```
;                                          RCSL:      43-GL2851
;                                          AUTHOR:    HKM
;                                          EDITED:    76.07.01
```

FLOPPY DISK DRIVER.

```
;                                          FD204
```

```
; KEYWORD: MUS, DRIVER, FLEXIBLE DISC, FLOPPY, DISKETTE, LISTING.
; ABSTRACT: DRIVER FOR FLEXIBLE DISC RC3650, UNIT 0.
;           ASCII  PAPER TAPE  RCSL:  43-GL2852
;           BINARY PAPER TAPE  RCSL:  43-GL2853
```

```
↑↑↑
                ;     *****   START OF FLOPPY DISC DRIVER, PRIMARY UNIT    *****
                ;  VERSION:  76.07.01/1
                .TITL FD204
                .NREL
        000012  .RDX 10
        000001  .TXTM 1
                ;
  00000'140001  FD00:           1B0+1B1+1       ; SPECIFICATION
  00001'000342'                 FD01            ; START OF PROGRAM
  00002'000000                  0               ; CHAIN
  00003'000470                  DE0-FD00        ; SIZE OF PROGRAM
  00004'043104  .TXT            .FD
  00005'030000  0<0>
  00006'000000  <0>.    ; NAME
```

```
↑↑↑
                ; PROCEDURE POSITION;
                ; IT IS ASSUMED THAT BLOCKCOUNT >= 0.
                ; THE PROCEDURE TEST THAT THE BLOCK IS WITHIN THE DISC LIMIT.
                ; IF THE POSITION > DISCSIZE THEN STATUS(POSITION ERROR).
                ; IF THE POSTION >= DISC SIZE THEN STATUS(END MEDIUM).
                ;
                ;    AC                CALL              RETURN
                ;    0                  -               DESTROYED
                ;    1                  -               SOFTSTATUS
                ;    2                 CUR                CUR
                ;    3                 LINK            DESTROYED
                ; FDBLOCK.CUR       BLOCKCOUNT     IF OUTSIDE DISC, THEN 2002
                ;                                  ELSE BLOCKCOUNT
                ; TRACK.CUR         OLD TRACK NO.       NEW TRACK NO.
                ; SECTOR.CUR            -              SECTOR
                ; RETURN:  LINK + 0: NORMAL;
                ;          GIVEUP  : HARD ERROR; ADDR. OUTSIDE TABLE OR DISC;
                ;
      000000    FDPOSITION:                  ; BEGIN POSITION;
 00007'055050           STA     3   POLINK,2  ;
 00010'021043           LDA     0   FDBLOCK,2 ;  GETPOSITION;
 00011'034465           LDA     3   .2002     ;
 00012'116032           ADCZ#   0,3 SZC        ;  IF POSITION >= DISC SIZE THEN
 00013'000412           JMP         POS05      ;    BEGIN END MEDIUM;
 00014'024114           LDA     1   SEM        ;      IF POSITION = DISCSIZE THEN
 00015'116404           SUB     0,3 SZR        ;        STATUS(END MEDIUM) ELSE
 00016'024462           LDA     1   POSERROR   ;        STATUS(END MEDIUM,POSITION);
 00017'004475           JSR         LOGOR      ;
 00020'020456           LDA     0   .2002     ;
 00021'025044           LDA     1   SLICE,2    ;      LOGICAL BLOCK NO.:=
 00022'006177           DIVIDE                 ;        DISCSIZE//SLICESIZE;
 00023'041042           STA     0   BLOCK,2    ;
 00024'002455           JMP@        .GIVEUP    ;      RETURN(GIVEUP);
                                               ;    END END MEDIUM;

                ; CALCULATE THE PHYSICAL TRACK AND SECTOR NUMBER;
                ; ENTRY:AC0=BLOCK NO.; EXIT:AC0:=TRACK NO.,SECTOR.CUR:=SECTOR;

 00025'025036   POS05:LDA     1   FDMODE,2   ;  IF MODE=
 00026'034116           LDA     3   .1B13      ;    PHYSICAL POSITION THEN
 00027'137404           AND     1,3 SZR        ;
 00030'000406           JMP         POS10      ;    BEGIN !PHYSICAL!
 00031'024444           LDA     1   POS26      ;
 00032'006177           DIVIDE                 ;      NEWTRACK:=(BLOCK//26);
 00033'175400           INC     3,3            ;
 00034'055040           STA     3   SECTOR,2   ;      SECTOR:=(BLOCK MOD 26) + 1;
 00035'000413           JMP         POS11      ;    END !PHYSICAL!;
 00036'024124   POS10:LDA     1   .POSN       ;    ELSE
 00037'006176           MULTIPLY               ;    BEGIN !LOGICAL!
 00040'121400           INC     1,0            ;
 00041'024434           LDA     1   POS26      ;      SECTOR:=
 00042'006177           DIVIDE                 ;        (BLOCK*N + 1) MOD 26;
 00043'175005           MOV     3,3 SNR        ;      IF SECTOR = 0 THEN
 00044'034431           LDA     3   POS26      ;        SECTOR:= 26;
 00045'055040           STA     3   SECTOR,2   ;
 00046'021043           LDA     0   FDBLOCK,2  ;     NEWTRACK:= BLOCK//26;
 00047'006177           DIVIDE                 ;    END !LOGICAL!;
```

↑↑↑

```
                ; AJUST THE HEAD TO THE CALCULATED TRACKNUMBER.
                ; IF THE OLD TRACK NO. IS UNDEFINED, THE HEAD IS POSITIONED
                ; TO TRACK00 FIRST.
                ;
                ; WHEN THE HEAD IS POSITIONED THE TRACK REGISTER IN THE
                ; CONTROLLER IS REDEFINED. THIS IS NESSESARY WHEN DEALING
                ; WITH DISKETTES WRITTEN BY FOREIGN COMPAGNIES (I.E. IBM)
                ; USING DELETED TRACKS.
                ; NORMALY THE TRACK REGISTER REMAINS ITS CONTENT.
