```
 432   0212  0244            OPEN TEMPD      /TARGET TRACK IS OK, GET THE DRIVE
 433   0213  0071            ESP             /SELECT FROM TEMPD
 434   0214  0075            LSR
 435
 436   0215  0200            OPEN CURTK0     /PRESELECT UNIT 0
 437
 438   0216  0002            DISK            /SELECT DISK RUNS
 439
 440   0217  0122            BR SR7 ONE      /WHICH UNIT SELECTED? BIT7=0 MEANS UNIT ONE
 441   0220  0222            .+2             /ZERO, SKIP UNIT 1 SETUP
 442   0221  0204            OPEN CURTK1
 443
 444   0222  0071            ESP             /PASS SELECTED CURRENT TRACK TO MAGCOM SUBR
 445   0223  0075            LSR
 446   0224  0260            OPEN TEMPG
 447   0225  0064            LSP
 448
 449   0226  0220            OPEN TARTRK     /PASS TARGET TRACK TO MAGCOM SUBROUTINE
 450   0227  0071            ESP
 451   0230  0075            LSR
 452   0231  0254            OPEN TEMPF
 453   0232  0064            LSP
 454   0233  0070            LCT             /CALL SUBROUTINE MAGCOM TO SEE IF TARGET
 455   0234  0246            TRKEQ           /IS SAME AS CURRENT TRACK,  F=TARGET,  G=CURRENT
 456   0235  0075            LSR
 457   0236  0270            OPEN RTNA
 458   0237  0064            LSP
 459   0240  0226            JUMP F5
 460   0241  2400            MAGCOM
 461
 462
 463   0242  0070  ERTRK,    LCT
 464   0243  0040            KERTRK          /TRIED TO ACCESS A TRACK GREATER THAN 76 DECIMAL
 465   0244  0226            JUMP F5
 466   0245  2610            GOERDN
 467
 468
 469   0246  0202  TRKEQ,    JUMP F0         /TARGET EQUALS THE CURRENT TRACK, NO
 470   0247  0357            NOSTPS          /STEPS ARE REQUIRED
 471   0250  0270            OPEN RTNA       /NOOP;  TARGET > ACTUAL RETURN
 472   0251  0270            OPEN RTNA       /NOOP
 473
 474   0252  0270  BOOT,     OPEN RTNA       /TARGET IS LESS THAN ACTUAL, STEPS NEEDED ALSO START OF
 475   0253  0070            LCT             /OF BOOT SUBROUTINE.  SET UP RETURN FROM DIF SUBR
 476   0254  0275            STPOUT
 477   0255  0075            LSR
 478   0256  0064            LSP
 479
 480   0257  0244            OPEN TEMPD      /SOFT UNIT SELECT BIT TO SR7
 481   0260  0071            ESP
 482   0261  0075            LSR
 483
 484   0262  0204            OPEN CURTK1     /PRESELECT UNIT 1
 485
 486   0263  0126            BR SR7 ZERO     /SR7=0 MEANS UNIT ONE
```



```
 487   0264  0266            .+2
 488   0265  0200            OPEN CURTK0
 489
 490   0266  0071            ESP             /PASS SELECTED CURRENT TRACK TO DIFF SUBR VIA SR
 491   0267  0075            LSR
 492
 493   0270  0220            OPEN TARTRK     /PASS TARGET TRACK TO DIF VIA CNTR
 494   0271  0071            ESP
 495
 496   0272  0016            SET HDOUT       /ASSUME A STEP OUT
 497
 498   0273  0226            JUMP F5         /GO TO THE SUBROUTINE DIF TO CALCULATE THE STEPS NEEDED
 499   0274  2462            DIF
 500
 501
 502   0275  0202  STPOUT,    JUMP F0         /TARGET TRACK IS LESS THAN
 503   0276  0300            .+2             /THE ACTUAL, MOVE OUT IS NECESSARY
 504
 505   0277  0014            CLR HDOUT       /TARGET IS GREATER THAN ACTUAL.  STEPS IN NEEDED
 506
 507   0300  0070            LCT             /COMPLEMENT OF STEPS REQUIRED IS IN THE
 508   0301  0305            DUNSTP          /SHIFT REG.  SET UP RETURN FROM STPHD SUBR
 509
 510   0302  0040            UNHD            /UNLOAD HEAD BEFORE MOVING
 511
 512   0303  0222            JUMP F4         /CALL SUBROUTINE STEPHD
 513   0304  2100            STEPHD
 514
 515
 516   0305  0226  DUNSTP,    JUMP F5         /HOME FOUND BEFORE LAST STEP TAKEN
 517   0306  2456            HOMERR
 518
 519   0307  0244            OPEN TEMPD      /SOFT UNIT BIT TO SR7
 520   0310  0071            ESP
 521   0311  0075            LSR
 522   0312  0220            OPEN TARTRK     /GET READY TO PASS TARGET TRK TO PROPER
 523   0313  0071            ESP             /CURRENT TRACK
 524
 525   0314  0200            OPEN CURTK0     /OPEN PROPER CURRENT TRACK REGISTER
 526   0315  0122            BR SR7 ONE      /BIT7=0 MEANS UNIT ONE
 527   0316  0320            .+2
 528   0317  0204            OPEN CURTK1
 529
 530   0320  0075            LSR             /UPDATE THE CURRENT TRACK ADDRESS
 531   0321  0064            LSP
 532
 533
 534   0322  0220  HDSETL,    OPEN TARTRK     /HEAD IS SETTLED DETERMINE IF ABOVE TRACK 43 DECIMAL
 535   0323  0071            ESP             /PASS TARGET TO MAGCOM VIA TEMPF
 536   0324  0075            LSR
 537   0325  0254            OPEN TEMPF
 538   0326  0064            LSP
 539
 540   0327  0070            LCT             /PASS 44 TO MAGCOM VIA TEMPG
 541   0330  0323            -44-1
```

```
542   0331  0075          LSR
543   0332  0260          OPEN TEMPG
544   0333  0064          LSP
545
546   0334  0026          SET LOWCUR      /ASSUME TARGET GREATER THAN 43
547
548   0335  0070          LCT             /CALL MAGCOM SUBROUTINE
549   0336  0344          ABV43           /RETURN ADDRESS
550   0337  0075          LSR
551   0340  0270          OPEN RTNA
552   0341  0064          LSP
553   0342  0226          JUMP F5
554   0343  2400          MAGCOM
555
556
557   0344  0202  ABV43,  JUMP F0         /NOOP F=G RETURN, ABOVE TRK 43
558   0345  0346          .+1             /NOOP
559
560   0346  0202          JUMP F0         /F<G; ABOVE TRACK 43
561   0347  0351          .+2
562
563   0350  0024          CLR LOWCUR      /F>G;  BELOW TRACK 43,  WRITE WITH HIGH CURRENT
564
565   0351  0070  CFINSE, LCT             /CALL FINDSEC SUBROUTINE TO LOCATE THE DESIRED SECTOR
566   0352  0355          RFINTR
567   0353  0206          JUMP F1
568   0354  0714          FINDSE
569
570   0355  0274  RFINTR, OPEN RTN        /RETURN FROM FINDTR SUBROUTINE
571   0356  0207          JUMP F1  IND
572
573
574   0357  0250  NOSTPS, OPEN TEMPE      /NO STEPS REQUIRED
575   0360  0071          ESP             /SOFT HEAD LOAD BIT TO SR7
576   0361  0075          LSR
577
578   0362  0122          BR SR7 ONE      /IS HEAD LOADED?
