

RK611/RK06

RK611/RK06 PERFORMANCE
MD-11-DZR6P-B
EXERCISER

EP DZR6P B DL
COPYRIGHT 1976

JAN 1978
digital
FICHE 1 OF 2
MADE IN USA

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
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	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	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	688	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	743	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	791	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	842	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	899	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	932	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	979	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	1026	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	1072	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	1121	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	1157	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	1208	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	1251	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	1285	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	1334	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	1391	1392	1393	1394	1395
1396	1397	1398	1399	1400	1401	1402	1403	1404	1405	1406	1407	1408	1409	1410
1411	1412	1413	1414	1415	1416	1417	1418	1419	1420	1421	1422	1423	1424	1425
1426	1427	1428	1429	1430	1431	14								

RK611/RK06

RK611/RK06 PERFORMANCE
MD-11-DZR6P-B
EXERCISER

EP DZR6P B DL

COPYRIGHT 1976

FICHE 2 OF 2

JAN 1978

digital

MADE IN USA



IDENTIFICATION

SEQ 0001

PRODUCT CODE: MAINDEC-11-0ZR6P-B
PRODUCT NAME: RK6.1/RK06 PERFORMANCE EXERCISER
DATE: AUG, 1976
MAINTAINER: DIAGNOSTIC GROUP
AUTHOR: ROY SPITZER

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

THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED UNDER A LICENSE AND MAY ONLY BE USED OR COPIED IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE.

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

COPYRIGHT (C) 1976 BY DIGITAL EQUIPMENT CORPORATION

TABLE OF CONTENTS

- 1.0 ABSTRACT
- 2.0 REQUIREMENTS
 - 2.1 EQUIPMENT
 - 2.2 PRELIMINARY PREREQUISITS
- 3.0 OPERATING PROCEDURES
 - 3.1 LOADING PROCEDURES
 - 3.2 STARTING PROCEDURE
 - 3.3 SYSTEM PARAMETERS
 - 3.4 SWITCHES
 - 3.4.1 SWITCH 15
 - 3.4.2 SWITCH 14
 - 3.4.3 SWITCH 13
 - 3.4.4 SWITCH 12
 - 3.4.5 SWITCH 11
 - 3.4.6 SWITCH 10
 - 3.4.7 SWITCH 9
 - 3.4.8 SWITCH 8
 - 3.4.9 SWITCH 7
 - 3.4.10 SWITCH 6
 - 3.4.11 SWITCH 5
 - 3.4.12 SWITCH 4
 - 3.4.13 SWITCH 3
 - 3.4.14 SWITCH 2
 - 3.4.15 SWITCH 1
 - 3.4.16 SWITCH 0
 - 3.5 RUN TIME
- 4.0 OPERATING PROCEDURE
 - 4.1 OPERATOR COMMANDS
 - 4.2 DRIVE PARAMETER (PER EACH DRIVE)
 - 4.2.1 FORMS FOR PACK AREA EXCLUSIONS
 - 4.3 DIAGNOSTIC DUMP
- 5.0 PROGRAM DESCRIPTION
- 6.0 PRINT OUTS
 - 6.1 PERFORMANCE SUMMARY TYPEOUT
 - 6.2 ERROR REPORTS
- APPENDIX A - DATA WRITTEN ON PACKS

1.0 ABSTRACT

THE RK06 PERFORMANCE EXERCISER PROGRAM WILL EXERCISE IN A RANDOM OVERLAPPED MANNER 1 TO 8 RK06 DISK DRIVES ATTACHED TO THE SAME RK06 UNIBUS CONTROLLER IN A DEDICATED STAND-ALONE MODE.

DRIVES UNDER TEST CAN BE ADDED TO OR DROPPED FROM THE TESTING SEQUENCE BY OPERATOR COMMAND.

AT ANY GIVEN POINT IN TIME, THE NEXT DRIVE ON WHICH AN OPERATION IS TO BE INITIATED IS CHOSEN RANDOMLY FROM AMONG THE RK06 DRIVES UNDER TEST NOT HAVING ANY ERROR CONDITIONS PRESENT AND NOT CURRENTLY UNDERGOING ERROR RECOVERY. THE COMMAND IS THEN CHOSEN RANDOMLY FROM THE FOLLOWING SET OF COMMANDS:

READ
WRITE
WRITE FOLLOWED BY WRITE CHECK

THEN THE CYLINDER, TRACK, SECTOR, WORD COUNT, AND DATA ARE DETERMINED RANDOMLY. THE DATA WRITTEN IS RANDOMLY SELECTED FROM A SET OF FIXED DATA PATTERNS. TO OPTIMIZE THE THROUGHPUT ON THE RK06 CONTROLLER EACH DATA TRANSFER COMMAND IS TRANSLATED TO AN EXPLICIT SEEK COMMAND FOLLOWED BY THE DATA TRANSFER COMMAND.

DRIVE ERRORS ARE QUEUED UP AS THEY OCCUR. WHILE ERROR RECOVERY IS BEING PROCESSED ON ONE DRIVE ALL OTHER DRIVES HAVING NO ERRORS WILL BE EXECUTING ORDERS GENERATED RANDOMLY AS DESCRIBED ABOVE. EACH DRIVE ERROR IS PROCESSED ON A FIRST-IN-FIRST-OUT BASIS. THIS MEANS THAT A DRIVE ERROR WILL BE REPORTED AND ERROR RECOVERY COMPLETELY PROCESSED BEFORE THE NEXT DRIVE ERROR IS REPORTED AND ERROR PROCESSING HAS BEGUN. ERROR RECOVERY IS PROCESSED AS DESCRIBED IN THE RK06 DISK DRIVE SPECIFICATION.

REPORTING OF SYSTEM ERRORS SUCH AS CONTROLLER PROBLEMS WILL BE IMMEDIATE UPON OCCURRENCE, NOT DEFERRED, PERHAPS CAUSING PREMATURE PROGRAM TERMINATION.

PERFORMANCE STATISTICS ARE KEPT ON EACH DRIVE. THESE STATISTICS INCLUDE BOTH OPERATION COUNTS AND ERROR SUMMARY INFORMATION. (SEE SECTION 6.1) THESE STATISTICS WILL BE INITIALIZED WHEN TESTING BEGINS ON A DRIVE. AT ANY TIME AFTER TESTING BEGINS ON THE DRIVE(S) UNDER PERFORMANCE EVALUATION, THE OPERATOR CAN DEMAND THESE PERFORMANCE STATISTICS. IF A REAL TIME CLOCK IS AVAILABLE, THE OPERATOR HAS THE OPTION OF HAVING THE PROGRAM GIVE PERIODIC PERFORMANCE SUMMARIES.

ONCE A PARTICULAR DRIVE HAS AN UNRECOVERABLE ERROR (OTHER THAN A DATA TRANSFER AND SEEK INCOMPLETE), THAT DRIVE WILL BE AUTOMATICALLY DEASSIGNED.

ONCE A PARTICULAR DRIVE HAS EXCEEDED AN ERROR THRESHOLD (DATA TRANSFER OR SEEK) SPECIFIED FOR A DRIVE, THAT DRIVE WILL BE AUTOMATICALLY DEASSIGNED UNLESS SWITCH 4 IS SET.

ONCE A PARTICULAR DRIVE HAS EXCEEDED THE MAXIMUM NUMBER OF PERMITTED OPERATIONS FOR THAT DRIVE, THE DRIVE WILL BE AUTOMATICALLY DESELECTED UNLESS SWITCH 5 IS SET.

2.0 REQUIREMENTS

2.1 EQUIPMENT

- PDP-11 WITH AT LEAST 16K OF MEMORY
- CONSOLE TERMINAL
- DECTAPE, PAPER TAPE READER; OR DECDISK
- RK611 CONTROLLER
- 1 TO 8 RK06 DISK DRIVE WITH FORMATTED PACK (16 BIT FORMAT)
- REAL TIME CLOCK (KW11-P OR KW11-L OPTIONAL) FOR INTERVAL PERFORMANCE REPORTING AND TIME ASSOCIATIONS WITH ERROR REPORTS

2.2 PRELIMINARY PROGRAMS

THE RK06 SUBSYSTEM IS ASSUMED TO BE BASICALLY OPERATIONAL AND FREE OF HARD FAULTS. THE FOLLOWING RK06 PROGRAMS ARE ASSUMED TO BE RUNNING WITHOUT ERROR BEFORE THIS PROGRAM IS RUN:

- RK611 DISKLESS CONTROLLER DIAGNOSTICS
- RK06 DRIVE DIAGNOSTICS
- RK611 FUNCTIONAL CONTROLLER DIAGNOSTIC
- RK06 SUBSYSTEM VERIFICATION
- RK06 PACK FORMATTER (IF PACKS ARE UNFORMATTED)

3.0 OPERATING PROCEDURE

3.1 LOADING PROCEDURE

THE PROGRAM CAN BE LOADED FROM PAPER TAPE USING ABSOLUTE LOADER OR FROM ANY MEDIA SUPPORTED BY XXDP.

3.2 STARTING PROCEDURES

AFTER THE PROGRAM IS LOADED, THE OPERATOR STARTS THE PROGRAM AT ONE OF THE FOLLOWING LOCATIONS:

LOCATION	FUNCTION
200	INITIALIZE ALL DRIVE STATISTIC TABLES, THE RANDOM NUMBER SEED, AND USE DEFAULT SYSTEM PARAMETERS. THE PROGRAM WILL THEN TYPE READY TO BEGIN PERFORMANCE TESTING.
204	RESTART PROGRAM. IF DRIVE IS UNDER TEST, CONTINUE TESTING DRIVE AND DO NOT DESTROY STATISTICS. IF PACK IS BEING WRITTEN, START WRITING PACK FROM THE BEGINNING. THE PROGRAM WILL THEN TYPE READY TO BEGIN PERFORMANCE TESTING.
214	START PROGRAM AND ALTER SYSTEM PARAMETERS. THE PROGRAM WILL THEN IDENTIFY ITSELF AND ASK FOR THE PARAMETERS AS DESCRIBED IN SECTION 3.3.2. THE PROGRAM WILL THEN TYPE READY TO BEGIN PERFORMANCE TESTING.

NOTE: IF A POWER FAIL OCCURS, THE PROGRAM WILL AUTOMATICALLY PERFORM THE RESTART FUNCTION.

THE PROGRAM IS NOW IN AN IDLE LOOP WAITING FOR THE INITIATION OF RK06 DRIVE TESTING.

THE OPERATOR THEN STARTS PERFORMANCE EVALUATION OF THE RK06 DRIVE(S) BY TYPING A CONTROL-C <C> FOLLOWED BY A TM OR PN AS DESCRIBED IN SECTION 4.1. IF THE PACK DOES NOT HAVE THE PROPER RANDOMLY CHOSEN

PREDETERMINED TEST PATTERNS, THE OPERATOR MUST USE THE WM COMMAND AS DESCRIBED IN SECTION 4.1 BEFORE INITIATING THE PERFORMANCE EVALUATION SEQUENCE ON THAT DRIVE.

3.3 SYSTEM PARAMETERS

3.3.1 DIRECT MEMORY ALTERATION

AS PART OF THE PROGRAM START-UP PROCEDURE, THE FOLLOWING SYSTEM PARAMETERS WILL BE LOADED BY THE OPERATOR OR DEFAULTED BY THE PROGRAM. THE OPERATOR MUST PHYSICALLY ALTER THE APPROPRIATE MEMORY LOCATIONS TO CHANGE THESE VALUES.

PARAMETER	TAG	DEFAULT VALUE
- KW11-P STATUS REGISTER ADDRESS	SLKCSR	172540
- KW11-P COUNTER BUFFER ADDRESS	SLKCSB	172542
- KW11-P VECTOR ADDRESS	SLPVEC	104
- KW11-L STATUS REGISTER ADDRESS	SLKS	177506
- KW11-L VECTOR ADDRESS	SLLVEC	100
- SYSTEM POWER (HERTZ)	HZ	60(DECIMAL)

3.3.2 PROGRAM MODIFIED PARAMETERS

THE FOLLOWING PARAMETERS ARE MODIFIED BY A 214 PROGRAM START.

A <CR> WILL DEFAULT CURRENT PARAMETERS. A CONTROL Z <^Z> <CR> WILL DEFAULT ALL THE REST OF THE PARAMETERS. A CONTROL C <^C> <CR> WILL RETURN TO FIRST PARAMETER ENTRY REQUEST FOR THIS DRIVE. A ROBOU CANCELS THE LAST CHARACTER TYPED. A CONTROL U <^U> CANCELS LINE BEING TYPED. ALL ENTRIES MUST BE TERMINATED WITH CARRIAGE RETURN.

PARAMETER -----	DEFAULT VALUE -----
- CPU IDENTIFIER (OCTAL NUMBER 0-377 FOR DUAL ACCESS PORT DISCRIMINATION)	0
- OVERLAY OF LOADER BY PROGRAM? (ALLOWS GREATER EXPANSION OF MEMORY FOR BUFFER SPACE IF LOADER IS OVERLAYED)	NO
- MAXIMUM DATA BUFFER (SEE NOTE) UP TO 5888 (23 SECTORS)	SEE NOTE
- NUMBER OF DATA WORDS COMPARED PER READ (0 - MAXIMUM BUFFER LENGTH) EACH READ COMMAND ISSUED BY THE PROGRAM HAS AN IMPLICIT DATA COMPARISON ASSOCIATED WITH IT VERIFYING THAT THE DATA READ IS ONE OF LEGAL RANDOM PATTERNS.	2 WORDS
- INTERVAL BETWEEN AUTOMATIC PERFORMANCE TYPE-OUTS (EVERY 1-255 MINUTES OR NO AUTOMATIC PERFORMANCE TYPE-OUTS)	NO AUTOMATIC PERFORMANCE TYPE-OUTS
- RK611/RK06 UNIBUS ADDRESS	177440
- RK611/RK06 VECTOR ADDRESS	210
- RK611/RK06 PRIORITY	5

NOTE

THE PROGRAM WILL SELECT A MAXIMUM BUFFER SIZE WHICH IS DETERMINED BY THE MEMORY AVAILABLE. THE MAXIMUM BUFFER SIZE ASSIGNED BY THE PROGRAM WILL BE 5888 WORDS. THE DEFAULT VALUE OF THIS PARAMETER WEIGHTS THE OPERATIONS TOWARDS MAXIMIZING DATA TRANSFERS PER UNIT OF TIME. BY CHANGING THIS PARAMETER TO 1 SECTOR (256 WORDS), THE OPERATIONS WILL BE WEIGHTED TOWARDS MAXIMIZING SEFKS PER UNIT OF TIME.

3.4 SWITCHES

3.4.1 SWITCH 15

SW<15> = 1 HALT ON ERROR

IF THIS SWITCH IS SET, THE PROGRAM WILL NOT ISSUE ANOTHER COMMAND TO THE PK06 SUBSYSTEM WHEN AN ERROR OCCURS AND THE PROGRAM WILL HALT IN THE SYSMAC ERROR HANDLER. IF THE OPERATOR DESIRES TO CONTINUE FROM WHERE THE PROGRAM LEFT OFF, THE OPERATOR SIMPLY PRESSES THE CONTINUE SWITCH ON THE CONSOLE PANEL AND THE PROGRAM CONTINUES WITH THE NEXT RANDOMLY GENERATED COMMAND.

3.4.2 SWITCH 14

SW<14> = 0 NORMAL OPERATION

THE OPERATION THE PROGRAM IN REGARD TO DRIVE SELECTION AND COMMAND GENERATION IS NORMAL. NORMAL OPERATION IS TO RANDOMLY SELECT A DRIVE, STORE THE PREVIOUS OPERATION EXECUTED ON THAT DRIVE, AND GENERATE A NEW RANDOM OPERATION TO BE EXECUTED ON THAT DRIVE. THE GENERATED OPERATION INCLUDES RANDOMLY GENERATED VALUES FOR THE FOLLOWING PARAMETERS:

- DRIVE COMMAND
- CYLINDER ADDRESS
- TRACK ADDRESS
- SECTOR ADDRESS
- WORD COUNT
- DATA PATTERN

THESE VALUES ARE STORED AS THE PREVIOUS OPERATION WHEN A NEW OPERATION IS GENERATED. IT IS IMPORTANT TO NOT THAT AT ANY POINT IN TIME THERE IS BOTH A CURRENT OPERATION AND A STORED PREVIOUS OPERATION ASSOCIATED WITH EACH DRIVE UNDER TEST. THE CURRENT OPERATION MAY HAVE ALREADY BEEN EXECUTED AND A NEW COMMAND HAS NOT YET BEEN GENERATED OR THE COMMAND IS WAITING TO BE EXECUTED.

SW<14> = 1 LOOP ON CURRENT OPERATIONS (TEST)

DO NOT GENERATE NEW RANDOM OPERATION BUT CONTINUE TO EXECUTE CURRENT OPERATIONS.

WHEN SWITCH 14 IS SET THE GENERATION OF A NEW OPERATION IS INHIBITED AND THE PROGRAM OPERATES AS FOLLOWS:

- 1.) A DRIVE IS RANDOMLY SELECTED.
- 2.) A SEEK IS EXECUTED ON THE DRIVE TO THE CYLINDER ADDRESS STORED IN THAT DRIVE'S PREVIOUS OPERATION.
- 3.) THE CURRENT OPERATION FOR THAT DRIVE IS EXECUTED.
- 4.) REPEAT THE ENTIRE PROCESS.

USEAGE

THIS SWITCH CAN BE USED IN CONJUNCTION WITH SWITCH 15 TO ENABLE THE REPTITION OF A RANDOMLY GENERATED OPERATION. THE OPERATOR WOULD START WITH SWITCH 15 SET AND SWITCH 14 RESET. WHEN THE PROGRAM HAS INDICATED THAT AN ERROR HAS OCCURRED BY HALTING. THE CURRENT OPERATION FOR THE DRIVE IN FRPOP WILL NOT HAVE BEEN MODIFIED WHEN THE PROGRAM HALTS. THE OPERATOR CAN THEN CAUSE ALL DRIVES TO LOOP ON THEIR INDIVIDUAL CURRENT OPERATIONS BY SETTING SWITCH 14, RESETING SWITCH 15, AND HITTING CONTINUE.

TO HELP ISOLATE PROBLEMS, THE OPERATOR CAN DROP THE RK06 DRIVE(S) FROM THE TEST SEQUENCE AT ANY TIME BY USING THE DN COMMAND AS DESCRIBED IN SECTION 4.1.

3.4.3 SWITCH 13 -----

SW<13> = 1 INHIBIT ERROR TYPE OUT

3.4.4 SWITCH 12 -----

NO EFFECT.

3.4.5 SWITCH 11

NO EFFECT.

3.4.6 SWITCH 10

SW<10> = 1 RING TELETYPE BELL IF ERROR

3.4.7 SWITCH 9

NO EFFECT.

3.4.8 SWITCH 8

NO EFFECT.

3.4.9 SWITCH 7

NO EFFECT.

3.4.10 SWITCH 6
-----SW<6> = 1 INHIBIT AUTOMATIC DESELECT FOR CLEARABLE UNSAFES,
SPEED LOSS, AND AC LOW.3.4.11 SWITCH 5
-----SW<5> = 1 INHIBIT AUTOMATIC DRIVE REASSIGNMENT IF OPERATION
COUNT THRESHOLD IS EXCEEDED.

3.4.12 SWITCH 4

SW<4> = 1 INHIBIT AUTOMATIC DRIVE REASSIGNMENT IF ERROR
THRESHOLD EXCEEDED.

3.4.13 SWITCH 3

SW<3> = 1 DISPLAY ENTIRE SECTOR READ BEFORE STARTING RETRY
SEQUENCE

ALL HARDWARE DETECTED DATA ERRORS WILL BE RETRIED
ON A SECTOR BASIS. IF THIS SWITCH IS SET, THE
DATA SECTOR IN ERROR WILL BE DISPLAYED ENTIRELY
BEFORE THE RETRY SEQUENCE BEGINS ON THE SECTOR IN
ERROR.

3.4.14 SWITCH 2

NO EFFECT.

