

.REM \*

1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45

IDENTIFICATION

PRODUCT CODE:	AC AROOF MC
PRODUCT NAME:	CZTEDEO TMO3 TE16/TO77 DATA RELIABILITY PROGRAM
DATE CREATED:	22 FEBRUARY 1984
MAINTAINER:	TAPE DIAGNOSTIC GROUP
AUTHOR:	J. HITT

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION. DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT MAY APPEAR IN THIS MANUAL.

THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED TO THE PURCHASER UNDER A LICENSE FOR USE ON A SINGLE COMPUTER SYSTEM AND CAN BE COPIED (WITH INCLUSION OF DIGITAL'S COPYRIGHT NOTICE) ONLY FOR USE IN SUCH SYSTEM, EXCEPT AS MAY OTHERWISE BE PROVIDED IN WRITING BY DIGITAL.

DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL.

COPYRIGHT (c) 1977, 1984 BY DIGITAL EQUIPMENT CORPORATION

TABLE OF CONTENTS

PARAGRAPH	SUBJECT	PAGE
1.	ABSTRACT	3
2.	REQUIREMENTS	3
3.	LOADING PROCEDURE	3
4.	STARTING PROCEDURE	4
5.	DATA PATTERNS	11
6.	RANDOMIZATION	12
7.	DYNAMIC PARAMETERS	13
8.	CONSOLE SWITCH	14
9.	ERROR PRINTOUTS	19
10.	STATISTICS PRINTOUT	27
11.	AUTO SEQUENCE	28
12.	TESTING PROCEDURES	30
13.	LISTING	32

44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84  
85  
86  
87  
88  
89  
90  
91  
92  
93  
94  
95  
96  
97  
98  
99  
100

689  
690  
691  
692  
693  
694  
695  
696  
697  
698  
699  
700  
701  
702  
703  
704  
705  
706  
707  
708  
709  
710  
711  
712  
713  
714  
715  
716  
717  
718  
719  
720  
721  
722  
723  
724  
725  
726  
727  
728  
729  
730  
731  
732  
733  
734  
735  
736  
737  
738  
739  
740  
741  
742

SW4:

SWITCH FOUR (4), WHEN SET TO A ONE (1), WILL CAUSE ANY DATA RELATED ERROR TO BE RETRIED. THE WRITE RETRY SCHEME CONSISTS OF REWRITING THE RECORD IN THE SAME SPOT ON TAPE FOUR (4) TIMES. IF ALL FOUR (4) REPEATS ARE SUCCESSFUL, THE RECORD IS CONSIDERED AS RECOVERED, AND A TAPE WRITE ERROR IS LOGGED. IF ANY OF THE FOUR (4) REPEATS IS UNSUCCESSFUL, A SKIP ERASE IS DONE, A SUSPECTED BAD TAPE SPOT IS LOGGED AT THIS BLOCK AND RECORD NUMBER, AND A SECOND RETRY OF FOUR REPEATS IS DONE. IF AFTER FOUR (4) RETRIES, THE RECORD CANNOT BE RECOVERED A NOTIFICATION IS PRINTED, AND TESTING IS RESUMED ON THE NEXT RECORD. IF 20(8) BAD TAPE SPOTS ARE FOUND, THE SLAVE WILL BE REWOUND AND REMOVED FROM TESTING WITH AN APPROPRIATE MESSAGE PRINTED. THE READ RETRY SCHEME CONSISTS OF REREADING THE RECORD UP TO EIGHT TIMES. IF ALL EIGHT REREADS ARE BAD, IT IS A HARD ERROR. IF ANY REREAD IS SUCCESSFUL, THIS IS A SOFT ERROR. IF THE ORIGINAL ERROR IS OF THE NON-RETRYABLE TYPE (IE: ILF,RMR,ILR,NEF,CBUSPE), THE RETRY SCHEME IS NOT ENTERED AND A MESSAGE IS PRINTED.

SW5:

SWITCH FIVE (5) WHEN SET DURING A READ FORWARD OR REVERSE WILL CAUSE THE TAPE TO CONTINUOUSLY READ THE CURRENT RECORD BY SPACING EITHER FORWARD OR REVERSE AND REREADING THAT RECORD. THIS TAPE MOVEMENT IS CALLED YOZZLING. THERE IS A SOFTWARE DELAY EXECUTED BETWEEN EACH SPACE/READ OF THE RECORD AND IT MAY BE VARIED BY TYPING CONTROL C ON THE TELETYPE DURING THE EXECUTION OF THE YOZZLE AND RESPONDING TO THE PRINTED REQUEST WITH A SIX (6) DIGIT VALUE. THE YOZZLE STALL IS PRESET TO A VALUE OF 3000 IN THE PROGRAM TO PREVENT EXCESSIVE TAPE WEAR, BUT MAY BE SET TO ANY VALUE THROUGH THE TELETYPE.

SW6-8:

THESE THREE (3) SWITCHES CONTROL THE RANDOMIZATION OF DATA AND BLOCK SIZE AND MAY BE SET AND RESET AT ANY TIME. THE ACTUAL CHANGE WILL TAKE PLACE BETWEEN BLOCK CYCLES.

SW9:

SWITCH NINE (9) WHEN SET WILL CAUSE ALL AVAILABLE TAPE UNITS TO BE REWOUND AT THE END OF THE CURRENT BLOCK CYCLE. TESTING WILL BE RESUMED AT A BLOCK COUNT OF ONE (1) WHEN ALL UNITS HAVE REACHED BOT.

744  
745  
746  
747  
748  
749  
750  
751  
752  
753  
754  
755  
756  
757  
758  
759  
760  
761  
762  
763  
764  
765  
766  
767  
768  
769  
770  
771  
772  
773  
774  
775  
776  
777  
778  
779  
780  
781  
782  
783  
784  
785  
786  
787  
788  
789  
790

SW10-13:

THESE SWITCHES ARE USED TO CONTROL THE  
ERROR HANDLING TO BE DONE ON THE TAPE  
OPERATION DESCRIBED BY SWITCHES 0-3.

- A. SWITCH TEN (10) WHEN SET TO A ONE  
WILL DISALLOW ANY ERROR PRINTOUTS MADE  
ON THE OPERATION IN PROGRESS. CATASTROPHIC  
FAILURES AND INFORMATION PRINTOUTS WILL  
STILL OCCUR. IE: UNIT NOT AVAILABLE, ILLEGAL  
BOT, DROP OR PICK OVERFLOW, AND EOT REWIND.
- B. SWITCH ELEVEN (11) WHEN SET TO A ONE  
WILL DISALLOW THE CHECKING FOR STATUS  
ERRORS ON READ (FORWARD OR REVERSE) OPERATIONS.
- C. SWITCH TWELVE (12) WHEN SET TO A ONE  
WILL DISALLOW THE CHECKING FOR STATUS  
ERRORS ON WRITE OPERATIONS.
- D. SWITCH THIRTEEN (13) WHEN SET TO A ONE  
WILL DISALLOW THE CHECKING OF READ  
DATA. THIS SWITCH HAS NO EFFECT ON  
STATUS CHECKING.

\*\*\*NOTE THAT WHEN SW11 OR 12 ARE SET, NOT ONLY ARE ERRORS NOT CHECKED, BU  
\*\*\*THEREFOR USE CAUTION TO ASSURE THAT OPERATIONS ARE NOT UNEXECUTED DUE  
\*\*\*\*DO NOT SET SW 11 OR 12 TO A ONE (1), DURING A RETRY SEQUENCE.

SW14:

SWITCH FOURTEEN (14) WHEN SET TO A ONE (1) WILL  
PRINT THE ACCUMULATED READ/WRITE STATISTICS FOR THE SELECTED  
SLAVE UNDER TEST AT THE END OF THE CURRENT BLOCK  
CYCLE. THE STATISTICS PRINTED ARE THE NUMBER OF BITS  
DROPPED OR PICKED, THE NUMBER OF RETRIES, WRITE ERRORS,  
READ ERRORS, AND DATA ERRORS.

SW15:

SWITCH FIFTEEN (15) WHEN SET TO A ONE,  
WILL CAUSE THE PROGRAM TO HALT ON ANY  
ERROR DETECTED BY THE OPERATION IN PROGRESS.  
IF BOTH SWITCH TEN (10) AND FIFTEEN (15)  
ARE SET, THE ACTUAL ERROR DETECTED WILL  
NOT BE PRINTED BUT WILL CAUSE A HALT.  
IF SWITCH TEN (10) IS RESET BEFORE PRESSING  
CONTINUE, THE ERROR WHICH CAUSED THE HALT  
WILL BE PRINTED BEFORE TESTING IS RESUMED.

792  
793  
794  
795  
796  
797  
798  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809  
810  
811  
812  
813  
814  
815  
816  
817  
818  
819  
820  
821  
822  
823  
824  
825  
826  
827  
828  
829  
830  
831  
832  
833  
834  
835  
836  
837  
838  
839  
840  
841

9. ERROR PRINTOUTS  
-----

THERE ARE THREE TYPES OF ERROR PRINTOUTS MADE BY THE PROGRAM; OPERATION ERRORS, DATA ERRORS, AND CONDITION ERRORS. EACH ERROR MESSAGE PRINTED IS PROCEEDED BY A TWO LINE HEADER WHICH CONTAINS THE DRIVE NUMBER, SLAVE NUMBER, DENSITY, PARITY, AND FORMAT ON THE FIRST LINE, AND THE BLOCK NUMBER, RECORD NUMBER, RECORD SIZE, AND ERROR TYPE ON THE SECOND.

A. OPERATION ERRORS:

THESE ARE ERRORS WHICH CAN OCCUR AS A DIRECT RESULT OF A TAPE OPERATION.

1. READ/WRITE STATUS ERRORS: THESE ARE DETECTED BY EITHER THE IMC3 ITSELF OR BY THE MASSBUS CONTROLLER. ALL STATUS ERRORS WILL BE REPORTED.
2. TAPE POSITION ERRORS: THESE ARE INDICATED BY AN INCORRECT SPACE OR REWIND OPERATION IN WHICH TAPE POSITION BECOMES UNRELIABLE.

B. DATA ERRORS:

DATA ERRORS WILL OCCUR WHEN TAPE IS BEING READ AND THE DATA FROM TAPE DOES NOT MATCH THE EXPECTED DATA. WHEN READING IN THE REVERSE DIRECTION, THE RECORD NUMBERS WILL BE COUNTED DOWN FROM LAST TO FIRST. THE CHARACTER NUMBERS IN REVERSE READS WILL ALSO BE COUNTED DOWN IN ORDER TO REFLECT TAPE POSITION RATHER THAN THE ORDER TRANSFERRED.

BECAUSE DATA RECORDS CAN BE UP TO FOUR THOUSAND CHARACTERS LONG, AN ERROR CONDITION WHICH WILL CAUSE THE ENTIRE RECORD TO READ INCORRECTLY COULD CAUSE A VERY LENGTHY PRINTOUT. THEREFORE, A COUNTER OF SUCCESSIVE BAD CHARACTERS IS EMPLOYED. IF TEN (10) CHARACTERS IN SUCCESSION ARE BAD, A NOTIFICATION IS PRINTED (BAD RECORD) AND THE NEXT TWENTY FIVE (25) CHARACTERS ARE SKIPPED BEFORE CHECKING IS RESUMED. IF THE BAD RECORD CONDITION OCCURS THREE (3) TIMES IN ONE RECORD, THE REST OF THE RECORD IS SKIPPED, DOWN TO THE LAST TEN (10) CHARACTERS WHICH WILL BE CHECKED. THE SKIPPING AND RESUMPTION OF CHECKING WILL ONLY BE DONE ON RECORDS WHICH ARE LONG ENOUGH TO ALLOW IT.

843  
844  
845  
846  
847  
848  
849  
850  
851  
852  
853  
854  
855  
856  
857  
858  
859  
860  
861  
862  
863  
864  
865  
866  
867  
868  
869  
870  
871  
872  
873  
874  
875  
876  
877  
878  
879  
880  
881  
882  
883  
884  
885  
886  
887  
888  
889  
890  
891  
892  
893  
894  
895  
896  
897  
898

C. CONDITION ERRORS: (CATASTROPHIC)

THESE PRINTOUTS REFLECT THE STATE OF THE TAPE SYSTEM  
EITHER BEFORE OR AFTER AN OPERATION

1. EOT: WHEN EOT (END OF TAPE) IS ENCOUNTERED DURING  
EITHER A READ OR WRITE, THE CYCLE IS COMPLETED  
ON THE SHORTENED BLOCK AFTER WHICH THE SLAVE  
WILL BE REWOUND AND FLAGGED AS UNAVAILABLE  
FOR TESTING UNTIL ALL SLAVES HAVE REACHED EOT AND  
ARE REWOUND. WHEN THE LAST AVAILABLE SLAVE  
HAS REACHED EOT AND BEEN REWOUND TO BOT,  
TESTING WILL BE RESUMED ON ALL SLAVES.
2. ILLEGAL BOT: WHEN A SLAVE ENCOUNTERS BOT DURING  
A READ, WRITE, OR SPACE OPERATION, AN ERROR  
IS PRINTED AND THE PROGRAM HALTED. THIS IS  
A CATASTROPHIC ERROR. TESTING MAY BE RESUMED  
BY PRESSING CONTINUE; BUT A RESTART IS  
SUGGESTED.
3. NO INTERRUPT RETURNED; EACH TAPE OPERATION SHOULD BE  
TERMINATED BY THE SETTING OF AN INTERRUPT IN  
THE CPU. IF NO INTERRUPT IS RETURNED WITHIN  
THE APPROPRIATE TIME, AN ERROR IS PRINTED.
4. NO MEDIUM ON-LINE; BEFORE AN OPERATION IS ATTEMPTED,  
THE TMO3 IS CHECKED FOR MOL. IF IT IS NOT  
SET, AN ERROR IS PRINTED, AND THE PROGRAM STOPPED.  
TESTING MAY BE RESUMED BY PRESSING CONTINUE.
5. NO BOT ON REWIND: AS EACH SLAVE IS REWOUND A CHECK  
IS MADE TO ASSURE THAT PROPER POSITION AT BOT  
IS ESTABLISHED. IF BOT IS NOT SET UPON COMPLETION OF  
A REWIND, AN ERROR IS PRINTED AND THE PROGRAM  
WILL HALT. PRESS CONTINUE TO RESUME TESTING.
6. POSITION ERROR: IF POSITION IS LOST DURING A RETRY,  
A MESSAGE IS PRINTED, THE TAPE REWOUND,  
AND REMOVED FROM TESTING UNTIL ALL ARE  
RESTARTED AT BLOCK ONE.
7. BAD TAPE OVERFLOW: IF 20(8) BAD TAPE SPOTS ARE FOUND,  
A MESSAGE IS PRINTED, THE TAPE REWOUND,  
AND REMOVED FROM TESTING UNTIL ALL ARE  
RESTARTED AT BLOCK ONE.
8. HARD READ ERROR: IF ANY HARD READ ERROR IS ENCOUNTERED  
DURING A RETRY, A MESSAGE IS PRINTED  
REGARDLESS OF THE SETTING OF SW10.
9. NON-RETRIALE: IF ANY NON RETRIABLE ERROR IS ENCOUNTERED, A  
MESSAGE IS PRINTED REGARDLESS OF THE SETTING OF SW10.

900  
901  
902  
903  
904  
905  
906  
907  
908  
909  
910  
911  
912  
913  
914  
915  
916  
917  
918  
919  
920  
921  
922  
923  
924  
925  
926  
927  
928  
929  
930  
931

D. EXAMPLES:

GLOSSARY:

BN \* CURRENT BLOCK NUMBER  
RN \* CURRENT RECORD NUMBER  
RS \* RECORD SIZE, IN FRAMES  
WE \* WRITE STATUS ERROR  
RE \* READ STATUS ERROR  
SE \* SPACE ERROR  
TM \* TAPE MARK  
F \* FORWARD  
R \* REVERSE  
CS1 \* RH/TE16 CONTROL REGISTER  
WC \* RH WORD COUNT  
BA \* RH BUS ADDRESS  
FC \* TE16 FRAME COUNT  
CS2 \* RH CONTROLLER STATUS  
DS \* TE16 DRIVE STATUS  
ER \* TE16 ERROR REGISTER  
AS \* ATTENTION SUMMARY  
CK \* TE16 CHECK CHARACTER  
DB \* RH DATA BUFFER  
MR \* TE16 MAINTENANCE REGISTER  
DT \* TE16 DRIVE TYPE  
SN \* TE16 SERIAL NUMBER  
IC \* TE16 TEST CONTROL  
\*F \* DATA FORMAT  
\*P \* PARITY  
\*D \* DENSITY  
\*PATRN \* DATA PATTERN NUMBER (R \* RANDOM)

933  
934  
935  
936  
937  
938  
939  
940  
941  
942  
943  
944  
945  
946  
947  
948  
949  
950  
951  
952  
953  
954  
955  
956  
957  
958  
959  
960  
961  
962  
963  
964  
965  
966  
967  
968  
969  
970  
971  
972  
973  
974  
975  
976  
977  
978

EXAMPLE 1: IN THIS EXAMPLE SLAVE 1 ON TMO3 0 WAS OPERATING AT 1600 BPI IN ODD PARITY USING THE NINE CHANNEL NORMAL DATA FORMAT. A WRITE STATUS ERROR WAS DETECTED. THE BAD STATUS INDICATES THAT AN UNCORRECTABLE DATA ERROR (BIT 6 OF ER) AND A PE FORMAT ERROR (BIT 7 OF ER) OCCURED DURING THE WRITE OPERATION OF THE SIXTH (6) RECORD OF THE FIFTY (50) RECORDS IN BLOCK (2). THE SIZE OF THE RECORD WAS TWO HUNDRED (200) FRAMES. THE CHECK CHARACTER REFLECTS THE BAD TRACK.

DRIVE NO. 0 \*SLAVE NO. 1 \*D 4 \*P 0 \*F 14 \*PATRN 1  
\*BN 2 \*RN 6-50 \*RS 200 \*WE  
CS1 144260  
CS2 100  
DS 150640  
ER 300  
WC 0  
CK 4

EXAMPLE 2: IN THIS EXAMPLE SLAVE 3 ON TMO3 1 WAS OPERATING AT 800 BPI IN EVEN PARITY USING THE NINE CHANNEL NORMAL DATA FORMAT. A READ STATUS ERROR WAS DETECTED DURING THE REVERSE READ OF THE TENTH (10) RECORD OF THE 25 RECORDS IN THIS BLOCK (12). THE SIZE OF THE RECORD IS TWENTY (20) FRAMES. THE PRINTOUT INDICATES THE DETECTION OF A VERTICAL PARITY ERROR (VPE: BIT 6 OF ER) AND A CYCLIC REDUNDENCY ERROR (CRC: BIT 15 OF ER). THE CRC CHARACTER, AS RECEIVED, IS NOT AS EXPECTED AND IS PRINTED SHOWING BOTH THE ACTUAL (FIRST) AND THE EXPECTED (LAST).

DRIVE NO. 2 \*SLAVE NO. 3 \*D 3 \*P 1 \*F 14 \*PATRN 3  
\*BN 12 \*RN 10 25 \*RS 20 \*RE R  
CS1 144276  
CS2 100  
DS 150600  
ER 100100  
WC 0  
CRC 767-777



JP

980  
981  
982  
983  
984  
985  
986  
987  
988  
989  
990  
991  
992  
993  
994  
995  
996  
997  
998  
999  
1000  
1001  
1002  
1003  
1004  
1005  
1006  
1007  
1008  
1009  
1010  
1011  
1012  
1013  
1014  
1015  
1016  
1017  
1018  
1019  
1020  
1021  
1022  
1023  
1024  
1025

EXAMPLE 3: IN THIS EXAMPLE, THE HEADER IS THE SAME AS IN EXAMPLE TWO (2) EXCEPT THAT THE ERROR TYPE REFLECTS A READ ERROR IN THE FORWARD DIRECTION. IT IS NORMAL FOR THE SYSTEM TO DETECT AN ERROR IN THE FORWARD AND REVERSE DIRECTION AT THE SAME RECORD. REMEMBER THAT IN REVERSE OPERATIONS THE RECORD NUMBER IS COUNTED DOWN SO THAT RECORD NUMBER TEN (10) WILL SHOWN IN THE PROPER POSITION IN BOTH FORWARD AND REVERSE.

DRIVE NO. 2 \*SLAVE NO 3 \*D 3 \*P 1 \*F 14 \*PATRN 2  
\*BN 12 \*RN 10-25 \*RS 20 \*RE F  
CS1 144270  
CS2 100  
DS 150600  
ER 100100  
WC 0  
CRC 767-777

EXAMPLE 4: IN EXAMPLES 2 AND 3 THE READ OPERATION RESULTED IN BAD STATUS, HOWEVER THE DATA ASSOCIATED WITH THE OPERATION WAS NOT BAD (OR WAS NOT CHECKED; SW 13-1). THIS EXAMPLE (4) SHOWS A PRINTOUT REFLECTING A READ STATUS ERROR ACCOMPANIED BY BAD DATA IN CHARACTERS FOUR (4) AND SIX (6).

DRIVE NO. 2 \*SLAVE NO. 3 \*D 3 \*P 1 \*F 14 \*PATRN 2  
\*BN 12 \*RN 10-25 \*RS 20 \*RE F  
CS1 144270  
CS2 100  
DS 150600  
ER 100100  
WC 0  
CRC 767-777  
CN 4  
G 11111111  
B 10111111  
CN 6  
G 11111111  
B 10111111

K2

1027  
1028  
1029  
1030  
1031  
1032  
1033  
1034  
1035  
1036  
1037  
1038  
1039  
1040  
1041  
1042  
1043  
1044  
1045  
1046  
1047  
1048  
1049  
1050  
1051  
1052  
1053  
1054  
1055  
1056  
1057  
1058  
1059  
1060  
1061  
1062  
1063  
1064  
1065  
1066  
1067  
1068  
1069  
1070  
1071

EXAMPLE 5: THIS EXAMPLE SHOWS A READ DATA ERROR WHICH OCCURRED, WITHOUT AN ACCOMPANYING STATUS ERROR, WHICH RESULTED IN A BAD RECORD.

DRIVE NO. 3 \*SLAVE NO. 1 \*D 4 \*P 0 \*F 14 \*PATRN R  
\*BN 100 \*RN 66-200 \*RS 2000 \*DE F  
CN 0  
G 11111111  
B 00000000  
CN 1  
G 11111111  
B 00000000  
CN 2  
G 11111111  
B 00000000  
CN 3  
G 11111111  
B 00000000  
CN 4  
G 11111111  
B 00000000  
CN 5  
G 11111111  
B 00000000  
CN 6  
G 11111111  
B 00000000  
CN 7  
G 11111111  
B 00000000  
BAD RECORD

EXAMPLE 6: THE FOLLOWING EXAMPLE SHOWS THE RESULT OF A SPACE OPERATION THAT SHOULD HAVE SPACED REVERSE OVER AN ENTIRE 100 RECORD BLOCK BUT WHICH TERMINATED AT THE END OF 40 RECORDS, LEAVING A POSITION ERROR OF 40

DRIVE NO. 2 \*SLAVE NO. 6 \*D 2 \*P 0 \*F 14  
\*BN 3 \*RN 100-100 \*RS 1000 \*SE R  
ERR AMT 40

1073  
1074  
1075  
1076  
1077  
1078  
1079  
1080  
1081  
1082  
1083  
1084  
1085  
1086  
1087  
1088  
1089  
1090  
1091  
1092  
1093  
1094  
1095  
1096  
1097  
1098  
1099  
1100  
1101  
1102  
1103  
1104  
1105  
1106  
1107  
1108  
1109  
1110  
1111  
1112  
1113  
1114  
1115  
1116  
1117  
1118  
1119  
1120

EXAMPLE 7: THIS EXAMPLE REFLECTS AN ERROR DETECTED WHILE WRITING A TAPE MARK (TM) AT THE END OF THE CURRENT DATA BLOCK PER OPTION RESPONSE TM=1. NOTE THAT THE TM RECORD NUMBER IS ONE GREATER THAN THE TOTAL NUMBER OF DATA RECORDS IN THE CURRENT BLOCK.

DRIVE NO. 1 \*SLAVE NO. 1 \*D 2 \*P 0 \*F 14  
\*BN 67 \*RN 101-100 \*RS 36 \*WE TM  
CS1 144226  
CS2 300  
DS 150604  
ER 1000  
WC 0

EXAMPLE 8: THIS EXAMPLE SHOWS TWO (2) PRINTOUTS REFLECTING A WRITE RETRY WHICH WAS NOT SUCCESSFUL THE FIRST TIME, BUT WHICH DID RECOVER ON THE SECOND. THE UNSUCCESSFUL RETRY IS LOGGED AS A SUSPECTED BAD TAPE SPOT BY ITS BLOCK AND RECORD NUMBER.

DRIVE NO. 0 \*SLAVE NO. 2 \*D 4 \*P 0 \*F 14 \*PATRN 6  
\*BN 2 \*RN 12-20 \*RS 667 \*WE  
CS1 144260  
CS2 100  
DS 150640  
ER 100  
WC 0  
\*\*\*ORIGINAL ERROR\*\*\*

DRIVE NO. 0 SLAVE NO. 2 \*D 4 \*P 0 \*F 14 \*PATRN 6  
\*BN 2 \*RN 12-20 \*RS 667 \*WE  
CS1 144260  
CS2 100  
DS 150640  
ER 100  
WC 0  
SUSPECT BAD TAPE  
RETRY: 0  
REPT: 0  
RECOVERED  
RETRY: 1

1132  
1133  
1134  
1135  
1136  
1137  
1138  
1139  
1140  
1141  
1142  
1143  
1144  
1145  
1146  
1147  
1148  
1149  
1150  
1151  
1152  
1153  
1154  
1155  
1156

EXAMPLE 9: IF . DURING A WRITE RETRY THE BACKSPACE  
OR THE ERASE OPERATION RESULT IN AN ERROR,  
THE ERROR WILL BE PRINTED AND THE PROGRAM  
HALTED. THIS EXAMPLE SHOWS THE ERROR PRINT  
FOR A SPACE AND AN ERASE (2 EXAMPLES)

DRIVE NO. 1 \*SLAVE NO. 1 \*D 3 \*P 0 \*F 14  
BN 12 \*RN 8-64 \*RS 500 \*SE RTRY  
ERR AMT 1

DRIVE NO. 1 \*SLAVE NO. 1 \*D 3 \*P 0 \*F 14  
\*BN 12 \*RN 8-64 \*RS 500 \*ERASE  
CS1 144224  
CS2 100  
DS 150600  
ER 400  
WC 0

EXAMPLE 10: THIS EXAMPLE SHOWS THE PRINTOUT FROM  
A REWIND OPERATION WHICH DOES NOT HAVE  
BOT SET AT THE END.

DRIVE NO. 2 \*SLAVE NO. 3 \*D 3 \*P 0 \*F 14  
\*BN 66 \*RN 15-20 \*RS 1000  
NOT BOT ON REWIND: HALT

EXAMPLE 11: THIS EXAMPLE SHOWS THE PRINTOUT MADE WHEN  
THERE IS NO INTERRUPT RETURNED AT THE END  
OF AN OPERATION.

DRIVE NO. 7 \*SLAVE NO. 7 \*D 2 \*P 1 \*F 14  
\*BN 1 \*RN 25-26 \*RS 1200  
NO INTERRUPT

1158  
1159  
1160  
1161  
1162  
1163  
1164  
1165  
1166  
1167  
1168  
1169  
1170  
1171  
1172  
1173  
1174  
1175  
1176  
1177  
1178  
1179  
1180  
1181  
1182  
1183  
1184  
1185  
1186  
1187  
1188  
1189  
1190  
1191  
1192  
1193  
1194  
1195  
1196  
1197  
1198  
1199  
1200  
1201  
1202  
1203  
1204  
1205  
1206  
1207

10. STATISTICS PRINTOUT  
-----

THE PROGRAM, THROUGH ITS ERROR CHECKING, IS ABLE TO GATHER CERTAIN STATISTICS ABOUT THE PERFORMANCE OF EACH UNIT UNDER TEST. THIS INFORMATION IS PRINTED OUT WHENEVER A UNIT IS REWOUND FROM END OF TAPE, OR BECAUSE IT IS TO BE REMOVED FROM TESTING DUE TO SOME CATASTROPHIC ERROR, (POSITION LOST, BAD TAPE OVERFLOW) THE STATISTICS MAY BE PRINTED AT ANY TIME BY SETTING SWITCH 14 TO A ONE (1). THIS PRESENTS A PICTURE OF PERFORMANCE UP TO THIS TIME. THE STATISTICS WILL BE CLEARED UPON REWIND OF THE UNIT; BUT NOT BY SETTING SW 14.

STATISTICS PRINT EXAMPLE (A HEADER WILL PRECEED THE STATS)

DROPS: 0 3 0 0 0 6 45 0  
PICKS: 1 0 0 0 0 0 0 2  
RETRY: 1  
WTERR: 2  
REFWD: 3  
OFT: 2  
HARD: 1  
DEFWD: 0  
REREV: 4  
SOFT: 1  
HARD: 3  
DEREV: 0  
2 BAD TAPE SPOTS  
0 \*BN 1 \*RN 2  
1 \*BN 15 \*RN 100

\*\* NOTE \*\* DROPS AND PICKS REFLECT CORE BIT POSITIONS.  
THE FOLLOWING IS A TABLE OF CORE BITS TO TRACK NUMBER.

TRACK NO.	7	6	5	3	9	1	8	2
CORE BIT	7	6	5	4	3	2	1	0

DROPS: NUMBER OF DATA BITS DROPPED; PER CORE BIT(SEE NOTE ABOVE)  
PICKS: NUMBER OF DATA BITS PICKED UP; PER CORE BIT(SEE NOTE ABOVE)  
RETRY: NUMBER OF WRITE RETRIES  
WTERR: NUMBER OF WRITE ERRORS NOT ASSOCIATED WITH BAD TAPE  
REFWD: NUMBER OF READ FORWARD STATUS ERRORS  
REREV: NUMBER OF READ REVERSE STATUS ERRORS  
SOFT: NUMBER OF RECOVERED READ ERRORS  
HARD: NUMBER OF UNRECOVERED READ ERRORS  
DEFWD: NUMBER OF FORWARD DATA ERRORS WITH NO ASSOCIATED STATUS ERROR  
DEREV: NUMBER OF REVERSE DATA ERRORS WITH NO ASSOCIATED STATUS ERROR

016  
017  
018  
019  
020  
021  
022  
023  
024  
025  
026  
027  
028  
029  
030  
031  
032  
033  
034  
035  
036  
037  
038  
039  
040  
041  
042  
043  
044  
045  
046  
047  
048  
049  
050  
051  
052  
053  
054  
055  
056  
057  
058  
059  
060  
061  
062  
063  
064  
065  
066  
067  
068  
069  
070  
071  
072  
073  
074  
075  
076  
077  
078  
079  
080  
081  
082  
083  
084  
085  
086  
087  
088  
089  
090  
091  
092  
093  
094  
095  
096  
097  
098  
099  
100  
101  
102  
103  
104  
105  
106  
107  
108  
109  
110  
111  
112  
113  
114  
115  
116  
117  
118  
119  
120  
121  
122  
123  
124  
125  
126  
127  
128  
129  
130  
131  
132  
133  
134  
135  
136  
137  
138  
139  
140  
141  
142  
143  
144  
145  
146  
147  
148  
149  
150  
151  
152  
153  
154  
155  
156  
157  
158  
159  
160  
161  
162  
163  
164  
165  
166  
167  
168  
169  
170  
171  
172  
173  
174  
175  
176  
177  
178  
179  
180  
181  
182  
183  
184  
185  
186  
187  
188  
189  
190  
191  
192  
193  
194  
195  
196  
197  
198  
199  
200  
201  
202  
203  
204  
205  
206  
207  
208  
209  
210  
211  
212  
213  
214  
215  
216  
217  
218  
219  
220  
221  
222  
223  
224  
225  
226  
227  
228  
229  
230  
231  
232  
233  
234  
235  
236  
237  
238  
239  
240  
241  
242  
243  
244  
245  
246  
247  
248  
249  
250  
251  
252  
253  
254  
255  
256  
257  
258  
259  
260  
261  
262  
263  
264  
265  
266  
267  
268  
269  
270  
271  
272  
273  
274  
275  
276  
277  
278  
279  
280  
281  
282  
283  
284  
285  
286  
287  
288  
289  
290  
291  
292  
293  
294  
295  
296  
297  
298  
299  
300  
301  
302  
303  
304  
305  
306  
307  
308  
309  
310  
311  
312  
313  
314  
315  
316  
317  
318  
319  
320  
321  
322  
323  
324  
325  
326  
327  
328  
329  
330  
331  
332  
333  
334  
335  
336  
337  
338  
339  
340  
341  
342  
343  
344  
345  
346  
347  
348  
349  
350  
351  
352  
353  
354  
355  
356  
357  
358  
359  
360  
361  
362  
363  
364  
365  
366  
367  
368  
369  
370  
371  
372  
373  
374  
375  
376  
377  
378  
379  
380  
381  
382  
383  
384  
385  
386  
387  
388  
389  
390  
391  
392  
393  
394  
395  
396  
397  
398  
399  
400  
401  
402  
403  
404  
405  
406  
407  
408  
409  
410  
411  
412  
413  
414  
415  
416  
417  
418  
419  
420  
421  
422  
423  
424  
425  
426  
427  
428  
429  
430  
431  
432  
433  
434  
435  
436  
437  
438  
439  
440  
441  
442  
443  
444  
445  
446  
447  
448  
449  
450  
451  
452  
453  
454  
455  
456  
457  
458  
459  
460  
461  
462  
463  
464  
465  
466  
467  
468  
469  
470  
471  
472  
473  
474  
475  
476  
477  
478  
479  
480  
481  
482  
483  
484  
485  
486  
487  
488  
489  
490  
491  
492  
493  
494  
495  
496  
497  
498  
499  
500  
501  
502  
503  
504  
505  
506  
507  
508  
509  
510  
511  
512  
513  
514  
515  
516  
517  
518  
519  
520  
521  
522  
523  
524  
525  
526  
527  
528  
529  
530  
531  
532  
533  
534  
535  
536  
537  
538  
539  
540  
541  
542  
543  
544  
545  
546  
547  
548  
549  
550  
551  
552  
553  
554  
555  
556  
557  
558  
559  
560  
561  
562  
563  
564  
565  
566  
567  
568  
569  
570  
571  
572  
573  
574  
575  
576  
577  
578  
579  
580  
581  
582  
583  
584  
585  
586  
587  
588  
589  
590  
591  
592  
593  
594  
595  
596  
597  
598  
599  
600  
601  
602  
603  
604  
605  
606  
607  
608  
609  
610  
611  
612  
613  
614  
615  
616  
617  
618  
619  
620  
621  
622  
623  
624  
625  
626  
627  
628  
629  
630  
631  
632  
633  
634  
635  
636  
637  
638  
639  
640

- SW7: 1-GENERATE RANDOM CHARACTER COUNT  
0-USE FIXED CHARACTER COUNT
- SW6: 1-GENERATE RANDOM RECORD COUNT  
0-USED FIXED RECORD COUNT
- SW5: 1-YOZZLE ON CURRENT RECORD  
0-DO NOT YOZZLE ON RECORD
- SW4: 1-DO WRITE/READ RETRIES  
0-DO NOT RETRI
- SW3: 1-DO NOT READ FORWARD  
0-READ FORWARD
- SW2: 1-DO NOT READ REVERSE  
0-READ REVERSE
- SW1: 1-READ FORWARD FIRST  
0-READ REVERSE FIRST
- SW0: 1-DO NOT WRITE  
0-WRITE

640  
641  
642  
643  
644  
645  
646  
647  
648  
649  
650  
651  
652  
653  
654  
655  
656  
657  
658  
659  
660  
661  
662  
663  
664  
665  
666  
667  
668  
669  
670  
671  
672  
673  
674  
675  
676  
677  
678  
679  
680  
681  
682  
683  
684  
685  
686  
687

SWITCH EXPLANATION AND EXAMPLES:

SWO-3:

THESE SWITCHES ARE USED TO CONTROL THE SEQUENCE OF MAG TAPE OPERATIONS PERFORMED ON EACH AVAILABLE UNIT. THE BLOCK OF DATA DESCRIBED THROUGH THE RESPONSES TO TELETYPE REQUESTS AT INITIAL START WILL BE EITHER WRITTEN OR READ FROM EACH AVAILABLE UNIT IN THE ORDER THAT THEY WERE ENTERED. THE SEQUENCE OF OPERATIONS IS CALLED A CYCLE, AND WILL BE PERFORMED CONTINUOUSLY UNTIL STOPPED BY THE OPERATOR. WHEN END OF TAPE IS REACHED, THE UNIT WILL BE REWOUND AND FLAGGED AS UNAVAILABLE FOR TEST UNTIL ALL UNITS HAVE REACH EOT, AT WHICH TIME TESTING IS RESUMED ON ALL AVAILABLE UNITS.

EXAMPLES: 0-3

- A. SWO=0, SW1=0, SW2=1, SW3=1  
WRITE ONLY X RECORDS OF Y CHARACTERS
- B. SWO=0, SW1=0, SW2=1, SW3=0  
WRITE THEN BACKSPACE AND READ FORWARD X RECORDS
- C. SWO=0, SW1=0, SW2=0, SW3=1  
WRITE THEN READ REVERSE X RECORDS.
- D. SWO=0, SW1=0, SW2=0, SW3=0  
WRITE THEN READ REVERSE AND READ FORWARD X RECORDS
- E. SWO=0, SW1=1, SW2=0, SW3=0  
WRITE THEN BACKSPACE AND READ FORWARD THEN REVERSE
- F. SWO=1, SW1=0, SW2=1, SW3=0  
READ TAPE FORWARD X RECORDS
- G. SWO=1, SW1=0, SW2=0, SW3=1  
READ TAPE REVERSE X RECORDS
- H. SWO=1, SW1=0, SW2=0, SW3=0  
READ TAPE REVERSE THEN FORWARD
- I. SWO=1, SW1=1, SW2=0, SW3=0  
READ TAPE FORWARD THEN REVERSE

{},'

689  
690  
691  
692  
693  
694  
695  
696  
697  
698  
699  
700  
701  
702  
703  
704  
705  
706  
707  
708  
709  
710  
711  
712  
713  
714  
715  
716  
717  
718  
719  
720  
721  
722  
723  
724  
725  
726  
727  
728  
729  
730  
731  
732  
733  
734  
735  
736  
737  
738  
739  
740  
741  
742

SW4:

SWITCH FOUR (4), WHEN SET TO A ONE (1), WILL CAUSE ANY DATA RELATED ERROR TO BE RETRIED. THE WRITE RETRY SCHEME CONSISTS OF REWRITING THE RECORD IN THE SAME SPOT ON TAPE FOUR (4) TIMES. IF ALL FOUR (4) REPEATS ARE SUCCESSFUL, THE RECORD IS CONSIDERED AS RECOVERED, AND A TAPE WRITE ERROR IS LOGGED. IF ANY OF THE FOUR (4) REPEATS IS UNSUCCESSFUL, A SKIP ERASE IS DONE, A SUSPECTED BAD TAPE SPOT IS LOGGED AT THIS BLOCK AND RECORD NUMBER, AND A SECOND RETRY OF FOUR REPEATS IS DONE. IF AFTER FOUR (4) RETRIES, THE RECORD CANNOT BE RECOVERED A NOTIFICATION IS PRINTED, AND TESTING IS RESUMED ON THE NEXT RECORD. IF 20(8) BAD TAPE SPOTS ARE FOUND, THE SLAVE WILL BE REWOUND AND REMOVED FROM TESTING WITH AN APPROPRIATE MESSAGE PRINTED. THE READ RETRY SCHEME CONSISTS OF REREADING THE RECORD UP TO EIGHT TIMES. IF ALL EIGHT REREADS ARE BAD, IT IS A HARD ERROR. IF ANY REREAD IS SUCCESSFUL, THIS IS A SOFT ERROR. IF THE ORIGINAL ERROR IS OF THE NON-RETRYABLE TYPE (IE; ILF,RMR,ILR,NEF,CBUSPE), THE RETRY SCHEME IS NOT ENTERED AND A MESSAGE IS PRINTED.

SW5:

SWITCH FIVE (5) WHEN SET DURING A READ FORWARD OR REVERSE WILL CAUSE THE TAPE TO CONTINUOUSLY READ THE CURRENT RECORD BY SPACING EITHER FORWARD OR REVERSE AND REREADING THAT RECORD. THIS TAPE MOVEMENT IS CALLED YOZZLING. THERE IS A SOFTWARE DELAY EXECUTED BETWEEN EACH SPACE/READ OF THE RECORD AND IT MAY BE VARIED BY TYPING CONTROL C ON THE TELETYPE DURING THE EXECUTION OF THE YOZZLE AND RESPONDING TO THE PRINTED REQUEST WITH A SIX (6) DIGIT VALUE. THE YOZZLE STALL IS PRESET TO A VALUE OF 3000 IN THE PROGRAM TO PREVENT EXCESSIVE TAPE WEAR, BUT MAY BE SET TO ANY VALUE THROUGH THE TELETYPE.

SW6-8:

THESE THREE (3) SWITCHES CONTROL THE RANDOMIZATION OF DATA AND BLOCK SIZE AND MAY BE SET AND RESET AT ANY TIME. THE ACTUAL CHANGE WILL TAKE PLACE BETWEEN BLOCK CYCLES.

SW9:

SWITCH NINE (9) WHEN SET WILL CAUSE ALL AVAILABLE TAPE UNITS TO BE REWOUND AT THE END OF THE CURRENT BLOCK CYCLE. TESTING WILL BE RESUMED AT A BLOCK COUNT OF ONE (1) WHEN ALL UNITS HAVE REACHED BOT.



744  
745  
746  
747  
748  
749  
750  
751  
752  
753  
754  
755  
756  
757  
758  
759  
760  
761  
762  
763  
764  
765  
766  
767  
768  
769  
770  
771  
772  
773  
774  
775  
776  
777  
778  
779  
780  
781  
782  
783  
784  
785  
786  
787  
788  
789  
790

SW10-13:

THESE SWITCHES ARE USED TO CONTROL THE  
ERROR HANDLING TO BE DONE ON THE TAPE  
OPERATION DESCRIBED BY SWITCHES 0-3.