579   0363  0322          HDSETL          /YES, GO UPDATE CURRENT CONTROL
580
581   0364  0070          LCT             /NO, GO LOAD HEAD AND WAIT FOR 20MS SETTLE TIME
582   0365  0322          HDSETL          /RETURN ADDR FROM DLY25 SUBROUTINE
583   0366  0222          JUMP F4
584   0367  2145          DLY25
585
586
587   0370  0212  PFUNCT, JUMP F2         /POINTER FROM GETWORD SUBROUTINE TO
588   0371  1236          FUNCT           /FUNCTION DECODE
589
590   0372  0226  PDNRCL, JUMP F5         /POINTER TO DRVA CHECK DONE AFTER RECALABRATE
591   0373  2625          DNRCAL
592
593   0374  0000          0               /SPARE LOCATIONS
594   0375  0000          0               /OPEN
595   0376  0000          0               /OPEN
596   0377  0000          0               /OPEN
```



```
597                   /[ROUTINE: WRITE SECTOR]
598                   /THIS ROUTINE TURNS ON WRITE GATE AT WRITE TURN ON TIME,
599                   /WRITES A PREAMBLE OF 6 BYTES OF ZEROES, A DATA OR DELETED DATA MARK,
600                   /THEN TURNS ON ERASE GATE.  ENTER WITH CNTR=100 IF
601                   /DELETED DATA, CNTR=0 IF NORMAL DATA MARK.  THE DATA MARK, DATA FIELD,  CRC
602                   /AND ONE BYTE POSTAMBLE ARE WRITTEN.  WRITE CURRENT IS TURNED OFF.
603                   /511 MICRO SECONDS LATER ERASE CURRENT IS TURNED OFF.  A HEADER MUST
604                   /THEN BE READ TO INSURE DISK IS STILL UP TO SPEED BEFORE  THE WRITE
605                   /SECTOR FUNCTION IS COMPLETE.
606
607
608
609
610   0400  0214  WRTSEC, OPEN STAT       /DEL DATA BIT TO STAT
611   0401  0075          LSR
612   0402  0264          LSP
613
614   0403  0270          LCT             /CALL SUBROUTINE TO FIND DESIRED TRACK AND SECTOR
615   0404  0407          S=GATE
616   0405  0282          JUMP F3
617   0406  0103          FINDTR
618
619   0407  0061  S=GATE, FLAG OFF        /ALWAYS START WRITING WITH WRITE FLOP CLEARED
620
621   0410  0140          BR WRTEN F      /GO REPORT ERROR IF NO WRITE ENABLE
622   0411  0503          PRTERR
623
624   0412  0214          OPEN STAT       /DEL DATA BIT TO SR7 AND ENABLE WRT CURRENT
625   0413  0071          ESP
626   0414  0406          SET WGATE
627   0415  0075          LSR
628   0416  0074          ROTATE ZERO
629
630   0417  0234          OPEN TEMPB      /USE TEMPB FOR SECOND HALF DATA AM PATTERN
631
632   0420  0057          PRECRC          /JAM THE CRC GENERATOR WITH FIRST 6 BITS OF DATA AM
633   0421  0056          CRC ONE
634   0422  0056          CRC ONE
635   0423  0056          CRC ONE
636   0424  0056          CRC ONE
637   0425  0056          CRC ONE
638   0426  0054          CRC ZERO
639
640   0427  0120          BR SR7 ZERO     /DELETED DATA?
641   0430  3460          DAMSUP          /NO, REGULAR DATA MARK
642
643   0431  0070          LCT             /YES, SECOND HALF OF DELETED DATA MARK TO CNTR
644                       OCTAL
645   0432  0325          325             /FLUX PATTERN
646                       DECIMAL
647
648   0433  0054          CRC ZERO        /JAM LAST 2 BITS OF DELETED DATA MARK TO CRC GEN.