3.4.15 SWITCH 1

SW<1> = 1 INHIBIT SOFTWARE DATA COMPARISONS

THE IMPLICIT DATA COMPARISONS ASSOCIATED WITH EACH
READ WILL NOT BE PERFORMED WHEN THIS SWITCH IS
SET. THE ONLY DATA CHECKING PERFORMED WILL BE
HARDWARE DATA CHECKING.

3.4.16 SWITCH 0

NO EFFECT.

3.5 RUN TIME

THIS PROGRAM IS DESIGNED TO RUN FOR AN EXTENDED PERIOD OF TIME.
(ESTIMATED RUN TIME WITH PRESENT DEFAULT PARAMETERS IS APPROXIMATELY
250 HOURS PER DRIVE.) MEANINGFUL INFORMATION CAN STILL BE DERIVED BY
RUNNING THIS PROGRAM FOR AN HOUR OR LESS SINCE STATISTICS ARE TAKEN AS
THE PROGRAM IS RUNNING.

4.0 OPERATING PROCEDURE

4.1 OPERATOR COMMANDS

AFTER THE PROGRAM IS LOADED AND HAS PRINTED OUT "READY TO BEGIN PERFORMANCE TESTING", THE OPERATOR CAN ONLY TYPE A CONTROL-G <G> OR A CONTROL-C <C>.

A CONTROL-G <G> IS USED FOR SOFTWARE SWITCH REGISTER MODIFICATION.

UPON RECEIVING A CONTROL-C <C> THE PROGRAM WILL RESPOND WITH "PLEASE TYPE COMMAND". THE FOLLOWING TWO CHARACTER COMMANDS FOLLOWED BY CARRIAGE RETURN WILL BE THE ONLY COMMANDS ACCEPTED: TN, PN, DN, SN, WN.

THE SECOND CHARACTER N=0,1,...,7 OR A DESIGNATES WHICH DRIVE IS REFERENCED BY THE COMMAND. IF N=A, ALL ADDRESSABLE DRIVES ARE REFERENCED. THE DESCRIPTION OF EACH OPERATOR COMMAND FOLLOWS:

- TN INITIATE TESTING OF DRIVE N AND USE PREVIOUS PARAMETERS. (DEFAULT PARAMETERS ARE USED IF PN COMMAND HAS NOT PREVIOUSLY BEEN ISSUED TO THE DRIVE(S) AT THE BEGINNING OF THEIR TEST SEQUENCE.) IF N=A, INITIATE TESTING ON ALL DRIVES.

- PN THIS COMMAND HAS TWO USES:

1.) CHANGE CURRENT DRIVE PARAMETERS AND INITIATE TESTING ON DRIVE N.

2.) CHANGE CURRENT DRIVE PARAMETERS AND CONTINUE TESTING ON DRIVE N.

IN BOTH CASES THE PROGRAM WILL THEN ASK FOR PARAMETERS IN THE ORDER AS DESCRIBED IN SECTION 5.5. A <CR> INDICATES DEFAULT THIS PARAMETER AND A CONTROL-Z <Z> INDICATES THE THE REST OF THE PARAMETERS FOR THIS DRIVE ARE TO BE DEFAULTED.

IF N=A, PERFORM PN COMMAND ON ALL DRIVES.

- DN DROP DRIVE FROM TESTING SEQUENCE AND PRINT PERFORMANCE SUMMARY. IF N=A, DROP ALL CURRENTLY BEING TESTED DRIVES FROM TESTING SEQUENCE

- SN DEMAND PERFORMANCE SUMMARY ON DRIVE N. IF N=A DEMAND PERFORMANCE SUMMARY ON ALL CURRENTLY BEING TESTED DRIVES.

- WN WRITE AND VERIFY THE RK06 DISK PACK ON DRIVE N WITH A RANDOM SET OF LEGAL PATTERNS. THIS IS USED PRIOR TO THE START OF THE TEST SEQUENCE TO GUARANTEE THAT THE PROPER PREDETERMINED TEST PATTERNS HAVE BEEN WRITTEN ON THE PACK. (SEE APPENDIX A FOR DETAILS) AFTER THE PACK HAS BEEN COMPLETELY WRITTEN TESTING WILL BEGIN USING THE PREVIOUS PARAMETERS AS DESCRIBED IN THE TN COMMAND.

WRITING OF THE PACKS WILL TAKE 4 TO 5 MINUTES PER PACK WITH MAXIMUM BUFFER SIZE. THE WRITING OF THE PACK MAY TAKE LONGER IF THE MAXIMUM BUFFER SIZE SPECIFIED IS SMALL. RANDOM EXERCISE CAN BE DONE ON SOME DRIVES WHILE EXERCISE ON OTHER DRIVES ARE IN PROGRESS.

NOTE: IF AN ERROR OCCURS DURING THE DRIVE ASSIGNMENT SEQUENCE OR WHILE WRITING THE RANDOM SET OF LEGAL PATTERNS, THE ERROR WILL BE REPORTED AND TESTING WILL NOT START.

EXAMPLES:

COMMAND SEQUENCE	ACTION TAKEN
TA<CR>	INITIATE TESTING ON ALL DRIVES ON SYSTEM
T1<CR>	INITIATE TESTING ON DRIVE 1
T3<CR>	INITIATE TESTING ON DRIVE 3 (TESTING STILL CONTINUES ON DRIVE 1)
SA<CR>	GATHER STATISTICS ON ALL DRIVES UNDER TEST.
DA<CR>	DROP ALL DRIVES UNDER TEST.

4.2 DRIVE PARAMETERS (PER EACH DRIVE)

A SET OF THESE PARAMETERS EXISTS FOR EACH DRIVE. A TN COMMAND WILL UTILIZE THE PREVIOUSLY ESTABLISHED PARAMETERS FOR THE DRIVE(S). A PN COMMAND WILL ALLOW THE OPERATOR TO ALTER ALL THE PARAMETERS FOR THE DRIVE(S).

A <CR> WILL DEFAULT CURRENT PARAMETERS. A CONTROL Z <^Z> <CR> WILL DEFAULT ALL THE REST OF THE PARAMETERS. A CONTROL C <^C> <CR> WILL RETURN TO FIRST PARAMETER ENTRY REQUEST. FOR THIS DRIVE. A SUBOUT CANCELS THE LAST CHARACTER TYPED. A CONTROL U <^U> CANCELS LINE BEING TYPED. ALL ENTRIES MUST BE TERMINATED WITH CARRIAGE RETURN.

PARAMETER -----	DEFAULT VALUE -----
- MAXIMUM CYLINDER FOR START OF DATA TRANSFER 0 - 632	632
- MINIMUM CYLINDER FOR START OF DATA TRANSFER 0 - 632	0
- MAXIMUM TRACK FOR START OF DATA TRANSFER 0 - 2	2
- MINIMUM TRACK FOR START OF DATA TRANSFER 0 - 2	0
- MAXIMUM SECTOR FOR START OF DATA TRANSFER 0 - 25	25
- MINIMUM SECTOR FOR START OF DATA TRANSFER 0 - 25	0
- RATIO OF READ/WRITE EVERY COMMAND RANDOMLY GENERATED BY THE PROGRAM WILL BE EITHER A READ OR A WRITE. FOR EACH DRIVE ONE CAN WEIGHT THE COMMANDS TOWARDS READING OR WRITING. THE POSSIBLE WEIGHTING RATIOS ARE: READ ONLY =0 7/1 =1 3/1 =2 5/3 =3 1/1 =4 3/5 =5 1/3 =6 1/7 =7	5/3
- DO WRITE CHECK AFTER EVERY WRITE COMMAND OTHERWISE, WRITE CHECK COMMANDS WILL BE ISSUED RANDOMLY AFTER WRITE COMMANDS.	NO
- CORRECTABLE READ ERROR THRESHOLD	10
- UNCORRECTABLE READ ERROR THRESHOLD	5
- SEEK ERROR THRESHOLD	5
- INHIBIT ERROR CORRECTION	NO
- OPERATION COUNT THRESHOLD*65K	4.290*10**9
- DATA TRANSFER THRESHOLD*520K WORDS	291*10**12
- SAMPLED COMPARIS	YES
- UP TO 5 EXCLUDED PACK AREAS (NO DATA TRANSFER WILL BE DONE TO THESE AREAS.)	NO EXCLUDED PACK AREAS

4.2.1 FORMATS FOR PACK AREA EXCLUSION

FORMAT	ACTION
-----	-----
CYL<CR>	ALL TRACK ON CYLINDER SPECIFIED WILL BE EXCLUDED.
CYL,TRK<CR>	THE ENTIRE TRACK WILL BE EXCLUDED.
CYL1,TRK1,CYL2,TRK2<CR>	ALL TRACKS FROM CYLINDER 1 TRACK 1 TO AND INCLUDING CYLINDER 2 TRACK 2 WILL BE EXCLUDED.

4.3 DIAGNOSTIC DUMP

DUE TO THE COMPLEXITY OF THIS PROGRAM A DUMP OF THE PROGRAM TABLES MUST ACCOMPANY EACH PROBLEM REPORT. THIS DUMP IS PROVIDED IN THE PROGRAM AND IS ACTIVATED BY STARTING THE PROGRAM AT LOCATION 220. THIS DUMP WILL PRINT THE PROCESSOR REGISTERS WHEN THE DUMP IS CALLED. IT WILL THEN PRINT ABOUT 12-15 TELETYPE PAGES OF SOFTWARE TABLES NECESSARY TO DETERMINE WHAT THE PROGRAM WAS DOING AT A GIVEN TIME. APPROXIMATE TIME OF DUMP IS 10-12 MINUTES.

5.0 PROGRAM DESCRIPTION

THIS PROGRAM ASSUMES THAT ONLY THE PATTERNS WRITTEN BY THIS PROGRAM WILL BE ON THE PACK. TO PRECONDITION THE PACK A WN COMMAND MUST BE DONE. PACK SERIAL, DRIVE SERIAL NUMBER, THE TIME, AND DRIVE M UNDER TEST WILL BE PRINTED AT THE END OF PACK PRECONDITIONING. THE OPERATION OF THE PROGRAM IS DEPENDENT ON THE SWITCH REGISTER, SYSTEM PARAMETERS, AND DRIVE PARAMETERS. DRIVES MAY BE DROPPED FROM TEST BY USING THE DN COMMAND. IF A DRIVE IS DROPPED FOR ANY REASON, PERFORMANCE STATISTICS WILL AUTOMATICALLY BE PRINTED.

5.1 PROGRAM RESTRICTIONS

THIS PROGRAM WILL NOT RUN IN CHAIN MODE UNDER XXDP. NO END-OF-PASS INDICATOR IS PROVIDED FOR ACT. THE APT PROTOCOL HAS NOT BEEN DETERMINED YET.

6.0 PRINT OUTS

6.1 PERFORMANCE SUMMARY TYPEOUT

THE PERFORMANCE SUMMARY WILL CONTAIN THE FOLLOWING INFORMATION:

DRIVE NUMBER
DRIVE SERIAL NUMBER
TIME OF REPORT (IF REAL TIME CLOCK IS AVAILABLE)
NUMBER OF ORDERS PERFORMED BY DRIVE
NUMBER OF SEEK OPERATIONS PERFORMED
TOTAL NUMBER OF WORDS WRITTEN BY DRIVE*65K
TOTAL NUMBER OF WORDS READ BY DRIVE*65K
NUMBER OF SOFT DATA ERRORS
 (ECC CORRECTABLE)
 (REREAD CORRECTABLE)
 (OFFSET CORRECTABLE)
NUMBER OF HARD DATA ERRORS
NUMBER OF SEEK INCOMPLETES
NUMBER OF MISPOSITIONING ERRORS
TOTAL NUMBER OF ALL OTHER ERRORS.

6.2 ERROR REPORTS

THE ERROR REPORTS WILL CONTAIN THE FOLLOWING INFORMATION:

DRIVE NUMBER
DRIVE SERIAL NUMBER
TIME ERROR OCCURRED (IF REAL TIME CLOCK IS AVAILABLE)
DESCRIPTION OF ERROR
PRESENT COMMAND
PREVIOUS COMMAND
PRESENT POSITION
PREVIOUS POSITION
ALL DRIVE STATUS REGISTERS
ALL CONTROLLER REGISTERS
MEMORY ADDRESS, GOOD DATA AND BAD DATA (IF DATA ERROR)
OFFSET VALUE (IF APPLICABLE)
HEADER READ (IF MISPOSITIONING ERROR)
ECC PATTERN AND POSITION REGISTER IF DATA CHECKED

APPENDIX A

FIGURE A-1 SHOWS THE DATA EXPECTED BY THIS PROGRAM FOR EACH SECTOR OF THE RK06 PACK USED BY THIS PROGRAM.

15	8 7	4 3	0	WORD

CPU ID	PAT	PAT		0
PATC (N)		PATC (N)		1
RANDOM PATTERN				2 TO
				N+1

ZERO FILL				N+2 TO
				255

CPU ID CPU IDENTIFIER SUPPLIED BY THE OPERATOR

PAT PATTERN USED

PATC PATTERN COUNT (NUMBER OF WORDS OF PATTERN WRITTEN)

FIGURE A-1
EXPECTED DATA FORMAT

DATA PATTERNS USED

PATTERN 0	PATTERN 1	PATTERN 2	PATTERN 3
165555	026455	007417	052525
133333	151322	170360	125252
165555	026455	007417	052525
133333	026455	007417	052525
165555	151322	170360	125252
133333	151322	170360	125252
165555	026455	007417	052525
133333	026455	007417	052525
165555	026455	007417	052525
133333	151322	170360	125252
165555	151322	170360	125252
133333	151322	170360	125252
165555	026455	007417	052525
133333	026455	007417	052525
165555	026455	007417	052525
133333	026455	007417	052525
165555	151322	170360	125252
133333	151322	170360	125252
165555	151322	170360	125252
133333	151322	170360	125252
165555	026455	007417	052525
133333	026455	007417	052525
165555	026455	007417	052525
133333	151322	170360	125252
165555	151322	170360	125252
133333	151322	170360	125252
165555	026455	007417	052525
133333	026455	007417	052525
165555	151322	170360	125252
133333	151322	170360	125252
165555	026455	007417	052525
133333	151322	170360	125252

PATTERN 4	PATTERN 5	PATTERN 6	PATTERN 7
000000	000000	000001	000001
010421	177777	000003	000002
021042	000000	000007	000004
031463	000000	000017	000010
042104	177777	000037	000020
052525	177777	000077	000040
063146	000000	000177	000100
073567	000000	000377	000200
104210	000000	000777	000400
114631	177777	001777	001000
125252	177777	003777	002000
135673	177777	007777	004000
146314	000000	017777	010000
156735	000000	037777	020000
167356	000000	077777	040000
177777	000000	177777	100000
177777	177777	177776	100000
167356	177777	177774	040000
156735	177777	177770	020000
146314	177777	177760	010000
135673	000000	177740	004000
125252	000000	177700	002000
114631	000000	177600	001000
104210	177777	177400	000400
073567	177777	177000	000200
063146	177777	176000	001000
052525	000000	174000	000040
042104	000000	170000	000020
031463	177777	160000	000010
021042	177777	140000	000004
010421	000000	100000	000002
000000	177777	000000	000001

PATTERN 10	PATTERN 11	PATTERN 12	PATTERN 13
177776	172666	153333	000000
177775	155555	066667	177777
177773	172666	153333	177777
177767	155555	066667	177777
177757	172666	153333	177777
177737	155555	066667	177777
177677	172666	153333	177777
177577	155555	066667	177777
177377	172666	153333	177777
176777	155555	066667	177777
175777	172666	153333	177777
173777	155555	066667	177777
167777	172666	153333	177777
157777	155555	066667	177777
137777	172666	153333	177777
077777	155555	066667	177777
077777	172666	153333	177777
177777	155555	066667	177777
157777	172666	153333	177777
167777	155555	066667	177777
173777	172666	153333	177777
175777	155555	066667	177777
176777	172666	153333	177777
177377	155555	066667	177777
177577	172666	153333	177777
177677	155555	066667	177777
177737	172666	153333	177777
177757	155555	066667	177777
177767	172666	153333	177777
177777	155555	066667	177777
177777	172666	153333	177777
177776	155555	066667	177777

[illegible]

27	OPERATIONAL SWITCH SETTINGS
43	BASIC DEFINITIONS
160	RK06 CONTROLLER REGISTER DEFINITION
181	DRIVE COMMANDS
212	CONTROL AND STATUS REGISTER 1 BITS
236	CONTROL AND STATUS REGISTER 2 BITS
255	ERROR REGISTER BIT DEFINITION
277	STATUS REGISTER BIT DEFINITION
295	MAINTENANCE REGISTER 1 BIT DEFINITION
312	DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE A
325	DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE P
339	DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE A
353	DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE N
367	COMMON MASKS
378	DEFAULT DRIVE VALUES
398	OFFSET VALUES
407	DEFAULT CONTROLLER THRESHOLD
411	PARAMETER BLOCK ALLOCATION
442	PARAMETERS PASSED TO THE DRIVER
459	PROGRAM DEVICE STATUS REGISTER DEFINITION
481	PARAMETERS PASSED FROM DRIVER TO PROGRAM
506	DRIVE SPECIFIC PARAMETERS
638	TRAP CATCHER
647	STARTING ADDRESS(ES)
656	APT PARAMETER BLOCK
678	COMMON TAGS
720	APT MAILBOX-ETABLE
788	ERROR POINTER TABLE
789	QUEUE HEAD AND TAIL ALLOCATIONS
796	SYSTEM PARAMETERS
825	TEMPORARY CONTROLLER REGISTER STORAGE
846	DRIVER PARAMETERS
902	TABLE OF INTERRUPT MASKS
913	PARAMETER BLOCK TABLE
924	WATCH-DOG TIMER COUNTS
934	PERFORMANCE EXERCISER STATUS INFORMATION
974	FREE SPACE BLOCK
988	PARAMETER BLOCK FOR DRIVE 0
1081	PARAMETER BLOCK FOR DRIVE 1
1174	PARAMETER BLOCK FOR DRIVE 2
1267	PARAMETER BLOCK FOR DRIVE 3
1360	PARAMETER BLOCK FOR DRIVE 4
1453	PARAMETER BLOCK FOR DRIVE 5
1546	PARAMETER BLOCK FOR DRIVE 6
1639	PARAMETER BLOCK FOR DRIVE 7
1731	--- RETRY PARAMETER BLOCK ---
1733	PARAMETER BLOCK 10 FOR DRIVE
1764	DATA PATTERNS AND DATA PATTERN TABLE
2310	PROGRAM SETUP
2320	INITIALIZE THE COMMON TAGS
2353	TYPE PROGRAM NAME
2749	MAIN IDLE LOOP
2805	OPERATOR COMMAND DECODER
2871	TEST DRIVE COMMAND
2907	CHANGE PARAMETERS COMMAND
2933	DROP DRIVE COMMAND