- A. SWITCH TEN (10) WHEN SET TO A ONE  
WILL DISALLOW ANY ERROR PRINTOUTS MADE  
ON THE OPERATION IN PROGRESS. CATASTROPHIC  
FAILURES AND INFORMATION PRINTOUTS WILL  
STILL OCCUR. IE: UNIT NOT AVAILABLE, ILLEGAL  
BOT, DROP OR PICK OVERFLOW, AND EOT REWIND.
- B. SWITCH ELEVEN (11) WHEN SET TO A ONE  
WILL DISALLOW THE CHECKING FOR STATUS  
ERRORS ON READ (FORWARD OR REVERSE) OPERATIONS.
- C. SWITCH TWELVE (12) WHEN SET TO A ONE  
WILL DISALLOW THE CHECKING FOR STATUS  
ERRORS ON WRITE OPERATIONS.
- D. SWITCH THIRTEEN (13) WHEN SET TO A ONE  
WILL DISALLOW THE CHECKING OF READ  
DATA. THIS SWITCH HAS NO EFFECT ON  
STATUS CHECKING.

\*\*\*NOTE THAT WHEN SW11 OR 12 ARE SET, NOT ONLY ARE ERRORS NOT CHECKED, B.  
\*\*\*THEREFOR USE CAUTION TO ASSURE THAT OPERATIONS ARE NOT UNEXECUTED DUE  
\*\*\*\*\*DO NOT SET SW 11 OR 12 TO A ONE (1), DURING A RETRY SEQUENCE.

SW14:

SWITCH FOURTEEN (14) WHEN SET TO A ONE (1) WILL  
PRINT THE ACCUMULATED READ/WRITE STATISTICS FOR THE SELECTED  
SLAVE UNDER TEST AT THE END OF THE CURRENT BLOCK  
CYCLE. THE STATISTICS PRINTED ARE THE NUMBER OF BITS  
DROPPED OR PICKED, THE NUMBER OF RETRIES, WRITE ERRORS,  
READ ERRORS, AND DATA ERRORS.

SW15:

SWITCH FIFTEEN (15) WHEN SET TO A ONE,  
WILL CAUSE THE PROGRAM TO HALT ON ANY  
ERROR DETECTED BY THE OPERATION IN PROGRESS.  
IF BOTH SWITCH TEN (10) AND FIFTEEN (15)  
ARE SET, THE ACTUAL ERROR DETECTED WILL  
NOT BE PRINTED BUT WILL CAUSE A HALT.  
IF SWITCH TEN (10) IS RESET BEFORE PRESSING  
CONTINUE, THE ERROR WHICH CAUSED THE HALT  
WILL BE PRINTED BEFORE TESTING IS RESUMED.

793  
794  
795  
796  
797  
798  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809  
810  
811  
812  
813  
814  
815  
816  
817  
818  
819  
820  
821  
822  
823  
824  
825  
826  
827  
828  
829  
830  
831  
832  
833  
834  
835  
836  
837  
838  
839  
840  
841

9. ERROR PRINTOUTS  
-----

THERE ARE THREE TYPES OF ERROR PRINTOUTS MADE BY THE PROGRAM; OPERATION ERRORS, DATA ERRORS, AND CONDITION ERRORS. EACH ERROR MESSAGE PRINTED IS PROCEEDED BY A TWO LINE HEADER WHICH CONTAINS THE DRIVE NUMBER, SLAVE NUMBER, DENSITY, PARITY, AND FORMAT ON THE FIRST LINE, AND THE BLOCK NUMBER, RECORD NUMBER, RECORD SIZE, AND ERROR TYPE ON THE SECOND.

A. OPERATION ERRORS:

THESE ARE ERRORS WHICH CAN OCCUR AS A DIRECT RESULT OF A TAPE OPERATION.

1. READ/WRITE STATUS ERRORS: THESE ARE DETECTED BY EITHER THE IM03 ITSELF OR BY THE MASSBUS CONTROLLER. ALL STATUS ERRORS WILL BE REPORTED.
2. TAPE POSITION ERRORS: THESE ARE INDICATED BY AN INCORRECT SPACE OR REWIND OPERATION IN WHICH TAPE POSITION BECOMES UNRELIABLE.

B. DATA ERRORS:

DATA ERRORS WILL OCCUR WHEN TAPE IS BEING READ AND THE DATA FROM TAPE DOES NOT MATCH THE EXPECTED DATA. WHEN READING IN THE REVERSE DIRECTION, THE RECORD NUMBERS WILL BE COUNTED DOWN FROM LAST TO FIRST. THE CHARACTER NUMBERS IN REVERSE READS WILL ALSO BE COUNTED DOWN IN ORDER TO REFLECT TAPE POSITION RATHER THAN THE ORDER TRANSFERRED.

BECAUSE DATA RECORDS CAN BE UP TO FOUR THOUSAND CHARACTERS LONG, AN ERROR CONDITION WHICH WILL CAUSE THE ENTIRE RECORD TO READ INCORRECTLY COULD CAUSE A VERY LENGTHY PRINTOUT. THEREFORE, A COUNTER OF SUCCESSIVE BAD CHARACTERS IS EMPLOYED. IF TEN (10) CHARACTERS IN SUCCESSION ARE BAD, A NOTIFICATION IS PRINTED (BAD RECORD) AND THE NEXT TWENTY FIVE (25) CHARACTERS ARE SKIPPED BEFORE CHECKING IS RESUMED. IF THE BAD RECORD CONDITION OCCURS THREE (3) TIMES IN ONE RECORD, THE REST OF THE RECORD IS SKIPPED, DOWN TO THE LAST TEN (10) CHARACTERS WHICH WILL BE CHECKED. THE SKIPPING AND RESUMPTION OF CHECKING WILL ONLY BE DONE ON RECORDS WHICH ARE LONG ENOUGH TO ALLOW IT.

843  
844  
845  
846  
847  
848  
849  
850  
851  
852  
853  
854  
855  
856  
857  
858  
859  
860  
861  
862  
863  
864  
865  
866  
867  
868  
869  
870  
871  
872  
873  
874  
875  
876  
877  
878  
879  
880  
881  
882  
883  
884  
885  
886  
887  
888  
889  
890  
891  
892  
893  
894  
895  
896  
897  
898

C. CONDITION ERRORS: (CATASTROPHIC)

THESE PRINTOUTS REFLECT THE STATE OF THE TAPE SYSTEM  
EITHER BEFORE OR AFTER AN OPERATION

1. EOT: WHEN EOT (END OF TAPE) IS ENCOUNTERED DURING  
EITHER A READ OR WRITE, THE CYCLE IS COMPLETED  
ON THE SHORTENED BLOCK AFTER WHICH THE SLAVE  
WILL BE REWOUND AND FLAGGED AS UNAVAILABLE  
FOR TESTING UNTIL ALL SLAVES HAVE REACHED EOT AND  
ARE REWOUND. WHEN THE LAST AVAILABLE SLAVE  
HAS REACHED EOT AND BEEN REWOUND TO BOT,  
TESTING WILL BE RESUMED ON ALL SLAVES.
2. ILLEGAL BOT: WHEN A SLAVE ENCOUNTERS BOT DURING  
A READ, WRITE, OR SPACE OPERATION, AN ERROR  
IS PRINTED AND THE PROGRAM HALTED. THIS IS  
A CATASTROPHIC ERROR. TESTING MAY BE RESUMED  
BY PRESSING CONTINUE; BUT A RESTART IS  
SUGGESTED.
3. NO INTERRUPT RETURNED: EACH TAPE OPERATION SHOULD BE  
TERMINATED BY THE SETTING OF AN INTERRUPT IN  
THE CPU. IF NO INTERRUPT IS RETURNED WITHIN  
THE APPROPRIATE TIME, AN ERROR IS PRINTED.
4. NO MEDIUM ON-LINE: BEFORE AN OPERATION IS ATTEMPTED,  
THE IM03 IS CHECKED FOR MOL. IF IT IS NOT  
SET, AN ERROR IS PRINTED, AND THE PROGRAM STOPPED.  
TESTING MAY BE RESUMED BY PRESSING CONTINUE.
5. NO BOT ON REWIND: AS EACH SLAVE IS REWOUND A CHECK  
IS MADE TO ASSURE THAT PROPER POSITION AT BOT  
IS ESTABLISHED. IF BOT IS NOT SET UPON COMPLETION OF  
A REWIND, AN ERROR IS PRINTED AND THE PROGRAM  
WILL HALT. PRESS CONTINUE TO RESUME TESTING.
6. POSITION ERROR: IF POSITION IS LOST DURING A RETRY,  
A MESSAGE IS PRINTED, THE TAPE REWOUND,  
AND REMOVED FROM TESTING UNTIL ALL ARE  
RESTARTED AT BLOCK ONE.
7. BAD TAPE OVERFLOW: IF 20(8) BAD TAPE SPOTS ARE FOUND,  
A MESSAGE IS PRINTED, THE TAPE REWOUND,  
AND REMOVED FROM TESTING UNTIL ALL ARE  
RESTARTED AT BLOCK ONE.
8. HARD READ ERROR: IF ANY HARD READ ERROR IS ENCOUNTERED  
DURING A RETRY, A MESSAGE IS PRINTED  
REGARDLESS OF THE SETTING OF SW10.
9. NON-RETRYABLE: IF ANY NON-RETRYABLE ERROR IS ENCOUNTERED, A  
MESSAGE IS PRINTED REGARDLESS OF THE SETTING OF SW10.

900  
901  
902  
903  
904  
905  
906  
907  
908  
909  
910  
911  
912  
913  
914  
915  
916  
917  
918  
919  
920  
921  
922  
923  
924  
925  
926  
927  
928  
929  
930  
931

D. EXAMPLES:

GLOSSARY:

BN = CURRENT BLOCK NUMBER  
RN = CURRENT RECORD NUMBER  
RS = RECORD SIZE, IN FRAMES  
WE = WRITE STATUS ERROR  
RE = READ STATUS ERROR  
SE = SPACE ERROR  
TM = TAPE MARK  
F = FORWARD  
R = REVERSE  
CS1 = RH/TE16 CONTROL REGISTER  
WC = RH WORD COUNT  
BA = RH BUS ADDRESS  
FC = TE16 FRAME COUNT  
CS2 = RH CONTROLLER STATUS  
DS = TE16 DRIVE STATUS  
ER = TE16 ERROR REGISTER  
AS = ATTENTION SUMMARY  
CK = TE16 CHECK CHARACTER  
DB = RH DATA BUFFER  
MR = TE16 MAINTENANCE REGISTER  
DT = TE16 DRIVE TYPE  
SN = TE16 SERIAL NUMBER  
TC = TE16 TEST CONTROL  
\*F = DATA FORMAT  
\*P = PARITY  
\*D = DENSITY  
\*PATRN = DATA PATTERN NUMBER (R = RANDOM)

933  
934  
935  
936  
937  
938  
939  
940  
941  
942  
943  
944  
945  
946  
947  
948  
949  
950  
951  
952  
953  
954  
955  
956  
957  
958  
959  
960  
961  
962  
963  
964  
965  
966  
967  
968  
969  
970  
971  
972  
973  
974  
975  
976  
977  
978

EXAMPLE 1: IN THIS EXAMPLE SLAVE 1 ON IM03 0 WAS OPERATING AT 1600 BPI IN ODD PARITY USING THE NINE CHANNEL NORMAL DATA FORMAT. A WRITE STATUS ERROR WAS DETECTED. THE BAD STATUS INDICATES THAT AN UNCORRECTABLE DATA ERROR (BIT 6 OF ER) AND A PE FORMAT ERROR (BIT 7 OF ER) OCCURED DURING THE WRITE OPERATION OF THE SIXTH (6) RECORD OF THE FIFTY (50) RECORDS IN BLOCK (2). THE SIZE OF THE RECORD WAS TWO HUNDRED (200) FRAMES. THE CHECK CHARACTER REFLECTS THE BAD TRACK.

DRIVE NO. 0 \*SLAVE NO. 1 \*D 4 \*P 0 \*F 14 \*PATRN 1  
\*BN 2 \*RN 6-50 \*RS = 200 \*WE  
CS1 144260  
CS2 100  
DS 150640  
ER 300  
WC 0  
CK 4

EXAMPLE 2: IN THIS EXAMPLE SLAVE 3 ON IM03 1 WAS OPERATING AT 800 BPI IN EVEN PARITY USING THE NINE CHANNEL NORMAL DATA FORMAT. A READ STATUS ERROR WAS DETECTED DURING THE REVERSE READ OF THE TENTH (10) RECORD OF THE 25 RECORDS IN THIS BLOCK (10). THE SIZE OF THE RECORD IS TWENTY (20) FRAMES. THE PRINTOUT INDICATES THE DETECTION OF A VERTICAL PARITY ERROR (VPE; BIT 6 OF ER) AND A CYCLIC REDUNDENCY ERROR (CRC; BIT 15 OF ER). THE CRC CHARACTER, AS RECEIVED, IS NOT AS EXPECTED AND IS PRINTED SHOWING BOTH THE ACTUAL (FIRST) AND THE EXPECTED (LAST).

DRIVE NO. 2 \*SLAVE NO. 3 \*D 3 \*P 1 \*F 14 \*PATRN 3  
\*BN 12 \*RN 10-25 \*RS 20 \*RE R  
CS1 144276  
CS2 100  
DS 150600  
ER 100100  
WC 0  
CRC 767-777

980  
981  
982  
983  
984  
985  
986  
987  
988  
989  
990  
991  
992  
993  
994  
995  
996  
997  
998  
999  
1000  
1001  
1002  
1003  
1004  
1005  
1006  
1007  
1008  
1009  
1010  
1011  
1012  
1013  
1014  
1015  
1016  
1017  
1018  
1019  
1020  
1021  
1022  
1023  
1024  
1025

EXAMPLE 3: IN THIS EXAMPLE, THE HEADER IS THE SAME AS  
IN EXAMPLE TWO (2) EXCEPT THAT THE ERROR TYPE  
REFLECTS A READ ERROR IN THE FORWARD  
DIRECTION. IT IS NORMAL FOR THE SYSTEM  
TO DETECT AN ERROR IN THE FORWARD AND  
REVERSE DIRECTION AT THE SAME RECORD.  
REMEMBER THAT IN REVERSE OPERATIONS THE  
RECORD NUMBER IS COUNTED DOWN SO THAT  
RECORD NUMBER TEN (10) WILL SHOWN IN  
THE PROPER POSITION IN BOTH FORWARD AND  
REVERSE.

DRIVE NO. 2 \*SLAVE NO 3 \*D 3 \*P 1 \*F 14 \*PATRN 2  
\*BN 12 \*RN 10-25 \*RS 20 \*RE F  
CS1 144270  
CS2 100  
DS 150600  
ER 100100  
WC 0  
CRC 767-777

EXAMPLE 4: IN EXAMPLES 2 AND 3 THE READ OPERATION  
RESULTED IN BAD STATUS, HOWEVER THE  
DATA ASSOCIATED WITH THE OPERATION WAS  
NOT BAD (OR WAS NOT CHECKED: SW 13-1).  
THIS EXAMPLE (4) SHOWS A PRINTOUT REFLECTING  
A READ STATUS ERROR ACCOMPANIED BY BAD  
DATA IN CHARACTERS FOUR (4) AND SIX (6).

DRIVE NO. 2 \*SLAVE NO. 3 \*D 3 \*P 1 \*F 14 \*PATRN 2  
\*BN 12 \*RN 10-25 \*RS 20 \*RE F  
CS1 144270  
CS2 100  
DS 150600  
ER 100100  
WC 0  
CRC 767-777  
CN 4  
G 11111111  
B 10111111  
CN 6  
G 11111111  
B 10111111

1027  
1028  
1029  
1030  
1031  
1032  
1033  
1034  
1035  
1036  
1037  
1038  
1039  
1040  
1041  
1042  
1043  
1044  
1045  
1046  
1047  
1048  
1049  
1050  
1051  
1052  
1053  
1054  
1055  
1056  
1057  
1058  
1059  
1060  
1061  
1062  
1063  
1064  
1065  
1066  
1067  
1068  
1069  
1070  
1071

EXAMPLE 5: THIS EXAMPLE SHOWS A READ DATA ERROR WHICH OCCURRED, WITHOUT AN ACCOMPANYING STATUS ERROR, WHICH RESULTED IN A BAD RECORD.

DRIVE NO. 3 \*SLAVE NO. 1 \*D 4 \*P 0 \*F 14 \*PATRN R  
\*BN 100 \*RN 66-200 \*RS 2000 \*DE F  
CN 0  
G 11111111  
B 00000000  
CN 1  
G 11111111  
B 00000000  
CN 2  
G 11111111  
B 00000000  
CN 3  
G 11111111  
B 00000000  
CN 4  
G 11111111  
B 00000000  
CN 5  
G 11111111  
B 00000000  
CN 6  
G 11111111  
B 00000000  
CN 7  
G 11111111  
B 00000000  
BAD RECORD

EXAMPLE 6: THE FOLLOWING EXAMPLE SHOWS THE RESULT OF A SPACE OPERATION THAT SHOULD HAVE SPACED REVERSE OVER AN ENTIRE 100 RECORD BLOCK BUT WHICH TERMINATED AT THE END OF 40 RECORDS. LEAVING A POSITION ERROR OF 40

DRIVE NO. 2 \*SLAVE NO. 6 \*D 2 \*P 0 \*F 14  
\*BN 3 \*RN 100-100 \*RS 1000 \*SE R  
ERR AMT 40

1073  
1074  
1075  
1076  
1077  
1078  
1079  
1080  
1081  
1082  
1083  
1084  
1085  
1086  
1087  
1088  
1089  
1090  
1091  
1092  
1093  
1094  
1095  
1096  
1097  
1098  
1099  
1100  
1101  
1102  
1103  
1104  
1105  
1106  
1107  
1108  
1109  
1110  
1111  
1112  
1113  
1114  
1115  
1116  
1117  
1118  
1119  
1120

EXAMPLE 7: THIS EXAMPLE REFLECTS AN ERROR DETECTED WHILE WRITING A TAPE MARK (TM) AT THE END OF THE CURRENT DATA BLOCK PER OPTION RESPONSE TM=1. NOTE THAT THE TM RECORD NUMBER IS ONE GREATER THAN THE TOTAL NUMBER OF DATA RECORDS IN THE CURRENT BLOCK.

DRIVE NO. 1 \*SLAVE NO. 1 \*D 2 \*P 0 \*F 14  
\*BN 67 \*RN 101-100 \*RS 36 \*WE TM  
CS1 144226  
CS2 300  
DS 150604  
ER 1000  
WC 0

EXAMPLE 8: THIS EXAMPLE SHOWS TWO (2) PRINTOUTS REFLECTING A WRITE RETRY WHICH WAS NOT SUCCESSFUL THE FIRST TIME, BUT WHICH DID RECOVER ON THE SECOND. THE UNSUCCESSFUL RETRY IS LOGGED AS A SUSPECTED BAD TAPE SPOT BY ITS BLOCK AND RECORD NUMBER.

DRIVE NO. 0 \*SLAVE NO. 2 \*D 4 \*P 0 \*F 14 \*PATRN 6  
\*BN 2 \*RN 12-20 \*RS 667 \*WE  
CS1 144260  
CS2 100  
DS 150640  
ER 100  
WC 0  
\*\*\*ORIGINAL ERROR\*\*\*

DRIVE NO. 0 SLAVE NO. 2 \*D 4 \*P 0 \*F 14 \*PATRN 6  
\*BN 2 \*RN 12-20 \*RS 667 \*WE  
CS1 144260  
CS2 100  
DS 150640  
ER 100  
WC 0  
SUSPECT BAD TAPE  
RETRY: 0  
REPT: 0  
RECOVERED  
RETRY: 1



1122  
1123  
1124  
1125  
1126  
1127  
1128  
1129  
1130  
1131  
1132  
1133  
1134  
1135  
1136  
1137  
1138  
1139  
1140  
1141  
1142  
1143  
1144  
1145  
1146  
1147  
1148  
1149  
1150  
1151  
1152  
1153  
1154  
1155  
1156

EXAMPLE 9: IF , DURING A WRITE RETRY THE BACKSPACE  
OR THE ERASE OPERATION RESULT IN AN ERROR,  
THE ERROR WILL BE PRINTED AND THE PROGRAM  
HALTED. THIS EXAMPLE SHOWS THE ERROR PRINT  
FOR A SPACE AND AN ERASE (2 EXAMPLES)

DRIVE NO. 1 \*SLAVE NO. 1 \*D 3 \*P 0 \*F 14  
BN 12 \*RN 8-64 \*RS 500 \*SE RTRY  
ERR AMT 1

DRIVE NO. 1 \*SLAVE NO. 1 \*D 3 \*P 0 \*F 14  
\*BN 12 \*RN 8-64 \*RS 500 \*ERASE  
CS1 144224  
CS2 100  
DS 150600  
ER 400  
WC 0

EXAMPLE 10: THIS EXAMPLE SHOWS THE PRINTOUT FROM  
A REWIND OPERATION WHICH DOES NOT HAVE  
BOT SET AT THE END.

DRIVE NO. 2 \*SLAVE NO. 3 \*D 3 \*P 0 \*F 14  
\*BN 66 \*RN 15-20 \*RS 1000  
NOT BOT ON REWIND: HALT

EXAMPLE 11: THIS EXAMPLE SHOWS THE PRINTOUT MADE WHEN  
THERE IS NO INTERRUPT RETURNED AT THE END  
OF AN OPERATION.

DRIVE NO. 7 \*SLAVE NO. 7 \*D 2 \*P 1 \*F 14  
\*BN 1 \*RN 25-26 \*RS 1200  
NO INTERRUPT

1158  
1159  
1160  
1161  
1162  
1163  
1164  
1165  
1166  
1167  
1168  
1169  
1170  
1171  
1172  
1173  
1174  
1175  
1176  
1177  
1178  
1179  
1180  
1181  
1182  
1183  
1184  
1185  
1186  
1187  
1188  
1189  
1190  
1191  
1192  
1193  
1194  
1195  
1196  
1197  
1198  
1199  
1200  
1201  
1202  
1203  
1204  
1205  
1206  
1207

10. STATISTICS PRINTOUT  
-----

THE PROGRAM, THROUGH ITS ERROR CHECKING, IS ABLE TO GATHER CERTAIN STATISTICS ABOUT THE PERFORMANCE OF EACH UNIT UNDER TEST. THIS INFORMATION IS PRINTED OUT WHENEVER A UNIT IS REWOUND FROM END OF TAPE, OR BECAUSE IT IS TO BE REMOVED FROM TESTING DUE TO SOME CATASTROPHIC ERROR. (POSITION LOST, BAD TAPE OVERFLOW) THE STATISTICS MAY BE PRINTED AT ANY TIME BY SETTING SWITCH 14 TO A ONE (1). THIS PRESENTS A PICTURE OF PERFORMANCE UP TO THIS TIME. THE STATISTICS WILL BE CLEARED UPON REWIND OF THE UNIT; BUT NOT BY SETTING SW 14.

STATISTICS PRINT EXAMPLE (A HEADER WILL PRECEED THE STATS)

DROPS: 0 3 0 0 0 6 45 0  
PICKS: 1 0 0 0 0 0 0 2  
RETRY: 1  
WTERR: 2  
REFWD: 3  
OFT: 2  
HARD: 1  
DEFWD: 0  
REREV: 4  
SOFT: 1  
HARD: 3  
DEREV: 0  
2 BAD TAPE SPOTS  
0 \*BN 1 \*RN 2  
1 \*BN 15 \*RN 100

\*\* NOTE \*\* DROPS AND PICKS REFLECT CORE BIT POSITIONS.  
THE FOLLOWING IS A TABLE OF CORE BITS TO TRACK NUMBER.

TRACK NO.	7	6	5	3	9	1	8	2
CORE BIT	7	6	5	4	3	2	1	0

DROPS: NUMBER OF DATA BITS DROPPED; PER CORE BIT(SEE NOTE ABOVE)  
PICKS: NUMBER OF DATA BITS PICKED UP; PER CORE BIT(SEE NOTE ABOVE)  
RETRY: NUMBER OF WRITE RETRIES  
WTERR: NUMBER OF WRITE ERRORS NOT ASSOCIATED WITH BAD TAPE  
REFWD: NUMBER OF READ FORWARD STATUS ERRORS  
REREV: NUMBER OF READ REVERSE STATUS ERRORS  
SOFT: NUMBER OF RECOVERED READ ERRORS  
HARD: NUMBER OF UNRECOVERED READ ERRORS  
DEFWD: NUMBER OF FORWARD DATA ERRORS WITH NO ASSOCIATED STATUS ERROR  
DEREV: NUMBER OF REVERSE DATA ERRORS WITH NO ASSOCIATED STATUS ERROR

1209  
1210  
1211  
1212  
1213  
1214  
1215  
1216  
1217  
1218  
1219  
1220  
1221  
1222  
1223  
1224  
1225  
1226  
1227  
1228  
1229  
1230  
1231  
1232  
1233  
1234  
1235  
1236  
1237  
1238  
1239  
1240  
1241  
1242  
1243  
1244  
1245  
1246  
1247  
1248  
1249  
1250

11. AUTO SEQUENCE

THE AUTO SEQUENCE (START AT ADDRESS 240) WILL EXECUTE A  
PREDETERMINED TEST PLAN ON ALL AVAILABLE SLAVES ON EACH  
AVAILABLE TMO3. THE ONLY OPERATOR RESPONSE IS TO THE TYPED  
REQUESTS FOR THE RH ADDRESS, VECTOR, CONTINUOUS OR SINGLE  
CYCLE, AND NRZ ONLY. ALL SWITCHES REMAIN ACTIVE AND MAY BE  
USED NORMALLY; HOWEVER THE IDEA IS TO LEAVE ALL SWITCHES  
DOWN AND ALLOW FULL EXECUTION OF THE TEST PLAN FOR  
SYSTEM CHECKOUT.

SAMPLE START AT 240(8): AUTO SEQUENCE.

LOAD ADDRESS 240(8), SET SWITCHES TO ZERO, PRESS START;

TE16 AUTO SEQUENCE TEST  
ENTER CONDITIONS IN OCTAL

REGISTER START = 172400(172440)  
VECTOR ADDRESS = 224(CR)  
NRZ ONLY: (0)  
AUTO CONT: (1)

THIS EXAMPLE SHOWS AN AUTO SEQUENCE START WITH THE RH  
AT BUS ADDRESS 172440 AND A VECTOR OF 224. ALL AVAILABLE  
HARDWARE WILL BE TESTED CONTINUOUSLY IN BOTH NRZ AND PE MODE.

AS EACH TMO3 AND ITS SLAVES ARE FOUND, A DIVIDER LINE OF  
ASTERICKS WILL BE PRINTED FOLLOWED BY A PRINTOUT OF THE  
TMO3 AND ITS SLAVES BEING TESTED. AS EACH TMO3 AND  
ITS SLAVES ARE FINISHED, ANOTHER DIVIDER IS PRINTED  
BEFORE TESTING IS RESUMED ON THE NEXT AVAILABLE DRIVE.

WHEN ALL AVAILABLE HARDWARE HAS BEEN TESTED,  
A PRINTOUT OF END OF SEQUENCE WILL BE DONE AND THE  
PROGRAM WILL EITHER HALT (AUTO CONT = 0) OR RESTART WITH  
THE FIRST AVAILABLE UNIT (AUTO CONT = 1).

1252  
1253  
1254  
1255  
1256  
1257  
1258  
1259  
1260  
1261  
1262  
1263  
1264  
1265  
1266  
1267  
1268  
1269  
1270  
1271  
1272  
1273  
1274  
1275  
1276  
1277  
1278  
1279  
1280  
1281  
1282  
1283  
1284

AUTO SEQUENCE TEST PLAN:

THE AUTO SEQUENCE WILL EXECUTE BOTH AN NRZ AND A PE CYCLE. EACH CYCLE WILL BE STARTED FROM BOT AND CONSIST OF VARIOUS DATA PATTERNS INTENDED TO BE WORST CASE FOR THAT PARTICULAR MODE.

1. NRZ CYCLE:

SIX (6) BLOCKS OF ONE HUNDRED (100) RECORDS OF FOUR THOUSAND (4000) CHARACTERS FOR EACH OF THE FOUR DATA PATTERNS.

PATTERN 1: ALL ONES DATA IN ALL BYTES  
PATTERN 10: WALKING ONE/ALL ONE  
PATTERN 14: WALKING ZERO/ALL ZERO  
RANDOM DATA: RANDOM

2. PE CYCLE: (IF NRZ ONLY = 0)

SIX BLOCKS OF ONE HUNDRED (100) RECORDS OF FOUR THOUSAND (4000) CHARACTERS EACH FOR EACH OF THREE DATA PATTERNS, THEN RANDOM DATA BLOCKS TO END OF TAPE.

PATTERN 10: WALKING ONE/ALL ONE  
PATTERN 14: WALKING ZERO/ALL ZERO  
PATTERN 15: THREE (3) 0 CHARACTERS, TWO (2) ALL CHARACTERS, THREE (3) THEN COMPLIMENT PATTERN, REPEATED FOR A FULL BUFFER  
RANDOM DATA: RANDOM

1286  
1287  
1288  
1289  
1290  
1291  
1292  
1293  
1294  
1295  
1296  
1297  
1298  
1299  
1300  
1301  
1302  
1303  
1304  
1305  
1306  
1307  
1308  
1309  
1310  
1311  
1312  
1313  
1314  
1315  
1316  
1317  
1318  
1319  
1320  
1321  
1322  
1323  
1324  
1325  
1326  
1327  
1328  
1329  
1330  
1331  
1332  
1333

12. TESTING PROCEDURES  
-----

AS PREVIOUSLY STATED THIS PROGRAM CONTAINS NO FIXED TESTS. THE ENTIRE TEST CYCLE TO BE EXECUTED IS DESCRIBED BY THE OPERATOR THOUGH RESPONSES TO TELETYPE REQUESTS FOR PARAMETERS AND CONSOLE SWITCH SETTINGS FOR OPERATION. THE OPERATION SELECTED WILL BE EXECUTED WITH THE PARAMETERS ENTERED CONTINUOUSLY ON EACH AVAILABLE UNIT, ONE BLOCK AT A TIME, UNTIL STOPPED BY THE OPERATOR. THE OPERATION MAY BE CHANGED DYNAMICALLY BY CHANGING THE CONSOLE SWITCHES AT ANY TIME. THE PROGRAM WILL ATTEMPT TO PERFORM ANY OPERATION SET AND THEREFORE CAUTION SHOULD BE TAKEN TO ASSURE THAT THE UNIT IS CAPABLE OF PERFORMING AS REQUESTED. FOR INSTANCE, ONE SHOULD NOT ATTEMPT TO PERFORM READ OPERATIONS ON A TAPE WHICH HAS NOT BEEN WRITTEN AS THE DATA, IF ANY, IS UNPREDICTABLE. HOWEVER, IF A TAPE HAS BEEN WRITTEN WITH THIS PROGRAM, IT CAN BE READ AS OFTEN AS DESIRED WITHOUT BEING REWRITTEN. THIS IS A GOOD PROCEDURE TO USE FOR TESTING TAPE COMPATABILITY. SCOPING OF TAPE UNITS BECOMES SIMPLE; BY SETTING THE DESIRED OPERATION AND ITS PARAMETER, A UNIT MAY BE CONTINUOUSLY EXERCISED IN ANY MANNER DESIRED. BY USING THE VARIOUS ERROR CONTROL SWITCHES AND ENTERING THE NEEDED STALL, ANY FUNCTION CAN BE SCOPED RATHER EASILY. RELIABILITY TESTING CAN BE PERFORMED BY USE OF THE RANDOMIZATION CAPABILITY. PERHAPS A CYCLE OF RANDOM TESTING MIGHT BE SET UP AND ALLOWED TO RUN FOR SOME PERIOD OF TIME, THE STATISTICAL COLLECTION OF DROPS AND PICKS IS THEN SIGNIFICANT. INTERMITTANT PROBLEMS CAN BE FOUND BY SETTING THE DESIRED OPERATION IN MOTION AND DISALLOWING ERROR PRINTOUTS WHILE ALLOWING A HALT ON ERROR. THE ERROR THAT CAUSED THE HALT CAN BE PRINTED BY RESETTING CONSOLE SWITCH TEN AND PRESSING CONTINUE. IF SOME PARTICULAR DATA PATTERN SHOULD BE CAUSING DATA ERROR, USE OF THE YOZZLE SWITCH AND ITS ASSOCIATED STALL WILL TO ALLOW SCOPING OF THIS PARTICULAR RECORD.

AS YOU SEE, THERE ARE MYRIAD TESTING PROCEDURES WHICH COULD BE PERFORMED. THE PARAMETERS, TAPE OPERATIONS, ERROR EXAMINATION AND REPORTING ARE ALL AT YOUR DISCRETION.

TRY IT, YOU'LL LIKE IT.

\*

1335  
1336  
1337  
1338  
1339  
1340  
1341  
1342  
1343  
1344  
1345  
1346  
1347  
1348  
1349  
1350  
1351  
1352  
1353  
1354  
1355  
1356  
1357  
1358  
1359  
1360  
1361  
1362  
1363  
1364  
1365  
1366  
1367  
1368  
1369  
1370  
1371  
1372  
1373  
1374  
1375  
1376  
1377  
1378  
1379  
1380  
1381  
1382  
1383  
1384  
1385  
1386  
1387  
1388  
1389  
1390

```
.LIST BIN,LOC,SEQ  
.TITLE CZTEDEO TMO3-TE16/TU77 DRT  
;DATA RELIABILITY TEST  
;AC-A800E-MC  
;21 FEB 1977  
;J.G.ADAMS  
  
;REVISED (++) J.G.ADAMS MAY 1978  
; ++B  
; ++B  
; ++B  
; ++B  
; ++B  
  
; (++C) M,PAGE FEB 79  
; ++C  
;  
;  
  
;REVISED JAN 1984 BY J.A.C.HITT  
;(CZTEDD)  
;  
;  
;  
;  
;  
  
;REVISED FEB 1984 BY J. HITT  
;(CZTEDE)  
  
.MCALL . $ACT11,.$EOP,$SAVE,$RESTORE,$CHAIN  
.NI IST MC  
.LIST ME  
.ENABLE ABS,AMA  
  
;CONSOLE SWITCHES*****  
  
;SW15: 1-STOP ON ERROR  
; 0-CONTINUE ON ERROR  
;SW14: 1-PRINT READ/WRITE STATS  
; 0-DO NOT PRINT STATS  
;SW13: 1-DO NOT CHECK DATA  
; 0-CHECK DATA  
;SW12: 1-DO NOT CHECK WRITE ERRORS  
; 0-CHECK WRITE ERRORS  
;SW11: 1-DO NOT CHECK READ ERRORS  
; 0-CHECK READ ERRORS  
;SW10: 1-DO NOT PRINT ERRORS  
; 0-PRINT ERRORS  
;SW9: 1-REWIND TAPE  
; 0-DO NOT REWIND  
;SW8: 1-USE RANDOM DATA  
; 0-USE FIXED DATA PATTERN  
;SW7: 1-USE RANDOM CHARACTER COUNT  
; 0-USE FIXED CHAR COUNT  
;SW6: 1-USE RANDOM RECORD COUNT
```

1)INCORRECT RECORD COUNT  
STORED WHEN EOT REACHED ON WRITE  
2)ADJUST STACK PTR ON BAD TAPE OVFLW  
3)ADDED TU77 TEST CAPABILITY  
4)DOES NOT GENERATE LRC/CRC ON FIRST  
RECORD IN AUTO ACCEPT MODE  
  
RECORD NUMBERING SYSTEM NOT CONSISTENT  
BETWEEN FORWARD AND REVERSE TAPE MOVEMEN  
FORMAT ERROR (BIT 4) MADE RETRYABLE  
  
FIX SO THAT RECORD SIZE CAN BE  
A MAXIMUM OF 10000 OCTAL BYTES  
LONG. THIS IS CONDITIONAL IN  
THAT THERE MUST BE ENOUGH MEMORY  
FOR THE BUFFER, OTHERWISE RECORD  
SIZE WILL BE 4000 OCTAL. THIS  
CORRECTS AID REPORT 0CC0001450  
  
ADD XON/XOFF FUNCTIONALITY FOR  
PRINTOUTS.