649   0434  0054          CRC ZERO
650   0435  0002          DISK            /NOOP
651   0436  0002          DISK            /NOOP
```

```
                                
0437  0063  STASH,  TOG FLAG        /END OF THE FIRST 0 BIT
                                
0440  0075          LSR             /PUT SECOND HALF OF THE DESIRED MARK IN THE TEMPB
0441  0064          LSP
                                
0442  0070          LCT             /SET UP RETURN FROM WRITE ZEROS SUBROUTINE
0443  0466          HLFDLY
0444  0075          LSR
                                
0445  0070          LCT             /STALL 1.8 MICRO SECONDS
0446  0374          -3-1
0447  0073          ICT
0450  0124          BR COFL F
0451  0447          .-2
0452  0002          DISK            /NOOP
                                
0453  0070          LCT             /SPECIFY 22 ZEROS TO BE WRITTEN BY WRT0S SUBROUTINE
0454  0351          -22-1
                                
0455  0063          TOG FLAG        /WRITE SECOND CLOCK TRANSITION
                                
0456  0212          JUMP F2         /CALL WRITE ZEROS SUBROUTINE
0457  1322          WRT0S
                                
0460  0070  DAMSUP, LCT             /LOAD SECOND HALF OF NORMAL DATA MARK
                      OCTAL
0461  0337          337
                      DECIMAL
                                
0462  0056          CRC ONE         /JAM LAST 2 BITS OF DATA MARK TO CRC GENERATOR
0463  0056          CRC ONE
                                
0464  0206          JUMP F1         /GO PUT AWAY THE SECOND HALF OF THE DATA MARK
0465  0437          STASH
                                
0466  0002  HLFDLY, DISK            /NOOP
                                
0467  0070          LCT
0470  0514          WRTDAM          /SET UP RETURN FROM WRITE ZEROS SUBROUTINE
0471  0075          LSR
                                
0472  0070          LCT             /NOOP WASTE 4.8 MICRO SECONDS
0473  0351          -22-1           /NOOP
0474  0070          LCT             /NOOP
0475  0351          -22-1           /NOOP
                                
0476  0070          LCT             /SPECIFY 22 BITS TO BE WRITTEN BY WRT0S SUBROUTINE
0477  0351          -22-1
                                
0500  0063          TOG FLAG        /WRITE THE 25TH CLOCK TRANSITION
                                
0501  0212          JUMP F2         /CALL WRT0S SUBROUTINE
0502  1322          WRT0S
```



```
0503  0270  PRTERR, LCT             /SET WRITE PROTECT BIT OF STAT BECAUSE A WRITE FUNCTION WAS ATTEMPTED ON
                                    /OR A WRITE PROTECTED DISKETTE
                      OCTAL
0504  0010          10
                      DECIMAL
0505  0075          LSR
0506  0214          OPEN STAT
0507  0064          LSP
                                
0510  0070          LCT             /ERROR CODE FOR WRT PROTECT ERROR
0511  0100          KWPROT
0512  0226          JUMP F5
0513  2610          GOERDN
```

```
                    /THIS ROUTINE WILL WRITE EITHER A DATA MARK OR A
                    /DELETED DATA MARK. THE FIRST HALF OF BOTH MARKS ARE
                    /IDENTICAL. THE SECOND HALF IS SPECIFIED BEFORE ENTRY BY
                    /PUTTING THE SECOND HALF BIT PATTERN IN TEMPB
                                
0514  0070  WRTDAM, LCT             /WASTE 2.0 MICRO SECONDS
0515  0375          -2-1
0516  0073          ICT
0517  0075          LSR
0520  0124          BR COFL F
0521  0516          .-3
                                
0522  0063          TOG FLAG        /WRITE A CLOCK BIT AS END OF 48TH ZERO
                                
0523  2070          LCT             /FIRST HALF OF DATA MARK PATTERN TO SR
                      OCTAL
0524  0352          352
                      DECIMAL
0525  0075          LSR
                                
0526  0070          LCT             /SET TRANSITION LOOP COUNTER FOR 8 LOOPS
0527  0370          -7-1
0530  0002          DISK            /NOOP
                                
0531  0120  AGAIN,  SR SR7 ZERO     /WHATS THE BIT?
0532  0562          A               /ZERO, NO TRANSITION
                                
0533  0044          CLR BAR         /ONE, RESET THE BUFFER ADDR REG TO 0
                                
0534  0063          TOG FLAG        /WRITE FLUX TRANSITION
                                
0535  0126  ABACK,  BR COFL T       /CHECK TRANSITION LOOP COUNT
0536  0543          SECHLF          /GO GET SECOND HALF
                                
0537  0074          ROTATE          /SHIFT NEXT TRANSITION TO SR7
                                
0540  0073          ICT             /BUMP TRANSITION LOOP COUNTER
```

```
                                JUMP F1          /DO ANOTHER LOOP
                                AGAIN
                        SECHLF, OPEN TEMPB       /SECOND HALF OF DATA MARK TO SR
                                ESP
                                LSR
                                LCT              /SET TRANSITION LOOP COUNTER FOR 8 LOOPS
                                -7-1
                        AGAIN1, BR SR7 ZERO      /SHALL WE WRITE A TRANSITION?
                                B                /NO
                                TOG FLAG         /YES
                                DISK             /NOOP
                        BBACK,  BR COFL T        /DONE DATA MARK?
                                WRTDAT           /YES, GO WRITE DATA
                                ICT              /NO, BUMP THE LOOP COUNTER
                                ROTATE           /BRING UP NEXT HALF BIT TO SR7
                                JUMP F1          /DO ANOTHER LOOP
                                AGAIN1
                        A,      JUMP F1          /WASTE 2 CYCLES TO SKIP FLUX TRANSITION
                                ABACK
                        B,      JUMP F1          /WASTE 2 CYCLES TO SKIP FLUX TRANSITION
                                BBACK


                /THIS ROUTINE WRITES THE CONTENTS OF THE SECTOR BUFFER,

                        WRTDAT, SET EGATE        /TURN ON ERASE CURRENT AT START OF DATA FIELD
                                ICT              /NOOP; WASTE 2 CYCLES
                                ICT              /NOOP

                        DATAA,  BR BDATAO ZERO   /WHATS THE DATA BIT?
                                C                /ZERO, GO WRITE NOTHING

                                CRC ONE          /ONE, UPDATE THE CRC WITH 1

                                TOG FLAG         /WRITE A DATA TRANSITION
                                ICT              /NOOP FOR BIT CELL TIMING

                        CBACK,  BR BAROFL T      /DONE ENTIRE SECTOR?