2969	GET DRIVE STATISTICS COMMAND
3005	WRITE PACK AND TEST COMMAND
3043	DROP DRIVE AND GATHER STATISTICS ROUTINES
3102	INITIATE DRIVE ASSIGNMENT SEQUENCE
3130	DRIVE ASSIGNMENT SEQUENCE
3715	WRITE WHOLE PACK WITH RANDOM DATA
3475	ALTER DRIVE PARAMETERS
3899	TEST DEFAULT PARAMETERS ROUTINE
3934	LOAD DISK EXCLUDED AREAS
4078	GENERATE NEW RANDOM COMMAND
4298	GET AVAILABLE BUFFER ROUTINE
4319	BUFFER ALLOCATION ROUTINE
4397	BUFFER RELEASE ROUTINE
4490	LOAD DATA BUFFER
4586	FIND FIRST DATA ERROR
4604	COMPARE DATA BUFFER
4790	NORMAL COMMAND RETURN
4985	CHECK BUS ADDRESS, WORD COUNT, SECTOR, TRACK, AND CYLINDER
5049	ABNORMAL TERMINATION
5098	COMMAND TIME OUT ROUTINE
5130	PRINT RK06 UNIBUS REGISTERS
5145	PRINT DRIVE STATUS
5180	PRINT CONTROLLER STATUS
5194	PRINT REGISTER CONTENTS
5205	ECC CORRECTION ROUTINE
5258	CONTROLLER ERROR DETECTED BY RK06 DRIVER
5358	CONTROLLER ERROR DUE TO REGISTER INCONSISTANCIES
5407	PRINT WHOLE DATA BUFFER
5447	PRINT DRIVE AND PACK SERIAL NUMBERS
5498	PRINT BEGINNING OF ERROR HEADER
5511	PRINT PREVIOUS COMMAND
5543	PRINT CURRENT GENERATED COMMAND
5586	PRINT ERROR MESSAGE AND DROP DRIVE
5634	PRINT ERROR MESSAGE
5729	HALT ON ERROR CHECK
5740	OTHER ERROR STATISTICS GATHERING
5750	SEEK INCOMPLETE STATISTICS GATHERING
5757	OPERATION INCOMPLETE STATISTICS GATHERING
5788	UNSUCCESSFUL RECOVERY CLEANUP
5846	ERROR RECOVERY SEQUENCE
6408	CALCULATE CYLINDER, TRACK, AND SECTOR FOR RETRY
6453	PRINT FIRST ERROR WORD IN BUFFER
6516	SUCCESSFUL RECOVERY CLEAN UP
6611	GET CURRENT STATUS INFORMATION
6625	RESTORE STATUS
6638	DETERMINE OFFSET FOR RETRY
6676	CALCULATE RETRY WORD COUNT
6704	RECOVERABLE CONTROLLER ERROR TERMINATION
6740	TEST FOR DROP DRIVE ERRORS
6880	ERROR RECOVERY SEQUENCE NORMAL RETURN
7163	CALCULATE WORDS TRANSFERRED UNTIL BAD SECTOR
7207	BAD SECTOR HANDLING
7304	ERROR RECOVERY SEQUENCE ABNORMAL RETURN
7365	DETERMINE IF CLOCK IS PRESENT
7383	SET UP CLOCK INTERRUPT
7400	KW11-L AND KW11-P INTERRUPT HANDLER

7421	PRINT TIME ROUTINE
7452	CONVERT DECIMAL LESS THAN 100
7471	RK611/RK06 UNIBUS DRIVER FOR QUEUED OPERATIONS (REV. 0.08)
7478	*WATCH-DOG TIMER
7557	*RK06 INTERRUPT SERVICE ROUTINE
7743	*READ ALL HEADERS INTERRUPT SEQUENCE
7802	*DRIVE ATTENTION SCANNER
7914	*ATTENTION ERROR HANDLER
7954	*ERROR CAUSING DRIVE TO UNLOAD
7974	*DUAL ACCESS INTERRUPT HANDLER
8017	*CONTROLLER CLEAR ROUTINE
8052	*COMMAND ISSUED BY DRIVER SERVICE ROUTINE
8123	*STORE RK611 UNIBUS REGISTERS
8155	*STORE CONTROLLER STATUS
8200	*GATHER DRIVE STATUS
8265	*COMMAND INITIATOR
8410	*SPECIAL COMMAND PROCESSING
8541	*ENQUEUE ROUTINE
8575	*POP QUEUE ROUTINE
8609	*SEARCH QUEUE FOR DRIVE NOT POSITIONING
8648	*REMOVE PARAMETER BLOCK FROM COMMAND INITIATION QUEUE
8701	*COMMAND OPTIMIZER
8812	MEMORY CHECK ENABLE TEST
8919	TYPE ROUTINE
8899	BINARY TO OCTAL (ASCII) CONVERSION
8942	TTY INPUT ROUTINE
9046	TK INITIALIZE ROUTINE
9065	TK SERVICE ROUTINE (ONLY CONTROL C IS REGNIZED)
9141	OCTAL TO BINARY CONVERSION ROUTINE
9197	DECIMAL TO BINARY CONVERSION ROUTINE
9758	ROUTINE TO SIZE MEMORY
9790	POWER DOWN AND UP ROUTINES
9320	INTEGER MULTIPLY ROUTINE
9374	RANDOM NUMBER GENERATOR ROUTINE
9411	SINGLE/DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINES
9506	APT COMMUNICATIONS ROUTINE
9563	TRAP DECODER
9586	TRAP TABLE
9599	ASCII MESSAGES
10286	*** DUMP MEMORY ***

```
1 .TITLE RK611/RK06 PERFORMANCE EXERCISER MAINDEC DZR6P-B
2 ;*COPYRIGHT (C) 1976
3 ;*DIGITAL EQUIPMENT CORP.
4 ;*MAYNARD, MASS. 01754
5 ;*
6 ;*PROGRAM BY ROY SPITZER
7 ;*
8 ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
9 ;*PACKAGE (MAINDEC-11-DZQAC-C1), MAR 24, 1976.
10 ;*
```

```
11 ;*****
12 ;*
13 ;* THE RK06 PERFORMANCE EXERCISER WILL EXERCISE 1 TO 8
14 ;* RK06 DISK DRIVES ATTACHED TO THE SAME RK06
15 ;* UNIRUS CONTROLLER IN A DEDICATED STAND ALONE MODE.
16 ;*
17 ;* UNDER NORMAL OPERATION SITUATIONS, THE DRIVE IS
18 ;* CHOSEN RANDOMLY FROM ANY ONE OF THE RK06 DRIVES UNDER TEST
19 ;* WHICH IS NOT CURRENTLY UNDERGOING ERROR RECOVERY.
20 ;* TO OPTIMIZE THE THROUGHOUT ON THE RK06 CONTROLLER EACH
21 ;* DATA TRANSFER COMMAND IS TRANSLATED INTO A SEEK FOLLOWED
22 ;* BY THE DATA TRANSFER COMMAND.
23 ;*
24 ;*
```

```
25 ;*****
26 .SBTTL OPERATIONAL SWITCH SETTINGS
```

```
27 ;*
28 ;* SWITCH USE
29 ;* -----
30 ;* 15 HALT ON ERROR
31 ;* 14 LOCK ON TEST
32 ;* 13 INHIBIT ERROR TIMEOUTS
33 ;* 10 BELL ON ERROR
34 ;* 6 INHIBIT AUTOMATIC DRIVE REASSIGNMENT IF UNSAFE,
35 ;* AC LOW, OR SPEED LOSS OCCURS
36 ;* 5 INHIBIT AUTOMATIC DRIVE REASSIGNMENT WHEN
37 ;* MAXIMUM COMMAND COUNT FOR DRIVE IS EXCEEDED
38 ;* 4 INHIBIT AUTOMATIC DRIVE REASSIGNMENT WHEN
39 ;* ERROR THRESHOLD EXCEEDED
40 ;* 3 DUMP ENTIRE SECTOR BEFORE BEGINNING RETRY SEQUENCE
41 ;* 1 INHIBIT SOFTWARE DATA COMPARISONS
```

```

42      .SBTTL  BASIC DEFINITIONS
43
44      ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
45      001100  STACK= 1100
46      .EQUIV  EMT,ERROR      ;;BASIC DEFINITION OF ERROR CALL
47      .EQUIV  IOT,SCOPE      ;;BASIC DEFINITION OF SCOPE CALL
48
49      ;*MISCELLANEOUS DEFINITIONS
50      000011  HT= 11          ;;CODE FOR HORIZONTAL TAB
51      000012  LF= 12          ;;CODE FOR LINE FEED
52      000015  CR= 15          ;;CODE FOR CARRIAGE RETURN
53      000200  CRLF= 200       ;;CODE FOR CARRIAGE RETURN-LINE FEED
54      177776  PS= 177776      ;;PROCESSOR STATUS WORD
55      .EQUIV  PS,PSW
56      177774  STKLM= 177774   ;;STACK LIMIT REGISTER
57      177772  PIRQ= 177772    ;;PROGRAM INTERRUPT REQUEST REGISTER
58      177570  DSWR= 177570    ;;HARDWARE SWITCH REGISTER
59      177570  DDISP= 177570   ;;HARDWARE DISPLAY REGISTER
60
61      ;*GENERAL PURPOSE REGISTER DEFINITIONS
62      000000  R0= 0           ;;GENERAL REGISTER
63      000001  R1= 1           ;;GENERAL REGISTER
64      000002  R2= 2           ;;GENERAL REGISTER
65      000003  R3= 3           ;;GENERAL REGISTER
66      000004  R4= 4           ;;GENERAL REGISTER
67      000005  R5= 5           ;;GENERAL REGISTER
68      000006  R6= 6           ;;GENERAL REGISTER
69      000007  R7= 7           ;;GENERAL REGISTER
70      000006  SP= 6           ;;STACK POINTER
71      000007  PC= 7           ;;PROGRAM COUNTER
72
73      ;*PRIORITY LEVEL DEFINITIONS
74      000000  PR0= 0          ;;PRIORITY LEVEL 0
75      000040  PR1= 40         ;;PRIORITY LEVEL 1
76      000100  PR2= 100        ;;PRIORITY LEVEL 2
77      000140  PR3= 140        ;;PRIORITY LEVEL 3
78      000200  PR4= 200        ;;PRIORITY LEVEL 4
79      000240  PR5= 240        ;;PRIORITY LEVEL 5
80      000300  PR6= 300        ;;PRIORITY LEVEL 6
81      000340  PR7= 340        ;;PRIORITY LEVEL 7
82
83      ;**SWITCH REGISTER* SWITCH DEFINITIONS
84      100000  SW15= 100000
85      040000  SW14= 40000
86      020000  SW13= 20000
87      010000  SW12= 10000
88      004000  SW11= 4000
89      002000  SW10= 2000
90      001000  SW09= 1000
91      000400  SW08= 400
92      000200  SW07= 200
93      000100  SW06= 100
94      000040  SW05= 40
95      000020  SW04= 20
96      000010  SW03= 10
97      000004  SW02= 4

```

98	000002	SW01= 2	
99	000001	SW00= 1	
100		.EQUIV SW09,SW9	
101		.EQUIV SW08,SW8	
102		.EQUIV SW07,SW7	
103		.EQUIV SW06,SW6	
104		.EQUIV SW05,SW5	
105		.EQUIV SW04,SW4	
106		.EQUIV SW03,SW3	
107		.EQUIV SW02,SW2	
108		.EQUIV SW01,SW1	
109		.EQUIV SW00,SW0	
110			
111		;*DATA BIT DEFINITIONS (BIT00 TO BIT15)	
112	100000	BIT15= 100000	
113	040000	BIT14= 40000	
114	020000	BIT13= 20000	
115	010000	BIT12= 10000	
116	004000	BIT11= 4000	
117	002000	BIT10= 2000	
118	001000	BIT09= 1000	
119	000400	BIT08= 400	
120	000200	BIT07= 200	
121	000100	BIT06= 100	
122	000040	BIT05= 40	
123	000020	BIT04= 20	
124	000010	BIT03= 10	
125	000004	BIT02= 4	
126	000002	BIT01= 2	
127	000001	BIT00= 1	
128		.EQUIV BIT09,BIT9	
129		.EQUIV BIT08,BIT8	
130		.EQUIV BIT07,BIT7	
131		.EQUIV BIT06,BIT6	
132		.EQUIV BIT05,BIT5	
133		.EQUIV BIT04,BIT4	
134		.EQUIV BIT03,BIT3	
135		.EQUIV BIT02,BIT2	
136		.EQUIV BIT01,BIT1	
137		.EQUIV BIT00,BIT0	
138			
139		;*BASIC MCPU TRAP VECTOR ADDRESSES	
140	000004	ERRVEC= 4	;;TIME OUT AND OTHER ERRORS
141	000010	RESVEC= 10	;;RESERVED AND ILLEGAL INSTRUCTIONS
142	000014	TRITVEC=14	;;TM BIT
143	000014	TRTVEC= 14	;;TRACE TRAP
144	000014	BPTVEC= 14	;;BREAKPOINT TRAP (BPT)
145	000020	IOTVEC= 20	;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
146	000024	PMRVEC= 24	;;POWER FAIL
147	000030	EMTVEC= 30	;;EMULATOR TRAP (EMT) **ERROR**
148	000034	TRAPVEC=34	;;"TRAP" TRAP
149	000060	TKVEC= 60	;;TTY KEYBOARD VECTOR
150	000064	TPVEC= 64	;;TTY PRINTER VECTOR
151	000240	PIRQVEC=240	;;PROGRAM INTERRUPT REQUEST VECTOR
152	000240	APRQR= PR5	;;RK611 PRIORITY
153	120210	AVECT1= 120210	;;RK611 VECTOR

154	177440	ABASE= 177440	;RK611 BASE
155	000114	MEMVEC= 114	;VECTOR FOR MEMORY CHECK ENABLE
156	172100	MEMBAS= 172100	;BUS ADDRESS FOR MEMORY CHECK ENAPLF
157	000001	PAR.EN= 1	;MEMORY ENABLE MEMORY CHECKING
158			
159		.SBTTL RK06 CONTROLLER REGISTER DEFINITION	
160			
161	000000	RKCS1= 0	;CONTROL AND STATUS REGISTER 1
162	000002	RKWC= 2	;WORD COUNT REGISTER
163	000004	RKBA= 4	;BUS ADDRESS REGISTER
164	000006	RKDA= 6	;DESIRED TRACK SECTOR REGISTER
165	000010	RKCS2= 10	;CONTROL AND STATUS REGISTER 2
166	000012	RKDS= 12	;DRIVE STATUS REGISTER
167	000014	RKER= 14	;ERROR REGISTER
168	000016	RKASOP= 16	;ATTENTION SUMMARY AND OFFSET REGISTER
169	000020	RKDC= 20	;DESIRED CYLINDER REGISTER
170	000020	RKDCYL= 20	;DESIRED CYLINDER REGISTER
171	000024	RKDB= 24	;DATA BUFFER
172	000026	RKMR1= 26	;MAINTENANCE REGISTER 1
173	000034	RKMR2= 34	;MAINTENANCE REGISTER 2
174	000036	RKMR3= 36	;MAINTENANCE REGISTER 3
175	000030	RKPOS= 30	;ECC POSITION INFORMATION
176	000030	RKECPS= 30	;ECC POSITION INFORMATION
177	000032	RKPAT= 32	;ECC PATTERN INFORMATION
178	000032	RKECPT= 32	;ECC PATTERN INFORMATION
179	000037	RKECPT= 37	;ECC PATTERN INFORMATION
180			
181		.SBTTL DRIVE COMMANDS	
182			
183	000101	SELDRV= 101	;SELECT DRIVE
184	000103	PACK= 103	;PACK ACKNOWLEDGE
185	000105	CLEAR= 105	;DRIVE CLEAR
186	000107	UNLOAD= 107	;UNLOAD
187	000111	SPTSPL= 111	;START SPINDLE
188	000113	RFCAL= 113	;RECALIBRATE
189	000115	OFFSET= 115	;OFFSET
190	000117	SEEK= 117	;SEEK
191	000121	RDDATA= 121	;READ DATA
192	000123	WRDATA= 123	;WRITE DATA
193	000125	RDHEAD= 125	;READ HEADER
194	000127	WRHEAD= 127	;WRITE HEADER AND DATA
195	000131	WRTCHK= 131	;WRITE CHECK
196			
197		; THE FOLLOWING ARE NOT DRIVE COMMANDS BUT ARE USED BY THE DRIVER	
198		; TO SIMULATE A SPECIFIC DESIRED OPERATION	
199	000140	RFLAS= 140	;RELEASE DRIVE
200	000141	RSTAT= 141	;GET ALL STATUS FROM DRIVE
201	000164	RDALHD= 164	;READ ALL HEADERS
202	000176	CONCLR= 176	;CONTROL CLEAR (BIT 15 OF CS1)
203	000177	SUBCLR= 177	;SUBSYSTEM CLEAR (BIT 5 OF CS2)
204	000300	INTR= 300	;GENERATE INTERRUPT TO CPU
205			
206		; DRIVE ISSUED SERVICE COMMANDS	
207			
208	000001	DR.SEL= 001	;DRIVE SELECT
209	000005	DR.CL= 005	;DRIVE CLEAR

210			
211		.SBTTL	CONTROL AND STATUS REGISTER 1 BITS
212			
213	000001	GO=	BIT0 ;GO BIT
214	000100	IE=	BIT6 ;INTERRUPT ENABLE
215	000200	RDY=	BIT7 ;CONTROLLER READY
216	000400	BA16=	BIT8 ;BUS ADDRESS BIT 16
217	001000	BA17=	BIT9 ;BUS ADDRESS BIT 17
218	002000	CDT=	BIT10 ;CONTROLLER DRIVE TYPE (0=RK06)
219	004000	CTO=	BIT11 ;CONTROLLER TIME OUT WAITING FOR
220			; DRIVE RESPONSE
221	010000	CFMT=	BIT12 ;CONTROLLER DRIVE FORMAT (0=22 SECTOR, 1=70 SECTOR)
222	020000	SPAR=	BIT13 ;DRIVE BUS PARITY ERROR DETECTED BY CONTROLLER
223	040000	DI=	BIT14 ;DRIVE INTERRUPT
224	100000	CERR=	BIT15 ;CONTROLLER ERROR
225	100000	CCLR=	BIT15 ;CONTROLLER CLEAR
226			
227			; THESE BIT DEFINITIONS ARE USED FOR ADDRESS
228			; THE HIGH BYTE OF RKCS1
229			
230	000001	B.BA16=	BIT0 ;BUS ADDRESS BIT 16
231	000002	B.BA17=	BIT1 ;BUS ADDRESS BIT 17
232	000004	B.CDT=	BIT2 ;CONTROLLER DRIVE TYPE (0=RK06)
233	000020	B.CFMT=	BIT4 ;CONTROLLER DRIVE FORMAT (0=22 SECTOR, 1=70 SECTOR)
234			
235		.SBTTL	CONTROL AND STATUS REGISTER 2 BITS
236			
237	000007	DRVMSK=	7 ;MASK FOR DRIVE SELECTION CODE
238	000010	DESL=	BIT3 ;DESELECT OR RELEASE DRIVE IN BITS 0-2
239	000010	RLS=	BIT3 ;DESELECT OR RELEASE DRIVE IN BITS 0-2
240	000020	BAI=	BIT4 ;BUS ADDRESS INCREMENT INHIBIT
241	000040	CLR=	BIT5 ;CLEAR CONTROLLER AND ALL DRIVES
242	000040	SCLR=	BIT5 ;CLEAR CONTROLLER AND ALL DRIVES
243	000100	IR=	BIT6 ;INPUT READY
244	000200	OR=	BIT7 ;OUTPUT READY
245	000400	UPE=	BIT8 ;UNIT FIELD ERROR
246	001000	MDS=	BIT9 ;MULTIPLE DRIVE SELECT
247	002000	PGE=	BIT10 ;PROGRAMMING ERROR
248	004000	NEM=	BIT11 ;NON-EXISTENT MEMORY
249	010000	NED=	BIT12 ;NON-EXISTENT DRIVE
250	020000	UPE=	BIT13 ;UNIT BUS PARITY ERROR
251	040000	WCE=	BIT14 ;WRITE CHECK ERROR
252	100000	DLT=	BIT15 ;DATA LATE ERROR
253			
254		.SBTTL	ERROR REGISTER BIT DEFINITION
255			
256	000001	ILC=	BIT0 ;ILLEGAL FUNCTION CODE
257		*ILF=	BIT0 ;ILLEGAL FUNCTION CODE
258	000002	SKI=	BIT1 ;SEEK INCOMPLETE
259	000004	ILF=	BIT2 ;ILLEGAL DRIVE FUNCTION
260	000004	NXF=	BIT2 ;ILLEGAL DRIVE FUNCTION
261	000010	DRPAR=	BIT3 ;DRIVE DETECTED DRIVE BUS PARITY ERROR
262	000020	FMTE=	BIT4 ;FORMAT ERROR
263	000040	DTYE=	BIT5 ;DRIVE TYPE ERROR
264	000100	ECH=	BIT6 ;ECC HARD
265	000200	BSE=	BIT7 ;BAD SECTOR ERROR