                ;
00050'025041    POS11:LDA     1    TRACK,2    ; BEGIN HEAD AJUSTMENT;
00051'041041          STA     0    TRACK,2    ;  OLDTRACK:=TRACK; TRACK:=NEWTRACK;
00052'106415          SUB#    0,1  SNR        ;  IF OLDTRACK=TRACK THEN
00053'000412          JMP           POS20     ;    GOTO REDEFINE TRACK ADDRESS;
00054'125414          INC#    1,1  SZR        ;
00055'000405          JMP           POS16     ;  IF OLDTRACK=UNDEFINED THEN
00056'020107          LDA     0    RECAL      ;    BEGIN RECALIBRATION;
00057'004423          JSR           FDEXECUTE ;      RECALIBRATE(DEVICE);
00060'004443          JSR           STACHECK  ;      TEST STATUS(DEVICE);
00061'021041          LDA     0    TRACK,2    ;    END RECALIBRATION;
00062'024415    POS16:LDA     1    TRSEACH    ; SEARCH TRACK:
00063'123000          ADD     1,0            ;
00064'004416          JSR           FDEXECUTE ;  SEARCH TRACK(DEVICE,TRACK NO.);
00065'024412    POS20:LDA     1    TRSEARCH   ; REDEFINE TRACK ADDRESS:
00066'021041          LDA     0    TRACK,2    ;  COMMAND:=
00067'123000          ADD     1,0            ;    TRACKSEARCH < 8 + TRACK NO.;
00070'025055          LDA     1    FDDISP,2   ;  COMMAND:=
00071'122400          SUB     1,0            ;    COMMAND - DISPLACEMENT.CUR;
00072'005073          JSR           IOSPEC,2  ;  REDEFINE TRACK ADDR(DEV,COMMAND);
00073'004430          JSR           STACHECK  ;  TEST STATUS(DEVICE);
00074'003050          JMP@          POLINK,2  ; END HEAD AJUSTMENT;
                                                ; END POSITION;

                ; CONSTANT:
00075'000032    POS26:        26              ; NO OF SECTORS PER TRACK;
00076'003722    .2002:        2002            ; DISC SIZE;
      000124    .POSN=        .7              ; N (EACH N'TH SECTOR IS USED)
                                              ;   ****** ODD ONLY ******
                ;
      000107    RECAL=        .1B6            ; RECALIBRATION COMMAND;
00077'001400    TRSEACH:      3B7             ; TRACK SEARCH COMMAND;
00100'000030    POSERROR:     1B11+1B12       ; STATUS(END MEDIUM,POSITION);


                ; STEPPING STONE NO. 2:
                ;
00101'000372'   .GIVEUP:      GIVEUP          ; RETURN ANSWER(BYTECOUNT:= 0);
```

↑↑↑

```
                ; PROCEDURE EXECUTE COMMAND;
                ; STARTS THE DEVICE ACCORDING TO THE COMMAND;
                ; THEN IT WAITS FOR INTERRUPT: MAX. WAIT TIME IS 2.4 SEC.;
                ;
                ; ENTRY FDEXE:
                ;    AC            CALL            RETURN
                ;    0           COMMAND          COMMAND
                ;    1             -          ADDR. OF DEVICE NO.
                ;    2           CUR               CUR
                ;    3           LINK              CUR
                ; FDCOMMAND.CUR    -              COMMAND
                ;
       000000   FDEXECUTE:                      ; BEGIN EXECUTE COMMAND;
00102'041037          STA     0   FDCOM,2      ;
00103'055024          STA     3   SAVE,2       ; ENTRY FDEXECUTE + 1:
00104'025035          LDA     1   DEVICE,2    ;
00105'006170          SETINTERRUPT             ;  SETINTERRUPT(DEVICE.CUR);
00106'021037          LDA     0   FDCOM,2     ;
00107'005056          JSR         IODOAS,2    ;  START DEVICE(COMMAND);
00110'030141          LDA     2   .120        ;  TIMER:= 120 PERIODS = 2.4SEC.;
00111'006003          WAITINTERRUPT            ;  WAITINTERRUPT(DEVICE.CUR,DELAY);
00112'000442          JMP         FDTIMER     ;+0: TIME OUT: STATUS(TIMER);
00113'003024          JMP@        SAVE,2      ;+1: INT. REC.;
                                               ; END EXECUTE COMMAND;


                ; PROCEDURE LOGOR;
                ; COMPUTES THE LOGICAL OR BETWEEN THE CONTENT OF AC1 AND THE
                ; SOFTWARE STATUS AND PLACES THE RESULT IN THE STATUS WORD;
                ;
                ; ENTRY LOGOR:
                ;    AC              CALL            RETURN
                ;    0                -            UNCHANGED
                ;    1              WORD 1        STATUS WORD
                ;    2              CUR              CUR
                ;    3              LINK             LINK
                ;
00114'031034    LOGOR:LDA     2   FDSOFT,2    ; BEGIN LOGICAL OR;
00115'150000          COM     2,2             ;  SOFTSTATUS:=
00116'147400          AND     2,1             ;    SOFTSTATUS OR WORD1;
00117'146000          ADC     2,1             ; ! AS IN -HOW TO USE THE...- !;
00120'030040          LDA     2   CUR         ;
00121'045034          STA     1   FDSOFT,2    ;
00122'001400          JMP         +0,3        ; END LOGICAL OR;
```

```
↑↑↑
                ; PROCEDURE TEST STATUS;
                ; THE DEVICE IS SENSED AND THE RECEIVING STATUS IS TESTED FOR:
                ; 1. HARD ERRORS (DISCONNECTED, OFF LINE, TIMER) =>
                ;    POSITION:= UNDEFINED AND THE ANSWER IS RETURNED;
                ; 2. TRACK ADDRESS ERROR (POSITION ERROR) =>
                ;    POSITION:= UNDEFINED AND IF LAST COMMAND <> READ BLOCK
                ;    THEN RETURN ANSWER;
                ; 3. ADDRESS FIELD ERROR (PARITY ERROR ON ADDRESS FIELD) =>
                ;    STATUS(PARITY):= 1, POSITION:= UNDEFINED AND IF THE