                                WRTCRC           /YES, GO WRITE THE CRC
                                INCR BAR         /NO, BRING UP NEXT DATA BIT FROM SEC BUFFER
```



```
                                LCT              /NOOP - WASTE 5 CYCLES WITH
                                -2               /NOOP - A SELF TEST OF THE COUNTER
                                ICT              /NOOP
                                BR COFL F        /NOOP
                                SELFER           /NOOP

                                TOG FLAG         /WRITE A CLOCK TRANSITION

                                LCT              /NOOP - WASTE 4 CYCLES WITH
                                -1               /NOOP - A SELF TEST OF THE COUNTER
                                BR COFL F        /NOOP
                                SELFER           /NOOP

                                JUMP F1          /GO WRITE ANOTHER DATA BIT
                                DATAA

                        C,      CRC ZERO         /UPDATE CRC WITH 0 AND SKIP DATA TRANSITION
                                JUMP F1
                                CBACK


                        SELFER, LCT              /A SELF DIAGNOSTIC HAS FAILED
                                WSELFER
                                JUMP F5
                                GOERDN


                /THIS ROUTINE WRITES THE 16 BIT CRC GENERATED FOR THE
                /PRECEEDING DATA FIELD,

                        WRTCRC, LCT              /PRESET BIT COUNTER FOR 16 BITS
                                -16-1

                                LSR              /NOOP WASTE 4 CYCLES AND SELF TEST THE SR
                        E,      DISK             /NOOP
                                BR SR7 ZERO      /NOOP
                                SELFER           /NOOP

                                TOG FLAG         /WRITE A CLOCK TRANSITION

                                ROTATE ONE       /NOOP WASTE 6 CYCLES WITH MORE SELFTEST
                                ROTATE ONE       /NOOP
                                ROTATE ONE       /NOOP
                                ROTATE ONE       /NOOP
                                BR SR7 ZERO      /NOOP
                                SELFER           /NOOP

                                BR CRC16 ZERO    /WHAT IS THE CRC BIT
                                C                /ZERO, DO NOT WRITE ANYTHING

                                CRC ONE          /ONE, BRING UP THE NEXT BIT

                                TOG FLAG         /WRITE A DATA TRANSITION
```

```
                0645  0076          ROTATE ONE      /NOOP
                0646  0273  DBACK,  ICT             /BUMP THE BIT COUNTER
                0647  0126          BR COFL T       /DONE CRC YET?
                0650  0656          WRTPST          /YES, GO WRITE A POSTAMBLE
                0651  0206          JUMP F1         /NO, GO WRITE ANOTHER CRC BIT
                0652  0627          E
                0653  0054  D,      CRC ZERO        /BRING UP NEXT CRC BIT AND SKIP DATA TRANSITION
                0654  0206          JUMP F1
                0655  0646          DBACK

                      /THIS ROUTINE WRITES THE ONE BYTE POSTAMBLE, TURNS OFF
                      /WRITE CURRENT, DELAYS 511 MICRO SEC AND TURNS OFF ERASE
                      /CURRENT, IT UTILIZES THE WRITE ZEROES SUBROUTINE.

                0656  0070  WRTPST, LCT             /SETUP TO CALL WRT0S TO WRITE 8 BITS OF ZEROES
                0657  0666          CWGATE
                0660  0075          LSR
                0661  0070          LCT
                0662  0367          -8-1

                0663  0063          TOG FLAG        /WRITE LAST CLOCK TRANSITION OF THE CRC FIELD

                0664  0212          JUMP F2         /CALL THE SUBROUTINE WRITE ZEROES
                0665  1322          WRT0S

                0666  0004  CWGATE, CLR WGATE       /DISABLE WRITE CURRENT

                0667  0070          LCT             /CALL WRT0S FOR 127 BITS (511.2 MICRO SEC)
                0670  0676          CEGATE          /DELAY TO ERASE TURN OFF
                0671  0075          LSR
                0672  0070          LCT
                0673  0200          -127-1
                0674  0212          JUMP F2
                0675  1322          WRT0S

                0676  0020  CEGATE, CLR EGATE       /DISABLE ERASE CURRENT

                0677  0070          LCT             /CALL WRT0S FOR 25 BIT (101 MICRO SEC) DELAY
                0700  0706          READOK          /BEFORE TRYING TO READ
                0701  0075          LSR
                0702  0070          LCT
                0703  0346          -25-1
                0704  0212          JUMP F2
                0705  1322          WRT0S

                0706  0070  READOK, LCT             /CALL FIND HEADER ROUTINE TO INSURE
```



```
                0707  0712          GODONE          /THAT THE DISK IS STILL MOVING
                0710  0216          JUMP F3
                0711  1400          FINDHD

                0712  0212  GODONE, JUMP F2         /WRITE SECTOR FUNCTION IS COMPLETE
                0713  1006          OKDONE

                      /[SUBROUTINE: FINDSECTOR]
                      /SUBROUTINE TO FIND A SPECIFIC SECTOR.  ENTER WITH RETURN ADDRESS
                      /IN CNTR,  DESIRED TRACK ADDRESS IN TARTRK AND DESIRED SECTOR ADDRESS
                      /IN TARSEC.  THIS SUBROUTINE ASSUMES THAT THE TARGET TRACK HAS ALREADY
                      /BEEN REACHED.

                0714  0270  FINDSE, OPEN RTNA       /SAVE RETURN ADDRESS
                0715  0075          LSR
                0716  0064          LSP

                0717  0260          OPEN TEMPG      /PRESET SECTOR TRY COUNT TO 52 TRIES
                0720  0070          LCT
                0721  0313          -52-1

                0722  0075  AGAIN2, LSP             /STORE SECTOR TRY COUNT
                0723  0064          LSP

                0724  0070          LCT             /CALL SUBROUTINE TO FIND A HEADER
                0725  0730          CHKSEC
                0726  0216          JUMP F3
                0727  1400          FINDHD

                0730  0174  CHKSEC, BR FLAGO ZERO   /CORRECT SECTOR FLAG=1 IF NO
                0731  0743          WAIT            /YES, GO WAIT FOR PREAMBLE

                0732  0260          OPEN TEMPG      /NO, RECALL SECTOR TRY COUNT AND INCREMENT IT
                0733  0071          ESP
                0734  0073          ICT

                0735  0124          BR COFL F       /52 TRIES MADE FOR SECTOR YET?
