B39   ;23132          BEN/STATE3-0,J/WR101            ;  WRITE WHOLE WORD
                                ;23133          ;111------------------------------;
                                ;23134          ALU_0+MASK+1,Q_ALU,
U 0E97, 0003,0010,11C0,F800,0084,8C7A   ;23135          SC_SC+K[.4]                     ;  LOAD MASK IN Q
                                ;23136  =;END   ;-------------------------------;
                                ;23137          ALU_D.ANDNOT.Q,N_AMX.Z_TST,BYTE,
                                ;23138          Q_D,D_ALU,FE_SC-K[.4],
U 0C7A, 081D,8024,11E0,F988,0134,AC7B   ;23139          RC[T7]_ALU                      ;  SAVE OVERFLOW DATA
                                ;23140          ;-------------------------------;
                                ;23141          ALU_Q-D,CLK.UBCC,
                                ;23142          FE_SC.ANDNOT.K[.4],SC_FE,
U 0C7B, 001D,3700,1180,F800,0195,4B39   ;23143          STATE3-0?,J/WR101               ;  JOIN COMMON WRITE ROUTINE
                                ;23144          ;-------------------------------;
```

```
                                ;23145  .TOC    ''      Decimal string        : FAULT PARAMETER SAVE-ROUTINES''
                                ;23146
                                ;23147  ;ROUTINE USED TO STORE SC,STATE,FE,AND LOW BYTE OF D IN R0
                                ;23148  ;THE ORDER IS   D,STATE*2,FE,SC
                                ;23149  ;HIGH ORDER BIT OF STATE IS LOST, AND 2 HIGH ORDER BITS OF FE AND SC ARE
                                ;23150  ;RESTORED AS SIGN-EXTENSIONS OF BIT 7.
                                ;23151          ;-------------------------------;
                                ;23152  SAVER0.0:
U 0C7C, 0818,0034,4980,FA00,0000,0C7D   ;23153          D_R[R0].AND.K[.FF]              ;  ENTER HERE IF D IS NOT SET UP
                                ;23154          ;-------------------------------;
                                ;23155  SAVER0: Q_K[SC],D_D.SWAP,               ;  GET SC AND LOW BYTE OF D
U 0C7D, 0B18,0038,1DC7,F800,0000,0C80   ;23156          SGN/CLR.SD+SS                   ;  CLEAR SD AND SS
                                ;23157          ;-------------------------------;
U 0C80, 0F1F,A030,01C0,F800,0000,0C81   ;23158          Q_Q.0XT[BYTE].OR.D,D_0          ;  MERGE D AND SC (IN HIGH/LOW BYTES)
                                ;23159          ;-------------------------------;
                                ;23160          EALU_STATE,
U 0C81, 0828,0038,0180,F800,1400,0C82   ;23161          D_PACK.FP.LEFT                  ;  GET STATE-REGISTER
                                ;23162          ;-------------------------------;
                                ;23163          EALU_FE,
U 0C82, 0828,0038,0180,F800,0000,6C84   ;23164          D_PACK.FP.LEFT                  ;  GET FE AND STATE IN MIDDLE BYTES
                                ;23165
                                ;23166  ; ************************************************
                                ;23167  ; * Patch no. 024, PCS 0C82 trapped to WCS 115B *
                                ;23168  ; ************************************************
                                ;23169
                                ;23170  DC.PA.24:
                                ;23171          ;-------------------------------;
U 0C84, 001D,0032,0180,FA80,0000,0020   ;23172          R[R0]_D.OR.Q,RETURN20           ;  SAVE RESULT IN R0, FINISHED
                                ;23173          ;-------------------------------;
                                ;23174
                                ;23175  RESTR0: ;ROUTINE TO RESTORE SC,STATE,FE, AND LOW BYTE OF D FROM R0
                                ;23176          ;EXPECTS THE ORDER D,STATE*2,FE,SC
                                ;23177          ;-------------------------------;
                                ;23178          ALU_R[R0],D_ALU,Q_ALU,          ;  GET THE DATA
U 0C85, 0800,003C,71C0,FA00,0084,6C86   ;23179          SC_K[.FFF8]                     ;  -8
                                ;23180          ;-------------------------------;
                                ;23181          D_DAL.SC,                       ;  SHIFT LEFT
U 0C86, 0D02,803C,0180,F800,0082,0C88   ;23182          SC_D.SXT[BYTE]                  ;  GET SC
                                ;23183          ;-------------------------------;
                                ;23184          FE_SC,                          ;  SAVE SC TEMPORARILY IN FE
U 0C88, 0002,803C,01E0,F800,0182,0C89   ;23185          SC_D.SXT[BYTE],Q_D              ;  GET FE-VALUE
                                ;23186          ;-------------------------------;
                                ;23187          D_LB,                           ;  GET DATA AGAIN
                                ;23188          QR/RIGHT2,
U 0C89, 080C,0038,0190,F800,0191,0C8A   ;23189          SC_FE,FE_SC,CLK.UBCC            ;  CLOCK FE-DATA IN EALU
                                ;23190          ;-------------------------------;
                                ;23191          STATE_Q(EXP),Q_K[SC],           ;  RESTORE STATE, DUPLICATE SC IN Q
U 0C8A, 0B19,2038,1DC0,F800,1408,6C8B   ;23192          D_D.SWAP
                                ;23193          ;-------------------------------;
                                ;23194          STATE_STATE.ANDNOT.K[.80],      ;  CLEAR INTERRUPT-BIT
                                ;23195          D_D.SXT[BYTE],                  ;  SIGN-EXTEND D
                                ;23196          R[R0]_ALU,
U 0C8B, 0802,803E,4180,FA80,1404,4110   ;23197          RETURN[110]
                                ;23198          ;-------------------------------;
```

```
                                ;23199  SGN.EXT.D:
                                ;23200          ;ROUTINE TO SIGN-EXTEND D-BYTE
                                ;23201          ;-------------------------------;
                                ;23202          Q_D.SXT[BYTE],                  ;  SIGN EXTEND D
U 0C8C, 0002,803E,01C0,F800,0000,0040   ;23203          RETURN40                        ;  RETURN40
                                ;23204          ;-------------------------------;
                                ;23205
                                ;23206
                                ;23207
                                ;23208          ;ROUTINE WHICH SAVES PC-DELTA,LOW BYTE OF R2, LOW WORD OF D , ALL IN R2
                                ;23209          ;THE ORDER IS R2.BYTE,D.WORD(REVERSED),PC.DELTA
                                ;23210          ;-------------------------------;
U 0C8D, 0819,2034,4980,F800,0000,0C90   ;23211  SAVER2.0:       D_Q.AND.K[.FF]
                                ;23212          ;-------------------------------;
                                ;23213  SAVER2: D_D.SWAP,                       ;  FLIP D AROUND
U 0C90, 0B00,003C,71C0,FA10,0084,6C91   ;23214          Q_R[R2],SC_K[.FFF8]             ;  GET R2-BYTE, AND SHIFT VALUE
                                ;23215          ;-------------------------------;
                                ;23216  SAVER2.1:
                                ;23217          D_DAL.SC,                       ;  SHIFT D AND Q TOGETHER
U 0C91, 0D14,0038,01C0,F800,0000,0810   ;23218          Q_PC                            ;  GET PC FOR CHUCK'S ROUTINE
                                ;23219          ;0*-------------------------------;
                                ;23220  =0*     R[R2]_D,LONG,                   ;  STORE UPPER 3 BYTES
U 0810, 0001,003D,0180,FA90,0000,0EB8   ;23221          CALL,J/BAKUP.PC                 ;  ROUTINE TO BACK UP PC
                                ;23222          ;1*-------------------------------;
                                ;23223          R[R2]_D,BYTE,                   ;  LOAD LAST BYTE
U 0812, 0001,803E,0180,FA90,0000,0110   ;23224          RETURN[110]                     ;  All done.
