```
 0001 INT14 DOMUS MACRO ASSEMBLER REV 02.00
01        000000 TESTO=0
02                .EOT
```

```
01              ;                                       RCSL:   43-GL10317
02              ;                                       AUTHOR: MLM/JHA
03              ;                                       EDITED: 80.08.25
04
05
06
07
08
09
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31              ;               INT14
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49              ; KEYWORD:      MUS, MUSIL, INTERPRETER, LISTING.
50              ; ABSTRACT:     INTERPRETER FOR MUSIL CODE,USED IN BASIC SYSTEM.
51              ;               ASCII PAPER TAPE RCSL: 43-GL0316
52              ;               INT13 REL.BIN. PAPER TAPE RCSL: 43-GL10318.
53              ;               INT13-TESTOUTPUT REL.BIN. PAPER TAPE RCSL:
54              ;               43-GL10319.
55
56
57
58
59
```

```
10003 INT14
01                ; TESTOUTPUT ON LPT.
02                ; THE ASSEMBLY CONSTANT TESTOUT SHOULD BE DEFINED ON A
03                ; SEPERATE TAPE
04                ; TESTOUT=0, MEANS NO TESTOUTPUT.
05                ;    -    =1, MEANS TESTOUTPUT ON LPT.
06                ; ON RC3803 'INT' SHOULD BE LOADED AFTER SYSTEM INITIALIZATION
07                ; (MUI) IF THE TESTOUTPUT-VERSION IS USED.
08                ; THE LPT DRIVER SHOULD ALWAYS BE LOADED TOGETHER WITH THE INTER-
09                ; PRETER. IN CASE OF LPT ERRORS THE ACTION IS: REPETITION UNTIL
10                ; LPT IS READY. IF TESTOUTPUT IS WANTED ON ANOTHER DRIVER THE
11                ; FILE NAME LPT SHOULD BE CHANGED. THIS NAME IS FOUND AS THE
12                ; FIRST WORD OF THE LOADED MODULE. A LIST COMMAND WILL GIVE
13                ; THE ADDRESS AS CURRENT NMAX.
14                ;
15                ; SWITCH 0 SET TESTOUTPUT WILL BE PRODUCED OF EVERY I/O
16                ; COMMAND CONTAINING SOME OF THE ZONE VARIABLES.
17                ; THE FORMAT IS:
18                ;     OP MODIF ZONE ZUSED ZSHAREL ZTOP ZREM ZFIRST ZLENGTH ZFORMAT
19                ;
20                ; SWITCH 1 SET WILL PRODUCE TESTOUTPUT FOR
21                ;           GO CODEAND MOVE
22                ;           FORMAT: OPERATION MODIFICATION AND 6 MODIFWORDS (OCTAL)
23                ; SWITCH 2 SET WILL PRODUCE TESTOUTPUT FOR
24                ;           MOVESTRING, COMPARE, TRANSLATE AND CONVERT
25                ;           FORMAT: OPERATION, MODIFICATION AND 4 MODIFWORDS (OCTAL)
26                ; SWITCH 3 SET WILL PRODUCE TESTOUTPUT FOR
27                ;           OPSTATUS, BINDEC, DECBIN
28                ;           FORMAT: OPERATION, MODIFICATION AND 2 MODIFWORDS (OCTAL)
29                ; SWITCH 4 SET WILL PRODUCE TESTOUTPUT FOR
30                ;           ARITC, ARIT, JUMP, LINK, CALL, OPMESS AND OTHERS
31                ;           FORMAT: OPERATION, MODIFICATION AND 1 MODIFWORD (OCTAL)
32                ; SWITCH 5 SET WILL PRODUCE TESTOUTPUT FOR
33                ;           STOP, OPTEST AND ARITD
34                ;           FORMAT: OPERATION AND MODIFICATION (OCTAL)
35
36
37
```

```
10004 INT14
                .TITL INT14
02        000012 .RDX    10
03        000001 .TXTM   1
04               .NREL
05               ;       *****     INTERPRETER      *****
06               ;
07
08
09        000225          .CL00=   INTPRETE-GOS              ; DEFINE IN MONITORS PAGE 0
10        000225          .LOC     .CL00                     ; INTPRETE ENTRY
11  00225 000000*         CL00                               ;
12        000227          .LOC     .+1                       ;
13  00227 001117*         CL6                                ; INTGIVEUP
14        000231          .LOC     .+1                       ;
15  00231 001146*         CL7                                ; INTBREAK
16        000234          .LOC     MZSTART
17
18  00234 000004*.CL0:    CL0                                ; CENTRAL LOOP ENTRY 0.
19  00235 000005*.CL1:    CL1                                ;    -     -      -  1.
20  00236 000125*.CL4:    CL4                                ; TAKEADDRESS ENTRY.
21  00237 000145*.CL5:    CL5                                ; TAKEVALUE ENTRY.
22  00240 002241          JMP@     .+1
23  00241 001131*         CL65                               ; PINTGIVEUP
24        000240 PINTG=160
25
26               .NREL
```

```
10005 INT14
01        000000 .IFN TESTOUT
02               ; ZONE USED TO WRITE THE TESTOUTPUT ON LPT
03               ;
04               IT53:                          ; OUTPUT ZONE DESCRIPTOR:
05               .TXT   .LPT<0><0>.             ; NAME
06                      IT51-IT53               ; SIZE
07                      3                       ; MODE
08                      1                       ; KIND
09               .RDX   2
10                      1100011111111011        ; GIVEUPMASK
11               .RDX   10
12                      IT99                    ; GIVEUPACTION
13                      0                       ; FILE
14                      0                       ; BLOCK
15                      0                       ; CONVERSION
16                      IT51                    ; BUFFER
17                      IT59-IT51               ; SIZE OF BUFFER
18                      0                       ; FORMAT
19                      0                       ; LENGTH
20                      IT52                    ; FIRST
21                      IT52                    ; TOP
22                      IT51                    ; USED SHARE
23                      IT50                    ; SHARE LENGTH
24                      IT50                    ; REMAINING
25               .BLK   ZAUX                    ; AUXILIARY
26
27               IT51:                          ; OUTPUT SHARE DESCRIPTOR:
28                      0                       ; OPERATION
29                      0                       ; COUNT
30                      0                       ; ADDRESS
31                      0                       ; SPECIAL
32                      IT51                    ; NEXT SHARE
33                      0                       ; STATE
34                      IT52                    ; FIRST SHARE
35               IT50=80                        ; SHARELENGTH=80
36               IT52=.*2                       ; FIRST SHARE:
37               .BLK   IT50+1/2                ; MAKES ROOM FOR SHARE;
38               IT59:                          ; TOP OF BUFFER
39               IT99: .REPEATSHARE             ; GIVEUPACTION;
40
41
42
```

```
10006 INT14
01              ; INTERPRETER TESTOUTPUT
02              ;
03              IT0:  READS  1              ; ENTRY TO TESTOUTPUT:
04                    LDA    0   ICMAS      ;
05                    AND    0,1 SNR        ;   IF NO TESTSWITCHES SET THEN
06                    JMP        IT17       ;   RETURN;
07                    LDA@   0   PC,2       ;
08                    LDA    3   .255       ;
09                    AND    0,3            ;   AC3:=MODIFBITS=WORD EXTRACT 8;
10                    STA    3   ITMO       ;   SAVE MODIFBITS;
11                    SUBS   3,0            ;   AC0:=OPERATIONS=WORD SHIFT (-8);
12                    STA    0   ITOP       ;   SAVE OPERATION;
13
14                    LDA    3   .128       ; SWITCH 0:
15                    MOVL   1,1 SZC        ;   IF SWITCH 0 SET AND
16                    ADCZ#  0,3 SZC        ;    OPERATION>=200(OCTAL) THEN
17                    JMP        IT1        ;   BEGIN
18                    JSR        IT18       ; OUTPUT(OP,MODIF);
19                    LDA    3   PC,2       ; I/O-INSTRUCTION:
20                    LDA    0   +1,3       ;     ZONE=WORD(PC+1);
21                    STA    0   IT8        ;     SAVE ZONE;
22                    LDA    2   IT05       ;     AC2:=TESTOUTPUTZONE;
23                    JSR        IT16       ;     WRITE(TEST,DELIM,ZONE);
24                    LDA    0   ZUSED,3    ;
25                    JSR        IT16       ;     WRITE(TEST,DELIM,ZUSED.ZONE);
26                    LDA    0   ZSHAREL,3  ;
27                    JSR        IT16       ;     WRITE(TEST,DELIM,SHAREL.ZONE);
28                    LDA    0   ZTOP,3     ;
29                    JSR        IT16       ;     WRITE(TEST,DELIM,TOP.ZONE);
30                    LDA    0   ZREM,3     ;
31                    JSR        IT16       ;     WRITE(TEST,DELIM,REM.ZONE);
32                    LDA    0   ZFIRST,3   ;
33                    JSR        IT16       ;     WRITE(TEST,DELIM,FIRST.ZONE);
34                    LDA    0   ZLENGTH,3  ;
35                    JSR        IT16       ;     WRITE(TEST,DELIM,LENGTH.ZONE);
36                    LDA    0   ZFORMAT,3  ;
37                    JSR        IT16       ;     WRITE(TEST,DELIM,FORMAT.ZONE);
38                    LDA    0   Z0,3       ;
39                    JSR        IT16       ;     WRITE(TEST,DELIM,Z0.ZONE);
40                    JMP        IT15       ;   END;
41
42
43
```

```
10007 INT14
01              ; INTERPRETER TESTOUTPUT
02              ;
03              IT1:  LDA     3   IC57       ; NORMAL INSTRUCTIONS:
04                    MOVL    1,1 SNC        ; SWITCH 1:
05                    JMP         IT2        ;
06                    SUBZ    0,3 SNC        ;   IF SWITCH 1 SET AND
07                    JMP         .+4        ;   OP>=60 OR OP=52 THEN
08                    LDA     3   IC52       ;   BEGIN
09                    SUB     0,3 SZR        ;
10                    JMP         IT2        ;
11                    JSR         IT18       ;     OUTPUT(OP,MODIF);
12                    LDA     3   .6         ;
13                    STA     3   ITCOUNT    ;     ITCOUNT:=6;
14              IT12: LDA     2   CUR        ; OUTMODIFWORDS:
15                    LDA     3   PC,2       ;
16                    LDA     2   IT05       ;
17                    STA     3   ITPC       ;     ITPC:=PC;
18              IT13: ISZ         ITPC       ;     WHILE ITCOUNT>0 DO
19                    LDA@    0   ITPC       ;     BEGIN ITPC:=ITPC+1;
20                    JSR         IT16       ;       WRITE(TEST,DELIM,WORD(ITPC);
21                    DSZ         ITCOUNT    ;       ITCOUNT:=ITCOUNT-1;
22                    JMP         IT13       ;     END;
23                    JMP         IT15       ;   END;
24              IT2:  LDA     3   IC44       ; SWITCH 2:
25                    MOVL    1,1 SZC        ;
26                    SUBZ    0,3 SNC        ;   IF SWITCH 2 SET AND
27                    JMP         IT3        ;    36<=OP<=44 THEN
28                    LDA     3   IC35       ;   BEGIN
29                    SUBZ    0,3 SZC        ;
30                    JMP         IT3        ;
31                    JSR         IT18       ;     OUTPUT(OP,MODIF);
32                    LDA     3   .4         ;     ITCOUNT:=4;
33                    STA     3   ITCOUNT    ;     GOTO MODIFWORDS;
34                    JMP         IT12       ;   END;
35              IT3:  LDA     3   IC57       ; SWITCH 3:
36                    MOVL    1,1 SZC        ;   IF SWITCH 3 SET AND