766	000400	HCRC=	BIT8	;HEADER CRC ERROR
267	000400	HVRC=	BIT8	;HEADER VRC ERROR
268	001000	COE=	BIT9	;CYLINDER ADDRESS OVERFLOW ERROR
269	002000	IDAE=	BIT10	;INVALID DISK ADDRESS ERROR
270	004000	WLE=	BIT11	;WRITE LOCK ERROR
271	010000	DTE=	BIT12	;DRIVE TIMING ERROR
272	020000	OPI=	BIT13	;OPERATION (SEARCH) INCOMPLETE
273	040000	UNS=	BIT14	;DRIVE UNSAFE
274	100000	DCR=	BIT15	;DATA CHECK
775				
776		.SBTTL	STATUS REGISTER BIT DEFINITION	
277				
278	000001	DRA=	BIT0	;DRIVE AVAILABLE (CONTROLLER IS SET IF ; THIS BIT IS RESET)
279				
780	000004	OFST=	BIT2	;DRIVE OFFSET
281	000010	ACLO=	BIT3	;AC LOW
282	000020	SPDLSS=	BIT4	;SPEED LOSS
283	000020	DCLO=	BIT4	;DC LOW
284	000040	OROT=	BIT5	;DRIVE OFF TRACK
285	000100	VV=	BIT6	;VOLUME VALID
286	000200	DRY=	BIT7	;DRIVE READY
287	000200	DRDY=	BIT7	;DRIVE READY
288	000400	DDT=	BIT8	;DRIVE TYPE (0=RKC6)
289	004000	WRL=	BIT11	;WRITE LOCK
290	020000	PIP=	BIT13	;POSITIONING IN PROGRESS
291	040000	DSC=	BIT14	;DRIVE STATUS CHANGE
292	100000	SVAL=	BIT15	;STATUS VALID
293				
794		.SBTTL	MAINTENANCE REGISTER 1 BIT DEFINITION	
295				
296	000017	MESMSF=	17	;MESSAGE MASK
297				
298	000020	PAT=	BIT4	;FORCE EVEN PARITY ON DRIVE BUS MESSAGE LINES
299	000040	DMD=	BIT5	;DIAGNOSTIC MODE
300	000100	MSP=	BIT6	;MAINTENANCE SELECT PULSE
301	000200	MIND=	BIT7	;MAINTENANCE INDEX
302	000400	MCLK=	BIT8	;MAINTENANCE CLOCK
303	001000	MFRD=	BIT9	;MAINTENANCE ENCODED READ DATA
304	002000	MEWD=	BIT10	;MAINTENANCE ENCODED WRITE DATA
305	004000	PCA=	BIT11	;PRECOMPENSATION ADVANCE
306	010000	PCD=	BIT12	;PRECOMPENSATION DELAY
307	020000	ECCW=	BIT13	;ECC WORD IS BEING READ OR WRITTEN
308	040000	WRTGAT=	BIT14	;WRITE GATE
309	100000	RDGATE=	BIT15	;READ GATE
310				
311		.SBTTL	DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE A	
312				
313	000040	S.DRA=	BIT5	;DRIVE AVAILABLE
314	000100	S.VV=	BIT6	;VOLUME VALID
315	000200	S.DRY=	BIT7	;DRIVE READY
316	000400	S.TYPE=	BIT8	;DRIVE TYPE
317	001000	S.FORM=	BIT9	;DRIVE FORMAT
318	002000	S.OFF=	BIT10	;OFFSET
319	004000	S.WRL=	BIT11	;WRITE LOCK
320	010000	S.SPIN=	BIT12	;SPINDLE ON
321	020000	S.PIP=	BIT13	;POSITIONING IN PROGRESS

322	040000	S.DSC= BIT14	;DRIVE STATUS CHANGE
323			
324		.SBTTL	DEFINITION OF DRIVE STATUS BYTE 00 MESSAGE B
325			
326	000040	S.ICVL= BIT5	;ILLEGAL CYLINDER ADDRESS
327	000100	S.ACLO= BIT6	;AC LOW
328	000200	S.PLT= BIT7	;DRIVE FAULT
329	000400	S.ILF= BIT8	;ILLEGAL FUNCTION
330	001000	S.PAR= BIT9	;DRIVE DETECTED DRIVE BUS PARITY ERROR
331	002000	S.SKI= BIT10	;SEEK INCOMPLETE
332	004000	S.WLE= BIT11	;WRITE LOCK ERROR
333	010000	S.SPLS= BIT12	;SPEED LOSS
334	010000	S.DCLO= BIT12	;DC LOW
335	020000	S.DROT= BIT13	;DRIVE OFF TRACK
336	040000	S.UNS= BIT14	;DRIVE UNSAFE
337			
338		.SBTTL	DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE A
339			
340	000020	S.XDOR= BIT4	;TRANSDUCER OR
341	000040	S.HDWM= BIT5	;HEADS HOME
342	000100	S.BRHM= BIT6	;BRUSHES HOME
343	000200	S.DOOR= BIT7	;DOOR INTERLOCKED
344	000400	S.CART= BIT8	;CARTRIDGE INTERLOCK
345	001000	S.SPCR= BIT9	;SPEED OK
346	002000	S.FWD= BIT10	;FORWARD
347	004000	S.RPV= BIT11	;REVERSE
348	010000	S.LOAD= BIT12	;HEADS LOADING
349	020000	S.RTZ= BIT13	;RETURN TO REF
350	040000	S.UNLD= BIT14	;HEADS UNLOADING
351			
352		.SBTTL	DEFINITION OF DRIVE STATUS BYTE 01 MESSAGE B
353			
354	000020	S.SECT= BIT4	;SECTOR ERROR
355	000040	S.WCLK= BIT5	;WRITE CLOCK AND NO WRITE GATE
356	000100	S.WGAT= BIT6	;WRITE GATE AND NO TRANSITIONS
357	000200	S.HDFL= BIT7	;HEAD FAULT
358	000400	S.MHD= BIT8	;MULTIPLE HEAD SELECT
359	001000	S.XFMR= BIT9	;INDEX ERROR
360	002000	S.DIB= BIT10	;DIRTY ERROR
361	004000	S.PLC= BIT11	;PLO ERROR
362	010000	S.NMOV= BIT12	;SEEK AND NO MOTION
363	020000	S.LIMD= BIT13	;LIMIT DETECT ON SEEK
364	040000	S.BRKE= BIT14	;SERVO-BRAKE
365			
366		.SBTTL	COMMON MASKS
367			
368	000007	M.DRV= 7	;DRIVE CODE
369	100000	M.PAR= BIT15	;PARITY
370	000003	M.ID= 3	;BYTE ID
371	017760	M.CDIF= 17760	;CYLINDER DIFFERENCE/OFFSET
372	017760	M.CADD= 17760	;CYLINDER ADDRESS
373	077770	M.SER= 77770	;DRIVE SERIAL NUMBER
374	000760	M.SECT= 760	;SECTOR COUNT
375	007000	M.HEAD= 7000	;HEAD DECODE
376			
377		.SBTTL	DEFAULT DRIVE VALUES

378			
379	000000	D.MNCL= 0	;MINIMUM CYLINDER
380	000632	D.MXCL= 410.	;MAXIMUM CYLINDER
381	000000	D.MNTR= 0	;MINIMUM TRACK
382	000002	D.MXTR= 2	;MAXIMUM TRACK
383	000000	D.MWSC= 0	;MINIMUM SECTOR
384	000025	D.MXSC= 21.	;MAXIMUM SECTOR
385	000003	D.RATE= 3	;READ/WRITE RATIO (5/3)
386	000000	D.AWCK= 0	;NO AUTOMATIC WRITE-CHECK AFTER EVERY WRITE
387	000012	D.CERT= 10.	;CORRECTABLE READ ERROR THRESHOLD
388	000005	D.UERT= 5	;UNCORRECTABLE READ ERROR THRESHOLD
389	000005	D.SKET= 5	;SEEK ERROR THRESHOLD
390	077777	D.CTRH= 77777	;COMMAND COUNT THRESHOLD
391	177770	D.CTRL= 177770	; (MAXIMUM CP COMMANDS ISSUED TO DRIVE)
392	077777	D.WTHI= 77777	;WORDS TRANSFERRED THRESHOLD
393	177770	D.WTLO= 177770	; (MAXIMUM NUMBER OF WORDS READ OR WRITTEN
394			; ON PACK)
395	000020	T.MER= 20	;THRESHOLD OF OTHER DRIVE ERRORS
396			
397		.SBTTL OFFSET VALUES	
398			
399	000060	OFFP12= 060	;OFFSET +1200 MICRO-INCHES
400	000040	OFFP8= 040	;OFFSET +800 MICRO-INCHES
401	000020	OFFP4= 020	;OFFSET +400 MICRO-INCHES
402	000220	OFFM4= 220	;OFFSET -400 MICRO-INCHES
403	000240	OFFM8= 240	;OFFSET -800 MICRO-INCHES
404	000760	OFFM12= 260	;OFFSET -1200 MICRO-INCHES
405			
406		.SBTTL DEFAULT CONTROLLER THRESHOLD	
407			
408	001000	T.DLT= 1000	;THRESHOLD FOR DATA LATE
409	000012	T.COMT= 10.	;THRESHOLD FOR OTHER CONTROLLER ERRORS

```

410 .SBTTL PARAMETER BLOCK ALLOCATION
411
412 ;* *****
413 ;* 1  I COMMAND          I DRIVE NO.          I 0
414 ;* 3  I CYLINDER ADDRESS          I 2
415 ;* 5  I TPACK          I SECTOR          I 4
416 ;* 7  IBA16-17,FORMAT,DRV TYPE I OFFSET          I 6
417 ;* 11 I BUS ADDRESS (LOW 16 BITS)          I 10
418 ;* 13 I WORD COUNT (2'S COMPLEMENT)          I 12
419 ;* 15 I PROGRAM DRIVE STATUS INFORMATION          I 14
420 ;* 17 I COMMAND AND STATUS REGISTER 1          I 16
421 ;* 21 I COMMAND AND STATUS REGISTER 2          I 20
422 ;* 23 I WORD COUNT REGISTER          I 22
423 ;* 25 I BUS ADDRESS REGISTER          I 24
424 ;* 27 I DESIRED TRACK AND SECTOR          I 26
425 ;* 31 I DESTRED CYLINDER          I 30
426 ;* 33 I ATTENTION SUMMARY AND DRIVE OFFSET          I 32
427 ;* 35 I ERROR REGISTER          I 34
428 ;* 37 I STATUS REGISTER          I 36
429 ;* 41 I MESSAGE LINE A STATUS BYTE 00          I 40
430 ;* 43 I MESSAGE LINE R STATUS BYTE 00          I 42
431 ;* 45 I MESSAGE LINE A STATUS BYTE 01          I 44
432 ;* 47 I MESSAGE LINE B STATUS BYTE 01          I 46
433 ;* 51 I MESSAGE LINE A STATUS BYTE 10          I 50
434 ;* 53 I MESSAGE LINE R STATUS BYTE 10          I 52
435 ;* 55 I MESSAGE LINE A STATUS BYTE 11          I 54
436 ;* 57 I MESSAGE LINE R STATUS BYTE 11          I 56
437 ;* 61 I ECC POSITION INFORMATION          I 60
438 ;* 63 I ECC PATTERN INFORMATION          I 62
439 ;* *****
440
441 .SBTTL PARAMETERS PASSED TO THE DRIVER
442
443 ; THE FOLLOWING DEFINITIONS ARE USED TO PASS PARAMETERS
444 ; TO THE RK06 DRIVER
445
446 000000 P.DRVN= 0 ;DRIVE NUMBER
447 000001 P.CMND= 1 ;COMMAND
448 000002 P.CVLN= 2 ;CYLINDER ADDRESS
449 000004 P.SECT= 4 ;SECTOR
450 000005 P.TRCK= 5 ;TRACK
451 000006 P.OPST= 6 ;OFFSET
452 000007 P.CS1H= 7 ;RKCS1 BITS 8-15
453 000007 P.BAHT= 7 ;BUS ADDRESS (BITS 16 AND 17)
454 000010 P.BALC= 10 ;BUS ADDRESS (BITS 0-15)
455 000012 P.WC= 12 ;WORD COUNT (2'S COMPLEMENT)
456 000014 P.PRST= 14 ;PROGRAM DRIVE STATUS INFORMATION
457
458 .SBTTL PROGRAM DEVICE STATUS REGISTER DEFINITION
459
460 000001 DRVUSE= BIT0 ;DRIVE IN USE
461 000002 DRVPOS= BIT1 ;DRIVE POSITIONING
462 000004 DRV PDT= BIT2 ;DRIVE POSITIONED FOR DATA TRANSFER
463 000010 UEXATT= BIT3 ;UNEXPECTED ATTENTION
464 000020 DRVHRD= BIT4 ;DRIVE HAS HARD ERROR
465 000040 DRVDSC= BIT5 ;DRIVE STATUS CHANGE DID NOT CLEAR

```

```

466      000100      CMDTO=  BIT6      ;NO TERMINATION TO COMMAND FOR AT
467      ;          ; LEAST 1 SECOND
468      000200      W.WCK=  BIT7      ;WRITE FOR WRITE WRITE CHECK
469      000400      NOCHK=  BIT8      ;NO CHECK, DO NOT SET INTERRUPT ENABLP
470      001000      PRSVAL= BIT9      ;PARAMETER STATUS WORDS VALID
471      ;          ; (SET WHEN REGR TERMINATION C6
472      ;          ; READ STATUS COMMAND)
473      002000      DRPDRV= BIT10      ;DROP DRIVE FROM TEST SEQUENCE
474      004000      MODSC=  BIT11      ;ATTENTION SET BUT DCS AND FAULT RESET
475      010000      DRVSZD= BIT12      ;DRIVE SEIZED BY OTHER PORT
476      020000      E.UNLD= BIT13      ;DRIVE UNLOADED DUE TO ERROR
477      040000      Q.INIT= BIT14      ;PARAMETER BLOCK ENQUEUED IN INITIATION QUEUE
478      100000      DTBAIT= BIT15      ;INHIBIT BUS ADDRESS INCREMENT
479
480      .SBTTL  PARAMETERS PASSED FROM DRIVER TO PROGRAM
481
482      ;          THE FOLLOWING DEFINITIONS ARE USED FOR REGISTER RETURNS
483      ;          FROM THE DRIVER TO THE CALLING PROGRAM
484
485      000016      P.CS1=  16      ;COMMAND AND STATUS REGISTER 1
486      000020      P.CS2=  20      ;COMMAND AND STATUS REGISTER 2
487      000022      P.WCR=  22      ;WORD COUNT REGISTER
488      000024      P.BAR=  24      ;BUS ADDRESS REGISTER
489      000026      P.DTS=  26      ;DESIRED TRACK SECTOR REGISTER
490      000030      P.DCYL= 30      ;DESIRED CYLINDER REGISTER
491      000032      P.ASOF= 32      ;ATTENTION SUMMARY/OFFSET REGISTER
492      000034      P.ER=   34      ;ERROR REGISTER
493      000036      P.DS=   36      ;STATUS REGISTER
494      000040      P.A00=  40      ;MESSAGE A STATUS BYTE 00
495      000042      P.B00=  42      ;MESSAGE B STATUS BYTE 00
496      000044      P.A01=  44      ;MESSAGE A STATUS BYTE 01
497      000046      P.B01=  46      ;MESSAGE B STATUS BYTE 01
498      000050      P.A10=  50      ;MESSAGE A STATUS BYTE 10
499      000052      P.B10=  52      ;MESSAGE B STATUS BYTE 10
500      000054      P.A11=  54      ;MESSAGE A STATUS BYTE 11
501      000056      P.B11=  56      ;MESSAGE B STATUS BYTE 11
502      000060      P.EPOS= 60      ;ECC POSITION INFORMATION
503      000062      P.EPAT= 62      ;ECC PATTERN INFORMATION
504
505      .SBTTL  DRIVE SPECIFIC PARAMETERS
506
507      000064      P.QLNK= 64      ;QUEUE LINK

```

```

508      ;      THE FOLLOWING DEFINITIONS ARE USED FOR DRIVE PARAMETER ACCESS
509
510      000066      P.MNCL= 66      ;MINIMUM CYLINDER
511      000070      P.MXCL= 70      ;MAXIMUM CYLINDER
512      000072      P.MNTR= 72      ;MINIMUM TRACK
513      000073      P.MXTR= 73      ;MAXIMUM TRACK
514      000074      P.MNSC= 74      ;MINIMUM SECTOR
515      000075      P.MXSC= 75      ;MAXIMUM SECTOR
516      000076      P.RATE= 76      ;READ/WRITE RATIO
517      000077      P.AWCK= 77      ;AUTOMATIC WRITE CHECK
518      000100      P.CERT= 100      ;CORRECTABLE HEAD ERROR THRESHOLD
519      000102      P.UBRT= 102      ;UNCORRECTABLE READ ERROR THRESHOLD
520      000104      P.SKET= 104      ;SEEK ERROR THRESHOLD
521      000106      P.MXCD= 106      ;MAXIMUM COMMANDS ISSUED TO DRIVE
522      000112      P.MXMT= 112      ;MAXIMUM NUMBER WORDS TRANSFERRED
523      000116      P.SMPL= 116      ;SAMPLE COMPARE FLAG
524      000117      P.ECMP= 117      ;ECC COMPARE FLAG
525      000120      P.IERC= 120      ;INHIBIT ERROR CORRECTION
526      000121      P.DPAT= 121      ;DATA PATTERN USED
527
528      ;      THE FOLLOWING DEFINITIONS ARE USED IN STORING
529      ;      CURRENT RANDOMLY GENERATED COMMAND
530
531      000122      P.RDPT= 122      ;CURRENT RANDOMLY GENERATED DATA PATTERN
532      000123      P.RCMD= 123      ;CURRENT RANDOMLY GENERATED COMMAND
533      000124      P.RCYL= 124      ;CURRENT RANDOMLY GENERATED CYLINDER
534      000126      P.RSEC= 126      ;CURRENT RANDOMLY GENERATED SECTOR
535      000127      P.RTRK= 127      ;CURRENT RANDOMLY GENERATED TRACK
536      000130      P.RWC= 130      ;CURRENT RANDOMLY GENERATED WORD COUNT
537      000132      P.RBAL= 132      ;CURRENT RANDOMLY GENERATED BUS ADDRESS
538      000134      P.RBAH= 134
539
540      ;      THE FOLLOWING DEFINITIONS ARE USED IN STORING
541      ;      LAST ISSUED
542
543      000135      P.LCMD= 135      ;LAST COMMAND
544      000136      P.LCYL= 136      ;LAST CYLINDER
545      000140      P.LSEC= 140      ;LAST SECTOR
546      000141      P.LTRK= 141      ;LAST TRACK
547      000142      P.LWC= 142      ;LAST WORD COUNT
548      000144      P.LDPT= 144      ;LAST DATA PATTERN
549      000145      P.BUFF= 145      ;BUFFER ALLOCATED

```

```

550      )      THE FOLLOWING IS USED FOR THE RETRY SEQUENCE
551
552      000146      P.RFCT= 146      ;RETRY COUNT
553      000147      P.RERD= 147      ;RETRY INDICATION FOR DATA ERRORS
554
555      ;0          NOT IN RETRY SEQUENCE
556      ;1-20      REREAD AT CENTER LINE
557      ;21        OFFSET +400 MICRO-INCHES
558      ;22-23     REREAD OFFSET +400 MICRO-INCHES
559      ;24        OFFSET -400 MICRO-INCHES
560      ;25-26     REREAD OFFSET -400 MICRO-INCHES
561      ;27        OFFSET +800 MICRO-INCHES
562      ;30-31     REREAD OFFSET +800 MICRO-INCHES
563      ;32        OFFSET -800 MICRO-INCHES
564      ;33-34     REREAD OFFSET -800 MICRO-INCHES
565      ;35        OFFSET +1200 MICRO-INCHES
566      ;36-37     REREAD OFFSET +1200 MICRO-INCHES
567      ;40        OFFSET -1200 MICRO-INCHES
568      ;41-42     REREAD OFFSET -1200 MICRO-INCHES
569      000150      P.DSTT= 150      ;DRIVE STATUS INFORMATION FOR ERROR PROCESSING
570
571      ;BIT 0      INITIAL DATA ERROR
572      ;BIT 1      DATA ERROR BEING PROCESSED
573      ;BIT 2      NON-DATA ERROR BEING PROCESSED
574      ;BIT 3      SECTOR ID ERROR PRINTED
575      ;BIT 4      READ HEADER ISSUED
576      ;BIT 6      WAITING FOR READ STATUS FOR
577      ;           POSITIONING TIME OUT
578      ;BIT 7      FIRST ERROR
579      ;BIT 11     RECALIBRATE HAS BEEN ISSUED
580      ;BIT 12     SEEK HAS BEEN ISSUED
581      ;BIT 13     OFFSET HAS BEEN ISSUED
582      ;BIT 14     BAD SECTOR PROCESSING FOR WRITE CHECK
583      ;BIT 15     ERROR ENGUEUED
584
585      )      THE FOLLOWING DEFINITIONS ARE USED IN STATISTICAL GATHERING
586
587      000152      P.NODR= 152      ;NUMBER OF CADDERS (2 WORDS)
588      000156      P.NWRT= 156      ;NUMBER OF WORDS WRITTEN (3 WORDS)
589      000164      P.NRD= 164      ;NUMBER OF WORDS READ (3 WORDS)
590      000172      P.SRRD= 172      ;NUMBER SOFT ERRORS (REPEAT CORRECTABLE)
591      000176      P.SOFP= 174      ;NUMBER SOFT ERRORS (OFFSET CORRECTABLE)
592      000200      P.SECC= 176      ;NUMBER SOFT ERRORS (ECC CORRECTABLE)
593      000202      P.HARD= 200      ;NUMBER OF HARD DATA ERRORS
594      000204      P.NSKI= 202      ;NUMBER OF SEEK INCOMPLETE
595      000206      P.NOPI= 204      ;NUMBER OF OPERATION INCOMPLETES
596      000206      P.NER= 206      ;NUMBER OF ALL OTHER ERRORS
  