                                ;23225  =;END   ;-------------------------------;
                                ;23226
                                ;23227  RESTR2: ;ROUTINE TO RESTORE D,R2, AND PC FROM R2
                                ;23228          ;EXPECTS THE ORDER TO BE R2.BYTE,D.WORD(REVERSED),PC.DELTA
                                ;23229
U 0C92, 0800,003C,7180,FA10,0084,609C   ;23230          D_R[R2],SC_K[.FFF8]             ;  GET DATA TO BE RESTORED,
                                ;23231          ;-------------------------------;
                                ;23232  =0******
                                ;23233          PC&VA_D.OXT[BYTE]+PC,           ;  RESTORE PC
                                ;23234          D_D.SWAP,                       ;  GET D IN RIGHT ORDER
U 009C, 0B17,8015,0180,F801,0200,0C8C   ;23235          CALL,J/SGN.EXT.D
                                ;23236          ;1******-------------------------;
                                ;23237          ALU_Q,                          ;  GET LOW BYTE
                                ;23238          R[R2]_ALU,LONG,                 ;  STORE IT IN R2
                                ;23239          D_DAL.SC,Q_ALU,                 ;  SHIFT D INTO PLACE
U 00DC, 0D01,203E,01C0,FA90,0000,0020   ;23240          RETURN20                        ;  ALL DONE
                                ;23241  =;END   ;-------------------------------;
```

```
                                        ;23242  33:
                                        ;23243          ;ROUTINE SAVE CONTEXT DURING FAULTS AND INTERRUPS FOR
                                        ;23244          ;THE INSTUCTIONS: MOVP,CMPP,CVTSP,CVTPN,ASHP,CVTLP
                                        ;23245          ;EXPECTS ADDRESSES IN T0 AND T1
                                        ;23246          ;EXPECTS LONGWORD IN RC[T5]
                                        ;23247          ;EXPECTS LENGTHS IN RC0 AND RC1
                                        ;23248
                                        ;23249          ;SAVES T0 IN R1
                                        ;23250          ;SAVES T1 IN R3
                                        ;23251          ;SAVES RC5 IN R0
                                        ;23252          ;SAVES RC0,RC1, AND PC-DELTA IN R2
                                        ;23253
                                        ;23254  SAVE.BCD:
                                        ;23255          ;-------------------------------;
U 0033, 0810,0038,C1F0,2D00,0000,0C94   ;23256          Q_ID[T0],ALU_RC[T0],D_ALU       ;  D GETS 1. LENGTH, Q GETS 1. ADDRESS
                                        ;23257          ;-------------------------------;
                                        ;23258          ALU_Q,LC_RC[T5]&R1_ALU,         ;  SAVE 1. ADDRESS IN R1
U 0C94, 0001,203C,01F8,FBA8,0084,6C95   ;23259          SC_R[.8],Q_0                    ;  NEED TO SHIFT
                                        ;23260          ;-------------------------------;
                                        ;23261          ALU_LC,R[R0]_ALU,               ;  SAVE RC5 IN R0
                                        ;23262          D_DAL.SC,                       ;  SHIFT LENGTH
U 0C95, 0D10,0038,C5F0,2E80,0000,0C96   ;23263          Q_ID[T1]                        ;  GET 2. ADDRESS
                                        ;23264          ;-------------------------------;
U 0C96, 0001,203C,0180,FA98,0000,0C98   ;23265          R[R3]_Q                         ;  SAVE DST-ADDRESS IN R3
                                        ;23266          ;-------------------------------;
U 0C98, 0010,0038,01C0,F908,0000,0010   ;23267          Q_RC[T1]                        ;  GET DST-LENGTH
                                        ;23268          ;-------------------------------;
                                        ;23269  =0***1****
                                        ;23270          ALU_Q.OXT[BYTE].OR.D,D_ALU,     ;  'OR'' THE LENGTHS TOGETHER
U 0010, 081F,A031,0180,F800,0000,0C90   ;23271          CAL[.J/SAVER2
                                        ;23272          ;-------------------------------;
U 0110, 0000,163C,0180,F800,0000,0B47   ;23273  MV.SV2: BEN/STATE7-4                    ;  TEST INTERRUPT-BIT
                                        ;23274  =0111   ;0111-------------------------;
                                        ;23275  CHUCK.FPD.RTN:
U 0B47, 0000,003E,0180,F800,0000,0002   ;23276          RETURN2                         ;  CHUCK'S MEMORY-FAULT-ROUTINE
                                        ;23277          ;1111-------------------------;
U 0B4F, 0000,003C,0180,F800,0000,04FA   ;23278          J/INT.I                         ;  TONY'S INTERRUPT-ROUTINE
                                        ;23279  =;END   ;-------------------------------;
                                        ;23280
```

```
                                        ;23281  RESTORE.BCD:    ;SUBROUTINE TO RESTORE CONTEXT FOR MOVP,CMPP,ETC.
