```
674E 066968   699      CALL STDATA                                    1
              700    SEND
6751 00       701    RTN
```

```
                    703  PHRASE
6752 D64A23         704    $IF @CCHAR .EQ. NUMBER THEN  Do we have a phrase?        1
6755 478E
6757 9108           705      DINC @PTCCIS  You bet we do!                           1
6759 D68008         706      $WHILE RAM(@PTCCIS) .EQ. SPACE                         2
675C 204764
675F 9108           707        DINC @PTCCIS                                         2
6761 056759         708      $END                                                   1
6764 D68008         709      $IF RAM(@PTCCIS) .EQ. NUMBER THEN  Null phrase (all spa2
6767 234760
676A 9108           710        DINC @PTCCIS  Yes. Go on to next char                2
676C 00             711        RTN                  Ignore this phrase. go get next c2
                    712      $END
6760 B00C03         713      DST @PTCCIS,@PTFCIP                                    1
                    714      $REPEAT
6770 9108           715        DINC @PTCCIS                                         2
6772 BC4A80         716        ST RAM(@PTCCIS),@CCHAR                               2
6775 08
6776 C5080A         717        $IF @PTCCIS .DH. @PTLCIS GOTO FNDNUM                 2
6779 6780
677B D64A23         718      $UNTIL @CCHAR .EQ. NUMBER                              1
677E 4770
                    719  FNDNUM
6780 B01008         720      DST @PTCCIS,@PTLCIP                                    1
6783 9310           721      DDEC @PTLCIP                                           1
6785 9108           722      DINC @PTCCIS  Point to char after #                    1
6787 0667BD         723      CALL GETTIM                                            1
678A 864C           724      CLR @SPLFLG  Clear spell-flag (don't spell)            1
678C 47AC           725    $ELSE                                                    1
678E B00C08         726      DST @PTCCIS,@PTFCIP                                    1
6791 CA4A41         727      $IF @CCHAR .HE. ASCIIA THEN  1st char alfabetic?       2
6794 479A
6796 864C           728        CLR @SPLFLG  Yes. clear spell-flag (don't spell)     2
6798 479D           729      $ELSE                                                  2
679A BE4C01         730        ST >1,@SPLFLG  NO. SET SPELL-FLAG (DO SPELL)         2
                    731      $END
                    732      $REPEAT
679D 9108           733        DINC @PTCCIS                                         2
679F 0667BD         734        CALL GETTIM                                          2
67A2 D651FF         735      $UNTIL @TIMLEN .NE. >FF  PTCC WILL POINT TO FOLLOWING CH
67A5 679D
67A7 B01008         736      DST @PTCCIS,@PTLCIP                                    1
67AA 9310           737      DDEC @PTLCIP                                           1
                    738    $END
67AC 00             739    RTN
```

```
                 741  TIMING
67AD CA51FE      742    $WHILE @TIMLEN .NOT. .HE. >FE                          1
67B0 67BC
67B2 A14B51      743      DADD @TIMLEN.@TOTTIM                                  1
67B5 9108        744      DINC @PTCCIS                                          1
67B7 0667BD      745      CALL GETTIM                                           1
67BA 47AD        746    $END
67BC 00          747    RTN
                 748
                 749  GETTIM
67BD BC4AB0      750    ST RAM(@PTCCIS).@CCHAR
67C0 08
67C1 B05408      751    DST @PTCCIS.@TEMP1
67C4 C5540A      752    $IF @TEMP1 .GH. @PTLCIS THEN                            1
67C7 47CD
67C9 BE51FE      753      ST >FE.@TIMLEN                                        1
67CC 00          754      RTN                                                   1
                 755    $END
67CD C64A3B      756    $IF @CCHAR .NOT. .H. SEMICO THEN                        1
67D0 682F
67D2 D64A20      757      $IF @CCHAR .EQ. SPACE THEN                            2
67D5 47DE
67D7 BE5106      758        ST >06.@TIMLEN                                      2
67DA 066840      759        CALL SKPSPC  Skip over all extra spaces             2
67DD 00          760        RTN                                                 2
                 761      $END
67DE D64A2B      762      $IF @CCHAR .EQ. PLUS THEN                             2
67E1 47ED
67E3 9154        763        DINC @TEMP1  Need to test the next char             2
67E5 066833      764        CALL NUMERC  Is it numeric                          2
67E8 682F        765        BS NOTIME  Was numeric, therefore not a timing char 2
67EA 8651        766        CLR @TIMLEN  NO, set as notiming                    2
67EC 00          767        RTN                                                 2
                 768      $END
67ED D64A2C      769      $IF @CCHAR .EQ. COMMA THEN                            2
67F0 47F6
67F2 BE5112      770        ST >12.@TIMLEN                                      2
67F5 00          771        RTN                                                 2
                 772      $END
67F6 D64A2E      773      $IF @CCHAR .EQ. PERIOD THEN                           2
67F9 480D
67FB 9354        774        DDEC @TEMP1  Go back to preceding char              2
67FD 066833      775        CALL NUMERC  Is it numeric?                         2
6800 4809        776        BR PTIME                                            2
6802 9554        777        DINCT @TEMP1  Yes, on to following char             2
6804 066833      778        CALL NUMERC  Is it numeric too?                     2
6807 682F        779        BS NOTIME  Both were numeric, therefore not a timing 2
6809 BE513C      780  PTIME ST >3C.@TIMLEN  Both were not numeric, therefore a ti2
680C 00          781        RTN                                                 2
                 782      $END
680D D64A20      783      $IF @CCHAR .EQ. HYPEN THEN                            2
6810 4810
6812 9154        784        DINC @TEMP1  Check next char                        2
6814 066833      785        CALL NUMERC  Is it numeric?                         2
6817 682F        786        BS NOTIME  Was numeric, therefore not a timing char 2
```

```
6819 BE510C   787        ST >0C,@TIMLEN  Was not numeric, therefore set as tim
681C 00       788        RTN
              789      $END
681D 064A3A   790      $IF @CCHAR .EQ. COLON THEN
6820 4826
6822 BE5130   791        ST >30,@TIMLEN
6825 00       792        RTN
              793      $END
6826 064A3B   794      $IF @CCHAR .EQ. SEMICO THEN
6829 482F
682B BE511E   795        ST >1E,@TIMLEN
682E 00       796        RTN
              797      $END
              798    $END
              799  NOTIME
682F BE51FF   800    ST >FF,@TIMLEN  Set as no timing char present
6832 00       801    RTN
```

```
                 803  *  This routine tests the char pointed to by PTCCIS and dete
                 804  *       1- it is within the current string boundaries
                 805  *       2- it is numeric  (is between '0' and '9')
                 806  *  If both of the above conditions are true, COND is set upo
                 807  *  Otherwise, COND will be reset
                 808  NUMERC
6833 C5540A      809    $IF @TEMP1 .NOT. .OR. @PTLCIS THEN                        1
6836 684C
6838 C50654      810      $IF @PTFCIS .NOT. .OR. @TEMP1 THEN                      2
683B 684C
683D CA8054      811        $IF RAM(@TEMP1) .HE. ZERO THEN                        3
6840 304840
6843 C68054      812          $IF RAM(@TEMP1) .NOT. .H. NINE THEN                 4
6846 374840
6849 040101      813            CEQ 201.201  Set condition bit to on              4
                 814          $END
                 815        $END
                 816      $END
                 817    $END
684C 01          818    RTNC
                 819
                 820  SKPSPC
6840 06E001      821    $WHILE RAM(1(PTCCIP)) .EQ. SPACE                          1
6850 0E2046
6853 58
6854 910E        822      DINC @PTCCIP                                            1
6856 4240        823    $END
6858 00          824    RTN
                 825
                 826  LOOKUP
6859 8766        827    DCLR  @BYTE1
                 828  TRYAGN
685B B05266      829    DST @BYTE1,@PHADDR  Check for >AA heading of on board Phro
685E 06691E      830    CALL LOADAD
6861 BEC000      831    ST >10.@D(WRITE)
6864 5A10
6866 D6C000      832    $IF @D(READ) .EQ. >AA THEN  Is there a Phrom out there?  1
6869 58AA48
686C 79
686D B01266      833      DST @BYTE1,@PTF@PH  Yes, prepare to search it from the b
6870 9112        834      DINC @PTF@PH  Just one beyond the header                1
6872 066824      835      CALL SEARCH                                             1
6875 8F4D48      836      $IF @DATAAD .DNE. 0 GOTO FOUND                          1
6878 83
                 837    $END
6879 A36680      838    DADD >8000.@BYTE1
687C 00
687D 8F6648      839    $IF @BYTE1 .DNE. 0 GOTO TRYAGN
6880 58
6881 8740        840    DCLR @DATAAD
                 841  FOUND
6883 00          842    RTN
```

```
                    844  SEARCH
6884 B05212         845    DST @PTFBPH,@PHADDR  LOADAD must have addr in PHADDR
6887 06691E         846    CALL LOADAD  Set up the Phrom to scan a word
688A BEC000         847    ST >10,@0(WRITE)
688D 5A10
688F 8616           848    CLR @PTLCPH                Save the lenght of the word in 2 b
6891 BC17C0         849    ST @0(READ),@PTLCPH+1        at PTLCPH
6894 0058
6896 A11612         850    DADD @PTFBPH,@PTLCPH     Then add front pointer giving end
6899 B01412         851    DST @PTFBPH,@PTCCPH  Set up current byte pointer into Phro
689C 9114           852    DINC @PTCCPH  1 byte beyond the lenght byte is first char
689E B00E0C         853    DST @PTFCIP,@PTCCIP  Reset current byte pointer into phras
                    854  *      Try comparing chars
                    855  NEXT
68A1 BEC000         856    ST >10,@0(WRITE)  Read a byte from Phrom
68A4 5A10
68A6 BC50C0         857    ST @0(READ),@PHDATA  Get data in from the Phrom
68A9 0058
68AB D45080         858    $IF @PHDATA .EQ. RAM(@PTCCIP) THEN  Compare Phrom with Phr
68AE 0E43FD
68B1 9114           859      DINC @PTCCPH  Equal, advance both pointers                1
68B3 910E           860      DINC @PTCCIP                                              1
68B5 06B00E         861      $IF RAM(@PTCCIP) .EQ. SPACE THEN  Need to skip extra sp2
68B8 2043C7
68BB 066840         862        CALL SKPSPC                                             2
68BE 06E001         863        $IF RAM(1(PTCCIP)) .EQ. NUMBER THEN                     3
68C1 0E2348
68C4 C7
68C5 910E           864          DINC @PTCCIP                                          3
                    865        $END
                    866      $END
68C7 C51416         867      $IF @PTCCPH .DH. @PTLCPH THEN  Scanned all char in Phrom
68CA 48F0
68CC C50E10         868        $IF @PTCCIP .DH. @PTLCIP THEN  Yes, scanned all char 3
68CF 48E8
68D1 B05216         869          DST @PTLCPH,@PHADDR  Yes, Word found                  3
68D4 A35200         870          DADD >6,@PHADDR  Move 5 bytes down from last char to
68D7 06
68D8 066950         871          CALL READAD  This will set up data addr in DATAAD  3
68DB BEC000         872          ST >10,@0(WRITE)                                      3
68DE 5A10
68E0 BC64C0         873          ST @0(READ),@STRLEN                                   3
68E3 0058
68E5 00             874          RTN                                                   3
68E6 48EE           875        $SELSE  No, Phrase is longer, follow the higher link 3
68E8 BF5200         876          DST >3,@PHADDR  Move 3 bytes beyond PTLCPH to get to
68EB 03
68EC 4900           877          BR NXTPHR                                             3
                    878        $END                                                    2
68EE 48FB           879      $SELSE
68F0 C50E10         880        $IF @PTCCIP .NOT. .DH. @PTLCIP GOTO NEXT  Go compare 8
68F3 48A1
68F5 BF5200         881        DST >1,@PHADDR  Move 1 byte beyond PTLCPH to get to  2
68F8 01
68F9 4900           882        BR NXTPHR                                             2
```

```
                  883       $END
68FB 491E         884     $SELSE                                                1
68FD C45080       885       $IF @PHDATA .H. RAM(@PTCCIP) THEN  Phrom word is before2
6900 0E4909
6903 8F5200       886         DST >1.@PHADDR  Go 1 byte beyond last char to lower 1
6906 01
6907 4900         887       $SELSE  Phrase word comes before Phrom word           2
6909 8F5200       888         DST >3.@PHADDR  Go 3 bytes beyond last char to higher2
690C 03
                  889       $END
                  890   NXTPHR
690D A15216       891       DADD @PTLCPH.@PHADDR                                  1
6910 066950       892       CALL READAD  Now get the new addr                     1
6913 8F4D49       893       $IF @DATAAD .DEQ. #0 THEN  Any more leaves on this tree2
6916 18
6917 00           894         RTN                    No, return empty handed      2
                  895       $END                     Yes. set up to check next word
6918 B0124D       896       DST @DATAAD.@PTFBPH  Store new addr in PTFBPH         1
691B 056824       897       B SEARCH  Go compare this new word!                   1
                  898     $END
                  899   *The program cannot reach this point!!