1391  
1392  
1393  
1394  
1395  
1396  
1397  
1398  
1399  
1400  
1401  
1402  
1403  
1404

;  
; SW5: 1=YOZZLE ON CURRENT RECORD  
; 0=DO NOT YOZZLE  
; SW4: 1=DO BOTH READ AND WRITE RETRIES  
; 0=INHIBIT RETRIES  
; SW3: 1=DO NOT READ FORWARD  
; 0=READ FORWARD  
; SW2: 1=DO NOT READ REVERSE  
; 0=READ REVERSE  
; SW1: 1=READ FORWARD FIRST  
; 0=READ REVERSE FIRST  
; SW0: 1=DO NOT WRITE  
; 0=WRITE  
; IF SWR <15::00> = 177777 OR NOT AVAILABLE USE SOFTWARE SWITCH REGISTER





```

1466
1467 ;TRAP CATCHERS*****
1468
1476 000020 000020 . *20
1477 000020 023110 .WORD TTOUT ;SET IOT TRAP TO TTOUT ROUTINE
1478 000022 000340 .WORD 340 ;PRIORITY LEVEL 7
1479
1480 000004 TYPE=IOT ;EQUATE TYPE TO AN IOT INSTRUCTION
1481 000034 . *34
1482 000034 023326 .WORD OCTP ;SET TRAP TRAP TO OCTP ROUTINE
1483 000036 000340 .WORD 340
1484 104400 TYPOCT=TRAP ;EQUATE TYPOCT TO TRAP INSTRUCTION
1485
(1) ;ACT11 HOOK *****
(1) 000040 $SVPC ;SAVE CURRENT LOCATION CTR
(1) 000042 . *42
(1) 000042 000000 .WORD 0
(1) 000046 000046 . *46
(1) 000046 005010 .WORD $ENDAD ;SET LOCATION 46
(1) 000052 000052 . *52
(1) 000052 000000 .WORD 0 ;SET LOCATION 52 = 0
(1) 000040 . *$SVPC ;RESTORE LOCATION CTR
(1)
1486 ;TTY INTERRUPT VECTOR*****
1487 000060 . *60
1488 000060 021050 .WORD TTINT ;TTY INTERRUPT HANDLER ADDRESS
1489 000062 000340 .WORD 340 ;PRIORITY LEVEL 7
1490
1491 ;SOFTWARE SWITCH REGISTER*****
1492 ;INVOKED IF SWR <15:;00> = 17777 OR NOT AVAILABLE
1493 000176 000176 . *176
1494 000176 000000 SWREG: .WORD 0
1495
1496 ;START ADDRESS*****
1497 000200 000200 . *200
1498 000200 000137 003032 JMP START ;ENTER PARAMETERS VIA TTY
1499
1500 000204 000204 . *204
1501 000204 000137 003250 JMP STARTC ;USE FIXED PARAMETERS; HOLD DATA
1502
1503 000210 000210 . *210
1504 000210 005037 014520 CLR RDEF
1505 000214 000137 003256 JMP STARTA ;USE FIXED PARAMETERS; NEW DATA
1506
1507 ;MAG TAPE INTERRUPT VECTOR*****
1508
1509 000224 000224 . *224
1510 000224 021274 MTINT ;MAG TAPE INTERRUPT HANDLER ADDRESS
1511 000226 000340 340
1512
1513 ;AUTO SEQUENCE START*****
1514
1515 000240 000240 . *240
1516 000240 005237 000747 INC ASEQF ;SET AUTO SEQUENCE FLAG
1517 000244 000137 003234 JMP STAUF ;GO TO START OF AUTO SEQUENCE

```

13

```

1519 ;SHORT CONVERSATION RESTART*****
1520
1521 000300 000300 .+300
1522 000300 005237 013560 INC SCVFL ;SET SHORT CONVERSATION FLAG
1523 000304 000137 003032 JMP START ;ENTER SHORT PARAMETER LIST
1524
1525 000510 .+510
1526 ;TU16 REGISTER EQUIVS*****
1527
1528 000510 172440 C1: 172440
1529 000512 172442 WC: 172442
1530 000514 172444 BA: 172444
1531 000516 172446 FC: 172446
1532 000520 172450 CS: 172450
1533 000522 172452 DS: 172452
1534 000524 172454 ER: 172454
1535 000526 172456 AS: 172456
1536 000530 172460 CC: 172460
1537 000532 172462 DB: 172462
1538 000534 172464 MR: 172464
1539 000536 172466 DT: 172466
1540 000540 172470 SN: 172470
1541 000542 172472 TC: 172472
1542
1543 ;CONSTANTS*****
1544
1545 000544 172440 REGS: 172440 ;STARTING REGISTER ADDRESS (CS1)
1546 000546 000224 VECT: 224 ;VECTOR ADDRESS (RH INTERRUPT)
1547 000550 000000 DVN: 0 ;DRIVE NUMBER
1548 000552 000000 UDES: 0 ;UNIT DESCRIPTION (PARITY,DENSITY,UNIT,FORMAT)
1549 000554 000100 RCNT: 100 ;RECORD COUNTER
1550 000556 174000 FMCNT: 174000 ;NUMBER OF CHAR (4000) OCTAL IN TWOS COMPLEMENT
1551 000560 174000 BUFBEG: 174000 ;MAXIMUM BUFFER SIZE
1552 000562 026544 WDATA: BUFBEG ;START OF WRITE BUFFER
1553 000564 032544 RDATA: BUFBEG+4000 ;START OF READ BUFFER
1554 000566 000001 PATRN: 1 ;DATA PATTERN SELECTOR (0 - 15) OCTAL
1555 000570 000000 RDCMD: 0 ;READ COMMAND
1556 000572 000001 TMEX: 1 ;TAPE MARK FLAG: 1=TM 0=NO TM
1557 000574 000000 CRCC: 0 ;CRC CORRECTION FLAG (YES=1,NO=0)
1558 000576 000000 INTRF: 0 ;INTERCHANGE READ 1=YES 0=NO
1559 000600 000000 SPFLG: 0 ;SINGLE PASS 1=YES 0=NO
1560 000602 000001 RSTAL: 1 ;READ STALL
1561 000604 000001 WSTAL: 1 ;WRITE STALL
1562 000606 000001 TSTAL: 1 ;TURN AROUND STAL
1563 000610 002000 YSTAL: 2000 ;YOZZLE STAL
1564 000612 000010 RETRY: 10 ;READ RETRY NUMBER
1565 000614 177776 PSW: 177776 ;PROCESSOR STATUS
1566 000616 177570 SWR: 177570 ;CONSOLE SWITCHES
1567 000620 177560 TKS: 177560 ;TTY READ STATUS REGISTER
1568 000622 177562 TKB: 177562 ;TTY READ BUFFER
1569 000624 177564 TPS: 177564 ;TTY PUNCH STATUS REGISTER
1570 000626 177566 TPB: 177566 ;TTY PUNCH OUTPUT REGISTER
1571 000630 177550 PRS: 177550 ;H/S READER STATUS REGISTER
1572 000632 177552 PRB: 177552 ;H/S READER BUFFER
1573 000634 153624 RANBAS: 153624 ;RANDOM NUMBER GENERATOR BASE
1574 000636 032561 RANSAV: 032561 ;RANDOM NUMBER BUFFER
  
```

1575	000640	000100	RCSAV:	100	;RECORD COUNT SAVE
1576	000642	174000	FCSAV:	174000	;FRAME COUNT SAVE
1577					
1578					
1579					;FLAGS AND COUNTERS*****
1580	000644	000000	TINF:	0	;TTY ENTRY FLAG
1581	000646		STFLG:		
1582	000646	000000	TOB:	0	;TTY OUTPUT BUFFER
1583	000650	000000	TIB:	0	;TTY INPUT BUFFER
1584	000652	000000	TEMP1:	0	;TEMP STORAGE
1585	000654	000000	TEMP2:	0	;TEMP STORAGE
1586	000656	000000	TEMP3:	0	;TEMP STORAGE
1587	000660	000000	EMADDR:	0	;ERROR MSG ADDRESS STORAGE
1588	000662	000000	BLCNTR:	0	;BLOCK COUNTER
1589	000664	000000	BBC:	0	;BAD RECORD COUNTER
1590	000666	000000	EOTREC:	0	;EOT FLAG
1591	000670	000000	RTRN:	0	;INTERRUPT RETURN STORAGE
1592	000672	000000	HDRFL:	0	;HEADER FLAG
1593	000674	000000	STAL:	0	;DELAY STORAGE
1594	000676	000000	PFLG:	0	;PRINT FLAG
1595	000700	000000	MTC1:	0	;MAG TAPE CONT REGISTER BUFFER
1596	000702	000000	UNP:	0	;UNIT TABLE POINTER
1597	000704	000000	TMFLG:	0	;TAPE MARK FLAG
1598	000706	000000	RPCNT:	0	;REPEAT COUNTER
1599	000710	000000	RTCNT:	0	;RETRY COUNTER
1600	000712	000000	DERFL:	0	;DATA ERROR FLAG
1601	000714	000000	SERFL:	0	;STATUS ERROR FLAG
1602	000716	000000	BCNT:	0	;BIT COUNTER
1603	000720	000000	RTYFL:	0	;RETRY FLAG
1604	000722	000000	UPS:	0	;UNIT POINTER SAVE
1605	000724	000000	BDPP:	0	;BITS DROPPED POINTER
1606	000726	000000	BPKP:	0	;BITS PICKED POINTER
1607	000730	000000	ERSAV:	0	;ERROR SAVE LOC
1608	000732	000000	BTFLG:	0	;BAD TAPE FLAG
1609	000734	000000	BTSTF:	0	;STATISTIC PRINT FLAG
1610	000736	000000	BTPT:	0	;BAD TAPE POINTER
1611	000740	000000	ERTFL:	0	;ERASE FLAG
1612	000742		ENDFLG:		
1613	000742	000000	ASEQ:	0	;AUTO SEQ FLAG
1614	000744	000000	ABL CNT:	0	;AUTO BLOCK COUNTER
1615	000746	000000	ASEQCF:	0	;AUTO SEQ CONTINUOUS FLAG
1616	000750	000000	\$CNTRLS:	0	;XON/XOFF FLAG

```

1618
1619
1620 ;UNIT ORDER AND DESCRIPTION TABLE *****
1621 000752 000000 UN1: 0 ;THIS TABLE IS LOADED
1622 000754 000000 UN2: 0 ;WITH UNIT NUMBERS AND
1623 000756 000000 UN3: 0 ;THEIR DESCRIPTIONS IN
1624 000760 000000 UN4: 0 ;THE ORDER THAT THEY
1625 000762 000000 UN5: 0 ;WILL BE TESTED
1626 000764 000000 UN6: 0
1627 000766 000000 UN7: 0
1628 000770 000000 UN8: 0
1629 000772 177777 UNX: -1

```

```

1630
1631 ;UNIT DROPS AND PICKS POINTERS*****
1632
1633 000774 001214 PIK1: BP00
1634 000776 001234 PIK2: BP10
1635 001000 001254 PIK3: BP20
1636 001002 001274 PIK4: BP30
1637 001004 001314 PIK5: BP40
1638 001006 001334 PIK6: BP50
1639 001010 001354 PIK7: BP60
1640 001012 001374 PIK8: BP70
1641 001014 001414 DRP1: BD00
1642 001016 001434 DRP2: BD10
1643 001020 001454 DRP3: BD20
1644 001022 001474 DRP4: BD30
1645 001024 001514 DRP5: BD40
1646 001026 001534 DRP6: BD50
1647 001030 001554 DRP7: BD60
1648 001032 001574 DRP8: BD70

```

```

1649
1650 ;UNIT BAD TAPE POINTERS*****
1651
1652 001034 001614 BTADDR: BT00
1653 001036 001720 BT01
1654 001040 002024 BT02
1655 001042 002130 BT03
1656 001044 002234 BT04
1657 001046 002340 BT05
1658 001050 002444 BT06
1659 001052 002550 BT07

```

```

1660
1661 ;UNIT WRITE RETRY COUNTER*****
1662
1663 ;SET START OF STATISTICS TABLE
1664 001054 STBL:
1665 001054 000000 RTY1: 0
1666 001056 000000 RTY2: 0
1667 001060 000000 RTY3: 0
1668 001062 000000 RTY4: 0
1669 001064 000000 RTY5: 0
1670 001066 000000 RTY6: 0
1671 001070 000000 RTY7: 0
1672 001072 000000 RTY8: 0
1673

```

```

1674                                     ;UNIT WRITE ERRORS*****
1675
1676 001074 000000      WTER1: 0
1677 001076 000000      WTER2: 0
1678 001100 000000      WTER3: 0
1679 001102 000000      WTER4: 0
1680 001104 000000      WTER5: 0
1681 001106 000000      WTER6: 0
1682 001110 000000      WTER7: 0
1683 001112 000000      WTER8: 0
1684
1685                                     ;UNIT READ FORWARD ERRORS*****
1686
1687 001114 000000      RDER1: 0
1688 001116 000000      RDER2: 0
1689 001120 000000      RDER3: 0
1690 001122 000000      RDER4: 0
1691 001124 000000      RDER5: 0
1692 001126 000000      RDER6: 0
1693 001130 000000      RDER7: 0
1694 001132 000000      RDER8: 0
1695
1696                                     ;UNIT DATA ERRORS FORWARD*****
1697
1698 001134 000000      DATER1: 0
1699 001136 000000      0
1700 001140 000000      0
1701 001142 000000      0
1702 001144 000000      0
1703 001146 000000      0
1704 001150 000000      0
1705 001152 000000      0
1706
1707                                     ;UNIT READ REVERSE ERRORS*****
1708
1709 001154 000000      RDERR1: 0
1710 001156 000000      0
1711 001160 000000      0
1712 001162 000000      0
1713 001164 00 000      0
1714 001166 000000      0
1715 001170 000000      0
1716 001172 000000      0
1717
1718                                     ;UNIT DATA ERRORS REVERSE*****
1719
1720 001174 000000      DEREV1: 0
1721 001176 000000      0
1722 001200 000000      0
1723 001202 000000      0
1724 001204 000000      0
1725 001206 000000      0
1726 001210 000000      0
1727 001212 000000      0

```

```

1729 ;DROPS + PICKS PER CHANNEL PER UNIT*****
1730
1731 001214 000000 BP00: 0
1732 001234 001234 . = . +16
1733 001234 000000 BP10: 0
1734 001254 001254 . = . +16
1735 001254 000000 BP20: 0
1736 001274 001274 . = . +16
1737 001274 000000 BP30: 0
1738 001314 001314 . = . +16
1739 001314 000000 BP40: 0
1740 001334 001334 . = . +16
1741 001334 000000 BP50: 0
1742 001354 001354 . = . +16
1743 001354 000000 BP60: 0
1744 001374 001374 . = . +16
1745 001374 000000 BP70: 0
1746 001414 001414 . = . +16
1747 001414 000000 BD00: 0
1748 001434 001434 . = . +16
1749 001434 000000 BD10: 0
1750 001454 001454 . = . +16
1751 001454 000000 BD20: 0
1752 001474 001474 . = . +16
1753 001474 000000 BD30: 0
1754 001514 001514 . = . +16
1755 001514 000000 BD40: 0
1756 001534 001534 . = . +16
1757 001534 000000 BD50: 0
1758 001554 001554 . = . +16
1759 001554 000000 BD60: 0
1760 001574 001574 . = . +16
1761 001574 000000 BD70: 0
1762 001614 . = . +16
1763
1764

```

```

1766
1767 ;UNIT BAD TAPE COUNTER;16 PER SLAVE*****
1768
1769 001614 000000 BT00: 0
1770 001720 001720 . = .+102
1771 001720 000000 BT01: 0
1772 002024 002024 . = .+102
1773 002024 000000 BT02: 0
1774 002130 002130 . = .+102
1775 002130 000000 BT03: 0
1776 002234 002234 . = .+102
1777 002234 000000 BT04: 0
1778 002340 002340 . = .+102
1779 002340 000000 BT05: 0
1780 002444 002444 . = .+102
1781 002444 000000 BT06: 0
1782 002550 002550 . = .+102
1783 002550 000000 BT07: 0
1784 002654 002654 . = .+102
1785
1786 ;UNIT END OF TAPE COUNTERS 1 PER SLAVE*****
1787
1788 002654 000000 EOTCO: 0
1789 002656 000000 0
1790 002660 000000 0
1791 002662 000000 0
1792 002664 000000 0
1793 002666 000000 0
1794 002670 000000 0
1795 002672 000000 0
1796
1797 ;UNIT READ FORWARD SOFT ERROR*****
1798
1799 002674 000000 RFSOFT: 0
1800 002676 000000 0
1801 002700 000000 0
1802 002702 000000 0
1803 002704 000000 0
1804 002706 000000 0
1805 002710 000000 0
1806 002712 000000 0
1807
1808 ;UNIT READ REVERSE SOFT ERROR*****
1809
1810 002714 000000 RRSOFT: 0
1811 002716 000000 0
1812 002720 000000 0
1813 002722 000000 0
1814 002724 000000 0
1815 002726 000000 0
1816 002730 000000 0
1817 002732 000000 0
1818

```

```

1820
1821                                ;UNIT READ FORWARD HARD ERROR*****
1822
1823 002734 000000                RHARD: 0
1824 002736 000000                0
1825 002740 000000                0
1826 002742 000000                0
1827 002744 000000                0
1828 002746 000000                0
1829 002750 000000                0
1830 002752 000000                0
1831
1832                                ;UNIT READ REVERSE HARD ERROR*****
1833
1834 002754 000000                RRHARD: 0
1835 002756 000000                0
1836 002760 000000                0
1837 002762 000000                0
1838 002764 000000                0
1839 002766 000000                0
1840 002770 000000                0
1841 002772 000000                0
1842                                ;SET END OF STATISTICS TABLE
1843 002774                FNDTBL:
1844
1845                                ;DATA PATTERN GENERATORS*****
1846
1847 002774 002774                DATBL: .                ;ENTRY TABLE
1848 002776 013772                DATA0: DAT0                ;EXTERNAL INPUT FROM H/S READER(SEE MAINDEC-11-D2TOP)
1849 003000 014132                DATA1: DAT1                ;ALL ONES
1850 003002 014152                DATA2: DAT2                ;ALL ZEROS
1851 003004 014156                DATA3: DAT3                ;WALKING ONE
1852 003006 014202                DATA4: DAT4                ;WALKING ZERO
1853 003010 014212                DATA5: DAT5                ;ALTERNATING ONE/ZERO
1854 003012 014220                DATA6: DAT6                ;ALTERNATING ZERO/ONE
1855 003014 014226                DATA7: DAT7                ;ALTERNATING ONE/ZERO IN ALTERNATING CHARACTERS
1856 003016 014254                DATA10: DAT10               ;WALKING ONE/ALL ONE IN ALTERNATING CHARACTERS
1857 003020 014304                DATA11: DAT11               ;ALL BITS 0/377
1858 003022 014324                DATA12: DAT12               ;ALL BITS 377/0
1859 003024 014346                DATA13: DAT13               ;ALTERNATING CHARACTERS 0 AND 377
1860 003026 014356                DATA14: DAT14               ;WALKING ZERO/ALL ZERO IN ALTERNATING CHARACTERS
1861 003030 014406                DATA15: DAT15               ;AUTO SEQUENCE PATTERN 0,0,-1,-1,-1,0,0
1862

```



1864  
1865  
1866  
1867  
1868  
1869  
1870  
1871  
1872  
1873  
1874  
1875  
1876  
1877  
1878  
1879  
1880  
1881  
1882  
1883  
1884  
1885  
1886  
1887  
1888  
1889  
1890  
1891  
1892  
1893  
1894  
1895  
1896  
1897  
1898  
1899  
1900  
1901  
1902  
1903  
1904  
1905  
1906  
1907  
1908  
1909  
1910  
(1)  
(1)  
(1)  
(1)  
(1)  
(1)  
(1)  
(1)  
1911

003032 012706 000500  
003036 005037 000742  
  
003042 013746 000004  
003046 012737 003076 000004  
003054 005000  
003056 005737 040000  
003062 005700  
003064 001012  
003066 012737 170000 000560  
003074 000406  
  
003076 012737 174000 000560  
003104 012700 177777  
003110 000002  
  
003112 013700 000560  
003116 005400  
003120 012737 026544 000562  
003126 012737 026544 000564  
003134 060037 000564  
003140 012637 000004  
  
003144 005027  
(1) 003146 000000  
(1)  
(1) 003150 005737 000042  
(1) 003154 001407  
(1) 003156 012737 000176 000616  
(1) 003164 005237 003146  
(1) 003170 000137 003174  
(1) 003174  
003174 122737 000006 000041

```
.EVEN  
;*****  
;PROGRAM START AND SEQUENCE FORMATTER;  
;  
;THIS ROUTINE IS USED TO PERFORM ALL HOUSEKEEPING,  
;DECIDE WHICH TRANSPORT TO TEST AND ITS AVAILABILITY,  
;LOAD THE WRITE BUFFER WITH THE SELECTED DATA PATTERN,  
;GENERATE ANY RANDOM NUMBER AND THEN EXECUTE  
;THE TEST CYCLE REQUESTED BY THE SWITCH SETTING,  
;AT THE END OF THE TEST CYCLE THE NEXT UNIT IS SELECTED  
;AND CHECKED FOR AVAILABILITY AND THE TEST CYCLE IS  
;EXECUTED ON IT.  
;THE READ WRITE STATS MAY BE PRINTED AT THE END OF  
;EACH TEST CYCLE VIA CONSOLE SWITCH FOURTEEN (14).  
;*****  
  
;START 200, & 300*****  
START: MOV 0500,SP ;SET STACK PTR  
CLR ASEQF ;CLEAR AUTO SEQUENCE FLAG  
;...JACH>>>(1/84)  
;THIS NEXT SECTION IS USED TO TEST IF THERE IS MORE THAN 16K  
;OF MEMORY. IF SO, THEN ALLOW LARGE RECORD LENGTHS (10000).  
;OTHERWISE DEFAULT TO 4000 OCTAL.  
;  
MOV @04, -(SP) ;SAVE CONTENTS OF TRAP  
MOV @AA1$, @04 ;SET UP NEW TRAP  
CLR R0 ;CLEAR "TRAP FLAG"  
TST @040000 ;TEST AT THE 16K BOUNDARY  
TST R0 ;CHECK R0 AND SEE IF STILL ZERO  
BNE BB1$ ;IF SO, THEN NO TRAP OCCURRED  
MOV @170000, BUFMAX ;IF NO TRAP, THEN ALLOW 10000 BYTE RECORDS  
BR BB1$ ;AND CONTINUE  
  
AA1$: MOV @174000, BUFMAX ;TRAP THEN ALLOW ONLY 4000 BYTE RECORDS  
MOV @177777, R0 ;SET THAT TRAP OCCURRED  
RTI ;AND RETURN FROM TRAP  
  
BB1$: MOV BUFMAX, R0 ;PUT MAX RECORD LENGTH IN R0 AND CONVERT  
NEG R0 ;IT TO A POSITIVE NUMBER  
MOV @BL* BEG, WDATA ;SET STARTING ADDRESS OF WRITE BUFFER  
MOV @BL* BEG, RDATA ;  
ADD R0, RDATA ;SET STARTING ADDRESS OF READ BUFFER  
MOV (SP), @04 ;RESTORE  
;...JACH<<<  
  
CLR (PC) ;CLEAR CHAIN INDICATOR  
CHNFLG: .WORD 0 ;CHAIN MODE INDICATOR  
;1110 = CHAIN/NOT CHAIN MODE  
TST @04, ? ;BRANCH IF IN DUMP MODE  
BEQ 50$  
MOV @SWREG, SWR ;INVOKE SOFTWARE SWR  
INC CHNFLG ;SET CHNFLG = CHAIN MODE  
JMP 3$ ;GO TO CHAIN ADDRESS  
  
50$:  
3$: CMPB @6, @041 ;BRANCH IF LOADED VIA INDP
```

```

1912 003202 001003      BNE      4$
1913 003204 000004 026314  TYPE,MSG120      ;ADVISE USER TO REMOVE TMDP FROM SLAVE
1914 003210 000000      HALT
1915 003212 005737 003146      4$:  TST      CHNFLG      ;SEE IF IN CHAIN MODE
1916 003216 001406      BEQ      STAUT
1917 003220 005237 000742      INC      ASEQF      ;SET AUTO SEQUENCE FLAG
1918 003224 000004 024357  TYPE,MSG30      ;TYPE TITLE
1919 003230 000137 021342      JMP      ASEGO      ;GO TO AUTO SEQUENCER
1920
1921
1922 003234 012737 000001 000644 ;START 240*****
1923 003240 005037 014520  STAUT:  MOV     01,TINF      ;SET TTY ENTRY FLAG
1924 003246 000405      CLR     RDFL      ;CLEAR RANDOM DATA FLAG
1925
1926
1927 003250 005037 000644 ;START 204*****
1928 003254 000442  STAUT:  CLR     TINF      ;CLEAR TTY INPUT FLAG
1929
1930
1931 003256 005037 000644 ;START 210*****
1932 003262 012700 000646  STAUT:  CLR     TINF      ;CLEAR TTY ENTRY FLAG
1933 003266 012701 000074  STAUT:  MOV     0STFLG,RO    ;GET STARTING ADDRESS OF FLAGS
1934 003270 105020      MOV     0ENDFLG-STFLG,R1
1935 003274 005301      1$:  CLR     (RO)+      ;CLEAR FLAGS AND COUNTERS
1936 003276 001375      DEC     R1
1937 003300 012706 000500      BNE     1$
1938 003304 004737 004234      MOV     0500,SP      ;SET STACK POINTER
1939 003310 012700 001054      JSR     PC,RANSET    ;GO RESET RANDOM BASE
1940 003314 012701 001720      MOV     0STTBL,RO    ;GET STARTING ADDRESS OF STAT TABLE
1941 003320 105020      MOV     0ENDTBL-STTBL,R1 ;AND 0 OF BYTES IN TABLE
1942 003322 005301      2$:  CLR     (RO)+      ;CLEAR STATISTIC COUNTERS
1943 003324 001375      DEC     R1
1944 003326 012700 000752      BNE     2$
1945 003332 022710 177777      MOV     0UN1,RO      ;SET ALL SLAVES ON LINE
1946 003336 001403      3$:  CMP     0-1,(RO)      ;BRANCH IF AT END OF TABLE
1947 003340 042720 040000      BEQ     4$
1948 003344 000772      BIC     040000,(RO)+ ;MARK SLAVE ON-LINE
1949 003346 012737 177777 013766 4$:  BR     3$
1950 003354 012737 000001 000662  STAUT:  MOV     01,BLCNTR    ;PRESET BLOCK COUNTER
1951 003362 013746 000004  STAUT:  MOV     004,(SP)    ;SAVE ERROR TRAP VECTOR
1952 003366 013746 000006      MOV     006,(SP)
1953 003372 022737 000176 000616  CMP     0SWREG,SWR    ;BRANCH IF SOFTWARE SWR
1954 003400 001413      BEQ     2$           ;ALREADY SELECTED
1955 003402 012737 003426 000004  MOV     01$,004      ;SET TIMEOUT TRAP TO 1$ BELOW
1956 003410 005037 000006      CLR     006
1957 003414 022777 177777 175174  CMP     017777,0SWR  ;BRANCH IF SWR = 177777 TRAP
1958 003422 001402      BEQ     3$           ;IF NOT AVAIL (1$) OTHERWISE
1959 003424 000404      BR     3$           ;GO TO 3$
1960 003426 022626      1$:  CMP     (SP)+,(SP)+    ;RESET STACK
1961 003430 012737 000176 000616 2$:  MOV     0SWREG,SWR    ;SET SWR = SOFTWARE SWR
1962 003436 012637 000006      3$:  MOV     (SP)+,006    ;RESTORE ERROR TRAP
1963 003442 012637 000004      MOV     (SP)+,004
1964 003446 012706 000500      MOV     0500,SP
1965 003452 004737 012062      JSR     PC,TINF
1966 003456 012777 000040 175034  MOV     040,0CS
1967 003464 005000  STAUT:  CLR     RO      ;POINT TO FIRST ENTRY

```

1968	003466	022760	177777	000752	1\$:	CMP	# 1,UN1(RO)	;BRANCH IF LAST ENTRY
1969	003474	001406				BEQ	2\$	
1970	003476	042760	100000	000752		BIC	#100000,UN1(RO)	;CLEAR EOT FLAG
1971	003504	062700	000002			ADD	#2,RO	;POINT TO NEXT UNIT ENTRY
1972	003510	000766				BR	1\$	;CONTINUE CLEARING
1973	003512	113737	005043	005043	2\$:	MOVB	REOTC+1,REOTC	;RESTORE EOT COUNTER
1974	003520	012777	000100	175072	START1:	MOV	#100,#TKS	;SET KEYBOARD IE BIT
1975	003526	013700	000702			MOV	UNP,RO	;RO = UNIT TABLE POINTER
1976	003532	022760	177777	000752	STAR1A:	CMP	#-1,UN1(RO)	;BRANCH IF LAST ENTRY
1977	003540	001404				BEQ	STAR1B	
1978	003542	016037	000752	000552		MOV	UN1(RO),UDES	;LOAD NEXT UNIT DESCRIPTION
1979	003550	000445				BR	START4	
1980	003552	005237	000662		STAR1B:	INC	BLCNTR	;BUMP BLOCK COUNTER
1981	003556	005737	000742			TST	ASEQF	;SEE IF AUTO SEQ
1982	003562	001411				BEQ	STAR1C	;IF NOT: BR
1983	003564	023737	000662	000744		CMP	BLCNTR,ABL CNT	;SEE IF DONE SEQ
1984	003572	001005				BNE	STAR1C	;IF NOT: BR
1985	003574	005037	000662			CLR	BLCNTR	;RESET BLOCK CNTR
1986	003600	005037	000702			CLR	UNP	;RESET UNIT POINTER
1987	003604	000207				RTS	PC	;RETURN TO AUTO SEQ
1988	003606	005037	000702		STAR1C:	CLR	UNP	
1989	003612	005000				CLR	RO	
1990	003614	016037	000752	000552		MOV	UN1(RO),UDES	;LOAD FIRST UNIT DESCRIPTION
1991	003622	105777	174770			TSTB	#SWR	;SEE IF RANDOM RECORD SIZE
1992	003626	100002				BPL	START2	;IF NOT: BR
1993	003630	004737	011776			JSR	PC,CCNTR	;GO GENERATE RANDOM RECORD SIZE
1994	003634	032777	000400	174754	START2:	BIT	#400,#SWR	;SEE IF RANDOM DATA
1995	003642	001402				BEQ	START3	;IF NOT: BR
1996	003644	004737	014456			JSR	PC,DATR	;GO GENERATE RANDOM DATA
1997	003650	032777	000100	174740	START3:	BIT	#100,#SWR	;SEE IF RANDOM RECORD COUNT
1998	003656	001402				BEQ	START4	;IF NOT: BR
1999	003660	004737	012036			JSR	PC,RCNTR	;GO GENERATE RANDOM RECORD COUNT
2000	003664	032760	140000	000752	START4:	BIT	#140000,UN1(RO)	;BRANCH IF UNIT AT EOT
2001	003672	001065				BNE	START7	;OR MARKED OFF LINE
2002	003674	012777	000040	174616		MOV	#40,#CS	;DO A MASSBUS CLEAR
2003	003702	013777	000550	174610		MOV	DVN,#CS	;SET DRIVE NUMBER
2004	003710	013777	000552	174624		MOV	UDES,#TC	;SET SLAVE NUMBER
2005	003716	105777	174600		1\$:	TSTB	#DS	;SEE IF SLAVE AVAIL.
2006	003722	100405				BMI	2\$	;IF SO: BR
2007	003724	005337	000674			DEC	STAL	
2008	003730	001372				BNE	1\$	;AWAIT TUR
2009	003732	000137	020426			JMP	OFFLINE	;GO MARK DRIVE OFF LINE
2010	003736	004737	013606		2\$:	JSR	PC,DSUP	;GO SET UP WRITE DATA
2011	003742	004737	005350			JSR	PC,INIT	;INIT SLAVE
2012	003746	004737	005044			JSR	PC,RWND	;REWIND
2013	003752	004737	005464			JSR	PC,WRITE	;WRITE
2014	003756	013737	000606	000674		MOV	TSTAL,STAL	;SET TURN AROUND DELAY
2015	003764	004737	011766			JSR	PC,STAL	;DELAY
2016	003770	004737	007322			JSR	PC,RSEQ	;GO TO READ SEQUENCER
2017	003774	013737	000606	000674		MOV	TSTAL,STAL	;SET TURN AROUND DELAY
2018	004002	004737	011766			JSR	PC,STAL	;DELAY
2019	004006	032777	040000	174602		BIT	#40000,#SWR	;SEE IF SHOULD PRINT STATISTICS
2020	004014	001414				BEQ	START7	;IF NOT: BR
2021	004016	012700	000001			MOV	#1,RO	;SET RECORD COUNTER TO 1
2022	004022	004737	022126			JSR	PC,PAPRT	;PRINT CYCLE NUMBER
2023	004026	004737	004056			JSR	PC,STP	;GO PRINT STATS

CCTEDEF0 IM03 TR16 10:17 DRI  
CZTDEF.P11 07-MAR-84 14:04

MACY11 30(1046) 07-MAR-84 14:21 PAGE 33-3

SEQ 0044

2024	004032	005237	000734		INC	BTSTF		;SET STAT ONLY PRINT
2025	004036	004737	007240		JSR	PC,BTPRT		;PRINT BAD TAPE STATS
2026	004042	005037	000734		CLR	BTSTF		;CLEAR FLAG
2027	004046	062737	000002	000702	START7:	ADD	#2,UNP	;POINT TO NEXT UNIT
2028	004054	000621			START8:	BR	START1	;CONTINUE

```

2030          ;***** SUBROUTINE TO PRINT STATISTICS *****
2031
2032 004056 004737 016504 STP: JSR PC,DPPRT ;PRINT DROPS AND PICKS
2033 004062 000004 025323 TYPE,MSG65 ;TYPE MSG
2034 004066 013700 000702 MOV UNP,R0
2035 004072 016003 001054 MOV RTY1(R0),R3
2036 004076 104400 TYPOCT ;PRINT RETRIES
2037 004100 000004 025434 TYPE,MSG73 ;TYPE MSG
2038 004104 016003 001074 MOV WTER1(R0),R3
2039 004110 104400 TYPOCT ;PRINT WRITE ERRORS
2040 004112 000004 025423 TYPE,MSG72 ;TYPE MSG
2041 004116 016003 001114 MOV RDER1(R0),R3
2042 004122 104400 TYPOCT ;PRINT READ FORWARD ERRORS
2043 004124 000004 026201 TYPE,MSG113 ;TYPE MSG
2044 004130 016003 002674 MOV RFSOFT(R0),R3
2045 004134 104400 TYPOCT ;PRINT FORWARD SOFT ERRORS
2046 004136 000004 026212 TYPE,MSG114 ;TYPE MSG
2047 004142 016003 002734 MOV RFHARD(R0),R3
2048 004146 104400 TYPOCT ;PRINT HARD FORWARE ERRORS
2049 004150 000004 025520 TYPE,MSG77 ;TYPE MSG
2050 004154 016003 001134 MOV DATER1(R0),R3
2051 004160 104400 TYPOCT ;PRINT DATA ERROR FORWARD NUMBER
2052 004162 000004 025355 TYPE,MSG68 ;TYPE MSG
2053 004166 016003 001154 MOV RDERR1(R0),R3
2054 004172 104400 TYPOCT ;PRINT REVESE ERROR NUMBER
2055 004174 000004 026201 TYPE,MSG113 ;TYPE MSG
2056 004200 016003 002714 MOV RRSOFT(R0),R3
2057 004204 104400 TYPOCT ;PRINT REVERSE SOFT ERROR
2058 004206 000004 026212 TYPE,MSG114 ;TYPE MSG
2059 004212 016003 002754 MOV RRHARD(R0),R3
2060 004216 104400 TYPOCT
2061 004220 000004 025507 TYPE,MSG76 ;TYPE MSG
2062 004224 016003 001174 MOV DEREV1(R0),R3
2063 004230 104400 TYPOCT ;PRINT DATA REVERSE ERROR NUMBER
2064 004232 000207 RTS PC ;RETURN
2065
2066          ;RANDOM BASE RESET*****
2067
2068 004234 012737 153624 000634 RANSET: MOV #153624,RANBAS ;RESET BASE
2069 004242 012737 032561 000636 MOV #32561,RANSAV ;RESET BUFFER
2070 004250 013737 000640 000554 MOV RCSAV,RCNT ;RESET RECORD COUNT
2071 004256 013737 000642 000556 MOV FCSAV,FMCNT ;RESET FRAME COUNT
2072 004264 000207 RTS PC
2073

```

```

2075 ;*****
2076 ;REWIND FROM EOT:
2077 ;
2078 ;WHEN ANY TRANSPORT BEING TESTED REACHES END OF TAPE
2079 ;DURING A READ OR WRITE OPERATION, IT WILL BE REWOUND
2080 ;AND FLAGGED AS UNAVAILABLE UNTIL ALL AVAILABLE UNITS
2081 ;HAVE REACHED EOT AT WHICH TIME ALL TESTING WILL BE RESUMED
2082 ;AT A BLOCK COUNT OF ONE (1). A MESSAGE WILL BE
2083 ;PRINTED ON THE SUPERVISORS CONSOLE AS EACH UNIT REACHES
2084 ;EOT AND IS REWOUND.
2085 ;*****
2086
2087 004266 013777 000552 174246 REOT: MOV UDES,0TC ;LOAD TAPE CONTROL REGISTER
2088 004274 013700 000702 MOV UNP,R0 ;GET UNIT POINTER
2089 004300 032760 040000 000752 BIT #40000,UN1(R0) ;BRANCH IF UNIT MARKED OFF-LINE
2090 004306 001014 BNE 2$
2091 004310 012777 000011 174172 MOV #11,0C1 ;DRIVE CLEAR
2092 004316 105777 174200 1$: TSTB #DS ;WAIT FOR DRY
2093 004322 100375 BPL 1$
2094 004324 012777 000007 174156 MOV #7,0C1 ;START REWIND
2095 004332 005737 000732 TST BTFLG ;SEE IF BAD TAPE OVERFLOW REWIND
2096 004336 001004 BNE 3$ ;IF SO: BR
2097 004340 013700 000666 2$: MOV EOTREC,R0
2098 004344 042700 100000 BIC #100000,R0 ;SET RECORD NUMBER OF EOT
2099 004350 005037 000666 3$: CLR EOTREC ;CLEAR EOT INDICATOR & REC COUNT
2100 004354 004737 022126 JSR PC,PAPRT ;PRINT HEADER
2101 004360 022737 000002 000732 CMP #2,BTFLG ;SEE IF POSITION ERROR
2102 004366 001004 BNE 4$ ;IF NOT: BR
2103 004370 012737 026074 004420 MOV #MSG109,6$ ;SET POSITION ERROR MSG
2104 004376 000407 BR 5$
2105 004400 022737 000001 000732 4$: CMP #1,BTFLG ;SEE IF BAD TAPE OVERFLOW
2106 004406 001006 BNE REOT1C ;IF NOT: BR
2107 004410 012737 025727 004420 MOV #MSG106,6$ ;SET BAD TAPE OVERFLOW MSG
2108 004416 000004 5$: TYPE ;TYPE MSG
2109 004420 000000 6$: .WORD 0 ;WILL CONTAIN MESSAGE ADDRESS
2110 004422 000411 BR REOT1E
2111 004424 000004 024060 REOT1C: TYPE,MSG20 ;TYPE EOT MSG
2112 004430 013704 000702 MOV UNP,R4
2113 004434 005264 002654 INC EOTC0(R4) ;BUMP CNTR
2114 004440 016403 002654 MOV EOTC0(R4),R3
2115 004444 104400 TYPOC1 ;PRINT EOT CNTR
2116 004446 000004 025752 REOT1E: TYPE,MSG16A ;TYPE MSG
2117 004452 005037 000732 CLR BTFLG ;CLEAR BAD TAPE FLAG
2118 004456 004737 004056 JSR PC,STP ;PRINT STATS
2119 004462 004737 007240 JSR PC,BTPRT ;PRINT BAD TAPE STATS
2120 004466 013700 000702 REOT2: MOV UNP,R0 ;GET UNIT POINTER
2121 004472 032760 040000 000752 BIT #40000,UN1(R0) ;BRANCH IF UNIT MARKED OFF-LINE
2122 004500 001010 BNE REOT2A
2123 004502 105777 174014 TSTB #DS ;BRANCH IF DRY SET
2124 004506 100405 BMI REOT2A
2125 004510 005337 000674 DEC STAL
2126 004514 001364 BNE REOT2 ;WAIT DRY
2127 004516 000137 000426 JMP OFFLINE ;GO MARK SLAVE OFFLINE
2128
2129 004522 105337 005042 REOT2A: DLCH REOTC ;SEE IF LAST UNIT TO REACH EOT
2130 004526 001410 BEQ REOT3 ;IF SO: BR

```

```

2131 004530 013700 000702          MOV      UNP,RO
2132 004534 052760 100000 000752    BIS      #100000,UN1(RO) ;SET EOT FLAG
2133 004542 005726          TST     (SP)+          ;RESET STACK POINTER
2134 004544 000137 004046          JMP     START7        ;GO TO NEXT UNIT
2135 004550 113737 005043 005042 REOT3:  MOVB   REOTC+1,REOTC ;RESTORE UNITS EOT COUNTER
2136 004556 005037 000702          CLR     UNP
2137 004562 005000          CLR     RO            ;POINT TO FIRST UNIT
2138 004564 016037 000752 000552 REOT4:  MOV     UN1(RO),UDES ;LOAD UNIT DESCRIPTION
2139 004572 013777 000552 173742    MOV     UDES,@TC      ;SELECT SLAVE
2140 004600 032760 040000 000752    BIT     #40000,UN1(RO) ;BRANCH IF UNIT NOT MARKED OFF-LINE
2141 004606 001412          BEQ     1$
2142 004610 032777 010000 173704    BIT     #10000,@DS    ;BRANCH IF MEDIUM NOT ON LINE
2143 004616 001427          BEQ     10$
2144 004620 062737 000401 005042    ADD     #401,REOTC    ;INCREMENT # OF UNITS UNDER TEST
2145 004626 042760 140000 000752    BIC     #140000,UN1(RO) ;MARK UNIT BACK ON-LINE
2146 004634 012777 000011 173646 1$:    MOV     #11,@C1      ;DRIVE CLEAR
2147 004642 105777 173654 2$:    TSTB   @DS           ;WAIT FOR DRIVE READY
2148 004646 100375          HPL     2$
2149 004650 012777 000007 173632    MOV     #7,@C1       ;REWIND UNIT
2150 004656 032777 000002 173636 3$:    BIT     #2,@DS       ;WAIT FOR BOT TO SET
2151 004664 001774          BEQ     3$
2152 004666 032777 020000 173626 4$:    BIT     #20000,@DS   ;WAIT FOR PIP TO CLEAR
2153 004674 001374          BNE     4$           ;AWAIT PIP RESET
2154
2155 004676 042760 100000 000752 10$:   BIC     #100000,UN1(RO) ;CLEAR EOT FLAG
2156 004704 062737 000002 000702    ADD     #2,UNP
2157 004712 013700 000702          MOV     UNP,RO       ;POINT TO NEXT UNIT
2158 004716 022760 177777 000702    CMP     #-1,UN1(RO)  ;BRANCH IF NOT LAST UNIT
2159 004724 001317          BNE     REOT4
2160 004726 005037 000702          CLR     UNP          ;CLEAR UNIT POINTER
2161 004732 005037 000644          CLR     TINF         ;CLEAR TTY INPUT FLAG
2162 004736 005737 000742          TST     ASEQF        ;SEE IF AUTO SEQ
2163 004742 001402          BEQ     REOTX        ;IF NOT; BR
2164 004744 005726          TST     (SP)+        ;RESET STACK POINTER
2165 004746 000207          RTS     PC            ;RETURN TO AUTO SEQ
2166 004750 004737 004234          JSR     PC,RANSET    ;GO RESET RANDOM BASE
2167 004754 012737 177777 013766    MOV     #-1,PATS     ;PRESET PATTERN
2168 004762 005037 014520          CLR     RDFL         ;CLEAR RANDOM FLAG
2169 004766 005737 000600          TST     SPFLG        ;SEE IF SINGLE PASS
2170 004772 001421          BEQ     REOTXX       ;IF NOT; BR
2171 004774 000004 025630          TEND:  TYPE,MSG100   ;TYPE MSG
2172 005000 013700 000042          MOV     @#42,RO      ;GET ACT11 RETURN ADDRESS
2173 005004 001405          BEQ     HERE         ;BRANCH IF NOT ACT11
2174 005006 000005          RESET
2175 005010 004710          $ENDAD: JSR     PC,(RO)
2176 005012 000240          NOP
2177 005014 000240          NOP
2178 005016 000240          NOP
2179 005020 000240          HERE:  NOP
2173 005022 005737 003146          TST     CHN1G        ;BRANCH IF NOT CHAIN MODE
2174 005026 001402          BEQ     1$
2175 005030 000137 001542          JMP     ASEQO        ;RETURN TO AUTO SEQUENCER
2176 005034 000000          1$:    HALT
2177 005036 000137 003354          REOTXX: JMP     STARTL      ;RESTART AT BLOCK NUMBER ONE
2178 005042 000000          REOTC: 0            ;EOT UNIT COUNTER

```

```

2180 ;*****
2181 ;REWIND ALL AVAIL TAPES:
2182 ;
2183 ;THIS ROUTINE; ENTERED VIA CONSOLE SWITCH NINE (9),
2184 ;WILL REWIND ALL AVAILABLE TAPES TO BOT NO MATTER
2185 ;WHERE THEY ARE CURRENTLY POSITIONED AND RESUME TESTING
2186 ;ON THE CURRENTLY SELECTED UNIT.
2187 ;*****
2188
2189 005044 032777 001000 173544 RWND: BIT #1000,@SWR ;SEE IF SHOULD REWIND
2190 005052 001001 BNE RWNDA ;IF SO: BR
2191 005054 00207 RTS PC ;ELSE EXIT
2192 005056 013737 000702 000722 RWNDA: MOV UNP,UPS ;SAVE UNIT POINTER
2193 005064 005037 000702 CLR UNP ;CLEAR POINTER
2194 005070 005037 000666 CLR EOTREC ;CLEAR EOT FLAG
2195 005074 113737 005043 005042 MOV# REOTC+1,REOTC ;..B RESTORE UNIT CTR
2196 005102 013700 000702 RWND0: MOV UNP,RO ;POINT TO UNIT ENTRY
2197 00510 022760 177777 000752 CMP #-1,UN1(RO) ;BRANCH IF LAST ENTRY
2198 005114 001437 BEQ RWND2
2199 005116 032760 140000 000752 BIT #140000,UN1(RO) ;BRANCH IF ALREADY REWINDING
2200 005124 001024 BNE RWND1A ;OR MARKED OFF LINE
2201 005126 016037 000752 000552 MOV UN1(RO),UDES ;SET UNIT DESCRIPTION
2202 005134 013777 000552 173400 MOV UDES,@TC ;LOAD COMMAND REGISTER
2203 005142 012777 000011 173340 MOV #11,@C1 ;DRIVE CLEAR
2204 005150 012777 000007 173332 MOV #7,@C1 ;START REWIND
2205 005156 105777 173340 1$: TSTB @DS
2206 005162 100405 BMI RWND1A ;IF DRY: BR
2207 005164 005337 000674 DEC STAL
2208 005170 001372 BNE 1$ ;AWAIT DRY
2209 005172 000137 020426 JMP OFFLINE ;GO MARK UNIT OFF LINE
2210 005176 042760 100000 000752 RWND1A: BIC #100000,UN1(RO) ;CLEAR EOT FLAG
2211 005204 062737 000002 000702 ADD #2,UNP ;BUMP POINTER
2212 005212 000733 BR RWND0 ;DO NEXT UNIT
2213 005214 005037 000702 RWND2: CLR UNP ;CLEAR POINTER
2214 005220 013700 000702 RWND3: MOV UNP,RO ;POINT TO UNIT ENTRY
2215 005224 022760 177777 000752 CMP #-1,UN1(RO) ;BRANCH IF LAST ENTRY
2216 005232 001433 BEQ RWND0
2217 005234 016037 000752 000552 MOV UN1(RO),UDES ;SET UNIT DESCRIPTION
2218 005242 032760 040000 000752 BIT #40000,UN1(RO) ;BRANCH IF UNIT MARKED OFF LINE
2219 005250 001015 BNE RWND5
2220 005252 013777 000552 173262 MOV UDES,@TC ;LOAD UNIT DESCRIPTION
2221 005260 032777 020000 173234 1$: BIT #20000,@DS
2222 005266 001374 BNE 1$ ;AWAIT PIP RESET
2223 005270 032777 000002 173224 BIT #2,@DS ;BRANCH IF SLAVE AT BOT
2224 005276 001002 BNE RWND5
2225 005300 000137 020426 JMP OFFLINE ;PRINT OFFLINE MESSAGE
2226 005304 062737 000002 000702 RWND5: ADD #2,UNP ;BUMP POINTER
2227 005312 012777 000011 173170 MOV #11,@C1 ;DRIVE CLEAR
2228 005320 000737 BR RWND3 ;DO NEXT UNIT
2229
2230 005322 013700 000722 RWNDX: MOV UPS,RO ;RESTORE UNIT POINTER
2231 005326 010037 000702 MOV RO,UNP
2232 005332 016037 000752 000552 MOV UN1(RO),UDES ;RESET UNIT DESCRIPTION
2233 005340 013777 000552 173174 MOV UDES,@TC
2234 005346 000207 RTS PC ;RETURN TO TEST
2235