```

```

596      ;      THE FOLLOWING IS USED FOR THE DRIVE ASSIGNMENT SEQUENCE
597
598      000210      P.ASSN= 210      ;ASSIGNMENT COMMAND SEQUENCE
599
600      ;0          INITIAL STATE
601      ;BIT 0      DRIVE SERIAL NUMBER
602      ;BIT 1      START SPINDLE
603      ;BIT 2      PACK ACKNOWLEDGE
604      ;BIT 3      READ PACK SERIAL NUMBER
605      ;BIT 4      WRITE PACK COMMAND ISSUED
606      ;BIT 5      END OF PACK WRITE
607      ;BIT 6      RECAL
608      ;BIT 7      RECAL AND RETRY READ PACK
609      ;          SERIAL NUMBER
610
611      000211      P.SEEK= 211      ;SEEK TO PREVIOUS CYLINDER FLAG
612
613      ;      THE FOLLOWING LOCATION IS USED TO INDICATE ERRORS
614
615      000212      P.ERR= 212      ;BIT 2  DATA COMPARISON ERROR
616
617      ;BIT 3  DRIVE ERROR NOT REPORTED
618      ;BIT 4  DRIVE SEIZED TIME OUT
619      ;BIT 5  DRIVE POSITIONING TIME OUT
620      ;BIT 6  ERROR WHILE ENQUEUED
621      ;BIT 7  ATTENTION WHEN DRIVE NOT IN USE
622      ;BIT 8  TIME OUT WHILE WAITING FOR HEADS
623      ;          TO LOAD AFTER ERROR
624      ;BIT 10 DRIVE NOT READY AFTER START SPINDLE
625      ;BIT 11 PACK ACKNOWLEDGE DID NOT SET VOLUME
626      ;          VALID
627
628      000214      P.ERHD= 214      ;HEADER ADDRESS OF FIRST DATA MISMATCH (3 BYTES)
629      000217      P.ERAD= 217      ;OFFSET COUNT FROM HEADER ADDRESS
630      000220      P.CMLG= 220      ;COMPARISON LENGTH
631      000222      P.BFCP= 222      ;BUFFER COMPARE LENGTH
632
633      ;      THE FOLLOWING LOCATIONS ARE USED FOR DRIVE SERIAL NUMBER,
634      ;      PACK SERIAL NUMBER, AND TRACK EXCLUSIONS
635
636      000224      P.SERL= 224      ;DRIVE SERIAL NUMBER
637      000226      P.PKSR= 226      ;CARTRIDGE SERIAL NUMBER
638      000237      P.EXAR= 232      ;EXCLUDED AREA INFORMATION

```

```

637      .SBTTL  TPAP CATCHER
638
639      000000      .=0
640      ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
641      ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
642      ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
643      000174      .=174
644      000174      000000      DISPREG: .WORD 0      ;;SOFTWARE DISPLAY REGISTER
645      000176      000000      SWRFG: .WORD 0      ;;SOFTWARE SWITCH REGISTER
646
647      000200      000137      006656      .SBTTL  STARTING ADDRESS(ES)
648      000204      000137      006616      JMP      @START ;;JUMP TO STARTING ADDRESS OF PROGRAM
649      000214      000214      JMP      RESTRT      ;BRANCH TO RESTART
650      000214      000137      006636      .=214
651      000220      000137      061052      JMP      PARSRT      ;GET SYSTEM PARAMETERS AND START TESTING
652      000224      000700      JMP      ..DUMP      ; *** MEMORY DUMP (DEBUG ONLY)
653      000226      004574      ..LOW: 700      ;START OF DUMP
654      001000      001000      ..HIGH: PATTAB-7      ;LAST ADDRESS OF DUMP
655      .=1000
656      .SBTTL  APT PARAMETER BLOCK
657      ;;*****
658      ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
659      ;;*****
660      001000      .SX=.      ;;SAVE CURRENT LOCATION
661      000024      .=24      ;;SET POWER FAIL TO POINT TO START OF PROGRAM
662      000024      000200      200      ;;FOR APT START UP
663      000044      000044      .=44      ;;POINT TO APT INDIRECT ADDRESS PTR.
664      000044      001000      $APTHDR ;;POINT TO APT HEADER BLOCK
665      001000      .=.SX      ;;RESET LOCATION COUNTER
666      ;;*****
667      ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
668      ;INTERFACE SPEC.
669
670      001000      $APTHD:
671      001000      000000      $HINTS: .WORD 0      ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
672      001002      001170      $MBADR: .WORD $MAIL      ;;ADDRESS OF APT MAILBOX (BITS 0-15)
673      001004      003000      $TSTM: .WORD 3000      ;;RUN TIME OF LONGEST TEST
674      001006      003000      $PASTM: .WORD 3000      ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
675      001010      003000      $UNITM: .WORD 3000      ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
676      001012      000052      .WORD $FTEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)

```

677
678
679
680
681
682
683 001100
684 001100 000000
685 001100 000000
686 001102 000
687 001103 000
688 001104 000000
689 001106 000000
690 001110 000000
691 001112 000000
692 001114 000
693 001115 001
694 001116 000000
695 001120 000000
696 001122 000000
697 001124 000000
698 001126 000000
699 001130 000000
700 001132 000000
701 001134 000
702 001135 000
703 001136 000000
704 001140 177570
705 001142 177570
706 001144 177560
707 001146 177562
708 001150 177564
709 001152 177566
710 001154 000
711 001155 002
712 001156 012
713 001157 000
714 001160 177607 000377
715 001164 077
716 001165 015
717 001166 000012
718
719
720
721
722
723 001170
724 001170 000000
725 001172 000000
726 001174 000000
727 001176 000000
728 001200 000000
729 001202 000000
730 001204 000000
731 001206 000000
732 001210

.SBTTL COMMON TAGS

;;*****
;THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
;USED IN THE PROGRAM.

.=1100

\$CHTAG: .WORD 0 ;;START OF COMMON TAGS
\$TSTNM: .BYTE 0 ;;CONTAINS THE TEST NUMBER
\$ERPLG: .BYTE 0 ;;CONTAINS BRPCR FLAG
\$ICNT: .WORD 0 ;;CONTAINS SUBTEST ITERATION COUNT
\$LPADR: .WORD 0 ;;CONTAINS SCOPE LOOP ADDRESS
\$LPERR: .WORD 0 ;;CONTAINS SCOPE RETURN FOR BRPCBS
\$ERTTL: .WORD 0 ;;CONTAINS TOTAL ERRORS DETECTED
\$ITEMB: .BYTE 0 ;;CONTAINS ITEM CONTROL BYTE
\$ERMAX: .BYTE 1 ;;CONTAINS MAX. ERRORS PER TEST
\$ERRPC: .WORD 0 ;;CONTAINS PC OF LAST ERROR INSTRUCTION
\$GDADR: .WORD 0 ;;CONTAINS ADDRESS OF "GOOD" DATA
\$BDADR: .WORD 0 ;;CONTAINS ADDRESS OF "BAD" DATA
\$GDDAT: .WORD 0 ;;CONTAINS "GOOD" DATA
\$BDDAT: .WORD 0 ;;CONTAINS "BAD" DATA
 .WORD 0 ;;RESERVED--NOT TO BE USED
\$AUTOR: .BYTE 0 ;;AUTOMATIC MODE INDICATOR
\$INTAG: .BYTE 0 ;;INTERUPT MODE INDICATOR
 .WORD 0
\$WR: .WORD DSWR ;;ADDRESS OF SWITCH REGISTER
\$DISPLAY: .WORD DDISP ;;ADDRESS OF DISPLAY REGISTER
\$TKS: 177560 ;;TTY KBD STATUS
\$TKB: 177562 ;;TTY KBD BUFFER
\$TPS: 177564 ;;TTY PRINTER STATUS REG. ADDRESS
\$TPR: 177566 ;;TTY PRINTER BUFFER REG. ADDRESS
\$NULL: .BYTE 0 ;;CONTAINS NULL CHARACTER FOR FILLS
\$FILLS: .BYTE 2 ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
\$FILLC: .BYTE 12 ;;INSERT FILL CHARS. AFTER A "LINE FEED"
\$TPFLG: .BYTE 0 ;;"TERMINAL AVAILABLE" FLAG (BIT<6>=0=YES)
\$BELL: .ASCII <207><377><377> ;;CODE FOR BELL
\$QUES: .ASCII /?/ ;;QUESTION MARK
\$CRLF: .ASCII <15> ;;CARRIAGE RETURN
\$LF: .ASCII <12> ;;LINE FEED
;;*****

.SBTTL APT MAILBOX-ETABLE

;;*****
.EVEN
\$MAIL: ;;APT MAILBOX
\$MSGTY: .WORD AMSGTY ;;MESSAGE TYPE CODE
\$FATAL: .WORD AFATAL ;;FATAL ERROR NUMBER
\$TESTN: .WORD ATESTN ;;TEST NUMBER
\$PASS: .WORD APASS ;;PASS COUNT
\$DEVCT: .WORD ADEVCT ;;DEVICE COUNT
\$UNIT: .WORD AUNIT ;;I/O UNIT NUMBER
\$MSGAD: .WORD AMSCAD ;;MESSAGE ADDRESS
\$MSGLC: .WORD AMSGLC ;;MESSAGE LENGTH
\$ETABLE: ;;APT ENVIRONMENT TABLE

```
733 001210 000 SFNV: .BYTE AENV ;;ENVIRONMENT BYTE
734 001211 000 SFNVH: .BYTE AENVH ;;ENVIRONMENT MODF BITS
735 001212 000000 $SWREG: .WORD ASWREG ;;APT SWITCH REGISTER
736 001214 000000 $USWR: .WORD AUSWR ;;USER SWITCHES
737 001216 000000 $CPUOP: .WORD ACPUOP ;;CPU TYPE,OPTIONS
738 ;* BITS 15-11=CPU TYPE
739 ;* 11/04=01,11/05=02,11/70-03,11/40=C4,11/45=C5
740 ;* 11/70=06,P0Q=07,Q=10
741 ;* BIT 10=REAL TIME CLOCK
742 ;* BIT 9=FLOATING POINT PROCESSOR
743 ;* BIT 8=MEMORY MANAGEMENT
744 001220 000 $MAMS1: .BYTE AMAMS1 ;;HIGH ADDRESS,M.S. BYTE
745 001221 000 $MTYP1: .BYTE AMTYP1 ;;MFH. TYPE,BLK#1
746 ;* MEM.TYPE BYTE -- (HIGH BYTE)
747 ;* 900 NSEC CORE=001
748 ;* 300 NSEC RIPCLEAR=002
749 ;* 500 NSEC MOS=003
750 001222 000000 $MADR1: .WORD AMADR1 ;;HIGH ADDRESS,BLK#1
751 ;* MEM.LAST ADDR.=3 BYTE, THIS WORD AND LOW CP "TYPE" RECVF
752 001224 000 $MAMS2: .BYTE AMAMS2 ;;HIGH ADDRESS,M.S. BYTE
753 001225 000 $MTYP2: .BYTE AMTYP2 ;;MFH.TYPE,PLK#2
754 001226 000000 $MADR2: .WORD AMADR2 ;;MEM.LAST ADDRESS,BLK#2
755 001230 000 $MAMS3: .BYTE AMAMS3 ;;HIGH ADDRESS,M.S.BYTE
756 001231 000 $MTYP3: .BYTE AMTYP3 ;;MEM.TYPE,PLK#3
757 001232 000000 $MADR3: .WORD AMADR3 ;;MEM.LAST ADDRESS,BLK#3
758 001234 000 $MAMS4: .BYTE AMAMS4 ;;HIGH ADDRESS,M.S.BYTE
759 001235 000 $MTYP4: .BYTE AMTYP4 ;;MEM.TYPE,BLK#4
760 001236 000000 $MADR4: .WORD AMADR4 ;;MEM.LAST ADDRESS,BLK#4
761 001240 120210 $VECT1: .WORD AVECT1 ;;INTERRUPT VECTOR#1,PUS PRIORITY#1
762 001242 000000 $VECT2: .WORD AVECT2 ;;INTERRUPT VECTOR#2BUS PRIORITY#2
763 001244 177440 $BASE: .WORD ABASE ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
764 001246 000000 $DEVH: .WORD ADEVH ;;DEVICE MAP
765 001250 000000 $CDW1: .WORD ACDW1 ;;CONTROLLER DESCRIPTION WORD#1
766 001252 000000 $CDW2: .WORD ACDW2 ;;CONTROLLER DESCRIPTION WORD#2
767 001254 000000 $DDW0: .WORD ADDW0 ;;DEVICE DESCRIPTOR WORD#0
768 001256 000000 $DDW1: .WORD ADDW1 ;;DEVICE DESCRIPTOR WORD#1
769 001260 000000 $DDW2: .WORD ADDW2 ;;DEVICE DESCRIPTOR WORD#2
770 001262 000000 $DDW3: .WORD ADDW3 ;;DEVICE DESCRIPTOR WORD#3
771 001264 000000 $DDW4: .WORD ADDW4 ;;DEVICE DESCRIPTOR WORD#4
772 001266 000000 $DDW5: .WORD ADDW5 ;;DEVICE DESCRIPTOR WORD#5
773 001270 000000 $DDW6: .WORD ADDW6 ;;DEVICE DESCRIPTOR WORD#6
774 001272 000000 $DDW7: .WORD ADDW7 ;;DEVICE DESCRIPTOR WORD#7
775 001274 000000 $DDW8: .WORD ADDW8 ;;DEVICE DESCRIPTOR WORD#8
776 001276 000000 $DDW9: .WORD ADDW9 ;;DEVICE DESCRIPTOR WORD#9
777 001300 000000 $DDW10: .WORD ADDW10 ;;DEVICE DESCRIPTOR WORD#10
778 001302 000000 $DDW11: .WORD ADDW11 ;;DEVICE DESCRIPTOR WORD#11
779 001304 000000 $DDW12: .WORD ADDW12 ;;DEVICE DESCRIPTOR WORD#12
780 001306 000000 $DDW13: .WORD ADDW13 ;;DEVICE DESCRIPTOR WORD#13
781 001310 000000 $DDW14: .WORD ADDW14 ;;DEVICE DESCRIPTOR WORD#14
782 001312 000000 $DDW15: .WORD ADDW15 ;;DEVICE DESCRIPTOR WORD#15
783
784
785 001314 SETEND:
786
```

```

787 .SBTTL QUEUE HEAD AND TAIL ALLOCATIONS
788
789 001314 000000 000000 AVAILQ: .WORD 0,0 ;HEAD AND TAIL OF AVAILABLE QUEUE
790 001320 000000 000000 BWAITQ: .WORD 0,0 ;HEAD AND TAIL OF BUFFER WAIT QUEUE
791 001324 000000 000000 CINITQ: .WORD 0,0 ;HEAD AND TAIL OF COMMAND INITIATION QUEUE
792 001330 000000 000000 ERRPRQ: .WORD 0,0 ;HEAD AND TAIL OF ERROR PROCESSING QUEUE
793
794 .SBTTL SYSTEM PARAMETERS
795
796 001334 172540 $LKCSF: .WORD 172540 ;ADDRESS OF KW11-P STATUS REGISTER
797 001336 172542 $LKCSB: .WORD 172542 ;ADDRESS OF KW11-P COUNTER BUFFER
798 001340 000104 $LPVEC: .WORD 104 ;KW11-P VECTOR ADDRESS
799 001342 177546 $LKS: .WORD 177546 ;ADDRESS OF KW11-L STATUS REGISTER
800 001344 000100 $LLVEC: .WORD 100 ;KW11-L VECTOR ADDRESS
801 001346 177514 $LSCS: .WORD 177514 ;ADDRESS OF PRINTER STATUS WORD
802 001350 177516 $LSDB: .WORD 177516 ;ADDRESS OF PRINTER DATA BUFFER
803 001352 074 HZ: .BYTE 60. ;74 (60 DECIMAL) IF SYSTEM IS 60 HZ.
804 ;62 (50 DECIMAL) IF SYSTEM IS 50 HZ.
805 001353 000 CLKFRQ: .BYTE 0 ;COUNT FOR SECOND
806 001354 013400 MAXBUF: .WORD 256.*73. ;MAXIMUM BUFFER FOR READ OR WRITE
807 001356 177777 PASCNT: .WORD 177777 ;PASS COUNT
808 001360 000 PERINT: .BYTE 0 ;INTERVAL BETWEEN PERFORMANCE SUMMARIES
809 001361 000 FLAG: .BYTE 0 ;RESTART AND PARAMETER FLAG
810 001362 000003 SOFCMP: .WORD 3 ;NUMBER OF SOFTWARE COMPARES
811 001364 000 CPUID: .BYTE 0 ;CPU DESIGNATION FOR PACK WRITING WITH DUAL ACCESS
812 001365 000 OVLYLD: .BYTE 0 ;OVERLAY LOADER
813 001366 000000 HOUR: .WORD 0 ;TIMER HOUR
814 001370 000000 MINUTE: .WORD 0 ;TIMER MINUTE
815 001372 000000 SECOND: .WORD 0 ;TIMER SECOND
816 001374 000 CLKPLG: .BYTE 0 ;KW11-P OR KW11-L PRESENT
817 001375 000 PERIOD: .BYTE 0 ;PERIOD SINCE LAST STATISTIC PRINT OUT
818 001376 000000 TIMHR: .WORD 0 ;HOUR COUNT FOR PRINT OUT
819 001400 000000 TIMMIN: .WORD 0 ;MINUTE COUNT FOR PRINT OUT
820 001402 000000 TIMSEC: .WORD 0 ;SECOND COUNT FOR PRINT OUT
821 001404 000000 SAVSWR: .WORD 0 ;SAVED SWITCH REGISTER FROM POWER FAIL
822
823 .SBTTL TEMPORARY CONTROLLER REGISTER STORAGE
824
825 001406 000000 T.CS1: .WORD 0 ;TEMPORARY STORAGE FOR COMMAND AND STATUS
826 ; REGISTER 1
827 001410 000000 T.CS2: .WORD 0 ;TEMPORARY STORAGE FOR COMMAND AND STATUS
828 ; REGISTER 2
829 001412 000000 T.WCR: .WORD 0 ;TEMPORARY STORAGE FOR WORD COUNT REGISTER
830 001414 000000 T.BA: .WORD 0 ;TEMPORARY STORAGE FOR BUS ADDRESS REGISTER
831 001416 000000 T.DA: .WORD 0 ;TEMPORARY STORAGE FOR DISK TRACK AND SECTOR
832 001420 000000 T.DC: .WORD 0 ;TEMPORARY STORAGE FOR DRIVE CYLINDER
833 001422 000000 T.ASCF: .WORD 0 ;TEMPORARY STORAGE FOR ATTENTION SUMMARY
834 ; AND OFFSET
835 001424 000000 T.ER: .WORD 0 ;TEMPORARY STORAGE FOR ERROR REGISTER
836 001426 000000 T.DS: .WORD 0 ;TEMPORARY STORAGE FOR DRIVE STATUS REGISTER
837 001430 000000 T.MR1: .WORD 0 ;TEMPORARY STORAGE FOR MAINTENANCE REGISTER 1
838 001432 000000 T.MR2: .WORD 0 ;TEMPORARY STORAGE FOR MAINTENANCE REGISTER 2
839 001434 000000 T.MR3: .WORD 0 ;TEMPORARY STORAGE FOR MAINTENANCE REGISTER 3
840 001436 000000 T.POS: .WORD 0 ;TEMPORARY STORAGE FOR ECC POSITION
841 001440 000000 T.PAT: .WORD 0 ;TEMPORARY STORAGE FOR FCC PATTERN
842 001442 000000 T.DR: .WORD 0 ;TEMPORARY STORAGE FOR DATA BUFFER REGISTER