```

```
                    901  LOADAD
691E B05452         902    DST  @PHADDR.@TEMP1     This is destructive, so copy addres
6921 B05652         903    DST  @PHADDR.@TEMP2      into temporary areas
6924 E65404         904    SRL  @TEMP1.4           Isolate the MSN of the MSB
6927 E65504         905    SRL  @TEMP1+1.4         Isolate the MSN of the LSB
692A B3560F         906    DAND >0F0F.@TEMP2       Isolate the LSN of both the MSB, LS
692D 0F
692E B75440         907    DOR  >4040.@TEMP1       Include a 4 as the MSN of all four
6931 40
6932 B75640         908    DOR  >4040.@TEMP2        to indicate a Load Address Command
6935 40
6936 BCC000         909    ST   @TEMP2+1.@0(WRITE)  Write out the LSN of the LSB
6939 5A57
693B BCC000         910    ST   @TEMP1+1,@0(WRITE)  Write out the LSN of the MSB
693E 5A55
6940 BCC000         911    ST   @TEMP2.@0(WRITE)    Write out the MSN of the LSB
6943 5A56
6945 BCC000         912    ST   @TEMP1,@0(WRITE)    Write out the MSN of the MSB
6948 5A54
694A BEC000         913    ST   >40.@0(WRITE)       Write out 0 as fifth nybble
694D 5A40
694F 00             914    RTN
                    915
                    916  READAD
6950 06691E         917    CALL LOADAD  PHADDR should be set up with the addr in it
6953 BEC000         918    ST >10.@0(WRITE)  Get high byte of addr
6956 5A10
6958 BC4DC0         919    ST @0(READ),@DATAAD  Store it in DATAAD
695B 0058
695D BEC000         920    ST >10.@0(WRITE)  Get low byte of addr
6960 5A10
6962 BC4EC0         921    ST @0(READ),@DATAAD+1  Store it in DATAAD+1
6965 0058
6967 00             922    RTN
                    923
                    924  STDATA
6968 D50204         925    $IF @PTLBSL .DEQ. @PTEBSL GOTO ERR6
696B 6BCD
696D BCB002         926    ST @TOTTIM.RAM(@PTLBSL)
6970 48
6971 9102           927    DINC @PTLBSL
6973 BCB002         928    ST @DATAAD.RAM(@PTLBSL)
6976 4D
6977 9102           929    DINC @PTLBSL
6979 BCB002         930    ST @DATAAD+1.RAM(@PTLBSL)
697C 4E
697D 9102           931    DINC @PTLBSL
697F 00             932    RTN
                    933
                    934  SETRW
6980 310004         935    MOVE 4 FROM ROM(>46) TO @READ
6983 580046
6986 00             936    RTN
```

```
                    938  SPEAK
6987 066980         939    CALL SETRW
698A C90002         940    $WHILE @PTFBSL .NOT. .DHE. @PTLBSL                    1
698D 6A13
698F 066A1F         941      CALL WAIT                                           1
6992 D6B000         942      $IF RAM(@PTFBSL) .NE. >FF THEN                      2
6995 FF69BA
6998 BC79B0         943        ST RAM(@PTFBSL),@TIMER                            2
699B 00
699C 8279           944        NEG @TIMER                                        2
699E BD52E0         945        DST RAM(1(PTFBSL)),@PHADDR                        2
69A1 0100
69A3 A30000         946        DADD 3,@PTFBSL                                    2
69A6 03
69A7 D27900         947  LOOP1 $IF @TIMER .LT. 0 GOTO LOOP1                      2
69AA 49A7
69AC 6E5269         948        $IF @PHADDR .NE. 0 THEN                           3
69AF B8
69B0 06691E         949          CALL LOADAD                                     3
69B3 BEC000         950          ST >50,@0(WRITE)                                3
69B6 5A50
                    951        $END
69B8 4A10           952      $$ELSE                                              2
69BA 9100           953        DINC @PTFBSL  Skip >FF                            2
69BC BD5EB0         954        DST RAM(@PTFBSL),@PTCBED  Set up pointer to front byte
69BF 00
69C0 D6805E         955        $IF RAM(@PTCBED) .NE. >60 GOTO ERR1 Invalid speech str
69C3 6048BE
69C6 8662           956        CLR @PTEBED                                       2
69C8 BC63EF         957        ST RAM(-1(PTCBED)),@PTEBED+1  Get total lenght of str2
69CB FFFF5E
69CE A1625E         958        DADD @PTCBED,@PTEBED  Lenght + first byte = last byte2
69D1 9500           959        DINCT @PTFBSL  Skip addr bytes                    2
                    960  DIRSPH
69D3 BEC000         961        ST >60,@0(WRITE)  Start Speak External            2
69D6 5A60
69D8 915E           962        DINC @PTCBED  Skip speak external command         2
69DA BD60B0         963        DST RAM(@PTCBED),@PTLBED  Set up pointer to last byte2
69DD 5E
69DE 955E           964        DINCT @PTCBED  Skip lenght bytes, this is first real b
69E0 A1605E         965        DADD @PTCBED,@PTLBED  Add lenght to first byte giving2
69E3 BE5610         966        ST 16,@TEMP2                                      2
69E6 D55E62         967  LOOPR   $IF @PTCBED .DEQ. @PTEBED GOTO CONTIN           2
69E9 6A10
69EB D55E60         968          $IF @PTCBED .DEQ. @PTLBED THEN                  3
69EE 49F6
69F0 066A1F         969            CALL WAIT                                     3
69F3 056903         970            B DIRSPH                                      3
                    971          $END
69F6 BCC000         972          ST RAM(@PTCBED),@0(WRITE)                       2
69F9 5A805E
69FC 915E           973          DINC @PTCBED                                    2
69FE 9256           974        DEC @TEMP2                                        2
6A00 49E6           975        BR LOOPR                                          2
                    976        $REPEAT
```

```
6A02 066A14   977          CALL GETSTS                                3
6A05 DA6940   978        $UNTIL .BIT6 @SPKSTS .EQ. 1                 2
6A08 6A02
6A0A BE5603   979        ST 8,@TEMP2                                 2
6A0D 0569E6   980        B LOOPR                                     2
              981      $END
              982  CONTIN
6A10 05698A   983    $END
6A13 00       984    RTN
              985
              986
              987  GETSTS
6A14 BC69C0   988    ST 20(READ),@SPKSTS
6A17 0058
6A19 BC69C0   989    ST 20(READ),@SPKSTS
6A1C 0058
6A1E 00       990    RTN
              991
              992  WAIT
              993    $REPEAT
6A1F 066A14   994      CALL GETSTS                                    1
6A22 DA6980   995    $UNTIL .BIT7 @SPKSTS .EQ. 0
6A25 4A1F
6A27 00       996    RTN
```

```
                  998  ***********************************************************
                  999  *
                 1000  *      BBBB     AAA     SSSS    III     CCC
                 1001  *      B   B   A   A   S         I     C
                 1002  *      BBBB    AAAAA    SSS      I     C
                 1003  *      B   B   A   A       S     I     C
                 1004  *      BBBB    A   A   SSSS     III     CCC
                 1005  *
                 1006  ********* S U B R O U T I N E   D E F I N I T I O N S ******
                 1007  *
                 1008  *
                 1009  ***********************************************************
                 1010  *  CALL SAY(........)
                 1011  *      Decode given parameter(s).  Store all data first, the
                 1012  *      go speak it all at once.
                 1013  ***********************************************************
                 1014  *
                 1015  SAY
6A28 066A85      1016    CALL GETPAR  Skip SAY header and "("
6A2B BF0C00      1017    DST 255,@0C  At 3 bytes/entry 255 bytes = 85 entries.
6A2E FF
6A2F 060033      1018    CALL GETSTR  Get the string space to be used as the speak
6A32 BF4A00      1019    DST SREF,@FAC  Set up a string FAC entry
6A35 1C
6A36 BE4C65      1020    ST >65,@FAC+2
6A39 B04E1C      1021    DST @SREF,@FAC+4
6A3C BF5000      1022    DST 255,@FAC+6
6A3F FF
6A40 0F17        1023    XML VPUSH      and make it semi-permenant!
6A42 B0004E      1024    DST @FAC+4,@PTFBSL  Set up pointers into the speak list
6A45 B0024E      1025    DST @FAC+4,@PTLBSL
6A48 B00400      1026    DST @PTFBSL,@PTEBSL
6A4B A30400      1027    DADD 255,@PTEBSL
6A4E FF
                 1028  DIRSPK
6A4F 066A95      1029    CALL GETPRM  Get parameter
6A52 8F506A      1030    $IF @FAC+6 .DNE. #0 THEN  Found non-null ASCII string    1
6A55 64
6A56 B0064E      1031      DST @FAC+4,@PTFCIS  Set up pointer to first char in stri
6A59 B00A50      1032      DST @FAC+6,@PTLCIS  Set up pointer-to-last-char-in-strin
6A5C A10A06      1033      DADD @PTFCIS,@PTLCIS    by adding lenght-of-string to p
6A5F 930A        1034      DDEC @PTLCIS            first-char-in-string and subtrac
6A61 0666E6      1035      CALL MAKESL  Go make the speak list                     1
                 1036    $END
                 1037  *      Next item could be direct string.