37                    SUBZ    0,3 SNC        ;   45<=OP<=57 THEN
38                    JMP         IT4        ;   BEGIN
39                    LDA     3   IC44       ;
40                    SUBZ    0,3 SZC        ;
41                    JMP         IT4        ;
42                    JSR         IT18       ;     OUTPUT(OP,MODIF);
43                    LDA     3   .3         ;     ITCOUNT:=3;
44                    STA     3   ITCOUNT    ;     GOTO MODIFWORDS;
45                    JMP         IT12       ;   END;
46              IT4:  LDA     3   IC35       ; SWITCH 4:
47                    MOVL    1,1 SZC        ;   IF SWITCH 4 SET AND
48                    SUBZ    0,3 SNC        ;   11<=OP<=35 THEN
49                    JMP         IT5        ;   BEGIN
50                    LDA     3   IC10       ;
51                    SUBZ    0,3 SZC        ;
52                    JMP         IT5        ;
53                    JSR         IT18       ;      OUTPUT(OP,MODIF);
54                    SUBZL   3,3            ;     ITCOUNT:=1;
55                    STA     3   ITCOUNT    ;     GOTO MODIFWORDS;
56                    JMP         IT12       ;   END;
57
58
59
```

```
10008 INT14
01                ; INTERPRETER TESTOUTPUT
02                ;
03                IT5:  LDA     3   IC10       ; SWITCH 5:
04                      MOVL    1,1 SZC        ;   IF SWITCH 5 SET AND OP<=10 THEN
05                      SUBZ    0,3 SNC        ;   BEGIN
06                      JMP         IT17       ;     OUTPUT(OP,MODIF);
07                      JSR         IT18       ;   END;
08
09                IT15: LDA     2   IT05       ; PRINT:
10                      LDA     1   .10        ;
11                      OUTCHAR                ;
12                      LDA     1   .32        ;
13                      OUTEND                 ;
14
15                IT17: LDA     2   CUR        ; RETURN:
16                      LDA@    1   PC,2       ;
17                      JMP@        .+1        ;
18                                  CL1+2
19
20                IT16: STA     3   IT9        ; WRITE(TEST,DELIM,CHAR):
21                      STA     1   IT11       ;   SAVE SWITCHES;
22                      LDA     1   IT10       ;
23                      OUTCHAR                ;   OUTCHAR(AC1);
24                      OUTOCTAL               ;   OUTOCTAL(AC0);
25                      LDA     3   IT8        ;   GET SAVED ZONE;
26                      LDA     1   IT11       ;   GET SAVED SWITCHES;
27                      JMP@        IT9        ; JUMP BACK;
28
29                IT18: STA     3   IT19       ; OUTPUT(OP,MODIF):
30                      LDA     0   ITOP       ;   GET OPERATION;
31                      LDA     2   IT05       ;   GET OUTPUTZONE;
32                      JSR         IT16       ;   WRITE;
33                      LDA     0   ITMO       ;   GET MODIFICATION;
34                      JSR         IT16       ;   WRITE;
35                      LDA     2   CUR        ;
36                      JMP@        IT19       ; RETURN;
37
38                .RDX   2                     ;
39                ICMAS: 1111110000000000      ; SWITCH MASK;
40                .RDX   10                    ;
41                IT05: IT53                   ; TESTZONE;
42                IT8:  0                      ; SAVED ZONE;
43                IT9:  0                      ; WORK.LINK;
44                IT10: 45                     ; CHARACTER -
45                IT11: 0                      ; SAVED SWITCHES;
46                IT19: 0                      ; WORK.LINK;
47                ITMO: 0                      ; SAVED MODIFICATION;
48                ITOP: 0                      ; SAVED OPERATION;
49                ITPC: 0                      ; WORK PC;
50                ITCOUNT: 6                   ; NO OF MODIFWORDS;
51                IC52: 42                     ;
52                IC57: 47                     ;
53                IC44: 36                     ;
54                IC35: 29                     ;
55                IC10:  8                     ;
56                .ENDC
```

```
!0009 INT14
01                ; A JUMP TO THE REENTRANT INTERPRETER IS EXECUTED BY A JSR@ INTPR.
02                ; INSTR:=WORD(PC.CUR).
03                ; OPERATION:=INSTR(0:7), MODIFICATIONBITS:=INSTR(8:15).
04                ; IF OPERATION>=200(OCTAL) THEN I/O-INSTRUCTION: PROC(ZONE,...).
05                ;                              ELSE NORMAL INSTRUCTION.
06                ; EXIT TO THE ACTUEL SUBINTERPRETER WITH
07                ;   AC0=MODIFICATION BITS
08                ;   AC1=OPERATIONS BITS.
09                ;   AC2=CURENT PROCESS.
10
11
12                                                    ; ENTRY TO INTERPRETER:
13                .XREL
14
15 00000*030040 CL00:   LDA      2        CUR       ;
16 00001*055033          STA      3        PC,2      ;
17 00002*102400          SUB      0,0               ;
18 00003*041034          STA      0        OP,2      ; OP.CUR:=0:
19 00004*030040 CL0:     LDA      2        CUR       ; CENTRAL LOOP:
20                                                    ;
21                CL1:                               ; CENTRAL LOOP 1:
22        000000  .IFN     TESTOUT
23                         JMP@              .+1     ; GOTO TESTOUTPUT;
24                                           IT0     ;
25                .ENDC
26        000001  .IFE     TESTOUT
27 00005*027033          LDA@     1        PC,2      ; INST:=W(PC.CUR);
28                .ENDC
29 00006*011033          ISZ               PC,2      ; PC.CUR:=PC.CUR+1:
30 00007*020143          LDA      0        .255      ;
31 00010*123400          AND      1,0               ; MODIF:=INST(8:15);
32 00011*106700          SUBS     0,1               ; OPERATION:=INSTR(0:7);
33 00012*004401          JSR               .+1      ; AC3:=CL14;
34 00013*137000 CL14:    ADD      1,3               ; AC3:=CL14+OPERATION;
35 00014*003402          JMP@              CL15,3   ; GOTO SUBINTPR.=CL11-CL14+CL14+OP;
36
37                ;NOTE:  THE TABLE MUST BE PLACED AFTER THIS INSTRUCTION;
38                ;       THIS LOOP MUST NOT BE CHANGED WITHOUT CONSULTING 'MUI'
39                ;       WHERE THE LOOP IS OVERWRITTEN TO BE ABLE TO USE THE
40                ;       INSTRUCTION 'FETCH' IN RC3803.
```

```
!0010 INT14
01       000015*CL11=.          ; BASE OF TABLE CONTAINING SUBINTERPRETERS.
02
03 00015*001146*        CL7     ; 0     : STOP
04
05 00016*000754*        CL100   ; 1     : ANDD      DIRECT OPERATIONS
06 00017*000764*        CL101   ; 2     : LOADD
07 00020*000773*        CL102   ; 3     : +D
08 00021*001004*        CL103   ; 4     : -D
09 00022*001015*        CL104   ; 5     : SHIFTD
10 00023*001052*        CL105   ; 6     : EXTRACTD
11 00024*001065*        CL106   ; 7     : *D
12 00025*001115*        CL107   ; 10    : /D
13
14 00026*000752*        CL110   ; 11    ; ANDC      OPERATIONS ON CONSTANTS
15 00027*000762*        CL111   ; 12    : LOADC
16 00030*000771*        CL112   ; 13    : +C
17 00031*001002*        CL113   ; 14    : -C
18 00032*001013*        CL114   ; 15    : SHIFTC
19 00033*001050*        CL115   ; 16    : EXTRACTC
20 00034*001063*        CL116   ; 17    : *C
21 00035*001073*        CL117   ; 20    : /C
22
23 00036*000747*        CL120   ; 21    ; AND       OPERATIONS ON VARIABLES
24 00037*000757*        CL121   ; 22    : LOAD
25 00040*000766*        CL122   ; 23    : +
26 00041*000777*        CL123   ; 24    : -
27 00042*001010*        CL124   ; 25    : SHIFT
28 00043*001045*        CL125   ; 26    : EXTRACT
29 00044*001060*        CL126   ; 27    : *
30 00045*001070*        CL127   ; 30    : /
31
32 00046*001042*        CL903   ; 31    : LOAD NEGATIVE
33 00047*001030*        CL901   ; 32    : LOAD BYTE
34 00050*001033*        CL902   ; 33    : LOAD BYTEWORD
35 00051*000562*        CL309   ; 34    : JUMP
36 00052*000537*        CL308   ; 35    : LINK (UNEVEN) !USED IN CMP03!
37 00053*000302*        CL301   ; 36    : MOVEWORD
38 00054*000316*        CL302   ; 37    : MOVESTRING
39 00055*000000         0       ; 40    : COMPAREWORD
40 00056*000477*        CL304   ; 41    ; COMPARESTRING
41 00057*000472*        CL305   ; 42    : STORE REGISTER
42 00060*000442*        CL322   ; 43    : TRANSLATE
43 00061*000415*        CL324   ; 44    : CONVERT
44 00062*000660*        CL310   ; 45    : OPMESS
45 00063*000665*        CL311   ; 46    : OPIN
46 00064*000326*        CL312   ; 47    : OPWAIT
47 00065*000516*        CL306   ; 50    : CALL          !USED IN CMP03!
48 00066*000340*        CL314   ; 51    : OPTEST
49 00067*000307*        CL320   ; 52    : MOVE
50 00070*000675*        CL313   ; 53    : OPSTATUS
51 00071*000524*        CL307   ; 54    : BINDEC
52 00072*000727*        CL315   ; 55    : DECBIN
53 00073*000351*        CL316   ; 56    ; INSERT
54 00074*000557*        CL360   ; 57    : GOTO
55 00075*000532*        CL303   ; 60    : G0 CODE
56 00076*000735*        CL370   ; 61    : INCR(I)
57 00077*000742*        CL375   ; 62    : DECR(I)
58 00100*000630*        CL380   ; 63    : XLINK (UNEVEN)
```

```
!0011 INT14
01 00101*000630*          CL380    ; 64     : PLINK (EVEN)
02 00102*000562*          CL309    ; 65     : PJUMP (UNEVEN)
03 00103*000611*          CL382    ; 66     : PCALL (EVEN)
04 00104*000621*          CL383    ; 67     : XCALL (UNEVEN)
05 00105*000107*          CL376    ; 70     : INCR(I,NO)
06 00106*000116*          CL377    ; 71     : DECR(I,NO)
07
08                ; END OF TABLE
09
10         000002 CL15=CL11-CL14              ; MUST EQUAL 2 !!! (SEE 'MUI')
11
12
13                ; PROCEDURE INCR(INTEGER VAR; INTEGER VALUE);
14 00107*006236 CL376:  JSR@                 .CL4     ; TAKEADDR(MODIF,VAR);
15 00110*045027         STA      1           SAVE3,2  ;
16 00111*006237         JSR@                 .CL5     ; TAKEVALUE(MODIF,VALUE);
17 00112*023027         LDA@     0           SAVE3,2  ;
18 00113*123000         ADD      1,0                  ;
19 00114*043027         STA@     0           SAVE3,2  ; VAR:=VAR+VALUE;
20 00115*002235         JMP@                 .CL1     ; RETURN
21
22                ;PROCEDURE DECR(INTEGER VAR; INTEGER VALUE);
23 00116*006236 CL377:  JSR@                 .CL4     ; TAKEADDR(MODIF,VAR);
24 00117*045027         STA      1           SAVE3,2  ;
25 00120*006237         JSR@                 .CL5     ; TAKEVALUE(MODIF,VALUE);
26 00121*023027         LDA@     0           SAVE3,2  ;
27 00122*122400         SUB      1,0                  ;
28 00123*043027         STA@     0           SAVE3,2  ; VAR:=VAR-VALUE;
29 00124*002235         JMP@                 .CL1     ; RETURN;
```

```
10012 INT14
01                ; PROCEDURE TAKEADDRESS(MODIF,ADDRESS);
02                ; GET THE ADDR OF AN INTEGER OR STRING ADDRESSED BY PC AND INCR(PC).
03                ; 2 BITS MODIF: 00 ADDR=WORD(PC). !INTEGER!
04                ;    -      -  : 01 ADDR=   -      !STRING!
05                ;    -      -  : 10 ADDR=   -      !FILE!
06                ;    -      -  : 11 ADDR=   -     .ADDR=ZN(CUR+ADDR(0:7)).ZFIRST
07                ;                                      +ADDR(8:15).