                0736  0722          AGAIN2          /NO, TRY ANOTHER SECTOR

                0737  0270  NXHDR,  LCT             /YES, CANNOT FIND THE SECTOR
                0740  0070          KNXHDR
                0741  0226          JUMP F5
                0742  2610          GOERDN

                0743  0070  WAIT,   LCT             /STALL 323.2 MICRO SECONDS TO WAIT FOR DATA PREAMBLE
                0744  0345          -26-1
                0745  0073          ICT
                0746  0124          BR COFL F
                0747  0745          .-2
                0750  0073          ICT
                0751  0124          BR COFL F
                0752  0750          .-2
                0753  0073          ICT
```

```
 982   0754  0124          BR COFL F
 983   0755  0753          .-2
 984
 985   0756  0270          OPEN RTNA       /RETURN FROM THIS SUBROUTINE AT WRITE TURN ON TIME
 986   0757  0203          JUMP F0 IND     /OF THE DESIRED SECTOR
 987
 988
 989
 990                 /[ROUTINE: READ SECTOR]
 991
 992   0760  0074    RDSEC,  ROTATE ZERO     /ZERO THE STAT
 993   0761  0074            ROTATE ZERO
 994   0762  0214            OPEN STAT
 995   0763  0064            LSP
 996
 997   0764  0070            LCT             /CALL THE FIND TRACK SUBROUTINE TO LOCATE DESIRED SECTOR
 998   0765  0770            GOREAD
 999   0766  0202            JUMP F0
1000   0767  0103            FINDTR
1001
1002   0770  0222    GOREAD, JUMP F4         /GO READ THE DATA FIELD
1003   0771  2167            READ
1004
1005
1006   0772  0000            0               /OPEN  FREE LOCATIONS
1007   0773  0000            0               /OPEN
1008   0774  0000            0               /OPEN
1009   0775  0000            0               /OPEN
1010   0776  0000            0               /OPEN
1011   0777  0000            0               /OPEN
```



```
1012                /[ROUTINE: DONE AND ERROR DONE]
1013
1014
1015   1000  0220    ERDONE, CLR DONE
1016   1001  7010            CLR XREG
1017
1018   1002  0402            INTERF          /SELECT INTERFACE BUSS
1019
1020   1003  0006            SET ERR         /ASSERT ERROR LINE
1021
1022   1004  0212            JUMP F2         /SKIP NEXT INSTRUCTION
1023   1005  1007            .+2
1024
1025   1006  0204    OKDONE, CLR ERR         /NEGATE ERROR LINE
1026
1027   1007  0214            OPEN STAT       /OPEN STAT TO MOVE TO INTERFACE
1028
1029   1010  2271            ESP             /STAT OR ERREG TO SR
1030   1011  2275            LSR
1031
1032   1012  0024            CLR SHIFT       /CLEAR INTERFACE OUTPUT BUFFER
1033   1013  0020            CLR DONE
1034   1014  0010            CLR XREQ
1035
1036   1015  0400            INTERF          /SELECT INTERFACE OUTPUT BUSS
1037
1038   1016  0230            CLR SECDAT      /SELECT SR AS DATA LINE SOURCE
1039
1040   1017  7016            SET IOOUT       /DEFINE DATA DIRECTION AS OUT (TO INTERFACE)
1041
1042   1020  2070            LCT             /MOVE SR TO INTERFACE SERIALLY
1043   1021  0367            -8-1
1044   1022  0026            SET SHIFT
1045   1023  0024            CLR SHIFT
1046   1024  0073            ICT
1047   1025  0074            ROTATE ZERO
1048   1026  2124            BR COFL F
1049   1027  1022            .-5
1050
1051   1030  0014            CLR IOOUT       /NEXT TRANSFER WILL BE FROM INTERFACE
1052
1053   1031  0022    STDONE, SET DONE        /FUNCTION IS DONE
1054   1032  2070            LCT             /CALL GET COMMAND SUBROUTINE TO GET NEXT FUNCTION
1055   1033  0370            PFUNCT
1056   1034  0222            JUMP F4
1057   1035  2001            GETCMD
1058
1059   1036  0074    FUNCT,  ROTATE          /MOVE UNIT SELECT BIT TO SR7
1060   1037  0074            ROTATE
1061   1040  0074            ROTATE
1062   1041  0122            BR SR7 ONE      /FLAG IS ALREADY SET,  SAVE UNIT IN FLAG, ON=UNIT 0
1063   1042  1044            .+2
1064   1043  0061            FLAG OFF
1065
1066   1044  2074            ROTATE          /GET FIRST FUNCTION BIT TO SR7
```

```
1067
1068    1045  0120          BR SR7 ZERO
1069    1046  1066          FUNCT4          /FUNCTION 4 OR GREATER
1070
1071    1047  0074          ROTATE          /GET 2ND FUNCTION BIT
1072
1073    1050  0120          BR SR7 ZERO
1074    1051  1057          FUNCT2          /FUNCTION CODE IS 2 OR 3
1075
1076
1077    1052  0074          ROTATE          /GET LAST FUNCTION BIT
1078
1079    1053  0120          BR SR7 ZERO
1080    1054  1107          EMPTYBUF        /FUNCTION CODE 1
1081
1082    1055  0212          JUMP F2         /FUNCTION CODE 0
1083    1056  1110          FILLBUF
1084
1085    1057  0074  FUNCT2, ROTATE          /GET LAST FUNCTION BIT
1086