```

```
843
844
845
846 001444 177440
847 001446 000210
848 001450 000240
849 001452 023676
850 001454 024766
851 001456 026022
852 001460 000000
853
854
855
856
857
858      000001
859      000002
860      000004
861      000010
862      000020
863      000040
864
865      000100
866      000400
867      001000
868      002000
869      040000
870      100000
871
872 001462 000000
873
874 001464 000400
875 001466 000400
876
877
878
879
880
881
882
883
884
885
886
887
888 001470 000100
889
890 001472 001000
891 001474 010000
892 001476 000000
893 001500 000000
894
895 001502      000
896 001503      002
897 001506      000
898 001507      377
```

```
      .SBTTL  DRIVER PARAMETERS

RKBAS:  .WORD  177440
RKVEC:  .WORD  210
RKPRI:  .WORD  PR5
A.NORM:  NORMAL
A.ABNL:  ABNORM
A.CONT:  CONTRL
E.CONT:  .WORD  0

E.CCLP= BIT0
E.NOAT= BIT1
E.UATT= BIT2
E.UDAT= BIT3
E.CLAT= BIT4
E.SCLP= BIT5

E.ILLD= BIT6
E.DLT=  BIT7
E.CERR= BIT9
E.DPAR= BIT10
E.CMTO= BIT14
E.MDS=  BIT15

O.WAIT:  .WORD  0

W.MTIM:  .WORD  400
W.MILT:  .WORD  400

O.WAIT:  .WORD  100
W.8SEC:  .WORD  1000
W.MIN:   .WORD  10000
HDR.AD:  .WORD  0
HDR.CT:  .WORD  0

I.ISRL:  .BYTE  0
H.HEAD:  .BYTE  2,4,10
W.TIME:  .PYTE  0
O.OVER:  .BYTE  -1
```

```
;ADDRESS OF RK611 UNIBUS ADDRESS BLOCK
;ADDRESS OF 6611 VECTOR
;RK611 INTERRUPT PRIORITY
;ADDRESS OF NORMAL RETURN FROM DRIVER
;ADDRESS OF ABNORMAL RETURN FROM DRIVER
;ADDRESS OF CONTROLLER ERROR RETURN
;CONTROLLER ERROR STATUS
; THIS LOCATION IS CLEARED WHEN EVERY COMMAND
; IS INITIATED. IF A CONTROLLED ERROR
; OCCURS THE FOLLOWING BIT ASSIGNMENT IS
; USED:
;
;CLEAR CONTROLLER DID NOT CLEAR ERROR
;NO ATTENTION IN ATTENTION SUMMARY REG
;UNSLICATED ATTENTION (SEQUENTIAL ONLY)
;UNEXPECTED DATA TYPE ERROR
;ATTENTION DID NOT RESET WITH CLEAR
;SUBSYSTEM CLEAR DID NOT CLEAR DRIVE
; ATTENTION
;ILLEGAL DRIVER COMMAND
;DATA LATE WHEN UNLOADING HEADER
;CONTROLLER ERROR DURING DRIVER SERVICING
;DRIVE DETECTED PARITY ERROR
;CONTROLLER COMMAND TIME OUT (QUEUED ONLY)
;MULTIPLE DRIVE SELECT

;PARAMETER BLOCK OF THE DRIVE
;WAITING FOR COMMAND COMPLETION
;LOOP COUNTER FOR MILLISECOND SCAN OF DRIVE
;16 MILLISECOND TIME FOR PROGRAM
;
;      CPU      VALUE
;      ---      -----
;
;      11/05     100
;      11/10
;      11/20
;      11/34
;      11/40
;      11/45     400
;      11/50
;      11/70
;SECOND COUNT COUNT FOR ALL COMMANDS
; EXCEPT START SPINDLE
;A SECOND FOR FIVE CYCLE CCMW
;MINUTE TIME FOR START SPINDLE
;ADDRESS USED FOR READ ALL HEADERS
;NUMBER OF HEADERS LEFT TO READ FOR READ
; ALL HEADERS
;INTERRUPT ON RELEASED COMMAND ISSUED
;HEAD OFCODES
;DRIVES BEING WATCH-DOG TIMED
;OVERLAPPED OPERATIONS
```

```

999
900      .SBTTL  TABLE OF INTERRUPT MASKS
901
902      001510      001      INTMSK: .BYTE      1      ;INTERRUPT FOR DRIVE 0
903      001511      002      .BYTE      2      ;INTERRUPT FOR DRIVE 1
904      001512      004      .BYTE      4      ;INTERRUPT FOR DRIVE 2
905      001513      010      .BYTE     10      ;INTERRUPT FOR DRIVE 3
906      001514      020      .BYTE     20      ;INTERRUPT FOR DRIVE 4
907      001515      040      .BYTE     40      ;INTERRUPT FOR DRIVE 5
908      001516      100      .BYTE    100      ;INTERRUPT FOR DRIVE 6
909      001517      200      .BYTE    200      ;INTERRUPT FOR DRIVE 7
910
911      .SBTTL  PARAMETER BLOCK TABLE
912
913      001520      001732      PRLKT:  PARM0      ;ADDRESS OF DRIVE 0'S PARAMETER BLOCK
914      001522      002710      .        PARM1      ;ADDRESS OF DRIVE 1'S PARAMETER BLOCK
915      001524      002466      .        PARM2      ;ADDRESS OF DRIVE 2'S PARAMETER BLOCK
916      001526      002744      .        PARM3      ;ADDRESS OF DRIVE 3'S PARAMETER BLOCK
917      001530      003222      .        PARM4      ;ADDRESS OF DRIVE 4'S PARAMETER BLOCK
918      001532      003500      .        PARM5      ;ADDRESS OF DRIVE 5'S PARAMETER BLOCK
919      001534      003756      .        PARM6      ;ADDRESS OF DRIVE 6'S PARAMETER BLOCK
920      001536      004734      .        PARM7      ;ADDRESS OF DRIVE 7'S PARAMETER BLOCK
921
922      .SBTTL  WATCH-DOG TIMER COUNTS
923
924      001540      000000      W.DRV:  .WORD      0      ;WATCH-DOG FOR DRIVE 0
925      001542      000000      .        .WORD      0      ;WATCH-DOG FOR DRIVE 1
926      001544      000000      .        .WORD      0      ;WATCH-DOG FOR DRIVE 2
927      001546      000000      .        .WORD      0      ;WATCH-DOG FOR DRIVE 3
928      001550      000000      .        .WORD      0      ;WATCH-DOG FOR DRIVE 4
929      001552      000000      .        .WORD      0      ;WATCH-DOG FOR DRIVE 5
930      001554      000000      .        .WORD      0      ;WATCH-DOG FOR DRIVE 6
931      001556      000000      .        .WORD      0      ;WATCH-DOG FOR DRIVE 7

```

```

932 .SBTTL PERFORMANCE EXERCISER STATUS INFORMATION
933
934 001560 000000 ERCONT: .WORD 0 ;CONTROLLER ERROR (RECOVERABLE)
935 ;BIT 0 CONTROLLER ERROR NOT FLAGGED
936 ;BIT 1 WORD COUNT NOT EQUAL 0
937 ;BIT 2 BUS ADDRESS INCORRECT
938 ;BIT 3 CYLINDER, TRACK, SECTOR INCORRECT
939 ;BIT 15 CONTROLLER ERROR DETECTED
940
941 001562 000000 ERRPRO: .WORD 0 ;ADDRESS OF PARAMETER BLOCK PROCESSING ERROR
942 001564 000000 TMPCNT: .WORD 0 ;TRANSFER COUNT CALCULATIONS
943 001566 000000 000000 TMPTRK: .WORD 0,0 ;RANDOM TRACK CALCULATIONS
944 001572 000000 000000 TMPSEC: .WORD 0,0 ;RANDOM SECTOR CALCULATIONS
945 001576 000000 000000 PARCYL: .WORD 0,0 ;PARAMETER CYLINDER MODIFICATION
946 001602 000000 000000 PARTRK: .WORD 0,0 ;PARAMETER TRACK MODIFICATION
947 001606 000000 000000 PARSEC: .WORD 0,0 ;PARAMETER SECTOR MODIFICATION
948 001612 000000 000000 EXAREA: .WORD 0,0 ;EXCLUDED AREA CALCULATIONS
949 001616 000000 000000 ECCPAT: .WORD 0,0 ;ECC PATTERN FOR ECC CORRECTION
950 001622 000000 DLTCNT: .WORD 0 ;DATA LATE COUNT
951 001624 000000 CNTCNT: .WORD 0 ;OTHER CONTROLLER ERRORS COUNT
952 001626 000 DRVCMT: .BYTE 0 ;DRIVE COUNT FOR OPERATOR COMPARE
953 001627 000 STATIS: .BYTE 0 ;REPORT INTERVAL STATISTICS
954 001630 000000 RTYBA: .WORD 0 ;RETRY BUS ADDRESS
955 001632 000000 RTYWC: .WORD 0 ;RETRY WORD COUNT
956 001634 000 RTYSEC: .BYTE 0 ;SECTOR ADDRESS RETRY
957 001635 000 RTYTRK: .BYTE 0 ;TRACK ADDRESS RETRY
958 001636 000000 RTYCYL: .WORD 0 ;CYLINDER ADDRESS RETRY
959 001640 000 RTYSCN: .BYTE 0 ;NUMBER OF SECTORS TO BE TRANSFERRED IN RETRY
960 001641 000 RTYCMD: .BYTE 0 ;COMMAND BEING RETRIED
961 001642 000000 HEAD1: .WORD 0 ;TEMPORARY STORAGE FOR FIRST WORD OF HEADER
962 001644 000000 HEAD2: .WORD 0 ;TEMPORARY STORAGE FOR SECOND WORD OF HEADER
963 001646 000000 HEAD3: .WORD 0 ;TEMPORARY STORAGE FOR THIRD WORD OF HEADER
964 001650 000 BSSECT: .BYTE 0 ;SECTOR IN BAD SECTOR SERVICING
965 001651 000 BSTRCK: .BYTE 0 ;TRACK IN BAD SECTOR SERVICING
966 001652 000000 BSCYLW: .WORD 0 ;CYLINDER IN BAD SECTOR SERVICING
967 001654 000000 BSMC: .WORD 0 ;WORD COUNT IN BAD SECTOR SERVICING
968 001656 000000 BSBA: .WORD 0 ;BUS ADDRESS IN BAD SECTOR SERVICING
969 001660 000000 BSWORD: .WORD 0 ;WORDS TRANSFERRED IN BAD SECTOR SERVICING
970 001662 000000 POSCMD: .WORD 0 ;POSITIONING TIME OUT COMMAND
971
972 .SBTTL FREE SPACE BLOCK
973
974 001664 000001 FBLKFC: .WORD 1 ;FREE BLOCK COUNT
975
976 001666 000000 000000 FBLK1: .WORD 0,0 ;FREE BLOCK 0
977 001672 000000 000000 .WORD 0,0 ;FREE BLOCK 1
978 0 576 000000 000000 .WORD 0,0 ;FREE BLOCK 2
979 001702 000000 000000 .WORD 0,0 ;FREE BLOCK 3
980 001706 000000 000000 .WORD 0,0 ;FREE BLOCK 4
981 001712 000000 000000 .WORD 0,0 ;FREE BLOCK 5
982 001716 000000 000000 .WORD 0,0 ;FREE BLOCK 6
983 001722 000000 000000 .WORD 0,0 ;FREE BLOCK 7
984 001726 000000 000000 .WORD 0,0 ;FREE BLOCK 8

```

.SBTTL PARAMETER BLOCK FOR DRIVE 0

985					
986					
987					
988	001732	000	PARM0:	.BYTE	0
989	001733	000		.BYTE	0
990	001734	000000		.WORD	0
991	001736	000		.BYTE	0
992	001737	000		.BYTE	0
993	001740	000		.BYTE	0
994	001741	000		.BYTE	0
995	001742	000000		.WORD	0
996	001744	000000		.WORD	0
997	001746	000000		.WORD	0
998	001750	000000		.WORD	0
999	001752	000000		.WORD	0
1000	001754	000000		.WORD	0
1001	001756	000000		.WORD	0
1002	001760	000000		.WORD	0
1003	001762	000000		.WORD	0
1004	001764	000000		.WORD	0
1005	001766	000000		.WORD	0
1006	001770	000000		.WORD	0
1007	001772	000000		.WORD	0
1008	001774	000000		.WORD	0
1009	001776	000000		.WORD	0
1010	002000	000000		.WORD	0
1011	002002	000000		.WORD	0
1012	002004	000000		.WORD	0
1013	002006	000000		.WORD	0
1014	002010	000000		.WORD	0
1015	002012	000000		.WORD	0
1016	002014	000000		.WORD	0
1017	002016	000000		.WORD	0
1018	002020	000000		.WORD	0
1019	002022	000632		.WORD	410.
1020	002024	000		.BYTE	0
1021	002025	002		.BYTE	2
1022	002026	000		.BYTE	0
1023	002027	025		.BYTE	21.
1024	002030	003		.BYTE	3
1025	002031	000		.BYTE	0
1026	002032	000012		.WORD	10.
1027	002034	000005		.WORD	5
1028	002036	000005		.WORD	5
1029	002040	177770	177777	.WORD	177770,177777
1030	002044	177770	177777	.WORD	177770,177777
1031	002050	000		.BYTE	0
1032	002051	000		.BYTE	0
1033	002052	000		.BYTE	0
1034	002053	000		.BYTE	0
1035	002054	000		.BYTE	0
1036	002055	000		.BYTE	0
1037	002056	000000		.WORD	0
1038	002060	000		.BYTE	0
1039	002061	000		.BYTE	0
1040	002062	000000		.WORD	0

;DRIVE 0
 ;COMMAND
 ;CYLINDER ADDRESS
 ;SECTOR ADDRESS
 ;TRACK ADDRESS
 ;OFFSET VALUE
 ;BUS ADDRESS (BITS 16 AND 17)
 ;BUS ADDRESS (BITS 0 - 15)
 ;WORD COUNT (2'S COMPLEMENT)
 ;PROGRAM DRIVE STATUS INFORMATION
 ;COMMAND AND STATUS REGISTER 1
 ;COMMAND AND STATUS REGISTER 2
 ;WORD COUNT REGISTER
 ;BUS ADDRESS REGISTER
 ;DESIRED TRACK AND SECTOR REGISTER
 ;DESIRED CYLINDER REGISTER
 ;ATTENTION SUMMARY/OFFSET REGISTER
 ;ERROR REGISTER
 ;STATUS REGISTER
 ;MESSAGE LINE A STATUS BYTE 00
 ;MESSAGE LINE B STATUS BYTE 00
 ;MESSAGE LINE A STATUS BYTE 01
 ;MESSAGE LINE B STATUS BYTE 01
 ;MESSAGE LINE A STATUS BYTE 10
 ;MESSAGE LINE B STATUS BYTE 10
 ;MESSAGE LINE A STATUS BYTE 11
 ;MESSAGE LINE B STATUS BYTE 11
 ;FCC POSITION INFORMATION
 ;ECC PATTERN INFORMATION
 ;QUEUE LINK
 ;MINIMUM CYLINDER
 ;MAXIMUM CYLINDER
 ;MINIMUM TRACK
 ;MAXIMUM TRACK
 ;MINIMUM SECTOR
 ;MAXIMUM SECTOR
 ;READ/WRITE RATIO
 ;AUTOMATIC WRITE CHECK
 ;CORRECTABLE READ ERROR THRESHOLD
 ;UNCORRECTABLE READ ERROR THRESHOLD
 ;SEEK ERROR THRESHOLD
 ;MAXIMUM NUMBER OF COMMANDS TO DRIVE
 ;MAXIMUM NUMBER OF WORDS TRANSFERRED TO DRIVE
 ;SAMPLE COMPARE FLAG
 ;FCC COMPARE FLAG
 ;INHIBIT ERROR CORRECTION
 ;DATA PATTERN USED
 ;CURRENT GENERATED DATA PATTERN
 ;CURRENT RANDOMLY GENERATED COMMAND
 ;CURRENT RANDOMLY GENERATED CYLINDER
 ;CURRENT RANDOMLY GENERATED SECTOR
 ;CURRENT RANDOMLY GENERATED TRACK
 ;CURRENT RANDOMLY GENERATED WORD COUNT

1041	002064	000000			.WORD	0	;CURRENT RANDOMLY GENERATED BUS ADDRESS
1042	002066	000			.BYTE	0	
1043	002067	000			.BYTE	0	;LAST COMMAND
1044	002070	000000			.WORD	0	;LAST CYLINDER
1045	002072	000			.BYTE	0	;LAST SECTOR
1046	002073	000			.BYTE	0	;LAST TRACK
1047	002074	000000			.WORD	0	;LAST WORD COUNT
1048	002076	000			.BYTE	0	;LAST DATA PATTERN USED
1049	002077	000			.BYTE	0	;BUFFER ALLOCATED
1050	002100	000			.BYTE	0	;PETA COUNT
1051	002101	000			.BYTE	0	;PEREAD STATUS
1052	002102	000000			.WORD	0	;DRIVE STATUS FLAGS
1053	002104	000000	000000		.WORD	0,0	;NUMBER OF ORDERS
1054	002110	000000	000000	000000	.WORD	0,0,0	;NUMBER OF WORDS WRITTEN
1055	002116	000000	000000	000000	.WORD	0,0,0	;NUMBER OF WORDS READ
1056	002124	000000			.WORD	0	;NUMBER OF SOFT ERRORS (REREAD CORRECTABLE)
1057	002126	000000			.WORD	0	;NUMBER OF SOFT ERRORS (OFFSET CORRECTABLE)
1058	002130	000000			.WORD	0	;NUMBER OF SOFT ERRORS (ECC CORRECTABLE)
1059	002132	000000			.WORD	0	;NUMBER OF HARD DATA ERRORS
1060	002134	000000			.WORD	0	;NUMBER OF SEEK INCOMPLETES
1061	002136	000000			.WORD	0	;NUMBER OF MISPOSITIONING ERRORS
1062	002140	000000			.WORD	0	;NUMBER OF ALL OTHER ERRORS
1063	002142	000			.BYTE	0	;ASSIGNMENT COMMAND SEQUENCE
1064	002143	000			.BYTE	0	;SEEK TO PREVIOUS COMMAND FLAG
1065	002144	000000			.WORD	0	;FRPCH STATUS INFORMATION
1066	002146	000000			.WORD	0	;HEADER ADDRESS OF FIRST MISCOMPARE ERROR
1067	002150	000			.BYTE	0	
1068	002151	000			.BYTE	0	;OFFSET ADDRESS FROM HEADER ADDRESS
1069	002152	000000			.WORD	0	;COMPARISON LENGTH
1070	002154	000000			.WORD	0	;BUFFER COMPARISON LENGTH
1071	002156	007777			.WORD	007777	;DRIVE SERIAL NUMBER
1072	002160	000000	000000		.WORD	0,0	;CARTRIDGE SERIAL NUMBER
1073	002164	000000	000000		.WORD	0,0	;FIRST EXCLUSION AREA
1074	002170	000000	000000		.WORD	0,0	;SECOND EXCLUSION AREA
1075	002174	000000	000000		.WORD	0,0	;THIRD EXCLUSION AREA
1076	002200	000000	000000		.WORD	0,0	;FOURTH EXCLUSION AREA
1077	002204	000000	000000		.WORD	0,0	;FIFTH EXCLUSION AREA