                                        ;23282          ;RESTORES TO FROM R1
                                        ;23283          ;T1 FROM R3
                                        ;23284          ;RC0 FROM R2
                                        ;23285          ;RC1 FROM R2
                                        ;23286          ;RC5 FROM R0
                                        ;23287          ;PC FROM R2
                                        ;23288          ;R15 FROM 0-RC1-1
                                        ;23289          ;-------------------------------;
U 0C99, 0800,003C,0180,FA08,0000,029E   ;23290          D_R[R1]
                                        ;23291          ;-------------------------------;
                                        ;23292  =0*****
U 029E, 0000,003D,C180,3C00,0000,0C92   ;23293          ID[T0]_D,CALL,J/RESTR2
                                        ;23294          ;-------------------------------;
                                        ;23295          ALU_D.SXT[BYTE],RC[T1]_ALU,     ;  RESTORE SRC-LENGTH 1
U 02BE, 0D02,803C,01C1,F988,0000,0C9A   ;23296          D_DAL.SC,Q_ALU,SGN/LOAD.SS      ;  SC HAS -8
                                        ;23297          ;-------------------------------;
U 0C9A, 0002,803C,0180,F980,0000,0C9B   ;23298          ALU_D.SXT[BYTE],RC[T0]_ALU      ;  RESTORE 2. LENGTH
                                        ;23299          ;-------------------------------;
U 0C9B, 0800,003C,0180,FA18,0000,0C9C   ;23300          D_R[R3]
                                        ;23301          ;-------------------------------;
                                        ;23302          ALU_0-Q-1,R[R15]_ALU,           ;  NEGATIVE LENGTH
U 0C9C, 001F,0008,C580,3EF8,0000,0C9D   ;23303          ID[T1]_D                        ;  SAVE DST-ADDRESS
                                        ;23304          ;-------------------------------;
                                        ;23305          STATE_K[ZERO],                  ;  CLEAR STATE-REGISTER
U 0C9D, 0000,003C,1980,FA00,1404,6C9E   ;23306          LAB_R[R0]
                                        ;23307          ;-------------------------------;
                                        ;23308          ALU_LA,RC[T5]_ALU,              ;  RETRIEVE RC 5 FROM R0
U 0C9E, 0000,003E,0180,F9A8,0000,0110   ;23309          RETURN[110]                     ;
                                        ;23310          ;-------------------------------;
```

```
                                ;23311  13:
                                ;23312  ADS.MEMORY.FAULT:
                                ;23313          ;ROUTINE TO PACK UP ADD/SUB-INSTRUCTION.
                                ;23314          ;-------------------------------;
                                ;23315          ALU_R[R15].AND.K[.FF],          ;  GET R15
                                ;23316          D_ALU,
U 0013, 0818,0035,4980,FA78,0000,0C90   ;23317          CALL,J/SAVER2           ;  SAVE R15,R2,AND PC-DELTA IN R2
                                ;23318          ;-------------------------------;
                                ;23319  113:    ALU_R[R0].AND.K[.FF],           ;  GET R0
                                ;23320          D_ALU,
U 0113, 0818,0035,4980,FA00,0000,0C7D   ;23321          CALL,J/SAVER0           ;  SAVE STATE AND R0 IN R0
                                ;23322          ;-------------------------------;
                                ;23323  133:
U 0153, 0000,003C,0180,F800,0000,0110   ;23324          J/MV.SV2                ;  TEST FOR INTERRUPT OR MEMORY FAULT
                                ;23325          ;-------------------------------;
                                ;23326
                                ;23327  43:
                                ;23328  ADDSUB.RESTART:
                                ;23329          ;ROUTINE TO RESUME ADD/SUB-INSTRUCTION AFTER A FAULT
                                ;23330          ;ENTER HERE FROM IRD
                                ;23331          ;-------------------------------;
                                ;23332          ALU_0+Q,SET.CC(LONG),           ;  CHANGE THIS EVENTUALLY (O.K. NOW)
U 0043, 001F,0015,0180,F800,0070,0C85   ;23333          CALL,J/RESTR0           ;  RESTORE R0 AND STATE FROM R0
                                ;23334          ;-------------------------------;
                                ;23335  153:    R[R0]_D,LONG,                   ;  RESTORE R0 SGN-EXTENDED
U 0153, 0001,003D,0180,FA80,0000,0C92   ;23336          CALL,J/RESTR2           ;  GET PC,R2 AND R15 FROM R2
                                ;23337          ;-------------------------------;
U 0173, 0802,973C,01F8,F800,0000,0B5E   ;23338  173:    Q_0,D_D.SXT[BYTE],STATE3-0?     ;  TEST CARRY-BIT OF STATE
                                ;23339          ;-------------------------------;
                                ;23340  =1110   ;BRANCH ON CARRY-BIT OF STATE
                                ;23341          ;1110---------------------------;
                                ;23342  ADSU.RE2:       R[R15]_D,LONG,          ;  RESTORE DEST-LENGTH
U 0B5E, 0001,003C,0180,FAF8,0000,0CA0   ;23343          J/ADSU.RE3
                                ;23344          ;1111---------------------------;
                                ;23345          ALU_Q+K[.FF]+1,SET.CC(BYTE),    ;  SET PSL C-BIT
U 0B5F, 0019,A010,4980,F800,0070,0B5E   ;23346          J/ADSU.RE2
                                ;23347  =;END   ;-------------------------------;
                                ;23348  ADSU.RE3:
                                ;23349          ALU_K[.9],D_ALU.LEFT,           ;  GENERATE ADDRESS 13
                                ;23350          ST/MUL-,                        ;  FOR RESTART ADDRESS
U 0CA0, 0838,1638,DB80,F800,0000,0E9E   ;23351          STATE7-4?,J/ADSU.RE4    ;  CHECK TO SEE IF WE WERE NEGATING
                                ;23352          ;-------------------------------;
                                ;23353  =110    ;BRANCH ON NEGATE-BIT OF STATE
                                ;23354          ;110----------------------------;
U 0E9E, 0000,003C,B580,3C00,0000,02AB   ;23355  ADSU.RE4:       ID[FPDA]_D,J/ADS.EN     ;  REENTER MAIN LOOP
                                ;23356          ;111----------------------------;
U 0E9F, 0000,003C,B580,3C00,0000,0C47   ;23357          ID[FPDA]_D,J/NEGATE     ;  NEGATE STRING
                                ;23358  =;END   ;-------------------------------;
```

```
                                ;23359  4B:
                                ;23360  MOV.RES:        ;ROUTINE TO RESTART MOVP,CVTTP, AND CVTSP AFTER A FAULT
                                ;23361          ;-------------------------------;
                                ;23362          ALU_0(A),N&Z_ALU,               ;  CLEAR N, SET Z-BIT
                                ;23363          RC[T7]_ALU,                     ;  CLEAR OVERFLOW-REGISTER
U 004B, 0003,003D,0180,F9B8,0060,0C99   ;23364          CALL,J7RESTORE.BCD
                                ;23365          ;-------------------------------;
                                ;23366  15B:    SC_RC[TO],CLK.UBCC,BYTE,        ;  LOAD SRC-LENGTH, CLOCK IT FOR CVTTP