                ;    LAST COMMAND <> READ BLOCK THEN RETURN ANSWER;
                ; ENTRY STACHECK:
                ;      AC                  CALL                    RETURN
                ;      0                    -                     DESTROYED
                ;      1                    -                      STATUS
                ;      2    IF LAST COMMAND= READ BLOCK THEN        CUR
                ;                          -1;
                ;      3                   LINK                   DESTROYED
                ; RETURN:   LINK: NO HARD ERRORS
                ;         GIVEUP: HARD ERROR
       000000   STACHECK:                    ; BEGIN PROCEDURE TEST STATUS;
 00123'141000          MOV     2,0          ;
 00124'030040          LDA     2   CUR      ;
 00125'041054          STA     0   RESEM,2  ;  SAVE COMMAND SPECIFICATION;
 00126'055051          STA     3   STALINK,2;
 00127'005071          JSR         IODIA,2  ;  NEW:= SENSE(DEVICE);
 00130'034433          LDA     3   HMASK    ;  IF NEW(DISCON,OFF LINE,TIMER)
 00131'137404          AND     1,3 SZR      ;    <> 0 THEN
 00132'000424          JMP         HARD1    ;    GOTO HARD ERROR;
 00133'034115          LDA     3   .1B12    ;
 00134'137404          AND     1,3 SZR      ;  IF NEW(TRACK ERROR) = 1 THEN
 00135'004411          JSR         STA5     ;    GO/NO-GO WRITE BLOCK;
 00136'004756          JSR         LOGOR    ;  STATUS:= STATUS OR NEW;
 00137'034104          LDA     3   SDEV1    ;  IF STATUS(ADDR. FIELD ERROR)= 1
 00140'137405          AND     1,3 SNR      ;    THEN
 00141'000404          JMP         STA1     ;    BEGIN ADDRESS ERROR;
 00142'024113          LDA     1   SPARITY  ;      STATUS:= STATUS OR PARITY;
 00143'004751          JSR         LOGOR    ;      GO/NO-GO(WRITE BLOCK);
 00144'004402          JSR         STA5     ;    END ADDRESS ERROR;
 00145'003051  STA1:   JMP@        STALINK,2;  RETURN(LINK);
                ;
 00146'102000  STA5:   ADC     0,0          ; GO/NO-GO:
 00147'041041          STA     0   TRACK,2  ;  POSITION:= UNDEFINED;
 00150'021054          LDA     0   RESEM,2  ;  IF COMMAND <> READ BLOCK THEN
 00151'101404          INC     0,0 SZR      ;
 00152'000403          JMP         HARD2    ;    GOTO NO-GO ELSE
 00153'001400          JMP         +0,3     ;    RETURN(GO);
                ;
       000000   FDTIMER:                    ; TIMEOUT: SOFTWARE TIMER ENTRY;
 00154'024117          LDA     1   .1B14    ;  STATUS:= TIMER;
 00155'135000  HARD2:  MOV     1,3          ; NO-GO:
 00156'102000  HARD1:  ADC     0,0          ; HARD ERROR:
 00157'041041          STA     0   TRACK,2  ;  POSITION:= UNDEFINED;
 00160'055034          STA     3   FDSOFT,2 ;
 00161'005075          JSR         IOCLEAR,2;  CLEAR, PULSE(DEVICE);
 00162'002717          JMP@        .GIVEUP  ;  RETURN(GIVEUP(RETURN ANSWER));
                ;
 00163'140002  HMASK:  1B0+1B1+1B14         ; HARD ERROR MASK;
                                           ; END PROCEDURE TEST STATUS;
```

```
↑↑↑
                ; PROCEDURE READBLOCK(BLOCK,SLICE);
                ;  THE DISC IS POSITIONED ACCORDING TO THE BLOCK AND SLICESIZE.
                ;  THEN THE POSITIONED SECTOR IS READ TO THE HARDWARE BUFFER
                ;  IN THE CONTROLLER AND THE FIELD SYNC. BYTE IS TESTED. IF IT
                ;  IS UNDEFINED THE STATUS(PARITY) IS SET BUT THE READING
                ;  CONTINUES. IF THE FIELD SYNC. BYTE IS EQUAL TO "DELETED"
                ;  SECTOR THEN THE STATUS(BLOCK SKIPPED) IS SET AND IF THE READ-
                ;  MODE= "NON SKIP BLOCK" THE READING IS CONTINUED ELSE THE NEXT
                ;  BLOCK IS READ (NOTE THAT THE WHOLE SLICE IS SKIPPED).
                ;  NOW THE DATA ARE DELIVERED TO THE ADDRESSED SHARE AND THE
                ;  NEXT SECTOR IN THE SLICE IS READ AS DESCRIBED ABOVE UNTIL
                ;  THE SLICE SIZE IS REACHED.
                ;  IF DESCRIPANCIES ARE SENSED BETWEEN THE EXPECTED AND THE READ
                ;  ADDRESS FIELD, THE STATUS(POSITION) IS SET, BUT THE
                ;  SECTOR DATA IS DELIVERED AND THE READING IS CONTINUED.
                ;
                ; ENTRY READB:
                ;    AC                  CALL                    RETURN
                ;    0                    -                    DESTROYED
                ;    1                    -                 SOFTWARE STATUS
                ;    2                   CUR                     CUR
                ;    3                   LINK                 DESTROYED
                ;
      000000    READBLOCK:                  ; BEGIN READ BLOCK;
00164'055053          STA    3   RELINK,2  ;
00165'004463    RB00: JSR          ISLICE    ;  INIT SLICE;
00166'004443    RB05: JSR          I.READ    ;  INIT READ;

                ; THE CONTENT OF THE ADDRESSED BLOCK IS NOW IN THE CONTROLLER
                ; BUFFER AND THE NEXT PIECE OF CODE FETCHES AND DELIVERS THE
                ; DATA TO THE ADDRESSED SHARE IN THE CORE.

00167'005064          JSR          IODIC,2   ;  GET(DEVICE,TRACK<8 + SYNC. BYTE);
00170'034143          LDA    3     .255      ;  MASKOUT(SYNC. BYTE);
00171'163400          AND    3,0             ; ! THE CUR. TRACK SENSED IS NOT
00172'034547          LDA    3     FDFB      ;   USED IN THIS DRIVER !