```



```

2236
2237
2238
2239
2240
2241
2242
2243
2244 005350 015746 000552      INIT:  MOV    UDES, -(SP)      ;GET UNIT DESCRIPTION
2245 005354 012777 000040 173156  MOV    #40, @CS        ;DO A MASS DIS CLEAR
2246 005362 013777 000550 173130  MOV    DVN, @CS        ;LOAD DRIVE
2247 005370 011677 173146      MOV    (SP), @TC        ;LOAD SLAVE # & SLAVE DESCRIPTION
2248 005374 042716 174377      BIC    #174377, (SP)    ;CLEAR ALL BUT DENSITY BITS
2249 005400 022726 001400      CMP    #1400, (SP)+     ;BRANCH IF NOT NRZ
2250 005404 001005      BNE    1$              ;
2251 005406 032777 000040 173106  BIT    #40, @DS        ;BRANCH IF SLAVE IS IN PE MODE
2252 005414 001422      BEQ    4$              ;PES = 0
2253 005416 000404      BR     2$              ;
2254 005420 032777 000040 173074 1$: BIT    #40, @DS        ;BRANCH IF SLAVE IS IN PE MODE
2255 005426 001015      BNE    4$              ;PES = 1
2256 005430 012777 000007 173052 2$: MOV    #7, @C1        ;LOAD REWIND COMMAND
2257 005436 105777 173060      20$: TSTB   @DS           ;WAIT FOR READY
2258 005442 100375      BPL    20$            ;
2259 005444 032777 020000 173050 3$: BIT    #20000, @DS    ;WAIT FOR PIP = 0
2260 005452 001374      BNE    3$              ;
2261 005454 012777 000011 173026  MOV    #11, @C1        ;CLEAR DRIVE
2262 005462 000207      4$:  RTS    PC          ;

```

```

2264 ;*****
2265 ;WRITE ROUTINE:
2266 ;
2267 ;THIS ROUTINE IS USED TO WRITE ONTO TAPE THE BLOCK
2268 ;OF DATA DESCRIBED BY THE OPERATOR AND SET UP
2269 ;IN THE SEQUENCE FORMATTER. THE TAPE UNIT TO BE USED
2270 ;HAS BEEN ASSIGNED BY THE SEQUENCE FORMATTER AND
2271 ;ITS PARAMETERS SET IN A UNIT DESCRIPTION WORD.
2272 ;AS EACH RECORD OF THE BLOCK IS WRITTEN, IT IS CHECKED
2273 ;FOR STATUS ERRORS, WORD COUNT ZERO, AND CORRECT CURRENT
2274 ;MEMORY ADDRESS. IF THE WRITE OPERATION RESULTS IN
2275 ;ANY ERROR CONDITION, A WRITE RETRY OF THAT OPERATION
2276 ;MAY BE DONE BY SETTING SWITCH FOUR (4) TO A ONE (1).
2277 ;THE RETRY CONSISTS OF A BACKSPACE, ERASE FORWARD, AND
2278 ;REWRITE OF THE RECORD. (SEE WRITE RETRY SUBROUTINE)
2279 ;AFTER ALL DATA RECORDS IN THE BLOCK HAVE BEEN
2280 ;WRITTEN, THE WRITE ROUTINE WILL EXECUTE A WRITE
2281 ;TAPE MARK COMMAND IF THE TTY RESPONSE TM=1 WAS
2282 ;MADE AT INITIAL START. THE TM IS COUNTED AS TOTAL
2283 ;DATA RECORDS PLUS ONE (IE: IF 100 DATA RECORDS; TM=RECORD 101)
2284 ;IF THE WRITE OPERATION (DATA OR TM) CAUSES THE SELECTED SLAVE
2285 ;TO REACH END OF TAPE (EOT) AND THERE IS TO BE NO READING DONE,
2286 ;(SW2 AND SW3 SET TO A 1) THEN THE SLAVE IS REWOUND AND
2287 ;FLAGGED AS UNAVAILABLE FOR TESTING UNTIL ALL SLAVES HAVE
2288 ;REACHED EOT AND BEEN REWOUND AT WHICH TIME TESTING IS
2289 ;RESUMED ON ALL AVAILABLE SLAVES.
2290 ;WRITE RETRY MAY BE ALLOWED VIA CONSOLE SWITCH FOUR (4).
2291 ;ERROR CHECKING MAY BE DISALLOWED VIA CONSOLE SWITCH
2292 ;TWELVE (12).
2293 ;WRITING TO TAPE MAY BE DISALLOWED VIA CONSOLE SWITCH
2294 ;ZERO (0).
2295 ;*****
2296
2297 005464 032777 000001 173124 WRITE: BIT 01,0SWR ;SEE IF SHOULD WRITE
2298 005472 001402 BEQ WRITE
2299 005474 000137 006244 JMP WEX ;IF NOT: BR
2300 005500 013700 000554 WRITE: MOV RCNT,RO ;RO=RECORD COUNT
2301 005504 012737 023746 000660 WO: MOV 0MSG5,EMADDR ;SET ERROR MSG ADDRESS
2302 005512 013777 000556 172776 MOV FMCNT,0FC ;LOAD CHAR COUNT
2303 005520 013777 000562 172766 MOV 00WDATA,0BA ;SET DATA ADDR
2304 005526 112737 000060 000700 MOVB 060,MTC1 ;SET WRITE OP COMMAND
2305 005534 012737 005546 000670 MOV 0W1,RTRN ;SET RETURN ADDRESS
2306 005542 000137 020506 JMP TAPG ;GO EXECUTE COMMAND
2307 005546 032777 002000 172746 W1: EQU 02000,0DS ;SEE IF EOT
2308 005554 001412 EQU 1$ ;IF NOT AT EOT: BR
2309 005556 005737 000666 EQU EOTREC ;BRANCH IF WRITTEN PAST EOT
2310 005562 100407 BMT 1$
2311 005564 005300 DEC RO ;ADJUST # OF RECORDS WRITTEN
2312 005566 052700 100000 BIS 010000,RO ;SET EOT INDICATOR
2313 005572 010037 000666 MOV RO,EOTREC ;SAVE RECORD COUNT
2314 005576 012700 000002 MOV 02,RO ;SET TO WRITE 1 LAST RECORD
2315 005602 032777 010000 173006 1$: BIT 010000,0SWR ;SEE IF SHOULD CHECK ERRORS
2316 005610 001002 BNE 2$ ;IF NOT: BR
2317 005612 004737 016636 JSR PC,ERCHK ;GO CHECK ERRORS
2318 005616 013737 000604 000674 2$: MOV WSTAL,STAL ;SET DELAY
2319 005624 004737 011766 JSR PC,STALL ;DELAY

```

2320	005630	005737	000720		TST	RTYFL		;SEE IF RETRY TIME
2321	005634	001401			BEQ	3\$		;IF NOT: BR
2322	005636	000207			RTS	PC		;ELSE RETURN
2323	005640	005737	000714	3\$:	TST	SERFL		;SEE IF WRITE ERROR
2324	005644	001446			BEQ	W5		;IF NOT: BR
2325	005646	013704	000702		MOV	UNP,R4		
2326	005652	005264	001074		INC	WTER1(R4)		;BUMP WRITE ERROR
2327	005656	005037	000714		CLR	SERFL		;CLEAR STATUS ERROR FLAG
2328	005662	032777	000020	172726	BIT	#20,@SWR		;SEE IF RETRY
2329	005670	001434			BEQ	W5		;IF NOT: BR
2330	005672	013703	000730		MOV	ERSAV,R3		
2331	005676	042703	102720		BIC	#102720,R3		;MASK UNRECOVERABLE ERROR
2332	005702	001407			BEQ	W4		;IF SO: BR
2333	005704	004737	022126		JSR	PC,PAPRT		;PRINT HEADER
2334	005710	000004	025531		TYPE,MSG78			;TYPE MSG
2335	005714	004737	011066		JSR	PC,NRTP		;PRINT ER FOR NON-RETRYABLE
2336	005720	000420			BR	W5		
2337	005722	013704	000702	W4:	MOV	UNP,R4		
2338	005726	005264	001054		INC	RTY1(R4)		;BUMP RETRY CNTR
2339	005732	032777	002000	172656	BIT	#2000,@SWR		;SEE IF PRINT ERRORS
2340	005740	001002			BNE	W4A		;IF NOT: BR
2341	005742	000004	025301		TYPE,MSG64			;TYPE MSG
2342	005746	005037	000710	W4A:	CLR	RTCNT		;CLEAR RETRY NUMBER
2343	005752	005037	000706		CLR	RPCNT		;CLEAR REPEAT COUNTER
2344	005756	004737	006300		JSR	PC,WRTY		;GO RETRY WRITE ERROR
2345	005762	005037	000720	W5:	CLR	RTYFL		;CLEAR RETRY COUNTER
2346	005766	005300			DEC	RO		;SEE IF DONE ALL
2347	005770	001245			BNE	W0		;IF NOT: BR
2348	005772	005737	000572	W6:	TST	TMEX		;SEE IF TM
2349	005776	001522			BEQ	WEX		;IF NOT: BR
2350	006000	005237	000704		INC	TMFLG		;SET TM FLAG
2351	006004	012737	025212	000660	WTM:	MOV	#MSG54,EMADDR	;POINT TO TM ERROR MSG
2352	006012	012737	000026	000700	MOV	#26,MTC1		;SET TM OP CODE
2353	006020	005077	172472		CLR	#FC		;LOAD FRAME COUNTER
2354	006024	013777	000562	172462	MOV	@WDATA,@BA		;LOAD BUS ADDRESS
2355	006032	012737	006044	000670	MOV	#WTMO,RTN		;SAVE RETURN ADDRESS
2356	006040	000137	020506		JMP	TAPG		;WRITE TM
2357	006044	032777	010000	172544	WTMO:	BIT	#10000,@SWR	;SEE IF SHOULD CHECK ERRORS
2358	006052	001074			BNE	WEX		
2359	006054	032777	000004	172440	BIT	#4,@DS		;SEE IF TM STATUS
2360	006062	001011			BNE	WTM1		;IF SO: BR
2361	006064	013737	000562	020340	MOV	@WDATA,CADER		;SET EXPT BUS ADDRESS
2362	006072	012737	000001	020346	MOV	#1,DRVER		;INDICATE ERROR
2363	006100	004737	017466		JSR	PC,ERPT		;PRINT TM ERROR
2364	006104	000404			BR	WTM2		
2365	006106	013703	000562	WTM1:	MOV	@WDATA,R3		;SET EXPT ADDRESS
2366	006112	004737	016730		JSR	PC,ER2		;GO CHECK FOR OTHER ERRORS
2367	006116	005737	000720	WTM2:	TST	RTYFL		;SEE IF RETRY
2368	006122	001401			BEQ	WTM3		;IF NOT: BR
2369	006124	000207			RTS	PC		;ELSE RETURN TO RETRY ROUTINE
2370	006126	005737	000714	WTM3:	TST	SERFL		;SEE IF WRITE ERROR
2371	006132	001444			BEQ	WEX		;IF NOT: BR
2372	006134	013704	000702		MOV	UNP,R4		
2373	006140	005264	001074		INC	WTER1(R4)		;BUMP WRITE ERROR
2374	006144	032777	000020	172444	BIT	#20,@SWR		;SEE IF SHOULD RETRY
2375	006152	001434			BEQ	WEX		;IF NOT: BR

```

2376 006154 013703 000730      MOV      ERSV,R3
2377 006160 042703 102720      BIC      #102720,R3      ;MASK UNRECOVERABLE ERROR
2378 006164 001407              BEQ      WTM4            ;IF SO: BR
2379 006166 004737 022126      JSR      PC,PAPRT       ;PRINT HEADER
2380 006172 000004 025531      TYPE,MSG78             ;TYPE MSG
2381 006176 004737 011066      JSR      PC,NRTP        ;PRINT ER FOR NON-RETRYABLE
2382 006202 000420              BR       WEX
2383 006204 005037 000706      WTM4:  CLR      RPCNT    ;CLEAR REPEAT CNTR
2384 006210 013704 000702      MOV      UNP,R4
2385 006214 005264 001054      INC      RTY1(R4)      ;BUMP RETRY CNTR
2386 006220 005037 000710      CLR      RTCNT         ;CLEAR RETRY CNTR
2387 006224 032777 002000 172364 BIT      #2000,@SWR     ;SEE IF PRINT ERRORS
2388 006232 001002              BNE      WTM4A         ;IF NOT: BR
2389 006234 000004 025301      TYPE,MSG64            ;TYPE MSG
2390 006240 004737 006300      WTM4A: JSR      PC,WRTY   ;GO DO RETRY
2391 006244 005037 000720      WEX:   CLR      RTYFL    ;CLEAR RETRY FLAG
2392 006250 005037 000704      CLR      TMFLG        ;CLEAR TAPE MARK FLAG
2393 006254 005737 000666      TST      EOTREC       ;BRANCH IF NOT AT EOT
2394 006260 100006              BPL      WRWX
2395 006262 032777 000014 172326 WRW:   BIT      #14,@SWR   ;BRANCH IF EITHER READ ENABLED
2396 006270 001002              BNE      WRWX
2397 006272 000137 004266      JMP      REOT         ;ELSE REWIND
2398 006276 000207      WRWX:  RTS      PC     ;EXIT

```

```

2400                                     |*****|
2401                                     |WRITE ERROR RETR|
2402                                     |*****|
2403                                     |*****|
2404                                     |*****|
2405 006300 012737 000001 000720 WRTY:  MOV    #1,RTYFL      ;SET RETRY FLAG
2406 006306 004737 006666 WRTY0:  JSR    PC,WRTSB    ;GO SPACE REVERSE FOR REPEAT
2407 006312 005737 000704          TST    TMFLG          ;SEE IF TAPE MARK TIME
2408 006316 001003          BNE    WRTYTM        ;IF SO: BR
2409 006320 004737 005504          JSR    PC,W0         ;REWRITE RECORD
2410 006324 000402          BR     WRTYR         ;GO ON
2411 006326 004737 006004 WRTYTM: JSR    PC,WTM        ;GO WRITE TAPE MARK AGAIN
2412 006332 005737 000714 WRTYR:  TST    SERFL        ;REWRITE GOOD
2413 006336 001022          BNE    WRTY2        ;IF NOT: BR
2414 006340 005237 000706          INC    RPCNT        ;BUMP REPEAT COUNTER
2415 006344 022737 000004 000706          CMP    #4,RPCNT     ;SEE IF FOUR GOOD REPEATS
2416 006352 001355          BNE    WRTY0        ;IF NOT: REPEAT
2417 006354 032777 002000 172234          BIT    #2000,#SWR   ;SEE IF PRINT
2418 006362 001007          BNE    WRTY1        ;IF NOT: BR
2419 006364 000004 025714          TYPE,MSG105        ;TYPE MSG
2420 006370 000004 025323          TYPE,MSG65        ;TYPE MSG
2421 006374 013703 000710          MOV    RTCNT,R3
2422 006400 104400          TYPOCT            ;PRINT RETRY NUMBER
2423 006402 000207 WRTY1:  RTS    PC         ;RESUME TESTING
2424 006404 013703 000730 WRTY2:  MOV    ERSA7,R3    ;GET ER
2425 006410 005037 000656          CLR    TEMP3        ;CLEAR RECOVERABLE ERROR INDICATOR
2426 006414 042703 102720          BIC    #102720,R3   ;MASK RECOVERABLE BITS
2427 006420 001412          BEQ    WRTY2A       ;IF RECOVERABLE: BR
2428 006422 004737 022126          JSR    PC,PAPRT     ;PRINT HEADER
2429 006426 000004 025531          TYPE,MSG78        ;TYPE MSG
2430 006432 004737 011066          JSR    PC,NRTP      ;PRINT ER
2431 006436 012737 000001 000656          MOV    #1,TEMP3    ;SET FLAG
2432 006444 000406          BR     WRTY2B       ;
2433 006446 032777 002000 172142 WRTY2A: BIT    #2000,#SWR   ;SEE IF PRINT
2434 006454 001022          BNE    WRTY3        ;IF NOT: BR
2435 006456 000004 026124          TYPE,MSG110       ;TYPE MSG
2436 006462 000004 025323 WRTY2B: TYPE,MSG65        ;TYPE MSG
2437 006466 013703 000710          MOV    RTCNT,R3
2438 006472 104400          TYPOCT            ;PRINT RETRY NUMBER
2439 006474 000004 026146          TYPE,MSG111       ;TYPE MSG
2440 006500 013703 000706          MOV    RPCNT,R3
2441 006504 104400          TYPOCT            ;PRINT REPEAT NUMBER
2442 006506 005737 000656          TST    TEMP3        ;SEE IF DID NON RECOVERABLE
2443 006512 001403          BEQ    WRTY3        ;IF NOT: BR
2444 006514 005037 000656          CLR    TEMP3        ;CLEAR FLAG
2445 006520 000207          RTS    PC         ;EXIT
2446 006522 005737 000710 WRTY3:  TST    RTCNT        ;SEE IF FIRST RETRY
2447 006526 001004          BNE    WRTY3A       ;IF NOT: BR
2448 006530 013704 000702          MOV    UNP,R4
2449 006534 005364 001074          DEC    WTR1(R4)     ;DECREMENT WRITE ERROR CNTR
2450 006540 013704 000702 WRTY3A: MOV    UNP,R4        ;GET UNIT NUMBER
2451 006544 016437 001034 000736          MOV    BTADDR(R4),B1PT ;GET ADDRESS OF UNIT BAD TAPE CNTR
2452 006552 017704 172160          MOV    #B1PT,R4    ;GET COUNTER
2453 006556 005737          TST    (R4)         ;SET POINTER OFFSET
2454 006560 010477 172152          MOV    R4,#B1PT
2455 006564 013703 000736          MOV    B1PT,R3

```

CS

```
2456 006570 060304      ADD      R3,R4          ;SET ABSOLUTE POINTER
2457 006572 013714 000662  MOV      BLCNTR,(R4)    ;SET BLOCK NUMBER
2458 006576 062704 000040  ADD      @40,R4        ;ADD RCNT OFFSET
2459 006602 013714 000554  MOV      RCNT,(R4)
2460 006606 160014      SUB      R0,(R4)        ;SET RECORD NUMBER
2461 006610 005214      INC      (R4)          ;CORRECT RECORD NUMBER
2462 006612 022777 000040 172116  CMP      @40,@BTPT     ;SEE IF TOO MANY BAD SPOTS
2463 006620 001002      BNF      WRTY4         ;IF NOT: BR
2464 006622 000137 007104      JMP      BT0V          ;ELSE GO TO BAD TAPE OVERFLOW
2465 006626 005237 000710      WRTY4: INC      RTCNT    ;BUMP RETRY COUNTER
2466 006632 022737 000004 000710  CMP      @4,RTCNT     ;SEE IF DONE 4 RETRIES
2467 006640 001410      BEQ      WRTY5        ;IF SO: BR
2468 006642 013704 000702      MOV      UNP,R4
2469 006646 005264 001054      INC      RTY1(R4)     ;BUMP RETRY COUNTER
2470 006652 005237 000740      INC      ERTFL        ;SET ERASE FLAG
2471 006656 000137 006306      JMP      WRTY0        ;DO NEXT RETRY
2472 006662 000137 007310      WRTY5: JMP      BT0R        ;ELSE GO TO BAD TAPE UNRECOVERABLE
2473
2474
2475      ;WRITE RETRY BACKSPACE-ERASE SUBROUTINE*****
2476 006666 005037 000714      WRTSB: CLR      SERFL      ;CLEAR FLAG
2477 006672 013737 000606 000674  MOV      TSTAL,STAL
2478 006700 004737 011766      JSR      PC,STALL     ;DO TURN AROUND DELAY
2479 006704 012737 025334 000660  MOV      @MSG66,EMADDR ;SET ERROR CODE
2480 006712 012777 177777 171576  MOV      @1,@FC       ;SET TO BACKSPACE 1 RECORD
2481 006720 013703 000564      MOV      @@RDATA,R3  ;SET EXPECTED BA
2482 006724 010377 171564      MOV      R3,@BA
2483 006730 012737 000032 000700  MOV      @32,MTCL     ;SET BACK SPACE OP CODE
2484 006736 012737 006750 000670  MOV      @1,@RTN      ;SET RETURN PC
2485 006744 000137 020506      JMP      TAPG        ;EXECUTE BACKSPACE COMMAND
2486 006750 004737 016730      1$: JSR      PC,ER2     ;CHECK ERRORS
2487 006754 004737 011766      JSR      PC,STALL    ;STALL
2488 006760 005737 000714      TST      SERFL       ;SEE IF ERROR
2489 006764 001406      BEQ      WRTSB1      ;IF NOT: BR
2490 006766 012737 000002 000732  WRTSB0: MOV      @2,BTFLG    ;SET FLAG
2491 006774 022626      CMP      (SP)+,(SP)+ ;RESET STACK
2492 006776 000137 004266      JMP      RE0T        ;GO REWIND AND REMOVE FROM TESTING
2493 007002 005737 000740      WRTSB1: TST      ERTFL   ;SEE IF SHOULD ERASE
2494 007006 001001      BNE      WRTSB2      ;IF SO: BR
2495 007010 000207      RTS      PC          ;RETURN
2496 007012 005037 000740      WRTSB2: CLR      ERTFL   ;CLEAR ERASE FLAG
2497 007016 005037 000706      CLR      RPCNT       ;CLEAR REPEAT CNTR
2498 007022 005037 000714      CLR      SERFL       ;CLEAR FLAG
2499 007026 012737 025346 000660  MOV      @MSG67,EMADDR ;SET ERROR CODE
2500 007034 005077 171456      CLR      @FC         ;CLEAR FRAME COUNT
2501 007040 012737 000024 000700  MOV      @4,MTCL     ;SET ERASE OP-CODE
2502 007046 013703 000562      MOV      @@WDATA,R3  ;SET EXPECTED BA
2503 007052 010377 171436      MOV      R3,@BA
2504 007056 012737 007070 000670  MOV      @1,@RTN      ;SET RETURN ADDRESS
2505 007064 000137 020506      JMP      TAPG        ;GO ERASE
2506 007070 004737 016730      1$: JSR      PC,ER1     ;GO CHECK ERRORS
2507 007074 005737 000714      TST      SERFL       ;SEE IF ERROR
2508 007100 001740      BEQ      WRTSB1      ;IF NOT: BR
2509 007102 000731      BR       WRTSB0
2510
2511      ;BAD TAPE OVERFLOW SUBROUTINE*****
```



```
2547  
2548  
2549  
2550 007240 000004 024355  
2551 007244 013704 000702  
2552 007250 016437 001034 000736  
2553 007256 017703 171454  
2554 007262 000241  
2555 007264 000003  
2556 007266 104400  
2557 007270 000004 026160  
2558 007274 005777 171436  
2559 007300 001001  
2560 007302 000207  
2561 007304 000137 007124  
2562  
2563  
2564  
2565 007310 004737 022126  
2566 007314 000004 026013  
2567 007320 000207  
2568
```

```
      ;BAD TAPE STATISTIC PRINT*****  
BTPRT: TYPE,MSG28          ;TYPE '<CR><LF>'  
      MOV      UNP,R4  
      MOV      BTADDR(R4),BTPT ;SET TABLE POINTER  
      MOV      @BTPT,R3  
      CLC  
      ROR      R3          ;CORRECT NUMBER  
      TYP OCT          ;PRINT NUMBER OF BAD SPOTS  
      TYPE,MSG112        ;TYPE MSG  
      TST      @BTPT      ;SEE IF ANY BAD SPOTS  
      BNE      BTPRT1     ;IF SO: BR  
      RTS      PC          ;ELSE RETURN  
BTPRT1: JMP      BTOVO     ;PRINT STATS  
  
      ;BAD TAPE UNRECOVERABLE SUBROUTINE*****  
BTUR:  JSR      PC,PAPRT   ;PRINT HEADER  
      TYPE,MSG107        ;TYPE MSG  
      RTS      PC          ;RESUME TESTING
```



```

2570 ;*****
2571 ;READ SEQUENCER:
2572 ;
2573 ;THIS ROUTINE IS USED TO DETERMINE THE SEQUENCE
2574 ;IN WHICH READ TAPE OPERATIONS ARE TO BE PERFORMED.
2575 ;THIS IS NECESSARY WHEN THE UNIT BEING TESTED IS
2576 ;CAPABLE OF READING DATA IN BOTH THE FORWARD AND
2577 ;REVERSE DIRECTIONS.  CONSOLE SWITCHES ONE (1), TWO (2),
2578 ;AND THREE (3) ARE USED TO DETERMINE THE READ SEQUENCE.
2579 ;CONSOLE SWITCH ONE (1) DETERMINES WHETHER TO READ
2580 ;THE BLOCK OF DATA FORWARD FIRST OR REVERSE FIRST.
2581 ;SWITCH TWO (2) DISALLOWS READING IN THE REVERSE
2582 ;DIRECTION AND SWITCH THREE (3) DISALLOWS READING IN
2583 ;THE FORWARD DIRECTION.
2584 ;*****
2585
2586 007322 005037 000570 RSEQ: CLR RDCMD
2587 007326 017704 171264 MOV @SWR,R4 ;READ SWITCHES
2588 007332 042704 177763 BIC #177763,R4 ;MASK READ BITS & SEE IF BOTH READS
2589 007336 001004 BNE RSR ;IF NOT: BR
2590 007340 032777 000002 171250 BIT #2,@SWR ;SEE IF READ REVERSE FIRST
2591 007346 001041 BNE RSFR ;IF NOT: BR
2592 007350 032777 000004 171240 RSR: BIT #4,@SWR ;SEE IF SHOULD READ REVERSE
2593 007356 001005 BNE RSF ;IF NOT: BR
2594 007360 012737 000001 000570 MOV #1,RDCMD ;LOAD READ REVERSE COMMAND
2595 007366 004737 007576 JSR PC,READ ;GO READ REVERSE
2596 007372 032777 000010 171216 RSF: BIT #10,@SWR ;SEE IF SHOULD READ FORWARD
2597 007400 001066 BNE RSEX ;IF NOT: BR
2598 007402 005737 000570 TST RDCMD ;SEE IF HAVE READ REVERSE
2599 007406 001406 BEQ RSFO ;IF NOT: BR
2600 007410 013737 000606 000674 MOV TSTAL,STAL
2601 007416 004737 011766 JSR PC,STALL ;DO READ STALL
2602 007422 000406 BR RSF1
2603 007424 032777 000001 171164 RSFO: BIT #1,@SWR ;SEE IF WRITE
2604 007432 001002 BNE RSF1 ;IF NOT: BR
2605 007434 004737 011514 JSR PC,BKSP ;GO BACKSPACE
2606 007440 005037 000570 RSF1: CLR RDCMD ;LOAD READ FORWARD COMMAND
2607 007444 004737 007576 JSR PC,READ ;GO READ
2608 007450 000442 BR RSEX ;GO TO EXIT
2609
2610 007452 012737 000001 000570 RSFR: MOV #1,RDCMD
2611 007460 032777 000010 171130 BIT #10,@SWR ;SEE IF SHOULD READ FORWARD
2612 007466 001012 BNE RSFR1 ;IF NOT: BR
2613 007470 032777 000001 171120 BIT #1,@SWR ;SEE IF WRITE
2614 007476 001002 BNE RSFR0 ;IF NOT: BR
2615 007500 004737 011514 JSR PC,BKSP ;GO BACKSPACE TO START
2616 007504 005037 000570 RSFR0: CLR RDCMD ;LOAD READ FORWARD COMMAND
2617 007510 004737 007576 JSR PC,READ ;GO READ FORWARD
2618 007514 032777 000004 171074 RSFR1: BIT #4,@SWR ;SEE IF SHOULD READ REVERSE
2619 007522 001015 BNE RSEX ;IF NOT: BR
2620 007524 005737 000570 TST RDCMD
2621 007530 001005 BNE RSFR2 ;IF READ REVERSE: BR
2622 007532 013737 000606 000674 MOV TSTAL,STAL ;DO READ STALL
2623 007540 004737 011766 JSR PC,STALL
2624 007544 012737 000001 000570 RSFR2: MOV #1,RDCMD ;LOAD READ REVERSE
2625 007552 004737 007576 JSR PC,READ ;GO READ REVERSE

```

CZTEDEO 1M03-TE16 1077 DRI  
CZTEDEF.P11 07-MAR-84 14:04

MACY11 30(1046) 07-MAR-84 14:21 PAGE 39-1

SEQ 0058

2626	007556	005037	000570
2627	007562	005737	000666
2628	007566	001402	
2629	007570	000137	004266
2630	007574	000207	
2631			

RSEX:	CLR	RDCMD	
	TST	EOTREC	;BRANCH IF EOT NOT REACHED
	BEQ	RSFRX	
	JMP	REOT	;REWIND AND REPORT STATS
RSFRX:	RTS	PC	;EXIT

2633  
2634  
2635  
2636  
2637  
2638  
2639  
2640  
2641  
2642  
2643  
2644  
2645  
2646  
2647  
2648  
2649  
2650  
2651  
2652  
2653  
2654  
2655  
2656  
2657  
2658  
2659  
2660  
2661  
2662  
2663  
2664  
2665  
2666  
2667  
2668  
2669  
2670  
2671  
2672  
2673  
2674  
2675  
2676  
2677  
2678  
2679  
2680  
2681  
2682  
2683  
2684  
2685  
2686  
2687  
2688

007576 013700 000554  
007602 005737 000666  
007606 100012  
007610 005737 000570  
007614 001407  
007616 042737 100000 000666  
007624 013703 000666  
007630 160300  
007632 005200  
007634 012737 023753 000660  
007642 005037 000704  
007646 005737 000570  
007652 001406  
007654 005737 000572  
007660 001403  
007662 005237 000704  
007666 005200  
007670 013777 000556 170620  
007676 013777 000564 170610  
007704 005737 000570  
007710 001417  
007712 013703 000556  
007716 005103  
007720 032737 000020 000552  
007726 001402  
007730 000241  
007732 006003  
007734 060377 170554  
007740 012737 000076 000700  
007746 000403

```
*****  
;READ ROUTINE:  
;  
;THIS ROUTINE PERFORMS THE READ OPERATION DETERMINED  
;BY THE READ SEQUENCE ROUTINE ONE RECORD AT A TIME.  
;AT THE END OF EACH READ OPERATION THE STATUS REGISTER  
;IS SCANNED FOR EITHER END OF TAPE OR BEGINNING OF TAPE.  
;IF EOT WAS REACHED, CONTROL WILL BE PASSED TO  
;THE EOT SUBROUTINE TO REWIND THE UNIT AND FLAG IT  
;UNAVAILABLE UNTIL ALL UNITS HAVE REACHED EOT.  
;IF BOT WAS REACHED AN ERROR IS PRINTED AND THE  
;PROGRAM WILL HALT. TESTING MAY BE RESUMED BY PRESSING  
;THE CONTINUE SWITCH.  
;IF A TAPE MARK IS EXPECTED (TM=1) THEN THE  
;READ ROUTINE EXPECTS THE FIRST RECORD OF A  
;READ REVERSE TO BE A TM, AND THE LAST RECORD  
;OF A READ FORWARD TO BE A TM. REMEMBER  
;THAT THE TM ADDS ONE (1) TO THE TOTAL NUMBER  
;OF RECORDS IN A BLOCK.  
;CONSOLE SWITCHES ELEVEN (11) AND THIRTEEN (13) DETERMINE WHETHER  
;OR NOT TO CHECK FOR STATUS ERRORS (11) OR DATA ERRORS (13).  
;CONSOLE SWITCH FIVE (5) IS USED TO CAUSE A CONTINUOUS  
;READ AND SPACE (FORWARD OR REVERSE) OF THE CURRENT  
;RECORD ON TAPE (YOZZLE).  
*****
```

```
READ:  MOV    RCNT,R0      ;LOAD REC CNTR  
       TST    EOTREC     ;SEE IF EOT  
       BPL    RDA        ;IF NOT: BR  
       TST    RDCMD      ;SEE IF READ FORWARD  
       BEQ    RDA        ;IF SO: BR  
       BIC    #100000,EOTREC ;CLEAR FLAG  
       MOV    EOTREC,R3   ;GET MODIFIED RECORD COUNT  
       SUB    R3,R0      ;SET RECORD AT  
       INC    R0         ;SET TO PROPER NUMBER OF RECORDS  
RDA:   MOV    #MSG6,EMADDR ;SET ERROR MSG ADDRESS  
       CLR    TMFLG  
       TST    RDCMD  
       BEQ    RDO        ;IF READ FORWARD: BR  
       TST    TMEX      ;SEE IF TM  
       BEQ    RDO        ;IF NOT: BR  
       INC    TMFLG     ;SET TM FLAG  
       INC    R0  
RDO:   MOV    FMCNT,#C    ;LOAD CHAR CNTR  
       MOV    @RDATA,@BA ;LOAD DATA ADDR  
       TST    RDCMD      ;SEE IF READ REVERSE  
       BEQ    R01A      ;IF NOT: BR  
       MOV    FMCNT,R3  
       COM    R3  
       BIT    #20,UDES   ;SEE IF CORE DUMP  
       BEQ    R01      ;IF NOT: BR  
       CLC  
       ROR    R3        ;R3 = FC2  
R01:   ADD    R3,@BA     ;SET REVERSE BUS ADDRESS  
       MOV    #76,MTC1  ;SET READ REVERSE  
       BR    R01A
```

```

2689 007750 012737 000070 000700 RD1A: MOV #70,MTC1 ;SET READ FORWARD
2690 007756 012737 007770 000670 RD1B: MOV #RD2,RTRN ;SET INTERRUPT RETURN ADDRESS
2691 007764 000137 020506 JMP TAPG ;GO EXECUTE TAPE COMMAND
2692 007770 005737 000570 RD2: TST RDCMD ;IGNORE EOT IF READ REVERSE
2693 007774 001014 BNE RD3
2694 007776 032777 002000 170516 BIT #2000,SDS ;SEE IF EOT
2695 010004 001410 BEQ RD3 ;IF NOT: BR
2696 010006 005737 000704 TST TMFLG ;SEE IF TM
2697 010012 001005 BNE RD3 ;IF SO: BR
2698 010014 010037 000666 MOV R0,EOTREC ;GET # OF RECORDS LEFT IN BLOCK TO READ
2699 010020 052737 100000 000666 BIS #100000,EOTREC ;SET EOT FLAG
2700 010026 032777 000002 170466 RD3: BIT #2,SDS ;SEE IF AT LOAD POINT
2701 010034 001407 BEQ RD4 ;IF NOT: BR
2702 010036 004737 022126 JSR PC,PAPRT ;PRINT CYCLE NUMBER
2703 010042 000004 024113 TYPE,MSG2 ;TYPE MSG
2704 010046 000000 HALT
2705 010050 000137 003256 JMP STARTA ;RESTART
2706 010054 032777 004000 170534 RD4: BIT #4000,SWR ;SEE IF SHOULD CHECK ERRORS
2707 010062 001116 BNE RD5 ;IF NOT: BR
2708 010064 005737 000704 TST TMFLG
2709 010070 001470 BEQ RD4B ;IF NO TM EXPT: BR
2710 010072 032777 000004 170422 BIT #4,SDS
2711 010100 001023 BNE RD4A ;IF TM RECVD: BR
2712 010102 013737 000564 020340 MOV #RDATA,CADER ;SAVE EXPT BUS ADDRESS
2713 010110 012737 000002 020346 MOV #2,DRVR ;SET TM STATUS ERROR FLAG
2714 010116 004737 017466 JSR PC,ERPT ;GO PRINT TM ERROR
2715 010122 013704 000702 MOV UNP,R4
2716 010126 005737 000570 TST RDCMD ;SEE IF READ REVERSE
2717 010132 001403 BEQ 1 ;IF NOT: BR
2718 010134 005264 001154 INC RDER1(R4) ;BUMP READ REVERSE ERROR
2719 010140 000500 BR RD6
2720 010142 005264 001114 1: INC RDER1(R4) ;BUMP READ FORWARD ERROR
2721 010146 000475 BR RD6
2722 010150 013703 000564 RD4A: MOV #RDATA,R2
2723 010154 005737 000570 TST RDCMD ;SEE IF READ REVERSE
2724 010160 001007 BNE RD4A0 ;IF SO: BR
2725 010162 032737 002000 000552 BIT #2000,ODES ;SEE IF IN PE
2726 010170 001025 BNE RD4A2 ;IF SO: BR
2727 010172 062703 000002 ADD #2,R3
2728 010176 000422 BR RD4A2
2729 010200 013704 000556 RD4A0: MOV FMCNT,R4
2730 010204 005104 COM R4
2731 010206 032737 000020 000552 BIT #20,ODES ;SEE IF CORE DUMP
2732 010214 001402 BEQ RD4A1 ;IF NOT: BR
2733 010216 000241 CLC
2734 010220 006004 ROR R4 ;SET TO FC/2
2735 010222 060403 RD4A1: ADD R4,R3 ;SET EXPT BUS ADDRESS
2736 010224 042703 000001 BIC #1,R3 ;MAKE EXPT ADDRESS EVEN
2737 010230 032737 002000 000552 BIT #2000,ODES ;SEE IF IN PE
2738 010236 001002 BNE RD4A2 ;IF SO: BR
2739 010240 162703 000002 SUB #2,R3
2740 010244 004737 016730 RD4A2: JSR PC,ER2
2741 010250 000402 BR RD4C
2742 010252 004737 016636 RD4B: JSR PC,ERCHK ;GO CHECK ERRORS
2743 010256 005737 000714 RD4C: TST SERFL
2744 010262 001416 BEQ RD5 ;IF NO ERROR: BR