08                ;        CALL                RETURN
09                ; AC0    MODIFBITS           MODIFBITS SHIFT(-2)
10                ; AC1                        ADDRESS
11                ; AC2    CUR                 CUR
12                ; AC3    LINK                DESTROYED
13                ;
14 00125*027033 CL4:    LDA@    1       PC,2    ; ADDRESS:=WORD(PC.CUR);
15 00126*011033         ISZ             PC,2    ; INCR(PC.CUR);
16 00127*101223         MOVZR   0,0     SNC     ; BITS:=MODIFBITS EXTRACT 2;
17 00130*101221         MOVZR   0,0     SKP     ; MODIFBITS:=MODIFBITS SHIFT (-2);
18 00131*101223         MOVZR   0,0     SNC     ; IF BITS<>11 THEN
19 00132*001400         JMP             +0,3    ; RETURN;
20 00133*055025         STA     3       SAVE1,2 ;
21 00134*034143         LDA     3       .255    ;
22 00135*137400         AND     1,3             ; FIELD:=ADDRESS(8:15);
23 00136*166700         SUBS    3,1             ; ADDRESS:=ZN.CUR
24 00137*133000         ADD     1,2             ;           (ADDRESS(0:7));
25 00140*031041         LDA     2       ZN,2    ; ADDRESS:=(ADDRESS.ZFIRST)
26 00141*025017         LDA     1       ZFIRST,2;           +FIELD;
27 00142*167000         ADD     3,1             ;
28 00143*030040         LDA     2       CUR     ;
29 00144*003025         JMP@            SAVE1,2 ; RETURN;
30
31                ; PROCEDURE TAKEVALUE(MODIF,VALUE)
32                ; GETS THE VALUE OF AN INTEGER.
33                ; 2 BITS MODIF: 00 VALUE=INST(PC); INC(PC) ;
34                ;    -      -  : 01 VALUE=R;
35                ;    -      -  : 10 VALUE=WORD(INST(PC)); INC(PC);
36                ;    -      -  : 11 VALUE=R;
37                ;        CALL                RETURN
38                ; AC0    MODIF               MODIF SHIFT (-2)
39                ; AC1                        VALUE
40                ; AC2    CUR                 CUR
41                ; AC3    LINK                DESTROYED
42                ;
43 00145*101222 CL5:    MOVZR   0,0     SZC     ;
44 00146*000413         JMP             CL52    ;
45 00147*101222         MOVZR   0,0     SZC     ; CASE MODIFBITS(14:15) OF
46 00150*000404         JMP             CL51    ;
47 00151*027033         LDA@    1       PC,2    ; 0: VALUE:=INST(PC.CUR):
48 00152*011033         ISZ             PC,2    ;    INCR(PC.CUR);
49 00153*001400         JMP             +0,3    ;    RETURN;
50 00154*031033 CL51:   LDA     2       PC,2    ; 2: VALUE:=WORD(INST(PC.CUR));
51 00155*027000         LDA@    1       +0,2    ;
52 00156*030040         LDA     2       CUR     ;
53 00157*011033         ISZ             PC,2    ;    INCR(PC.CUR);
54 00160*001400         JMP             +0,3    ;    RETURN;
55 00161*025032 CL52:   LDA     1       R,2     ; 1 AND 3:
56 00162*101220         MOVZR   0,0             ;    VALUE:=R.CUR
57 00163*001400         JMP             +0,3    ;    RETURN;
```

```
10013 INT14
01                ; ZONEPROCESSES;
02                ; CALL: PROCEDURE(ZONE,PARAM1,PARAM2);
03
04                ; REPEATSHARE(Z);
05 00164*004507 CL203:  JSR              CL20     ; GETZONEADDR;
06 00165*031027         LDA      2       SAVE3,2  ;
07 00166*034040         LDA      3       CUR      ;
08 00167*021013         LDA      0       ZBUFFE,2 ; PC.CUR:=SAVEPC.ZONE;
09 00170*037412         LDA@     3   PROG,3       ;
10 00171*024107         LDA      1   .1B6         ;
11 00172*137405         AND      1,3 SNR          ;
12 00173*000403         JMP          .+3          ; IF PAGED PROGRAM THEN
13 00174*006357         GETADR                    ; GETADR(SAVEPC.ZONE);
14 00175*161000         MOV      3,0              ;
15 00176*034040         LDA      3   CUR          ;
16 00177*041433         STA      0       PC,3     ;
17 00200*035014         LDA      3       ZSIZE,2  ; LINK:=SAVELINK.ZONE;
18 00201*002203         .REPEATSHARE              ; CALL: AC2=ZONE, AC3=LINK;
19
20                ; INCHAR(Z,INTEGER VARIABLE NAME);
21 00202*004471 CL202:  JSR              CL20     ; GETZONEADDR;
22 00203*037033         LDA@     3       PC,2     ;
23 00204*011033         ISZ              PC,2     ;
24 00205*025400         LDA      1       +0,3     ; AC1:=PARAM1;
25 00206*035030         LDA      3       SAVE4,2  ;
26 00207*031027         LDA      2       SAVE3,2  ; AC2:=ZONE;
27 00210*007400         JSR@             +0,3     ; EXIT TO MONITOR ENTRY;
28 00211*030040 CL212:  LDA      2       CUR      ;
29 00212*035033         LDA      3       PC,2     ;
30 00213*047777         STA@     1       -1,3     ;
31 00214*002235         JMP@             .CL1     ; RETURN;
32
```

```
10014 INT14
01              ; SUBINTERPRETERTABLE CONTINUED;
02       000215*CL21=.                          ; I/O-TABLESTART:
03 00215*000240*         CL200     ; 200        : GETREC:       AC0=ADDR
04 00216*000254*         CL201     ; 201        : PUTREC:       AC0=VAL
05 00217*000266*         CL206     ; 202        : WAITTRANSFER
06 00220*000164*         CL203     ; 203        : REPEATSHARE
07 00221*000400*         CL205     ; 204        : TRANSFER:     AC0=VAL, AC1=VAL
08 00222*000266*         CL206     ; 205        : INBLOCK
09 00223*000266*         CL206     ; 206        : OUTBLOCK
10 00224*000202*         CL202     ; 207        : INCHAR:       AC1=ADDR
11 00225*000266*         CL206     ; 210        : BACKSPACE
12 00226*000266*         CL206     ; 211        : OUTSPACE
13 00227*000260*         CL204     ; 212        : OUTCHAR:      AC1=VAL
14 00230*000266*         CL206     ; 213        : OUTNL
15 00231*000260*         CL204     ; 214        : OUTEND:       AC1=VAL
16 00232*000251*         CL207     ; 215        : OUTTEXT:      AC0=ADDR
17 00233*000254*         CL201     ; 216        : OUTOCTAL:     AC0=VAL
18 00234*000400*         CL205     ; 217        : SETPOS        AC0=VAL, AC1=VAL
19 00235*000260*         CL204     ; 220        : CLOSE:        AC1=VAL
20 00236*000254*         CL201     ; 221        : OPEN:         AC0=VAL
21 00237*000266*         CL206     ; 222        : WAITZONE
22                                              ; THE NEXT ENTRY IN THE I/O-TABLE
23                                              ; IS 346-200(OCTAL) FOR CREATEENTRY,
24                                              ; SEE REL.ADDR 146.
25                                              ; THE EMTHY TABLEELEMENTS ARE USED
26                                              ; FOR CODE.
27
28       000000 .IFN     CL21-CL11+GOS-GETREC   ; IF ADDR USED FOR MONITORENTRIES
29                       .BLK 30000             ; ARE NOT CORRECT BECAUSE OF
30              .ENDC                           ; CHANGES IN MONITOR OR INTERPRE-
31                                              ; TER, A BLOCK OF 30 K IS CREATED
32                                              ; TO INDICATE AN ERROR.
```

```
10015 INT14
01                ; GETREC(Z,VARIABLE NAME);
02  00240*004433 CL200:  JSR              CL20     ; GETREC:
03  00241*037033          LDA@     3       PC,2     ;
04  00242*011033          ISZ              PC,2     ;
05  00243*021400          LDA      0       +0,3     ;
06  00244*035030          LDA      3       SAVE4,2  ; GET OP;
07  00245*031027          LDA      2       SAVE3,2  ; GET ZONE;
08  00246*007400          JSR@             +0,3     ; EXIT TO MONITOR ENTRY;
09  00247*105000          MOV      0,1              ;
10  00250*000741          JMP              CL212    ;
11
12                ; OUTTEXT(Z,STRING VARIABLE NAME);
13  00251*004422 CL207:   JSR              CL20     ;
14  00252*006236          JSR@             .CL4     ; TAKEADDRESS(MODIF,AC1);
15  00253*000403          JMP              .+3      ;
16  00254*004417 CL201:   JSR              CL20     ;
17  00255*006237          JSR@             .CL5     ; TAKEVALUE(MODIF,AC1);
18  00256*121000          MOV      1,0             ;
19  00257*000410          JMP              CL209    ; GOTO EXECUTE;
20  00260*004413 CL204:   JSR              CL20     ;
21  00261*006237          JSR@             .CL5     ; TAKEVALUE(MODIF,AC0);
22  00262*000405          JMP              CL209    ; GOTO EXECUTE;
23  00263*004410 CL208:   JSR              CL20     ; GETZONEADDR;
24  00264*006236          JSR@             .CL4     ; TAKEADDR(MODIF,FILE);
25  00265*125221          MOVZR    1,1     SKP      ; ADDR:=FILE//2;
26  00266*004405 CL206:   JSR              CL20     ;
27
28                ; EXECUTE;
29                ;              CALL:     EXIT TO MONITOR I/O-TABLE
30                ; AC0          PARAM2    PARAM2
31                ; AC1          PARAM1    PARAM1
32                ; AC2          CUR       ZONE
33                ; AC3                    OP
34  00267*035030 CL209:   LDA      3        SAVE4,2  ; EXECUTE:
35  00270*031027          LDA      2        SAVE3,2  ;
36  00271*007400          JSR@             +0,3     ; EXIT TO MONITOR ENTRY;
37  00272*002234          JMP@             .CL0     ; RETURN AND GET CUR;
38
39                ; PROCEDURE GETZONEADDR;
40  00273*101220 CL20:    MOVZR    0,0             ;
41  00274*101220          MOVZR    0,0             ;   MODIF:=MODIF SHIFT -2;
42  00275*045030          STA      1       SAVE4,2 ;   SAVE4:=OP;
43  00276*027033          LDA@     1       PC,2    ;   AC1:=WORD(PC.CUR);
44  00277*011033          ISZ              PC,2    ;   INCR(PC.CUR);
45  00300*045027          STA      1       SAVE3,2 ;   ZONE.CUR:=ZN;
46  00301*001400          JMP              +0,3    ; RETURN;
47                ;
48                ; END ZONEPROCESSES;
```

```
10016 INT14
01                ; MOVEWORD(TO ADDR=PC,VALUE=PC+1);
02                ; TO ADDR  ,MODIFBITS(14:15)
03                ; VALUE    ,     -     (12:13)
04                CL301:                              ; MOVEWORD:
05 00302*006236           JSR@              .CL4      ;   ADDRESS:=INST(PC.CUR);
06 00303*045030           STA       1       SAVE4,2   ;
07 00304*006237           JSR@              .CL5      ;   TAKEVALUE(MODIF,R.CUR);
08 00305*047030           STA@      1       SAVE4,2   ;   WORD(ADDRESS):=VALUE;
09 00306*002235           JMP@              .CL1      ; RETURN;
10
11                ; MOVE(FROM STRING,FROMINDEX,TO STRING,TOINDEX,COUNT);
12                ; GETS A NEW MODIFBITS IN PC.CUR;
13                ; SAVE1: WORK
14                ; SAVE2: COUNT                              , MODIFBITS(6:7)
15                ; SAVE3: ADDR OF TOSTRING(TOINDEX)          ,     -     (14:15),(12:13)
16                ; SAVE4:   -  -  FROMSTRING(FROMINDEX)      ,     -     (10:11),(8:9)
17                ; SAVE5: WORK
18                CL320:                              ; MOVE:
19 00307*023033           LDA@      0       PC,2      ;   GET NEW MODIFBITS IN PC.CUR;
20 00310*011033           ISZ               PC,2      ;
21 00311*004460           JSR               CL333     ;   TAKEADDRESS AND MODIFY;
22 00312*045030           STA       1       SAVE4,2   ;   SAVE4.CUR:= VAL;
23 00313*004456           JSR               CL333     ;   TAKEADDRESS AND MODIFY;
24 00314*045027           STA       1       SAVE3,2   ;   SAVE3.CUR:=VAL;
25 00315*000402           JMP               .+2       ;
26
27                ; MOVESTRING(FROM ADDR, TO ADDR,COUNT);
28                ; SAVE2:  COUNT     , MODIFBITS(10:11)
29                ; SAVE3:  FROM ADDR,      -     (14:15)
30                ; SAVE4:  TO ADDR  ,      -     (12:13)
31                ; SAVE5:  WORK
32 00316*004471 CL302:  JSR               CL321     ; MOVESTRING:
33 00317*006237           JSR@              .CL5      ;   TAKEVALUE(SAVE,COUNT);
34 00320*045026           STA       1       SAVE2,2   ;   SAVE2.CUR:= COUNT;
35 00321*024404           LDA       1       .S2       ;
36 00322*133000           ADD       1,2               ;   PARAMADDR:= .SAVE2+(CUR);
37 00323*006224           MOVE                        ;   MOVE(PARAMADDR);
38 00324*002234           JMP@              .CL0      ; RETURN;
39 00325*000026 .S2:                        SAVE2     ;
40
```

```
10017 INT14
01                ; PROCEDURE OPWAIT(LENGTH);
02                CL312:                                  ; OPWAIT:
03 00326*011033           ISZ             PC,2          ;   INCR(PC);
04 00327*031034           LDA     2       OP,2          ;
05 00330*151005           MOV     2,2     SNR           ;   IF OP.CUR<>0 THEN
06 00331*002234           JMP*            .CL0          ;   BEGIN
07 00332*006005           WAITANSWER                    ;     WAITANSWER
08 00333*171000           MOV     3,2                   ;     LENGTH=W(PC.CUR-1):=BYTECOUNT;
09 00334*035033           LDA     3       PC,2          ;     OP.CUR:=0(=STATUS);
10 00335*047777           STA*    1       -1,3          ;
11 00336*041034           STA     0       OP,2          ;   END;
12 00337*002235           JMP*            .CL1          ; RETURN;
13                                                      ;
14                ; PROCEDURE OPTEST;
15                CL314:                                  ; OPTEST:
16 00340*035034           LDA     3       OP,2          ;   OP.CUR:=BUFFERADDR;
17 00341*035405           LDA     3       RECEIV,3;
18 00342*102400           SUB     0,0                   ;
19 00343*175122           MOVZL   3,3     SZC           ;   IF RECEIVER.BUF<0 !ANSWERED!