00173'116405          SUB    0,3   SNR       ;  IF SYNC. BYTE = HEX(FB) THEN
00174'000412          JMP          RB22      ;    GOTO GETBYTE;
00175'034543          LDA    3     FDF8      ;
00176'116404          SUB    0,3   SZR       ;  IF SYNC. BYTE <> HEX(F8) THEN
00177'000422          JMP          RB26      ;    GOTO UNDEFINED FIELD ELSE
00200'024105          LDA    1     SDEV2     ; SKIPBLOCK:
00201'004713          JSR          LOGOR     ;    STATUS:= STATUS OR SKIPBLOCK;
00202'025036          LDA    1     FDMODE,2  ;
00203'034117          LDA    3     .1B14     ;
00204'137405          AND    1,3   SNR       ;  IF OPERATION=NORMAL(SKIPBLOCK)
00205'000417          JMP          RB30      ;    THEN GOTO RETURN(LINK) ELSE
00206'005067    RB22: JSR          IODIB,2   ; GETBYTE: CHAR:= BYTE(DEVICE,COUNT);
00207'006173          CONBYTE                ;  CONCHAR:=
00210'025026          LDA    1     ADDRESS,2 ;    CONBYTE(CONV,CHAR);
00211'006175          PUTBYTE                ;  PUTBYTE(ADDRESS,CONCHAR);
00212'011026          ISZ          ADDRESS,2 ;  ADDRESS:= ADDRESS + 1;
00213'015027          DSZ          COUNT,2   ;  COUNT:= COUNT - 1;
00214'000772          JMP          RB22      ;  IF COUNT <> 0 THEN GETBYTE;
00215'004706    RB24: JSR          STACHECK  ; RETURN(LINK): TEST STATUS(DEVICE);
00216'004444          JSR          T.SLICE   ;  TEST SLICE COUNT;
00217'000747          JMP          RB05      ;+0: NOT FINISHED;
00220'003053          JMP@         RELINK,2  ;+1: SLICE READ: RETURN(LINK);
```

```
↑↑↑
00221'024113  RB26: LDA     1   SPARITY    ; UNDEFINED FIELD:
00222'004672        JSR         LOGOR      ;  STATUS:= STATUS OR PARITY;
00223'000763        JMP         RB22       ;  GOTO GETBYTE;

00224'035025  RB30: LDA     3   BUF,2      ; SKIP BLOCK:
00225'035410        LDA     3   MESS2,3    ;  BYTECOUNT:= 0;
00226'055026        STA     3   ADDRESS,2  ;  ! ADDRESS.CUR:= MESS2.BUF !
00227'011042        ISZ         BLOCK,2    ;  UPSPACE BLOCK;
00230'000735        JMP         RB00       ;  READBLOCK(BLOCK,SLICE);
                                           ; END READ BLOCK;


                ; PROCEDURE INIT READ;
                ;  THE PROCEDURE INITIATES THE SLICE COUNT, SETS THE POSITION
                ;  OF THE DISC, READ IN THE ADDRESSED SECTOR AND
                ;  TESTS THE STATUS.
                ;
                ; ENTRY I.READ:
                ;  AC                CALL                  RETURN
                ;  0                  -                    DESTROYED
                ;  1                  -                    STATUS
                ;  2                 CUR                     CUR
                ;  3                 LINK                 DESTROYED
                ;
      000000    I.READ:                      ; BEGIN INIT READ;
00231'055046        STA     3   I.TLINK,2 ;
00232'004425        JSR         I.COUNT    ;  INIT COUNT, SETPOSITION;
00233'021040        LDA     0   SECTOR,2   ;
00234'004646        JSR         FDEXECUTE  ;  READ BLOCK(SECTOR);
00235'152000        ADC     2,2            ;  COMMAND SPEC.:= READ BLOCK;
00236'004665        JSR         STACHECK   ;  TEST STATUS(COMMAND SPEC.);
00237'003046        JMP@        I.TLINK,2  ; END INIT READ;


                ; PROCEDURE INIT WRITE;
                ;  FIRST IT SETS THE DATA FIELD SYNC. BYTE IN THE CONTROLLER.
                ;  NEXT SECTOR IS WRITTEN AND THE STATUS IS TESTED.
                ;
                ; ENTRY I.WRITE:
                ;  AC            CALL              RETURN
                ;  0        FIELD SYNC. BYTE       DESTROYED
                ;  1              -                STATUS
                ;  2             CUR                CUR
                ;  3             LINK             DESTROYED
                ;
      000000    I.WRITE:                     ; BEGIN INIT WRITE;
00240'055046        STA     3   I.TLINK,2 ;
00241'005062        JSR         IODOC,2    ;  OUTCHAR(DATA FIELD SYNC. BYTE);
00242'020110        LDA     0   WRITE      ;  COMMAND:=
00243'025040        LDA     1   SECTOR,2   ;    WRITE + SECTOR;
00244'123000        ADD     1,0            ;
00245'004635        JSR         FDEXECUTE  ;  OUTPUT SECTOR(COMMAND);
00246'004655        JSR         STACHECK   ;  TEST STATUS;
00247'003046        JMP@        I.TLINK,2  ; END INIT WRITE;

      000110    WRITE=       .1B7          ; WRITE RECORD;
```

```
↑↑↑
                ; PROCEDURE INIT SLICE COUNT;
                ;  THE SLICE COUNTER:= SLICE SIZE AND THE PHYSICAL BLOCK NO. IS
                ;  CALCULATED.
                ;
                ; ENTRY ISLICE:
                ;  AC            CALL              RETURN
                ;  0              -                 ZERO
                ;  1              -            PHYSICAL BLOCKCOUNT
                ;  2             CUR                CUR
                ;  3             LINK              DESTROYED
                ;
      000000    ISLICE:                       ; BEGIN INIT SLICE COUNT;
00250'055047          STA     3   I.SLINK,2 ;
00251'021044          LDA     0   SLICE,2   ;  SLICE COUNT:=
00252'041045          STA     0   FDSLICE,2 ;    SLICE SIZE;
00253'025042          LDA     1   BLOCK,2   ;
00254'006176          MULTIPLY            ;  PHYSICAL BLOCK:=
00255'045043          STA     1   FDBLOCK,2 ;    LOGICAL BLOCK * SLICESIZE;
00256'003047          JMP@        I.SLINK,2 ; END INIT SLICE COUNT;

                ; PROCEDURE INITCOUNT;
                ;  PRESETS THE BYTECOUNT VALUE TO 128
                ;  AND CALL THE PROCEDURE POSITION.
                ;
                ; ENTRY I.COUNT:
                ;  AC            CALL              RETURN
                ;  0              -                DESTROYED
                ;  1              -                STATUS
                ;  2             CUR                CUR
                ;  3             LINK             DESTROYED
                ;
      000000    I.COUNT:                      ; BEGIN INITCOUNT;
00257'020111          LDA     0   .128      ;
00260'041027          STA     0   COUNT,2   ;  COUNT.CUR:= 128;
00261'002455          JMP@        .POSITION;  SETPOSITION(FDBLOCK);
                                  ; END INITCOUNT;

                ; PROCEDURE TEST SLICE SIZE;
                ;  INCREASES THE PHYSICAL BLOCKCOUNT AND DECREASES THE
                ;  SLICECOUNT. IF THE SLICECOUNT GOES TO ZERO THE LOGICAL
                ;  BLOCK NUMBER IS INCREASED.