1078				
1079			.SETTL	PARAMETER BLOCK FOR DRIVE 1
1080				
1081	002210	001	PARM1:	.BYTE 1 ;DRIVE 1
1082	002211	000		.BYTE 0 ;COMMAND
1083	002212	000000		.WORD 0 ;CYLINDER ADDRESS
1084	002214	000		.BYTE 0 ;SECTOR ADDRESS
1085	002215	000		.BYTE 0 ;TRACK ADDRESS
1086	002216	000		.BYTE 0 ;OFFSET VALUE
1087	002217	000		.BYTE 0 ;BUS ADDRESS (BITS 16 AND 17)
1088	002220	000000		.WORD 0 ;BUS ADDRESS (BITS 0 - 15)
1089	002222	000000		.WORD 0 ;WORD COUNT (2'S COMPLEMENT)
1090	002224	000000		.WORD 0 ;PROGRAM DRIVE STATUS INFORMATION
1091	002226	000000		.WORD 0 ;COMMAND AND STATUS REGISTER 1
1092	002230	000000		.WORD 0 ;COMMAND AND STATUS REGISTER 2
1093	002232	000000		.WORD 0 ;WORD COUNT REGISTER
1094	002234	000000		.WORD 0 ;BUS ADDRESS REGISTER
1095	002236	000000		.WORD 0 ;DESIRED TRACK AND SECTOR REGISTER
1096	002240	000000		.WORD 0 ;DESIRED CYLINDER REGISTER
1097	002242	000000		.WORD 0 ;ATTENTION SUMMARY/OFFSET REGISTER
1098	002244	000000		.WORD 0 ;ERROR REGISTER
1099	002246	000000		.WORD 0 ;STATUS REGISTER
1100	002250	000000		.WORD 0 ;MESSAGE LINE A STATUS BYTE 00
1101	002252	000000		.WORD 0 ;MESSAGE LINE B STATUS BYTE 00
1102	002254	000000		.WORD 0 ;MESSAGE LINE A STATUS BYTE 01
1103	002256	000000		.WORD 0 ;MESSAGE LINE B STATUS BYTE 01
1104	002260	000000		.WORD 0 ;MESSAGE LINE A STATUS BYTE 10
1105	002262	000000		.WORD 0 ;MESSAGE LINE B STATUS BYTE 10
1106	002264	000000		.WORD 0 ;MESSAGE LINE A STATUS BYTE 11
1107	002266	000000		.WORD 0 ;MESSAGE LINE B STATUS BYTE 11
1108	002270	000000		.WORD 0 ;ECC POSITION INFORMATION
1109	002272	000000		.WORD 0 ;ECC PATTERN INFORMATION
1110	002274	000000		.WORD 0 ;QUEUE LINK
1111	002276	000000		.WORD 0 ;MINIMUM CYLINDER
1112	002300	000637		.WORD 410. ;MAXIMUM CYLINDER
1113	002302	000		.BYTE 0 ;MINIMUM TRACK
1114	002303	002		.BYTE 2 ;MAXIMUM TRACK
1115	002304	000		.BYTE 0 ;MINIMUM SECTOR
1116	002305	025		.BYTE 21. ;MAXIMUM SECTOR
1117	002306	003		.BYTE 3 ;READ/WRITE LATENCY
1118	002307	000		.BYTE 0 ;AUTOMATIC WRITE CHECK
1119	002310	000012		.WORD 10. ;CORRECTABLE READ ERROR THRESHOLD
1120	002312	000005		.WORD 5 ;UNCORRECTABLE READ ERROR THRESHOLD
1121	002314	000005		.WORD 5 ;SEEK ERROR THRESHOLD
1122	002316	177770	177777	.WORD 177770,177777 ;MAXIMUM NUMBER OF COMMANDS TO DRIVE
1123	002322	177770	177777	.WORD 177770,177777 ;MAXIMUM NUMBER OF WORDS TRANSFERRED TO DRIVE
1124	002326	000		.BYTE 0 ;SAMPLE COMPARE FLAG
1125	002327	000		.BYTE 0 ;ECC COMPARE FLAG
1126	002330	000		.BYTE 0 ;INHIBIT ERROR CORRECTION
1127	002331	000		.BYTE 0 ;DATA PATTERN USED
1128	002332	000		.BYTE 0 ;CURRENT GENERATED DATA PATTERN
1129	002333	000		.BYTE 0 ;CURRENT RANDOMLY GENERATED COMMAND
1130	002334	000000		.WORD 0 ;CURRENT RANDOMLY GENERATED CYLINDER
1131	002336	000		.BYTE 0 ;CURRENT RANDOMLY GENERATED SECTOR
1132	002337	000		.BYTE 0 ;CURRENT RANDOMLY GENERATED TRACK
1133	002340	000000		.WORD 0 ;CURRENT RANDOMLY GENERATED WORD COUNT

114

1134	002342	000000			.WORD	0	;CURRENT RANDOMLY GENERATED BUS ADDRESS
1135	002344	000			.BYTE	0	
1136	002345	000			.BYTE	0	;LAST COMMAND
1137	002346	000000			.WORD	0	;LAST CYLINDER
1138	002350	000			.BYTE	0	;LAST SECTOR
1139	002351	000			.BYTE	0	;LAST TRACK
1140	002352	000000			.WORD	0	;LAST WORD COUNT
1141	002354	000			.BYTE	0	;LAST DATA PATTERN USED
1142	002355	000			.BYTE	0	;BUFFER ALLOCATED
1143	002356	000			.BYTE	0	;RETRY COUNT
1144	002357	000			.BYTE	0	;REREAD STATUS
1145	002360	000000			.WORD	0	;DRIVE STATUS FLAGS
1146	002362	000000	000000		.WORD	0,0	;NUMBER OF ORDERS
1147	002366	000000	000000	000000	.WORD	0,0,0	;NUMBER OF WORDS WRITTEN
1148	002374	000000	000000	000000	.WORD	0,0,0	;NUMBER OF WORDS READ
1149	002402	000000			.WORD	0	;NUMBER OF SOFT ERRORS (REREAD CORRECTABLE)
1150	002404	000000			.WORD	0	;NUMBER OF SOFT ERRORS (OFFSET CORRECTABLE)
1151	002406	000000			.WORD	0	;NUMBER OF SOFT ERRORS (ECC CORRECTABLE)
1152	002410	000000			.WORD	0	;NUMBER OF HARD DATA ERRORS
1153	002412	000000			.WORD	0	;NUMBER OF SEEK INCOMPLETES
1154	002414	000000			.WORD	0	;NUMBER OF MISPOSITIONING ERRORS
1155	002416	000000			.WORD	0	;NUMBER OF ALL OTHER ERRORS
1156	002420	000			.BYTE	0	;ASSIGNMENT COMMAND SEQUENCE
1157	002421	000			.BYTE	0	;SEEK TO PREVIOUS COMMAND FLAG
1158	002422	000000			.WORD	0	;ERROR STATUS INFORMATION
1159	002424	000000			.WORD	0	;HEADER ADDRESS OF FIRST MISCOMPARE SPACE
1160	002426	000			.BYTE	0	
1161	002427	000			.BYTE	0	;OFFSET ADDRESS FROM HEADER ADDRESS
1162	002430	000000			.WORD	0	;COMPARISON LENGTH
1163	002432	000000			.WORD	0	;BUFFER COMPARISON LENGTH
1164	002434	007777			.WORD	007777	;DRIVE SERIAL NUMBER
1165	002436	000000	000000		.WORD	0,0	;CARTRIDGE SERIAL NUMBER
1166	002442	000000	000000		.WORD	0,0	;FIRST EXCLUSION AREA
1167	002446	000000	000000		.WORD	0,0	;SECOND EXCLUSION AREA
1168	002452	000000	000000		.WORD	0,0	;THIRD EXCLUSION AREA
1169	002456	000000	000000		.WORD	0,0	;FOURTH EXCLUSION AREA
1170	002462	000000	000000		.WORD	0,0	;FIFTH EXCLUSION AREA

1171				
1172			.SBTTL	PARAMETER BLOCK FOR DRIVE 2
1173				
1174	002466	002	PARM2:	.BYTE 2 ;DRIVE 2
1175	002467	000		.BYTE 0 ;COMMAND
1176	002470	000000		.WORD 0 ;CYLINDER ADDRESS
1177	002472	000		.BYTE 0 ;SECTOR ADDRESS
1178	002473	000		.BYTE 0 ;TRACK ADDRESS
1179	002474	000		.BYTE 0 ;OFFSET VALUE
1180	002475	000		.BYTE 0 ;BUS ADDRESS (BITS 16 AND 17)
1181	002476	000000		.WORD 0 ;BUS ADDRESS (BITS 0 - 15)
1182	002500	000000		.WORD 0 ;WORD COUNT (2'S COMPLEMENT)
1183	002502	000000		.WORD 0 ;PROGRAM DRIVE STATUS INFORMATION
1184	002504	000000		.WORD 0 ;COMMAND AND STATUS REGISTER 1
1185	002506	000000		.WORD 0 ;COMMAND AND STATUS REGISTER 2
1186	002510	000000		.WORD 0 ;WORD COUNT REGISTER
1187	002512	000000		.WORD 0 ;BUS ADDRESS REGISTER
1188	002514	000000		.WORD 0 ;DESIRED TRACK AND SECTOR REGISTER
1189	002516	000000		.WORD 0 ;DESIRED CYLINDER REGISTER
1190	002520	000000		.WORD 0 ;ATTENTION SUMMARY/OFFSET REGISTER
1191	002522	000000		.WORD 0 ;ERROR REGISTER
1192	002524	000000		.WORD 0 ;STATUS REGISTER
1193	002526	000000		.WORD 0 ;MESSAGE LINE A STATUS BYTE 00
1194	002530	000000		.WORD 0 ;MESSAGE LINE B STATUS BYTE 00
1195	002532	000000		.WORD 0 ;MESSAGE LINE A STATUS BYTE 01
1196	002534	000000		.WORD 0 ;MESSAGE LINE B STATUS BYTE 01
1197	002536	000000		.WORD 0 ;MESSAGE LINE A STATUS BYTE 10
1198	002540	000000		.WORD 0 ;MESSAGE LINE B STATUS BYTE 10
1199	002542	000000		.WORD 0 ;MESSAGE LINE A STATUS BYTE 11
1200	002544	000000		.WORD 0 ;MESSAGE LINE B STATUS BYTE 11
1201	002546	000000		.WORD 0 ;ECC POSITION INFORMATION
1202	002550	000000		.WORD 0 ;ECC PATTERN INFORMATION
1203	002552	000000		.WORD 0 ;QUEUE LINK
1204	002554	000000		.WORD 0 ;MINIMUM CYLINDER
1205	002556	000632		.WORD 410. ;MAXIMUM CYLINDER
1206	002560	000		.BYTE 0 ;MINIMUM TRACK
1207	002561	002		.BYTE 2 ;MAXIMUM TRACK
1208	002562	000		.BYTE 0 ;MINIMUM SECTOR
1209	002563	025		.BYTE 21. ;MAXIMUM SECTOR
1210	002564	003		.BYTE 3 ;READ/WRITE FACT
1211	002565	000		.BYTE 0 ;AUTOMATIC WRITE CHECK
1212	002566	000012		.WORD 10. ;CORRECTABLE READ ERROR THRESHOLD
1213	002570	000005		.WORD 5 ;UNCORRECTABLE READ ERROR THRESHOLD
1214	002572	000005		.WORD 5 ;SEEK ERROR THRESHOLD
1215	002574	177770	177777	.WORD 177770,177777 ;MAXIMUM NUMBER OF COMMANDS TO CAUSE
1216	002600	177770	177777	.WORD 177770,177777 ;MAXIMUM NUMBER OF WORDS TRANSFERRED TO DRIVE
1217	002604	000		.BYTE 0 ;SAMPLE COMPARE FLAG
1218	002605	000		.BYTE 0 ;ECC COMPARE FLAG
1219	002606	000		.BYTE 0 ;INHIBIT ERROR CORRECTION
1220	002607	000		.BYTE 0 ;DATA PATTERN USED
1221	002610	000		.BYTE 0 ;CURRENT GENERATED DATA PATTERN
1222	002611	000		.BYTE 0 ;CURRENT RANDOMLY GENERATED COMMAND
1223	002612	000000		.WORD 0 ;CURRENT RANDOMLY GENERATED CYLINDER
1224	002614	000		.BYTE 0 ;CURRENT RANDOMLY GENERATED SECTOR
1225	002615	000		.BYTE 0 ;CURRENT RANDOMLY GENERATED TRACK
1226	002616	000000		.WORD 0 ;CURRENT RANDOMLY GENERATED WORD COUNT

1227	002620	000000		.WORD	0	;CURRENT RANDOMLY GENERATED BUS ADDRESS
1228	002622	000		.BYTE	0	
1229	002623	000		.BYTE	0	;LAST COMMAND
1230	002624	000000		.WORD	0	;LAST CYLINDER
1231	002626	000		.BYTE	0	;LAST SECTOR
1232	002627	000		.BYTE	0	;LAST TRACK
1233	002630	000000		.WORD	0	;LAST WORD COUNT
1234	002632	000		.BYTE	0	;LAST DATA PATTERN USED
1235	002633	000		.BYTE	0	;BUFFER ALLOCATED
1236	002634	000		.BYTE	0	;RETRY COUNT
1237	002635	000		.BYTE	0	;REREAD STATUS
1238	002636	000000		.WORD	0	;DRIVE STATUS FLAGS
1239	002640	000000	000000	.WORD	0,0	;NUMBER OF ORDERS
1240	002644	000000	000000 000000	.WORD	0,0,0	;NUMBER OF WORDS WRITTEN
1241	002652	000000	000000 000000	.WORD	0,0,0	;NUMBER OF WORDS READ
1242	002660	000000		.WORD	0	;NUMBER OF SOFT ERRORS (REREAD CORRECTABLE)
1243	002662	000000		.WORD	0	;NUMBER OF SOFT ERRORS (OFFSET CORRECTABLE)
1244	002664	000000		.WORD	0	;NUMBER OF SOFT ERRORS (ECC CORRECTABLE)
1245	002666	000000		.WORD	0	;NUMBER OF HARD DATA ERRORS
1246	002670	000000		.WORD	0	;NUMBER OF SEEK INCOMPLETES
1247	002672	000000		.WORD	0	;NUMBER OF MISPOSITIONING ERRORS
1248	002674	000000		.WORD	0	;NUMBER OF ALL OTHER ERRORS
1249	002676	000		.BYTE	0	;ASSIGNMENT COMMAND SEQUENCE
1250	002677	000		.BYTE	0	;SEEK TO PREVIOUS COMMAND FLAG
1251	002700	000000		.WORD	0	;ERROR STATUS INFORMATION
1252	002702	000000		.WORD	0	;HEADER ADDRESS OF FIRST MISCOMPARE ERROR
1253	002704	000		.BYTE	0	
1254	002705	000		.BYTE	0	;OFFSET ADDRESS FROM HEADER ADDRESS
1255	002706	000000		.WORD	0	;COMPARISON LENGTH
1256	002710	000000		.WORD	0	;BUFFER COMPARISON LENGTH
1257	002712	007777		.WORD	007777	;DRIVE SERIAL NUMBER
1258	002714	000000	000000	.WORD	0,0	;CARTAGE SERIAL NUMBER
1259	002720	000000	000000	.WORD	0,0	;FIRST EXCLUSION AREA
1260	002724	000000	000000	.WORD	0,0	;SECOND EXCLUSION AREA
1261	002730	000000	000000	.WORD	0,0	;THIRD EXCLUSION AREA
1262	002734	000000	000000	.WORD	0,0	;FOURTH EXCLUSION AREA
1263	002740	000000	000000	.WORD	0,0	;FIFTH EXCLUSION AREA

1264				
1265				
1266				
1267	002744	003	PARM3:	.BYTE 3
1268	002745	000		.BYTE 0
1269	002746	000000		.WORD 0
1270	002750	000		.BYTE 0
1271	002751	000		.BYTE 0
1272	002752	000		.BYTE 0
1273	002753	000		.BYTE 0
1274	002754	000000		.WORD 0
1275	002756	000000		.WORD 0
1276	002760	000000		.WORD 0
1277	002762	000000		.WORD 0
1278	002764	000000		.WORD 0
1279	002766	000000		.WORD 0
1280	002770	000000		.WORD 0
1281	002777	000000		.WORD 0
1282	002774	000000		.WORD 0
1283	002776	000000		.WORD 0
1284	003000	000000		.WORD 0
1285	003002	000000		.WORD 0
1286	003004	000000		.WORD 0
1287	003006	000000		.WORD 0
1288	003010	000000		.WORD 0
1289	003012	000000		.WORD 0
1290	003014	000000		.WORD 0
1291	003016	000000		.WORD 0
1292	003020	000000		.WORD 0
1293	003022	000000		.WORD 0
1294	003024	000000		.WORD 0
1295	003026	000000		.WORD 0
1296	003030	000000		.WORD 0
1297	003032	000000		.WORD 0
1298	003034	000632		.WORD 410.
1299	003036	000		.BYTE 0
1300	003037	002		.BYTE 2
1301	003040	000		.BYTE 0
1302	003041	025		.BYTE 21.
1303	003042	003		.BYTE 3
1304	003043	000		.BYTE 0
1305	003044	000012		.WORD 10.
1306	003046	000005		.WORD 5
1307	003050	000005		.WORD 5
1308	003052	177770	177777	.WORD 177770,177777
1309	003056	177770	177777	.WORD 177770,177777
1310	003067	000		.BYTE 0
1311	003063	000		.BYTE 0
1312	003064	000		.BYTE 0
1313	003065	000		.BYTE 0
1314	003066	000		.BYTE 0
1315	003067	000		.BYTE 0
1316	003070	000000		.WORD 0
1317	003072	000		.BYTE 0
1318	003073	000		.BYTE 0
1319	003074	000000		.WORD 0

1320	003076	000000			.WORD	0	;CURRENT RANDOMLY GENERATED BUS ADDRESS
1321	003100	000			.BYTE	0	
1322	003101	000			.BYTE	0	;LAST COMMAND
1323	003102	000000			.WORD	0	;LAST CYLINDER
1324	003104	000			.BYTE	0	;LAST SECTOR
1325	003105	000			.BYTE	0	;LAST TRACK
1326	003106	000000			.WORD	0	;LAST WORD COUNT
1327	003110	000			.BYTE	0	;LAST DATA PATTERN USED
1328	003111	000			.BYTE	0	;BUFFER ALLOCATED
1329	003112	000			.BYTE	0	;RETRY COUNT
1330	003113	000			.BYTE	0	;REREAD STATUS
1331	003114	000000			.WORD	0	;DRIVE STATUS FLAGS
1332	003116	000000	000000		.WORD	0,0	;NUMBER OF ORDERS
1333	003122	000000	000000	000000	.WORD	0,0,0	;NUMBER OF WORDS WRITTEN
1334	003130	000000	000000	000000	.WORD	0,0,0	;NUMBER OF WORDS READ
1335	003136	000000			.WORD	0	;NUMBER OF SOFT ERRORS (REREAD CORRECTABLE)
1336	003140	000000			.WORD	0	;NUMBER OF SOFT ERRORS (OFFSET CORRECTABLE)
1337	003142	000000			.WORD	0	;NUMBER OF SOFT ERRORS (ECC CORRECTABLE)
1338	003144	000000			.WORD	0	;NUMBER OF HARD DATA ERRORS
1339	003146	000000			.WORD	0	;NUMBER OF SEEK INCOMPLETES
1340	003150	000000			.WORD	0	;NUMBER OF MISPOSITIONING ERRORS
1341	003152	000000			.WORD	0	;NUMBER OF ALL OTHER ERRORS
1342	003154	000			.BYTE	0	;ASSIGNMENT COMMAND SEQUENCE
1343	003155	000			.BYTE	0	;SEEK TO PREVIOUS COMMAND FLAG
1344	003156	000000			.WORD	0	;ERROR STATUS INFORMATION
1345	003160	000000			.WORD	0	;HEADER ADDRESS OF FIRST MISCOMPARE ERROR
1346	003162	000			.BYTE	0	
1347	003163	000			.BYTE	0	;OFFSET ADDRESS FROM HEADER ADDRESS
1348	003164	000000			.WORD	0	;COMPARISON LENGTH
1349	003166	000000			.WORD	0	;BUFFER COMPARISON LENGTH
1350	003170	007777			.WORD	007777	;DRIVE SERIAL NUMBER
1351	003172	000000	000000		.WORD	0,0	;CARTAGE SERIAL NUMBER
1352	003176	000000	000000		.WORD	0,0	;FIRST EXCLUSION AREA
1353	003202	000000	000000		.WORD	0,0	;SECOND EXCLUSION AREA
1354	003206	000000	000000		.WORD	0,0	;THIRD EXCLUSION AREA
1355	003212	000000	000000		.WORD	0,0	;FOURTH EXCLUSION AREA
1356	003216	000000	000000		.WORD	0,0	;FIFTH EXCLUSION AREA