6A64 D642B3      1038    $IF @CHAT .EQ. COMMAS THEN  Direct string present       1
6A67 4A7E
6A69 066A95      1039      CALL GETPRM  Get next parameter                         1
6A6C 8F506A      1040      $IF @FAC+6 .DNE. #0 THEN  Found non-null direct speak 12
6A6F 79
6A70 BE4BFF      1041        ST >FF,@TOTTIM  Mark TOTTIM as direct string addr with
6A73 B0404E      1042        DST @FAC+4,@DATAAD  Put addr of string in DATAAD     2
6A76 066968      1043        CALL STDATA  And put this on the speak list          2
                 1044      $END
6A79 D642B3      1045      $IF @CHAT .EQ. COMMAS GOTO DIRSPK  Now another word stri
```

```
6A7C 6A4F
                    1046    $END
6A7E 066987         1047    CALL SPEAK  Go say all that is on the Speak list
6A81 0F18           1048    XML VPOP  Get rid of the string space
6A83 485C           1049    BR RETURN  Return to main program
```

```
                 1051  ***********************************************************
                 1052  *
                 1053  *       Get parameter and check for string arguments
                 1054  *
                 1055  ***********************************************************
                 1056  GETPAR
6A85 8E8089      1057    $IF @GROMFL .EQ. 0 THEN Working on VDP program          1
6A88 4A8D
6A8A B02C56      1058      DST  @FAC+12,@CPU2C   and update the program pointer  1
                 1059    $END IF
6A8D 0F18        1060    XML  PGMCHR            Get next token
6A8F D642B7      1061    $IF @CHAT.NE. LPAR$ GOTO ERR2 Quit if no left parenthesis
6A92 4BC3
6A94 00          1062    RTN
                 1063
                 1064  GETPRM
6A95 0F18        1065    XML  PGMCHR            Get next token
6A97 0EB6        1066    PARSE RPAR$
6A99 D64C65      1067    $IF @FAC+2 .NE. >65 GOTO ERR3  Not a string, something is
6A9C 4BC8
6A9E 00          1068    RTN                    Return normal
```

```
                  1070  ***********************************************************
                  1071  *
                  1072  *       SPGET subprogram.  Load speech data from external
                  1073  *       device.  Use standard file I/O.
                  1074  *
                  1075  ***********************************************************
                  1076  *
                  1077  SPGET
6A9F 066A85       1078    CALL GETPAR              Skip header and "("
                  1079  NXTPAR
6AA2 066A95       1080    CALL GETPRM  Get parameter
6AA5 8F50EB       1081    $IF @FAC+6 .ONE. #0 THEN  Found non-null ASCII string    1
6AA8 57
6AA9 B0064E       1082      DST @FAC+4,@PTFCIS  Set up pointer to first char in stri
6AAC B00A50       1083      DST @FAC+6,@PTLCIS  Set up pointer-to-last-char-in-strin
6AAF A10A06       1084      DADD @PTFCIS,@PTLCIS     by adding length-of-string to o
6AB2 930A         1085      DDEC @PTLCIS             first-char-in-string and subtrac
6AB4 066780       1086      CALL SETRW                                                1
6AB7 B00806       1087      DST @PTFCIS,@PTCCIS                                       1
6ABA 0667B0       1088      CALL GETTIM                                               1
6ABD 8648         1089      CLR @TOTTIM                                               1
6ABF 0667A0       1090      CALL TIMING                                               1
6AC2 C5020A       1091      $IF @PTCCIS .NOT. .GR. @PTLCIS THEN                       2
6AC5 6B57
6AC7 066752       1092        CALL PHRASE                                             2
6ACA D64C01       1093        $IF @SPLFLG .EQ. 1 THEN                                 3
6ACD 4AD2
6ACF B0100C       1094          DST @PTFCIP,@PTLCIP                                   3
                  1095        $END
6AD2 066859       1096        CALL LOOKUP                                             2
6AD5 8F4D4A       1097        $IF @DATAAD .DEQ. 0 THEN                                3
6AD8 E0
6AD9 BF4071       1098          DST >71F4,@DATAAD                                     3
6ADC F4
6ADD BE6451       1099          ST >51,@STRLEN                                        3
                  1100        $END
6AE0 B00140       1101        DST @DATAAD,@PHRADD                                     2
6AE3 BC0064       1102        ST @STRLEN,@PHLEN                                       2
6AE6 A20003       1103        ADD >3,@PHLEN                                           2
6AE9 D64283       1104        $IF @CHAT .NE. COMMAS GOTO ERR2                         2
6AEC 4BC3
6AEE 0F18         1105        XML PGMCHR                                              2
6AF0 0F13         1106        XML SYM                                                 2
6AF2 0F14         1107        XML SMB                                                 2
6AF4 0F17         1108        XML VPUSH                                               2
6AF6 860C         1109        CLR @0C                                                 2
6AF8 BC0000       1110        ST @PHLEN,@0D                                           2
6AFB 060038       1111        CALL GETSTR                                             2
6AFE 066980       1112        CALL SETRW                                              2
6B01 BF4A00       1113        DST SREF,@FAC                                           2
6B04 1C
6B05 BE4C65       1114        ST >65,@FAC+2                                           2
6B08 BD4E1C       1115        DST @SREF,@FAC+4                                        2
6B0B 8650         1116        CLR @FAC+6                                              2
6B0D BC5100       1117        ST @PHLEN,@FAC+7                                        2
```

```
6B10 BEE000  1118          ST >60.RAM(0(SREF))                          2
6B13 1C60
6B15 86E001  1119          CLR RAM(1(SREF))                             2
6B18 1C
6B19 BCE002  1120          ST @PHLEN.RAM(2(SREF))                       2
6B1C 1CC0
6B1E A3E001  1121          DADD -3.RAM(1(SREF))                         2
6B21 1CFFFD
6B24 B05201  1122          DST @PHRADD,@PHADDR                          2
6B27 06691E  1123          CALL LOADAD                                  2
6B2A C60003  1124          $WHILE @PHLEN .N. 3                          3
6B2D 4B55
6B2F BEC000  1125            ST >10.@0(WRITE)                           3
6B32 5A10
6B34 BC68C0  1126            ST @0(READ).@BYTE3                         3
6B37 0056
6B39 8667    1127            CLR @BYTE2                                 3
6B3B BE5409  1128            ST >8.@TEMP1                               3
6B3E E26700  1129  RNDAG       DERC @BYTE2.#1                           3
6B41 01
6B42 E36600  1130              DSLL @BYTE1.#1                           3
6B45 01
6B46 9254    1131            DEC @TEMP1                                 3
6B48 433E    1132            BR RNDAG                                   3
6B4A BCE003  1133            ST @BYTE1.RAM(3(SREF))                     3
6B4D 1C66
6B4F 911C    1134            DINC @SREF                                 3
6B51 9200    1135            DEC @PHLEN                                 3
6B53 482A    1136          $SEND                                        3
6B55 0F15    1137          XML ASSGNV                                   2
             1138        $END                                           2
             1139      $END
6B57 D642B3  1140      $IF @CHAT .EQ. COMMAS GOTO NXTPAR
6B5A 6AA2
             1141    RETURN
6B5C D642B6  1142      $IF @CHAT .NE. RPARS GOTO ERR2
6B5F 4BC3
6B61 0F18    1143      XML PGMCHR
6B63 060012  1144      CALL RPL
```

```
                 1146          BASE 0,0,>300,>300,0,0,>60
6B66 15A9AC      1147  ERR$1   DATA 21,:ILLEGAL SPEECH FORMAT:
6B69 ACA5A7
6B6C A1AC80
6B6F B3B0A5
6B72 A5A3A8
6B75 80A6AF
6B78 B2A0A1
6B7B B4
6B7C 13A9AE      1148  ERR$2   DATA 19.:INCORRECT STATEMENT:
6B7F A3AFB2
6B82 B2A5A3
6B85 B480B3
6B88 B4A15+
6B8B A5A0A5
6B8E AEB4
6B90 16B3B4      1149  ERR$3   DATA 22.:STRING/NUMBER MISMATCH:
6B93 B2A9AE
6B96 A78FAE
6B99 B5A0A2
6B9C A5B280
6B9F A0A9B3
6BA2 A0A1B4
6BA5 A3A8
6BA7 16B3B0      1150  ERR$6   DATA 22.:SPEECH STRING TOO LONG:
6BAA A5A5A3
6BAD A680B3
6BB0 B4B2A9
6BB3 AEA780
6BB6 B4AFAF
6BB9 80ACAF
6BBC AEA7
                 1151          BASE 0,0,>300,>300,0,0,0
                 1152  ERR1
6BBE 06001C      1153    CALL ERR$$
6BC1 6B66        1154    DATA #ERR$1
                 1155
                 1156  ERR2
6BC3 06001C      1157    CALL ERR$$
6BC6 6B7C        1158    DATA #ERR$2
                 1159
                 1160  ERR3
6BC8 06001C      1161    CALL ERR$$
6BCB 6B90        1162    DATA #ERR$3
                 1163
                 1164  ERR6
6BCD 06001C      1165    CALL ERR$$
6BD0 6BA7        1166    DATA #ERR$6
6BD2 000066      1167  ENDLNK DATA #0.#EXIT
6BD5 72
                 1168          END


 ERRORS=    0

 LENGTH= 3030 (>0B96)


 260 SYMBOLS USED
```

```
                 1        TITLE " GROM
                 2   ***                                                  *********
                 3   **************DEBUGGER VERSION ***************************
                 4   ***                                                  *********
                 5   *EDTTOP EQU  >FFD8->8300      TOP MEMORY FOR EDITOR(REAL GROM
 78F0            6   EDTTOP EQU  >FBF0->8300      TOP MEMORY FOR EDITOR(DEBUGGER)
                 7   ***********************************************************
                 8   *    STANDALONE EDITOR & ASSEMBLER FOR 99/4                *
                 9   ***********************************************************
                10   ***********************************************************
                11   *    USE GCA COMMAND ON THE FOLLOWING FILES                *
                12   *    GCA FILE NAME              SD2.ASSM.SRC.GCA           *
                13   *                                                          *
                14   *    OUT PUT FILE NAME IS       SD2.ASSM.OBJ.GROM          *
                15   *    TO LIST WHOLE GPL CODE     SD2.ASSM.LST.GROM          *
                16   *                                                          *
                17   *    CONCATINATE LOADER TO THIS GPL OBJECT FILE            *
                18   *                               SD2.ASSM.OBJ.GROM          *
                19   *                               SD2.ASSM.OBJ.ALUTIL        *
                20   *                     OUTPUT FILE SD2.ASSM.OBJ.ASSMBLR     *
                21   *                                                          *
                22   *    AUTHOR SUMIKO ENDO     Sept 1981                      *
                23   ***********************************************************
                24   ****
                25   *    EQUATES
                26   *
                27   *    XML EQUATES
                28   *
 0021           29   LOADX  EQU  >21        EXECUTE LOADER
 0022           30   SAVEX  EQU  >22        EXECUTE SAVE ROUTINE
 0023           31   EDITX  EQU  >23        EXECUTE EDITOR
                32   *
 0021           33   ASM    EQU  >21        EXECUTE ASSEMBLER
                34   *
 0021           35   ALSUP  EQU  >21        EXECUTE ASSEMBLY LANGUAGE CODE
 0022           36   TGOBLD EQU  >22        TAGGED OBJECT LOADER.
                37   *
                38   *    INTERFACE WITH BASIC
                39   *
 0012           40   RPL    EQU  >0012      RETURN TO BASIC MAIN PROGRAM.
                41   *
                42   *    GENERAL EQUATES
                43   *
 8300           44   OFFSET EQU  >8300      EQUATE FOR ACCESSING E-RAM
 9000           45   RAMTOP EQU  >2000-OFFSET ONE LOCATION IN EXPANSION MEMORY
 9000           46   IDCODE EQU  >2000-OFFSET ID CODE FOR EDITOR, ASSM, LOADER
 001F           47   CURSOR EQU  >1F        CHARACTER NUMBER FOR CURSOR
 0018           48   CHAR2S EQU  >18        CHARACTER TABLE ADDRESS IN MONITOR
 004A           49   CHAR3S EQU  >4A        LOWER CASE CHARACTERS
 0020           50   SPACE  EQU  >20        SPACE CHARACTER.
 0010           51   DSRLNK EQU  >10        LINK TO DSR ROUTINE.