                ;
                ; ENTRY T.SLICE:
                ;  AC            CALL              RETURN
                ;  0              -                UNCHANGED
                ;  1              -                UNCHANGED
                ;  2             CUR                CUR
                ;  3             LINK               LINK
                ;  RETURN:  LINK + 0: SLICECOUNT <> 0,
                ;           LINK + 1: SLICECOUNT =  0;
                ;
      000000    T.SLICE:                      ; BEGIN TEST SLICE SIZE;
00262'011043          ISZ       FDBLOCK,2 ;  PHYSICAL BLOCK + 1;
00263'015045          DSZ       FDSLICE,2 ;  SLICECOUNT - 1;
00264'001400          JMP       +0,3      ;  IF SLICECOUNT<>0 THEN RETURN(LINK);
00265'011042          ISZ       BLOCK,2   ; SLICECOUNT=0: LOGICAL BLOCK + 1;
00266'001401          JMP       +1,3      ;  RETURN(LINK+1);
                                  ; END TEST SLICE SIZE;
```

```
↑↑↑
                ; PROCEDURE WRITE/READ BLOCK;
                ; FIRST THE PROCEDURE WRITE NORMAL IS CALLED TO WRITE THE
                ;   ADDRESSED SHARE.
                ; NEXT THE BLOCK IS READ BACK TO THE CONTROLLER BUFFER
                ;   BUT ONLY THE STATUS INFORMATION IS USED;
                ;
                ; ENTRY WRCHECK:
                ;   AC            CALL              RETURN
                ;   0              -               DESTROYED
                ;   1              -                STATUS
                ;   2             CUR                CUR
                ;   3             LINK             DESTROYED
                ;
      000000    WRCHECK:                        ; BEGIN PROCEDURE WRITE/READ BLOCK;
00267'055053          STA     3   RELINK,2     ;
00270'004416          JSR         WRNORMAL    ;  WRITE BLOCK NORMAL;
00271'015042          DSZ         BLOCK,2     ;  BACKSPACE BLOCK;
00272'000401          JMP         .+1         ;
00273'004755          JSR         ISLICE      ;  INIT SLICE COUNT;
00274'004735    WRC05:JSR         I.READ      ;  INIT READ;
00275'004765          JSR         T.SLICE     ;  TEST SLICE COUNT;
00276'000776          JMP         WRC05       ;+0: NOT FINISHED;
00277'003053          JMP@        RELINK,2    ;+1: END PROCEDURE WRITE/READ BLOCK;

                ; PROCEDURE ERASE(BLOCK);
                ;  THE PROCEDURE WRITES THE "DELETE" CHARACTER IN FRONT OF THE
                ;  DATA FIELD OF THE FIRST SECTOR OF THE SLICE FOLLOWED BY
                ;  UNDEFINED DATA. AT EXIT THE BLOCKCOUNT IS INCREASED.
                ;
                ; ENTRY WRERASE:
                ;  AC            CALL              RETURN
                ;  0              -               DESTROYED
                ;  1              -                STATUS
                ;  2             CUR                CUR
                ;  3             LINK             DESTROYED
                ;
      000000    WRERASE:                        ; BEGIN ERASE;
00300'055052          STA     3   WRLINK,2     ;
00301'004747          JSR         ISLICE      ;  INIT SLICE COUNT;
00302'020436          LDA     0   FDF8        ;  WRITE(FDF8,DUMMY.DATA);
00303'004735          JSR         I.WRITE     ;
00304'011042          ISZ         BLOCK,2     ;  UPSPACE BLOCK;
00305'003052          JMP@        WRLINK,2    ; END ERASE;
```

↑↑↑

```
                ; PROCEDURE WRITE BLOCK NORMAL;
                ; FIRST THE HEADS ARE POSITIONED. THEN IT CALLS THE PROC.
                ; OUTCHAR. WITH DATA FIELD SYNC. BYTE = HEX(FB). THEN IT
                ; WRITES THE BLOCK ON THE POSITIONED SECTOR AND TEST STATUS;
                ;
                ; ENTRY WRNORMAL:
                ;     AC          CALL          RETURN
                ;     0            -           DESTROYED
                ;     1            -          SOFTWARE STATUS
                ;     2           CUR             CUR
                ;     3           LINK          DESTROYED
                ;
       000000   WRNORMAL:                      ; BEGIN PROCEDURE WRITE;
00306'055052          STA     3   WRLINK,2     ;
00307'004741          JSR         ISLICE       ;  INIT SLICE COUNT;
00310'004747   WR05:  JSR         I.COUNT      ;  INIT COUNT, SETPOSITION;
00311'020106          LDA     0   SDEV3         ;
00312'123405          AND     1,0 SNR          ;  IF DISC WRITE PROTECTED THEN
00313'000404          JMP         WR95         ;    BEGIN ILLEGAL;
00314'024107          LDA     1   SILLE        ;      STATUS:=STATUS OR ILLEGAL(6);
00315'006417          JSR@        .LOGOR       ;      RETURN TO GIVEUP;
00316'000454          JMP         GIVEUP       ;    END ILLEGAL;;
00317'025026   WR95:  LDA     1   ADDRESS,2    ;  FOR COUNT:= 128
00320'011026          ISZ         ADDRESS,2    ;    STEP -1 UNTIL 0 DO
00321'006174          GETBYTE                  ;    BEGIN !OUTCHAR DATA!
00322'006173          CONBYTE                  ;      X:= GETBYTE(ADDRESS);
00323'005060          JSR         IODOB,2      ;      DATA:= CONBYTE(X);
00324'015027          DSZ         COUNT,2      ;      OUTCHAR(DATA);
00325'000772          JMP         WR95         ;      ADDRESS:= ADDRESS + 1;
                                               ;    END !OUTCHAR DATA!