U 015B, 0010,8938,0180,F900,0092,0A28   ;23367          IR2-1?                          ;  TEST OPCODE
                                ;23368          ;-------------------------------;
                                ;23369  =00     ;BRANCH ON IR<2-1>
                                ;23370          ;00-----------------------------;  THIS IS PART OF CVTSP-ROUTINE
                                ;23371  S2P.1:  STATE_K[7ERO],                  ;  CLEAR STATE-REGISTER
U 0A28, 0000,003C,19B0,F800,1404,6BB8   ;23372          OK/RIGHT,1/S2P.10               ;  CVTSP, IR<2-0>=001
                                ;23373          ;10-----------------------------;
U 0A2A, 0000,003C,79B0,F800,0084,62F0   ;23374  =10     SC_K[.30],OK/RIGHT,J/MVP.I0     ;  MOVP,  IR<2-0>=100
                                ;23375          ;11-----------------------------;
                                ;23376          ALU_0-LC,RC[T6]_ALU,            ;  CVTTP, IR<2-0>=110
U 0A2B, 0013,0000,0580,F9B0,0084,ABB2   ;23377          SC_SC-K[.1],J/T2P.I1
                                ;23378  =;END   ;-------------------------------;
                                ;23379  4C:
                                ;23380  L2P.RES:        ;ROUTINE TO RESTART CVTLP,CVTPT,CVTPS AFTER A FAULT
                                ;23381          ;-------------------------------;
                                ;23382          ALU_0(A),N&Z_ALU.V&C_0,         ;  SET Z-BIT, CLEAR N,V,C
                                ;23383          RC[T7]_ALU,                     ;  CLEAR OVERFLOW-REGISTER
U 004C, 0003,003D,0180,F9B8,0050,0C99   ;23384          CALL,J7RESTORE.BCD
                                ;23385          ;-------------------------------;
U 015C, 0010,1B38,0180,F900,0082,0B6D   ;23386  15C:    SC_RC[TO],IR0?                  ;  LOAD SRC-LENGTH IN SC
                                ;23387          ;-------------------------------;
                                ;23388  =1101   ;BRANCH ON IR0
                                ;23389          ;1101---------------------------;
U 0B6D, 0000,093C,C1F0,2C00,0000,0A32   ;23390          Q_ID[TO],IR2-1?,J/P2N.RES       ;  CVTPT,IR<2-0>=100 CVTPS,IR<2-0>=000
                                ;23391
                                ;23392  ; ************************************************
                                ;23393  ; * Patch no. 027, PCS 0B6D trapped to WCS 115E *
                                ;23394  ; ************************************************
                                ;23395
                                ;23396          ;1111---------------------------;
U 0B6F, 0841,203C,C1F0,2C00,0000,0CA1   ;23397          ALU_Q,D_ALU.RIGHT,Q_ID[TO]      ;  CVTLP, IR<2-0>=001
                                ;23398  =;END   ;-------------------------------;
U 0CA1, 0C00,003C,01E0,F800,0000,0072   ;23399          Q_D,D_Q,J/L2P00
                                ;23400          ;-------------------------------;
                                ;23401  =10     ;BRANCH ON IR<1>
                                ;23402          ;10-----------------------------;
                                ;23403  P2N.RES:
U 0A32, 0011,2014,0180,FAF8,0000,0CA2   ;23404          ALU_Q+LC,R[R15]_ALU,J/P2N.RES.1
                                ;23405          ;11-----------------------------;
U 0A33, 0000,003C,0180,F800,0000,00F0   ;23406          J/P2L00                         ;  CVTPL, IR<2-0>=110
                                ;23407  =;END   ;-------------------------------;
                                ;23408  P2N.RES.1:
U 0CA2, 0010,0038,05C0,F908,0084,ACA4   ;23409          Q_RC[T1],SC_SC-K[.1]            ;  GET DST-LENGTH, ADJUST SRC-LENGTH
                                ;23410          ;-------------------------------;
                                ;23411          STATE_K[.1],                    ;  INITIALIZE STATE-REGISTER
U 0CA4, 001F,0010,05C0,F800,1404,60F7   ;23412          ALU_0+Q+1,Q_ALU,J/P2T.I00       ;
```

```
                                    ;23413  0C5:
                                    ;23414  RESTART.ASHP:
                                    ;23415          ;ROUTINE TO RESTART ASHP-INSTRUCTION AFTER FAULT.
                                    ;23416          ;-------------------------------;
U 00C5, 0800,003C,01C0,FA00,0082,0CA5 ;23417        D_R[R0],SC_ALU,Q_ALU            ;  GET SHIFT-COUNT AND ROUNDING-OPERAND
                                    ;23418          ;-------------------------------;
                                    ;23419          D_D.SWAP,ALU_0(A),N&Z_ALU.V&C_0,;  CLEAR N-BIT, SET Z-BIT
                                    ;23420          SGN/CLR.SD+SS,                  ;  CLEAR SS
                                    ;23421          RC[T2]_ALU,                     ;  CLEAR RC2 FOR TEMPORARY STORAGE
U 0CA5, 0B03,003C,0187,F990,0050,00Fo ;23422        J/ASHP.REEN.0
                                    ;23423          ;-------------------------------;
                                    ;23424  45:
                                    ;23425  CMP.RESTART:
                                    ;23426          ;ROUTINE TO RESTART CMPP-INSTRUCTION
                                    ;23427          ;REENTERS MAIN-ROUTINE AT CMP411
                                    ;23428          ;WITH D=DST-ADDRESS,Q=DST-LENGTH,
                                    ;23429          ;FE=SRC-LENGTH+1
                                    ;23430          ;-------------------------------;
                                    ;23431          ALU_0(A),N&Z_ALU.V&C_0,         ;  SET Z-BIT, CLEAR V AND C
U 0045, 0003,003D,0180,F800,0050,0C99 ;23432        CAL[,J/RESTORE.BCD              ;  RESTORE ID[T0],ID[T1],RC0,RC1
                                    ;23433
                                    ;23434          ;-------------------------------;
                                    ;23435  155:    SC_RC[T0],                      ;  LOAD SRC1-LENGTH IN SC
U 0155, 0C10,0038,C1F0,2D00,0082,0CA6 ;23436        D_Q,Q_ID[T0]                    ;  RETRIEVE SRC1-ADDRESS
                                    ;23437
                                    ;23438          ;-------------------------------;
                                    ;23439          FE_SC+1,                        ;  INCREMENT SRC1-LENGTH
                                    ;23440          ALU_Q[INST.DEP]D,               ;  SUBTRACT 1 FROM Q
                                    ;23441          R[RT5]_ALU,                     ;  STORE SRC1-ADDRESS IN R15
U 0CA6, 001D,200C,C5F0,2EF8,0100,CCA8 ;23442        Q_ID[TT]                        ;  GET SRC2-ADDRESS
                                    ;23443
                                    ;23444          ;-------------------------------;
                                    ;23445          D_Q,ALU_D-K[.1],Q_ALU,          ;ADJUST ADDRESS
U 0CA8, 0C19,0000,05C0,F800,0000,0B34 ;23446        J7CMP41T                        ;  REJOIN MAIN ROUTINE
                                    ;23447
                                    ;23448          ;-------------------------------;
```

```
                                        ;23449  MULT.SAVE:
                                        ;23450          ;ROUTINE TO SAVE CONTEXT DURING FAULTS IN MULTIPLY.