                52   *
                53   *    EQUATES FOR PERIPHERAL ACCESS BOCK
                54   *
 000A           55   PABLEN EQU  >0A        PAB LENGTH. STANDARD
```

```
0000   SCREEN EQU >00      SCREEN OFFSET FOR PAB
003C   MAXLEN EQU 60       MAXIMUM LENGTH OF FILE NAME
1000   PAB1   EQU >1000    PAB ADDRESS USED FOR ALL FILE TRANSIT
1080   BUF1   EQU >1080    BUFFER FOR PAB1
1100   PAB2   EQU >1100    Second PAB
1180   BUF2   EQU >1180    Second BUFFER
1200   PAB3   EQU >1200    Third PAB
1280   BUF3   EQU >1280    Third BUFFER
1300   PAB4   EQU >1300    Fourth PAB for COPY statement
1380   BUF4   EQU >1380    Buffer also used to load progrm file
       *
       *    CPU RAM EQUATES
       *
0000   TEMP   EQU >00      Temporary storage area
0002   TEMP2  EQU >02
0003   TEMP3  EQU >03
0004   TEMP4  EQU >04
       *
0010   OSR    EQU >10      6 bytes temp loc for general load ro
0018   CHRBUF EQU >18      Char buffer for blinking cursor
       *
001C   SREF   EQU >1C      Pointing to the PAB start address.
0001   FLG    EQU 01       Flag and status byte
0002   BUF    EQU 02       Buffer address
0004   LEN    EQU 04       Length of record
0005   CHRCNT EQU 05       Character count
0006   RNM    EQU 06       Record number
0008   SCR    EQU 08       SCREEN OFFSET
0009   NLEN   EQU 09       Name length
       *
0020   LOC    EQU >20      Location of a cursor
0022   ERRCOD EQU >22      Error IO code
0024   STLN   EQU >24      Original cursor address
0026   ENDLN  EQU >26      Adress of the end of input
       *
0048   FLAG   EQU >48      Flag bytes
0000   INSRT  EQU 0        Bit 0 for insert mode.
0000   VFLG   EQU 0        Bit 0 for peek or poke to VDP RAM
0001   TP     EQU 1        Bit 1 on = Thermal printer is used
0002   NOTHIN EQU 2        BIT 2 ON = NO INPUT LINE IS ACCEPTED
0003   BSC    EQU 3        Bit 3 on = Called from BASIC program
       *COMP  EQU 4        Bit for compressed source file
0005   ASRC   EQU 5        Bit 5 on = Source file is specified
0006   AOBJ   EQU 6        Bit 6 on = Object file is specified
0007   ALST   EQU 7        Bit 7 on = list file is specified
       *
0049   FLAG2  EQU >49
0000   LD     EQU 0        Bit 0 on = loader is being executed
0001   VAR    EQU 1        Bit 1 on = save with var 80 format
       *EDT   EQU 2         Bit 2 on = Error in editor
0003   OTHR   EQU 3        Bit 3 on = Flag for "other" program
       *ETOP  EQU 4         BIT 4 ON = EDITOR MEMORY WRITTEN OV
0005   GPL    EQU 5        Bit 5 on = GPL program is called fro
0006   LDUTIL EQU 6        Bit 6 on = first load during LOAD &
       *
```

```
004A   111  FAC    EQU  >4A
004C   112  FAC2   EQU  FAC+2
004E   113  FAC4   EQU  FAC+4
0050   114  FAC6   EQU  FAC+6
0054   115  FAC10  EQU  FAC+10
0056   116  FAC12  EQU  FAC+12
005C   117  ARG    EQU  >5C          Used in conversion of int to float
       118  *
       119  *      STANDARD EQUATES FOR GPL INTERPRETER
       120  *
0070   121  MEMLEN EQU  >70
0073   122  SUBSTK EQU  >73          Subroutine stack pointer.
0074   123  KEYBRD EQU  >74          Keyboard number to scan.
0075   124  KEY    EQU  >75
000D   125  ETR    EQU  >0D          Key code for enter.
000F   126  BACK   EQU  >0F          Key code for going back.
0076   127  JOYY   EQU  >76
0077   128  JOYX   EQU  >77
0078   129  RANDOM EQU  >78
0079   130  TIMER  EQU  >79
007A   131  MPC    EQU  >7A
007C   132  STATUS EQU  >7C
       133  *
       134  ******
       135  *
       136         GROM   3
       137         GRG    0
6000 AA   138      DATA   >AA
6001 010100 139    DATA   1,1,0,#0,#PROG
6004 000060
6007 10
6008 00006A 140    DATA   #0,#LINK01,#0,#0
6008 520000
600E 0000
6010 000060 141    PROG   DATA   #0,#BEGIN
6013 25
6014 104544 142    DATA   16,:EDITOR/ASSEMBLER:
6017 49544F
601A 522F41
601D 535345
6020 404240
6023 4552
       143  ******
       144  ***
       145  ******
       146  ****
       147  ***************************************************
       148  *      USER INTERFACE                             *
       149  ***************************************************
       150  ***************************************************
       151  *      BEGIN -- First entry point                 *
       152  ***************************************************
       153  BEGIN
6025 878F9D 154.   DCLR @IDCODE        CLEAR ALL FILES IN MEMORY
6028 00
```

```
                 155  ****
                 156  *      Initilization section
                 157  *      Color >F5 (color table 0 through 12), blue screen
                 158  *      Define VDP registers, load character sets,
                 159  *      No motion and no sprites
                 160  ****
                 161  *
                 162  BEGIN2
6029 066525      163          CALL COLOR              Load color table
602C 8E7405      164          ST   S.@KEYBRD          Scan keyboard S.
602F 03          165          SCAN                    Enable keyboard for /4A
6030 8E7440      166          $IF @KEYBRD .EQ. 0 THEN  MUST BE 4A KEYBOARD      1
6033 3D
6034 BF4A0B      167            DST  >B00.@FAC                                  1
6037 00
6038 06004A      168            CALL CHAR3$           Load lower case character 1
603B 4056        169          $ELSE                                             1
603D BF4A0A      170            DST >A00.@FAC                                   1
6040 00
6041 060018      171            CALL CHAR2$           Load semi-lower character 1
6044 BF4A0B      172            DST >B00.@FAC         First character definition 1
6047 00
6048 8EB04A      173  CHRLP     ST   >C0.RAM(@FAC)    Place two dots for lower case 1
604B C0
604C A34A00      174            DADD 8.@FAC                                     1
604F 08
6050 CB4A0C      175            $IF @FAC .DL. >C00 GOTO CHRLP  Fix next character 1
6053 004048
                 176          $END IF
6056 8674        177          CLR  @KEYBRD            Scan standard keyboard
                 178  *
6058 BF4A09      179          DST  >900,@FAC          Load small characters.
605B 00
605C 060018      180          CALL CHAR2$
605F 86A800      181          CLR  RAM(>800)          Clean up TI logo stuff.
6062 3500FF      182          MOVE >FF FROM RAM(>800) TO RAM(>801)
6065 A801A8
6068 00
                 183  *
                 184  *
6069 390007      185          MOVE 7 FROM ROM(#VDPRG2) TO VDP(1) SET UP VDP REGIST
606C 016A4B
                 186  *       CHCEK IF ERAM EXISTS

***** ERROR IN IMMEDIATE FIELD  LINE=  187  COLUMN= 17

***** ERROR IN BRANCH ADDRESS  LINE=  187
606F 000000      187          CALL BUG1               TEST IF ERAM EXISTS.
                 188  *
6072 310010      189          MOVE 16 FROM ROM(#CDATA) TO CHAR(>1E) LOAD CURSOR CHA
6075 A8F06A
6078 3B
                 190  *
                 191  ************************************************************
                 192  *       START -- Second entry point                        *
                 193  ************************************************************
                 194  START
```

```
                 195  ****
                 196  *****
                 197  *       Initial screen to select editor, assembler, loader
                 198  ****
6079 BE737E      199          ST   >7E,@SUBSTK        Adjust subroutine stack pointer
607C 8722        200          DCLR @ERRCOD            Clear error code
607E 8748        201          DCLR @FLAG              Clear all flags
6080 0720        203          ALL  SPACE
                 204          FMT  XPT=2,YPT=1,':* EDITOR/ASSEMBLER * :';
                 205               2^,11<,':PRESS::',1^,27<;
                 206               ':1 TO EDIT:',1^,23<;
                 207               ':2   ASSEMBLE:',1^,19<;
                 208               ':3   LOAD AND RUN:',1^,15<;
                 209               ':4   RUN:',24<,1^;
                 210               ':5   RUN PROGRAM FILE:',6^,11<;
6082 06FF02      211               '>A',':1981  TEXAS INSTRUMENTS:'
6116 03          213  SCAN10 SCAN
6117 4116        214          BR   SCAN10
6119 D675OF      215          $IF @KEY .EQ. BACK THEN                                1
611C 411F
611E 08          216            EXIT                                                 1
                 217          $END
611F A67531      218          SUB  >31,@KEY                    Convert from ASCII
6122 CA7505      219          $IF @KEY .HE. 5 GOTO SCAN10      Check input range
6125 6116
6127 8A75        220          CASE @KEY
6129 4133        221          BR   EDIT                        Load a file
612B 4632        222          BR   ASSEM                       Save a file

LAST ERROR FOUND AT LINE 187

***** ERROR IN IMMEDIATE FIELD  LINE= 223  COLUMN= 17

***** ERROR IN BRANCH ADDRESS  LINE= 223
6120 0000        223          BR   LOAD                        Temp edit address

LAST ERROR FOUND AT LINE 223

***** ERROR IN IMMEDIATE FIELD  LINE= 224  COLUMN= 16

***** ERROR IN BRANCH ADDRESS  LINE= 224
612F 0000        224          BR   RUN                         assemble a file
6131 4557        225          BR   OTHER                       Run other program
```

```
                 227  *
                 228  ***********************************************************
                 229  *        SCREEN FOR EDITOR                                *
                 230  ***********************************************************
                 231  ****
                 232  EDIT
6133 056525      234          CALL COLOR          Reload color table
6136 8748        235          DCLR @FLAG
                 236          FMT  XPT=2,YPT=2,':* EDITOR *:',1^,22<;
                 237               ':PRESS:::',1^,26<;
                 238               ':1 TO LOAD:',1^,23<;
                 239               ':2    EDIT:',1^,23<;
                 240               ':3    SAVE:',1^,23<;
                 241               ':4   PRINT:',1^,22<;
6138 08FF02      242               ':5   PURGE:',1^,22<
6192 8748        244          DCLR  @FLAG              Clear all flags
6194 BE737E      245          ST    >7E,@SUBSTK        Reset subroutine stack
6197 BF2000      246          DST   >C2,@LOC           Put a marker.
619A C2
                 247  *
619B 03          248  SCAN70 SCAN
619C 4198        249          BR   SCAN70
619E 067500      250          $IF @KEY .EQ. BACK GOTO START
61A1 6079
61A3 A67531      251          SUB  >31,@KEY
61A6 CA7505      252          $IF @KEY .NE. 5 GOTO SCAN70
61A9 619B
                 253  *                                   Put a marker in the right place
61AB BC0075      254          ST   @KEY,@TEMP
61AE AA0040      255          MUL  >40,@TEMP
61B1 A12000      256          DADD @TEMP,@LOC
61B4 BEB020      257          ST   CURSOR.RAM(@LOC)
61B7 1F
                 258  *
61B8 BC0475      259          ST   @KEY,@TEMP4          Keep the key information
                 260  ***********************************************************
                 261  *        CODE FOR PURGE(INSTRUCTION)                      *
                 262  ***********************************************************
61BB 067504      263          $IF @KEY .EQ. 4 THEN                                1
61BE 41E4
61C0 08A112      264            FMT  2^,':ARE YOU SURE(Y/N)? ',                   1
61C3 415245
61C6 20594F
61C9 552053
61CC 555245
61CF 28592F
61D2 4E293F
61D5 20F8
61D7 06679F      265            CALL YORN                                         1
61DA 067559      266            $IF @KEY .EQ. :Y: THEN                            2
61DD 41E2
61DF 066549      267              CALL CLRTOP          Clear top of memory for edit
                 268            $END IF
61E2 4133        269            BR  EDIT                                          1
                 270          $END IF
```

```
                271  ***************************************************
                272  *       Take care of SAVE, PRINT, EDIT            *
                273  ***************************************************
61E4 D67503     274          $IF @KEY .EQ. 3 GOTO OPEN  IF PRINT, DO NOT LOAD EDIT
61E7 61FC
                275  *
                276  *   Load editor
                277  *       Check if editor is already there(temporary)
61E9 078F90     278          $IF @IDCODE .DEQ. >55AA GOTO OPEN
61EC 0055AA
61EF 61FC
                279  *       LOAD EDITOR
61F1 0665B2     280          CALL DOWNLD       CALL DOWN LOADER
61F4 6623       281          DATA #EDNAME
61F6 066504     282          CALL DOWN10       LOAD IT DOWN.