00326'020413          LDA     0   FDFB         ;
00327'004711          JSR         I.WRITE      ;  INIT WRITE;
00330'004732          JSR         T.SLICE      ;  TEST SLICE COUNT;
00331'000757          JMP         WR05         ;+0: NOT FINISHED;
00332'003052          JMP@        WRLINK,2     ;+1: SLICE WRITTEN:
                                               ; END PROCEDURE WRITE;


                ; STEPPING STONES NO. 1:

00333'000164' .READBLOCK:       READBLOCK ; READ BLOCK;
00334'000114' .LOGOR:           LOGOR     ; STATUS:= STATUS OR (AC1);
00335'000123' .STACHECK:        STACHECK  ; TEST STATUS;
00336'000007' .POSITION:        FDPOSITION; SETPOSITION(PHYSICAL BLOCK);


                ; CONSTANTS:

00337'000032   SB26:        26              ; NO. OF SECTORS PER TRACK;
00340'000370   FDF8:        7B10+3B12       ; HEX(F8);
00341'000373   FDFB:        7B10+3B12+3B15  ; HEx(FB);
```

```
↑↑↑
                ; DRIVER START AND BREAK:
00342'101122  FD01: MOVZL   0,0 SZC         ; BREAK: IF POWER BREAK THEN
00343'000404        JMP         FD05        ;    REPEAT THE LAST OPERATION;
00344'102400        SUB     0,0             ;
00345'041055        STA     0   FDDISP,2    ;  TRACK DISPLACEMENT:= 0;
00346'041030        STA     0   RESERVER,2; CURRENT RESERVER:= 0;
00347'102000  FD05: ADC     0,0             ;  TRACK ADDRESS:= UNDEFINED;
00350'041041        STA     0   TRACK,2     ;

                ; START OF PROGRAM:
00351'102400  FD10: SUB     0,0             ; BEGIN DRIVER ENTRY;
00352'041034        STA     0   FDSOFT,2    ;  STATUS:= 0;
00353'006164        NEXTOPERATION           ;  WAIT OPERATION;
00354'000471        JMP         FDCON       ;+0: CONTROL;
00355'000411        JMP         FDINP       ;+1: INPUT;
00356'004454        JSR         GETSLICE    ;+2: OUTPUT: GET THE SLICE SIZE;
00357'004440        JSR         GETBLOCK    ;  TEST FOR RANDOM OPERATION;
00360'101222        MOVZR   0,0 SZC         ;  IF OPERATION = WRITE/READ THEN
00361'000403        JMP         FD11        ;    GOTO WRITE/READ ELSE
00362'004724        JSR         WRNORMAL    ;    WRITE NORMAL;
00363'000412        JMP         FDEND       ;    RETURN ANSWER;

00364'004703  FD11: JSR         WRCHECK     ; WRITE/READ: WRITE/READ;
00365'000410        JMP         FDEND       ;    RETURN ANSWER;

00366'004444  FDINP:JSR         GETSLICE    ;  GET THE SLICE SIZE;
00367'004430        JSR         GETBLOCK    ;  TEST FOR RANDOM OPERATION;
00370'006743        JSR@        .READBLOCK; READ POSITIONED BLOCK;
00371'000404        JMP         FDEND       ;  RETURN ANSWER;

00372'035025  GIVEU:LDA     3   BUF,2       ; GIVEUP: BYTE COUNT := 0;
00373'021410        LDA     0   MESS2,3     ; ! I.E. ADDRESS.CUR:=
00374'041026        STA     0   ADDRESS,2   ;   MESS2.BUF !

      000000  FDEND:                       ; RETURN ANSWER:
00375'005076        JSR         IONIOP,2    ;  CLEAR DEVICE;
00376'035025        LDA     3   BUF,2       ;
00377'021042        LDA     0   BLOCK,2     ;
00400'101400        INC     0,0             ;  MESS3.BUF:=
00401'041411        STA     0   MESS3,3     ;    BLOCKCOUNT + 1;
00402'021034        LDA     0   FDSOFT,2    ;
00403'176400        SUB     3,3             ;
00404'024102        LDA     1   SOFFL       ;  IF STATUS(OFFLINE) THEN
00405'107404        AND     0,1 SZR         ;    TRACK DISPLACEMENT:= 0;
00406'055055        STA     3   FDDISP,2    ;
00407'024115        LDA     1   .1B12       ;  IF STATUS(POSITION) THEN
00410'107404        AND     0,1 SZR         ;
00411'024113        LDA     1   SPARITY     ;    MASK:= -,PARITY ELSE
00412'124000        COM     1,1             ;    MASK:= ALL ONES;
00413'123400        AND     1,0             ;  STATUS:= STATUS AND MASK;
00414'025055        LDA     1   FDDISP,2    ;  FILECOUNT:= TRACK DISPLACEMENT;
00415'006165        RETURNANSWER            ;  RETURN(MESS0.BUF = STATUS,
                                            ;         MESS1.BUF = BYTECOUNT,
                                            ;         MESS2.BUF = FILECOUNT(
                                            ;               TRACK DISPLACEMENT),
                                            ;         MESS3.BUF = BLOCKCOUNT);
00416'000733        JMP         FD10        ;  GOTO DRIVER ENTRY;
```

↑↑↑

```
                ; PROCEDURE GETBLOCK(OPERATION);
                ; IF OPERATION B(15) THEN BLOCKCOUNT.CUR:= MESS3.BUF - 1;
                ; THE INTENDED USE IS FOR RANDOM OPERATION, YET "SETPOSITION"
                ; USES THE PROCEDURE TOO.
                ;
                ; ENTRY GETBLOCK:
                ;  AC           CALL            RETURN
                ;  0          OPERATION     OPERATION SHIFT (-1)
                ;  1             -        IF OPERATION B(15) THEN
                ;                          MESS3.BUF ELSE UNCHANGED.
                ;  2            CUR             CUR
                ;  3            LINK          MESS2.BUF
                ;  FDMODE.CUR                OPERATION SHIFT (-1)
                ;
      000000    GETBLOCK:                     ; PROCEDURE GETBLOCK;
00417'055052          STA     3   WRLINK,2    ;
00420'035025          LDA     3   BUF,2       ;
00421'101223          MOVZR   0,0 SNC         ; BEGIN ! GETBLOCK !