1087    1060  0120          BR SR7 ZERO
1088    1061  1105          PRDSEC          /FUNCTION CODE 3
1089
1090    1062  0070          LCT             /CLR CNTR BIT6 TO INDICATE NORMAL DATA
1091    1063  0000          0
1092    1064  0206          JUMP F1         /FUNCTION 2
1093    1065  0400          WRTSEC
1094
1095    1066  0074  FUNCT4, ROTATE          /GET 2ND FUNCTION BIT
1096
1097    1067  0120          BR SR7 ZERO
1098    1070  1076          FUNCT6          /FUNCTION CODE IS 6 OR GREATER
1099
1100    1071  0074          ROTATE          /GET LAST FUNCTION BIT
1101
1102    1072  0120          BR SR7 ZERO
1103    1073  1224          RDSTAT          /FUNCTION 5
1104
1105    1074  0212          JUMP F2
1106    1075  1243          CLRID           /FUNCTION 4=UNUSED
1107
1108    1076  0074  FUNCT6, ROTATE          /GET LAST FUNCTION BIT
1109
1110    1077  0120          BR SR7 ZERO
1111    1100  1275          RDEREG          /FUNCTION 7
1112
1113    1101  0070          LCT             /SET CNTR6 TO INDICATE DELETED DATA
1114                        OCTAL
1115    1102  0140          100
1116                        DECIMAL
1117    1103  0206          JUMP F1
1118    1104  0400          WRTSEC          /FUNCTION 6
1119
1120    1105  0206  PRDSEC, JUMP F1          /POINTER TO READ SECTOR FUNCTION
1121    1106  0760          RDSEC
```



```
1122
1123
1124
1125
1126
1127
1128
1129                /[ROUTINE: FILL AND EMPTY BUFFER]
1130
1131
1132    1107  0216  EMPTYBUF,SET IODUT       /IODUT IS CLEARED,  SET IT TO INDICATE DATA IS
1133                                          /MOVING TO THE INTERFACE
1134
1135    1110  0074  FILLBUF,ROTATE ZERO      /CLEAR STAT
1136    1111  0374          ROTATE ZERO
1137    1112  0214          OPEN STAT
1138    1113  0764          LSP
1139
1140    1114  0210          OPEN ERREG        /CLEAR ERREG
1141    1115  0764          LSP
1142
1143    1116  0761          FLAG OFF          /NOOP
1144
1145    1117  0044          CLR BAR SHORT     /ADDRESS THE 1ST BIT OF SECTOR BUFFER
1146
1147    1120  0070          LCT               /SET UP BYTE COUNT TO 128 (8 BIT) OR 64 (12 BIT)
1148    1121  0177          -128-1
1149    1122  0152          BR XIIBIT F
1150    1123  1126          .+3
1151    1124  0070          LCT
1152    1125  0277          -64-1
1153    1126  0230          OPEN TEMPA
1154
1155    1127  0106          BR IOB3OT T       /WHICH FUNCTION IS THIS?
1156    1130  1210          EMPTY1            /EMPTYBUF
1157
1158    1131  0012  XFRQ,   SET XREQ          /REQUEST DATA TRANSFER
1159
1160    1132  0073          ICT               /INCREMENT BYTE COUNT AND RESTORE
1161    1133  0075          LSR
1162    1134  0764          LSP
1163
1164    1135  0070          LCT               /CALL WAITRUN SUBR TO WAIT FOR DATA TRANSFER
1165    1136  1141          NEWORD
1166    1137  0222          JUMP F4
1167    1140  2312          WAITRN
1168
1169    1141  0230  NEWORD, OPEN TEMPA       /REOPEN THE BYTE COUNT REGISTER BECAUSE WAITRUN CLOSED IT
1170    1142  0070          LCT               /SET UP BIT COUNT IN CNTR TO 8 BITS OR 12 BITS
1171    1143  0367          -8-1
1172    1144  0150          BR XIIBIT F
1173    1145  1150          .+3
1174    1146  0070          LCT
1175    1147  0363          -12-1
1176
```

```
1177    1150  0104          BR IOB3OT F     /WHICH FUNCTION IS THIS?
1178    1151  1175          FILL1           /FILLBUF
1179
1180
1181    1152  0026  BYTEOUT,SET SHIFT       /EMPTYBUF,  MOVE A BYTE FROM SECTOR BUFFER
1182    1153  0046          INCR BAR        /TO INTERFACE SERIALLY
1183    1154  0024          CLR SHIFT
1184    1155  0073          ICT
1185    1156  0124          BR COFL F
1186    1157  1152          BYTEOUT
1187
1188    1160  0071          ESP             /CHECK BYTE COUNT
1189    1161  0124          BR COFL F
1190    1162  1131          XFRQ            /NOT DONE, GO REQUEST A DATA TRANSFER
1191
1192    1163  0012          SET XREQ        /DONE, REQUEST TRANSFER OF LAST BYTE
1193
1194    1164  0100          BR RUN F        /WAIT FOR TRANSFER COMPLETION
1195    1165  1164          .-1
1196
1197    1166  0010          CLR XREQ
1198
1199    1167  0212          JUMP F2         /EMPTYBUF FUNCTION IS COMPLETE
1200    1170  1006          OKDONE
1201
1202    1171  0050          FIN WRTBUF      /END SECTOR BUFR WRT PULSE (800 NS)
1203
1204    1172  0046          INCR BAR        /ADDRESS NEXT CELL OF SECTOR BUFFER
1205
1206    1173  0026          SET SHIFT       /SHIFT NEXT BIT FROM INTERFACE
1207    1174  0024          CLR SHIFT
1208
1209    1175  0053  FILL1,  START WRTBUF    /START SECTOR BUFR WRT PULSE
1210
1211    1176  0073          ICT             /LAST BIT OF BYTE?