20 00344*100000           COM     0,0                   ;   THEN !ANSWERED! R:=1
21 00345*041032           STA     0       R,2           ;   ELSE R:=0;
22 00346*002235           JMP*            .CL1          ; RETURN;
23
24                ; SUBINTERPRETER TABLE CONTINUED:
25       000347*CL23=     .
26 00347*000254*          CL201                         ; NEWCAT
27 00350*000266*          CL206                         ; FREECAT
28       000000 .IFN      CL23-CL11+GOS-NEWCAT
29                        .BLK 30000
30                .ENDC
31
32                ; INSERT(BYTE,TO RECORD,RECORD INDEX);
33                ; SAVE1:
34                ; SAVE2: BYTE VALUE                   , MODIFBITS(14:15)
35                ; SAVE3: ADDR OF RECORD(INDEX)        ,      -     (12:13)+(10:11)
36                ; SAVE4:
37                ; SAVE5: WORK
38 00351*006237 CL316:    JSR*            .CL5          ; INSERT:
39 00352*034143           LDA     3       .255          ;   TAKEVALUE(MODIF,VALUE);
40 00353*137400           AND     1,3                   ;
41 00354*055026           STA     3       SAVE2,2       ;
42 00355*004414           JSR             CL333         ;   TAKEADDRESS AND MODIFY;
43 00356*021026           LDA     0       SAVE2,2       ;
44 00357*006175           PUTBYTE                       ;
45 00360*002235           JMP*            .CL1          ; RETURN;
```

```
10018 INT14
01                ; SUBINTERPRETER-TABLE CONTINUED:
02 00361*000000           0                          ; EMPTY ELEMENT;
03 00362*000000           0                          ;
04        000363*CL22=     .                          ; CAT-ENTRIES:
05 00363*000400*           CL205     ; 346           : CREATE ENTRY;
06 00364*000263*           CL208     ; 347           : LOOKUP ENTRY;
07 00365*000263*           CL208     ; 350           : CHANGE ENTRY;
08 00366*000266*           CL206     ; 351           : REMOVE ENTRY;
09 00367*000400*           CL205     ; 352           : INIT CAT
10 00370*000263*           CL208     ; 353           : SETENTRY;
11                ; END OF I/O-TABLE;
12
13
14        000000 .IFN      CL22-CL11+GOS-CREATE       ; IF ADDR USED FOR MONITORENTRIES
15                         .BLK 30000                 ; ARE NOT CORRECT BECAUSE OF
16                .ENDC                               ; CHANGES IN MONITOR OR INTERPRE-
17                                                    ; TER, A BLOCK OF 30 K IS CREATED
18                                                    ; TO INDICATE AN ERROR.
19
20                ; TAKEADDRES AND MODIFY;
21                ; ADDRESS           , MODIFBITS(14:15)
22                ; VALUE             ,     -    (12:13)
23                ; SAVE3,   WORK
24                ; AC1:=RESULT=ADDRESS+VALUE
25 00371*055031 CL333:  STA      3        SAVE5,2 ; TAKEADDRESS AND MODIFY:
26 00372*006236          JSR@              .CL4    ;   TAKEADDRESS(MODIF,ADDRESS);
27 00373*045027          STA      1        SAVE3,2 ;
28 00374*006237          JSR@              .CL5    ;   TAKEVALUE(MODIF,VALUE);
29 00375*035027          LDA      3        SAVE3,2 ;
30 00376*167000          ADD      3,1              ;   VAL:=ADDRESS+VALUE;
31 00377*003031          JMP@              SAVE5,2 ; RETURN TO LINK;
32
33                ; ZONEPROCESSES;
34                ; CALL: PROCEDURE(ZONE,PARAM1,PARAM2);
35
36                ; TRANSFER(Z,LENGTH,OPERATION), SETPOSITION(Z,FILE,BLOCK);
37 00400*004673 CL205:  JSR                CL20    ; GETZONEADDR;
38 00401*006237          JSR@              .CL5    ; TAKEVALUE(MODIF,PARAM1);
39 00402*045031          STA      1        SAVE5,2 ; SAVE(PARAM1);
40 00403*006237          JSR@              .CL5    ; TAKEVALUE (MODIF,PARAM2);
41 00404*121000          MOV      1,0              ; AC0:=PARAM2;
42 00405*025031          LDA      1        SAVE5,2 ; AC1:=PARAM1;
43 00406*000661          JMP               CL209   ; GOTO EXECUTE;
44
```

```
!0019 INT14
01                ; TAKEADDRESSES(ADDR1,ADDR2);
02                ; SAVE3:  ADDR1     , MODIFBITS(14:15)
03                ; SAVE4:  ADDR2     ,     -     (12:13)
04 00407*055031 CL321:  STA     3       SAVE5,2 ; TAKEADDRESSES:
05 00410*006236         JSR@            .CL4    ;   TAKEADDRESS(MODIF,SAVE4.CUR);
06 00411*045027         STA     1       SAVE3,2 ;
07 00412*006236         JSR@            .CL4    ;   TAKEADDRESS(MODIF,SAVE3.CUR);
08 00413*045030         STA     1       SAVE4,2 ;
09 00414*003031         JMP@            SAVE5,2 ; RETURN TO LINK;
10
11                ; CONVERT(FROMSTRING,TO STRING,CONVERTTABLE,COUNT).
12                ; SAVE1: ADDR OF CONVERTTABLE   , MODIFBITS(10:11)
13                ; SAVE2:  -   -  COUNT          ,     -     (8:9)
14                ; SAVE3:  -   -  FROM STRING    ,     -     (14:15)
15                ; SAVE4:  -   -  TO STRING      ,     -     (12:13)
16                ; SAVE5: WORK
17
18                ; NOTE: OVERWRITTEN BY 'MUI' !!
19 00415*004772 CL324:  JSR             CL321   ; CONVERT:
20 00416*006236         JSR@            .CL4    ;   TAKEADDRESSES(INSTRING,OUTSTRIN
21 00417*045025         STA     1       SAVE1,2 ;   TAKEADDRESS(TABLE);
22 00420*006237         JSR@            .CL5    ;   TAKEVALUE(COUNT);
23 00421*125400         INC     1,1             ;
24 00422*045026         STA     1       SAVE2,2 ;
25 00423*004414 CL325:  JSR             CL327   ; WHILE COUNT>0 DO
26 00424*004410         JSR             CL326   ; BEGIN
27 00425*025025         LDA     1       SAVE1,2 ;   GETBYTE(INCR(INSTRING),BYTE);
28 00426*107000         ADD     0,1             ;   GETBYTE(TABLE+BYTE,BYTE);
29 00427*006174         GETBYTE                 ;
30 00430*025030         LDA     1       SAVE4,2 ;   PUTBYTE(OUTSTRING,BYTE);
31 00431*006175         PUTBYTE                 ;
32 00432*011030         ISZ             SAVE4,2 ;   INCR(OUTSTRING)
33 00433*000770         JMP             CL325   ; END;
34 00434*025027 CL326:  LDA     1       SAVE3,2 ; INCR(ADDR);
35 00435*011027         ISZ             SAVE3,2 ;
36 00436*002174         .GETBYTE                ;
37 00437*015026 CL327:  DSZ             SAVE2,2 ; COUNT:=COUNT-1;
38 00440*001400         JMP             +0,3    ;
39 00441*002235         JMP@            .CL1    ; RETURN;
40
```

```
!0020 INT14
01              ; TRANSLATE(FROMBYTE,TOBYTE,CONVERTTABLE);
02              ; SAVE: TABLEADDR//2            , MODIFBITS(10:11)
03              ; SAVE1:
04              ; SAVE2:
05              ; SAVE3: ADDR OF FROMBYTE       ,     -     (14:15)
06              ; SAVE4:  -   -  TOBYTE         ,     -     (12:13)
07              ; SAVE5: WORK
08              CL322:                                  ; TRANSLATE:
09 00442*004745         JSR             CL321   ;  TAKEADDRESSES(INSTRING,OUTSTRING,
10 00443*006236         JSR@            .CL4    ;  TAKEADDRESS(TABLE);
11 00444*125220         MOVZR   1,1             ;
12 00445*045024         STA     1       SAVE,2  ;
13 00446*025027         LDA     1       SAVE3,2 ;
14 00447*006174         GETBYTE                 ;
15 00450*101300         MOVS    0,0             ;  AC0(BIT0-7)=BYTE
16 00451*041027         STA     0       SAVE3,2 ;
17 00452*035024         LDA     3       SAVE,2  ;  TABLEADDR:=TABLEADDR//2;
18 00453*021400 CL323:  LDA     0       +0,3    ;  GET TABLEBYTE:
19 00454*025401         LDA     1       +1,3    ;  VALUE(ARG):=DEFAULTVALUE;
20 00455*101005         MOV     0,0     SNR     ;  IF WORD(TABLEADDR)!ARG,VALUE!<>0
21 00456*000410         JMP             CL329   ;  BEGIN
22 00457*175400         INC     3,3             ;    TABLEADDR:=TABLEADDR+1;
23 00460*025027         LDA     1       SAVE3,2 ;    GET FROMBYTE; !FROMBYTE=ARG IN
24 00461*122400         SUB     1,0             ;    ARG:=ARG-FROMBYTE;
25 00462*024143         LDA     1       .255    ;
26 00463*107400         AND     0,1             ;    AC1:=VALUE;
27 00464*122404         SUB     1,0     SZR     ;    IF ARG<>0 THEN GOTO GET TABLEB
28 00465*000766         JMP             CL323   ;  END;
29 00466*121000 CL329:  MOV     1,0             ;
30 00467*025030         LDA     1       SAVE4,2 ;
31 00470*006175         PUTBYTE                 ;  TOBYTE:=VALUE(ARG);
32 00471*002235         JMP@            .CL1    ; RETURN;
33
34              ; STORE REGISTER;
35              ; STORE R.CUR IN ADDRESS=WORD(PC.CUR) AND INCR(PC.CUR);
36              CL305:                                  ; STORE:
37 00472*037033         LDA@    3       PC,2    ;   ADDRESS:=WORD(PC.CUR);
38 00473*011033         ISZ             PC,2    ;
39 00474*021032         LDA     0       R,2     ;
40 00475*041400         STA     0       +0,3    ;   WORD(ADDRESS):=R.CUR;
41 00476*002235         JMP@            .CL1    ; RETURN;
42
```

```
10021 INT14
01                ; COMPARESTRING(STRING1,STRING2);
02                ; SAVE : BYTE1
03                ; SAVE1:
04                ; SAVE2: COUNT                    , MODIFBITS(10:11)
05                ; SAVE3: ADDR OF STRING1          ,      -    (14:15)
06                ; SAVE4:  -   -  STRING2          ,      -    (12:13)
07                ; SAVE5: WORK
08
09                ; NOTE: OVERWRITTEN BY 'MUI' !!