```

2745	010264	013704	000702		MOV	UNP,R4	
2746	010270	005737	000570		TST	RDCMD	;SEE IF READ REVERSE
2747	010274	001003			BNE	RD4D	;IF SO: BR
2748	010276	005264	001114		INC	RDER1(R4)	;BUMP READ FORWARD ERROR
2749	010302	000402			BR	RD4E	
2750	010304	005264	001154	RD4D:	INC	RDERR1(R4)	;BUMP READ REVERSE ERROR
2751	010310	004737	010506	RD4E:	JSR	PC,RDRY	;GO RETRY
2752	010314	005037	000720		CLR	RTYFL	;CLEAR RETRY FLAG
2753	010320	032777	020000	170270	RD5:	BIT	#20000,@SWR
2754	010326	001005			BNE	RD6	;SEE IF SHOULD DO DATA CHECK
2755	010330	005737	000704		TST	TMFLG	;IF NOT: BR
2756	010334	001002			BNE	RD6	
2757	010336	004737	015064		JSR	PC,DCHK	;GO CHECK DATA
2758	010342	005037	000714	RD6:	CLR	SEFL	;CLEAR STATUS ERROR FLAG
2759	010346	004737	C13730		JSR	PC,DS3	;CLEAR BUFFER
2760	010352	032777	000040	170236	BIT	#40,@SWR	;SEE IF SHOULD YOZZLE
2761	010360	001402			BEQ	RD7	;IF NOT: BR
2762	010362	004737	011102		JSR	PC,YOZ	;ELSE GO YOZZLE
2763	010366	013737	000602	000674	RD7:	MOV	RSTAL,STAL
2764	010374	004737	011766		JSR	PC,STALL	;SET DELAY
2765	010400	005737	000570		TST	RDCMD	;STALL
2766	010404	001403			BEQ	RD7A	;SEE IF READ REVERSE
2767	010406	005037	000704		CLR	TMFLG	;IF NOT: BR
2768	010412	000405			BR	RD10	;CLEAR TAPE MARK FLAG
2769	010414	005737	000666	RD7A:	TST	EOTREC	;SEE IF EOT FOUND
2770	010420	100002			BPL	RD10	;IF NOT: BR
2771	010422	012700	000001		MOV	#1,R0	;SET TO EOT
2772	010426	005300		RD10:	DEC	R0	
2773	010430	001402			BEQ	RD11	;IF DONE ALL: BR
2774	010432	000137	007670		JMP	R00	
2775	010436	005737	000570	RD11:	TST	RDCMD	;SEE IF READ REVERSE
2776	010442	001016			BNE	RDEX	;IF SO: BR
2777	010444	005737	000666		TST	EOTREC	;SEE IF FOUND EOT
2778	010450	100413			BMI	RDEX	;IF SO: BR
2779	010452	005737	000572		TST	TMEX	;SEE IF TM EXPECTED
2780	010456	001410			BEQ	RDEX	;IF NOT: BR
2781	010460	005737	000704		TST	TMFLG	;SEE IF TM FOUND
2782	010464	001005			BNE	RDEX	;IF SO: BR
2783	010466	005237	000704		INC	TMFLG	;ELSE SET FLAG
2784	010472	005200			INC	R0	;SET RECORD COUNT TO ONE
2785	010474	000137	007670		JMP	R00	;GO READ TM
2786	010500	005037	000704	RDX:	CLR	TMFLG	
2787	010504	000207		RDX:	RTS	PC	;EXIT

```

2789 ;*****
2790 ;READ ERROR RETRY SUBROUTINE:
2791 ;
2792 ;THIS SUBROUTINE WILL RETRY ALL DATA RELATED
2793 ;READ ERRORS UP TO EIGHT (8) TIMES. IF ALL
2794 ;FOUR RETRIES ARE BAD, IT IS CONSIDERED
2795 ;A HARD ERROR. IF ANY ARE GOOD, IT IS A
2796 ;SOFT ERROR. RETRIES MAY BE INHIBITED
2797 ;VIA SWITCH FOUR (SW4=0: INHIBIT RETRIES)
2798 ;*****
2799
2800 010506 032777 000020 170102 RDRTY: BIT #20,@SWR ;SEE IF RETRY INHIBITED
2801 010514 001001 BNE RDRT0 ;IF NOT: BR
2802 010516 000207 RTS PC ;ELSE RETURN
2803
2804 010520 013703 000730 RDRT0: MOV ERSAV,R3
2805 010524 022703 100000 CMP #100000,R3 ;++B BRANCH IF OTHER THAN CORRECTED READ ERROR
2806 010530 001011 BNE 1$ ;++B
2807 010532 032777 000040 167762 BIT #40,@DS ;++B BRANCH IF NRZ
2808 010540 001405 BEQ 1$ ;++B
2809 010542 005037 000714 CLR SERFL ;++B CLEAR ERROR FLAG
2810 010546 000004 026504 TYPE,MSG124 ;++B TYPE 'CORRECTED PE DATA ERROR'
2811 010552 000447 BR RDRT2 ;++B INC SOFT COUNTS
2812 010554 042703 102720 1$: BIC #102720,R3 ;MARK NON-RECOVERABLE ERROR BITS
2813 010560 001407 BEQ RDRT1 ;IF NOT: BR
2814 010562 004737 022126 JSR PC,PAPRT ;PRINT HEADER
2815 010566 000004 025571 TYPE,MSG79 ;TYPE MSG
2816 010572 004737 011066 JSR PC,NRTP ;PRINT ER FOR NON-RETRYABLE ERROR
2817 010576 000207 RTS PC ;RETURN
2818 010600 032777 002000 170010 RDRT1: BIT #2000,@SWR ;SEE IF PRINT INHIBITED
2819 010606 001002 BNE RDRT18 ;IF SO: BR
2820 010610 000004 025301 TYPE,MSG64 ;TYPE MSG
2821 010614 005037 000710 RDRT18: CLR RTCNT ;CLEAR RETRY COUNTER
2822 010620 005037 000714 RDRTG: CLR SERFL ;CLEAR STATUS ERROR FLAG
2823 010624 012737 000002 000720 MOV #2,RTYFL ;SET READ RETRY FLAG
2824 010632 004737 011102 JSR PC,YOZ ;GO TO YOZZLE TO RETRY READ
2825 010636 005737 000714 TST SERFL ;SEE IF RETRY ERROR
2826 010642 001026 BNE RDRT5 ;IF SO: BR
2827 010644 032777 002000 167744 BIT #2000,@SWR
2828 010652 001007 BNE RDRT2
2829 010654 000004 025714 TYPE,MSG105 ;TYPE MSG
2830 010660 000004 025323 TYPE,MSG65 ;TYPE MSG
2831 010664 013703 000710 MOV RTCNT,R3
2832 010670 104400 TYPUCT ;PRINT RETRY NUMBER
2833 010672 013704 000702 RDRT2: MOV UNP,R4
2834 010676 005737 000570 TST RDCMD ;SEE IF READ REVERSE
2835 010702 001003 BNE RDRT3 ;IF SO: BR
2836 010704 005264 002674 INC RFSOFT(R4) ;ELSO BUMP FORWARD SOFT ERROR COUNTER
2837 010710 000402 BR RDRT4
2838 010712 005264 002714 RDRT3: INC RRSOFT(R4) ;BUMP ERRORS SOFT CNTR
2839 010716 000207 RDRT4: RTS PC ;RETURN
2840 010720 013703 000730 RDRT5: MOV ERSAV,R3 ;GET ER
2841 010724 005037 000656 CLR TEMP3 ;CLEAR RECOVERABLE ERROR INDICATOR
2842 010730 042703 102720 BIC #102720,R3 ;MASK RECOVERABLE BITS
2843 010734 001417 BEQ RDRT5A ;IF RECOVERABLE: BR
2844 010736 004737 022126 JSR PC,PAPRT ;PRINT HEADER

```

```

2845 010742 000004 025571          TYPE,MSG79          ;TYPE MSG
2846 010746 004737 011066          JSR      PC,NRTF    ;PRINT ER
2847 010752 012737 000001 000656  MOV      #1,TEMP3   ;SET FLAG
2848 010760 000404                    BR      RDRT5B
2849 010762 032777 002000 167626  RDRT5A:  BIT      #2000,@SWR ;SEE IF PRINT INHIBITED
2850 010770 001013                    BNE     RDRT6       ;IF SO: BR
2851 010772 000004 025323          RDRT5B:  TYPE,MSG65  ;TYPE MSG
2852 010776 013703 000710          MOV      RTCNT,R3
2853 011002 104400                    TYPOCT          ;PRINT RETRY NUMBER
2854 011004 005737 000656          TST      TEMP3     ;SEE IF DID NON-RECOVERABLE
2855 011010 001403                    BEQ     RDRT6       ;IF NOT: BR
2856 011012 005037 000656          CLR     TEMP3     ;CLEAR FLAG
2857 011016 000207                    RTS      PC        ;EXIT
2858 011020 005237 000710          RDRT6:  INC      RTCNT
2859 011024 023737 000710 000612  CMP      RTCNT,RETRY ;SEE IF DONE 8 RETRIES
2860 011032 001272                    BNE     RDRTG       ;IF NOT: BR
2861 011034 000004 026223          TYPE,MSG115
2862 011040 013704 000702          MOV      UNP,R4
2863 011044 005737 000570          TST      RDCMD     ;SEE IF READ REVERSE
2864 011050 001003                    BNE     RDRT7       ;IF SO: BR
2865 011052 005264 002734          INC     RFHARD(R4) ;BUMP FORWARD HARD ERROR CNTR
2866 011056 000402                    BR      RDRTX
2867 011060 005264 002754          RDRT7:  INC     RRHARD(R4) ;BUMP REVERSE HARD ERROR CNTR
2868 011064 000207          RDRTX:  RTS      PC        ;RETURN
2869
2870 011066 013703 000730          NRTF:   MOV      ERSV,R3 ;GET ER REGISTER
2871 011072 104400                    TYPOCT          ;PRINT ER
2872 011074 004737 020364          JSR      PC,FRPRT  ;PRINT F OR R
2873 011100 000207          RTS      PC        ;RETURN
2874
2875          ;*****
2876          ;YOZZLE SUBROUTINE:
2877          ;
2878          ;THIS SUBROUTINE, ENTERED VIA SWITCH FIVE (5), IS USED TO PERFORM
2879          ;A CONTINUOUS READ AND SPACE OVER OF THE CURRENT RECORD ON TAPE.
2880          ;FULL STATUS AND DATA CHECKING MAY BE PERFORMED
2881          ;OR NOT VIA CONSOLE SWITCHES ELEVEN (11) AND THIRTEEN (13).
2882          ;A SOFTWARE DELAY IS PERFORMED BETWEEN EACH READ
2883          ;AND SPACE OPERATION AND MAY BE VARIED BY TYPING
2884          ;CNTRL C ON THE TTY AND ENTERING A VALUE IN RESPONSE
2885          ;TO THE PRINTED REQUEST.
2886          ;*****
2887 011102 013737 000610 000674  YOZ:   MOV      YSTAL,STAL
2888 011110 004737 011766          JSR      PC,STALL  ;DO YOZZLE STALL
2889 011114 012737 177777 167374  YOZO:  MOV      #1,@#C    ;SET TO 1 RECORD SPACING
2890 011122 005737 000570          TST      RDCMD     ;SEE IF READ REVERSE
2891 011126 001404                    BEQ     YOZA        ;IF NOT: BR
2892 011130 112737 000030 000700  MOVB    #30,MTC1   ;SET TO SPACE FORWARD
2893 011136 000403                    BR      YOZB
2894 011140 112737 000032 000700  YOZA:  MOVB    #32,MTC1   ;SET TO SPACE REVERSE
2895 011146 012737 011166 000670  YOZB:  MOV      #YOZC,RTRN ;SET RETURN ADDRESS
2896 011154 012737 177775 000674  MOV      #177775,STAL ;SET TIME MULTIPLIER
2897 011162 000137 020506          JMP      TAPG      ;GO YOZZLE
2898 011166 005737 000704          YOZC:  TST      TMFLG    ;SEE IF TM
2899 011172 001404                    BEQ     1$         ;IF NOT: BR
2900 011174 012737 040000 000674  MOV      #40000,STAL ;SET TM STALL

```

2901	011202	000403				BR	2\$		
2902	011204	013737	000610	000674	1\$:	MOV	YSTAL,STAL		
2903	011212	004737	011766		2\$:	JSR	PC,STALL		;DO YOZZLE STALL
2904	011216	013777	000564	167270		MOV	@#RDATA,@BA		;SET BUS ADDRESS
2905	011224	005737	000570			TST	RDCMD		;SEE IF READ REVERSE
2906	011230	001416				BEQ	YOZC1		;IF NOT: BR
2907	011232	013703	000556			MOV	FMCNT,R3		
2908	011236	005103				COM	R3		
2909	011240	032737	000020	000552		BIT	#20,UDES		;SEE IF CORE DUMP
2910	011246	001401				BEQ	YOZC0		;IF NOT: BR
2911	011250	006203				ASR	R3		;R3 = FC/2
2912	011252	060377	167236		YOZC0:	ADD	R3,@BA		;SET REVERSE BUS ADDRESS
2913	011256	012737	000076	000700		MOV	#76,MTC1		;SET READ REVERSE
2914	011264	000403				BR	YOZC2		
2915	011266	012737	000070	000700	YOZC1:	MOV	#70,MTC1		;SET READ FORWARD
2916	011274	013777	000556	167214	YOZC2:	MOV	FMCNT,@FC		;SET CHARACTER COUNT
2917	011302	012737	011314	000670		MOV	#YOZD,RTRN		;SET RETURN ADDRESS
2918	011310	000137	020506			JMP	TAPG		;GO READ
2919	011314	032777	004000	167274	YOZD:	BIT	#4000,@SWR		;SEE IF SHOULD CHECK ERRORS
2920	011322	001047				BNE	YOZE		;IF NOT: BR
2921	011324	005737	000704			TST	TMFLG		;SEE IF TAPE MARK TIME
2922	011330	001442				BEQ	YOZD1		;IF NOT: BR
2923	011332	005737	000570			TST	RDCMD		;SEE IF READ REVERSE
2924	011336	001425				BEQ	YOZD0		;IF NOT: BR
2925	011340	013703	000564			MOV	@#RDATA,R3		
2926	011344	013704	000556			MOV	FMCNT,R4		
2927	011350	005104				COM	R4		
2928	011352	032737	000020	000552		BIT	#20,UDES		;SEE IF CORE DUMP
2929	011360	001401				BEQ	YOZD4		;IF NOT: BR
2930	011362	006204				ASR	R4		;SET TO FC/2
2931	011364	060403			YOZD4:	ADD	R4,R3		;SET EXPT BUS ADDRESS
2932	011366	042703	000001			BIC	#1,R3		;MAKE EXPT ADDRESS EVEN
2933	011372	032737	002000	000552		BIT	#2000,UDES		;SEE IF PE
2934	011400	001001				BNE	YOZD2		;IF SO: BR
2935	011402	005743				TST	(R3)		;SET EXPT BA
2936	011404	004737	016730		YOZD2:	JSR	PC,ER2		;GO CHECK ERRORS
2937	011410	000430				BR	YOZF		
2938	011412	013703	000564		YOZD0:	MOV	@#RDATA,R3		
2939	011416	032737	002000	000552		BIT	#2000,UDES		;SEE IF PE
2940	011424	001001				BNE	YOZD3		;IF SO: BR
2941	011426	005723				TST	(R3)		;SET EXPT BA
2942	011430	004737	016730		YOZD3:	JSR	PC,ER2		;GO CHECK ERRORS
2943	011434	000416				BR	YOZF		
2944	011436	004737	016636		YOZD1:	JSR	PC,ERCHK		;ELSE GO CHECK ERRORS
2945	011442	005737	000720		YOZE:	TST	RTYFL		;SEE IF RETRY
2946	011446	001013				BNE	YOZG		;IF SO: BR
2947	011450	032777	020000	167140		BIT	#20000,@SWR		;SEE IF SHOULD CHECK DATA
2948	011456	001005				BNE	YOZF		;IF NOT: BR
2949	011460	005737	000704			TST	TMFLG		;SEE IF TAPE MARK
2950	011464	001002				BNE	YOZI		;IF SO: BR
2951	011466	004737	015064			JSR	PC,DCHK		;ELSE GO CHECK DATA
2952	011472	004737	013730		YOZF:	JSR	PC,DS3		;GO CLEAR DATA AREA
2953	011476	032777	000040	167112	YOZG:	BIT	#40,@SWR		;SEE IF SHOULD CONTINUE YOZZLE
2954	011504	001402				BEQ	YOZH		;IF NOT: BR
2955	011506	000137	011114			JMP	YOZO		
2956	011512	000207			YOZH:	RTS	PC		;EXIT



```

2958 ;*****
2959 ;BACKSPACE SUBROUTINE:
2960 ;
2961 ;THIS SUBROUTINE IS USED TO PERFORM THE
2962 ;BACKSPACE OPERATION REQUIRED BY THE READ
2963 ;ROUTINE FOR READ FORWARD AFTER WRITING.
2964 ;IF A TAPE MARK IS EXPECTED (TM=1) THEN THE SPACE
2965 ;ROUTINE ASSUMES THAT THE TM WILL BE FIRST WHEN
2966 ;BACKSPACING. THEREFORE TWO OPERATIONS ARE REQUIRED
2967 ;TO SPACE OVER A BLOCK. FIRST SPACE OVER THE TM, THEN
2968 ;SPACE OVER THE DATA RECORDS.
2969 ;A CHECK FOR RECORD COUNT ZERO IS MADE AT THE
2970 ;END OF THE SPACE OPERATION TO ASSURE THAT PROPER
2971 ;TAPE POSITIONING WAS DONE.
2972 ;*****
2973
2974 011514 013737 000606 000674 BKSP: MOV TSTAL,STAL
2975 011522 004737 011766 JSR PC,STALL ;DO TURN AROUND STALL
2976 011526 012737 024002 000660 MOV #MSG10,EMADDR
2977 011534 013703 000564 MOV @RDATA,R3 ;SET EXPECTED BA
2978 011540 010377 166750 MOV R3,@BA
2979 011544 005737 000572 TST TMEX ;SEE IF TM
2980 011550 001436 BEQ B0 ;IF NOT: BR
2981 011552 012777 177777 166736 MOV #-1,@FC
2982 011560 012737 000032 000700 MOV #32,MTC1
2983 011566 012737 011600 000670 MOV #1$,RTRN
2984 011574 000137 020506 JMP TAPG ;SPACE TO TM
2985 011600 032777 010000 167010 1$: BIT #10000,@SWR ;SEE IF SHOULD CHECK ERROR
2986 011606 001017 RNE B0 ;IF NOT: BR
2987 011610 012737 025221 000660 MOV #MSG55,EMADDR
2988 011616 032777 000004 166676 BIT #4,@DS ;SEE IF TM
2989 011624 001006 BNE 2$ ;IF SO: BR
2990 011626 013737 000564 020340 MOV @RDATA,CADER
2991 011634 004737 017466 JSR PC,ERPT ;PRINT ERROR
2992 011640 000402 BR B0
2993 011642 004737 016730 2$: JSR PC,ER2
2994 011646 013700 000554 B0: MOV RCNT,R0
2995 011652 005737 000666 TST EOTREC ;BRANCH IF EOT NOT DETECTED
2996 011656 100007 BPL 1$
2997 011660 042737 100000 000666 BIC #100000,EOTREC ;CLEAR EOT INDICATOR
2998 011666 013703 000666 MOV EOTREC,R3 ;GET # OF RECORDS LEFT IN BLOCK
2999 011672 160300 SUB R3,R0 ;FORM # OF RECORDS TO BACK SPACE
3000 011674 005200 INC R0
3001 011676 012737 024002 000660 1$: MOV #MSG10,EMADDR ;SET ERROR MESSG ADDRESS
3002 011704 012737 011742 000670 MOV #2$,RTRN ;SET RETURN PC
3003 011712 012777 177777 166576 MOV #-1,@FC ;SET TO BACKSPACE 1 RECORD
3004 011720 013703 000564 MOV @RDATA,R3 ;SET EXPECTED BA
3005 011724 010377 166564 MOV R3,@BA
3006 011730 012737 000032 000700 MOV #32,MTC1 ;SET SPACE REVERSE
3007 011736 000137 020506 JMP TAPG ;GO DO SPACE
3008 011742 004737 016730 2$: JSR PC,ER2
3009 011746 013737 000606 000674 MOV TSTAL,STAL ;DO STALL
3010 011754 004737 011766 JSR PC,STALL ;STALL
3011 011760 005300 DEC R0 ;DECREMENT # OF RECORD TO BACKSPACE
3012 011762 001345 BNE 1$
3013 011764 000207 RTS PC ;EXIT

```

B3

CZTEDFO 1M03 TE16 1077 DRY  
CZTEDE.P11 07 MAR 84 14:04

MACY11 30(1046) 07 MAR 84 14:21 PAGE 42 1

SEQ 0066

3015  
3016  
3017  
3018  
3019  
3020  
3021  
3022  
3023  
3024  
3025  
3026  
3027  
3028  
3029  
3030  
3031  
3032  
3033  
3034  
3035

011766 005337 000674  
011772 001375  
011774 000207

```

;*****
;STALL ROUTINE;
;
;THIS ROUTINE IS USED TO PROVIDE SOFTWARE DELAYS
;DURING READ, WRITE, TURN AROUND, AND YOZZLE.
;THE DELAY TIMES MAY BE SET BY THE OPERATOR AT
;INITIAL START FROM 200(8) OR MAY BE MODIFIED
;AT ANY TIME BY ENTERING CNTRL C ON THE TTY AND
;INSERTING NEW VALUES IN RESPONSE TO THE REQUEST.
;THE READ STALL AND THE WRITE STALL ARE DELAYS
;EXECUTED BETWEEN EACH RECORD OF THE DATA BLOCK.
;THE TURN AROUND STALL IS EXECUTED EACH TIME
;THE DIRECTION OF TAPE MOVEMENT IS CHANGED AND
;ALSO EACH TIME THE TAPE OPERATION CHANGES FROM
;WRITE TO READ OR READ TO WRITE. THE YOZZLE
;STALL IS EXECUTED ONLY DURING THE YOZZLE ROUTINE.
;*****
STALL: DEC STAL
      BNE STALL      ;DELAY
      RTS PC        ;EXIT

```

C6

```

3037
3038
3039
3040
3041
3042
3043
3044
3045
3046
3047
3048
3049 011776 012701 177760          CCNTR: MOV      0-20,R1          ;SET HIGH LIMIT
3050 012002 013702 000560          MOV      BUFMAX,R2        ;SET LOW LIMIT
3051 012006 004737 022430          JSR      PC,RANG          ;GO GENERATE NUMBER
3052 012012 042737 000001 000636  BIC      01,RANSAB        ;
3053 012020 013737 000636 000556  MOV      RANSAB,FMCNT     ;SET CHAR COUNT
3054 012026 012737 177777 013766  MOV      0-1,PATS        ;PRESET DATA PATTERN
3055 012034 000207                RTS      PC               ;EXIT
3056
3057
3058
3059
3060
3061
3062
3063
3064
3065
3066
3067 012036 012702 000001          RCNTR: MOV      01,R2          ;SET LOW LIMIT
3068 012042 012701 000500          MOV      0500,R1         ;SET HIGH LIMIT
3069 012046 004737 022430          JSR      PC,RANG          ;GO GENERATE NUMBER
3070 012052 013737 000636 000554  MOV      RANSAB,RCNT     ;SET RECORD COUNT
3071 012060 000207                RTS      PC               ;EXIT
3072
3073
  
```

3075  
 3076  
 3077  
 3078  
 3079  
 3080  
 3081  
 3082  
 3083  
 3084  
 3085  
 3086  
 3087  
 3088  
 3089  
 3090  
 3091  
 3092  
 3093  
 3094  
 3095  
 3096  
 3097  
 3098  
 3099  
 3100  
 3101  
 3102  
 3103  
 3104  
 3105  
 3106  
 3107  
 3108  
 3109  
 3110  
 3111  
 3112  
 3113  
 3114  
 3115  
 3116  
 3117  
 3118  
 3119  
 3120  
 3121  
 3122  
 3123  
 3124  
 3125  
 3126  
 3127  
 3128  
 3129  
 3130

```

;*****
;TEST CONDITION ENTRY ROUTINE:
;
;THIS ROUTINE IS USED TO ALLOW THE OPERATOR
;TO ENTER, AT THE TTY, THE NECESSARY PARAMETERS
;TO RUN THE PROGRAM AS HE WISHES. THE
;ROUTINE IS ONLY ENTERED UPON INITIAL STARTING
;FROM LOCATION 200(8).
;THE MAIN PURPOSE OF THIS ROUTINE IS TO ESTABLISH
;A TABLE OF DEVICES TO BE TESTED. THIS TABLE
;CONSISTS OF AN ENTRY FOR EACH OF ONE (1) TO
;EIGHT (8) DEVICES. EACH ENTRY CONTAINS THE
;SLAVE NUMBER, DENSITY, PARITY, AND
;FORMAT. THE INFORMATION IS ENTERED
;IN RESPONSE TO PRINTED REQUESTS AT THE TTY.
;SLAVES MAY BE ENTERED IN ANY ORDER. EACH
;PARAMETER IS CHECKED FOR LEGALITY BEFORE BEING
;SET INTO THE TABLE.
;THE DRIVE NUMBER REQUEST WILL ALSO CHECK THE MASGBUS
;FOR THE PRESENCE OF THE REQUESTED DRIVE. IF IT IS NOT FOUND,
;A NON-EXIST DRIVE MESSAGE WILL BE PRINTED AND ANOTHER DRIVE
;REQUEST MADE. WHEN THE DRIVE IS FOUND, THE RESPONSE IS STORED
;AND CONTROL PASSED TO THE SLAVE SELECT ROUTINE.
;THE SLAVE SELECT ROUTINE ALSO CHECKS FOR THE PRESENCE OF THE
;SLAVE. IF IT IS NOT PRESENT, A MESSAGE IS PRINTED AND ANOTHER
;REQUEST IS ISSUED. WHEN THE SELECTED SLAVE IS FOUND TO BE
;PRESENT, A MESSAGE IS PRINTED IF IT IS A 7 CHANNEL DRIVE
;TO ASSIST IN SELECTING DENSITY, PARITY, AND FORMAT.
;UPON COMPLETION OF THE DEVICE TABLE, REQUESTS
;ARE PRINTED FOR ENTRY OF THE NUMBER OF CHARACTERS
;PER RECORD AND THE NUMBER OF RECORDS PER BLOCK. THE
;NEXT REQUEST IS FOR A PATTERN NUMBER TO BE USED
;FOR WRITING AND CHECKING OF READ DATA.
;FOLLOWING THE PATTERN REQUEST IS THE TAPE MARK OPTION.
;RESPONDING TO THE REQUEST (TM=) WITH A ONE (1)
;WILL CAUSE THE PROGRAM TO WRITE A TM AT THE
;END OF EACH DATA BLOCK AND TO EXPECT THE
;TM TO BE DETECTED IN EITHER READ FORWARD AND REVERSE
;OR DURING SPACE OPERATION. A RESPONSE OF ZERO (TM=0)
;DISALLOWS WRITING OF THE TM AND CAUSES THE READ
;AND SPACE ROUTINES TO EXPECT NO TM TO BE PRESENT.
;THE LAST REQUESTS ARE FOR ENTRY OF THE DESIRED
;WRITE, READ, AND TURN AROUND STALLS.
;*****
  
```

012062	005737	000644	TINP:	TST	TINF	;SEE IF SHOULD INPUT FROM TTY
012066	001002			BNE	1\$	;IF SO: BR
012070	000137	013404		JMP	TINP4	;GET SWITCHES
012074	005037	000702	1\$:	CIR	UNP	;CLEAR TABLE POINTER
012100	005037	005042		CIR	REOTC	;CLEAR EOT UNIT COUNTER
012104	012737	024431	012130	MOV	@MSG31,41\$	;GET TITLE MSG
012112	005737	000742		TST	ASEQF	;SEE IF AUTO SEQ
012116	001403			BEQ	4\$	;IF NOT: BR
012120	012737	024357	012130	MOV	@MSG30,41\$	;SET AUTO SEQ HDR
012126	000004		4\$:	TYPE		;TYPE MSG

3131	012130	000000		41\$:	.WORD 0	;ADDRESS OF APPROPRIATE TITLE MSG	
3132	012132	105077	177772		CLRB @41\$	;DO NOT TYPE TITLE ON RESTART	
3133	012136	000004	024513		TYPE,MSG31A	;TYPE INSTRUCTIONS	
3134	012142	105037	024513		CLRB MSG31A	;DO NOT TYPE STARTUP INSTRUCTIONS ON RESTART	
3135	012146	005737	013560		TST SCVFL	;SEE IF SHORT CONVERSATION	
3136	012152	001065			BNE 6\$	;IF SO; BR	
3137	012154	000004	025445		TYPE,MSG74	;REQUEST REGISTER START	
3138	012160	013703	000544		MOV REGS,R3		
3139	012164	104400			TYPOCT	;PRINT CURRENT REG START	
3140	012166	012705	000544		MOV @REGS,R5	;SAVE ADDRESS LOCATION	
3141	012172	012701	000007		MOV @7,R1	;SET SIZE OF ENTRY	
3142	012176	012702	176400		MOV @176400,R2	;SET UPPER LIMIT	
3143	012202	012703	172300		MOV @172300,R3	;SET LOWER LIMIT	
3144	012206	004737	022612		JSR PC,TTR	;GO GET RESPONSE	
3145							
3146	012212	000004	025470		TYPE,MSG75	;REQUEST INTERRUPT VECTOR ADDRESS	
3147	012216	013703	000546		MOV VECT,R3		
3148	012222	104400			TYPOCT	;PRINT CURRENT VECTOR	
3149	012224	012705	000546		MOV @VECT,R5	;SET SAVE LOCAT ON	
3150	012230	012701	000004		MOV @4,R1	;SET SIZE OF ENTRY	
3151	012234	012702	000224		MOV @224,R2	;SET UPPER LIMIT	
3152	012240	012703	000150		MOV @150,R3	;SET LOWER LIMIT	
3153	012244	004737	022612		JSR PC,TTR	;GO GET RESPONSE	
3154	012250	013700	000546		MOV VECT,R0	;GET VECTOR ADDRESS	
3155	012254	012720	021274		MOV @MTINT,(R0)	;LOAD VECTOR WITH HANDLER ADDRESS	
3156	012260	012710	000342		MOV @340,(R0)	;LOAD PRIORITY LEVEL	
3157	012264	013700	000544		MOV REGS,R0	;GET STARTING REGISTER ADDRESS	
3158	012270	012701	000016		MOV @16,R1	;SET NUMBER OF REGISTERS	
3159	012274	012702	000510		MOV @C1,R2	;GET FIRST ADDRESS LOCATION	
3160	012300	010022		5\$:	MOV R0,(R2)	;BUILD TABLE OF ADDRESSES	
3161	012302	062700	000002		ADD @2,R0	;BUMP ADDRESS	
3162	012306	005301			DEC R1	;SEE IF DONE	
3163	012310	001373			BNE 5\$	;IF NOT; BR	
3164	012312	005737	000742		TST ASEQF	;SEE IF AUTO SEQ	
3165	012316	001403			BEQ 6\$	;IF NOT; BR	
3166	012320	005726			TST (SP)	;RESET STACK POINTER	
3167	012322	000137	021312		JMP ASEQ	;GO TO AUTO SEQUENCE	
3168							
3169	012326	012777	000040	166164	6\$:	MOV @40,@CS	;INITIALIZE
3170	012334	000004	025156		TYPE,MSG52A	;REQUEST DRIVE (TMO3) @	
3171	012340	012705	000550		MOV @DVN,R5	;GET ADDRESS	
3172	012344	012701	000002		MOV @2,R1	;SET SIZE OF RESPONSE	
3173	012350	012702	000007		MOV @7,R2	;SET UPPER LIMIT	
3174	012354	012703	000000		MOV @0,R3	;SET LOWER LIMIT	
3175	012360	004737	022612		JSR PC,TTR	;GO GET DRIVE NUMBER	
3176	012364	013777	000550	166126	MOV DVN,@CS		
3177	012372	005777	166112		TST @C1	;ACCESS DRIVE	
3178	012376	032777	010000	166114	BIT @10000,@CS	;SEE IF NED	
3179	012404	001403			BEQ TINPO	;IF NOT; BR	
3180	012406	000004	025402		TYPE,MSG71	;TYPE 'NON-EXISTANT DRIVE'	
3181	012412	000745			BR 6\$	;RETRY DVN	
3182							
3183	012414	012705	000654		TINPO: MOV @TEMP2,R5	;SET ADDRESS FOR RESPONSE	
3184	012420	000004	024600		TYPE,MSG32	;REQUEST SLAVE (TE16,TU??) @	
3185	012424	005037	000654		CLR TEMP2	;CLEAR BUFFER	
3186	012430	012701	000002		MOV @2,R1	;SET NUMBER OF CHARACTERS TO INPUT	

```

3187 012434 012702 000007      MOV      #7,R2      ;SET MAXIMUM LIMIT
3188 012440 012703 000000      MOV      #0,R3      ;SET MINIMUM LIMIT
3189 012444 004737 022612      JSR      PC,TTR     ;GO GET UNIT NUMBER
3190 012450 005737 000652      TST      TEMP1     ;SEE IF HAVE NEW PARAMETER
3191 012454 001010              BNE      TINPOB    ;IF SO: BR
3192 012456 013700 000702      MOV      UNP,R0
3193 012462 001754              BEQ      TINPO     ;BRANCH IF FIRST ENTRY
3194 012464 012760 177777 000752  MOV      #-1,UN1(R0) ;SET END UNIT TABLE
3195 012472 000137 013012      JMP      TINP2C    ;GO GET RECORD COUNT
3196 012476 013700 000702      TINPOB: MOV      UNP,R0
3197 012502 011560 000752      MOV      (R5),UN1(R0) ;SET NEW SLAVE #
3198 012506 012777 000040 166004  MOV      #40,@CS   ;DO A MASS BUS CLEAR
3199 012514 013777 000550 165776  MOV      @VN,@CS   ;LOAD DRIVE #
3200 012522 016077 000752 166012  MOV      UN1(R0),@TC ;LOAD SLAVE NUMBER
3201 012530 032777 002000 166000  BIT      #2000,@DT ;SEE IF SLAVE PRESENT
3202 012536 001003              BNE      TINPOD    ;IF SO: BR
3203 012540 000004 025234      TYPE,MSG57        ;TYPE NON-EXISTANT SLAVE
3204 012544 000723              BR       TINPO     ;REDO
3205 012546 017703 165764      TINPOD: MOV      @DT,R3 ;GET CONTENTS OF DT REG
3206 012552 042703 000007      BIC      #7,R3     ;CLEAR DRIVE TYPE #
3207 012556 022703 142050      CMP      #142050,R3 ;SEE IF 9TRK TMO3
3208 012562 001407              BEQ      TINPOE    ;IF SO: BR
3209 012564 000004 025127      TYPE,MSG50        ;TYPE 'ILLEGAL DRIVE TYPE'
3210 012570 017703 165742      MOV      @DT,R3
3211 012574 042703 000007      BIC      #7,R3     ;CLEAR SLAVE #
3212 012600 104400              TYPOCT
3213 012602 004737 023562      TINPOE: JSR      PC,SNPT ;PRINT DRIVE TYPE REGISTER
3214
3215 012606 000004 024613      TINP1:  TYPE,MSG33    ;REQUEST DENSITY
3216 012612 005037 000654      CLR      TEMP2     ;CLEAR BUFFER
3217 012616 012701 000002      MOV      #2,R1     ;SET NUMBER OF CHARACTERS TO INPUT
3218 012622 012702 000004      MOV      #4,R2     ;SET MAXIMUM LIMIT
3219 012626 012703 000003      MOV      #3,R3     ;SET MINIMUM LIMIT
3220 012632 004737 022612      JSR      PC,TTR     ;GO GET DENSITY
3221 012636 012703 000010      MOV      #10,R3    ;SET POSITION FACTOR
3222 012642 004737 013562      JSR      PC,TPOS    ;GO LOAD DENSITY INTO PROPER POSITION
3223
3224 012646 000315              TINP2:  SWAB      (R5) ;IF DENSITY
3225 012650 022715 000004      CMP      #4,(R5)  ;IS 1600BPI
3226 012654 001415              BEQ      1$       ;THEN SKIP PARITY REQUEST
3227 012656 000004 024626      TYPE,MSG34        ;REQUEST PARITY
3228 012662 005037 000654      CLR      TEMP2     ;CLR BFR
3229 012666 012701 000002      MOV      #2,R1     ;SET NUMBER OF CHAR. TO INPUT
3230 012672 012702 000001      MOV      #1,R2     ;SET HIGH LIMIT
3231 012676 012703 000000      MOV      #0,R3     ;SET LOW LIMIT
3232 012702 004737 022612      JSR      PC,TTR     ;GO INPUT PARITY
3233 012706 000402              BR       1$       ;SKIP 1600 BPI PARITY SETTING
3234 012710 012715 000000      1$:  MOV      #0,(R5) ;SET ODD PARITY FOR 1600 BPI
3235 012714 012703 000003      2$:  MOV      #3,R3   ;SET POSITION FACTOR
3236 012720 004737 013562      JSR      PC,TPOS    ;GO POSITION PARITY
3237
3238 012724 000004 025200      TINP2A: TYPE,MSG53    ;REQUEST FORMAT
3239 012730 005037 000654      CLR      TEMP2
3240 012734 012701 000003      MOV      #3,R1
3241 012740 012702 000017      MOV      #17,R2
3242 012744 012703 000000      MOV      #0,R3

```

3243	012750	004737	022612		JSR	PC,TTR		;GO GET FORMAT
3244	012754	012703	000004		MOV	#4,R3		
3245	012760	004737	013562		JSR	PC,TPOS		
3246	012764	005237	005042		TINP2B: INC	REOTC		;BUMP EOT UNIT COUNTER
3247	012770	022737	000016	000702	CMP	#16,UNP		;SEE IF DONE UNITS
3248	012776	001405			BEQ	TINP2C		;IF SO: BR
3249	013000	062737	000002	000702	ADD	#2,UNP		;POINT TO NEXT UNIT
3250	013006	000137	012414		JMP	TINP0		;ELSE LOOK FOR NEXT UNIT
3251								
3252								
3253	013012	005037	000702		TINP2C: CLR	UNP		;CLEAR UNIT POINTER
3254	013016	113737	005042	005043	MOVB	REOTC,REOTC+1		;SET # OF UNITS TO TEST
3255								
3256	013024	000004	024640		TINP3: TYPE,MSG35			;REQUEST RECORDS PER BLOCK
3257	013030	013703	000554		MOV	RCNT,R3		
3258	013034	104400			TYPOCT			;PRINT RECORD COUNT
3259	013036	012705	000554		MOV	#RCNT,R5		;SET RECORD COUNT ADDRESS
3260	013042	012701	000007		MOV	#7,R1		;SET NUMBER OF CHARACTERS TO INPUT
3261	013046	012702	177777		MOV	#177777,R2		;SET MAXIMUM LIMIT
3262	013052	012703	000001		MOV	#1,R3		;SET MINIMUM LIMIT
3263	013056	004737	022612		JSR	PC,TTR		;GO GET RECORD COUNT
3264	013062	013737	000554	000640	MOV	RCNT,RCSAV		;SAVE RECORD COUNT
3265								
3266	013070	000004	024660		TYPE,MSG36			;REQUEST CHARACTERS PER RECORD
3267	013074	005437	000556		NEG	FMCNT		
3268	013100	013703	000556		MOV	FMCNT,R3		
3269	013104	104400			TYPOCT			;PRINT CHAR COUNT
3270	013106	012705	000556		MOV	#FMCNT,R5		;SET CHARACTER COUNT ADDRESS
3271	013112	012701	000007		MOV	#7,R1		;SET NUMBER OF CHARACTERS TO INPUT
3272	013116	013702	000560		MOV	BUFMAX,R2		;SET MAXIMUM LIMIT
3273	013122	005402			NEG	R2		;MAKE IT POSITIVE
3274	013124	012703	000004		MOV	#4,R3		;SET MINIMUM LIMIT
3275	013130	004737	022612		JSR	PC,TTR		;GO GET CHARACTER COUNT
3276	013134	005437	000556		NEG	FMCNT		;SET TO TWO'S COMPLIMENT
3277	013140	013737	000556	000642	MOV	FMCNT,FCSAV		;SAVE FRAME COUNT
3278								
3279	013146	000004	024676		TYPE,MSG37			;REQUEST PATTERN #
3280	013152	013703	000566		MOV	PATRN,R3		
3281	013156	104400			TYPOCT			;PRINT PATTERN
3282	013160	005037	014130		CLR	DOFL		;CLEAR EXTERNAL DATA FLAG
3283	013164	012705	000566		MOV	#PATRN,R5		;SET PATTERN NUMBER ADDRESS
3284	013170	012701	000003		MOV	#3,R1		;SET NUMBER OF CHARACTERS TO INPUT
3285	013174	012702	000015		MOV	#15,R2		;SET MAXIMUM LIMIT
3286	013200	012703	000000		MOV	#0,R3		;SET MINIMUM LIMIT
3287	013204	004737	022612		JSR	PC,TTR		;GO GET PATTERN NUMBER
3288								
3289	013210	000004	025366		TYPE,MSG39			;REQUEST TAPE MARK
3290	013214	013703	000572		MOV	TMFX,R3		
3291	013220	104400			TYPOCT			;PRINT CURRENT TM FLAG SETTING
3292	013222	012705	000572		MOV	#TMFX,R5		;GET TM FLAG ADDRESS
3293	013226	012701	000002		MOV	#2,R1		;SET SIZE OF RESPONSE
3294	013232	012702	000001		MOV	#1,R2		;SET UPPER LIMIT
3295	013236	012703	000000		MOV	#0,R3		;SET LOWER LIMIT
3296	013242	004737	022612		JSR	PC,TTR		;TM 1-YES
3297								
3298	013246	000004	024070		TYPE,MSG21			;REQUEST INTERCHANGE READ

3299	013252	013703	000576	MOV	INTRF,R3	
3300	013256	104400		TYPOCT		;PRINT CURRENT SETTING
3301	013260	012705	000576	MOV	#INTRF,R5	;GET FLAG ADDRESS
3302	013264	012701	000002	MOV	#2,R1	;SET SIZE OF RESPONSE
3303	013270	012702	000001	MOV	#1,R2	;SET UPPER LIMIT
3304	013274	012703	000000	MOV	#0,R3	;SET LOWER LIMIT
3305	013300	004737	022612	JSR	PC,TTR	;GO GET RESPONSE
3306						
3307	013304	000004	024713	TYPE,MSG38		;REQUEST SINGLE PASS
3308	013310	013703	000600	MOV	SPFLG,R3	
3309	013314	104400		TYPOCT		;PRINT CURRENT SETTING
3310	013316	012705	000600	MOV	#SPFLG,R5	;SET ADDRESS OF FLAG
3311	013322	012701	000002	MOV	#2,R1	;SET SIZE OF RESPONSE
3312	013326	012702	000001	MOV	#1,R2	;SET UPPER LIMIT
3313	013332	012703	000000	MOV	#0,R3	;SET LOWER LIMIT
3314	013336	004737	022612	JSR	PC,TTR	;GO GET RESPONSE
3315						
3316	013342	000004	024731	TINP3A: TYPE,MSG39		;REQUEST CRC CORRECTION
3317	013346	013703	000574	MOV	CRCC,R3	
3318	013352	104400		TYPOCT		
3319	013354	012705	000574	MOV	#CRCC,R5	
3320	013360	012701	000002	MOV	#2,R1	
3321	013364	012702	000001	MOV	#1,R2	
3322	013370	012703	000000	MOV	#0,R3	
3323	013374	004737	022612	JSR	PC,TTR	
3324	013400	004737	022462	JSR	PC,GTSWR	;GET SWITCHES
3325	013404	005737	013560	TINP4: TST	SCVFL	;BRANCH IF SHCRT CONVERSATION
3326	013410	001060		BNE	TINPX	
3327	013412	005737	000644	1\$: TST	TINF	;BRANCH IF NO TTY INPUT
3328	013416	001455		BEQ	TINPX	
3329	013420	000004	024767	TYPE,MSG40		;REQUEST READ STALL
3330	013424	013703	000602	MOV	RSTAL,R3	
3331	013430	104400		TYPOCT		;PRINT READ STALL
3332	013432	012705	000602	MOV	#RSTAL,R5	;SET READ STALL ADDRESS
3333	013436	012701	000007	MOV	#7,R1	;SET NUMBER OF CHARACTERS TO INPUT
3334	013442	012702	177777	MOV	#1,R2	;SET MAXIMUM LIMIT
3335	013446	012703	000001	MOV	#1,R3	;SET MINIMUM LIMIT
3336	013452	004737	022612	JSR	PC,TTR	;GO GET READ STALL
3337						
3338	013456	000004	025016	TYPE,MSG41		;REQUEST WRITE STALL
3339	013462	013703	000604	MOV	WSTAL,R3	
3340	013466	104400		TYPOCT		;PRINT READ STALL
3341	013470	012705	000604	MOV	#WSTAL,R5	;SET WRITE STALL ADDRESS
3342	013474	012701	000007	MOV	#7,R1	;SET NUMBER OF CHARACTERS TO INPUT
3343	013500	012702	177777	MOV	#1,R2	;SET MAXIMUM LIMIT
3344	013504	012703	000001	MOV	#1,R3	;SET MINIMUM LIMIT
3345	013510	004737	022612	JSR	PC,TTR	;GO GET WRITE STALL
3346						
3347	013514	000004	025027	TYPE,MSG42		;REQUEST TURN AROUND STALL
3348	013520	013703	000606	MOV	TSTAL,R3	
3349	013524	104400		TYPOCT		;PRINT TA STALL
3350	013526	012705	000606	MOV	#TSTAL,R5	;SET TURN AROUND STALL ADDRESS
3351	013532	012701	000007	MOV	#7,R1	;SET NUMBER OF CHARACTERS TO INPUT
3352	013536	012702	177777	MOV	#1,R2	;SET MAXIMUM LIMIT
3353	013542	012703	000001	MOV	#1,R3	;SET MINIMUM LIMIT
3354	013546	004737	022612	JSR	PC,TTR	;GO GET TURN AROUND STALL