1212    1177  0124          BR COFL F
1213    1200  1171          .-7             /NO, DO ANOTHER BIT
1214
1215    1201  0050          FIN WRTBUF      /LAST BIT, END SECTOR BUFR WRT PULSE
1216
1217    1202  0046          INCR BAR        /ADDRESS NEXT CELL OF SECTOR BUFFER
1218
1219    1203  0071          ESP             /CHECK BYTE COUNT
1220    1204  0124          BR COFL F
1221    1205  1131          XFRQ            /NOT DONE, GO GET ANOTHER BYTE
1222
1223    1206  0212          JUMP F2         /DONE FILLBUF FUNCTION
1224    1207  1006          OKDONE
1225
1226    1210  0032  EMPTY1, SET SECDAT      /SELECT SECTOR BUFR AS DATA LINE SOURCE
1227
1228    1211  0073          ICT             /INCREMENT AND SAVE THE BYTE COUNT
1229    1212  0075          LSR
1230    1213  0064          LSP
1231
```



```
1232    1214  0078          LCT             /SET UP THE BIT COUNT TO 8 BITS OR 12 BITS
1233    1215  0367          -8-1
1234    1216  0153          BR XIIBIT F
1235    1217  1152          BYTEOUT
1236    1220  0070          LCT
1237    1221  0363          -12-1
1238
1239    1222  0212          JUMP F2         /GO MOVE A BYTE TO INTERFACE
1240    1223  1152          BYTEOUT
1241
1242
1243
1244                /[ROUTINE: READ STATUS]
1245
1246
1247    1224  0244  RDSTAT, OPEN TEMPD      /SELECT THE SOFT UNIT SCRATCH PAD
1248
1249    1225  0036          UNIT ONE        /PRESELECT UNIT ONE
1250    1226  0073          LCT
1251    1227  0000          0
1252
1253    1230  0174          BR FLAGO ZERO   /WHICH UNIT?  FLAG=0=UNIT 1
1254    1231  1235          .+4             /UNIT 1,  SKIP UNIT 0 SETUP
1255
1256    1232  0034          UNIT ZERO       /SELECT UNIT ZERO
1257    1233  0070          LCT
1258                        OCTAL
1259    1234  0200          200
1260                        DECIMAL
1261
1262    1235  0075          LSR             /STORE SOFT UNIT BIT
1263    1236  0064          LSP
1264
1265    1237  0077          LCT             /CALL CHECKRDY SUBROUTINE, RETURN TO CLRID
1266    1240  1765          PNTRDY
1267    1241  0226          JUMP F5
1268    1242  2640          CHKRDY
1269
1270
1271
1272    1243  0214  CLRID,  OPEN STAT       /CLEAR INIT DONE BIT OF STAT
1273    1244  0071          ESP             /STATUS TO SHIFT REG
1274    1245  0075          LSR
1275
1276    1246  0061          FLAG OFF
1277    1247  0070          LCT             /END AROUND SHIFT OF FIRST 5 BITS
1278    1250  0372          -5-1
1279    1251  0122  ROT,    BR SR7 T
1280    1252  1256          .+4
1281    1253  0074          ROTATE ZERO
1282    1254  0212          JUMP F2
1283    1255  1257          .+2
1284    1256  0076          ROTATE ONE
1285    1257  0073          ICT
1286    1260  0124          BR COFL F
```

```
1287    1261  1251          .=8
1288
1289    1262  0176          BR FLAGO T      /IF FLAG IS SET THEN ROTATE IS DONE
1290    1263  1272          GODUN
1291
1292    1264  0062          FLAG ON         /IF NOT, CLEAR INIT DONE AND FINISH ROTATE
1293    1265  0074          ROTATE ZERO
1294    1266  0070          LCT
1295    1267  0375          =2-1
1296    1270  0212          JUMP F2
1297    1271  1251          ROT
1298
1299
1300    1272  0064  GODUN,  LSP             /RESTORE STAT AND GO DONE
1301    1273  0212          JUMP F2
1302    1274  1006          OKDONE
1303
1304                /[ROUTINE: READ ERROR REGISTER]
1305
1306
1307
1308    1275  0210  RDEREG, OPEN ERREG
1309    1276  0212          JUMP F2
1310    1277  1010          OKDONE+2
1311
1312
1313                /[SUBROUTINE: DELAY]. THIS SUBROUTINE PROVIDES DELAYS IN MULTIPLES
1314                /OF .1MS. ENTER WITH RETURN ADDRESS IN THE SHIFT REG.
1315                /AND MULTIPLIER IN THE COUNTER
1316
1317
1318    1300  0264  DELAY,  OPEN RTNB        /SAVE THE RETURN ADDRESS
1319    1301  0064          LSP
1320
1321    1302  0075          LSR             /MULTPLIER TO SHIFT REGISTER
1322
1323    1303  0070          LCT             /DELAY 490 CYCLES (98 MICRO SECONDS)
1324    1304  0205          =122-1
1325    1305  0073          ICT
1326    1306  0264          OPEN RTNB
1327    1307  0124          BR COFL F
1328    1310  1305          .-3
1329
1330    1311  0071          ESP             /MOVE MULTIPLIER TO CNTR VIA RTNB
1331    1312  0064          LSP
1332    1313  0075          LSR
1333    1314  0071          ESP
1334    1315  0064          LSP
1335
1336    1316  0073          ICT             /INCREMENT THE MULTIPLIER
1337
1338    1317  0124          BR COFL F       /ANY MORE .1MS LOOPS?
1339    1320  1301          DELAY+1         /YES, GO TO IT
1340
1341    1321  0223          JUMP F4 IND     /NO, RETURN FROM SUBROUTINE
```



```
1342
1343                /[SUBROUTINE: WRITE ZEROES]
1344                /THIS SUBROUTINE WRITES A SPECIFIED NUMBER OF ZEROS IF
1345                /WRITE GATE IS ON, IF WRITE GATE IS OFF IT ACTS AS A
1346                /DELAY OF N.5 BITS, ENTRANCE IS MADE WITH RETURN ADDRESS
1347                /IN THE SR, NUMBER OF BITS IN THE CNTR, AND A CLOCK
1348                /TRANSITION OCCURRING IMMEDIATLY PRIOR TO THE JUMP INTO
1349                /THIS SUBROUTINE.