10                CL304:                                ; COMPARESTRING:
11 00477*004710           JSR             CL321     ;   TAKEADDRESSES(STRING1,STRING2);
12 00500*006237           JSR*            .CL5      ;   TAKEVALUE(MODIF,VALUE);
13 00501*045026           STA     1       SAVE2,2   ;   SAVE2.CUR:= VALUE;
14 00502*004732 CL34:     JSR             CL326     ;   REPEAT
15 00503*041024           STA     0       SAVE,2    ;     GETBYTE(SAVE3,SAVE);
16 00504*025030           LDA     1       SAVE4,2   ;
17 00505*006174           GETBYTE                   ;     GETBYTE(SAVE4,R);
18 00506*011030           ISZ             SAVE4,2   ;     INCR(SAVE4.CUR);
19 00507*025024           LDA     1       SAVE,2    ;
20 00510*106400           SUB     0,1               ;     R.CUR:=RESULT OF COMPARE;
21 00511*045032           STA     1       R,2       ;
22 00512*125004           MOV     1,1     SZR       ;     IF R.CUR<>0 THEN
23 00513*002235           JMP*            .CL1      ;     RETURN;
24 00514*004723           JSR             CL327     ;   UNTIL COUNT=0;
25 00515*000765           JMP             CL34      ; RETURN;
26
27                ; CALL
28                ;   1: PC POINTS AT THE FIRST CELL IN THE PROCEDURE, WHICH CONTAINS
29                ;      THE ADDR OF THE CELL USED TO SAVE THE LINK=CALLED FROM.
30                ;   2: LINK=PC+1 (JUMP BACK TO) IS WRITEN IN THIS SAVE LINKCELL.
31                ;   3: PC:=SAVELINKCELL+1 (=NOW CONTINUE HERE)
32 00516*037033 CL306:  LDA*    3       PC,2      ; LINK:=W(PC.CUR);
33 00517*011033           ISZ             PC,2      ; INCR(PC.CUR);
34 00520*021033           LDA     0       PC,2      ;
35 00521*041400           STA     0       +0,3      ; LINKWORD.LINK:=PC.CUR;
36 00522*175400 CL365:    INC     3,3               ; PC.CUR:=LINK+1:
37 00523*000435           JMP             CL360+1   ;
38
39                ; BINDEC(BIN.VALUE NAME, DEC.VALUE NAME);
40 00524*006237 CL307:    JSR*            .CL5      ; TAKEVALUE(MODIF,VALUE);
41 00525*045030           STA     1       SAVE4,2   ;
42 00526*006236           JSR*            .CL4      ; TAKEADDRESS(MODIF,ADDRESS);
43 00527*021030           LDA     0       SAVE4,2   ;
44 00530*006232           BINDEC                    ;
45 00531*002235           JMP*            .CL1      ; RETURN;
46
47                ; GO CODE;
48                ;           CALL:                   RETURN:
49                ; AC0       MODIFBITS=-INDEX        WORD(PC.CUR)
50                ; AC1
51                ; AC2       CUR                     CUR
52                ; AC3                               ADDR=CUR+INDEX
53 00532*155000 CL303:    MOV       2,3             ; GO CODE:
54 00533*116400           SUB       0,3             ;   ADDR:=CUR+INDEX;
55 00534*023033           LDA*      0       PC,2    ;   MODIFBITS:=WORD(PC.CUR);
56 00535*011033           ISZ               PC,2    ;   INCR(PC.CUR);
57 00536*003400           JMP*              +0,3    ; GOTO W(ADDR);
58
```

```
!0022 INT14
01                ; LINK;
02                ; PC POINTS AT THE CELL, WHICH CONTAINS THE ADDRESS OF THE RETURN
03                ; ADDRESSCELL.
04                ; IF RETURNADDRESS>=0 THEN PC:=POINT AND CONTINUE HERE
05                ;                     ELSE EXIT FROM BOTTOM OF A
06                ;                     GIVEUPPROCEDURE. NOTE THAT STDGIVEUP IS AL-
07                ;                     WAYS CALLED BEFORE ENTRY TO A USER GIVEUP-
08                ;                     PROCEDURE.
09                ;                     ZONE:=-POINT;
10                ;                     PC.CUR:=SAVEPC.ZONE(=ZBUFFE.ZONE) ( PC IN PROG
11                ;                     GOTO SAVELINK.ZONE(=ZSIZE.ZONE)=RETURN FROM WA
12 00537*037033 CL308:  LDA@    3       PC,2     ; LINK:
13 00540*035400         LDA     3       +0,3     ; POINT:=WORD(WORD(PC.CUR));
14 00541*175113         MOVL#   3,3     SNC      ; IF POINT >=0 THEN
15 00542*000416         JMP             CL360+1  ; GOTO "GOTO";
16 00543*170400 CL385:  NEG     3,2              ; ZONE:=-POINT;
17 00544*021013         LDA     0       ZBUFFE,2;
18 00545*125212         MOVR#   1,1     SZC      ; IF OPERATION(BIT15)=0 THEN
19 00546*000403         JMP             .+3      ; BEGIN !PAGED!
20 00547*006357         GETADR                   ;   GETADR(SAVEPC.ZONE);
21 00550*161000         MOV     3,0              ; END;
22 00551*034040         LDA     3       CUR      ;
23 00552*041433         STA     0       PC,3     ; PC.CUR:=SAVEPC.ZONE;
24 00553*003014         JMP@            ZSIZE,2  ; GOTO SAVELINK.ZONE;
25
26 00554*121000 CL358:  MOV     1,0              ;
27 00555*101222 CL359:  MOVZR   0,0     SZC      ; IF OPERATION(BIT15)=1 THEN
28 00556*000430         JMP             CL350    ; GOTO PJUMP;
29 00557*037033 CL360:  LDA@    3       PC,2     ; GOTO:
30 00560*055033         STA     3       PC,2     ;
31 00561*002235         JMP@            .CL1     ; RETURN;
```

```
10023 INT14
01                ; JUMP: JUMP=34, PJUMP=65;
02                ; IF R.CUR SATISFIES THEN CONDITION AT THE CONTENT OF PC
03                ;                       ELSE INCR(PC) AND CONTINUE.
04 00562*115005 CL309:  MOV     0,3     SNR     ; JUMP:
05 00563*000771         JMP             CL358   ;   IF MODIF=0 THEN GOTO "GOTO".
06 00564*021501         LDA     0       BIT,3   ;   AC0:=1BIT(MODIF);
07 00565*125222         MOVZR   1,1     SZC     ; IF PJUMP THEN
08 00566*101400         INC     0,0             ; AC0(BIT15):=1;
09 00567*004404         JSR             CL399   ;   INDEX:=0;
10 00570*100160         1B0+1B9+1B10+1B11       ;   COND(0) "=,>=,<=" R=0;
11 00571*100034         1B0+1B11+1B12+1B13      ;   COND(1) "<=,<>,<" R<0;
12 00572*100052         1B0+1B10+1B12+1B14      ;   COND(2) ">=,<>,>" R>0;
13 00573*025032 CL399:  LDA     1       R,2     ;
14 00574*125005         MOV     1,1     SNR     ;   IF R.CUR<>0 THEN
15 00575*000404         JMP             CL398   ;   BEGIN
16 00576*175400         INC     3,3             ;     INDEX:=1;
17 00577*125103         MOVL    1,1     SNC     ;     IF R.CUR>0 THEN INDEX:=2;
18 00600*175400         INC     3,3             ;   END;
19 00601*025400 CL398:  LDA     1       +0,3    ;   C:=COND(INDEX);
20 00602*107404         AND     0,1     SZR     ;   IF MODIF AND C THEN
21 00603*000752         JMP             CL359   ;   GOTO "GOTO";
22 00604*011033         ISZ             PC,2    ; SKIP:
23 00605*002235         JMP@            .CL1    ;
24                                              ; MODIF:
25                                              ; =     : 12
26                                              ; >=    : 13
27                                              ; <=    : 14
28                                              ; <>    : 9
29                                              ; <     : 10
30                                              ; >     : 11
```

```
!0024 INT14
01 00606*023033 CL350:  LDA#     0        PC,2     ; PJUMP:
02 00607*006357         GETADR                     ;
03 00610*000750         JMP              CL360+1 ;
04
05              ; PCALL:
06
07 00611*027033 CL382:  LDA#     1        PC,2     ; ADDR LINKWORD;
08 00612*011033         ISZ              PC,2     ;
09 00613*021033         LDA      0       PC,2     ; ADDR:=RETURNADDR;
10 00614*006360         GETPOINT                   ;   GETPOINT(RETURNADDR);
11 00615*121000         MOV      1,0               ;
12 00616*165000         MOV      3,1               ;
13 00617*006357         GETADR                     ;   GETADR(LINKWORD);
14 00620*000404         JMP              CL381    ;
15
16              ;XCALL:
17 00621*037033 CL383:  LDA#     3        PC,2     ; ADDR LINKWORD
18 00622*011033         ISZ              PC,2     ;
19 00623*025033         LDA      1       PC,2     ; RETURNADDR;
20 00624*045777 CL381:  STA      1       -1,3     ; LINKWORD-1:=RETURNADDR;
21 00625*102400         SUB      0,0               ;
22 00626*041400         STA      0       +0,3     ; LINKWORD:=0;
23 00627*000673         JMP              CL365    ;
24
25              ; XLINK AND PLINK:
26
27 00630*023033 CL380:  LDA#     0        PC,2     ;
28 00631*125212         MOVR#    1,1      SZC      ; IF PAGED THEN
29 00632*115001         MOV      0,3      SKP      ;
30 00633*006357         GETADR                     ; GETADR(LINKWORD)
31 00634*021777         LDA      0       -1,3     ; LINKWORD-1
32 00635*035400         LDA      3       +0,3     ; HELP:=LINKWORD(0);
33 00636*175004         MOV      3,3      SZR      ; IF HELP <> 0 THEN
34 00637*000704         JMP              CL385    ; !EXIT FROM BOTTOM OF GIVEUP-PROC.!
35 00640*125212         MOVR#    1,1      SZC      ; ELSE
36 00641*115001         MOV      0,3      SKP      ; IF PLINK THEN
37 00642*006357         GETADR                     ; GETADR(RETURNPOINT);
38 00643*000715         JMP              CL360+1 ;
```

```
10025 INT14
01                ;PROCEDURE SENDTO OPERATOR(COMMAND,ADDRESS,COUNT,BUF);
02                ;
03                ;         CALL:              RETURN:
04                ; AC0     COMMAND            IRRELEVANT
05                ; AC1     ADDRESS              -
06                ; AC2     CUR                BUF
07                ; AC3     LINK               CUR
08
09 00644*055024 CL400:  STA     3       SAVE,2  ; SEND TO OPERATOR:
10 00645*060277          INTDS                   ; DIRTY WORK!