3369  
3370  
3371  
3372  
3373  
3374  
3375  
3376  
3377  
3378  
3379  
3380  
3381  
3382  
3383  
3384  
3385  
3386  
3387 013606 005737 014520 DSUP: TST RDFL ;SEE IF DID RANDOM DATA  
3388 013612 001044 BNE DS2A ;IF NOT: BR  
3389 013614 005737 000742 DSO: TST ASEQF ;SEE IF AUTO SEQ  
3390 013620 001406 BEQ DSOC ;IF NOT: BR  
3391 013622 005737 000566 TST PATRN ;SEE IF AUTO RANDOM  
3392 013626 100003 BPL DSOC ;IF NOT: BR  
3393 013630 004737 014456 JSR PC,DATR ;ELSE GO GENERATE RANDOM DATA  
3394 ; RTS PC ;++B DELETED  
3395 013634 000433 BR DS2A ;++B GENERATE EXPECTED LRC/CRC & CLEAR READ BFR  
3396 013636 023737 000566 013766 DSOC: CMP PATRN,PATS ;SEE IF NEW PATTERN  
3397 013644 001014 BNE DS0A ;IF SO: BR  
3398 013646 013703 000552 MOV UDES,R3 ;GET UNIT DESCRIPTION  
3399 013652 042703 177767 BIC #177767,R3 ;MASK EVEN PARITY  
3400 013656 023703 013770 CMP PARS,R3 ;SEE IF SAME AS LAST TIME  
3401 013662 001404 BEQ DS0B ;IF SO: BR  
3402 013664 010337 013770 MOV R3,PARS ;SAVE PARITY  
3403 013670 004737 014522 JSR PC,CRC/LRC ;GO GENERATE EXPT CRC/LRC  
3404 013674 000207 DS0B: RTS PC  
3405 013676 013703 000562 DS0A: MOV @#WDATA,R3 ;R3 = ADDRS OF WRITE BUFFER  
3406 013702 013701 000566 MOV PATRN,R1 ;R1 = PATTERN SELECTOR  
3407 013706 010137 013766 MOV R1,PATS  
3408 013712 062701 000001 ADD #1,R1 ;BUMP POINTER  
3409 013716 006301 ASL R1 ;MAKE PATTERN SELECTOR EVEN  
3410 013720 004771 002771 JSR PC,@DATB(R1) ;GO GENERATE PATTERN  
3411 013724 004737 014522 DS2A: JSR PC,CRC/LRC ;GO GENERATE EXPT CRC/LRC  
3412 013730 013702 000556 DS3: MOV #MCNT,R2 ;R2=BUFFER SIZE  
3413 013734 006202 ASR R2 ;R2=FRAME CNT/2  
3414 013736 013701 000564 MOV @#RDATA,R1 ;R1=READ DATA START  
3415 013742 005021 DS4: CLR (R1) ;CLEAR BUFFER  
3416 013744 005202 INC R2 ;SEE IF DONE ALL  
3417 013746 001375 BNE DS4 ;IF NOT: BR  
3418 013750 013737 000552 013770 MOV UDES,PARS ;GET UNIT DESCRIPTION  
3419 013756 042737 177767 013770 BIC #177767,PARS ;MASK PARITY  
3420 013764 000207 RTS PC ;EXIT  
3421 013766 177777 PATS: 1 ;PATTERN NUMBER SAVE  
3422 013770 000000 PARS: 0  
3423  
3424

```

3426
3427
3428
3429 013772 005737 014130      DATO:  TST    DOFL      ;EXTERNAL DATA INPUT FROM H/S READER (256 CHARACTER MAXIMUM)
3430 013776 001401              BEQ     1$          ;BRANCH IF SHOULD DO EXTERNAL INPUT
3431 014000 000207              RTS     PC          ;++B RETURN
3432 014002 012737 000001 014130 1$:  MOV     @1,DOFL    ;SET EXTERNAL FLAG
3433 014010 005077 164614      CLR     @PRS       ;CLEAR READER STATUS
3434 014014 005037 000652      CLR     TEMP1     ;CLEAR FOR USE AS CHARACTER FLAG
3435 014020 052777 000001 164602 DATOA:  BIS     @1,@PRS   ;START READER
3436 014026 105777 164576      DATOB:  TSTB    @PRS   ;SEE IF DONE
3437 014032 100375              BPL     DATOB     ;IF NOT : BR
3438 014034 005001              CLR     R1        ;CLEAR SAVE LOCATION
3439 014036 117701 164570      MOVB   @PRB,R1   ;SAVE CHARACTER
3440 014042 005737 000652      TST    TEMP1     ;SEE IF HAVE FOUND START CHARACTER
3441 014046 001011              BNE    DATOC     ;IF SO : BR
3442 014050 105701              TSTB   R1        ;SEE IF CHARACTER IS 0
3443 014052 001762              BEQ    DATOA     ;IF SO : BR
3444 014054 012737 000001 000652  MOV     @1,TEMP1  ;ELSE SET CHARACTER FOUND FLAG
3445 014062 010137 000654      MOV     R1,TEMP2  ;SAVE DATA SIZE
3446 014066 010102              MOV     R1,R2     ;SAVE DATA SIZE
3447 014070 000753              BR     DATOA     ;GO GET FIRST DATA CHAR
3448 014072 110123              DATOC:  MOVB   R1,(R3)+ ;LOAD BUFFER
3449 014074 005302              DEC     R2        ;SEE IF READ ALL
3450 014076 001350              BNE    DATOA     ;IF NOT : BR
3451 014100 013701 000562      DATOD:  MOV     @#WDATA,R1 ;R1 = START OF WRITE BUFFER
3452 014104 013702 000654      MOV     TEMP2,R2  ;R2 = SIZE OF DATA FIELD
3453 014110 112123              DATOE:  MOVB   (R1)+,(R3)+ ;REPEAT LOAD OF DATA FIELD
3454 014112 023703 000564      CMP     @#RDATA,R3 ;SEE IF DONE
3455 014116 003001              BGT    DATOF     ;IF NOT: BR
3456 014120 000207              RTS     PC        ;++B RETURN
3457 014122 005302              DATOF:  DEC     R2  ;SEE IF AT END OF DATA FIELD
3458 014124 001371              BNE    DATOE     ;IF NOT : BR
3459 014126 000764              BR     DATOD     ;ELSE RESTART FIELD
3460 014130 000000              DOFL:  0          ;EXTERNAL DATA FLAG=1 IF ALREADY DONE
3461

```

16

```
3463 ;ALL ONES*****
3464
3465 014132 012701 177777 DAT1: MOV #1,R1 ;R1=DATA
3466 014136 012702 002002 DAT1A: MOV #2002,R2 ;R2=WORD COUNT *2
3467 014142 010123 1$: MOV R1,(R3)+ ;LOAD BUFFER
3468 014144 005302 DEC R2 ;SEE IF DONE
3469 014146 001375 BNE 1$ ;IF NOT: BR
3470 014150 000207 RTS PC
3471
3472 ;ALL ZEROS*****
3473
3474 014152 005001 DAT2: CLR R1 ;R1=DATA
3475 014154 000770 BR DAT1A ;LOAD BUFFER
3476
3477 ;WALKING ONE*****
3478
3479 014156 012701 000001 DAT3: MOV #1,R1 ;R1=DATA
3480 014162 000241 CLC
3481 014164 012702 004004 DAT3A: MOV #4004,R2 ;R2=CHARACTER COUNT*4
3482 014170 110123 1$: MOVB R1,(R3)+ ;LOAD BUFFER
3483 014172 106101 ROLB R1 ;SET NEXT CHARACTER
3484 014174 005302 DEC R2 ;SEE IF DONE
3485 014176 001374 BNE 1$ ;IF NOT: BR
3486 014200 000207 RTS PC
3487
3488 ;WALKING ZERO*****
3489
3490 014202 012701 000376 DAT4: MOV #376,R1 ;R1=START OF DATA
3491 014206 000261 SEC
3492 014210 000765 BR DAT3A ;LOAD BUFFER
3493
3494 ;ALTERNATING ONE/ZERO*****
3495
3496
3497 014212 012701 052525 DAT5: MOV #52525,R1 ;R1=DATA
3498 014216 000747 BR DAT1A ;LOAD BUFFER
3499
3500 ;ALTERNATING ZERO/ONE*****
3501
3502 014220 012701 125252 DAT6: MOV #125252,R1 ;R1=DATA
3503 014224 000744 BR DAT1A ;LOAD BUFFER
3504
3505 ;ONE/ZERO IN ALTERNATING WORDS*****
3506
3507 014226 012701 125252 DAT7: MOV #125252,R1 ;SET WORD 1
3508 014232 012702 052525 MOV #52525,R2 ;SET WORD 2
3509 014236 012704 001002 MOV #1002,R4 ;SET NUMBER OF ENTRIES
3510 014242 010123 1$: MOV R1,(R3)+ ;LOAD WORD 1
3511 014244 010223 MOV R2,(R3)+ ;LOAD WORD 2
3512 014246 005304 DEC R4 ;SEE IF DONE
3513 014250 001374 BNE 1$ ;IF NOT: BR
3514 014252 000207 RTS PC
3515
```

```

3517                                     ;WALKING ONE/ALL ONE IN ALTERNATING CHARS****
3518
3519 014254 012702 002002      DAT10: MOV    #2002,R2      ;SET BUFFER SIZE
3520 014260 012701 000001      MOV    #1,R1        ;SET WALK BASE
3521 014264 000241              CLC
3522 014266 012713 177400      1$:  MOV    #177400,(R3) ;LOAD ALL ONE BYTE
3523 014272 050123              BIS    R1,(R3)+    ;LOAD WALK BYTE
3524 014274 106101              ROLB   R1          ;WALK ONE
3525 014276 005302              DEC    R2
3526 014300 001372              BNE    1$         ;DO FULL BUFFER
3527 014302 000207              RTS    PC
3528
3529                                     ;ALL BITS 0-377*****
3530
3531 014304 005001      DAT11: CLR    R1          ;R1=STARTING DATA
3532 014306 012702 004004      MOV    #4004,R2    ;R2=CHARACTER COUNT+4
3533 014312 110123      1$:  MOVB  R1,(R3)+    ;LOAD BUFFER
3534 014314 105201              INCB  R1          ;BUMP DATA
3535 014316 005302              DEC   R2          ;SEE IF DONE
3536 014320 001374              BNE   1$         ;IF NOT: BR
3537 014322 000207              RTS   PC         ;RETURN
3538
3539                                     ;ALL BITS 377-0*****
3540
3541 014324 012701 000377      DAT12: MOV    #377,R1 ;R1=STARTING DATA
3542 014330 012702 004004      MOV    #4004,R2    ;R2=CHARACTER COUNT+4
3543 014334 110123      1$:  MOVB  R1,(R3)+    ;LOAD BUFFER
3544 014336 105301              DECB  R1          ;BUMP DATA
3545 014340 005302              DEC   R2          ;SEE IF DONE
3546 014342 001374              BNE   1$         ;IF NOT: BR
3547 014344 000207              RTS   PC         ;RETURN
3548
3549                                     ;ALTERNATING CHARACTERS 0 AND 377*****
3550
3551 014346 012701 000377      DAT13: MOV    #377,R1 ;R1 DATA
3552 014352 000137 014136      JMP    DAT1A      ;LOAD BUFFER
3553
3554                                     ;WALKING ZERO/ALL ZERO IN ALTERNATING CHARS*****
3555
3556 014356 012702 002002      DAT14: MOV    #2002,R2 ;SET BUFFER SIZE
3557 014362 012701 000376      MOV    #376,R1     ;SET WALK BASE
3558 014366 000261              SEC
3559 014370 010113      1$:  MOV    R1,(R3)    ;LOAD WALK BYTE
3560 014372 042723 177400      BIC   #177400,(R3)+ ;CLEAR HIGH BYTE
3561 014376 106101              ROLB  R1          ;WALK ZERO BIT
3562 014400 005302              DEC   R1
3563 014402 001372              BNE   1$         ;FILL BUFFER
3564 014404 000207              RTS   PC         ;RETURN
3565

```

```

3570                                     ;AUTO SEQUENCE PATTERN*****
3571
3572 014406 012702 000200      DAT15: MOV    #200,R2          ;SET NUMBER OF ENTRIES
3573 014412 012701 014436      1$:   MOV    #APATS,R1        ;SET START OF PATTERN
3574 014416 012704 000010            MOV    #10,R4           ;SET SIZE OF PATTERN
3575 014422 012123              2$:   MOV    (R1)+,(R3)+      ;FILL BUFFER
3576 014424 005304                    DEC    R4                  ;SEE IF DONE PATTERN
3577 014426 001375                    BNE    2$                  ;IF NOT; BR
3578 014430 005302                    DEC    R2                  ;SEE IF DONE BUFER
3579 014432 001367                    BNE    1$                  ;IF NOT; BR
3580 014434 000207                    RTS    PC                  ;RETURN
3581
3582 014436 000000      APATS:  0
3583 014440 177400               177400
3584 014442 000377               377
3585 014444 000000               0
3586 014446 177777               -1
3587 014450 000377               377
3588 014452 177400               177400
3589 014454 177777               -1
3590
3591                                     ;RANDOM DATA GENERATOR SUBROUTINE*****
3592
3593 014456 013704 000556      DATR:  MOV    FMCNT,R4          ;SET NUMBER OF FRAMES
3594 014462 013703 000562            MOV    @#WDATA,R3        ;SET ADDRESS OF START OF BUFFER
3595 014466 012701 177777            MOV    #-1,R1           ;SET HIGH LIMIT
3596 014472 005002                    CLR    R2                  ;SET LOW LIMIT
3597 014474 004737 022430      1$:   JSR    PC,RANG           ;GO GENERATE NUMBER
3598 014500 013723 000636            MOV    RANSV,(R3)+      ;LOAD BUFFER
3599 014504 005204                    INC    R4                  ;SEE IF DONE WHOLE BUFFER
3600 014506 001372                    BNE    1$                  ;IF NOT; BR
3601 014510 012737 000001 014520      MOV    #1,RDFL           ;SET RANDOM DATA FLAG
3602 014516 000207                    RTS    PC                  ;EXIT
3603 014520 000000      RDFL:  0                  ;RANDOM DATA SELECT FLAG

```

```

3605
3606
3607
3608
3609
3610
3611
3612
3613
3614 014522 013700 000556 CRCLRC: MOV FMCNT,R0 ;SET RECORD SIZE
3615 014526 005400 NEG R0
3616 014530 013701 000562 MOV @0WDATA,R1 ;SET START OF BUFFER
3617 014534 005037 015056 CLR XORS
3618 014540 111104 CL0: MOVSB (R1),R4 ;GET CHARACTER
3619 014542 004737 014730 JSR PC,CLP ;GO GET PARITY OF CHARACTER
3620 014546 004737 015032 JSR PC,XOR ;XOR CHARACTER
3621 014552 000241 CLC
3622 014554 006004 ROR R4 ;ROTATE 1 RIGHT
3623 014556 103014 BCC CL2 ;IF NO CARRY: BR
3624 014560 052704 000400 BIS #400,R4 ;SET BIT NINE
3625 014564 000241 CLC
3626 014566 010405 CL1: MOV R4,R5 ;SAVE CHARACTER
3627 014570 042705 177703 BIC #177703,R5
3628 014574 005105 COM R5
3629 014576 042705 177703 BIC #177703,R5
3630 014602 042704 000074 BIC #74,R4
3631 014606 050504 BIS R5,R4 ;COMPLIMENT BITS 2,3,4,5
3632 014610 010437 015056 CL2: MOV R4,XORS
3633 014614 005300 DEC R0
3634 014616 001350 BNE CL0 ;BRANCH IF NOT LAST CHAR
3635 014620 013704 015056 CLLAST: MOV XORS,R4
3636 014624 005137 015056 COM XORS
3637 014630 042737 177050 015056 BIC #177050,XORS
3638 014636 042704 177727 BIC #177727,R4 ;COMPLIMENT ALL BUT BITS 3&5
3639 014642 050437 015056 BIS R4,XORS
3640 014646 013737 015056 015060 MOV XORS,EXCRC ;SAVE EXPECTED CRC
3641 014654 013700 000556 MOV FMCNT,R0
3642 014660 005400 NEG R0
3643 014662 013701 000562 MOV @0WDATA,R1 ;DO EXPT LRC
3644 014666 005037 015056 CLR XORS
3645 014672 111104 CL3: MOVSB (R1),R4
3646 014674 004737 014730 JSR PC,CLP ;GET PARITY
3647 014700 004737 015032 JSR PC,XOR ;XOR CHARACTER
3648 014704 005300 DEC R0
3649 014706 001371 BNE CL3 ;DO ALL FOR LRC
3650 014710 013704 015060 MOV EXCRC,R4
3651 014714 004737 015032 JSR PC,XOR ;XOR CRC TO DATA
3652 014720 013737 015056 015062 MOV XORS,EXLRC ;SAVE EXPT LRC
3653 014726 000207 RTS PC ;RETURN
3654 014730 005704 CLP: TST R4 ;SEE IF 0 CHAR
3655 014732 001010 BNE CLPE ;IF NOT: BR
3656 014734 032737 000010 000552 BIT #10,XORS ;SEE IF EVEN PARITY
3657 014742 001404 BEQ CLPE ;IF NOT: BR
3658 014744 012704 000420 MOV #400,R4 ;SET 0 CHAR EVEN PARITY
3659 014750 005201 INC R1 ;HUMP POINTER
3660 014752 000207 RTS PC ;RETURN
  
```





```

3691
3692
3693
3694
3695
3696
3697
3698
3699
3700
3701
3702
3703
3704
3705
3706 015064 005037 000664          DCHK: CLR      BBC          ;CLEAR BAD RECORD CNTR
3707 015070 005037 000712          CLR      DERFL        ;CLEAR DATA ERROR FLAG
3708 015074 013705 000556          MOV      FMCNT,R5     ;LOAD CHAR COUNT
3709 015100 032737 000020 000552  BIT      *20,UDES     ;SEE IF CORE DUMP
3710 015106 001401                    BEQ      DCHK0        ;IF NOT: BR
3711 015110 006205                    ASR      R5           ;R5 = FC/2
3712 015112 013701 000562          DCHK0: MOV      @WDATA,R1 ;SET WRITE DATA ADDR
3713 015116 013702 000564          MOV      @RDATA,R2   ;SET READ DATA ADDR
3714 015122 032737 000010 000552  BIT      *10,UDES     ;SEE IF EVEN PARITY
3715 015130 001430                    BEQ      DFOC0        ;IF NOT: BR
3716 015132 032737 000020 000552  BIT      *20,UDES     ;SEE IF CORE DUMP PARITY
3717 015140 001024                    BNE      DFOC0        ;IF SO: BR
3718 015142 032737 002000 000552  BIT      *2000,UDES   ;SEE IF PE MODE
3719 015150 001020                    BNE      DFOC0        ;IF SO: BR
3720 015152 105711                    DFOF: TSTB      (R1)   ;SEE IF 0 CHAR
3721 015154 001404                    BEQ      DFOD         ;IF SO: BR
3722 015156 005201                    INC      R1           ;BUMP POINTER
3723 015160 005205                    DFOE: INC      R5           ;SEE IF DONE
3724 015162 001373                    BNE      DFOF        ;IF NOT: BR
3725 015164 000406                    BR       DFOC         ;ELSE CONTINUE
3726 015166 112721 000020          DFOD: MOVB      *20,(R1)+ ;SET 20 IN PLACE OF 0
3727 015172 012737 177777 013766  MOV      *-1,PATS     ;SET PATTERN GENERATE FLAG
3728 015200 000767                    BR       DFOE
3729 015202 013705 000556          DFOC: MOV      FMCNT,R5 ;RESET CHAR CNT
3730 015206 013701 000552          MOV      @WDATA,R1   ;RESET DATA ADDRESS
3731 015212 005737 000570          DFOCO: TST      RDCMD  ;SEE IF READ REVERSE
3732 015216 001462                    BEQ      DFO         ;IF NOT: BR
3733 015220 013704 000556          DFOB: MOV      FMCNT,R4 ;GET FRAME COUNT
3734 015224 005404                    NEG      R4           ;SET TO WHOLE NUMBER
3735 015226 032737 000020 000552  BIT      *20,UDES     ;SEE IF CORE DUMP
3736 015234 001402                    BEQ      DFOBO       ;IF NOT: BR
3737 015236 000241                    CLC
3738 015240 006004                    ROR      R4           ;SET TO FC/2
3739 015242 060401          DFOBO: ADD      R4,R1     ;POINT TO START OF WRITE DATA
3740 015244 060402                    ADD      R4,R2       ;POINT TO START OF READ DATA
3741 015246 032737 000001 000556  BIT      *1,FMCNT     ;SEE IF ODD FRAME COUNT
3742 015254 001401                    BEQ      DFOA        ;IF NOT: BR
3743 015256 105722                    TSTB    (R2)+        ;BUMP POINTER
3744 015260 032737 000020 000552  DFOA: BIT      *20,UDES ;SEE IF CORE DUMP
3745 015266 001431                    BEQ      DFOA4       ;IF NOT: BR
3746 015270 000241                    CLC

```

E 7

CZTEDEO 1M03-TE16-TU77 DRT  
CZTEDE.P11 07-MAR-84 14:04

MACY11 30(1046) 07-MAR-84 14:21 PAGE 54 1

SEQ 0082

3747	015272	132742	000001		BITB	#1, -(R2)		;SEE IF BIT 0 = 1
3748	015276	001401			BEQ	DFOA0		;IF NOT: BR
3749	015300	000261			SEC			
3750	015302	106012		DFOA0:	RORB	(R2)		
3751	015304	000241			CLC			
3752	015306	132712	000001		BITB	#1, (R2)		
3753	015312	001401			BEQ	DFOA1		
3754	015314	000261			SEC			
3755	015316	106012		DFOA1:	RORB	(R2)		;POSITION BITS FOR REVERSE CORE DUMP
3756	015320	000241			CLC			
3757	015322	132712	000001		BITB	#1, (R2)		
3758	015326	001401			BEQ	DFOA2		
3759	015330	000261			SEC			
3760	015332	106012		DFOA2:	RORB	(R2)		
3761	015334	000241			CLC			
3762	015336	132712	000001		BITB	#1, (R2)		
3763	015342	001401			BEQ	DFOA3		
3764	015344	000261			SEC			
3765	015346	106012		DFOA3:	RORB	(R2)		
3766	015350	005202			INC	R2		;RESET POINTER
3767	015352	124142		DFOA4:	CMPB	-(R1), -(R2)		;TEST DATA CHARACTER
3768	015354	001010			BNE	DF1		;IF NOT GOOD: BR
3769	015356	105037	000664		CLRB	BBC		;CLEAR BAD RECORD COUNTER
3770	015362	000411			BR	DF2		
3771	015364	122122		DFO:	CMPB	(R1)*, (R2)*		;CHECK DATA
3772	015366	001003			BNE	DF1		;IF BAD: BR
3773	015370	105037	000664		CLRB	BBC		;CLEAR BAD RECORD CNTR
3774	015374	000404			BR	DF2		
3775	015376	004737	016134	DF1:	JSR	PC, DRPKF		;GO GET DROPS AND PICKS
3776	015402	004737	015470		JSR	PC, DERR		;GO DO PRINT
3777	015406	005205		DF2:	INC	R5		;BUMP CHAR CNTR
3778	015410	001404			BEQ	DF3		;IF DONE ALL: BR
3779	015412	005737	000570		TST	RDCMD		;SEE IF READ REVERSE
3780	015416	001762			BEQ	DFO		;IF NOT: BR
3781	015420	000717			BR	DFOA		;ELSE CONTINUE READ REV
3782	015422	005037	000672	DF3:	CLR	HDRFL		;CLEAR HEADER FLAG
3783	015426	005737	000712		TST	DERFL		;SEE IF HAD DATA ERROR
3784	015432	001415			BEQ	DFX		;IF NOT: BR
3785	015434	005737	000714		TST	SERFL		
3786	015440	001012			BNE	DFX		;IF NOT DATA ERROR ONLY: BR
3787	015442	013704	000702		MOV	UNP, R4		
3788	015446	005737	000570		TST	RDCMD		;SEE IF READ REVERSE
3789	015452	001003			BNE	DF4		;IF SO: BR
3790	015454	005264	001134		INC	DATER1(R4)		;BUMP DATA ERROR FORWARD COUNTER
3791	015460	000402			BR	DFX		
3792	015462	005264	001174	DF4:	INC	DI REV1(R4)		;BUMP REVERSE DATA ERROR
3793	015466	000207		DFX:	RTS	PC		;EXIT
3794								

3796  
3797  
3798  
3799  
3800  
3801  
3802  
3803  
3804  
3805  
3806  
3807  
3808  
3809  
3810  
3811  
3812  
3813  
3814  
3815  
3816  
3817  
3818  
3819  
3820  
3821  
3822  
3823  
3824  
3825  
3826  
3827  
3828  
3829  
3830  
3831  
3832  
3833  
3834  
3835  
3836  
3837  
3838  
3839  
3840  
3841  
3842  
3843  
3844  
3845  
3846  
3847  
3848  
3849  
3850  
3851

015470 032777 002000 163120  
015476 001057  
015500 005237 000676  
015504 005737 000672  
015510 001006  
015512 004737 022126  
015516 000004 023722  
015522 004737 020364  
015526 000004 023741  
015532 010203  
015534 163703 000564  
015540 005303  
015542 005737 000570  
015546 001402  
015550 010503  
015552 005103  
015554 104400  
015556 000004 023727  
015562 005737 000570  
015566 001402  
015570 111103  
015572 000401  
015574 114103  
015576 004737 023504  
015602 000004 023734  
015606 005737 000570  
015612 001402  
015614 111203

```
*****  
; DATA ERROR SUBROUTINE:  
;  
; THIS SUBROUTINE IS USED TO PRINT OUT ANY  
; ERRORS FOUND DURING THE DATA CHECK.  
; EACH CHARACTER FOUND BAD WILL BE PRINTED  
; IN BIT FORMAT ALONG WITH ITS EXPECTED CHARACTER.  
; AN ERROR HEADER CONSISTING OF THE UNIT NUMBER,  
; BLOCK NUMBER, RECORD NUMBER, SIZE OF RECORD, AND  
; ERROR TYPE (READ FORWARD, READ REVERSE, WRITE, ETC)  
; IS PRINTED ONLY ONCE FOR EACH RECORD FOUND BAD.  
; A COUNT IS MADE OF THE NUMBER OF SUCCESSIVE BAD  
; CHARACTERS, AND IF TEN (10) SUCCESSIVE BAD CHARACTERS  
; ARE FOUND IN A SINGLE RECORD, A MESSAGE INDICATING  
; A BAD RECORD CONDITION IS PRINTED AND THE NEXT  
; TWENTY (20) CHARACTERS ARE SKIPPED BEFORE CHECKING  
; IS RESUMED. IF THE BAD RECORD CONDITION IS FOUND  
; THREE TIMES IN A RECORD, ALL REMAINING DATA IS  
; SKIPPED EXCEPT THE FINAL TEN (10) CHARACTERS.  
; THIS SKIPPING IS OF COURSE ONLY POSSIBLE IN  
; RECORDS WHICH CONTAIN A SUFFICIENT NUMBER OF CHARACTERS.  
; PRINTING OF ERRORS MAY BE DISALLOWED AT ANY TIME  
; BY SETTING CONSOLE SWITCH TEN (10) TO A ONE.  
; THE OPERATOR MAY CAUSE THE PROGRAM TO HALT ON ANY ERROR  
; BY SETTING CONSOLE SWITCH FIFTEEN (15) TO A ONE.  
*****  
DERR: BIT #2000,DSWR ;BRANCH IF NO ERROR  
BNE DERR4 ;PRINTOUT DESIRED  
DERR0: INC PFLG ;SET PRINT FLAG  
TST HDRFL ;SEE IF HAVE PRINTED HEADER  
BNE DERR0A ;IF SO; BR  
JSR PC,PAPRT ;PRINT CYCLE NUMBER  
TYPE,MSG1 ;TYPE DATA ERROR TAG 'DE'  
DERR0A: JSR PC,FRPRT ;PRINT F OR R  
TYPE,MSG4 ;TYPE CHAR # TAG 'CN'  
MOV R2,R3  
SUB @RDATA,R3 ;POINT TO CHAR  
DEC R3  
TST RDCMD ;SEE IF READ REVERSE  
BEQ DERR0B ;IF NOT; BR  
MOV R5,R3 ;GET CHAR NUMBER  
COM R3  
DERR0B: TYPOCT ;PRINT CHAR NUMBER  
TYPE,MSG2 ;TYPE GOOD CHAR TAG 'G'  
TST RDCMD ;SEE IF READ REVERSE  
BEQ DERR0C ;IF NOT; BR  
MOVB (R1),R5 ;GET CHAR  
BR DERR0D  
DERR0C: MOVB -(R1),R3 ;LOAD EXPECTED DATA  
DERR0D: JSR PC,DOUT ;GO PRINT CHAR  
TYPE,MSG3 ;TYPE BAD CHARACTER TAG 'B'  
TST RDCMD ;SEE IF READ REVERSE  
BEQ DERR1 ;IF NOT; BR  
MOVB (R2),R3 ;GET CHAR
```

3852	015616	000401			BR	DERR2		
3853	015620	114203			DERR1: MOV	-(R2),R3		
3854	015622	004737	023504		DERR2: JSR	PC,DOUT	;PRINT BAD CHAR	
3855	015626	005737	000570			TST	RDCMD	;BRANCH IF READ
3856	015632	001001				BNE	DERR4	;REVERSE
3857	015634	122122			DERR3: CMPB	(R1)+,(R2)+	;RESET POINTERS	
3858	015636	105237	000664		DERR4: INCB	BBC	;BUMP BAD RECORD CNTR	
3859	015642	122737	000010	000664		CMPB	#10,BBC	;SEE IF BLD BTH
3860	015650	001107				BNE	DEREX	;IF NOT: BR
3861	015652	032777	002000	162736		BIT	#2000,#SWR	;SEE IF PRINT INHIBIT
3862	015660	001002				BNE	1\$	;IF SO: BR
3863	015662	000004	024022			TYPE,MSG15		;TYPE 'BAD RECORD'
3864	015666	105037	000664		1\$: CLR	BBC	;RESET BAD RECORD CNTR	
3865	015672	105237	000665			INCB	BBC+1	;BUMP AMOUNT
3866	015676	122737	000003	000665		CMPB	#3,BBC+1	;SEE IF HAD 3 BLD BTHS
3867	015704	101047				BHI	DERR4B	;IF NOT: BR
3868	015706	022705	177767			CMP	#177767,R5	;SEE IF ON LAST EIGHT CHARS
3869	015712	101464				BLOS	DERR6	;IF SO: BR
3870	015714	012705	177767			MOV	#177767,R5	;SET CHAR CNTR TO 8
3871	015720	005737	000570			TST	RDCMD	;SEE IF READ REVERSE
3872	015724	001416				BEQ	DERR4A	;IF NOT: BR
3873	015726	013701	000562			MOV	#0WDATA,R1	;GET START OF BUFFER
3874	015732	013702	000564			MOV	#0RDATA,R2	;GET START OF BUFFER
3875	015736	062701	000010			ADD	#10,R1	
3876	015742	062702	000010			ADD	#10,R2	;POINT TO START +10
3877	015746	032737	000001	000556		BIT	#1,FMCNT	;SEE IF ODD FRAME COUNT
3878	015754	001445				BEQ	DEREX	;IF NOT: BR
3879	015756	105722				TSTB	(R2)+	;BUMP POINTER
3880	015760	000443				BR	DEREX	
3881	015762	013737	000556	000652	DERR4A: MOV	FMCNT,TEMP1	;LOAD CHAR COUNT	
3882	015770	005437	000652			NEG	TEMP1	;++B
3883	015774	162737	000010	000652		SUB	#10,TEMP1	;POINT TO BUFFER -8
3884	016002	013701	000652			MOV	TEMP1,R1	;POINT TO NEXT CHAR
3885	016006	063701	000562			ADD	#0WDATA,R1	;POINT TO NEXT WRITE CHAR
3886	016012	013702	000652			MOV	TEMP1,R2	;POINT TO END OF READ DATA -8 FORWARD
3887	016016	063702	000564			ADD	#0RDATA,R2	;POINT TO NEXT CHAR
3888	016022	000422				BR	DEREX	;EXIT
3889	016024	062705	000024		DERR4B: ADD	#24,R5	;SKIP 20 CHARS	
3890	016030	103415				BCS	DERR6	;IF EXCEED RECORD SIZE: BR
3891	016032	005737	000570			TST	RDCMD	;SEE IF READ REVERSE
3892	016036	001405				BEQ	DERR5	;IF NOT: BR
3893	016040	162701	000024			SUB	#24,R1	
3894	016044	162702	000024			SUB	#24,R2	;RESET POINTERS
3895	016050	000407				BR	DEREX	
3896	016052	062701	000024		DERR5: ADD	#24,R1	;SKIP 20 CHARS	
3897	016056	062702	000024			ADD	#24,R2	;SKIP FORWARD 20 CHARS
3898	016062	000402				BR	DEREX	
3899	016064	012705	177777		DERR6: MOV	#1,R5	;SET TO EOR	
3900	016070	005777	162522		DEREX: TST	#SWR	;BRANCH IF NOT HALT ON ERROR	
3901	016074	100012				BPL	DEREX1	
3902	016076	000000				HALT		
3903	016100	005737	000676			TST	PHLG	;SEE IF PRINTED
3904	016104	001006				BNE	DEREX1	;IF SO: BR
3905	016106	032777	002000	162502		BIT	#2000,#SWR	;SEE IF SHOULD PRINT
3906	016114	001002				BNE	DEREX1	;IF NOT: BR
3907	016116	000137	015500			JMP	DERRO	;ELSE PRINT

H/

CZTEDEO IM03-TE16-TU77 DRI  
CZTEDEF,P11 07-MAR-84 14:04

MACY11 30(1046) 07-MAR-84 14:21 PAGE 55-2

SEQ 0085

3908 016122 005037 000676  
3909 016126 005237 000712  
3910 016132 000207  
3911

DEREX1: CLR PFLG  
INC DERFL  
RTS PC

;CLEAR FLAG  
;BUMP DATA ERROR FLAG  
;RETURN

```

3913
3914
3915 ;*****
3916 ;DROPS AND PICKS SUBROUTINE;
3917 ;
3918 ;THIS SUBROUTINE IS USED TO ACCUMULATE FROM
3919 ;EACH BAD DATA CHARACTER FOUND THE NUMBER
3920 ;OF BITS WHICH WERE EITHER DROPPED OR PICKED UP.
3921 ;TWO COUNTERS PER SLAVE ARE USED TO ACCUMULATE THIS
3922 ;INFORMATION AND CAN STORE UP TO 32K DROPS
3923 ;OR PICKS BEFORE OVERFLOWING. IF OVERFLOW IS
3924 ;ABOUT TO OCCUR, THESE ACCUMULATORS ARE
3925 ;PRINTED IN OCTAL AND RESET TO ZERO.
3926 ;THE CONTENTS OF THE ACCUMULATORS MAY BE
3927 ;DISPLAYED AT ANY TIME BY SETTING CONSOLE
3928 ;SWITCH FOURTEEN TO A ONE (1). THE PRINTOUT WILL OCCUR
3929 ;AT THE END OF THE CURRENT BLOCK CYCLE.
3930 ;*****
3931 016134 005037 000652 DRPKF: CLR TEMP1
3932 016140 005037 000654 CLR TEMP2
3933 016144 005037 000656 CLR TEMP3
3934 016150 111137 000652 MOV (R1),TEMP1 ;LOAD GOOD CHAR
3935 016154 111237 000654 MOV (R2),TEMP2 ;LOAD BAD CHAR
3936 016160 013704 000702 MOV UNP,R4
3937 016164 016437 000774 000726 MOV PIK1(R4),BPKP
3938 016172 016437 001014 000724 MOV DRP1(R4),BDPP
3939 016200 005737 000570 TST RDCMD ;SEE IF READ REVERSE
3940 016204 001005 BNE DRPK ;IF SO: BR
3941 016206 124142 CMPB -(R1),-(R2) ;POINT TO CHAR
3942 016210 112137 000652 MOV (R1)+,TEMP1 ;LOAD GOOD CHAR
3943 016214 112237 000654 MOV (R2)+,TEMP2 ;LOAD BAD CHAR
3944 016220 004737 016232 DRPK: JSR PC,DROP ;GET DROPS
3945 016224 004737 016440 JSR PC,PICK ;GET PICKS
3946 016230 000207 RTS PC ;EXIT
3947
3948 016232 113703 000652 DROP: MOV TEMP1,R3 ;R3 = GOOD CHAR
3949 016236 113704 000654 MOV TEMP2,R4 ;R4 = BAD CHAR
3950 016242 140403 DPC: BICB R4,R3 ;GET DROPS/PICKS
3951 016244 001001 BNE DPCG ;IF SOME: BR
3952 016246 000207 RTS PC ;RETURN
3953 016250 012737 000010 000716 DPCG: MOV #10,BCNT ;SET NUMBER TO CHECK
3954 016256 132703 000001 DPCO: BITB #1,R3 ;SEE IF DROPPED OR PICKED THIS BIT
3955 016262 001451 BEQ DPC2 ;IF NOT: BR
3956 016264 105737 000656 TSTB TEMP3 ;SEE IF ON PICKS
3957 016270 001014 BNE DPC1 ;IF SO: BR
3958 016272 005277 162426 INC @BDPP ;BUMP DROP CNTR
3959 016276 100043 BPL DPC2 ;IF NO OVERFLOW: BR
3960 016300 032777 002000 162310 BIT #2000,@SWR ;SEE IF HAVE PRINTED DATA
3961 016306 001402 BEQ DPCOA ;IF SO: BR
3962 016310 004737 022126 JSR PC,PAPRT ;PRINT CYCLE NUMBER
3963 016314 004737 016504 JSR PC,PPRT ;PRINT DROPS AND PICKS
3964 016320 000413 BR DPC2A
3965 016322 005277 162400 DPC1: INC @BPKP ;BUMP PICK CNTR
3966 016326 100027 BPL DPC2 ;& BR IF NO OVERFLOW
3967 016330 032777 002000 162260 BIT #2000,@SWR ;SEE IF HAVE PRINTED DATA
3968 016336 001402 BEQ DPC1A ;IF SO: BR

```

3969	016340	004737	022126			JSR	PC,PAPRT		;PRINT CYCLE NUMBER
3970	016344	004737	016504			DPC1A: JSR	PC,DPPRT		;PRINT DROPS AND PICKS
3971	016350	013704	000702			DPC2A: MOV	UNP,R4		
3972	016354	016403	001014				MOV	DRP1(R4),R3	;SET DROP POINTER
3973	016360	016404	000774				MOV	PIK1(R4),R4	;SET PICK POINTER
3974	016364	012737	000010	000716			MOV	#10,BCNT	;SET NUMBER OF BITS
3975	016372	005023				DPC2B: CLR	(R3)+		;CLEAR DROPS
3976	016374	005024					CLR	(R4)+	;CLEAR PICK
3977	016376	005337	000716				DEC	BCNT	;SEE IF DONE
3978	016402	001375					BNE	DPC2B	;IF NOT: BR
3979	016404	000207					RTS	PC	;EXIT
3980	016406	000241				DPC2: CLC			
3981	016410	106003					RORB	R3	;GET NEXT BIT
3982	016412	005337	000716				DEC	BCNT	;SEE IF DONE
3983	016416	001407					BEQ	DPC3	
3984	016420	062737	000002	000726			ADD	#2,BPKP	
3985	016426	062737	000002	000724			ADD	#2,BDPP	
3986	016434	000710					BR	DPC0	;CONTINUE
3987	016436	000207				DPC3: RTS	PC		;RETURN
3988	016440	013704	000702			PICK: MOV	UNP,R4		;GET UNIT POINTER
3989	016444	016437	000774	000726			MOV	PIK1(R4),BPKP	;SET PICK POINTER
3990	016452	016437	001014	000724			MOV	DRP1(R4),BDPP	;SET DROP POINTER
3991	016460	113704	000652				MOVB	TEMP1,R4	;R4 = GOOD CHAR
3992	016464	113703	000654				MOVB	TEMP2,R3	;R3 = BAD CHAR
3993	016470	112737	000001	000656			MOVB	#1,TEMP3	;SET PICK FLAG
3994	016476	004737	016242				JSR	PC,DPC	;GO CHECK PICKS
3995	016502	000207					RTS	PC	;EXIT
3996	016504	000004	024333			DPPRT: TYPE,MSG26			;TYPE 'DROPS'
3997	016510	013704	000702				MOV	UNP,R4	
3998	016514	016437	001014	000724			MOV	DRP1(R4),BDPP	;SET DROP POINTER
3999	016522	016437	000774	000726			MOV	PIK1(R4),BPKP	;SET PICK POINTER
4000	016530	062737	000016	000724			ADD	#16,BDPP	
4001	016536	062737	000016	000726			ADD	#16,BPKP	
4002	016544	012737	000010	000716			MOV	#10,BCNT	;SET NUMBER TO PRINT
4003	016552	017703	162146			DPPRT0: MOV	#BDPP,R3		
4004	016556	104400					TYPOCT		;PRINT DROPS
4005	016560	005337	000716				DEC	BCNT	;SEE IF DONE
4006	016564	001404					BEQ	DPPRT1	;IF NOT: BR
4007	016566	162737	000002	000724			SUB	#2,BDPP	;BUMP POINTER
4008	016574	000766					BR	DPPRT0	;CONTINUE FOR ALL 8 BITS
4009	016576	012737	000010	000716		DPPRT1: MOV	#10,BCNT		;SET NUMBER TO PRINT
4010	016604	000004	024344				TYPE,MSG27		;TYPE 'PICKS'
4011	016610	017703	162112			DPPRT2: MOV	#BPKP,R3		
4012	016614	104400					TYPOCT		;PRINT PICKS
4013	016616	005337	000716				DEC	BCNT	;SEE IF DONE
4014	016622	001404					BEQ	DPPRTX	;IF SO: BR
4015	016624	162737	000002	000726			SUB	#1,BPKP	;BUMP POINTER
4016	016632	000766					BR	DPPRT2	;CONTINUE FOR ALL 8 BITS
4017	016634	000207				DPPRTX: RTS	PC		;RETURN