61F9 066549     283          CALL CLRTOP             CLEAR EDITOR INFORMATION
                284  *
                285  *
                286  OPEN
61FC D60401     287          $IF @TEMP4 .EQ. 1 GOTO EEDIT GO TO EDITOR
61FF 6265
                288  *       CHECK FOR VARIABLE SO IN CASE OF SAVE
6201 060402     289          $IF @TEMP4 .EQ. 2 THEN          SAVE FILE          1
6204 4237
6206 08FF02     290            FMT XPT=2,YPT=18,':VAR SO FORMAT(Y/N)? :'          1
6209 FE1213
620C 564152
620F 203830
6212 20464F
6215 524041
6218 542859
621B 2F4E29
621E 3F20FB
6221 06679F     291            CALL YORN        GET YES OR NO ANSWER             1
6224 D67559     292            $IF @KEY .EQ. :Y: THEN                            2
6227 422C
6229 B64902     293              SB @FLAG2,VAR    SET VARIABLE FORMAT FLAG        2
                294            $END IF
622C D6750F     295            $IF @KEY .EQ. BACK GOTO EDIT  GO BACK TO EDIT SCREE
622F 6133
6231 BF2002     296            DST >282,@LOC      ADJUST THE CURSOR               1
6234 82
6235 4238       297            $ELSE                                              1
6237 BF2002     298              DST >222,@LOC    SPECIFY CURSOR LOCATION         1
623A 22
                299          $END IF
623B 31000A     300          MOVE 10 FROM ROM(#FLN) TO RAM(@LOC)
623E B02062
6241 58
6242 A32000     301          DADD >20,@LOC        CURSOR FOR FILE NAME.
6245 20
6246 BF1C10     302          DST  PAB1,@SREF
6249 00
624A 0663A0     303          CALL DSR1            Get the file name and PAB
                304  *       PAB is all cleared, so it is in open mode.
```

```
6240 D60402   305          $IF @TEMP4 .EQ. 2 GOTO ESAVE
6250 6285
6252 D60403   306          $IF @TEMP4 .EQ. 3 GOTO PRINT
6255 6209
6257 8E0462   307          $IF @TEMP4 .EQ. 0 GOTO ELOAD
625A 72
              308  ***
625B 46494C   310  FLN     DATA :FILE NAME?:
              312  *****
```

```
                 314  ****
                 315  ************************************************************
                 316  *       EDITOR - should be changed later                   *
                 317  ************************************************************
                 318  EEDIT
6265 0F23        319          XML  EDITX  Execute editor
6267 67C4        320          BS   EMSG1  Error from editor
6269 390001      321          MOVE 1 FROM ROM(#ON) TO VDP(1)
626C 016271
626F 4133        322          BR   EDIT
6271 E0          323  ON      DATA >E0
                 324  *
                 325  ************************************************************
                 326  *       ELOAD ---- routine to load a file                  *
                 327  ************************************************************
                 328  ELOAD
6272 06627B      329          CALL FIXVAR      Try to open in fix and var foramt
                 330  *
                 331  ************************************************************
                 332  *       LOADER FOR SOURCE FILES                            *
                 333  ************************************************************
                 334  ***
                 335  * Note:  End of file will be checked when I/O error happens
                 336  *        while reading a record passed end-of-file.
                 337  *        See ERRIO section to see how load gets finished.
                 338  ****
                 339  READ
                 340  *       FIXVAR routine places 02 in PAB area for read.
                 341  *       ST   02,RAM(@SREF)  Ready to read a record.
6275 0F21        342          XML  LOADX  XML to load routine in editor
6277 67C7        343          BS   EMSG   Loader error
6279 4133        344          BR   EDIT   Finished...go back
                 345  ****
                 346  **************
                 347  * Subroutine to try to open in fix or var 80 format
                 348  **************
                 349  FIXVAR
627B BEE001      350          ST   &00000100,RAM(FLG(SREF))  Input mode.
627E 1C04
6280 B249FD      351          RB   @FLAG2,VAR      Fix format flag is on.
                 352  LOAD2
6283 B0561C      353          DST  @SREF,@FAC12
6286 A35600      354          DADD PABLEN-1,@FAC12  Set pointer to name length
6289 09
628A 060010      355          CALL DSRLNK          Open it.
628D 08          356          DATA 8
628E 6871        357          BS   ERRIO           File not found.
6290 DAE001      358          CLOG >E0,RAM(FLG(SREF)) Check for error code.
6293 1CE0
6295 429C        359          BR   TRY
6297 BEB01C      360          ST   02,RAM(@SREF)   Output mode.
629A 02
629B 00          361          RTN                  No error, go back to the caller
629C BC00E0      362  TRY     ST   RAM(FLG(SREF)),@TEMP Copy flag byte
629F 011C
```

```
62A1 B2001F   363          AND  >1F,@TEMP          Delete the error code
62A4 D60004   364          $IF @TEMP .EQ. 4 THEN                          1
62A7 42B3
62A9 BEE001   365            ST  &00010100,RAM(FLG(SREF)) Try with variable reco
62AC 1C14
62AE B64902   366            SB  @FLAG2,VAR        Variable 80 format now.  1
62B1 4283     367            BR  LOAD2                                      1
              368          $END IF
62B3 4871     369          BR  ERRIO               Otherwise error!!!
              370  ******
              371  ************************************************************
              372  *       SAVE SOURCE FILE                                   *
              373  ************************************************************
              374  ESAVE
62B5 BEE001   375          ST   02,RAM(FLG(SREF))        Output mode
62B8 1C02
62BA DA4902   376          $IF .BIT(VAR) @FLAG2 .EQ. 1 THEN                1
62BD 62C4
62BF BEE001   377            ST  >12,RAM(FLG(SREF))      Var 80 output mode  1
62C2 1C12
              378          $END IF
62C4 0663E9   379          CALL DSR3                     Open it.
              380  *
62C7 BEB01C   381          ST   03,RAM(@SREF)     Store OP code for write
62CA 03
62CB BEE005   382          ST   80,RAM(CHRCNT(SREF)) Store character count.
62CE 1C50
              383  *
              384  *   Write one record at a time.
              385  *
              386  WRITE
62D0 0F22     387          XML  SAVEX            Execute SAVE in editor.
62D2 67C7     388          BS   EMSG             SAVE error
62D4 0663E5   389          CALL CLOSE            Close the file
62D7 4133     390          BR   EDIT             Go back to editor screen
              391  *
              392  *
              393  ************************************************************
              394  *       PRINT ROUTINE TO PRINT OUT FILES                   *
              395  ************************************************************
              396  PRINT
62D9 06627B   397          CALL FIXVAR           Open with fix or var 80 forma
              398  *       ST   02,RAM(@SREF)          Ready for read!
              399  *
              400  *       Input file is defined. Now get output device
              401  *
62DC 08FF02   403          FMT  XPT=2,YPT=21,:DEVICE NAME?:
62EF BF2002   405          DST  >2C2,@LOC
62F2 C2
62F3 BF1C11   406          DST  PAB2,@SREF             Open next file
62F6 00
62F7 0663A0   407          CALL DSR1                   Get name and prepare PAB
62FA BEE005   408          ST   80,RAM(CHRCNT(SREF)) Store charcter count
62FD 1C50
62FF D7B04E   409          $IF RAM(@FAC4) .DEQ. :TP: THEN Check which TP   1
```

```
6302 545043
6305 0A
6306 86E004   410          CLR  RAM(LEN(SREF)) Change logical record length  1
6309 1C
              411        $END IF
630A BEE001   412        ST  >12,RAM(FLG(SREF)) Open as variable 80 dsiplay.
630D 1C12
630F BFE002   413        DST  BUF1,RAM(BUF(SREF)) Forse to use same buffer are
6312 1C1080
6315 0663E9   414        CALL DSR3              Open it!
6318 D6E004   415        $IF RAM(LEN(SREF)) .EQ. >20 THEN Old TP!!!!          1
631B 1C2043
631E 22
631F B64802   416          SB  @FLAG,TP         Set a flag                    1
              417        $END IF
6322 BEB01C   418        ST   03,RAM(@SREF)     Ready for print!
6325 03
              419  *
6326 DA4802   420        $IF .BIT(TP) @FLAG .EQ. 0 GOTO FAST   RS232 case.
6329 638D
              421  PRINT1
632B BF1C10   422        DST  PAB1,@SREF        Now get first file
632E 00
632F 0663D0   423        CALL CLRBUF            Clean up buffer area
6332 0663E9   424        CALL DSR3              Read one record
6335 BF1C11   425        DST  PAB2,@SREF        Prepare for print !!!!
6338 00
              426  *
              427  *     Next routine checks the buffer and replace characters
              428  *
6339 BF0010   429        DST  BUF1,@TEMP        Check buffer area for CC!
633C 80
633D A30000   430        DADD 79,@TEMP          Check from end of record
6340 4F
6341 BE0250   431        ST   80,@TEMP2         Set the counter
              432  LOOP
6344 BC03B0   433        ST   RAM(@TEMP),@TEMP3 Get character code.
6347 00
6348 D60320   434        $IF @TEMP3 .EQ. SPACE GOTO MORE Skip space
634B 6361
634D D6030C   435        $IF @TEMP3 .EQ. >0C THEN  If page feed skip lines.   1
6350 435F
6352 BEB000   436          ST   SPACE,RAM(@TEMP)  Put space in.                1
6355 20
6356 0663E9   437          CALL DSR3            Just skip a few lines.          1
6359 0663E9   438          CALL DSR3                                            1
635C 0663E9   439          CALL DSR3                                            1
              440        $END IF
635F 4367     441        BR   PRINT2            No need to check. print out.
              442  MORE
6361 9300     443        DDEC @TEMP             Decrement the buffer pointer
6363 9202     444        DEC  @TEMP2            Decrement counter
6365 4344     445        BR   LOOP              Not passed the last loc.
              446  **********
              447  *     DST  PAB2,@SREF        Already specified.