11 00646*040407          STA     0       CL40A   ; ONLY ONE PROCESS AT TIME,
12 00647*044410          STA     1       CL40C   ; THE ENTERPRETER IS REENTRANT.
13 00650*024404          LDA     1       CL401   ; AC1:=ADDR, AC2:=NAME OF RECEIVER.
14 00651*031035          LDA     2       .OPER,2 ; SENDMESSAGE(MESS0.CUR,OPERATOR);
15 00652*006004          SENDMESSAGE             ;
16 00653*003424          JMP@            SAVE,3  ;
17 00654*000655*CL401:   .+1     ; MESSAGE BUILD UP IN NEXT 3 CELLS:
18 00655*000000 CL40A:   0       ; OPERATION
19 00656*000120 CL40B:   80      ; COUNT
20 00657*000000 CL40C:   0       ; ADDRESS
21
22                ; PROCEDURE OPMESS(STRING VARIABLE NAME);
23                CL310:                          ; OPMESS:
24 00660*006236          JSR@            .CL4    ;   TAKEADDRESS(MODIF,ADDRESS);
25 00661*020121          LDA     0       .3      ;   OUTPUTMODE;
26 00662*004762          JSR             CL400   ;
27 00663*006005          WAITANSWER              ;
28 00664*002234          JMP@            .CL0    ; RETURN;
29                                               ;
30                ; PROCEDURE OPIN(STRING VARIABLE NAME);
31                CL311:                          ; OPIN:
32 00665*006236          JSR@            .CL4    ;   TAKEADDRESS(MODIF,ADDRESS);
33 00666*021034          LDA     0       OP,2    ;   OP.CUR CONTAINS BUFFERADDR, SEE
34 00667*101004          MOV     0,0     SZR     ;   IF OP.CUR <> 0 THEN
35 00670*002235          JMP@            .CL1    ;   RETURN1;
36 00671*020120          LDA     0       .1      ;   INPUTMODE;
37 00672*004752          JSR             CL400   ;
38 00673*051434          STA     2       OP,3    ;   OP.CUR:=BUFERADDRESS;
39 00674*002234          JMP@            .CL0    ; RETURN;
40                                               ;
41                ; OPSTATUS(ZONE.Z0,ERRORS);
42                ; SAVE1:
43                ; SAVE2: VALUE            , MODIFBITS(14:15)
44                ; SAVE3: ADDRESS          ,      -    (12:13)
45                ; SAVE4: LINK
46                ; SAVE5:
47                CL313:                          ; OPSTATUS:
48 00675*006237          JSR@            .CL5    ;   TAKEVALUE(MODIF,VALUE);
49 00676*045026          STA     1       SAVE2,2 ;   SAVE2.CUR:= VALUE;
50 00677*006236          JSR@            .CL4    ;   TAKEADDRESS(SAVE3,ADDRESS);
51 00700*045027          STA     1       SAVE3,2 ;
52 00701*004402          JSR             CL402   ;   OUTMESSAGE;
53 00702*002234          JMP@            .CL0    ; RETURN;
54
```

```
10026 INT14
01                ; PROCEDURE OUTMESSAGE;
02                ; THE PROCEDURE OUTPUTS A STRING CORRESPONDING TO EACH BIT IN
03                ; WORD, IF THE BIT IS SET;
04                ;            CALL:
05                ; SAVE2:     WORD
06                ; SAVE3:     ADDRESS OF STRING
07                ; SAVE4:     LINK
08 00703*055030 CL402:  STA      3        SAVE4,2 ;
09                CL403:                            ; REP:
10 00704*021026         LDA      0        SAVE2,2 ;  WORD:=SAVE2.CUR;
11 00705*101005         MOV      0,0      SNR     ;  IF WORD=0 THEN
12 00706*003030         JMP@             SAVE4,2 ;  RETURN;
13 00707*101123         MOVZL    0,0      SNC     ;  IF WORD(0)=1 THEN
14 00710*000407         JMP              CL404   ;  BEGIN
15 00711*041026         STA      0        SAVE2,2 ;    SAVE2:=WORD SHIFT 1;
16 00712*025027         LDA      1        SAVE3,2 ;    ADDRESS:=SAVE3.CUR;
17 00713*020121         LDA      0        .3      ;    COMMAND:=OUT;
18 00714*004730         JSR              CL400   ;    SEND TO OPERATOR;
19 00715*006005         WAITANSWER               ;    WAITANSWER(IRR,IRR,BUF);
20 00716*171001         MOV      3,2      SKP     ;  END
21 00717*041026 CL404:  STA      0        SAVE2,2 ;  ELSE SAVE2.CUR:=WORD SHIFT 1;
22
23                                                ; SKIP:
24 00720*025027         LDA      1        SAVE3,2 ;   ADDRESS:=SAVE3.CUR;
25 00721*006174         GETBYTE                  ;   REPEAT
26 00722*125400         INC      1,1             ;     GETBYTE; ADDR:=ADDR+1;
27 00723*101004         MOV      0,0      SZR     ;   UNTIL BYTE=0;
28 00724*000775         JMP              .-3     ;   SAVE3.CUR:=ADDR;
29 00725*045027         STA      1        SAVE3,2 ;
30 00726*000756         JMP              CL403   ;
31
32                ; PROCEDURE DECBIN(DEC.VAL NAME, BIN.VAL NAME);
33 00727*006236 CL315:  JSR@             .CL4    ; DECBIN:
34 00730*006233         DECBIN                   ;   TAKEADDRESS(MODIF,ADDR);
35 00731*037033         LDA@     3       PC,2    ;   ADDR:=WORD(PC.CUR);
36 00732*011033         ISZ              PC,2    ;   W(ADDR):=BIN.VALUE;
37 00733*045400         STA      1       +0,3    ;   INCR(PC.CUR);
38 00734*002235         JMP@             .CL1    ; RETURN;
39
40                ; PROCEDURE INCR(INTEGER VARIABLE);
41                CL370:                            ; INCR:
42 00735*006236         JSR@             .CL4    ;  TAKEADDRESS(MODIF,VARIABLE);
43 00736*135000         MOV      1,3             ;
44 00737*011400         ISZ              +0,3    ;  INCR(INTEGER VARIABLE);
45 00740*002235         JMP@             .CL1    ;
46 00741*002235         JMP@             .CL1    ;
47
48                ; PROCEDURE DECR(INTEGER VARIABLE);
49                CL375:                            ; DECR:
50 00742*006236         JSR@             .CL4    ;  TAKEADDR(MODIF,VARIABLE);
51 00743*135000         MOV      1,3             ;
52 00744*015400         DSZ              +0,3    ; DECR(INTEGER VARIABLE);
53 00745*002235         JMP@             .CL1    ;  RETURN
54 00746*002235         JMP@             .CL1    ;  RETURN
```

```
10027 INT14
01                ; AT DIRECT OPERATIONS: AC0=MODIF=VALUE;
02 00747*037033 CL120:  LDA@    3       PC,2    ; AND :    AC0:=R AND WORD(WORD(PC))
03 00750*021400         LDA     0       +0,3    ;
04 00751*000402         JMP             .+2     ;
05 00752*023033 CL110:  LDA@    0       PC,2    ; ANDC:    AC0:=R AND WORD(PC);
06 00753*011033         ISZ             PC,2    ;
07 00754*025032 CL100:  LDA     1       R,2     ; ANDD:    AC0:=R AND AC0;
08 00755*123400         AND     1,0             ;
09 00756*000406         JMP             CL101
10
11 00757*037033 CL121:  LDA@    3       PC,2    ; LOAD :   AC0:=WORD(WORD(PC));
12 00760*021400         LDA     0       +0,3    ;
13 00761*000402         JMP             .+2     ;
14 00762*023033 CL111:  LDA@    0       PC,2    ; LOADC:   AC0:=WORD(PC);
15 00763*011033         ISZ             PC,2    ;
16 00764*041032 CL101:  STA     0       R,2     ; LOADD:   R:=AC0;
17 00765*002235         JMP@            .CL1    ; RETURN WITH R=AC0;
18
19 00766*037033 CL122:  LDA@    3       PC,2    ; + :      AC1:=R+WORD(WORD(PC));
20 00767*021400         LDA     0       +0,3    ;
21 00770*000402         JMP             .+2     ;
22 00771*023033 CL112:  LDA@    0       PC,2    ; +C:      AC1:=R+WORD(PC);
23 00772*011033         ISZ             PC,2    ;
24 00773*025032 CL102:  LDA     1       R,2     ; +D:      AC1:=R+AC0;
25 00774*107000         ADD     0,1             ;
26 00775*045032 CL132:  STA     1       R,2     ;
27 00776*002235         JMP@            .CL1    ; RETURN WITH R=AC1;
28
29 00777*037033 CL123:  LDA@    3       PC,2    ; - :      AC1:=R-WORD(WORD(PC));
30 01000*021400         LDA     0       +0,3    ;
31 01001*000402         JMP             .+2     ;
32 01002*023033 CL113:  LDA@    0       PC,2    ; -C:      AC1:=R-WORD(PC);
33 01003*011033         ISZ             PC,2    ;
34 01004*025032 CL103:  LDA     1       R,2     ; -D:      AC1:=R-AC0;
35 01005*106400         SUB     0,1             ;
36 01006*045032         STA     1       R,2     ;
37 01007*002235         JMP@            .CL1    ; RETURN WITH R=AC1;
38
39 01010*037033 CL124:  LDA@    3       PC,2    ; SHIFT :  AC0:=R SHIFT WORD(WORD(P
40 01011*021400         LDA     0       +0,3    ;
41 01012*000402         JMP             .+2     ;
42 01013*023033 CL114:  LDA@    0       PC,2    ; SHIFTC:  AC0:=R SHIFT WORD(PC);
43 01014*011033         ISZ             PC,2    ;
44 01015*115005 CL104:  MOV     0,3     SNR     ; SHIFTD:  AC0:=R SHIFT AC0;
45 01016*002235         JMP@            .CL1    ;
46 01017*025032         LDA     1       R,2     ;
47 01020*101113         MOVL#   0,0     SNC     ; IF SHIFTS>0 THEN
48 01021*174400         NEG             3,3     ; COUNT:=-SHIFTS;
49 01022*101113         MOVL#   0,0     SNC     ; FOR COUNT:=COUNT STEP -1 UNTIL 0 D
50 01023*125121         MOVZL   1,1     SKP     ;  R:= R SHIFT SIGN(SHIFTS);
51 01024*125220         MOVZR   1,1             ;
52 01025*175404         INC     3,3     SZR     ;
53 01026*000774         JMP             .-4     ;
54 01027*000746         JMP             CL132   ;
55
```

```
10028 INT14
01 01030*006236 CL901:  JSR@            .CL4     ; LOAD BYTE:
02                                               ; TAKEADDRESS(MODIF,ADDRESS);
03 01031*006174         GETBYTE                  ; GETBYTE(ADDRESS,VALUE);
04 01032*000732         JMP             CL101    ;
05
06 01033*006236 CL902:  JSR@            .CL4     ; LOAD WORD;
07                                               ; TAKEADDRESS(MODIF,ADDRESS);
08 01034*006174         GETBYTE                  ; GETBYTE(ADDRESS,BYTE);
09 01035*101300         MOVS    0,0              ;
10 01036*041032         STA     0       R,2      ; VALUE:=BYTE*256;
11 01037*125400         INC     1,1              ;
12 01040*006174         GETBYTE                  ; GETBYTE(ADDRESS+1,BYTE);
13 01041*000732         JMP             CL102    ; VALUE:=VALUE+BYTE;
14
15                                               ; LOAD NEGATIVE;
16 01042*006237 CL903:  JSR@            .CL5     ; TAKEVALUE(MODIF,VAL);
17 01043*120400         NEG     1,0              ; VALUE:=-VALUE;
18 01044*000720         JMP             CL101    ;
19
20 01045*037033 CL125:  LDA@    3       PC,2     ; EXTRACT : AC0:=R EXTRACT W(W(PC));