00422'000405          JMP         FDP01       ;  IF CURRENT OPERATION B(15) THEN
00423'025411          LDA     1   MESS3,3     ;    BLOCKCOUNT(INTERNAL):=
00424'045042          STA     1   BLOCK,2     ;      MESS3.BUF - 1;
00425'015042          DSZ         BLOCK,2     ;
00426'000401          JMP         .+1         ;  OPERATION.CUR:=
00427'041036    FDP01:STA     0   FDMODE,2    ;    CURRENT OPERATION SHIFT (-1);
00430'035410          LDA     3   MESS2,3     ;
00431'003052          JMP@        WRLINK,2    ; END !GETBLOCK!


                ; PROCEDURE GETSLICE(COUNT);
                ;  CALCULATES THE SLICE SIZE:  SLICE:= COUNT//128;
                ;  IF COUNT MOD 128 <> 0 THE STATUS(BLOCKERROR)
                ;  IS SET, BUT THE OPERATION IS CONTINUED;
                ;
                ; ENTRY GETSLICE:
                ;  AC              CALL              RETURN
                ;  0             OPERATION         UNCHANGED
                ;  1             BYTECOUNT         DESTROYED
                ;  2               CUR               CUR
                ;  3               LINK              LINK
                ;
      000000    GETSLICE:                     ; BEGIN GET SLICE SIZE;
00432'055052          STA     3   WRLINK,2    ;
00433'041036          STA     0   FDMODE,2    ;
00434'121000          MOV     1,0             ;
00435'024111          LDA     1   .128        ;  SLICE SIZE:=
00436'006177          DIVIDE                  ;    BYTECOUNT//128;
00437'024111          LDA     1   SBLOC       ;
00440'175004          MOV     3,3 SZR         ;  IF BYTECOUNT MOD 128 <> 0 THEN
00441'006673          JSR@        .LOGOR      ;    STATUS(BLOCKERROR);
00442'041044          STA     0   SLICE,2     ;
00443'021036          LDA     0   FDMODE,2    ;
00444'007052          JSR@        WRLINK,2    ; END GET SLICE SIZE;
```

```
↑↑↑
                ; CONTROL:                      ; BEGIN CONTROL;

00445'006171  FDCON:SETRESERVATION              ;  SETRESERVATION;
00446'006172        SETCONVERTION               ;  SET CONVERTION ADDRESS;

00447'101220        MOVZR   0,0                 ; TERMINATION: NOT USED
00450'101223        MOVZR   0,0 SNC             ; POSITION: IF NOT POSITION THEN
00451'000412        JMP         CON40           ;    GOTO ERASE;
00452'101100        MOVL    0,0                 ;
00453'004744        JSR         GETBLOCK        ;  BLOCKCOUNT:= GETBLOCK(POSITION);
00454'174512        NEGL#   3,3 SZC             ;  IF FILECOUNT <= 0 THEN
00455'000406        JMP         CON40           ;    BEGIN !REDEFINE TRACK!
00456'160400        NEG     3,0                 ;
00457'024660        LDA     1   SB26            ;
00460'006177        DIVIDE                      ;      DISPLACEMENT:=
00461'041055        STA     0   FDDISP,2        ;        -(MESS2.BUF(FILECOUNT))//26;
00462'021036        LDA     0   FDMODE,2        ;    END !REDEFINE TRACK!

00463'101220  CON40:MOVZR   0,0                 ; ERASE:
00464'101222        MOVZR   0,0 SZC             ;    IF ERASE THEN
00465'004613        JSR         WRERASE         ;      ERASE BLOCK;

00466'006647        JSR@        .STACHECK       ;  SENSE STATUS; ALL CONTROL COMM.;
00467'000706        JMP         FDEND           ;  RETURN ANSWER;
                                                ; END CONTROL;
```

```
↑↑↑
                ; PROCESS DESRIPTOR

       000061   DEVNO=     49                  ; DEVICE NUMBER

  00470'000000  DE0:       0                   ; EXT PROCESS IN THE QUEUE
  00471'000000              0                   ; PREV. PROCESS
  00472'000000              0                   ; NEXT PROCESS IN CHAIN
  00473'000100              FD02-DE0            ; PROCESS DESCRIPTOR SIZE
  00474'043104  .TXT        .FD
  00475'030000  0<0>
  00476'000000  <0>.        ; NAME OF PROCESS
  00477'000477'             DE0+EVENT           ; FIRST EVENT
  00500'000477'             DE0+EVENT           ; LAST EVENT
  00501'000000              0                   ; FIRST MESSAGE BUFFER
  00502'000000'             FD00                ; PROGRAM ADDRESS
  00503'000000              0                   ; PROCESS STATE
  00504'000000              0                   ; TIMER COUNT
  00505'100000              1B0                 ; PRIORITY:= DRIVER
  00506'000342'             FD01                ; BREAK ADDRESS
  00507'000000              0                   ; SAVED AC0
  00510'000000              0                   ; SAVED AC1
  00511'000470'             DE0                 ; SAVED AC2
  00512'000470'             DE0                 ; SAVED AC3
  00513'000704"             FD01*2              ; PROCESS STATUS WORD (PSW)
  00514'000000              0         ;SAVE     ; WORK USED BY MONITOR UTILITIES
                ; OPTIONAL WORDS
  00515'000000              0         ;BUF      ; MESSAGE BUFFER ADDRESS
  00516'000000              0         ;ADDRESS  ; CURRENT BYTE ADDRESS OF SHARE
  00517'000000              0         ;COUNT    ; CURRENT VALUE OF COUNT
  00520'000000              0         ;RESERVER; RESERVER
  00521'000000              0         ;CONVT    ; CONVERTION TABLE ADDRESS
  00522'100166              CLEAR               ; STANDARD INTERRUPT CLEAR ADDRESS
```

```
↑↑↑
                ; OTHER OPTIONAL WORDS

       000033   TR.SYNC=   .-DE0
00523'000000              0          ;CUR+27; LAST READ(TRACK<8+SYNC.BYTE)
       000034   FDSOFT=    .-DE0
00524'000000              0          ;CUR+28; UNMODIFIED STATUS
       000035   DEVICE=    .-DE0
00525'000061              DEVNO              ; DEVICE ADDRESS
       000036   FDMODE=    .-DE0
00526'000000              0                  ; CURRENT MODE BITS
       000037   FDCOMMAND=.-DE0
00527'000000              0                  ; IO-COMMAND EXCL. (FDUNIT)
       000040   SECTOR=    .-DE0
00530'000000              0                  ; CALC. SECTOR NO.