```

```
                 448  PRINT2
6367 0663E9      449          CALL DSR3                  Call to write one line
                 450  *
                 451  *   The following section takes care of Thermal Printer case
                 452  *
636A A60220      453          SUB >20,@TEMP2             Find how many more to move
636D CE0200      454          $IF @TEMP2 .GT. 0 THEN     More bytes to print?      1
6370 4380
6372 A3E002      455            DADD >20,RAM(BUF(SREF))  Shift buffer address      1
6375 1C0020
6378 BCE005      456            ST   @TEMP2,RAM(CHRCNT(SREF)) Put count            1
637B 1C02
637D 056367      457            B    PRINT2              Print one more line       1
                 458          $END IF
6380 BFE002      459          DST BUF1,RAM(BUF(SREF))    Readjust pointer
6383 1C1080
6386 BEE005      460          ST  >20,RAM(CHRCNT(SREF))  Readjust character count
6389 1C20
638B 432B        461          BR   PRINT1                Print more line
                 462  *****
                 463  ****
                 464  FAST
6380 BF1C10      465          DST  PAB1,@SREF            Read a record
6390 00
6391 066300      466          CALL CLRBUF                Cean up buffer
6394 0663E9      467          CALL DSR3
6397 BF1C11      468          DST  PAB2,@SREF            Print it out
639A 00
639B 0663E9      469          CALL DSR3
639E 4380        470          BR   FAST                  Repeat the process
                 471  ***********************************************************
                 472  *        DSR PREPARATION AND CALL ROUTINES                *
                 473  ***********************************************************
                 474  DSR1
63A0 066300      475          CALL CLRBUF                Clean up the PAB buffer area
                 476  DSR2
63A3 0663FE      477          CALL CLRPAB                Clean up PAB area
63A6 B0E002      478          DST  @SREF,RAM(BUF(SREF))  Find buffer address
63A9 1C1C
63AB A3E002      479          DADD >80,RAM(BUF(SREF))    Buffer address
63AE 1C0080
63B1 BEE008      480          ST   SCREEN,RAM(SCR(SREF)) Screen offset
63B4 1C00
63B6 BFE004      481          DST  >5000,RAM(LEN(SREF))  Record length
63B9 1C5000
63BC 06640A      482          CALL NAME                  Get the name
63BF 8F5063      483        $IF @FAC6 .DNE. 0 THEN       Do not move 0 bytes.      1
63C2 CF
                 484  COPYNM
63C3 3450E0      485          MOVE @FAC6 FROM RAM(@FAC4) TO RAM(PABLEN(SREF)) Name1
63C6 0A1CB0
63C9 4E
63CA BCE009      486          ST   @FAC6+1,RAM(NLEN(SREF)) Put name length in PAB 1
63CD 1C51
                 487        $END IF
```

```
63CF 00          488        RTN
                 489  ****
                 490  ****
                 491  CLRBUF
63D0 A31C00      492        DADD >80.@SREF              @SREF= buffer address
63D3 80
63D4 BEB01C      493        ST   SPACE.RAM(@SREF)       Clean up the PAB buffer.
63D7 20
63D8 35004F      494        MOVE 79 FROM RAM(@SREF) TO RAM(1(SREF))
63DB E0011C
63DE B01C
63E0 A71C00      495        DSUB >80.@SREF              @SREF= PAB address
63E3 80
63E4 00          496        RTN
                 497  ******
                 498  ************************************************************
                 499  *     CLOSE A FILE                                         *
                 500  ************************************************************
                 501  *
                 502  CLOSE
63E5 BEB01C      503        ST   1.RAM(@SREF)           Store close op code.
63E8 01
                 504  DSR3
63E9 BD561C      505        DST  @SREF.@FAC12
63EC A35600      506        DADD PABLEN-1.@FAC12          Pointer to name length.
63EF 09
63F0 060010      507        CALL DSRLNK                 Call DSR routine
63F3 08          508        DATA 8
63F4 4871        509        BS   ERRIO                  File not found
63F6 DAE001      510        CLOG >E0.RAM(FLG(SREF))     Any other error?
63F9 1CE0
63FB 4871        511        BR   ERRIO
63FD 00          512        RTN
                 513  *
                 514  *
                 515  ****
                 516  CLRPAB
63FE 86B01C      517        CLR  RAM(@SREF)
6401 350045      518        MOVE MAXLEN+9 FROM RAM(@SREF) TO RAM(1(SREF)) Clear P
6404 E0011C
6407 B01C
6409 00          519        RTN
                 520  ******
                 521  ****   Get file name from screen
                 522  *      FAC4 = points to file name in VDP
                 523  *      FAC6 = 2 bytes name length
                 524  *
                 525  NAME
640A 06643D      526        CALL SCANSO                 Get user input.. file name
640D B02024      527        DST  @STLN.@LOC             Relocate to scan file name.
6410 8E003C      528        ST   MAXLEN.@TEMP           Set the counter.
6413 8750        529        DCLR @FAC6                  Clear length of file name.
                 530  SKIPSP
6415 D6B020      531        $IF RAM(@LOC) .EQ. SPACE THEN   Skip spaces          1
6418 204428
```

```
641B 9120     532          DINC  @LOC          Check next location.          1
641D 9200     533          DEC   @TEMP         Decrement counter.            1
641F 4415     534          BR    SKIPSP        Not finished scanning.        1
6421 B02024   535          DST   @STLN,@LOC    Input file name again.        1
6424 0A4804   536          $IF .BIT(NOTHIN) @FLAG .EQ. 1 GOTO NAMEND  No input
6427 443C
6429 440A     537          BR    NAME          No file name on the line.     1
              538        $END IF
642B B04E20   539        DST  @LOC,@FAC4       Starting address of file name.
              540  COUNT
642E D6B020   541        $IF RAM(@LOC) .NE. SPACE THEN                       1
6431 20643C
6434 9150     542          DINC @FAC6          Increment name length.        1
6436 9120     543          DINC @LOC           Look further.                 1
6438 9200     544          DEC  @TEMP          Decrement counter.            1
643A 442E     545          BR   COUNT          Check more in the line.       1
              546        $END IF
              547  NAMEND
643C 00       548        RTN
```

```
                 550  *******
                 551  ***    SCAN ROUTINE TO INPUT FILE NAME
                 552  ******
                 553  SCANS0
6430 BE181F      554          ST   CURSOR,@CHRBUF     Load cursor character
6440 BD2420      555          DST  @LOC,@STLN         Copy start address
6443 BD2620      556          DST  @LOC,@ENDLN        Copy end address
                 557  SCANS1
6446 8679        558          CLR  @TIMER             Set timer
6448 C01880      559          EX   RAM(@LOC),@CHRBUF  Alternate chars to blink
644B 20
                 560          $REPEAT
644C 03          561            SCAN                                          1
644D 6456        562            BS  GETKEY              Scan uitil key is reconcnised
644F CA790F      563          $UNTIL @TIMER .HE. 15     Wait for 6/60 of a,second
6452 444C
6454 4446        564          BR   SCANS1
                 565  GETKEY
6456 D68020      566          $IF RAM(@LOC) .EQ. CURSOR THEN  Key is detected     1
6459 1F4460
645C C01880      567            EX  RAM(@LOC),@CHRBUF         Load cursor char back
645F 20
                 568          $END IF
6460 B00020      569          DST  @LOC,@TEMP                 Find length to the cu
6463 A50024      570          DSUB @STLN,@TEMP
6466 CA7520      571          $IF @KEY .L. >20 GOTO CONTO2    Control keys
6469 4485
646B DA4801      572          TBR  @FLAG,INSRT                Check for insert mode
646E 44D0        573          BR   INS04                      Insert a character
                 574  DSP
6470 BC8020      575          ST   @KEY,RAM(@LOC)             Validate the key
6473 75
6474 C52026      576          $IF @LOC .DH. @ENDLN THEN                           1
6477 447C
6479 BD2620      577            DST  @LOC,@ENDLN        Adjust end of line        1
                 578          $END IF
                 579  ********
                 580  *       Cursor right
                 581  *******
                 582  RGT
647C C6013C      583          $IF @TEMP+1 .H. >MAXLEN GOTO SCANS1 No more to the ri
647F 6446
6481 9120        584          DINC @LOC               Shift cursor to the right
6483 4446        585          BR   SCANS1             Go back to scan more
                 586  ******
                 587  *       Handle control keys
                 588  ******
                 589  CONTO2
6485 B248FE      590          RB   @FLAG,INSRT        Reset insert flag
6488 D6750F      591          $IF @KEY .EQ. BACK THEN         *** Shift-Z         1
648B 4499
6480 DA4820      592            $IF .BIT(ASRC) @FLAG .EQ. 1 GOTO ERRIO2  From asemb
6490 4893
6492 DA4804      593            $IF .BIT(NOTHIN) @FLAG .EQ. 1 GOTO START From execl
6495 4079
```

```
6497 4133    594          BR   EDIT                                          1
             595        $END IF
6499 D67509  596        $IF @KEY .EQ. >09 GOTO RGT        *** Cursor right.
649C 647C
649E D67508  597        $IF @KEY .EQ. >08 THEN            *** Cusor left.    1
64A1 44AB
64A3 8E0164  598          $IF @TEMP+1 .EQ. 0 GOTO SCANS1 Can't go left any mo
64A6 46
64A7 9320    599          DDEC @LOC                       Move cursor to the le
64A9 4446    600          BR   SCANS1                     Go back to scan more1.
             601        $END IF
64AB D67500  602        $IF @KEY .EQ. >0D GOTO SCNEND     *** Enter line
64AE 6503
64B0 D67503  603        $IF @KEY .EQ. >03 THEN            *** Delete key     1
64B3 44D3
64B5 BEB020  604          ST  SPACE.RAM(@LOC)             Store blank        1
64B8 20
64B9 B00026  605          DST @ENDLN.@TEMP                Find how many bytes t
64BC A50020  606          DSUB @LOC.@TEMP                                    1
64BF CE0100  607          $IF @TEMP+1 .LE. 0 GOTO SCANS1 Nothing to the right
64C2 4446
64C4 340080  608          MOVE @TEMP FROM RAM(1(LOC)) TO RAM(@LOC) Move rest1
64C7 20E001
64CA 20
64CB BEB026  609          ST   SPACE,RAM(@ENDLN)          Delete last character
64CE 20
64CF 9326    610          DDEC @ENDLN                     Adjust end of line 1
64D1 4446    611          BR   SCANS1                     Go back to scan again
             612        $END IF
64D3 D67504  613        $IF @KEY .EQ. >04 THEN            *** Insert key     1
64D6 4501
64D8 B64801  614          SB @FLAG,INSRT                  Turn on flag       1
64DB 4446    615          BR SCANS1                       Go back to scan    1
             616  INS04
64DD B00026  617          DST  @ENDLN,@TEMP               Two lines filled?  1
64E0 A50024  618          DSUB @STLN,@TEMP                Find length of file b
64E3 C6013C  619          $IF @TEMP+1 .H. MAXLEN GOTO SCANS1 No space to inse
64E6 6446
64E8 B00026  620          DST  @ENDLN,@TEMP               Find bytes to move. 1
64EB A50020  621          DSUB @LOC.@TEMP                                    1
64EE 9100    622          DINC @TEMP                      Number of bytes to m
64F0 3400A3  623          MOVE @TEMP FROM RAM(@LOC) TO RAM(>3C0) Move to tem
64F3 C0B020
64F6 3400E0  624          MOVE @TEMP FROM RAM(>3C0) TO RAM(1(LOC)) Move back1
64F9 0120A3
64FC C0
64FD 9126    625          DINC @ENDLN                     Adjust end of line 1
64FF 4470    626          BR DSP                          Activate the charact
             627        $END IF
6501 4446    628        BR   SCANS1                       Ignore other control
             629  SCNEND
6503 00      630        RTN                               Finish entering name.
             631  ***
```

```
                 633  ******
                 634  ******
                 635  ***
                 636  CHKRAM
6504 BC008F      637          ST   @RAMTOP,@TEMP     Save data first
6507 9000
6509 BE8F9D      638          ST   >FF,@RAMTOP       Try to write to ERAM to see it
650C 00FF
650E D68F9D      639          *IF @RAMTOP .NE. >FF GOTO ERRRAM
6511 00FF48
6514 31
6515 868F9D      640          CLR  @RAMTOP           Try again with 00 to make sure.