4019  
4020  
4021  
4022  
4023  
4024  
4025  
4026  
4027  
4028  
4029  
4030  
4031  
4032  
4033  
4034  
4035  
4036  
4037  
4038  
4039  
4040  
4041  
4042  
4043  
4044  
4045  
4046  
4047  
4048  
4049  
4050  
4051  
4052  
4053  
4054  
4055  
4056  
4057  
4058  
4059  
4060  
4061  
4062  
4063  
4064  
4065  
4066  
4067  
4068  
4069  
4070  
4071  
4072  
4073  
4074

016636 013703 000556  
016642 032703 000001  
016646 001401  
016650 005303  
016652 005403  
016654 032737 000020 000552  
016662 001401  
016664 006203  
016666 032737 000010 000700  
016674 001413  
016676 005737 000570  
016702 001405  
016704 013703 000564  
016710 162703 000002  
016714 000405  
016716 063703 000564  
016722 000402  
016724 063703 000562  
016730 032777 040000 161566  
016736 001403  
016740 005726  
016742 000137 020426  
016746 010337 020340  
016752 012704 000007  
016756 012701 020342  
016762 005021  
016764 005304  
016766 001375  
016770 020377 161520  
016774 001402

```
*****  
;STATUS CHECK SUBROUTINE:  
;  
;THIS SUBROUTINE IS USED TO PERFORM A CHECK OF  
;BOTH THE MASSBUS CONTROLLER (RH11) AND THE TAPE  
;CONTROLLER (TMO2). THE RH11 IS CHECKED FOR ERRORS  
;AS REFLECTED IN REGISTERS CS1 AND CS2 AND ALSO THAT  
;THE BUS ADDRESS (BA) AND WORD COUNT (WC) ARE  
;CORRECT. THE TMO2 IS CHECKED FOR DRIVE STATUS (DS),  
;DRIVE ERRORS (ER), AND PROPER FRAME COUNT. THE SPECIAL  
;CHECK CHARACTERS (CRC+LRC) ARE ALSO CHECKED WHEN  
;APPROPRIATE (IE: NRZ READ OR WRITE). CERTAIN TYPES  
;OF DRIVE ERRORS IN PE OPERATION WILL BE ACCOMPANIED  
;BY THE DISPLAY OF THE DEAD TRACK REGISTER (CC). THESE  
;TYPES ARE ER BITS 15,10,7,6. THE PRINTOUTS OF BAD  
;CRC,LRC,FC, AND BA WILL SHOW BOTH THE EXPECTED AND  
;RECEIVED VALUES (IE: EXPT-RCVD). ONLY THOSE REGISTERS  
;WHICH ARE IN ERROR WILL BE PRINTED AND ALL PRINTOUTS  
;ARE IN OCTAL FORMAT WITH NO LEADING ZEROS. AS IN  
;DATA ERRORS, STATUS ERRORS ARE PRECEDED BY HEADER  
;DESCRIBING THE HARDWARE UNDER TEST, THE BLOCKING  
;INFORMATION, AND THE ERROR TYPE.  
*****  
ERCHK: MOV FMCNT,R3 ;GET FRAME COUNT  
BIT #1,R3 ;SEE IF ODD  
BEQ 1$ ;IF NOT: BR  
DEC R3 ;BUMP COUNT  
1$: NEG R3  
BIT #20,UDES ;SEE IF CORE DUMP  
BEQ 2$ ;IF NOT: BR  
ASR R3 ;SET TO FC/2  
2$: BIT #10,MTC1 ;SEE IF WRITE OP  
BEQ 4$ ;IF SO: BR  
TST RDCMD  
BEQ 3$  
MOV @RDATA,R3  
SUB #2,R3 ;SET POINTER  
BR ER2  
3$: ADD @RDATA,R3 ;BUILD EXPT READ ADDRESS  
BR ER2  
4$: ADD @WDATA,R3 ;BUILD EXPT WRITE ADDRESS  
ER2: BIT #40000,@ER ;BRANCH IF NOT UNSAFE  
BEQ 1$  
TST (SP)+ ;ADJUST STACK  
JMP OFFLINE ;GO MARK UNIT OFFLINE  
1$: MOV R3,CADR ;SAVE ADDRESS  
MOV #7,R4  
MOV #BAER,R1  
2$: CLR (R1)+ ;CLEAR FLAGS  
DEC R4  
BNE 2$  
CMP R3,@BA ;SEE IF ADDRESS OK  
BEQ 3$ ;IF SO: BR
```



4075	016776	005237	020342			INC	BAER		;SET BUS ADDRESS ERROR
4076	017002	032737	000010	000700	3\$:	BIT	#10,MTC1		;SEE IF WRITE OPER
4077	017010	001006				BNE	5\$		;IF NOT: BR
4078	017012	005777	161500		4\$:	TST	FC		;SEE IF FC=0
4079	017016	001440				BEQ	R3		;IF SO: BR
4080	017020	005237	020350			INC	FCER		;SET FC ERROR
4081	017024	000435				BR	R3		
4082	017026	032737	000040	000700	5\$:	BIT	#40,MTC1		;SEE IF SPACE OPER
4083	017034	001766				BEQ	4\$		;IF SO: BR
4084	017036	005737	000704			TST	TMFLG		;SEE IF TM TIME
4085	017042	001011				BNE	7\$		;IF SO: BR
4086	017044	013703	000556			MOV	FCNT,R3		
4087	017050	005403				NEG	R3		;R3 = EXPT RECORD SIZE
4088	017052	020377	161440		6\$:	CMP	R3,#FC		;SEE IF FC = EXPT
4089	017056	001420				BEQ	R3		;IF SO: BR
4090	017060	005237	020350			INC	FCER		;SET FC ERROR FLAG
4091	017064	000415				BR	R3		
4092	017066	032737	002000	000552	7\$:	BIT	#2000,UDES		;SEE IF PE
4093	017074	001346				BNE	4\$		;IF SO: BR
4094	017076	005737	000570			TST	RDCMD		;SEE IF READ REVERSE
4095	017102	001003				BNE	8\$		;IF SO: BR
4096	017104	012703	000002			MOV	#2,R3		
4097	017110	000760				BR	6\$		;LOOK FOR EXPT = 2
4098	017112	012703	000001		8\$:	MOV	#1,R3		
4099	017116	000755				BR	6\$		;GO CHECK FC FOR TM
4100									
4101	017120	032777	160000	161362	ER3:	BIT	#160000,#C1		;SEE IF COUNT ERROR
4102	017126	001437				BEQ	R4		
4103	017130	017703	161364			MOV	#C3,R3		;GET CONT STATUS REG
4104	017134	042703	000307			BIC	#307,R3		;MASK OUT IR,OR,UNIT NO. & SEE IF OTHER ERRORS
4105	017140	001406				BEQ	1\$		;IF NOT: BR
4106	017142	005737	000704			TST	TMFLG		;SEE IF TAPE MARK TIME
4107	017146	001425				BEQ	3\$		;IF NOT: BR
4108	017150	042703	001000			BIC	#1000,R3		;MASK MISSED TRANS & BR IF OTHER ERRORS
4109	017154	001022				BNE	3\$		
4110	017156	032777	060000	161324	1\$:	BIT	#60000,#C1		;SEE IF EITHER TRE OR MCPE
4111	017164	001420				BEQ	R4		;IF NOT: BR
4112	017166	005737	000704			TST	TMFLG		;SEE IF TM TIME
4113	017172	001413				BEQ	3\$		;IF NOT: BR
4114	017174	017703	161324			MOV	#ER,R3		;GET ERROR REGISTER
4115	017200	032737	000010	000552		BIT	#10,UDES		;SEE IF EVEN PARITY
4116	017206	001402				BEQ	2\$		;IF NOT: BR
4117	017210	042703	000100			BIC	#100,R3		;MASK PAR
4118	017214	042703	001000		2\$:	BIC	#1000,R3		;MASK FCE
4119	017220	001402				BEQ	R4		;IF NO ERRORS EXCEPT FCE: BR
4120	017222	005237	020344		3\$:	INC	CONER		;SET CONT ERROR FLAG
4121									
4122	017226	032777	040000	161266	ER4:	BIT	#40000,#DS		;SEE IF DRIVE ERROR
4123	017234	001420				BEQ	R6		;IF NOT: BR
4124	017236	005737	000704			TST	TMFLG		;SEE IF TAPE MARK TIME
4125	017242	001413				BEQ	2\$		;IF NOT: BR
4126	017244	017703	161254			MOV	#ER,R3		;GET ER
4127	017250	032737	000010	000552		BIT	#10,UDES		;SEE IF EVEN PARITY
4128	017256	001402				BEQ	1\$		;IF NOT: BR
4129	017260	042703	000100			BIC	#100,R3		;MASK PAR
4130	017264	042703	001000		1\$:	BIC	#1000,R3		;MASK OUT FCE & BRANCH IF

4131	017270	001402				BEQ	ER6		;NO OTHER ERRORS
4132	017272	005237	020346		2\$:	INC	DRVER		;SET DRIVER ERROR FLAG
4133									
4134	017276	013737	015060	020362	ER6:	MOV	EXCRC,CRCSV		;SAVE EXPECTED CRC
4135	017304	013737	015060	020360		MOV	EXLRC,LRCV		;AND EXPECTED LRC
4136	017312	032737	002000	000552		BIT	#2000,UDES		
4137	017320	001062				BNE	ERPT		;IF IN PE MODE: BR
4138	017322	032777	020000	161266		BIT	#20000,BSWR		;SEE IF NO DATA CHECK
4139	017330	001056				BNE	ERPT		;IF NOT: BR (ALLOW READ OF UNKNOWN TAPES)
4140	017332	032737	000040	000700		BIT	#40,MTCL		;SEE IF WRITE OR READ OP
4141	017340	001452				BEQ	ERPT		;IF NOT: BR
4142	017342	005737	000704			TST	TMFLG		;SEE IF TAPE MARK TIME
4143	017346	001405				BEQ	1\$		;IF NOT: BR
4144	017350	005037	015060			CLR	EXCRC		
4145	017354	012737	000023	015062		MOV	#23,EXLRC		;SET CRC/LRC FOR TM
4146	017362	032737	000060	000552	1\$:	BIT	#60,UDES		;SEE IF FORMAT 14
4147	017370	001036				BNE	ERPT		;IF NOT: BR
4148	017372	017703	161132			MOV	#CC,R3		;GET CRC CHARACTER
4149	017376	042703	177000			BIC	#177000,R3		
4150	017402	023703	015060			CMR	EXCRC,R3		
4151	017406	001402				BEQ	2\$		;IF CRC GOOD: BR
4152	017410	005237	020354			INC	CRCER		;SET ERROR FLAG
4153	017414	017703	161114		2\$:	MOV	#MR,R3		;GET LRC
4154	017420	000303				SWAB	R3		
4155	017422	005703				TST	R3		
4156	017424	100002				BPL	3\$		
4157	017426	052703	000400			BIS	#400,R3		
4158	017432	042703	177000		3\$:	BIC	#177000,R3		
4159	017436	023703	015062			CMR	EXLRC,R3		
4160	017442	001411				BEQ	ERPT		;IF LRC GOOD: BR
4161	017444	010337	020356			MOV	R3,ACTLRC		;SAVE ACTUAL LRC
4162	017450	005237	020352			INC	LRCER		;SET LRC ERROR FLAG
4163	017454	005737	000570			TST	RDCMD		;SEE IF READ REVERSE
4164	017460	001402				BEQ	ERPT		;IF NOT: BR
4165	017462	005037	020352			CLR	LRCER		;ELSE CLEAR LRC ERROR
4166	017466	012703	000006		ERPT:	MOV	#6,R3		
4167	017472	005037	000714			CLR	SERFL		;CLEAR ERROR FLAG
4168	017476	005037	000730			CLR	ERSAV		
4169	017502	012704	020342			MOV	#BAER,R4		
4170	017506	005724			ERPTT:	TST	(R4)+		;SEE IF ANY ERROR
4171	017510	001004				BNE	ERPTG		;IF SO: BR
4172	017512	005303				DEC	R3		
4173	017514	001374				BNE	ERPTT		
4174	017516	000137	020304			JMP	ERPX1		
4175	017522	005237	000714		ERPTG:	INC	SERFL		;SET ERROR FLAG
4176	017526	017737	160772	000730		MOV	#ER,ERSAV		;SAVE ERROR REGISTER
4177	017534	032777	002000	161054		BIT	#2000,BSWR		;SEE IF PRINT
4178	017542	001420				BEQ	ERPTO		;IF SO: BR
4179	017544	022737	000002	000720		CMR	#2,RTYFL		;SEE IF READ RETRY
4180	017552	001006				BNE	ERPTG1		;IF NOT: BR
4181	017554	013703	000710			MOV	RTCNT,R3		
4182	017560	005203				INC	R3		;BUMP RETRY COUNT
4183	017562	020337	000612			CMR	R3,RETRY		;SEE IF LAST RETRY
4184	017566	001406				BEQ	ERPTO		;IF SO: BR
4185	017570	022737	000002	020346	ERPTG1:	CMR	#2,DRVER		;SEE IF TM STATUS ERROR
4186	017576	001402				BEQ	ERPTO		;IF SO: BR

4187	017600	000137	020164		JMP	ERPX0	
4188	017604	005237	000676	ERPT0:	INC	PFLG	
4189	017610	004737	022126		JSR	PC,PAPRT	;PRINT HEADER
4190	017614	013737	000660	017624	MOV	EMADDR,1\$	;GET ADDRESS OF ERROR MSG HEADER
4191	017622	000004			TYPE		
4192	017624	000000		1\$:	.WORD	0	;ADDRESS OF ERROR MESSAGE HEADER
4193	017625	004737	020364		JSR	PC,FRPRT	;PRINT F OR R
4194	017632	005737	000704		TST	TMFLG	
4195	017636	001406			BEQ	ERPT1	
4196	017640	022737	025212	000660	CMP	#MSG54,EMADDR	
4197	017646	001402			BEQ	ERPT1	
4198	017650	000004	025230		TYPE,MSG56		;TYPE 'TM'
4199	017654	005737	020344	ERPT1:	TST	CONER	
4200	017660	001412			BEQ	ERPT2	;IF NO CONT ERROR: BR
4201	017662	000004	020137		TYPE,MSG23		;TYPE 'CS1'
4202	017666	017733	160616		MOV	@C1,R3	
4203	017672	104400			TYPOCT		;PRINT CONTROL 1
4204	017674	000004	024164		TYPE,MSG23D		;TYPE CS TAG
4205	017700	017703	160614		MOV	@CS,R3	
4206	017704	104400			TYPOCT		;PRINT CONT STATUS
4207	017706	005737	020346	ERPT2:	TST	DRVER	
4208	017712	001412			BEQ	ERPT3	;IF SO DRIVE ERROR: BR
4209	017714	000004	024172		TYPE,MSG23E		;TYPE DS TAG
4210	017720	017703	160576		MOV	@DS,R3	
4211	017724	104400			TYPOCT		;PRINT DRIVE STATUS
4212	017726	000004	024177		TYPE,MSG23F		;TYPE ER TAG
4213	017732	017703	160566		MOV	@ER,R3	
4214	017736	104400			TYPOCT		;PRINT DRIVE ERROR
4215	017740	005737	020342	ERPT3:	TST	BAER	
4216	017744	001412			BEQ	ERPT4	;IF NO BA ERROR: BR
4217	017746	000004	024152		TYPE,MSG23B		;TYPE BA TAG
4218	017752	017703	160536		MOV	@BA,R3	
4219	017756	104400			TYPOCT		;PRINT BUS ADDRESS
4220	017760	000004	023720		TYPE,DASH		
4221	017764	013703	020340		MOV	CADER,R3	
4222	017770	104400			TYPOCT		;PRINT EXPT BUS ADDRESS
4223	017772	005737	020350	ERPT4:	TST	FCER	
4224	017776	001405			BEQ	ERPT5	;IF NO FC ERROR: BR
4225	020000	000004	024157		TYPE,MSG23C		;TYPE FC TAG
4226	020004	017703	160506		MOV	@FC,R3	
4227	020010	104400			TYPOCT		;PRINT FRAME COUNT
4228	020012	000004	024145	ERPT5:	TYPE,MSG23A		;TYPE WC TAG
4229	020016	017703	160470		MOV	@WC,R3	
4230	020022	104400			TYPOCT		;PRINT WORD COUNT
4231	020024	005737	020354		TST	CRCER	
4232	020030	001414			BEQ	ERPT5A	;IF NO CRC ERROR: BR
4233	020032	000004	025255		TYPE,MSG58		;TYPE CRC TAG
4234	020036	017703	160466		MOV	@CC,R3	
4235	020042	042703	177000		BIC	#177000,R3	
4236	020046	104400			TYPOCT		;PRINT ACTUAL CRC
4237	020050	000004	023720		TYPE,DASH		
4238	020054	013703	015060		MOV	EXCRC,R3	
4239	020060	104400			TYPOCT		;PRINT EXPECTED CRC
4240	020062	005737	020352	ERPT5A:	TST	LRCER	
4241	020066	001412			BEQ	ERPT6	;IF NO LRC ERROR: BR
4242	020070	000004	025263		TYPE,MSG59		;TYPE LRC ERR TAG

```

4243 020074 013703 020356      MOV      ACTLRC,R3
4244 020100 104400                TYP OCT      ;PRINT ACTUAL LRC
4245 020102 000004 023720      TYPE,DASH
4246 020106 013703 015062      MOV      EXLRC,R3
4247 020112 104400                TYP OCT      ;PRINT EXPECTED LRC
4248 020114 005737 020346      ERPT6:  TST      DRIVER
4249 020120 001420                BEQ      ERPT7      ;IF NO DRIVE ERROR: BR
4250 020122 032737 002000 000552  BIT      @2000,ODES
4251 020130 001414                BEQ      ERPT7      ;IF NO PE: BR
4252 020132 017704 160366      MOV      @ER,R4
4253 020136 042704 075477      BIC      @75477,R4      ;MASK OUT ALL BUT BITS 15,10,7,6
4254 020142 001407                BEQ      ERPT7      ;IF NO CONDITIONALS SET: BR
4255 020144 000004 024211      TYPE,MSG23H          ;TYPE CC TAG
4256 020150 017703 160354      MOV      @CC,R3
4257 020154 042703 177000      BIC      @177000,R3    ;MASK CC
4258 020160 104400                TYP OCT      ;PRINT CHECK CHARACTERS
4259 020162 000240                ERPT7:  NOP
4260 020164 005777 160426      ERPX0:  TST      @SWR
4261 020170 100012                BPL      ERPX
4262 020172 000000                HALT
4263 020174 005737 000676      TST      PFLG
4264 020200 001006                BNE      ERPX
4265 020202 032777 002000 160406  BIT      @2000,@SWR
4266 020210 001002                BNE      ERPX
4267 020212 000137 017604      JMP      ERPT0
4268 020216 005037 000676      ERPX:   CLR      PFLG
4269 020222 005737 000574      TST      CRCC
4270 020226 001007                BNE      1$
4271 020230 012777 000040 160262  MOV      @40,@CS
4272 020236 013777 000550 160254  MOV      @VN,@CS
4273 020244 000414                BR       2$
4274 020246 012777 000011 160234  1$:  MOV      @11,@C1
4275 020254 017704 160246      MOV      @AS,R4
4276 020260 010477 160242      MOV      R4,@AS
4277 020264 013704 000510      MOV      C1,R4
4278 020270 005204                INC      R4
4279 020272 152714 000100      BISR      @100,(R4)
4280 020276 013777 000552 160236  2$:  MOV      @DES,@IC
4281 020304 032737 000040 000700  ERPX1: BIT      @40,HTC1
4282 020312 001411                BEQ      ERPX2
4283 020314 005737 000704      TST      TMFLG
4284 020320 001406                BEQ      ERPX2      ;IF NOT TM TIME: BR
4285 020322 013737 020362 015060  MOV      CRCSV,EXCRC
4286 020330 013737 020360 015062  MOV      LRCSV,EXLRC
4287 020336 000207      ERPX2: RTS      PC
4288 020340 000000      CADER:  0
4289 020342 000000      BAER:   0
4290 020344 000000      CONER:  0
4291 020346 000000      DRVER:  0
4292 020350 000000      FCEP:   0
4293 020352 000000      LRCEP:  0
4294 020354 000000      CRCEP:  0
4295 020356 000000      ACTLRC: 0
4296 020360 000000      LRCSV:  0
4297 020362 000000      CRCSV:  0
4298

```

```

4300
4301
4302
4303
4304
4305
4306
4307
4308 020364 032737 000010 000700 FRPRT: BIT 010,MTC1 ;SEE IF WRITE COMMAND
4309 020372 001414 BEQ 3$ ;IF SO: BR
4310 020374 012737 024055 020422 MOV 0MSG17,2$ ;PRSET MESSAGE TO READ REVERSE
4311 020402 032737 000002 000700 BIT 02,MTC1 ;BRANCH IF REVERSE
4312 020410 001003 BNE 1$
4313 020412 012737 024052 020422 MOV 0MSG16,2$ ;SET FORWARD MESSAGE
4314 020420 000004 1$: TYPE ;TYPE MSG
4315 020422 000000 2$: .WORD 0
4316 020424 000207 3$: RTS PC ;EXIT
4317
4318
4319 ;ROUTINE TO MARK UNIT OFF LINE
4320
4321 020426 013701 000702 OFFLINE:MOV UNP,R1 ;GET UNIT POINTER
4322 020432 052761 040000 000752 BIS 40000,UN1(R1) ;MARK UNIT OFF LINE
4323 020440 000004 024261 TYPE,MSG2$ ;TYPE 'SLAVE UNSAFE NO FURTHER TESTING ON SLAVE
4324 020444 005737 000742 TST ASEQF ;BRANCH IF NOT IN AUTO SEQUENCE
4325 020450 001406 BEQ 1$
4326 020452 000004 026446 TYPE,MSG12$ ;TYPE 'AUTO SEQ TEST WILL RESTART
4327 020456 012706 000500 MOV 0500,SP ;RESET STACK PTR
4328 020462 000137 021342 JMP ASEQ0 ;RESTART AUTO SEQ
4329 020466 105337 005043 1$: DECB REOTC+1 ;DECREMENT UNITS TO TEST CTR
4330 020472 001003 BNE 2$
4331 020474 000004 026410 TYPE,MSG12$ ;TYPE 'NO UNITS LEFT TO TEST; HALT
4332 020500 000000 HALT
4333 020502 000137 004266 2$: JMP REOT
4334

```

4337  
4338  
4339  
4340  
4341  
4342  
4343  
4344  
4345  
4346  
4347  
4348  
4349  
4350  
4351  
4352  
4353  
4354  
4355  
4356  
4357  
4358  
4359  
4360  
4361  
4362  
4363  
4364  
4365  
4366  
4367  
4368  
4369  
4370  
4371  
4372  
4373  
4374  
4375  
4376  
4377  
4378  
4379  
4380  
4381  
4382  
4383  
4384  
4385  
4386  
4387  
4388  
4389  
4390  
4391  
4392

020506 005037 000652  
020512 013777 000550 160000  
020520 032777 040000 157776  
020526 001402  
020530 000137 020426  
020534 032777 020000 157760  
020542 001410  
020544 004737 022126  
020550 000004 026244  
020554 032777 020000 157740  
020562 001374  
020564 022737 000026 000700  
020572 001003  
020574 012704 177777  
020600 000406  
020602 013704 000556  
020606 032704 000001  
020612 001401  
020614 005304  
020616 000261  
020620 006004  
020622 032737 000020 000552  
020630 001402  
020632 000261  
020634 006004  
020636 010477 157650  
020642 012777 000011 157640

```

;*****
;TAPE COMMAND EXECUTE SUBROUTINE;
;
;THIS SUBROUTINE IS USED TO EXECUTE THE
;MAG TAPE COMMAND DESCRIBED BY THE READ
;OR WRITE ROUTINE. THE FINAL COMMAND IS
;SENT TO THE DEVICE REGISTER ALONG WITH THE
;INTERRUPT ENABLE AND GO BITS.
;ONCE THE COMMAND IS ISSUED, AN INTERRUPT
;TIMER IS STARTED AND IF NO INTERRUPT IS RETURNED
;BEFORE TIME OUT OCCURS, AN ERROR WILL BE
;PRINTED AND THE PROGRAM STOPPED. TESTING MAY
;BE RESUMED BY PRESSING THE CONTINUE SWITCH.
;TWO INTERRUPT HANDLERS ARE USED, ONE FOR MAG TAPE
;AND ANOTHER FOR TELETYPE (TTY).
;UPON RECEIPT OF A MAG TAPE INTERRUPT, HOUSEKEEPING
;IS PERFORMED AND CONTROL RETURNED TO THE CALLING
;ROUTINE (READ,WRITE,ETC).
;RECEIPT OF A TTY INTERRUPT WILL CAUSE THE
;PROGRAM TO CHECK FOR ENTRY OF A CNTRL C CHARACTER.
;IF NOT CNTRL C, THEN CONTINUATION OF WAIT FOR MAG
;TAPE INTERRUPT IS RETURNED. IF, HOWEVER, THE TTY
;INTERRUPT WAS CAUSED BY ENTRY OF A CNTRL C,
;THEN AT THIS TIME REQUESTS FOR NEW STALL VALUES
;ARE PRINTED AND THE RESPONSES ENTERED. RESUMPTION
;OF TAPE INTERRUPT WAIT IS THEN RESUMED.
;*****
TAPG: CLR      TEMP1
      MOV     DVN,%CS      ;SET DRIVE NO.
      BIT     @40000,%ER   ;SEE IF UNIT SAFE
      BEQ    TAPG3       ;IF SO: BR
      JMP    OFFLINE     ;GO MARK UNIT OFF-LINE
TAPG3: BIT     @20000,%DS  ;SEE IF PIP RESET
      BEQ    TAPG3F      ;IF SO: BR
      JSR    PC,PAPRT    ;PRINT HEADER
      TYPE  MSG116       ;TYPE MSG
      BIT     @20000,%DS  ;SEE IF PIP RESET
      BNE    1$         ;AWAIT PIP RESET
TAPG3F: CMP    @26,MIC1   ;SEE IF WRITE TM
      BNE    TAPG3A     ;IF NOT: BR
      MOV    @1,R4      ;ELSE SET FC FOR -1
      BR    TAPG3B
TAPG3A: MOV    FMCNT,R4
      BIT     @1,R4
      BEQ    TAPG3B
      DEC    R4
TAPG3B: SEC
      ROR    R4          ;SET WC = FC/2 FOR NORMAL FORMAT
      BIT     @20,%DE5   ;SEE IF CORE DUMP FORMAT
      BEQ    TAPG3C     ;IF NOT: BR
      SEC
      ROR    R4          ;SET WC = FC/4 FOR CORE DUMP
TAPG3C: MOV    R4,%WC    ;SET WORD COUNT
      MOV    @11,%C1    ;DRIVE CLEAR

```

4393	020650	017777	157642	157640	MOV	0FC,0FC	;RESET FC LOADED
4394	020656	005737	000576		TST	INTRF	;SEE IF INTERCHANGE READ
4395	020662	001407			BEQ	TAPG3D	;IF NOT: BR
4396	020664	032737	000040	000700	BIT	040,MTC1	;SEE IF READ OP
4397	020672	001403			BEQ	TAPG3D	;IF NOT: BR
4398	020674	012777	000003	157632	MOV	03,0MR	;SET INTERCHANGE READ MAINT. MODE
4399	020702	013704	000700		TAPG3D: MOV	MTC1,R4	;GET COMMAND
4400	020706	042704	177707		BIC	0177707,R4	;MASK OP CODE
4401	020712	022704	000030		CMP	030,R4	;SEE IF SPACE OP CODE
4402	020716	001403			BEQ	TAPG3E	;IF SO: BR
4403	020720	012737	177740	000674	MOV	0-40,STAL	;SET INTERRUPT DELAY MULT TO 40
4404	020726	052737	000101	000700	TAPG3E: BIS	0101,MTC1	;SET INTERRUPT ENABLE AND GO
4405	020734	000240			NOP		
4406	020736	013777	000700	157544	MOV	MTC1,0C1	;EXECUTE COMMAND
4407	020744	005077	157644		CLR	0PSW	;CLEAR PRIORITY
4408	020750	005037	000652		CLR	TEMP1	
4409	020754	005237	000652		TAPG4: INC	TEMP1	;SEE IF HAVE TIMED OUT
4410	020760	001375			BNE	TAPG4	;IF NOT: BR
4411	020762	005237	000674		INC	STAL	
4412	020766	001372			BNE	TAPG4	;DO TIME DELAY MULTIPLIER
4413	020770	012777	000340	157616	TAPG5: MOV	0340,0PSW	;RESET PRIORITY
4414	020776	032777	002000	157612	BIT	02000,0SWR	;SEE IF SHOULD PRINT ERRORS
4415	021004	001013			BNE	TAPG6	;IF NOT: BR
4416	021006	004737	022126		JSR	PC,PAPRT	;PRINT CYCLE NUMBER
4417	021012	013737	000660	021022	MOV	EMADDR,10	
4418	021020	000004			TYPE		;TYPE MSG
4419	021022	000000			10: .WORD	0	
4420	021024	004737	020364		JSR	PC,FRPRT	;PRINT F OR R
4421	021030	000004	024242		TYPE,MSG24		;TYPE 'NO INTERRUPT'
4422	021034	005777	157556		TAPG6: TST	0SWR	;BRANCH IF NOT HALT ON ERROR
4423	021040	100001			BPL	TAPG7	
4424	021042	000000			HALT		
4425	021044	000137	021276		TAPG7: JMP	MTINTA	;RETURN TO CALLING ROUTINE
4426							

```

4428
4429
4430 021050 017746 157546      TTINT: ;TTY INTERRUPT HANDLER
4431 021054 042716 000200      MOV    @TKB,-(SP)      ;GET CHARACTER
4432 021060 122716 000003      BIC    @200,(SP)      ;STRIP PARITY BIT
4433 021064 001005                CMPB   @3,(SP)        ;BRANCH IF NOT +C
4434 021066 000005                BNE    1$
4435 021070 005077 157520      RESET ;RESET ALL I/O
4436 021074 000137 000200      CLR    @PSW           ;CLEAR PSW
4437 021100 122716 000001      1$:  JMP    @@200          ;RESTART PROGRAM
4438 021104 001015                CMPB   @1,(SP)        ;BRANCH IF NOT +A
4439 021106 022737 000176 000616 BNE    2$
4440 021114 001014                CMP    @SWREG,SWR     ;BRANCH IF HARDWARE SWR IS INVOKED
4441 021116 012737 177570 000616 BNE    3$
4442 021124 004737 022546      MOV    @177570,SWR    ;INVOKE HARWARE SWR
4443 021130 000004 026362      JSR    PC,.SAVE       ;SAVE REGISTERS ON THE STACK
4444 021134 004737 022570      TYPE ,MSG121          ;TYPE 'HARDWARE SWR IN USE'
4445 021140 022716 000007      JSR    PC,.RESTORE    ;RESTORE REGISTERS
4446 021144 001005                CMP    @7,(SP)        ;BRANCH IF NOT +G
4447 021146 012737 000176 000616 BNE    4$
4448 021154 004737 022462      3$:  MOV    @SWREG,SWR     ;INVOKE SOFTWARE SWR
4449 021160 022716 000002      4$:  JSR    PC,GTSWR       ;GET SWITCHES
4450 021164 001041                CMP    @2,(SP)        ;BRANCH IF NOT +B
4451 021166 004737 022546      BNE    6$
4452 021172 005237 013560      JSR    PC,.SAVE       ;SAVE REGISTERS ON THE STACK
4453 021176 004737 013342      INC    SCVFL          ;SET FLAG
4454 021202 032777 000040 157406 JSR    PC,TINP3A      ;GO CHECK CRC CORRECTION
4455 021210 001425                BIT    @40,@SWR       ;BRANCH IF NOT YOZZLING
4456 021212 000004 025052      BEQ    5$
4457 021216 013703 000610      TYPE ,MSG44          ;REQUEST NEW YOZZLE STALL
4458 021222 104400                MOV    YSTAL,R3
4459 021224 012705 000610      TYPOCT ;PRINT PRESENT STALL
4460 021230 012701 000007      MOV    @YSTAL,R5     ;SET ADDRESS OF YSTL
4461 021234 012702 177777      MOV    @7,R1         ;SET NUMBER OF CHAR TO INPUT
4462 021240 012703 002000      MOV    @-1,R2        ;SET MAXIMUM LIMIT
4463 021244 004737 022612      MOV    @2000,R3      ;SET MINIMUM LIMIT
4464 021250 004737 022570      JSR    PC,ITR        ;GO GET VALUE
4465 021254 005726                JSR    PC,.RESTORE    ;RESTORE REGISTERS
4466 021256 012716 011102      TST   (SP)+          ;POP CHARACTER OF THE STACK
4467 021262 000002                MOV    @YOZ,(SP)     ;RETURN TO 'YOZ'
4468 021264 004737 022570      RTI ;RETURN TO YOZ
4469 021270 005726                5$:  JSR    PC,.RESTORE    ;POP CHARACTER OFF THE STACK
4470 021272 000002                6$:  TST   (SP)+          ;RETURN
4471
4472
4473 021274 000240                ;MAG TAPE INTERRUPT HANDLER
4474 021276 042777 000037 157230 MTINT: NOP
4475 021304 013716 000670      MTINTA: BIC    @57,@MR     ;CLEAR MAINT MODE
4476 021310 000002                MOV    RTRN,(SP)     ;SET RETURN TO (RTRN)
                                RTI ;RETURN

```



```

4478 ;*****
4479 ;AUTO SEQUENCE
4480 ;
4481 ;THIS ROUTINE ,ENTERED VIA STARTING ADDRESS 240
4482 ;WILL EXERCISE ALL AVAILABLE SLAVES ON ALL AVAILABLE
4483 ;DRIVES IN BOTH PE AND NRZ ACCORDING TO THE PRESELECTED
4484 ;TEST PLAN. IF NRZ ONLY, PE TESTING WILL NOT BE ATTEMPTED.
4485 ;*****
4486
4487 021312 000004 025677 ASEQ: TYPE,MSG104 ;REQUEST 'AUTO CONT'
4488 021316 012705 000746 MOV #ASEQCF,R5 ;SET ADDRESS OF ENTRY
4489 021322 012701 000002 MOV #2,R1 ;SET SIZE OF ENTRY
4490 021326 012702 000001 MOV #1,R2 ;SET UPPER LIMIT
4491 021332 012703 000000 MOV #0,R3 ;SET LOWER LIMIT
4492 021336 004737 022612 JSR PC,TTR ;GO GET INPUT
4493 021342 005037 000550 ASEQ0: CLR DVN ;SET DRIVE # 0
4494 021346 004737 021454 ASEQ1: JSR PC,HRDS ;GO SELECT HARDWARE CONFIGURATION
4495 021352 000004 025647 TYPE,MSG101 ;TYPE '*****'
4496 021356 000004 025156 TYPE,MSG52A ;TYPE 'DRIVE (TMO3) = '
4497 021362 013703 000550 MOV DVN,R3
4498 021366 104400 TYPOCT ;PRINT DRIVE #
4499 021370 000004 026535 TYPE,SPACE
4500 021374 000004 024600 TYPE,MSG32 ;TYPE ' SLAVE # = '
4501 021400 012700 000752 MOV #UN1,R0 ;POINT TO START OF SLAVE TABLE
4502 021404 012003 1$: MOV (R0)+,R3
4503 021406 100402 BMI 2$
4504 021410 104400 TYPOCT ;PRINT SLAVE TABLE
4505 021412 000774 BR 1$ ;DO ALL
4506 021414 004737 021640 2$: JSR PC,AMOD1 ;GO DO MODE 1(NRZ)
4507 021420 004737 021772 JSR PC,AMOD2 ;GO DO MODE 2(PE)
4508 021424 022737 000007 000550 ASEQ4: CMP #7,DVN ;SEE IF DONE ALL DRIVES
4509 021432 001403 BEQ ASEQX ;IF SO: BR
4510 021434 005237 000550 INC DVN ;BUMP DRIVE NUMBER
4511 021440 000742 BR ASEQ1 ;CONTINUE
4512 021442 005737 000746 ASEQX: TST ASEQCF ;SEE IF CONTINUOUS AUTO SEQ
4513 021446 001335 RNE ASEQ0 ;**B CONTINUE TESTING
4514 021450 000137 004774 JMP TEND

```

```

4516
4517
4518
4519 021454 005037 005042          HRDS: CLR      REOTC          ;CLEAR EOT UNIT CNTR
4520 021460 012777 000040 157032  MOV     #40,@CS        ;INIT
4521 021466 013777 000550 157024  MOV     DVN,@CS        ;SET DRIVE
4522 021474 005777 157010          TST     @C1            ;ACCESS DRIVE
4523 021500 032777 010000 157012  BIT     #10000,@CS     ;TEST FOR NON-EXISTANT DRIVE
4524 021506 001403          BEQ     2$             ;IF DRIVE AVAIL: BR
4525 021510 005726          1$: TST     (SP)+        ;RESET STACK POINTER
4526 021512 000137 021424          JMP     ASEQ4          ;GO SEE IF TRIED ALL DRIVES
4527 021516 017700 157014          2$: MOV     @DT,RO      ;++B GET CONTENTS OF DRIVE TYPE REG
4528 021522 042700 002007          BIC     #2007,RO      ;++B CLEAR SPR AND SPEED BITS
4529 021526 022700 140050          CMP     #140050,RO    ;++B BRANCH IF NOT TMO3 MAGTAPE DRIVE
4530 021532 001366          BNE     1$
4531 021534 005000          CLR     RO
4532 021536 012701 000752          MOV     #UN1,R1       ;SET START OF SLAVE TABLE
4533 021542 005737 003146          TST     CHNFLG        ;BRANCH IF NOT IN CHAIN MODE
4534 021546 001410          BEQ     3$
4535 021550 122737 000006 000041  CMPB   #6,@#41        ;BRANCH IF NOT LOADED VIA TMDP
4536 021556 001004          BNE     3$
4537 021560 005737 000550          TST     DVN           ;BRANCH IF NOT DRIVE 0
4538 021564 001001          BNE     3$
4539 021566 005200          INC     RO
4540 021570 010077 156746          3$: MOV     RO,@TC      ;DO NOT TEST SLAVE 0
4541 021574 032777 010000 156720  BIT     #10000,@DS     ;SELECT SLAVE
4542 021602 001404          BEQ     4$             ;SEE IF SLAVE AVAIL FOR TEST(MOL)
4543 021604 062737 000401 005042  ADD     #401,REOTC     ;IF NOT: BR
4544 021612 010021          MOV     RO,(R1)+       ;INCREMENT UNITS TO TEST COUNT
4545 021614 005200          4$: INC     RO         ;LOAD SLAVE # INTO SLAVE TABLE
4546 021616 022700 000010          CMP     #10,RO        ;STEP TO NEXT SLAVE
4547 021622 001362          BNE     3$             ;BRANCH IF ALL SLAVE NOT DONE
4548 021624 005737 005042          5$: TST     REOTC      ;SEE IF FOUND ANY SLAVES
4549 021630 001727          BEQ     1$             ;IF NOT: BR
4550 021632 012711 177777          MOV     #-1,(R1)      ;TERMINATE SLAVE TABLE
4551 021636 000207          RTS     PC             ;RETURN TO SEQ

```

```

4553
4554                                     ;SUBROUTINE TO SELECT NRZ AUTO TEST MODE*****
4555
4556 021640 005037 000662          AMOD1: CLR      BLCNTR          ;ASSURE BLOCK COUNTER IS 0
4557 021644 012701 000752          MOV      #UN1,R1          ;GET START OF SLAVE TABLE
4558 021650 052721 001700          1$:  BIS      #1700,(R1)+    ;SET ALL SLAVE TO NRZ,NORM,ODD
4559 021654 022711 177777          CMP      #-1,(R1)        ;LOOP UNTIL REACHED END OF TABLE
4560 021660 001373
4561 021662 004737 005056          RNE      1$
4562 021666 012737 000006 000744  JSR      PC,RWANDA        ;GO REWIND ALL AVAIL. SLAVES
4563 021674 013737 000560 000556  MOV      #6,ABL CNT       ;SET NUMBER OF BLOCKS FOR MODE 1
4564 021702 012737 000100 000554  MOV      BUFMAX,FMCNT     ;SET FC = MAX BUFFER SIZE
4565 021710 012737 000001 000566  MOV      #100,RCNT        ;SET REC CNTR = 100
4566 021716 005037 000572          MOV      #1,PATRN        ;SELECT PATTERN 1
4567 021722 005037 000576          CLR      TMEX            ;ASSURE NO TMK
4568 021726 004737 003464          CLR      INTRF           ;ASSURE NORMAL READ
4569 021732 012737 000010 000566  JSR      PC,STAUTO        ;GO DO AUTO MODE 1
4570 021740 004737 003464          MOV      #10,PATRN       ;SELECT PATTERN 10
4571 021744 012737 000014 000566  JSR      PC,STAUTO        ;GO DO PATTERN 10
4572 021752 004737 003464          MOV      #14,PATRN       ;SELECT PATTERN 14
4573 021756 012737 177777 000566  3$:  MOV      #-1,PATRN       ;SELECT AUTO RANDOM DATA
4574 021764 004737 003464          JSR      PC,STAUTO
4575 021770 000207          RTS      PC              ;RETURN TO SEQ