                                        ;23451          ;LEAVES R3 ALONE (HAS MULTIPLICAND-ADDRESS)
                                        ;23452          ;SAVE MULTIPLIER ADDRESS IN R1
                                        ;23453          ;LEAVE R5 ALONE, HAS DST-ADDRESS
                                        ;23454          ;ALSO SAVES PC-DELTA,R0(MULTIPLICAND-BYTE),
                                        ;23455          ;R4 (CURRENT DST-LENGTH),R2(MULTIPLICAND-LENGTH)
                                        ;23456          ;T0 (CURRENT MULTIPLICAND-DIGIT),
                                        ;23457          ;T6 (MULTIPLIER ADDRESS)
                                        ;23458          ;RC6 (MULTIPLIER-COUNT)
                                        ;23459          ;R1 (PRODUCT-LENGTH)
                                        ;23460          ;T7 (MULTIPLIER-LENGTH),STATE,
                                        ;23461          ;T3-R5(OFFSET FROM ABSOLUTE DST-ADDRESS)
                                        ;23462          ;T4 ORIGINAL DST-LENGTH
                                        ;23463
                                        ;23464          ;THIS ROUTINE SAVES T4,R1,R0,AND STATE IN R0
                                        ;23465          ;RC6,T7,R2, AND PC-DELTA ARE SAVED IN R2
                                        ;23466          ;R4,T3-R5,T0 ARE SAVED IN R4
                                        ;23467
                                        ;23468          ;-------------------------------;
                                        ;23469          R[R1]_Q,                        ;  SAVE MULTIPLIER-ADDRESS IN R1
U 0CA9, 0001,203C,D1F0,2E88,0100,048D   ;23470          FF_SC,Q_ID[T4]                  ;  GET DST-LENGTH
                                        ;23471          ;-------------------------------;
                                        ;23472  =0*****
                                        ;23473          ALU_Q,SC_ALU,                   ;  SAVE DST-LENGTH IN R0 (VIA SC)
U 048D, 0001,203D,0180,F800,0082,0C7C   ;23474          CAL[,J/SAVER0.0                 ;  SAVE FE,SC,STATE IN R0
                                        ;23475          ;-------------------------------;
U 04AD, 0810,0038,DDF0,2D30,00C0,0CAA   ;23476          D_RC[T6],Q_ID[T7]               ;  GET MULTIPLIER-LENGTH
                                        ;23477  =;END   ;-------------------------------;
U 0CAA, 0B00,003C,6980,F800,0084,6059   ;23478          SC_K[.FFE8],D_D.SWAP            ;  PACK IT INTO D
                                        ;23479          ;-------------------------------;
                                        ;23480  =0***1****
U 0059, 0D00,003D,0180,F800,0000,0C90   ;23481          D_DAL.SC,CALL,J/SAVER2          ;  SHIFT Q INTO D AS WELL
                                        ;23482          ;-------------------------------;
U 0159, 0000,003C,CDF0,2E28,0000,0CAB   ;23483          Q_ID[T3],LAB_R[R5]              ;  GET ABSOLUTE AND RELATIVE
                                        ;23484                                          ;  PRODUCT-ADDRESSES
                                        ;23485  =;END   ;-------------------------------;
                                        ;23486          LA_RA[R4],                      ;  SAVE IT ALL IN R4
                                        ;23487          ALU_Q-LB,D_ALU,Q_ID[T0],
U 0CAB, 080D,2000,C1F0,2CA0,0000,0CB1   ;23488          J/SAVER4                        ;  DON'T RETURN FROM THIS ROUTINE
                                        ;23489          ;-------------------------------;
```

```
                                ;23490  MUL.RES.1:
                                ;23491          ;ROUTINE TO RESTORE CONTEXT FOR MULP-INSTRUCTION.
                                ;23492          ID[T3]_D,                       ;  RESTORE ABSOLUTE PRODUCT-ADDRESS
U 0CAC, 0803,A03C,CD80,3C00,0000,04C0   ;23493          ALU_Q.0XT[BYTE],D_ALU           ;  GET MULTIPLICAND DIGIT
                                ;23494          ;-------------------------------;
                                ;23495  =0*****
U 04C0, 0000,003D,C180,3C00,0000,0C92   ;23496          ID[T0]_D,CALL,J/RESTR2          ;  ID[T0] GETS MULTIPLICAND-DIGIT
                                ;23497          ;-------------------------------;
                                ;23498          ALU_D.AND.K[.FF],RC[T6]_ALU,    ;  GET RC-COUNT
U 04E0, 0D19,0034,4980,F9B0,0000,0CB1   ;23499          D_DAL.SC                        ;  SC HAS -8 FROM RESTR2
                                ;23500  =;END   ;-------------------------------;
                                ;23501  =0***1****
                                ;23502          ID[T7]_D,                       ;  MULTIPLIER LENGTH
U 00B1, 0000,003D,DD80,3C00,0000,0C85   ;23503          CALL,J/RESTR0                   ;  GET CONTENTS OF R0
                                ;23504          ;-------------------------------;
U 01B1, 0818,0038,1D80,F800,0081,0CAD   ;23505          D_K[SC],SC_FE                   ;  PART OF DATA IS IN SC AND FE
                                ;23506  =;END   ;-------------------------------;
U 0CAD, 0018,0038,1DC0,F800,0000,0CB0   ;23507          Q_K[SC]
                                ;23508          ;-------------------------------;
U 0CB0, 0002,A03C,D1C0,3C00,0000,072E   ;23509          ALU_Q.SXT[BYTE],Q_ALU,ID[T4]_D  ;  T4 GETS PRODUCT LENGTH
                                ;23510  =;END   ;-------------------------------;
                                ;23511  =0****  R[R1]_Q,Q_ID[T7],              ;  GET MULTIPLIER-LENGTH
U 072E, 0001,203D,DDF0,2E88,0000,0990   ;23512          CALL,J/LOAD.MULTIPLIER          ;  SUBROUTINE TO RELOAD M'PLIER IN RC
                                ;23513          ;-------------------------------;
U 073E, 0810,1738,0180,F930,0000,0B76   ;23514          D_RC[T6],STATE3-0?              ;  DETERMINE WHERE TO RESTART
                                ;23515  =;END   ;-------------------------------;
                                ;23516  =0110   ;BRANCH ON 1.READ AND HI/LO-BITS OF STATE
                                ;23517          ;0110-----------------------------;
U 0B76, 0F00,173C,0180,F800,0000,042A   ;23518          D_0,STATE3-0?,J/MULT.RE1        ;  TEST READ/WRITE-BIT
                                ;23519          ;0111-----------------------------;
U 0B77, 0F00,003C,D1F0,2C00,0000,033A   ;23520          D_0,Q_ID[T4],J/MUL.F2           ;  FAULTED DURING SIGN-CHANGE
                                ;23521          ;1110-----------------------------;