6518 00
6519 8E8F9D      641          *IF @RAMTOP .NE. 00 GOTO ERRRAM
651C 004831
651F BC8F9D      642          ST   @TEMP,@RAMTOP     Save it back
6522 0000
6524 00          643          RTN
                 644  *                              Load UTILITY routines to ERAM.
                 645  ****
                 646  COLOR
6525 BEA300      647          ST   >00,RAM(>300)     Stop sprites
6528 00
6529 BEA380      648          ST   >F5,RAM(>380)     LOAD COLOR TABLE
652C F5
652D 35001F      649          MOVE 31 FROM RAM(>380) TO RAM(>381)
6530 A381A3
6533 80
6534 0720        650          ALL  SPACE
6536 04F5        651          BACK >F5
6538 00          652          RTN
                 653  ****
                 654  COLOR2
6539 BEA380      655          ST   >13,RAM(>380)     Change color to green
653C 13
653D 35001F      656          MOVE 31 FROM RAM(>380) TO RAM(>381)
6540 A381A3
6543 80
6544 0720        657          ALL  SPACE
6546 04F3        658          BACK >F3
6548 00          659          RTN
                 660  ****
                 661  CLRTOP
6549 868F78      662          CLR  @EDTTOP           Clear 6 bytes of top of memory in E
654C F0
654D 350005      663          MOVE 5 FROM @EDTTOP TO @EDTTOP+1
6550 8F78F1
6553 8F78F0
6556 00          664          RTN
```

```
                 666  ****
                 667  ************************************************************
                 668  *       RUN OTHER PROGRAMS FROM DISKETTE                   *
                 669  ************************************************************
                 670  ***
                 671  OTHER
6557 0720        672          ALL  SPACE
6559 08FF02      673          FMT  XPT=2,YPT=2,':* RUN PROGRAM FILE *:'
655C FE0213
655F 2A2052
6562 554E20
6565 50524F
6568 475241
656B 402046
656E 494C45
6571 202AFB
6574 B64908      674          SB   @FLAG2.OTHR  Set bit for "other" program
                 675  *
6577 BF2001      676          DST  >102,@LOC    Put file name? on screen.
657A 02
657B 31000A      677          MOVE 10 FROM ROM(#FLN) TO RAM(@LOC)
657E B02062
6581 5B
6582 A32000      678          DADD >40,@LOC
6585 40
6586 B64804      679          SB   @FLAG,NOTHIN
6589 06640A      680          CALL NAME         Get file name input.
                 681  *
658C 8F5045      682          $IF @FAC6 .DEQ. 0 THEN Default file name UTIL     1
658F 97
6590 066582      683            CALL DOWNLD       Call down load program.        1
6593 6620        684              DATA #UTNAME      Using DSK1.UTIL1             1
6595 45A6        685            $ELSE                                            1
                 686  *
6597 0665BF      687            CALL DOWN02       Prepare PAB area               1
659A 0663C3      688            CALL COPYNM       Copy name into PAB area.       1
659D D7B04E      689            $IF RAM(@FAC4) .DEQ. :CS: THEN Cassette interface.2
65A0 435345
65A3 A6
65A4 0720        690              ALL SPACE       Clear screen for promot        2
                 691            $END IF
                 692          $END IF
65A6 066504      693          CALL DOWN10         Load it.
                 694  * Following branches to >8300(@TEMP) to execute the program
65A9 066539      695          CALL COLOR2         Color for execution time
65AC 0FF0        696          XML  >F0            Get the first address and branch
65AE 67F7        697          BS   ERRXML         In case of error

LAST ERROR FOUND AT LINE   224

***** ERROR IN IMMEDIATE FIELD  LINE=  698  COLUMN= 17

***** ERROR IN BRANCH ADDRESS  LINE=  698
65B0 0000        698          BR   EX20           End of execution code
                 699  ***
                 700  ************************************************************
                 701  *       GENERAL DOWN LOAD PROGRAM                          *
                 702  ************************************************************
                 703  *       First 6 bytes of the program file consists of :    *
```

```
                704  *        1st word:  0000 if no more file to load,          *
                705  *                   Non-zero if there is more to load      *
                706  *        2nd word:  length of the program file             *
                707  *        3rd word:  RAM address to load the program        *
                708  *   These information will be stored in CPU at >8310(@DSR) *
                709  ***********************************************************
                710  *
                711  DOWNLD
6582 8810       712          FETCH  @DSR                    Get the file name addr
6584 8811       713          FETCH  @DSR+1
6586 330005     714          MOVE 5 FROM ROM(@DSR) TO RAM(PAB1+15)    Copy name
6589 AF100F
658C 000010
                715  DOWN02
658F 8700       716          DCLR @TEMP                 Used as XML branch address
65C1 BF1C10     717          DST  PAB1,@SREF                            Get PAB addr
65C4 00
65C5 31000F     718          MOVE 15 FROM ROM(#LDDATA) TO RAM(@SREF)  Prepare PAB
65C8 B01C66
65CB 14
65CC 310014     719          MOVE 20 FROM ROM(#LODMSG) TO RAM(>202)   Display mess
65CF A20267
65D2 8B
65D3 00         720          RTN
                721  DOWN10
65D4 0663E9     722          CALL DSR3                                  Load the pro
65D7 350006     723          MOVE 6 FROM RAM(#BUF4) TO @DSR             Copy 6 bytes i
65DA 10AF13
65DD 80
65DE 8F0045     724          $IF @TEMP .DEQ. 0 THEN   Copy address of first exec &
65E1 E5
65E2 B00014     725            DST @DSR+4,@TEMP                                    1
                726          $END IF
65E5 A71483     727          DSUB OFFSET,@DSR+4                         Take away of
65E8 00
65E9 3412C0     728          MOVE @DSR+2 FROM RAM(#BUF4+6) TO @0(DSR+4) Move to RA
65EC 0014AF
65EF 1386
65F1 8F1066     729          $IF @DSR .DNE. 0 THEN    If there are more files to l
65F4 05
65F5 8702       730            DCLR @TEMP2              TENP2 = points to last char1
65F7 BC03AF     731            ST  RAM(PAB1+9),@TEMP2+1 Get the name length first
65FA 1009
65FC A30210     732            DADD PAB1+9,@TEMP2       Add PAB address            1
65FF 09
6600 908002     733            INC RAM(@TEMP2)          Change last char in name   1
6603 45D4       734            BR  DOWN10               Load more file             1
                735          $END IF
6605 BEA202     736          ST  SPACE,RAM(>202)        Clean up the screen
6608 20
6609 350013     737          MOVE 19 FROM RAM(>202) TO RAM(>203)
660C A203A2
660F 02
6610 0663FE     738          CALL CLRPAB
6613 00         739          RTN
```

```
6614 050013   740  LDDATA DATA 05,0,#BUF4,#0,#>2100,#10,:DSK1.:
6617 800000
661A 210000
661D 0A4453
6620 4B312E
6623 454449   741  EDNAME DATA :EDIT1:
6626 5431
6628 415353   742  ASNAME DATA :ASSM1:
662B 4031
662D 555449   743  UTNAME DATA :UTIL1:
6630 4C31
```

```
                745  ****** TITLE   ASSEM
                746  ***********************************************************
                747  *      July 17, 1981      Author S.Endo                    *
                748  ***********************************************************
                749  *      ASSEMBLER                                          *
                750  ***********************************************************
                751  *****
      9DD2      752  OPTLN  EQU  >2DD2-OFFSET   Option line for assembler
                753  ******
                754  ASSEM
6632 0720       755          ALL  SPACE
6634 08FF02     757          FMT  XPT=2,YPT=2,':* ASSEMBLER *:'
                759  *
6648 8748       760          DCLR @FLAG                 Clear all flags.
664A BE737E     761          ST   >7E,@SUBSTK           Reset subroutine stack.
664D B64820     762          SB   @FLAG,ASRC  Now we are in assembler
                763  *
6650 D78F9D     764          $IF @IDCODE .DEQ. >AA55 GOTO ASM10 It's already loade
6653 00AA55
6656 6687
6658 0892A0     766          FMT  19<,1^,':LOAD ASSEMBLER(Y/N)? :'
                768  *
6672 06679F     769          CALL YORN          Get response, yes or no.
6675 D6750F     770          $IF @KEY .EQ. BACK GOTO START      Back key or :n: key
6678 6079
667A D6754E     771          $IF @KEY .EQ. :N: GOTO START       will take you back to
667D 6079
                772  *****
                773  *       LOAD ASSEMBLER HERE
                774  *****
667F 0665B2     775          CALL DOWNLD                Down load assembler
6682 6628       776          DATA #ASNAME
6684 0665D4     777          CALL DOWN10                Down load it.
                778  ************
                779  *  Give additional space for assembler --- COPY needs this
                780  ************
                781  ASM10
6687 BFAF13     782          DST  >0116,RAM(BUF4)
668A 800116
668D BF5613     783          DST  BUF4,@FAC12
6690 80
6691 BE4C04     784          ST   4,@FAC2               Number of files disk DSR handles
6694 060010     785          CALL DSRLNK                Call files
6697 0A         786          DATA 10
                787  ************
                788  *  START OPENING FILES HERE
                789  ************
6698 08FF02     791          FMT  XPT=2,YPT=4,':SOURCE FILE NAME?  :'   Open a source file  ***********
669B BF1C10     793          DST  #PAB1,@SREF           Put source PAB address.
669E 00
66B9 BF2000     794          DST  >0C2,@LOC             Get loc for source file name.
66BC C2
66BD 0663A3     795          CALL DSR2                  Prepare PAB area
                796  *
66C0 BFE002     797          DST  BUF1,RAM(BUF(SREF))   Source buffer address.
```

```
66C3 1C1080
                 798  *        Next subroutine tries to open with fix and var format
                 799  *        ST   04,RAM(FLG(SREF))  Disp. fixed, seq. input
                 800  *        ST   >14,RAM(FLG(SREF)) Disp, variable, seq, input
                 801           CALL FIXVAR'                    Open it!!!
66C6 066278      802  *
                 803  ************ Open an object file.  ***********
                 805           FMT  XPT=2,YPT=8.':OBJECT FILE NAME?:'
66C9 08FF02      807           DST  #PAB2,@SREF        Put obj PAB address
66E1 BF1C11
66E4 00
66E5 BF2001      808           DST  >142,@LOC          Set file name pointer.
66E8 42
66E9 0663A3      809           CALL DSR2               Prepare PAB area
                 810  *
66EC BEE001      811           ST   00,RAM(FLG(SREF)) Open with update mode
66EF 1C00
66F1 BFE002      812           DST  BUF2,RAM(BUF(SREF)) Buffer address.
66F4 1C1180
66F7 0663E9      813           CALL DSR3               Open output file.
66FA B64840      814           SB   @FLAG,AOBJ         Flag for obj file open
                 815  *
                 816  ************ Open a list file  **********
                 818           FMT  XPT=2,YPT=12.':LIST FILE NAME?:'
66FD 08FF02      820           DST  #PAB3,@SREF        Put list PAB address
6713 BF1C12
6716 00
6717 B64804      821           SB   @FLAG,NOTHIN       No file name is OK.
671A BF2001      822           DST  >1C2,@LOC          Set pointer to name
671D C2
671E 0663A3      823           CALL DSR2               Prepare PAB area.
                 824  *
6721 8F5067      825           $IF @FAC6 .DEQ. 0 GOTO ASKOPT No list file. Skip.
6724 36
6725 BEE001      826           ST   >12,RAM(FLG(SREF)) Open with output mode
6728 1C12
672A BFE002      827           DST  BUF3,RAM(BUF(SREF)) Buffer address.
672D 1C1280
6730 0663E9      828           CALL DSR3               Open output file.
6733 B64880      829           SB   @FLAG,ALST         List file is open
                 830  *
                 831  ********** Ask for options  ***********
                 832  ASKOPT
                 833  * Clean up OPTLN beforehand ??????