```

4577  
4578  
4579  
4580  
4581  
4582  
4583  
4584  
4585  
4586  
4587  
4588  
4589  
4590  
4591  
4592  
4593  
4594  
4595  
4596  
4597  
4598  
4599  
4600  
4601

021772 005037 000662  
021776 012701 000752  
022002 042711 001700  
022006 052721 002300  
022012 022711 177777  
022016 001371  
022020 004737 005056  
022024 012737 000006 000744  
022032 013737 000560 000556  
022040 012737 000100 000554  
022046 012737 000010 000566  
022054 004737 003464  
022060 012737 000014 000566  
022066 004737 003464  
022072 012737 000015 000566  
022100 004737 003464  
022104 012737 177777 000744  
022112 012737 177777 000566  
022120 004737 003464  
022124 000207

```

;SUBROUTINE TO SELECT PE AUTO TEST MODE*****
AMOD2: CLR BLCNTR ;CLEAR BLOCK CNTR
MOV #UN1,R1 ;SET START OF SLAVE TABLE
1$: BIC #1700,(R1) ;CLEAR NRZ
BIS #2300,(R1) ;SET TO PE NORM, ODD
CMP #-1,(R1) ;LOOP UNTIL END OF TABLE
BNE 1$
JSR PC,RWVDA ;REWIND ALL SLAVES
MOV #6,ABL CNT ;SET AUTO BLOCK COUNT
MOV BUFMAX,FMCNT ;SET FC = MAX BUFFER SIZE
MOV #100,RCNT ;SET REC CNTR TO 100
MOV #10,PATRN ;SELECT PATTERN 10
JSR PC,STAUTO ;GO DO AUTO SEQ
MOV #14,PATRN ;SELECT PATTERN 14
JSR PC,STAUTO
MOV #15,PATRN ;SELECT PATTERN 15
JSR PC,STAUTO
MOV #-1,ABL CNT ;FORCE TO END OF TAPE
MOV #-1,PATRN ;SELECT AUTO RANDOM DATA
JSR PC,STAUTO
3$: RTS PC ;RETURN TO SEQ
```

4603  
4604  
4605  
4606  
4607  
4608  
4609  
4610  
4611  
4612  
4613  
4614  
4615  
4616  
4617  
4618  
4619  
4620  
4621  
4622  
4623  
4624  
4625  
4626  
4627  
4628  
4629  
4630  
4631  
4632  
4633  
4634  
4635  
4636  
4637  
4638  
4639  
4640  
4641  
4642  
4643  
4644  
4645  
4646  
4647  
4648  
4649  
4650  
4651  
4652  
4653  
4654  
4655  
4656  
4657  
4658

022126 000004 025154  
022132 013703 000550  
022136 104400  
022140 000004 024600  
022144 013703 000552  
022150 042703 177770  
022154 104400  
022156 000004 023722  
022162 013703 000552  
022166 000303  
022170 042703 177770  
022174 104400  
022176 000004 025271  
022202 005003  
022204 032737 000010 000552  
022212 001401  
022214 005203  
022216 104400  
022220 000004 025275  
022224 013703 000552  
022230 006003  
022232 006003  
022234 006003  
022236 006003  
022240 042703 177760  
022244 104400  
022246 000004 023765  
022252 005737 000566  
022256 100003  
022260 000004 024055  
022264 000403  
022266 013703 000566  
022272 104400  
022274 000004 024007  
022300 013703 000662  
022304 104400  
022306 000004 024015  
022312 010003  
022314 032737 000010 000700  
022322 001416

```

;*****
;ERROR HEADER PRINT SUBROUTINE:
;
;THIS ROUTINE IS USED TO PRINT OUT A HEADER
;WITH EACH ERROR MESSAGE. THE PRINT IS IN TWO
;LINES AND CONTAINS THE FOLLOWING INFORMATION.
;LINE 1: DRIVE NO. SLAVE NO. DENSITY PARITY FORMAT
;LINE 2: CURRENT BLOCK NUMBER, RECORD NUMBER IN
;WHICH THE ERROR OCCURED PLUS THE TOTAL NUMBER
;OF RECORDS IN THIS BLOCK, THE RECORD SIZE (NUMBER
;OF CHARACTERS), AND THE ERROR TYPE (READ,WRITE, SPACE, ETC)
;PLUS THE TAPE DIRECTION (FORWARD OR REVERSE).
;ALL NUMBERS ARE IN OCTAL.
;*****
PAPRT:  TYPE,MSG52          ;TYPE 'DRIVE # = '
        MOV     DVN,R3
        TYPOCT          ;PRINT DRIVE NUMBER
        TYPE,MSG32       ;TYPE 'SLAVE # = '
        MOV     UDES,R3
        BIC     #177770,R3
        TYPOCT          ;PRINT SLAVE NUMBER
        TYPE,MSG1        ;TYPE DENSITY TAG '*DE'
        MOV     UDES,R3
        SWAB    R3
        BIC     #177770,R3
        TYPOCT          ;PRINT DENSITY
        TYPE,MSG61       ;TYPE PARITY TAG '*P'
        CLR     R3
        BIT     #10,UDES
        BEQ     PAPRTO
        INC     R3
        ;SET PARITY INDICATOR = EVEN
        ;PRINT PARITY BIT STATE
        ;TYPE FORMAT TAG '*F'
PAPRTO:  TYPOCT
        TYPE,MSG62
        MOV     UDES,R3
        ROR     R3
        ROR     R3
        ROR     R3
        ROR     R3
        ;POSITION FORMAT BITS
        BIC     #177760,R3
        TYPOCT
        TYPE,MSG8
        TST     PATRN
        BPL     PAPRTC
        ;TYPE 'R' FOR RANDOM
PAPRTA:  TYPE,MSG17
        BR     PAPRTD
PAPRTC:  MOV     PATRN,R3
        TYPOCT
        ;PRINT PATRN NUMBER
        ;TYPE BLOCK # TAG '*BN'
PAPRTD:  TYPE,MSG15
        MOV     BLCNTR,R3
        TYPOCT
        ;PRINT NUMBER
        TYPE,MSG14
        MOV     RO,R3
        ;TYPE RECORD # TAG '*RN'
        BIT     #10,MIC1
        ;GET # OF RECORDS LEFT TO PROCESS
        BEQ     PAPRT1
        ;SEE IF WRITE OPERATION
        ;IF SO: BR

```



```

4682
4683
4684
4685
4686
4687
4688
4689
4690
4691 022430 063737 000636 000634 RANG: ADD RANSAV,RANBAS
4692 022436 063737 000634 000636 ADD RANBAS,RANSAV ;GET NEW NUMBER
4693 022444 023701 000636 CMP RANSAV,R1 ;SEE IF NUMBER TOO BIG
4694 022450 101367 BHI RANG ;IF SO: BR
4695 022452 020237 000636 CMP R2,RANSAV ;SEE IF NUMBER TOO SMALL
4696 022456 101364 BHI RANG ;IF SO: BR
4697 022460 000207 RTS PC ;EXIT
4698
4699 ;SUBROUTINE TO GET NEW SOFTWARE SWR
4700
4701 022462 022737 000176 000616 GTSWR: CMP #SWREG,SWR ;BRANCH IF SOFTWARE SWR
4702 022470 001025 BNE 1$ ;NOT INVOKED
4703 022472 004737 022546 JSR PC,.SAVE ;SAVE REGISTERS ON THE STACK
4704 022476 000004 023700 TYPE,$MSWR
4705 022502 017703 156110 MOV #SWR,R3 ;GET CURRENT SWR
4706 022506 104400 TYPOCT
4707 022510 000004 023710 TYPE,$MNEW ;REQUEST NEW SWR SETTING
4708 022514 013705 000616 MOV SWR,R5 ;TTR ROUTINE RETURNS VALUE TO (R5)
4709 022520 012701 000007 MOV #7,R1 ;LIMIT RESPONSE TO 7 CHARS
4710 022524 012702 177777 MOV #177777,R2 ;BETWEEN 0 AND 177777
4711 022530 012703 000000 MOV #0,R3
4712 022534 004737 022612 JSR PC,TTR ;GET RESPONSE
4713 022540 004737 022570 JSR PC,.RESTORE ;RESTORE REGISTERS
4714 022544 000207 1$: RTS PC ;RETURN
4715
4716 ;;ROUTINE TO SAVE REGISTERS ON THE STACK
.SAVE: MOV #5,-(SP) ;;R5 IS SAVED AT 12(SP)
MOV #4,-(SP) ;;R4 IS SAVED AT 10(SP)
MOV #3,-(SP) ;;R3 IS SAVED AT 6(SP)
MOV #2,-(SP) ;;R2 IS SAVED AT 4(SP)
MOV #1,-(SP) ;;R1 IS SAVED AT 2(SP)
MOV #0,-(SP) ;;R0 IS SAVED AT (SP)
(1) 022546 010546 MOV 14(SP),-(SP) ;;PUSH RETURN PC ON THE STACK
(1) 022550 010446 RTS PC ;;RETURN TO CALLER
(1) 022552 010546
(1) 022554 010246
(1) 022556 010146
(1) 022560 010046
(1) 022562 016646 000014
(1) 022566 000207
4717 ;;ROUTINE TO RESTORE REGISTERS SAVED ON THE STACK
.RESTORE: MOV (SP)+,14(SP) ;;SAVE RETURN PC ON STACK
MOV (SP)+,#0
MOV (SP)+,#1
MOV (SP)+,#2
MOV (SP)+,#3
MOV (SP)+,#4
MOV (SP)+,#5
(1) 022570 012666 000014 RTS PC ;RETURN
(1) 022574 012600
(1) 022576 012601
(1) 022600 012602
(1) 022602 012603
(1) 022604 012604
(1) 022606 012605
(1) 022610 000207
(1)

```

```

4719 ;*****
4720 ;TTY ENTRY SUBROUTINE;
4721 ;
4722 ;THIS SUBROUTINE IS USED BY THE TEST CONDITION
4723 ;ENTRY ROUTINE TO READ THE RESPONSE ENTERED
4724 ;AT THE TTY AND CHECK THEM FOR LEGALITY AND
4725 ;LIMITS. ALL RESPONSE MUST BE TYPED IN OCTAL
4726 ;(0-7) AND MUST FALL WITHIN THE LIMITS SET BY
4727 ;THE CALLING ROUTINE.
4728 ;IF AN ENTRY IS ILLEGAL OR OUTSIDE THE LIMITS,
4729 ;A QUESTION MARK IS TYPED (?) AND THE RESPONSE
4730 ;MAY BE REENTERED.
4731 ;ENTRIES MAY NOT EXCEED SIX (6) CHARACTERS AND
4732 ;MAY BE TERMINATED AT LESS THAN SIX BY TYPING A
4733 ;CARRIAGE RETURN
4734 ;*****
4735
4736 022612 010146 TTR: MOV R1, -(SP) ;SAVE CHAR COUNT
4737 022614 011601 10$: MOV (SP),R1 ;RESTORE CHAR COUNT (FOR +U)
4738 022616 005037 000652 CLR TEMP1 ;CLEAR FIRST CHARACTER FLAG
4739 022622 005000 CLR R0
4740 022624 004757 023036 1$: JSR PC,TTIM ;GO READ CHARACTER
4741 022630 122737 000003 000650 CMPB #3,TIB ;BRANCH IF NOT +C
4742 022636 001003 BNE 11$
4743 022640 000005 RESET
4744 022642 000137 000200 JMP @#200 ;RESTART AT 200
4745 022646 122737 000015 000650 11$: CMPB #15,TIB ;SEE IF CR
4746 022654 001004 BNE 2$ ;IF NOT: BR
4747 022656 005737 000652 TST TEMP1 ;SEE IF FIRST CHARACTER
4748 022662 001455 BEQ 9$ ;IF SO: BR
4749 022664 000447 BR 6$ ;ELSE GO LOAD VALUE
4750 022666 122737 000025 000650 2$: CMPB #25,TIB ;BRANCH IF NOT CONTROL U
4751 022674 001003 BNE 21$
4752 022676 000004 024355 TYPE,MSG28 ;TYPE <CR><LF>
4753 022702 000744 BR 10$
4754 022704 122737 000177 000650 21$: CMPB #177,TIB ;BRANCH IF NOT 'RUBOUT'
4755 022712 001010 BNE 3$
4756 022714 000241 CLC ;REMOVE LAST CHARACTER
4757 022716 006000 ROR R0
4758 022720 006200 ASR R0
4759 022722 006200 ASR R0
4760 022724 000004 026312 TYPE,MSG118 ;TYPE '\ '
4761 022730 005201 INC R1 ;DEC CHAR RECEIVED COUNT
4762 022732 000734 BR 1$ ;GET NEXT CHARACTER
4763 022734 122737 000060 000650 3$: CMPB #60,TIB ;SEE IF CHAR IS LESS THAN 0
4764 022742 101027 BHI TIBR
4765 022744 122737 000070 000650 4$: CMPB #70,TIB ;SEE IF CHAR IS GREATER THAN 7
4766 022752 101423 BLOS TIBR
4767 022754 005237 000652 5$: INC TEMP1 ;SET FIRST CHARACTER FLAG
4768 022760 006300 ASL R0
4769 022762 006300 ASL R0 ;SHIFT 3 LEFT
4770 022764 006300 ASI R0
4771 022766 042737 177770 000650 BIC #177770,TIB ;STRIP ASCII
4772 022774 053700 000650 BIS TIB,R0 ;LOAD CHARACTER
4773 023000 005301 DEC R1 ;SEE IF DONE
4774 023002 001310 BNE 1$ ;IF NOT: BR

```



4775	023004	020002		6:	CMP	R0,R2		;SEE IF EXCEEDED MAXIMUM LIMIT
4776	023006	101005			BHI	TINER		
4777	023010	020300		7:	CMP	R3,R0		;SEE IF BELOW MINIMUM LIMIT
4778	023012	101003			BHI	TINER		
4779	023014	010015		8:	MOV	R0,(R5)		;LOAD VALUE
4780	023016	005726		9:	TST	(SP)+		;POP CHAR COUNT OFF STACK
4781	023020	000207			RTS	PC		;EXIT
4782								
4783	023022	000004	025046		TINER:	TYPE,MSG43		;TYPE '?'
4784	023026	005726			TST	(SP)+		;POP CHAR COUNT OFF STACK
4785	023030	162716	000020		SUB	0,0,(SP)		;RESET SP TO START OF VALUE ROUTINE
4786	023034	000207			RTS	PC		;REDO VALUE ENTRY

```

4788
4789
4790
4791 023036 005277 155556 TFIN: INC 0TKS
4792 023042 105777 155557 1$: TSTB 0TKS
4793 023046 100375 BPL 1$
4794 023050 017737 155546 000650 MOV 0TKB,TIB
4795 023056 042737 177600 000650 BIC 0177600,TIB ;STRIP PARITY BIT
4796 023064 022737 000015 000650 CMP 015,TIB ;BRANCH IF NOT <CR>
4797 023072 001003 BNE 2$
4798 023074 000004 024355 TYPE,MSG28 ;TYPE '<CR><LF>'
4799 023100 000402 BR 3$
4800 023102 000004 000650 2$: TYPE,TIB ;ECHO CHARACTER
4801 023106 000207 3$: RTS PC
4802
4803
4804 ;TTY OUTPUT SUBROUTINE*****
4805 023110 010446 TTOUT: MOV R4,-(SP) ;SAVE R4 ON THE STACK
4806 023112 010346 MOV R3,-(SP)
4807 023114 017604 000004 MOV 04(SP),R4 ;GET ADDRESS OF MESSAGE TO TYPE
4808 023120 062766 000002 000004 ADD 02,4(SP) ;ADJUST RETURN PC
4809 023126 111437 000646 10$: MOVB (R4),TOB ;GET A CHARACTER
4810 023132 001431 BEQ 3$ ;AND BRANCH IF END OF MSG
4811 023134 122724 000045 CMPB 045,(R4)+ ;BRANCH IF CRLF CHARACTER (*)
4812 023140 001403 BEQ 1$
4813 023142 004737 023224 JSR PC,TOG ;ECHO CHARACTER
4814 023146 000767 BR 10$
4815
4816 023150 112737 000015 000646 1$: MOVB 015,TOB
4817 023156 004737 023224 JSR PC,TOG
4818 023162 012703 000006 MOV 06,R3
4819 023166 005037 000646 2$: CLR TOB
4820 023172 004737 023224 JSR PC,TOG
4821 023176 005303 DEC R3
4822 023200 001372 BNE 2$ ;DO FILLERS
4823 023202 112737 000012 000646 MOVB 012,TOB
4824 023210 004737 023224 JSR PC,TOG
4825 023214 000744 BR 10$
4826 023216 012603 3$: MOV (SP),R3 ;RESTORE REGISTERS
4827 023220 012604 MOV (SP),R4
4828 023222 000002 RTI ;RETURN

```



```

4849                                ;OCTAL OUTPUT SUBROUTINE*****
4850
4851 023326 005037 023502      OCTP:  CLR    OFL                ;CLEAR FLAG FOR LEADING ZERO
4852 023332 010304                MOV    R3,R4                ;SEE IF NUMBER IS ZERO
4853 023334 001003                BNE    1$                  ;IF NOT ZERO: BR
4854 023336 000004 026537      TYPE,DIGIT0
4855 023342 000434                BR     4$                  ;SPACE AND EXIT
4856 023344 100004      1$:    BPL    3$                  ;BRANCH IF MSD IS A '0'
4857 023346 012704 000001      MOV    #1,R4
4858 023352 004737 023442      JSR    PC,OCTPG            ;PRINT 1
4859 023356 006004      3$:    ROR    R4
4860 023360 006004                ROR    R4
4861 023362 006004                ROR    R4                ;POSITION DIGIT
4862 023364 006004                ROR    R4
4863 023366 000304                SWAB   R4
4864 023370 004737 023442      JSR    PC,OCTPG            ;PRINT DIGIT 2
4865 023374 006004                ROR    R4
4866 023376 000304                SWAB   R4
4867 023400 004737 023442      JSR    PC,OCTPG            ;PRINT DIGIT 3
4868 023404 006104                ROL    R4
4869 023406 006104                ROL    R4
4870 023410 000304                SWAB   R4
4871 023412 004737 023442      JSR    PC,OCTPG            ;PRINT DIGIT 4
4872 023416 006004                ROR    R4
4873 023420 006004                ROR    R4
4874 023422 006004                ROR    R4
4875 023424 004737 023442      JSR    PC,OCTPG            ;PRINT DIGIT 5
4876 023430 004737 023442      JSR    PC,OCTPG            ;PRINT DIGIT 6
4877 023434 000004 026535      4$:    TYPE,SPACE          ;TYPE A SPACE
4878 023440 000002                RTI                       ;EXIT
4879
4880 023442 042704 177770      OCTPG: BIC    #177770,R4
4881 023446 001003                BNE    1$
4882 023450 005737 023502      TST    OFL
4883 023454 001410                BEQ    2$
4884 023456 005237 023502      1$:    INC    OFL
4885 023462 052704 000260      BIS    #260,R4
4886 023466 010437 000646      MOV    R4,TOB
4887 023472 004737 023224      JSR    PC,TOG
4888 023476 010304      2$:    MOV    R3,R4
4889 023500 000207                RTS    PC
4890 023502 000000      OFL:  0                    ;FIRST CHAR FLAG
4891
4892
4893                                ;DATA CHARACTER OUTPUT SUBROUTINE*****
4894
4895 023504 012704 000010      DOUT:  MOV    #10,R4        ;SET NUMBER TO PRINT
4896 023510 110346                MOVB   R3, -(SP)          ;GET CHAR TO OUTPUT
4897 023512 106316      1$:    ASLB   (SP)          ;BRANCH IF BIT IS A ZERO
4898 023514 103003                BCC    2$
4899 023516 000004 026541      TYPE,DIGIT1
4900 023522 000402                BR     3$
4901 023524 000004 026537      2$:    TYPE,DIGIT0
4902 023530 005304      3$:    DEC    R4
4903 023532 001367                BNE    1$
4904 023534 005726                TST    (SP)+             ;POP STACK

```

```

4905 023536 000207          RTS      PC
4906
4907 023540 113703 000657    DOUTD:  MOVB   TEMP3+1,R3
4908 023544 004737 023504    JSR     PC,DOUT
4909 023550 013703 000656    MOV     TEMP3,R3
4910 023554 004737 023504    JSR     PC,DOUT
4911 023560 000207          RTS      PC
4912
4913          ;TU16 SERIAL NUMBER PRINT SUBROUTINE*****
4914
4915 023562 017703 154752    SNPT:   MOV     @SN,R3          ;GET CONTENTS OF SERIAL # REG
4916 023566 000004 023775    TYPE,MSG9          ;TYPE SN TAG
4917 023572 010304          MOV     R3,R4
4918 023574 000304          SWAB   R4
4919 023576 006004          ROR    R4
4920 023600 006004          ROR    R4
4921 023602 006004          ROR    R4
4922 023604 006004          ROR    R4
4923 023606 004737 023654    JSR     PC,SNPG     ;PRINT FIRST DIGIT
4924 023612 010304          MOV     R3,R4
4925 023614 000304          SWAB   R4
4926 023616 004737 023654    JSR     PC,SNPG     ;PRINT SECOND DIGIT
4927 023622 010304          MOV     R3,R4
4928 023624 006004          ROR    R4
4929 023626 006004          ROR    R4
4930 023630 006004          ROR    R4
4931 023632 006004          ROR    R4
4932 023634 004737 023654    JSR     PC,SNPG     ;PRINT THIRD DIGIT
4933 023640 010304          MOV     R3,R4
4934 023642 004737 023654    JSR     PC,SNPG     ;PRINT FOURTH DIGIT
4935 023646 000004 024355    TYPE,MSG28        ;TYPE <CR><LF>
4936 023652 000207          RTS     PC          ;EXIT
4937 023654 012737 000260 000646 SNPG:  MOV     #260,TOB    ;SET NUMBER BASE
4938 023662 042704 177760    BIC    #177760,R4  ;MASK NUMBER
4939 023666 050437 000646    BIS    R4,TOB      ;BUILD DIGIT
4940 023672 004737 023224    JSR     PC,TOG      ;GO TYPE
4941 023676 000207          RTS     PC          ;RETURN
4942

```

```

4944
4945                                     ;ERROR MESSAGES*****
4946
4947 023700 051445 051127 036440 $MSWR: .ASCIZ /*SWR - /
      023706 000040
4948 023710 047040 053505 036440 $MNEW: .ASCIZ / NEW = /
      023716 000040
4949 023720 000055          DASH: .ASCIZ /- /
4950 023722 042052 020105 000    MSG1: .ASCIZ /*DE /
4951 023727 045    035507 000040 MSG2: .ASCIZ /*G; /
4952 023734 041045 020073 000    MSG3: .ASCIZ /*B; /
4953 023741 045    047103 000040 MSG4: .ASCIZ /*CN /
4954 023746 053452 020105 000    MSG5: .ASCIZ /*WE /
4955 023753 052    042522 000040 MSG6: .ASCIZ /*RE /
4956 023760 051052 020123 000    MSG7: .ASCIZ /*RS /
4957 023765 052    040520 051124 MSG8: .ASCIZ /*PATRN /
      023772 020116 000
4958 023775 123    035116 000040 MSG9: .ASCIZ /SN: /
4959 024002 051452 020105 000    MSG10: .ASCIZ /*SE /
4960 024007 045    041052 020116 MSG13: .ASCIZ /**BN /
      024014 000
4961 024015 052    047122 000040 MSG14: .ASCIZ /*Rr /
4962 024022 020045 020040 020040 MSG15: .ASCIZ /*          BAD RECORD**/
      024030 020040 020040 041040
      024036 042101 051040 041505
      024044 051117 022504 000045
4963 024052 043040 000    MSG16: .ASCIZ / F /
4964 024055 040    000122    MSG17: .ASCIZ / R /
4965 024060 042440 052117 021440 MSG20: .ASCIZ / EOT # /
      024066 000040
4966 024070 047111 042524 041522 MSG21: .ASCIZ /INTERCHANGE READ? /
      024076 040510 043516 020105
      024104 042522 042101 020077
      024112 000
4967 024113 045    046111 042514 MSG22: .ASCIZ /*ILLEGAL BOT: HALT*/
      024120 040507 020114 047502
      024126 035124 044040 046101
      024134 022524 000
4968 024137 045    051503 020061 MSG23: .ASCIZ /*CS1 /
      024144 000
4969 024145 045    041527 000040 MSG23A: .ASCIZ /*WC /
4970 024152 041045 020101 000    MSG23B: .ASCIZ /*BA /
4971 024157 045    041506 000040 MSG23C: .ASCIZ /*FC /
4972 024164 041445 031123 000040 MSG23D: .ASCIZ /*CS2 /
4973 024172 042045 020123 000    MSG23E: .ASCIZ /*DS /
4974 024177 045    051105 000040 MSG23F: .ASCIZ /*ER /
4975 024204 040445 020123 000    MSG23G: .ASCIZ /*AS /
4976 024211 045    045503 000040 MSG23H: .ASCIZ /*CK /
4977 024216 042045 020102 000    MSG23I: .ASCIZ /*DB /
4978 024223 045    051115 000040 MSG23J: .ASCIZ /*MP /
4979 024230 042045 020124 000    MSG23K: .ASCIZ /*DI /
4980 024235 045    041524 000040 MSG23L: .ASCIZ /*TC /
4981 024242 047045 020117 047111 MSG24: .ASCIZ /*NO INTERRUPT*/
      024250 042524 051122 050125
      024256 022524 000
4982 024261 045    046123 053101 MSG25: .ASCIZ /*SLAVE UNSAFE-TEST DISCONTINUED ON SLAVE*/

```

H( )

	024266	020105	047125	040523			
	024274	042506	052055	051505			
	024302	020124	044504	041523			
	024310	047117	044524	052516			
	024316	042105	047440	020116			
	024324	046123	053101	022505			
	024332	000					
4983	024333	045	051104	050117	MSG26:	.ASCIZ	/DROPS: /
	024340	035123	000040				
4984	024344	050045	041511	051513	MSG27:	.ASCIZ	/PICKS: /
	024352	020072	000				
4985	024355	045	000		MSG28:	.ASCIZ	/ /
4986	024357	045	052045	047515	MSG30:	.ASCIZ	***TM03-TE16/TU77 AUTO SEQUENCE (CZTEDEO)**;..B
	024364	026463	042524	033061			
	024372	052057	033525	020067			
	024400	052501	047524	051440			
	024406	050505	042525	041516			
	024414	020105	041450	052132			
	024422	042105	030105	022451			
	024430	000					
4987	024431	045	052045	030115	MSG31:	.ASCIZ	***TM03-TE16/TU77 DATA RELIABILITY TEST (CZTEDEO)**;..B
	024436	026463	042524	033061			
	024444	052057	033525	020067			
	024452	040504	040524	051040			
	024460	046105	040511	044502			
	024466	044514	054524	052040			
	024474	051505	020124	041450			
	024502	052132	042105	030105			
	024510	022451	000				
4988	024513	124	050131	020105	MSG31A:	.ASCIZ	/TYPE <CR> TO TERMINATE ALL REQUESTS & *C TO RESTART*/
	024520	041474	037122	052040			
	024526	020117	042524	046522			
	024534	047111	052101	020105			
	024542	046101	020114	042522			
	024550	052521	051505	051524			
	024556	023040	057040	020103			
	024564	047524	051040	051505			
	024572	040524	052122	000045			
4989	024600	046123	053101	020105	MSG32:	.ASCIZ	/SLAVE # = /
	024606	020043	020075	000			
4990	024613	104	047105	044523	MSG33:	.ASCIZ	/DENSITY = /
	024620	054524	036440	000040			
4991	024626	040520	044522	054524	MSG34:	.ASCIZ	/PARITY = /
	024634	036440	000040				
4992	024640	042522	047503	042122	MSG35:	.ASCIZ	/RECORD COUNT = /
	024646	041440	052517	052116			
	024654	036440	000040				
4993	024660	044103	051101	041440	MSG36:	.ASCIZ	/CHAR COUNT = /
	024666	052517	052116	036440			
	024674	000040					
4994	024676	040520	052124	051105	MSG37:	.ASCIZ	/PATTERN # = /
	024704	020116	020043	020075			
	024712	000					
4995	024713	123	047111	046107	MSG38:	.ASCIZ	/SINGLE PASS? /
	024720	020105	040520	051523			
	024726	020077	000				

4996	024731	103	041522	041440	MSG39:	.ASCIZ	/CRC CORRECTION (YES=1,NO=0)? /
	024736	051117	042522	052103			
	024744	047511	020116	054450			
	024752	051505	030475	047054			
	024760	036517	024460	020077			
	024766	000					
4997	024767	045	042445	052116	MSG40:	.ASCIZ	/ENTER STALLS READ = /
	024774	051105	051440	040524			
	025002	046114	022523	042522			
	025010	042101	036440	000040			
4998	025016	051127	052111	020105	MSG41:	.ASCIZ	/WRITE = /
	025024	020075	000				
4999	025027	124	051125	020116	MSG42:	.ASCIZ	/TURN AROUND = /
	025034	051101	052517	042116			
	025042	036440	000040				
5000	025046	037445	000045		MSG43:	.ASCIZ	/?#/
5001	025052	042445	052116	051105	MSG44:	.ASCIZ	/ENTER YOZZLE STALL = /
	025060	054440	055117	046132			
	025066	020105	052123	046101			
	025074	020114	020075	000			
5002	025101	045	051105	020122	MSG45:	.ASCIZ	/ERR AMT /
	025106	046501	020124	000			
5003	025113	045	047516	020124	MSG49:	.ASCIZ	/NOT AVAIL /
	025120	053101	044501	020114			
	025126	000					
5004	025127	045	046111	042514	MSG50:	.ASCIZ	/ILLEGAL DRIVE TYPE /
	025134	040507	020114	051104			
	025142	053111	020105	054524			
	025150	042520	000040				
5005	025154	022445			MSG52:	.ASCII	/#/
5006	025156	051104	053111	020105	MSG52A:	.ASCIZ	/DRIVE (TMO3) # = /
	025164	052050	030115	024463			
	025172	021440	036440	000040			
5007	025200	047506	046522	052101	MSG53:	.ASCIZ	/FORMAT = /
	025206	036440	000040				
5008	025212	053452	020105	046524	MSG54:	.ASCIZ	/WE TM/
	025220	000					
5009	025221	052	042523	052040	MSG55:	.ASCIZ	/SE TM/
	025226	000115					
5010	025230	052040	000115		MSG56:	.ASCIZ	/ TM/
5011	025234	047045	047117	042455	MSG57:	.ASCIZ	/NON-EXIST SLAVE/
	025242	044530	052123	051440			
	025250	040514	042526	000			
5012	025255	045	051103	020103	MSG58:	.ASCIZ	/CRC /
	025262	000					
5013	025263	045	051114	020103	MSG59:	.ASCIZ	/RC /
	025270	000					
5014	025271	052	020120	000	MSG61:	.ASCIZ	/P /
5015	025275	052	020106	000	MSG62:	.ASCIZ	/F /
5016	025301	045	047452	044522	MSG64:	.ASCIZ	/ORIGINAL ERROR/
	025306	044507	040516	020114			
	025314	051105	047522	025122			
	025322	000					
5017	025323	045	042522	051124	MSG65:	.ASCIZ	/RETRY: /
	025330	035131	000040				
5018	025334	051452	020105	052122	MSG66:	.ASCIZ	/SE RTR: /



5019 025342 054522 000040  
025346 042452 040522 042523 MSG67: .ASCIZ /\*ERASE/  
025354 000  
5020 025355 045 042522 042522 MSG68: .ASCIZ /\*REREV: /  
025362 035126 000040  
5021 025366 040524 042520 046440 MSG69: .ASCIZ /\*TAPE MARK? /  
025374 051101 037513 000040  
5022 025402 047045 047117 042455 MSG71: .ASCIZ /\*NON-EXIST DRIVE/  
025410 044530 052123 042040  
025416 044522 042526 000  
5023 025423 045 042522 053506 MSG72: .ASCIZ /\*REFWD: /  
025430 035104 000040  
5024 025434 053445 042524 051122 MSG73: .ASCIZ /\*WTFRR: /  
025442 020072 000  
5025 025445 045 042522 044507 MSG74: .ASCIZ /\*REGISTER START = /  
025452 052123 051105 051440  
025460 040524 052122 036440  
025466 000040  
5026 025470 042526 052103 051117 MSG75: .ASCIZ /\*VECTOR ADRS = /  
025476 040440 051104 020123  
025504 020075 000  
5027 025507 045 042504 042522 MSG76: .ASCIZ /\*DEREV: /  
025514 035126 000040  
5028 025520 042045 043105 042127 MSG77: .ASCIZ /\*REFWD: /  
025526 020072 000  
5029 025531 045 047516 026516 MSG78: .ASCIZ /\*NON-RETRYABLE WRITE ERROR; ER /  
025536 042522 051124 040531  
025544 046102 020105 051127  
025552 052111 020105 051105  
025560 047522 035122 042440  
025566 020122 000  
5030 025571 045 047516 026516 MSG79: .ASCIZ /\*NON-RETRYABLE READ ERROR; ER /  
025576 042522 051124 040531  
025604 046102 020105 042522  
025612 042101 042440 051122  
025620 051117 020072 051105  
025626 000040  
5031 025630 042445 042116 047440 MSG100: .ASCIZ /\*END OF PASS \*/  
025636 020106 040520 051523  
025644 022440 000  
5032 025647 045 025045 025052 MSG101: .ASCIZ /\*\*\*\*\*\*  
025654 025052 025052 025052  
025662 025052 025052 025052  
025670 025052 025052 022452  
025676 000  
5033 025677 101 052125 020117 MSG104: .ASCIZ /\*AUTO CONT.? /  
025704 047503 052116 037456  
025712 000040  
5034 025714 051045 041505 053117 MSG105: .ASCIZ /\*RECOVERED/  
025722 051105 042105 000  
5035 025727 052 040502 020104 MSG106: .ASCIZ /\*BAD TAPE OVERFLOW/  
025734 040524 042520 047440  
025742 042526 043122 047514  
025750 000127  
5036 025752 051045 053505 047111 MSG16A: .ASCIZ /\*REWIND TAPE; RESTART AT BLOCK 1/  
025760 020104 040524 042520

	025766	020073	042522	052123	
	025774	051101	020124	052101	
	026002	041040	047514	045503	
	026010	030440	000		
5037	026013	045	047125	042522	MSG107: .ASCII /*UNRECOVERABLE BAD SPOT/
	026020	047503	042526	040522	
	026026	046102	020105	040502	
5038	026034	020104	050123	052117	.ASCIZ /*BAD RECORD LEFT ON TAPE*/
	026042	041045	042101	051040	
	026050	041505	051117	020104	
	026056	042514	052106	047440	
	026064	020116	040524	042520	
	026072	000045			
5039	026074	050052	051517	052111	MSG109: .ASCIZ /*POSITION LOST IN RETRY/
	026102	047511	020116	047514	
	026110	052123	044440	020116	
	026116	042522	051124	000131	
5040	026124	051445	051525	042520	MSG110: .ASCIZ /*SUSPECT BAD TAPE/
	026132	052103	041040	042101	
	026140	052040	050101	000105	
5041	026146	051045	050105	040505	MSG111: .ASCIZ /*REPEAT: /
	026154	035124	000040		
5042	026160	041040	042101	052040	MSG112: .ASCIZ / BAD TAPE SPOTS*/
	026166	050101	020105	050123	
	026174	052117	022523	000	
5043					
5044	026201	045	051440	043117	MSG113: .ASCIZ /* SOFT: /
	026206	035124	000040		
5045					
5046	026212	020045	040510	042122	MSG114: .ASCIZ /* HARD: /
	026220	020072	000		
5047					
5048	026223	045	040510	042122	MSG115: .ASCIZ /*HARD READ ERROR/
	026230	051040	040505	020104	
	026236	051105	047522	000122	
5049	026244	051445	040514	042526	MSG116: .ASCIZ /*SLAVE REWINDING; WILL RESTART AT BOT/
	026252	051040	053505	047111	
	026260	044504	043516	020072	
	026266	044527	046114	051040	
	026274	051505	040524	052122	
	026302	040440	020124	047502	
	026310	000124			
5050	026312	000134			MSG118: .ASCIZ /\
5051	026314	051045	046505	053117	MSG120: .ASCIZ /*REMOVE TMDP FROM SLAVE TO BE TESTED*/
	026322	020105	046524	050104	
	026330	043040	047522	020115	
	026336	046123	053101	020105	
	026344	047524	041040	020105	
	026352	042524	052123	042105	
	026360	000045			
5052	026362	044045	051101	053504	MSG121: .ASCIZ /*HARDWARE SWR IN USE*/
	026370	051101	020105	053523	
	026376	020122	047111	052440	
	026404	042523	000045		
5053	026410	047516	051440	040514	MSG122: .ASCIZ /NO SLAVES LEFT TO TEST; HALT*/
	026416	042526	020123	042514	

CZTEDEO 1M03 TE16/TU77 DRI  
CZTEDE.P11 07-MAR-84 14:04

MACY11 30(1046) 07-MAR-84 14:21 PAGE 70-5

SEQ 0115

	026424	052106	052040	020117	
	026432	042524	052123	020072	
	026440	040510	052114	000045	
5054	026446	040445	052125	026517	MSG123: .ASCIZ /*AUTO SEQ: TEST WILL RESTART*/
	026454	042523	035121	052040	
	026462	051505	020124	044527	
	026470	046114	051040	051505	
	026476	040524	052122	000045	
5055	026504	041445	051117	042522	MSG124: .ASCIZ /*CORRECTED PE DATA ERROR/
	026512	052103	042105	050040	
	026520	020105	040504	040524	
	026526	042440	051122	051117	
	026534	000			
5056	026535	040	000		SPACE: .ASCIZ ' '
5057	026537	060	000		DIGIT0: .ASCIZ '0'
5058	026541	061	000		DIGIT1: .ASCIZ '1'
5059					
5060		026544			
5061	026544	036544			BUFBEQ: .EVEN ;READ AND WRITE BUFFER AREA
5062		000001			.END





CZTEDFO TMO3 TE 16 TU?? DRT  
 CZTEDE.P1: 07-MAR-84 14:04

MACY11 30(1046) 07 MAR-84 14:21 PAGE 71 P  
 CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0118

DAT10	014250	1856	35190					
DAT11	014304	1857	35310					
DAT12	014324	1858	35410					
DAT13	014346	1859	35510					
DAT14	014356	1860	35560					
DAT15	014406	1861	35720					
DAT2	014152	1850	34740					
DAT3	014156	1851	34790					
DAT3A	014164	34810	3492					
DAT4	014202	1852	34900					
DAT5	014212	1853	34970					
DAT6	014220	1854	35020					
DAT7	014226	1855	35070					
DB	000532	15370						
DCHK	015064	2757	2951	37060				
DCHKO	015112	3710	37120					
DEREV1	001174	17200	2062	37920				
DEREX	016070	3860	3878	3800	3888	3895	3898	39000
DEREX1	016122	3901	3904	3906	39080			
DERFL	000712	16000	37070	3783	39090			
DERR	015470	3776	38240					
DERR0	015500	38260	3907					
DERR0A	015526	3828	38320					
DERR0B	015554	3837	38400					
DERR0C	015574	3843	38460					
DERR0D	015576	3845	38470					
DERR1	015620	3850	38530					
DERR2	015622	3852	38540					
DERR3	015634	38570						
DERR4	015636	3825	3856	38580				
DERR4A	015762	3872	38810					
DERR4B	016024	3867	38820					
DERR5	016052	3892	38960					
DERR6	016064	3869	3890	38990				
DEX	015466	3784	3786	3791	37930			
DFC	015364	3732	37710	3780				
DF0A	015260	3742	37440	3781				
DF0AC	015302	3748	37500					
DF0A1	015316	3753	37550					
DF0A2	015332	3758	37600					
DF0A3	015346	3763	37650					
DF0A4	015352	3745	37670					
DF0B	015220	37330						
DF0B0	015242	3736	37390					
DF0C	015202	3725	37290					
DF0C0	015212	3715	3717	3719	37310			
DF0D	015166	3721	37260					
DF0E	015160	37230	3728					
DF0F	015152	37200	3724					
DF1	015376	3768	3772	37750				
DF2	015406	3770	3774	37770				
DF3	015422	3778	37820					
DF4	015462	3789	37920					
DIGIT0	020537	4854	4901	50570				
DIGIT1	020541	4899	50580					
LOUT	023504	3847	3854	48950	4908	4910		

M10

CZTEDEO IM03-TE16/1077 DRT  
CZTEDE.P11 07-MAR-84 14:04

MACY11 30(1046) 07-MAR-84 14:21 PAGE 72  
CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0129

\$CHAIN	1365#	1910
\$RESTO	1365#	4717
\$SAVE	1365#	4716
.\$ACT1	1365#	1485
.\$EOP	1365#	2172

. ABS. 036544 000

ERRORS DETECTED: 0

CZTEDE.BIN,CZTEDE.LST/CRF/NL;TOC=CZTEAE.SML/ML,CZTEDE.P11  
RUN-TIME: 5 10 1 SECONDS  
RUN-TIME RATIO: 46/17=2.6  
CORE USED: 14K (28 PAGES)