                                ;23522          ALU_Q+K[.1],                    ;  ADJUST LENGTH
                                ;23523          D_ALU,CLK.UBCC,BYTE,            ;  FAULTED WHILE READING M'PLICAND
U 0B7E, 0819,A014,0580,F800,0010,0BFD   ;23524          J/MULM00                        ;  READ ANOTHER BYTE OF MULTIPLICAND
                                ;23525          ;1111-----------------------------;
                                ;23526          ALU_D.0XT[BYTE]-K[.1],          ;  FAULTED WHILE UPDATING PRODUCT
                                ;23527          RC[T6]_ALU,                     ;  OR TOOK AN INTERRUPT
U 0B7F, 001B,8000,0580,F9B0,0000,0BF9   ;23528          J/MULM                          ;  MULTIPLY RC-REGISTERS
                                ;23529  =;END   ;-------------------------------;
                                ;23530  =01*    ;BRANCH ON 1.WRITE BIT OF STATE
                                ;23531          ;01*------------------------------;
                                ;23532  MULT.RE1:
                                ;23533          ALU_Q+Q+1,Q_ALU,                ;  IT WAS NOT SIGN-BYTE THAT FAULTED
U 042A, 001F,0010,01C0,F800,0000,0828   ;23534          J/MULR.1
                                ;23535          ;11*------------------------------;
U 042E, 0000,003C,0180,F800,0000,0566   ;23536          J/MULSGN0                       ;  FAULT IN SIGN-BYTE
                                ;23537  =;END   ;-------------------------------;
```

```
                                        ;23538  SAVER4: ;ROUTINE TO SAVE LA,D,AND Q IN R4
                                        ;23539          ;NOT A SUBROUTINE, JUMPS DIRECTLY TO MEMORY-FAULT OR INTERRUPT
                                        ;23540          ;-------------------------------;
                                        ;23541          D_D.SWAP,
U 0CB1, 0B00,003C,7180,F800,0084,6CB2   ;23542          SC_K[.FFF8]                     ;  MANIPULATE DATA BY
                                        ;23543          ;-------------------------------;
                                        ;23544          D_DAL.SC,                       ;  BY SHIFTING
U 0CB2, 0D18,0034,49C0,FA20,0000,0CB3   ;23545          A[U_R[R4].AND.K[.FF],Q_ALU
                                        ;23546          ;-------------------------------;
U 0CB3, 0819,0024,4980,F800,0000,0CB4   ;23547          ALU_D.ANDNOT.K[.FF],D_ALU       ;  AND MASKING
                                        ;23548          ;-------------------------------;
                                        ;23549          ALU_D.OR.Q,R[R4]_ALU,           ;  AND MERGING
U 0CB4, 001D,0030,0180,FAA0,0000,0110   ;23550          J/MV.SV2                        ;  UNTIL IT ALL FITS IN R4
                                        ;23551          ;-------------------------------;
                                        ;23552
                                        ;23553  RESTR4: ;ROUTINE TO RESTORE D,Q,AND R4 FROM R4
                                        ;23554          ;-------------------------------;
U 0CB5, 0800,003C,01C0,FA20,0000,0CB8   ;23555          D_R[R4],Q_ALU                   ;  AND NOW UNRAVEL IT AGAIN
                                        ;23556          ;-------------------------------;
                                        ;23557          D_D.SWAP,ALU_Q.SXT[BYTE],Q_ALU, ;  BY SWAPPING
U 0CB8, 0B02,A03C,71C0,FA28,0084,6CB9   ;23558          SC_K[.FFF8],[AB_R[R5]           ;  AND SHIFTING
                                        ;23559          ;-------------------------------;
                                        ;23560          ALU_Q,R[R4]_ALU,Q_D,            ;  UNTIL IT IS BACK
U 0CB9, 0D01,203E,01E0,FAA0,0000,0001   ;23561          D_DAL.SC,RETURN1                ;  WHERE IT CAME FROM
                                        ;23562          ;-------------------------------;
```

```
                                    ;23563  DIV.SAVE:
                                    ;23564          ;ROUTINE TO SAVE CONTEXT FOR DIVIDE PACKED BCD-STRINGS
                                    ;23565          ;EXPECTS:
                                    ;23566          ;ID[T6] HAS LOW DIVISOR-ADDRESS
                                    ;23567          ;ID[T7] HAS DIVISOR-LENGTH/2(CONSTANT)
                                    ;23568          ;ID[T9] HAS ORIGINAL QUOTIENT-LENGTH
                                    ;23569          ;R2 HAS DIVIDEND-LENGTH.OR.K[.1](INITIALLY)
                                    ;23570          ;R3 HAS DIVIDEND-ADDRESS
                                    ;23571          ;R4 HAS QUOTIENT-LENGTH(CURRENT)
                                    ;23572          ;R5 HAS QUOTIENT-ADDRESS
                                    ;23573          ;RC5 HAS CURRENT DIGIT OF QUOTIENT
                                    ;23574
                                    ;23575          ;SAVE T6 IN R1
                                    ;23576          ;SAVE RC5,STATE,D  IN R0
                                    ;23577          ;SAVE T7,R2 AND PC-DELTA IN R2
                                    ;23578          ;SAVE R4 AND T9 IN R4
                                    ;23579          ;-------------------------------;
U 0CBA, 0010,0038,D9F0,2D28,0082,0012 ;23580        Q_ID[T6],SC_RC[T5]              ;  GET ADDRESS AND DATA
                                    ;23581          ;-------------------------------;
                                    ;23582  =0**01****
U 0012, 0001,203D,0180,FA88,0000,0C7D ;23583        R[R1]_Q,CALL,J/SAVER0           ;  SAVE RC5, STATE, AND D IN R0
                                    ;23584          ;-------------------------------;
                                    ;23585  =0**11****
                                    ;23586          Q_ID[T7],D_R[R2],               ;  SAVE T7,R2 AND PC-DELTA IN R2
U 0032, 0800,003D,DDF0,2E10,0000,0C8D ;23587        CALL,J/SAVER2.0
                                    ;23588          ;-------------------------------;
                                    ;23589  =1**11****
                                    ;23590          LA_RA[R4],Q_ID[T9],             ;  SAVE R4 AND T9 IN R4
U 0132, 0000,003C,E5F0,2CA0,0000,0CB1 ;23591        J/SAVER4                        ;  DO NOT RETURN HERE
                                    ;23592  =;END    ;-------------------------------;
```

```
                                ;23593  47:
                                ;23594  MULP.DIVP.RESTORE:
                                ;23595          ;ROUTINE TO RESUME DECIMAL MULTIPLYING AND DIVIDING  AFTER A FAULT.