6736 08FF02      834           FMT  XPT=2,YPT=16.':OPTIONS?:'
6739 FE1007
673C 4F5054
673F 494F4E
6742 533FFB
6745 BF2002      835           DST  >242,@LOC          Specify cursor location
6748 42
6749 06640A      836           CALL NAME
674C B248FB      837           RB   @FLAG,NOTHIN       No input line was OK so far.
                 838  * Option is transfered to ERAM
674F 8F5047      839           $IF @FAC6 .DEQ. 0 THEN  No option. It's OK  1
                                                                            1
6752 56
6753 BD4E20      840           DST  @LOC,@FAC4
```

```
                    841          $END IF
6756 35000F         842          MOVE 15 FROM RAM(@FAC4) TO @OPTLN   Move to RAM
6759 8F9D02
675C B04E
                    843  ***
                    844  ************  Now assemble !!!!!!  *********
                    845  ***
675E 066549         846          CALL CLRTOP        Clear editor information
6761 0720           847          ALL  SPACE
6763 0F21           848          XML  ASM           Assemble the program.
6765 67C7           849          BS   EMSG          Error routine shared with editor
                    850  *********   Close files before going back to menu   ******
                    851  *
                    852  ASMOUT
6767 066549         853          CALL CLRTOP            Clear editor information
676A BF1C10         854          DST  PAB1,@SREF        Close source file
676D 00
676E 0663E5         855          CALL CLOSE
6771 BF1C11         856          DST  PAB2,@SREF        Close object file
6774 00
6775 0663E5         857          CALL CLOSE
6778 DA4820         858          $IF .BIT(ALST) @FLAG .EQ. 1 THEN                1
677B 6784
677D BF1C12         859            DST  PAB3,@SREF      Close list file           1
6780 00
6781 0663E5         860            CALL CLOSE                                     1
                    861          $END
6784 8648           862          CLR  @FLAG
                    863  *********
6786 066938         864          CALL CONT          Issue message and continue.
6789 4079           865          BR   START         Go back to menu screen.
                    866  *******
                    867  *******
                    868  ******
                    869  *       DATA AREA
                    870  *****
678B 4F4E45         872  LODMSG DATA :ONE MOMENT PLEASE...:
```

```
                 875  ******
                 876  ****
                 877  *****
                 878  *       Get "yes" or "no" response from keyboard
                 879  *****
                 880  YORN
679F BE181F      881          ST   CURSOR,@CHRBUF
                 882  SCAN30
67A2 8679        883          CLR  @TIMER
67A4 C0187D      884          EX   CB,@CHRBUF
                 885          $REPEAT          Waiting for input.
67A7 03          886            SCAN                                    1
67A8 67B1        887            BS SCAN35                               1
67AA CA7906      888          $UNTIL @TIMER .HE. 6
67AD 47A7
67AF 47A2        889          BR   SCAN30
                 890  SCAN35
67B1 D6750F      891          $IF @KEY .EQ. BACK GOTO YNEND
67B4 67C0
67B6 D67559      892          $IF @KEY .EQ. :Y: GOTO YNEND
67B9 67C0
67BB D6754E      893          $IF @KEY .NE. :N: GOTO SCAN30
67BE 47A2
                 894  YNEND
67C0 BC7D75      895          ST   @KEY,CB      Show character on screen.
67C3 00          896          RTN
                 897  ****
                 898  ***
                 899  *
```

```
                 901  ***    TITLE   ERROR
                 902  ************************************************************
                 903  *       July 17, 1981       Author S.Endo                  *
                 904  ************************************************************
                 905  *       ERROR HANDLING ROUTINES                            *
                 906  ************************************************************
                 907  *                                                          *
                 908  *   All GPL errors are handled in this section             *
                 909  ************************************************************
                 910  *    EMSG -- error message retured from assembly language  *
                 911  *            utility program such as ASSM, EDIT, ALUTIL    *
                 912  *******
                 913  EMSG1
67C4 066525      914          CALL COLOR        Special entry for editor
                 915  EMSG
67C7 066810      916          CALL FIX          Fix screen and branch BASIC error
67CA CA2208      917          $IF @ERRCOD .L. 8 GOTO ERRIO
67CD 4871
67CF A62208      918          SUB  8,@ERRCOD
6702 C62205      919          $IF @ERRCOD .H. 5 GOTO RETURN   No error
6705 67E8
6707 0668A7      920          CALL CLSALL       Close all files. No error checks.
67DA 8A22        921          CASE @ERRCOD
67DC 4823        922          BR   ERRMEM       Code 8: Memory full
67DE 481C        923          BR   WRNCHR       Code 9: Control character
67E0 483C        924          BR   BADTAG       Code A: Bad tag
67E2 4843        925          BR   ERRCSM       Code B: Checksum error
67E4 484A        926          BR   ERRDDF       Code C: Duplicate definition
67E6 4851        927          BR   ERRURR       Code D: Unresolved references
                 928  *
                 929  RETURN
67E8 DA4820      930          $IF .BIT(ASRC) @FLAG .EQ. 1 GOTO ASMOUT
67EB 4767

LAST ERROR FOUND AT LINE  698

***** ERROR IN IMMEDIATE FIELD  LINE=  931  COLUMN= 44

***** ERROR IN ADDRESS FIELD  LINE=  931
67ED DA4901      931          $IF .BIT(LD) @FLAG2 .EQ. 1 GOTO LOAD
67F0 6067
67F2 0668C4      932          CALL CLSNO
67F5 4133        933          BR   EDIT   Go back to editor
                 934  ************************************************************
                 935  *       ERRXML -- Error returned from user's program       *
                 936  *                 or program file loaded by selection 5    *
                 937  ************************************************************
                 938  ERRXML
67F7 066810      939          CALL FIX          Fix screen and branch to BASIC error
67FA 066525      940          CALL COLOR        Reload screen
                 941  *       The following error can only be used by user executio
67FD D6220F      942          $IF @ERRCOD .EQ. >0F THEN                          1
6800 4809
6802 0668E4      943  ERRPR2   CALL ERROR       :Program not found:                1
6805 6A29        944            DATA #PRGNF                                      1
6807 4850        945            BR   ERRNM2     Clear msg. Go back for new input. 1
                 946          $END IF
6809 0668E4      947          CALL ERROR        :XML error:
680C 69BF        948          DATA #XMLERR
```

```
680E 4029      949          BR   BEGIN2      Initialize all except top memory
               950   ***
               951   *      FIX -- Fix screen and branch to BASIC errors
               952   ****
               953   FIX
6810 390001    954          MOVE 1 FROM ROM(#ON) TO VDP(1) Turn on the screen
6813 016271
6816 DA4808    955          $IF .BIT(BSC) @FLAG .EQ. 1 GOTO ERRBSC Error in BASIC
6819 409C
681B 00        956          RTN
               957   **
               958   ************************************************************
               959   *      General warning routines                            *
               960   ************************************************************
               961   WRNCHR
681C 0668D8    962          CALL WARN        :Control character removed:
681F 69CA      963          DATA #CHRWRN
6821 47E8      964          BR   RETURN
               965   *
               966   ************************************************************
               967   *      General error routines                              *
               968   ************************************************************
               969   ERRMEM
6823 0668E4    970          CALL ERROR       :memory full error:
6826 6982      971          DATA #MEM
6828 47E8      972          BR   RETURN
               973   *
               974   ERRPRG
682A 0668E4    975          CALL ERROR       :Program not found:
682D 6A29      976          DATA #PRGNF
682F 4079      977          BR   START       Go back to menu screen
               978   *                       Nothing is destroyed yet.
               979   *
               980   ERRRAM
6831 DA4808    981          $IF .BIT(BSC) @FLAG .EQ. 1 GOTO ERRRMB Go back to BAS
6834 4E57
6836 0668E4    982          CALL ERROR       :no memory expansion:
6839 6990      983          DATA #RAMERR
683B 0B        984          EXIT
               985   *
               986   BADTAG
683C 0668E4    987          CALL ERROR       :ILLEGAL TAG:
683F 69E4      988          DATA #ILLTAG

LAST ERROR FOUND AT LINE   931

***** ERROR IN IMMEDIATE FIELD  LINE=  989  COLUMN= 17

***** ERROR IN BRANCH ADDRESS  LINE=  989
6841 0000      989          BR   LOAD
               990   *
               991   ERRCSM
6843 0668E4    992          CALL ERROR       :checksum error:
6846 69F0      993          DATA #CSMERR

LAST ERROR FOUND AT LINE   989

***** ERROR IN IMMEDIATE FIELD  LINE=  994  COLUMN= 17

***** ERROR IN BRANCH ADDRESS  LINE=  994
6848 0000      994          BR   LOAD
```

```
684A 0668E4   997          CALL ERROR        :duplicate definition:
684D 69FF     998          DATA #DUPDEF

LAST ERROR FOUND AT LINE   994

***** ERROR IN IMMEDIATE FIELD  LINE=  999  COLUMN= 17

***** ERROR IN BRANCH ADDRESS  LINE=  999
684F 0000     999          BR   LOAD
              1000  *
              1001  ERRURR
6851 0668E4   1002         CALL ERROR        :unresolved references:
6854 6A14     1003         DATA #UNRREF

LAST ERROR FOUND AT LINE   999

***** ERROR IN IMMEDIATE FIELD  LINE= 1004  COLUMN= 17

***** ERROR IN BRANCH ADDRESS  LINE= 1004
6856 0000     1004         BR   LOAD
              1005  *
              1006  ERRNAM
6858 0668E4   1007         CALL ERROR        : Name too long:
6858 69B1     1008         DATA #NAM
              1009  ERRNM2
685D 8F5068   1010         $IF @FAC6 .DEQ. 0 GOTO ERRNM3
6860 6C
6861 BEB04E   1011         ST   SPACE,RAM(@FAC4)        Delete name on screen
6864 20
6865 3450E0   1012         MOVE @FAC6 FROM RAM(@FAC4) TO RAM(1(FAC4))
6868 014EB0
686B 4E
              1013  ERRNM3
686C 06694B   1014         CALL CLNMSG                  Clean up message area

LAST ERROR FOUND AT LINE 1004

***** ERROR IN IMMEDIATE FIELD  LINE= 1015  COLUMN= 17

***** ERROR IN BRANCH ADDRESS  LINE= 1015
686F 0000     1015         BR   EXEC
              1016  ************************************************************
              1017  *        ERRIO -- section for IO errors                    *
              1018  ************************************************************
              1019  ERRIO
6871 BC4EE0   1020     ST   RAM(FLG(SREF)),@FAC4      Get error code out.
6874 011C
6876 B24EE0   1021     AND  >E0,@FAC4                 Take error code bits
6879 E64E05   1022     SRL  @FAC4,5                   Move 5 bits to right justif
687C B64E30   1023     OR   >30,@FAC4                 Add ASCII offset.
              1024  *
687F B2E001   1025     AND >1F,RAM(FLG(SREF))         Get error code out
6882 1C1F
              1026  *
              1027  *   Next line checks if error is reading after end-of -file.
              1028  *
6884 064E35   1029     $IF @FAC4 .EQ. >35 THEN    In case of just an end-of-file1
6887 488E
6889 0668A7   1030         CALL CLSALL            Close all the files          1
688C 4133     1031         BR   EDIT              Branch back to editor        1
              1032     $END IF
              1033  *
```

```
6B8E 0668E4  1034      CALL ERROR        Regular I/O error.
6891 698E    1035      DATA #IO
             1036  ERRIO2
6893 0668A7  1037      CALL  CLSALL        Close all files without checking err
             1038  ***
```