21 01046*021400         LDA     0       +0,3     ;
22 01047*000402         JMP             .+2      ;
23 01050*023033 CL115:  LDA@    0       PC,2     ; EXTRACTC: AC0:=R EXTRACT WORD(PC);
24 01051*011033         ISZ             PC,2     ;
25 01052*034114 CL105:  LDA     3       .16      ; EXTRACTD: AC0:=R EXTRACT AC0;
26 01053*116400         SUB     0,3              ;
27 01054*021501         LDA     0       BIT,3    ;
28 01055*100520         NEGZL   0,0              ;
29 01056*100000         COM     0,0              ;
30 01057*000675         JMP             CL100    ;
31
32 01060*037033 CL126:  LDA@    3       PC,2     ; * :         AC0:=R*WORD(WORD(PC));
33 01061*021400         LDA     0       +0,3     ;
34 01062*000402         JMP             .+2      ;
35 01063*023033 CL116:  LDA@    0       PC,2     ; *C:         AC0:=R*WORD(PC);
36 01064*011033         ISZ             PC,2     ;
37 01065*025032 CL106:  LDA     1       R,2      ; *D:         AC0:=R*AC0;
38 01066*006176         MULTIPLY                 ;
39 01067*000706         JMP             CL132    ;
40
```

```
!0029 INT14
01 01070*037033 CL127:  LDA@     3       PC,2     ; / :         AC0:=R/WORD(WORD(PC));
02 01071*025400         LDA      1       +0,3     ;
03 01072*000402         JMP              .+2      ;
04 01073*027033 CL117:  LDA@     1       PC,2     ; /C:         AC0:=R/WORD(PC);
05 01074*011033         ISZ              PC,2     ;
06 01075*102520 CL137:  SUBZL    0,0             ;     SAVE2=FORTEGN:=1(POS);
07 01076*041026         STA      0       SAVE2,2  ;     DIVIDEND:=R.CUR;
08 01077*021032         LDA      0       R,2      ;     IF DIVIDEND<0 THEN
09 01100*101133         MOVZL#   0,0     SNC      ;     BEGIN
10 01101*000403         JMP              CL147    ;       DIVISOR:=-DIVISOR;
11 01102*124400         NEG      1,1             ;       DIVIDEND:=-DIVIDEND;
12 01103*100400         NEG      0,0             ;     END;
13 01104*125133 CL147:  MOVZL#   1,1     SNC      ;     IF DIVISOR<0 THEN
14 01105*000403         JMP              CL157    ;     BEGIN
15 01106*045026         STA      1       SAVE2,2  ;       FORTEGN:=NEGATIV;
16 01107*124400         NEG      1,1             ;       DIVISOR:=-DIVISOR;
17 01110*006177 CL157:  DIVIDE                   ;     END;
18 01111*025026         LDA      1       SAVE2,2  ;     DIVISION AF TO POS. TAL;
19 01112*125122         MOVZL    1,1     SZC      ;     IF FORTEGN=NEGATIV THEN
20 01113*100400         NEG      0,0             ;     QVOTIENT:=-QVOTIENT;
21 01114*000650         JMP              CL101    ;
22 01115*105000 CL107:  MOV      0,1             ; /D:         AC0:=R/AC0;
23 01116*000757         JMP              CL137    ;
24
```

```
10030 INT14
01                ; STDGIVEUP
02                ; STDGIVEUP IS ALWAYS CALLED BEFORE JUMPING FURTHER TO THE USER-
03                ; GIVEUPPROCEDURE.
04                ; ENTRY: A JUMP TO STDGIVEUP IS DONE FROM WAITTRANSFER BY A JMP@
05                ;        ZGIVEUP. ZGIVEUP=CL6 AND SIZE=ADDR OF USER GIVEUPPROCEDURE,
06                ;        SAVELINK.ZONE=ZSIZE:=WAITTRANSFERS RETURNLINK,
07                ;        SAVEPC.ZONE  =ZBUF :=PC+1.
08                ;        THE TWO SAVELOCATIONS ARE USED AT RETURN FROM USER GIVEUP.
09                ; EXIT : PC POINTING AT USER GIVEUPPROCEDURE AND RETURNLINK FROM THE
10                ;        GIVEUPPROCEDURE IS SET TO -ZONE TO INDICATE THAT STDGIVEUP
11                ;        HAVE BEEN CALLED. (IT IS POSSIBLE TO CALL A USER GIVEUP-
12                ;        PROCEDURE DIRECTLY).
13                ;         CALL        RETURN
14                ; AC0                 PC.CUR
15                ; AC1
16                ; AC2      ZONE
17                ; AC3      WAITTRANSFER LINK
18                ; SIZE.Z   ZONE GIVEUP ADDRESS
19
20
21 01117*055014 CL6:     STA     3       ZSIZE,2 ; SAVELINK.ZONE:=LINK;
22 01120*035003          LDA     3       SIZE,2  ; GIVEUPPROCEDURE:=SIZE.ZONE;
23 01121*140400          NEG     2,0             ;
24 01122*041400          STA     0       +0,3    ; PROC(0):=-ZONE;
25 01123*165400          INC     3,1             ; START IN USERGIVEUPPROC:=PROC+1;
26 01124*034040          LDA     3       CUR     ;
27 01125*021433          LDA     0       PC,3    ;
28 01126*041013          STA     0       ZBUFFE,2; SAVEPC.ZONE:=PC.CUR;
29 01127*045433          STA     1       PC,3    ; PC.CUR:=START;
30 01130*002234          JMP@            .CL0    ; GOTO NEXT0;
31
32
33                ; IN PAGED PROGRAMS THE FOLLOWING STANDARD GIVEUP IS USED:
34
35 01131*055014 CL65:    STA     3       ZSIZE,2 ; SAVELINK.ZONE:=LINK;
36 01132*034040          LDA     3       CUR     ;
37 01133*021433          LDA     0       PC,3    ;
38 01134*006360          GETPOINT                ;
39 01135*055013          STA     3       ZBUFFE,2; SAVEPC.ZONE:=POINT(PC.CUR);
40 01136*021003          LDA     0       SIZE,2  ; GIVEUPPROC:=SIZE.ZONE;
41 01137*006357          GETADR                  ; GETADR(GIVEUPPROC);
42 01140*140400          NEG     2,0             ;
43 01141*041400          STA     0       +0,3    ; PROC(0):=-ZONE;
44 01142*165400          INC     3,1             ;
45 01143*034040          LDA     3       CUR     ;
46 01144*045433          STA     1       PC,3    ; PC.CUR:=STARTADDR;
47 01145*002234          JMP@            .CL0    ; GOTO NEXT0;
```

```
10031 INT14
01                ; STDBREAK, STOP
02                ;          CALL          RETURN
03                ; AC0      ERRORNO
04                ; AC1      STATUS
05                ; AC2      CUR           CUR
06                ; AC3                    PROG
07                ; SAVE     ZONE
08 01146*041025 CL7:     STA      0        SAVE1,2 ; SAVE ERRORNO;
09 01147*045026          STA      1        SAVE2,2 ; SAVE STATUS;
10 01150*021024          LDA      0        SAVE,2  ; SAVE=ADDR OF ZONE USED ON DEVICE,
11 01151*041031          STA      0        SAVE5,2 ; WHICH IS BREAKED;
12 01152*034521          LDA      3        .ZN     ; ZN.CUR=START OF ZONEADDR.TABLE;
13 01153*157000          ADD      2,3              ; HELP:=NORMAL START OF ZONE;
14 01154*027012          LDA*     1        PROG,2 ;
15 01155*020107          LDA      0        .1B6    ;
16 01156*107405          AND      0,1      SNR     ; IF PAGED PROGRAM THEN
17 01157*000406          JMP              CL702   ; BEGIN
18 01160*021400          LDA      0        +0,3    ;   IF CCOROUT=0 THEN
19 01161*101005          MOV      0,0      SNR     ;   DISP:=1
20 01162*101401          INC      0,0      SKP     ;   ELSE
21 01163*020127          LDA      0        .12     ;   DISP:=12;
22 01164*000413          JMP              CL703   ; END ELSE
23 01165*031400 CL702:   LDA      2        +0,3    ; BEGIN !NOT PAGED PROGRAM!
24 01166*021007          LDA      0        ZGIVE,2 ;
25 01167*030506          LDA      2        .INTGIV ;   IF ZGIVEUP.ZONE=INTGIVEUP THEN
26 01170*142405          SUB      2,0      SNR     ;   DISP:=0
27 01171*000406          JMP              CL703   ;   ELSE
28 01172*030040          LDA      2        CUR     ;   BEGIN
29 01173*025047          LDA      1        TRECOR,2;       IF TRECORD(0:0)=1 THEN !NEW
30 01174*020127          LDA      0        .12     ;       DISP:=12
31 01175*125123          MOVZL    1,1      SNC     ;       ELSE DISP:=7;
32 01176*020124          LDA      0        .7      ;   END;
33                                                 ; END;
34 01177*030040 CL703:   LDA      2        CUR     ;
35 01200*117000          ADD      0,3              ; START OF ZONES:=HELP+DISP;
36 01201*055030          STA      3        SAVE4,2 ; SAVE4:=START OF ZONES IN CUR;
37 01202*025011 CL70:    LDA      1        BUFFE,2 ; BUFFE.CUR=END OF PROCESSDESCRIPTOR
38 01203*136432          SUBZ#    1,3      SZC     ;  WHILE ADDR OF ZONES<START OF BUFF
39 01204*000436          JMP              CL710   ;  BEGIN
40 01205*034040          LDA      3        CUR     ;
41 01206*033430          LDA*     2        SAVE4,3 ;
42 01207*025004          LDA      1        ZMODE,2 ;
43 01210*020147          LDA      0        .M256   ;
44 01211*107400          AND      0,1              ;
45 01212*125400          INC      1,1              ;
46 01213*045004          STA      1        ZMODE,2 ;     ZMODE:=ZMODE(0:7)+1; (=INPUT)
47 01214*020460          LDA      0        .CL701  ;     RETURN FROM GIVEUP=CL701 IF
48 01215*041007          STA      0        ZGIVE,2 ;     CALLED BY CLOSE;
49 01216*021005          LDA      0        ZKIND,2 ;
50 01217*041427          STA      0        SAVE3,3 ;     SAVE KIND.ZONE;
51 01220*101120          MOVZL    0,0              ;     IF 1B0=COROUTINE SET IN KIND TI
52 01221*101220          MOVZR    0,0              ;     CLOSE MUST NOT CALL CWANSWER (W
53 01222*041005          STA      0        ZKIND,2 ;     WAITTRANSFER);
54 01223*006220          CLOSE                     ;     CLOSE(ZN);
55 01224*020451 CL701:   LDA      0        .INTGIVE;
56 01225*034040          LDA      3        CUR     ;
57 01226*027412          LDA*     1        PROG,3  ;
58 01227*034107          LDA      3        .1B6    ; IF PAGED PROGRAM THEN
59 01230*137414          AND#     1,3      SZR     ; ZGIVEUP.ZONE:=PINTGIVEUP
60 01231*020445          LDA      0        .PINT   ; ELSE ZGIVEUP.ZONE:=INTGIVEUP;
```

```
 0032 INT14
01 01232*041007           STA     0       ZGIVE,2 ;
02 01233*034040           LDA     3       CUR     ;
03 01234*021427           LDA     0       SAVE3,3 ;    GET SAVED KIND;
04 01235*041005           STA     0       ZKIND,2 ;
05 01236*030040 CL715:    LDA     2       CUR     ; SKIP:
06 01237*011030           ISZ             SAVE4,2 ;    INCR(ADDR OF ZONES);
07 01240*035030           LDA     3       SAVE4,2 ;
08 01241*000741           JMP             CL70    ;  END;
09 01242*006011 CL710:    CLEANPROCESS            ;
10 01243*021025           LDA     0       SAVE1,2 ;  GET ERRORNO;
11 01244*034122           LDA     3       .5      ;