                                ;23596          ;-------------------------------;
U 0047, 0800,003C,0180,FA08,0000,084C   ;23597          D_R[R1]                         ;  RESTORE THE REGISTERS WHICH ARE
                                ;23598          ;-------------------------------;
U 084C, 0000,003D,D980,3C00,0000,0CB5   ;23599  =0      ID[T6]_D,CALL,J/RESTR4          ;  COMMON TO THE TWO INSTRUCTIONS
                                ;23600          ;-------------------------------;
                                ;23601          ALU_Q.SXT[BYTE],D_ALU,Q_D,
U 084D, 0802,A93C,01E0,F800,0000,0056   ;23602          IR2=1?                          ;  TEST OP-CODE TO TAKE SEPARATE PATHS
                                ;23603  =;END   ;-------------------------------;
                                ;23604  =0**01**10
                                ;23605          ;BRANCH ON BIT 1 OF OP-CODE
                                ;23606          ;-------------------------------;
                                ;23607          ALU_Q.SXT[BYTE]+LB,
U 0056, 080E,A014,01E0,F800,0000,0CAC   ;23608          Q_D,D_ALU,J/MUL.RES.1           ;  MULP-RESTORE
                                ;23609          ;-------------------------------;
                                ;23610  DIVP.RESTORE:
U 0057, 0000,003D,E580,3C00,0000,0C92   ;23611          ID[T9]_D,CALL,J/RESTR2          ;  RESTORE QUOTIENT LENGTH
                                ;23612          ;-------------------------------;
                                ;23613  =0**11**11
U 0077, 0000,003D,DD80,3C00,0000,0C85   ;23614          ID[T7]_D,CALL,J/RESTR0          ;  RESTORE DIVISOR LENGTH
                                ;23615          ;-------------------------------;
                                ;23616  =1**11**11
U 0177, 0019,0034,6180,FAF8,0000,0742   ;23617          ALU_D.AND.K[.F],R[R15]_ALU      ;  RESTORE R15 WITH LEADING DIGIT
                                ;23618  =;END   ;-------------------------------;
                                ;23619  =0****
                                ;23620          ALU_K[SC],RC[T5]_ALU,D_ALU,     ;  RESTORE QUOTIENT BYTE
U 0742, 0818,0039,1D80,F9A8,0000,09C0   ;23621          CAL[,J/DIVDR                    ;  RESTORE DIVISOR, LOAD FPDA
                                ;23622          ;-------------------------------;
                                ;23623          STATE_STATE.ANDNOT.K[.1],       ;  CLEAR LOW BIT (FOR CONSTRAINTS)
U 0752, 0000,163C,0580,FA70,1604,4EA4   ;23624          VA_R[SP],STATE7-4?              ;  PREPARE TO READ DIVIDEND FROM STACK
```

```
B  1  Character string      : SKPC, LOCC
C  1  Character string      : SKPC, LOCC
D  1  Character string      : SKPC, LOCC
E  1  Character string      : SKPC/LOCC LONGWORD OPERATIONS
F  1  Character string      : SKPC/LOCC LONGWORD OPERATIONS
G  1  Character string      : SKPC/LOCC LONGWORD OPERATIONS
H  1  Character string      : SKPC/LOCC LONGWORD OPERATIONS
I  1  Character string      : SKPC/LOCC LONGWORD OPERATIONS
J  1  Character string      : SKPC - DETERMINE WHICH BYTE
K  1  Character string      : SKPC/LOCC FPD + RESTART
L  1  Character string      : SPANC, SCANC
M  1  Character string      : SPANC, SCANC
N  1  Character string      : SPANC, SCANC
B  2  Character string      : SPANC/SCANC RESTART
C  2  Character string      : CMPC3, CMPC5
D  2  Character string      : CMPC3, CMPC5
E  2  Character string      : CMPC3, CMPC5
F  2  Character string      : CMPC3, CMPC5
G  2  Character string      : CMPC3, CMPC5
H  2  Character string      : CMPC3, CMPC5
I  2  Character string      : CMPC3, CMPC5
J  2  Character string      : CMPC3, CMPC5
K  2  Character string      : CMPC3, CMPC5
L  2  Character string      : CMPC3, CMPC5
M  2  Character string      : CMPC3, CMPC5
N  2  Character string      : MATCHC
B  3  Character string      : MATCHC
C  3  Character string      : MATCHC OUTER LOOP
D  3  Character string      : MATCHC OUTER LOOP
E  3  Character string      : MATCHC INNER LOOP
F  3  Character string      : MATCHC INNER LOOP
G  3  Character string      : MATCHC TERMINATION
H  3  Character string      : MATCHC TERMINATION
I  3  Character string      : MATCHC FPD + RESTART
J  3  Character string      : MATCHC FPD + RESTART
K  3  Character string      : MOVTC, MOVTUC
L  3  Character string      : MOVTC, MOVTUC
M  3  Character string      : MOVTC, MOVTUC
N  3  Character string      : MOVTC, MOVTUC
B  4  Character string      : MOVTC/MOVTUC LOOP EXITS
C  4  Character string      : MOVTC/MOVTUC LOOP EXITS
D  4  Character string      : MOVTC/MOVTUC LOOP EXITS
E  4  Character string      : MOVTC/MOVTUC LOOP EXITS
F  4  EDIT.MIC
G  4  Edit instruction      : ALGORITHM
H  4  Edit instruction      : ALGORITHM
I  4  Edit instruction      : EDITPC entry
J  4  Edit instruction      : EDITPC entry
K  4  Edit instruction      : SIGN EVALUATION
L  4  Edit instruction      : SIGN EVALUATION
M  4  Edit instruction      : PATTERN DECODE
N  4  Edit instruction      : PATTERN DECODE
B  5  Edit instruction      : PATTERN DECODE
C  5  Edit instruction      : PATTERN DECODE
D  5  Edit instruction      : PATTERN DECODE
E  5  Edit instruction      : PATTERN DECODE
F  5  Edit instruction      : PATTERN DECODE
G  5  Edit instruction      : BRIEF PATTERNS
H  5  Edit instruction      : BRIEF PATTERNS
I  5  Edit instruction      : MOVE + FLOAT
J  5  Edit instruction      : MOVE + FLOAT
K  5  Edit instruction      : MOVE + FLOAT
L  5  Edit instruction      : MOVE + FLOAT
M  5  Edit instruction      : MOVE + FLOAT
N  5  Edit instruction      : OTHER PATTERNS
B  6  Edit instruction      : OTHER PATTERNS
C  6  Edit instruction      : ADJUST INPUT
D  6  Edit instruction      : ADJUST INPUT
E  6  Edit instruction      : ADJUST INPUT
F  6  Edit instruction      : ADJUST INPUT