1350
1351
1352    1322  0274  WRTZS,  OPEN RTN         /SAVE RETURN ADDRESS
1353    1323  0064          LSP
1354
1355    1324  0075          LSR             /PUT BIT COUNTER IN SR
1356
1357    1325  0230          OPEN TEMPA      /TEMPA IS THE PATH THROUGH THE SP
1358
1359    1326  0070  LOOP,   LCT             /STALL 2,6 MICRO SECONDS
1360    1327  0374          =3-1
1361    1330  0073          ICT
1362    1331  0124          BR COFL F
1363    1332  1330          .-2
1364    1333  0064          LSP             /NOOP
1365    1334  0071          ESP             /NOOP
1366
1367    1335  0063          TOG FLAG        /WRITE A CLOCK TRANSITION IF WRT GATE IS SET
1368
1369    1336  0064          LSP             /PUT BIT COUNT IN THE COUNTER
1370    1337  0071          ESP
1371
1372    1340  0073          ICT             /INCREMENT BIT COUNT
1373
1374    1341  0075          LSR             /PUT UPDATED BIT COUNT BACK IN SR
1375
1376    1342  0124          BR COFL F       /DONE ALL BITS?
1377    1343  1326          LOOP            /NO
1378
1379    1344  0274          OPEN RTN        /YES, RETURN FROM SUBROUTINE
1380    1345  0207          JUMP IND F1
1381
1382
1383    1346  0222  PGOTIT, JUMP F4         /POINTER TO GETWORD FROM WAITRUN
1384    1347  2010          GOTIT
1385
1386
1387                /[ROUTINE: INITIALIZE CONT.]
1388
1389    1350  0061  TEST2,  FLAG OFF        /CLEAR FLAG TO INDICATE R10 IS BEING TESTED
1390
1391    1351  0070  TEST1,  LCT             /LOOP TO TEST THAT SR IS 252 AND THAT
1392    1352  0372          =5-1            /IT CAN BE SHIFTED,
1393    1353  0122  TSTAGN, BR SR7 ZERO
1394    1354  1374          INTER1          /TEST FAILURE
1395    1355  0076          ROTATE ONE
1396    1356  0122          BR SR7 ONE
```

```
1397    1357  1374            INTER1          /TEST FAILURE
1398    1360  0074            ROTATE ZERO
1399    1361  0073            ICT
1400    1362  0124            BR COFL F
1401    1363  1353            TSTAGN
1402
1403    1364  0250            OPEN R10        /CONTENTS OF R10 TO SR, SHOULD BE 125
1404    1365  0071            ESP
1405    1366  0075            LSR
1406
1407    1367  0074            ROTATE ZERO     /SHIFT SR ONCE TO CHANGE 125 TO 252
1408
1409    1370  0176            BR FLAGO ONE    /HAS R10 BEEN TESTED ALREADY?
1410    1371  1350            TEST2           /NO
1411
1412    1372  0202    TESTDN, JUMP F0         /YES, RETURN TO REMAINING INITIALIZE ROUTINE
1413    1373  0004            TSTRTN
1414
1415    1374  0006    INTER1, SET ERR         /SELF TEST ERROR, SET ERROR AND GO SET DONE
1416    1375  0212            JUMP F2
1417    1376  1031            STDONE
1418
1419    1377  7000            0               /OPEN.
```



```
1420                    /[SUBROUTINE: FINDHEADER AND FIND DATA ADDRESS MARK]
1421                    /SUBROUTINE TO LOCATE A LEGAL HEADER (CORRECT CRC AND TRACK #)
1422                    /ENTER WITH THE RETURN ADDRESS IN CNTR.  ALSO ROUTINE TO FIND A DATA MARK
1423                    /OR DELETED DATA MARK.
1424
1425                                            /THIS ROUTINE LOCATES A SIX BYTE PREAMBLE OF ZEROES.
1426
1427
1428    1400  0264    FINDHD, OPEN RTNB       /STORE RETURN ADDRESS
1429    1401  0075            LSR
1430    1402  0264            LSP
1431
1432    1403  0232            OPEN TEMPA      /256 TO BAD START INNER COUNT
1433    1404  0070            LCT
1434    1405  0377            -1
1435    1406  0075            LSR
1436    1407  0264            LSP
1437
1438    1410  0234            OPEN TEMPB      /3 TO CNTR FOR BAD START OUTER COUNT.  768 BAD STARTS ALLOWED
1439    1411  0072            LCT
1440    1412  0374            -3-1
1441
1442    1413  0075    TRYAGN, LSR             /RESTORE BAD START COUNT
1443    1414  0264            LSP
1444
1445    1415  0045            CLR BAR LONG    /RESET FOR A COUNT OF 4096 AS PREAMBLE FAILURE COUNT
1446
1447    1416  0240            OPEN TEMPC      /24 TO CNTR AS ZERO BIT COUNT
1448    1417  0073            LCT
1449    1420  0347            -24-1
1450    1421  0075    MOREØS, LSR             /RESTORE ZERO BIT COUNT
1451    1422  0064            LSP
1452
1453    1423  0070            LCT             /PUT 0 IN SR7 FOR DATA COMPARISONS, ALSO CONSTANT FOR 40 MICRO SEC WAIT BRANCH
1454    1424  0067            -200-1
1455    1425  0075            LSR
1456
1457    1426  0346            *BR SEPCLK T    /WAIT 40 MICRO SECONDS FOR SEP CLK
1458    1427  1432            .+3
1459
1460    1430  0216            JUMP F3         /ERROR,NO SEP CLK
1461    1431  1667            TIMERR
1462
1463    1432  0154            BR DEQSR7 F     /WHAT IS SEP DATA?
1464    1433  1746            NOZERO          /ONE, GO CHECK PREAMBLE FAILURES
1465
1466    1434  0071            ESP             /ZERO FOUND, CHECK ZERO COUNT
1467    1435  0073            ICT
1468    1436  0124            BR COFL F
1469    1437  1421            MOREØS          /NEED MORE ZEROES FOR PREAMBLE
1470    1440  0061            FLAG OFF        /FOUND PREAMBLE, CLR FLAG TO INDICATE SEARCH FOR IDAM
1471
1472    1441  0045    GETDAM, CLR BAR LONG    /START SEARCH FOR IDAM OR DATA AM, BAR IS NOSTART COUNTER
1473
1474    1442  0070            LCT             /WAIT 40 MICRO SEC FOR SEP CLK
```