12 01245*116405           SUB     0,3     SNR     ; IF ERRNO=5 THEN
13 01246*000431           JMP             CL816   ; GOTO WTBREAK;
14 01247*034423           LDA     3       CL813   ; TEXT:='BREAK ';
15 01250*024420           LDA     1       CL811   ; CHAR:="+";
16 01251*101133           MOVZL#  0,0     SNC     ; IF ERRNO<0 THEN
17 01252*000404           JMP             .+4     ; BEGIN
18 01253*041032           STA     0       R,2     ;    POSBREAK:=FALSE;
19 01254*100400           NEG     0,0             ;    ERRNO:=-ERRNO;
20 01255*024414           LDA     1       CL812   ;    CHAR:="-";
21 01256*123000           ADD     1,0             ; END;
22 01257*041403           STA     0       +3,3    ; TEXT(BYTE7:8):=CHAR,ERRORNO;
23
```

```
!0033 INT14
01 01260*165120 CL71:    MOVZL   3,1              ; OPERATORMESS:
02 01261*020121          LDA     0       .3       ; OPERATION:=OUTPUT;
03 01262*006446          JSR@            .CL400   ; SEND TO OPER(3,TEXT,ADDR);
04 01263*006005          WAITANSWER               ; WAITANSWER(BUF,IRR,IRR);
05 01264*171000          MOV     3,2              ;
06 01265*006013 CL72:    STOPPROCESS              ; STOPPROCESS(CUR);
07 01266*035012          LDA     3       PROG,2   ; GOTO
08 01267*003401          JMP@            PSTART,3; PSTART.PROG.CUR;
09 01270*020060 CL811:   32*256+48                ; SP,0
10 01271*026460 CL812:   "-*256+48                ; -,0
11 01272*000000'CL813:   .CL83                    ; ADDR:
12               .NREL
13 00000'005102 .CL83:   .TXT    .<10>BREAK <0>. ;
14       051105
15       040513
16       020000
17       000000
18               .XREL
19 01273*000041 .ZN:     ZN                       ; NOTE: USED IN TEXT
20 01274*001224*.CL701: CL701                     ;
21 01275*000226 .INTGIVE: INTGIVEUP               ; INTERPRETER GIVEUPADDRESS;
22 01276*000240 .PINTGIV: PINTGIVEUP              ; INTERPR. GIVEUP FOR PAGED PROGRAMS
23
24               ;WAIT TRANSFER BREAK:
25               CL816:                           ; WTBREAK:
26 01277*035031          LDA     3       SAVES,2  ; ZONE:=SAVES.CUR;
27 01300*020126          LDA     0       .10      ;
28 01301*041777          STA     0       -1,3     ; ZONE(-1):=' <10>';
29 01302*126000          ADC     1,1              ; AC1:=-1;
30 01303*167120          ADDZL   3,1              ; TEXT:=ZONE.ZNAME*2-1;
31 01304*020121          LDA     0       .3       ; SEND TO OPER(OUT,TEXT,ADDR);
32 01305*006423          JSR@            .CL400   ;
33 01306*006005          WAITANSWER               ; WAITANSWER(BUF,IRR,IRR);
34 01307*171000          MOV     3,2              ;
35 01310*025026          LDA     1       SAVE2,2  ; GET STATUS;
36 01311*102000          ADC     0,0              ; COUNT:=-1;
37 01312*101440          INCO    0,0              ; REPEAT
38 01313*125103          MOVL    1,1     SNC      ;  COUNT:=COUNT+1; STATUS:=STATUS*2
39 01314*000776          JMP             .-2      ; UNTIL CARRY;
40 01315*024415          LDA     1       .50      ; CHAR1:='2';
41 01316*034126          LDA     3       .10      ;
42 01317*162423          SUBZ    3,0     SNC      ; IF COUNT>=10 THEN
43 01320*163001          ADD     3,0     SKP      ; BEGIN  COUNT:=COUNT-10;
44 01321*125400          INC     1,1              ;  CHAR:='3'
45 01322*034407          LDA     3       CL814    ; END;
46 01323*045403          STA     1       +3,3     ; TEXT(7:8):= ' ' CON CHAR;
47 01324*024135          LDA     1       .48      ;
48 01325*107300          ADDS    0,1              ; TEXT(9:10):= COUNT+48;
49 01326*045404          STA     1       +4,3     ;
50 01327*000731          JMP             CL71     ; GOTO OPERATORMESS;
51 01330*000644*.CL400: CL400                     ; SEND TO OPERATOR;
52 01331*000005'CL814:   .CL84
53               .NREL
54 00005'020105 .CL84:   .TXT    . ERROR <50><48>.
55       051122
56       047522
57       020062
58       030000
59               .XREL
60 01332*020062 .50:     32*256+50                ; SP,2
```

```
 0034 INT14
01
02                CL820:
03                          .END
0000 SOURCE LINES IN ERROR
```

CL0	000004*	4/18	9/19					
CL00	000000*	4/11	9/15					
CL1	000005*	4/19	8/18	9/21				
CL100	000754*	10/05	27/07	28/30				
CL101	000764*	10/06	27/09	27/16	28/04	28/18	29/21	
CL102	000773*	10/07	27/24	28/13				
CL103	001004*	10/08	27/34					
CL104	001015*	10/09	27/44					
CL105	001052*	10/10	28/25					
CL106	001065*	10/11	28/37					
CL107	001115*	10/12	29/22					
CL11	000015*	10/01	11/10	14/28	17/28	18/14		
CL110	000752*	10/14	27/05					
CL111	000762*	10/15	27/14					
CL112	000771*	10/16	27/22					
CL113	001002*	10/17	27/32					
CL114	001013*	10/18	27/42					
CL115	001050*	10/19	28/23					
CL116	001063*	10/20	28/35					
CL117	001073*	10/21	29/04					
CL120	000747*	10/23	27/02					
CL121	000757*	10/24	27/11					
CL122	000766*	10/25	27/19					
CL123	000777*	10/26	27/29					
CL124	001010*	10/27	27/39					
CL125	001045*	10/28	28/20					
CL126	001060*	10/29	28/32					
CL127	001070*	10/30	29/01					
CL132	000775*	27/26	27/54	28/39				
CL137	001075*	29/06	29/23					
CL14	000013*	9/34	11/10					
CL147	001104*	29/10	29/13					
CL15	000002	9/35	11/10					
CL157	001110*	29/14	29/17					
CL20	000273*	13/05	13/21	15/02	15/13	15/16	15/20	15/23
		15/26	15/40	18/37				
CL200	000240*	14/03	15/02					
CL201	000254*	14/04	14/17	14/20	15/16	17/26		
CL202	000202*	13/21	14/10					
CL203	000164*	13/05	14/06					
CL204	000260*	14/13	14/15	14/19	15/20			
CL205	000400*	14/07	14/18	18/05	18/09	18/37		
CL206	000266*	14/05	14/08	14/09	14/11	14/12	14/14	14/21
		15/26	17/27	18/08				
CL207	000251*	14/16	15/13					
CL208	000263*	15/23	18/06	18/07	18/10			
CL209	000267*	15/19	15/22	15/34	18/43			
CL21	000215*	14/02	14/28					
CL212	000211*	13/28	15/10					
CL22	000363*	18/04	18/14					
CL23	000347*	17/25	17/28					
CL301	000302*	10/37	16/04					
CL302	000316*	10/38	16/32					
CL303	000532*	10/55	21/53					
CL304	000477*	10/40	21/10					
CL305	000472*	10/41	20/36					
CL306	000516*	10/47	21/32					
CL307	000524*	10/51	21/40					
CL308	000537*	10/36	22/12					
CL309	000562*	10/35	11/02	23/04				

```
CL310 000660*       10/44   25/23
CL311 000665*       10/45   25/31
CL312 000326*       10/46   17/02
CL313 000675*       10/50   25/47
CL314 000340*       10/48   17/15
CL315 000727*       10/52   26/33
CL316 000351*       10/53   17/38
CL320 000307*       10/49   16/18
CL321 000407*       16/32   19/04   19/19   20/09   21/11
CL322 000442*       10/42   20/08
CL323 000453*       20/18   20/28
CL324 000415*       10/43   19/19
CL325 000423*       19/25   19/33
CL326 000434*       19/26   19/34   21/14
CL327 000437*       19/25   19/37   21/24
CL329 000466*       20/21   20/29
CL333 000371*       16/21   16/23   17/42   18/25
CL34  000502*       21/14   21/25
CL350 000606*       22/28   24/01
CL358 000554*       22/26   23/05
CL359 000555*       22/27   23/21
CL360 000557*       10/54   21/37   22/15   22/29   24/03   24/38
CL365 000522*       21/36   24/23
CL370 000735*       10/56   26/41
CL375 000742*       10/57   26/49
CL376 000107*       11/05   11/14
CL377 000116*       11/06   11/23
CL380 000630*       10/58   11/01   24/27
CL381 000624*       24/14   24/20
CL382 000611*       11/03   24/07
CL383 000621*       11/04   24/17
CL385 000543*       22/16   24/34
CL398 000601*       23/15   23/19
CL399 000573*       23/09   23/13
CL4   000125*        4/20   12/14
CL400 000644*       25/09   25/26   25/37   26/18   33/51
CL401 000654*       25/13   25/17
CL402 000703*       25/52   26/08
CL403 000704*       26/09   26/30
CL404 000717*       26/14   26/21
CL40A 000655*       25/11   25/18
CL40B 000656*       25/19
CL40C 000657*       25/12   25/20
CL5   000145*        4/21   12/43
CL51  000154*       12/46   12/50
CL52  000161*       12/44   12/55
CL6   001117*        4/13   30/21
CL65  001131*        4/23   30/35
CL7   001146*        4/15   10/03   31/08
CL70  001202*       31/37   32/08
CL701 001224*       31/55   33/20
CL702 001165*       31/17   31/23
CL703 001177*       31/22   31/27   31/34
CL71  001260*       33/01   33/50
CL710 001242*       31/39   32/09
CL715 001236*       32/05
CL72  001265*       33/06
CL811 001270*       32/15   33/09
CL812 001271*       32/20   33/10
CL813 001272*       32/14   33/11
```

```
CL814  001331*      33/45   33/52
CL816  001277*      32/13   33/25
CL820  001333*      34/02
CL901  001030*      10/33   28/01
CL902  001033*      10/34   28/06
CL903  001042*      10/32   28/16
PINTG  000240        4/24   33/22
TESTO  000000        1/01    5/01    9/22    9/26
.50    001332*      33/40   33/60
.CL0   000234        4/18   15/37   16/38   17/06   25/28   25/39   25/53
                    30/30   30/47
.CL00  000225        4/09    4/10
.CL1   000235        4/19   11/20   11/29   13/31   16/09   17/12   17/22
                    17/45   19/39   20/32   20/41   21/23   21/45   22/31
                    23/23   25/35   26/38   26/45   26/46   26/53   26/54
                    27/17   27/27   27/37   27/45
.CL4   000236        4/20   11/14   11/23   15/14   15/24   16/05   18/26
                    19/05   19/07   19/20   20/10   21/42   25/24   25/32
                    25/50   26/33   26/42   26/50   28/01   28/06
.CL40  001330*      33/03   33/32   33/51
.CL5   000237        4/21   11/16   11/25   15/17   15/21   16/07   16/33
                    17/38   18/28   18/38   18/40   19/22   21/12   21/40
                    25/48   28/16
.CL70  001274*      31/47   33/20
.CL83  000000'      33/11   33/13
.CL84  000005'      33/52   33/54
.INTG  001275*      31/25   31/55   33/21
.PINT  001276*      31/60   33/22
.S2    000325*      16/35   16/39
.ZN    001273*      31/12   33/19
```

```
.CL5   000237          4/21    11/16    11/25    15/17    15/21    16/07    16/33
                      17/38    18/28    18/38    18/40    19/22    21/12    21/40
                      25/48    28/16
.CL70  001274*        31/47    33/20
.CL83  000000'        33/11    33
```