       000041   TRACK=     .-DE0
00531'177777              -1                 ; CALC. TRACK NO.;-1 -> UNDEFINED
       000042   BLOCK=     .-DE0
00532'000000              0                  ; LOGICAL BLOCKCOUNT (=MESS3.BUF - 1)
       000043   FDBLOCK=   .-DE0
00533'000000              0                  ; PHYSICAL BLOCKCOUNT (INTERNAL)
       000044   SLICE=     .-DE0
00534'000000              0                  ; SLICE SIZE = BYTES//128
       000045   FDSLICE=   .-DE0
00535'000000              0                  ; SLICE COUNT: FOR EACH PHYSICAL BLOCK
                                             ;    DECREMENT FROM SLICESIZE TO ZERO
       000046   I.TLINK=   .-DE0
00536'000000              0                  ; RETURN(LOCAL TRANSPUT)
       000047   I.SLINK=   .-DE0
00537'000000              0                  ; RETURN(SLICE COUNT TEST)
       000050   POLINK=    .-DE0
00540'000000              0                  ; RETURN(POSITION)
       000051   STALINK=   .-DE0
00541'000000              0                  ; RETURN(TEST STATUS)
       000052   WRLINK=    .-DE0
00542'000000              0                  ; RETURN(WRITE,ERASE)
       000053   RELINK=    .-DE0
00543'000000              0                  ; RETURN(READBLOCK,WRITE/READ)
       000054   RESEM=     .-DE0
00544'000000              0                  ; COMMAND SPECIFICATION;
                                             ;  USED BY "STACHECK";
                                             ;  IF -1 THEN LAST COMMAND=READ BLOCK
                                             ;  ELSE ANY OTHER COMMAND;
       000055   FDDISP=    .-DE0
00545'000000              0                  ; TRACK DISPLACEMENT, USED FOR
                                             ;  DISCETTES WITH DELETED TRACKS;
```

↑↑↑

```
                ; ROUTINES INPUT/OUTPUT:
                ;
      000056    IODOAS=    .-DE0              ; CONTROL COMMAND:
00546'061161            DOAS    0    DEVNO     ;  START DEVICE(COMMAND);
00547'001400            JMP          +0,3      ;  RETURN(LINK);
                ;
      000060    IODOB=     .-DE0              ; OUTCHAR:
00550'062061            DOB     0    DEVNO     ;  OUTCHAR(CHAR,DEVNO);
00551'001400            JMP          +0,3      ;  RETURN(LINK);
                ;
      000062    IODOC=     .-DE0              ; OUTSYNC:
00552'063061            DOC     0    DEVNO     ;  OUTSYNC(FIELD SYNC. BYTE,DEVNO);
00553'001400            JMP          +0,3      ;  RETURN(LINK);
                ;
      000064    IODIC=     .-DE0              ; INSYNC:
00554'062461            DIC     0    DEVNO     ;  INSYNC(TRACK<8 + SYNC. BYTE,DEVNO);
00555'041033            STA     0    TR.SYNC,2 ;  SAVE, FOR SERVICE ONLY;
00556'001400            JMP          +0,3      ;  RETURN(LINK);
                ;
      000067    IODIB=     .-DE0              ; INCHAR:
00557'061461            DIB     0    DEVNO     ;  INCHAR(CHAR,DEVNO);
00560'001400            JMP          +0,3      ;  RETURN(LINK);
                ;
      000071    IODIA=     .-DE0              ; SENSE NORMAL:
00561'064461            DIA     1    DEVNO     ;  SENSE(STATUS);
00562'001400            JMP          +0,3      ;  RETURN(LINK);
                ;
      000073    IOSPEC=    .-DE0              ; REDEFINE TRACK ADDRESS:
00563'061061            DOA     0    DEVNO     ;  LOAD THE TRACK ADDRESS;
00564'001400            JMP          +0,3      ;  RETURN(LINK);
                ;
      000075    IOCLEAR=   .-DE0              ; CLEAR ALL:
00565'060261            NIOC         DEVNO     ;  CLEAR(DONE,BUSY);
                ;
      000076    IONIOP=    .-DE0              ; PULSE:
00566'060361            NIOP         DEVNO     ;  CLEAR BUFFER;
00567'001400            JMP          +0,3      ;  RETURN(LINK);
                ;
      000000    FD02:                         ; END OF PROCESS DESCRIPTOR;

                ;     *****   END OF FLOPPY DISC DRIVER   *****

      000470'   .END DE0
```

```
BLOCK    000042
CON40    000463'
DE0      000470'
DEVIC    000035
DEVNO    000061
FD00     000000'
FD01     000342'
FD02     000570'
FD05     000347'
FD10     000351'
FD11     000364'
FDBLO    000043
FDCOM    000037
FDCON    000445'
FDDIS    000055
FDEND    000375'
FDEXE    000102'
FDF6     000340'
FDFB     000341'
FDINP    000366'
FDMOD    000036
FDP01    000427'
FDPOS    000007'
FD[redacted]I    000045
FDSOF    000034
FDTIM    000154'
GETBL    000417'
GETSL    000432'
GIVEU    000372'
HARD1    000156'
HARD2    000155'
HMASK    000163'
IOCLE    000075
IODIA    000071
IODIB    000067
IODIC    000064
IODOA    000056
IODOB    000060
IODOC    000062
IONIO    000076
IOSPE    000073
ISLIC    000250'
I.[redacted]U    000257'
I.REA    000231'
I.SLI    000047
I.TLI    000046
I.WRI    000240'
LOGOR    000114'
POLIN    000050
POS05    000025'
POS10    000036'
POS11    000050'
POS16    000062'
POS20    000065'
POS26    000075'
POSER    000100'
RB00     000165'
RB05     000166'
RB22     000206'
RB24     000215'
```

```
RB26     000221'
RB30     000224'
READB    000164'
RECAL    000107
RELIN    000053
RESEM    000054
SB26     000337'
SECTO    000040
SLICE    000044
STA1     000145'
STA5     000146'
STACH    000123'
STALI    000051
TRACK    000041
TRSEA    000077'
TR.SY    000033
T.SLI    000262'
WR05     000310'
WR95     000317'
WRC05    000274'
WR[illegible]E    000267'
WR[illegible]RA   000300'
WRITE    000110
WRLIN    000052
WRNOR    000306'
.2002    000076'
.GIVE    000101'
.LOGO    000334'
.POSI    000336'
.POSN    000124
.READ    000333'
.STAC    000335'
```