G  6  Edit instruction      : TERMINATION
H  6  Edit instruction      : TERMINATION
I  6  Edit instruction      : TERMINATION
J  6  Edit instruction      : FPD + RESTART
K  6  Edit instruction      : FPD + RESTART
L  6  Edit instruction      : FPD + RESTART
M  6  Edit instruction      : FPD + RESTART
N  6  Edit instruction      : FPD + RESTART
B  7  Edit instruction      : FPD + RESTART
C  7  DECMAL.MIC
D  7  Decimal string        : MOVP
E  7  Decimal string        : MOVP
F  7  Decimal string        : MOVP
G  7  Decimal string        : MOVP
H  7  Decimal string        : MOVP
I  7  Decimal string        : MOVP
J  7  Decimal string        : CMPP3, CMPP4
K  7  Decimal string        : CMPP3, CMPP4
L  7  Decimal string        : CMPP3, CMPP4
M  7  Decimal string        : CMPP3, CMPP4
N  7  Decimal string        : CMPP3, CMPP4
B  8  Decimal string        : CMPP3, CMPP4
C  8  Decimal string        : CMPP3, CMPP4
D  8  Decimal string        : CMPP3, CMPP4
E  8  Decimal string        : CVTLP
F  8  Decimal string        : CVTLP
G  8  Decimal string        : CVTLP
H  8  Decimal string        : CVTLP
I  8  Decimal string        : CVTLP
J  8  Decimal string        : CVTLP
K  8  Decimal string        : CVTPL
L  8  Decimal string        : CVTPL
M  8  Decimal string        : CVTPL
N  8  Decimal string        : CVTPL
B  9  Decimal string        : CVTPL
C  9  Decimal string        : CVTPL
D  9  Decimal string        : CVTPS
E  9  Decimal string        : CVTPS
F  9  Decimal string        : CVTPT
G  9  Decimal string        : CVTPT
H  9  Decimal string        : CVTPT
I  9  Decimal string        : CVTPT
J  9  Decimal string        : CVTPT
K  9  Decimal string        : CVTPT
L  9  Decimal string        : CVTPT
M  9  Decimal string        : CVTPT
N  9  Decimal string        : CVTPT
B 10  Decimal string        : CVTTP
C 10  Decimal string        : CVTTP
D 10  Decimal string        : CVTTP
```

```
E 10  Decimal string        : CVTTP
F 10  Decimal string        : CVTTP
G 10  Decimal string        : CVTTP
H 10  Decimal string        : CVTTP
I 10  Decimal string        : CVTSP
J 10  Decimal string        : CVTSP
K 10  Decimal string        : CVTSP
L 10  Decimal string        : ADDP4, ADDP6, SUBP4, SUBP6
M 10  Decimal string        : ADDP4, ADDP6, SUBP4, SUBP6
N 10  Decimal string        : ADDP4, ADDP6, SUBP4, SUBP6
B 11  Decimal string        : ADDP4, ADDP6, SUBP4, SUBP6
C 11  Decimal string        : ADDP4, ADDP6, SUBP4, SUBP6
D 11  Decimal string        : ADDP4, ADDP6, SUBP4, SUBP6
E 11  Decimal string        : ADDP4, ADDP6, SUBP4, SUBP6
F 1   Decimal string        : ADDP4, ADDP6, SUBP4, SUBP6
G 11  Decimal string        : ADDP4, ADDP6, SUBP4, SUBP6
H 11  Decimal string        : ADDP4, ADDP6, SUBP4, SUBP6
I 11  Decimal string        : ADDP4, ADDP6, SUBP4, SUBP6
J 11  Decimal string        : ADDP4, ADDP6, SUBP4, SUBP6
K 11  Decimal string        : ADDP4, ADDP6, SUBP4, SUBP6
L 11  Decimal string        : ADDP4, ADDP6, SUBP4, SUBP6
M 11  Decimal string        : ADDP4, ADDP6, SUBP4, SUBP6
N 11  Decimal string        : ADDP4, ADDP6, SUBP4, SUBP6
B 12  Decimal string        : MULP
C 12  Decimal string        : MULP
D 12  Decimal string        : MULP
E 12  Decimal string        : MULP
F 12  Decimal string        : MULP
G 12  Decimal string        : MULP
H 12  Decimal string        : MULP
I 12  Decimal string        : MULP
J 12  Decimal string        : MULP
K 12  Decimal string        : MULP
L 12  Decimal string        : MULP
M 12  Decimal string        : MULP
N 12  Decimal string        : MULP
B 13  Decimal string        : MULP
C 13  Decimal string        : MULP
D 13  Decimal string        : DIVP
E 13  Decimal string        : DIVP
F 13  Decimal string        : DIVP
G 13  Decimal string        : DIVP
H 13  Decimal string        : DIVP
I 13  Decimal string        : DIVP
J 13  Decimal string        : DIVP
K 13  Decimal string        : DIVP
L 13  Decimal string        : DIVP
M 13  Decimal string        : DIVP
N 13  Decimal string        : DIVP
B 14  Decimal string        : DIVP
C 14  Decimal string        : DIVP
D 14  Decimal string        : DIVP
E 14  Decimal string        : DIVP
F 14  Decimal string        : DIVP
G 14  Decimal string        : ASHP
H 14  Decimal string        : ASHP
I 14  Decimal string        : ASHP
J 14  Decimal string        : ASHP
K 14  Decimal string        : ASHP
L 14  Decimal string        : ASHP
M 14  Decimal string        : ASHP
N 14  Decimal string        : ASHP
B 15  Decimal string        : BCD-READ SUBROUTINE
C 15  Decimal string        : BCD-READ SUBROUTINE
D 15  Decimal string        : BCD-READ SUBROUTINE
E 15  Decimal string        : BCD-READ SUBROUTINE
F 15  Decimal string        : BCD-READ-WITH-WRITE-CHECK SUBROUTINE
G 15  Decimal string        : BCD-READ-WITH-WRITE-CHECK SUBROUTINE
H 15  Decimal string        : BCD-READ-WITH-WRITE-CHECK SUBROUTINE
I 15  Decimal string        : BCD-WRITE SUBROUTINE
J 15  Decimal string        : BCD-WRITE SUBROUTINE
K 15  Decimal string        : BCD-WRITE SUBROUTINE
L 15  Decimal string        : BCD-WRITE SUBROUTINE
M 15  Decimal string        : BCD-WRITE SUBROUTINE
N 15  Decimal string        : FAULT PARAMETER SAVE-ROUTINES
B 16  Decimal string        : FAULT PARAMETER SAVE-ROUTINES
C 16  Decimal string        : FAULT PARAMETER SAVE-ROUTINES
D 16  Decimal string        : FAULT PARAMETER SAVE-ROUTINES
E 16  Decimal string        : FAULT PARAMETER SAVE-ROUTINES
F 16  Decimal string        : FAULT PARAMETER SAVE-ROUTINES
G 16  Decimal string        : FAULT PARAMETER SAVE-ROUTINES
H 16  Decimal string        : FAULT PARAMETER SAVE-ROUTINES
I 16  Decimal string        : FAULT PARAMETER SAVE-ROUTINES
J 16  Decimal string        : FAULT PARAMETER SAVE-ROUTINES
K 16  Decimal string        : FAULT PARAMETER SAVE-ROUTINES
L 16  Decimal string        : FAULT PARAMETER SAVE-ROUTINES
```