1357				
1358				
1359				.SBTTL PARAMETER BLOCK FOR DRIVE 4
1360	003227	004	PARM4:	.BYTE 4 ;DRIVE 4
1361	003227	000		.BYTE 0 ;COMMAND
1362	003224	000000		.WORD 0 ;CYLINDER ADDRESS
1363	003226	000		.BYTE 0 ;SECTOR ADDRESS
1364	003227	000		.BYTE 0 ;TRACK ADDRESS
1365	003230	000		.BYTE 0 ;OFFSET VALUE
1366	003231	000		.BYTE 0 ;BUS ADDRESS (BITS 16 AND 17)
1367	003232	000000		.WORD 0 ;BUS ADDRESS (BITS 0 - 15)
1368	003234	000000		.WORD 0 ;WORD COUNT (2'S COMPLEMENT)
1369	003236	000000		.WORD 0 ;PROGRAM DRIVE STATUS INFORMATION
1370	003240	000000		.WORD 0 ;COMMAND AND STATUS REGISTER 1
1371	003242	000000		.WORD 0 ;COMMAND AND STATUS REGISTER 2
1372	003244	000000		.WORD 0 ;WORD COUNT REGISTER
1373	003246	000000		.WORD 0 ;BUS ADDRESS REGISTER
1374	003250	000000		.WORD 0 ;DESIRED TRACK AND SECTOR REGISTER
1375	003252	000000		.WORD 0 ;DESIRED CYLINDER REGISTER
1376	003254	000000		.WORD 0 ;ATTENTION SUMMARY/OFFSET REGISTER
1377	003256	000000		.WORD 0 ;CRPOR REGISTER
1378	003260	000000		.WORD 0 ;STATUS REGISTER
1379	003262	000000		.WORD 0 ;MESSAGE LINE A STATUS BYTE 00
1380	003264	000000		.WORD 0 ;MESSAGE LINE B STATUS BYTE 00
1381	003266	000000		.WORD 0 ;MESSAGE LINE A STATUS BYTE 01
1382	003270	000000		.WORD 0 ;MESSAGE LINE B STATUS BYTE 01
1383	003272	000000		.WORD 0 ;MESSAGE LINE A STATUS BYTE 10
1384	003274	000000		.WORD 0 ;MESSAGE LINE B STATUS BYTE 10
1385	003276	000000		.WORD 0 ;MESSAGE LINE A STATUS BYTE 11
1386	003300	000000		.WORD 0 ;MESSAGE LINE B STATUS BYTE 11
1387	003302	000000		.WORD 0 ;FCC POSITION INFORMATION
1388	003304	000000		.WORD 0 ;FCC PATTERN INFORMATION
1389	003306	000000		.WORD 0 ;COUPLE LINK
1390	003310	000000		.WORD 0 ;MINIMUM CYLINDER
1391	003312	000632		.WORD 410. ;MAXIMUM CYLINDER
1392	003314	000		.BYTE 0 ;MINIMUM TRACK
1393	003315	002		.BYTE 2 ;MAXIMUM TRACK
1394	003316	000		.BYTE 0 ;MINIMUM SECTOR
1395	003317	025		.BYTE 21. ;MAXIMUM SECTOR
1396	003320	003		.BYTE 3 ;READ/WRITE RATIO
1397	003321	000		.BYTE 0 ;AUTOMATIC WHITE CHECK
1398	003322	000012		.WORD 10. ;CORRECTABLE READ ERROR THRESHOLD
1399	003324	000005		.WORD 5 ;UNCORRECTABLE READ ERROR THRESHOLD
1400	003326	000005		.WORD 5 ;SEEK ERROR THRESHOLD
1401	003330	177770	177777	.WORD 177770,177777 ;MAXIMUM NUMBER OF COMMANDS TO DRIVE
1402	003334	177770	177777	.WORD 177770,177777 ;MAXIMUM NUMBER OF WORDS TRANSFERRED TO DRIVE
1403	003340	000		.BYTE 0 ;SAFE COMPARE FLAG
1404	003341	000		.BYTE 0 ;ECC COMPARE FLAG
1405	003342	000		.BYTE 0 ;INHIBIT ERROR CORRECTION
1406	003343	000		.BYTE 0 ;DATA PATTERN USED
1407	003344	000		.BYTE 0 ;CURRENT GENERATED DATA PATTERN
1408	003345	000		.BYTE 0 ;CURRENT RANDOMLY GENERATED COMMAND
1409	003346	000000		.WORD 0 ;CURRENT RANDOMLY GENERATED CYLINDER
1410	003350	000		.BYTE 0 ;CURRENT RANDOMLY GENERATED SECTOR
1411	003351	000		.BYTE 0 ;CURRENT RANDOMLY GENERATED TRACK
1412	003352	000000		.WORD 0 ;CURRENT RANDOMLY GENERATED WORD COUNT

AS

1413	003354	000000			.WORD	0	;CURRENT RANDOMLY GENERATED BUS ADDRESS
1414	003356	000			.BYTE	0	
1415	003357	000			.BYTE	0	;LAST COMMAND
1416	003360	000000			.WORD	0	;LAST CYLINDER
1417	003362	000			.BYTE	0	;LAST SECTOR
1418	003363	000			.BYTE	0	;LAST TRACK
1419	003364	000000			.WORD	0	;LAST WORD COUNT
1420	003366	000			.BYTE	0	;LAST DATA PATTERN USED
1421	003367	000			.BYTE	0	;BUFFER ALLOCATED
1422	003370	000			.BYTE	0	;RETRY COUNT
1423	003371	000			.BYTE	0	;REREAD STATUS
1424	003372	000000			.WORD	0	;DRIVE STATUS FLAGS
1425	003374	000000	000000		.WORD	0,0	;NUMBER OF ORDERS
1426	003400	000000	000000	000000	.WORD	0,0,0	;NUMBER OF WORDS WRITTEN
1427	003406	000000	000000	000000	.WORD	0,0,0	;NUMBER OF WORDS READ
1428	003414	000000			.WORD	0	;NUMBER OF SOFT ERRORS (REREAD CORRECTABLE)
1429	003416	000000			.WORD	0	;NUMBER OF SOFT ERRORS (OFFSET CORRECTABLE)
1430	003420	000000			.WORD	0	;NUMBER OF SOFT ERRORS (ECC CORRECTABLE)
1431	003422	000000			.WORD	0	;NUMBER OF HARD DATA ERRORS
1432	003424	000000			.WORD	0	;NUMBER OF SEEK INCOMPLETES
1433	003426	000000			.WORD	0	;NUMBER OF MISPOSITIONING ERRORS
1434	003430	000000			.WORD	0	;NUMBER OF ALL OTHER ERRORS
1435	003432	000			.BYTE	0	;ASSIGNMENT COMMAND SEQUENCE
1436	003433	000			.BYTE	0	;SEEK TO PREVIOUS COMMAND FLAG
1437	003434	000000			.WORD	0	;ERROR STATUS INFORMATION
1438	003436	000000			.WORD	0	;HEADER ADDRESS OF FIRST MISCOMPARE ERROR
1439	003440	000			.BYTE	0	
1440	003441	000			.BYTE	0	;OFFSET ADDRESS FROM HEADER ADDRESS
1441	003442	000000			.WORD	0	;COMPARISON LENGTH
1442	003444	000000			.WORD	0	;BUFFER COMPARISON LENGTH
1443	003446	007777			.WORD	007777	;DRIVE SERIAL NUMBER
1444	003450	000000	000000		.WORD	0,0	;CARTRIDGE SERIAL NUMBER
1445	003454	000000	000000		.WORD	0,0	;FIRST EXCLUSION AREA
1446	003460	000000	000000		.WORD	0,0	;SECOND EXCLUSION AREA
1447	003464	000000	000000		.WORD	0,0	;THIRD EXCLUSION AREA
1448	003470	000000	000000		.WORD	0,0	;FOURTH EXCLUSION AREA
1449	003474	000000	000000		.WORD	0,0	;FIFTH EXCLUSION AREA

1450					
1451					
1452					
1453	003500	005			
1454	003501	000			
1455	003502	000000			
1456	003504	000			
1457	003505	000			
1458	003506	000			
1459	003507	000			
1460	003510	000000			
1461	003512	000000			
1462	003514	000000			
1463	003516	000000			
1464	003520	000000			
1465	003522	000000			
1466	003524	000000			
1467	003526	000000			
1468	003530	000000			
1469	003532	000000			
1470	003534	000000			
1471	003536	000000			
1472	003540	000000			
1473	003542	000000			
1474	003544	000000			
1475	003546	000000			
1476	003550	000000			
1477	003552	000000			
1478	003554	000000			
1479	003556	000000			
1480	003560	000000			
1481	003562	000000			
1482	003564	000000			
1483	003566	000000			
1484	003570	000632			
1485	003572	000			
1486	003573	002			
1487	003574	000			
1488	003575	025			
1489	003576	003			
1490	003577	000			
1491	003600	000012			
1492	003602	000005			
1493	003604	000005			
1494	003606	177770	177777		
1495	003612	177770	177777		
1496	003616	000			
1497	003617	000			
1498	003620	000			
1499	003621	000			
1500	003622	000			
1501	003623	000			
1502	003624	000000			
1503	003626	000			
1504	003627	000			
1505	003630	000000			

.SBTTL PARAMETER BLOCK FOR DRIVE 5			
PARM5:	.BYTE	5	;DRIVE 5
	.BYTE	0	;COMMAND
	.WORD	0	;CYLINDER ADDRESS
	.BYTE	0	;SECTOR ADDRESS
	.BYTE	0	;TRACK ADDRESS
	.BYTE	0	;OFFSET VALUE
	.BYTE	0	;BUS ADDRESS (BITS 16 AND 17)
	.WORD	0	;BUS ADDRESS (BITS 0 - 15)
	.WORD	0	;WORD COUNT (2'S COMPLEMENT)
	.WORD	0	;PROGRAM DRIVE STATUS INFORMATION
	.WORD	0	;COMMAND AND STATUS REGISTER 1
	.WORD	0	;COMMAND AND STATUS REGISTER 2
	.WORD	0	;WORD COUNT REGISTER
	.WORD	0	;BUS ADDRESS REGISTER
	.WORD	0	;DESIRED TRACK AND SECTOR REGISTER
	.WORD	0	;DESIRED CYLINDER REGISTER
	.WORD	0	;ATTENTION SUMMARY/OFFSET REGISTER
	.WORD	0	;ERROR REGISTER
	.WORD	0	;STATUS REGISTER
	.WORD	0	;MESSAGE LINE A STATUS BYTE 00
	.WORD	0	;MESSAGE LINE B STATUS BYTE 00
	.WORD	0	;MESSAGE LINE A STATUS BYTE 01
	.WORD	0	;MESSAGE LINE B STATUS BYTE 01
	.WORD	0	;MESSAGE LINE A STATUS BYTE 10
	.WORD	0	;MESSAGE LINE B STATUS BYTE 10
	.WORD	0	;MESSAGE LINE A STATUS BYTE 11
	.WORD	0	;MESSAGE LINE B STATUS BYTE 11
	.WORD	0	;ECC POSITION INFORMATION
	.WORD	0	;ECC PATTERN INFORMATION
	.WORD	0	;QUEUE LINK
	.WORD	0	;MINIMUM CYLINDERS
	.WORD	410.	;MAXIMUM CYLINDERS
	.BYTE	0	;MINIMUM TRACK
	.BYTE	2	;MAXIMUM TRACK
	.BYTE	0	;MINIMUM SECTOR
	.BYTE	21.	;MAXIMUM SECTOR
	.BYTE	3	;READ/WRITE RATIO
	.BYTE	0	;AUTOMATIC WRITE CHECK
	.WORD	10.	;CORRECTABLE READ ERROR THRESHOLD
	.WORD	5	;UNCORRECTABLE READ ERROR THRESHOLD
	.WORD	5	;SEEK ERROR THRESHOLD
	.WORD	177770,177777	;MAXIMUM NUMBER OF COMMANDS TO DRIVE
	.WORD	177770,177777	;MAXIMUM NUMBER OF WORDS TRANSFERRED TO DRIVE
	.BYTE	0	;SAMPLE COMPARE FLAG
	.BYTE	0	;ECC COMPARE FLAG
	.BYTE	0	;INHIBIT PRFCT COLLECTION
	.BYTE	0	;DATA PATTERN USED
	.BYTE	0	;CURRENT GENERATED DATA PATTERN
	.BYTE	0	;CURRENT RANDOMLY GENERATED COMMAND
	.WORD	0	;CURRENT RANDOMLY GENERATED CYLINDER
	.BYTE	0	;CURRENT RANDOMLY GENERATED SECTOR
	.BYTE	0	;CURRENT RANDOMLY GENERATED TRACK
	.WORD	0	;CURRENT RANDOMLY GENERATED WORD COUNT

1506	003632	000000			.WORD	0	;CURRENT RANDOMLY GENERATED BUS ADDRESS
1507	003634	000			.BYTE	0	
1508	003635	000			.BYTE	0	;LAST COMMAND
1509	003636	000000			.WORD	0	;LAST CYLINDER
1510	003640	000			.BYTE	0	;LAST SECTOR
1511	003641	000			.BYTE	0	;LAST TRACK
1512	003642	000000			.WORD	0	;LAST WORD COUNT
1513	003644	000			.BYTE	0	;LAST DATA PATTERN USED
1514	003645	000			.BYTE	0	;BUFFER ALLOCATED
1515	003646	000			.BYTE	0	;RETRY COUNT
1516	003647	000			.BYTE	0	;REREAD STATUS
1517	003650	000000			.WORD	0	;DRIVE STATUS FLAGS
1518	003652	000000	000000		.WORD	0,0	;NUMBER OF ORDERS
1519	003656	000000	000000	000000	.WORD	0,0,0	;NUMBER OF WORDS WRITTEN
1520	003664	000000	000000	000000	.WORD	0,0,0	;NUMBER OF WORDS READ
1521	003672	000000			.WORD	0	;NUMBER OF SOFT ERRORS (BEFORE CORRECTABLE)
1522	003674	000000			.WORD	0	;NUMBER OF SOFT ERRORS (OFFSET CORRECTABLE)
1523	003676	000000			.WORD	0	;NUMBER OF SOFT ERRORS (ECC CORRECTABLE)
1524	003700	000000			.WORD	0	;NUMBER OF HARD DATA ERRORS
1525	003702	000000			.WORD	0	;NUMBER OF SEEK INCOMPLETES
1526	003704	000000			.WORD	0	;NUMBER OF MISPOSITIONING ERRORS
1527	003706	000000			.WORD	0	;NUMBER OF ALL OTHER ERRORS
1528	003710	000			.BYTE	0	;ASSIGNMENT COMMAND SEQUENCE
1529	003711	000			.BYTE	0	;SEEK TO PREVIOUS COMMAND FLAG
1530	003712	000000			.WORD	0	;ERROR STATUS INFORMATION
1531	003714	000000			.WORD	0	;HEADER ADDRESS OF FIRST MISCOMPARE ERROR
1532	003716	000			.BYTE	0	
1533	003717	000			.BYTE	0	;OFFSET ADDRESS FROM HEADER ADDRESS
1534	003720	000000			.WORD	0	;COMPARISON LENGTH
1535	003722	000000			.WORD	0	;BUFFER COMPARISON LENGTH
1536	003724	007777			.WORD	007777	;DRIVE SERIAL NUMBER
1537	003726	000000	000000		.WORD	0,0	;CARTAGE SERIAL NUMBER
1538	003732	000000	000000		.WORD	0,0	;FIRST EXCLUSION AREA
1539	003736	000000	000000		.WORD	0,0	;SECOND EXCLUSION AREA
1540	003742	000000	000000		.WORD	0,0	;THIRD EXCLUSION AREA
1541	003746	000000	000000		.WORD	0,0	;FOURTH EXCLUSION AREA
1542	003752	000000	000000		.WORD	0,0	;FIFTH EXCLUSION AREA

1543				
1544			.SBTTL	PARAMETER BLOCK FOR DRIVE 6
1545				
1546	003756	006	PARM6:	.BYTE 6 ;DRIVE 6
1547	003757	000		.BYTE 0 ;COMMAND
1548	003760	000000		.WORD 0 ;CYLINDER ADDRESS
1549	003762	000		.BYTE 0 ;SECTION ADDRESS
1550	003763	000		.BYTE 0 ;TRACK ADDRESS
1551	003764	000		.BYTE 0 ;OFFSET VALUE
1552	003765	000		.BYTE 0 ;BUS ADDRESS (BITS 16 AND 17)
1553	003766	000000		.WORD 0 ;BUS ADDRESS (BITS 0 - 15)
1554	003770	000000		.WORD 0 ;WORD COUNT (2'S COMPLEMENT)
1555	003772	000000		.WORD 0 ;PROGRAM DRIVE STATUS INFORMATION
1556	003774	000000		.WORD 0 ;COMMAND AND STATUS REGISTER 1
1557	003776	000000		.WORD 0 ;COMMAND AND STATUS REGISTER 2
1558	004000	000000		.WORD 0 ;WORD COUNT REGISTER
1559	004002	000000		.WORD 0 ;BUS ADDRESS REGISTER
1560	004004	000000		.WORD 0 ;DESIRED TRACK AND SECTION REGISTER
1561	004006	000000		.WORD 0 ;DESIRED CYLINDER REGISTER
1562	004010	000000		.WORD 0 ;ATTENTION SUMMARY/CEPSET REGISTER
1563	004012	000000		.WORD 0 ;ERROR REGISTER
1564	004014	000000		.WORD 0 ;STATUS REGISTER
1565	004016	000000		.WORD 0 ;MESSAGE LINE A STATUS BYTE 00
1566	004020	000000		.WORD 0 ;MESSAGE LINE B STATUS BYTE 00
1567	004022	000000		.WORD 0 ;MESSAGE LINE A STATUS BYTE 01
1568	004024	000000		.WORD 0 ;MESSAGE LINE B STATUS BYTE 01
1569	004026	000000		.WORD 0 ;MESSAGE LINE A STATUS BYTE 10
1570	004030	000000		.WORD 0 ;MESSAGE LINE B STATUS BYTE 10
1571	004032	000000		.WORD 0 ;MESSAGE LINE A STATUS BYTE 11
1572	004034	000000		.WORD 0 ;MESSAGE LINE B STATUS BYTE 11
1573	004036	000000		.WORD 0 ;ECC POSITION INFORMATION
1574	004040	000000		.WORD 0 ;ECC PATTERN INFORMATION
1575	004042	000000		.WORD 0 ;JOUFUF LINK
1576	004044	000000		.WORD 0 ;MINIMUM CYLINDER
1577	004046	000632		.WORD 410. ;MAXIMUM CYLINDER
1578	004050	000		.BYTE 0 ;MINIMUM TRACK
1579	004051	007		.BYTE 2 ;MAXIMUM TRACK
1580	004052	000		.BYTE 0 ;MINIMUM SECTOR
1581	004053	025		.BYTE 21. ;MAXIMUM SECTOR
1582	004054	003		.BYTE 3 ;READ/WRITE MATIC
1583	004055	000		.BYTE 0 ;AUTOMATIC WRITE CHECK
1584	004056	000012		.WORD 10. ;CORRECTABLE READ ERROR THRESHOLD
1585	004060	000005		.WORD 5 ;UNCORRECTABLE READ ERROR THRESHOLD
1586	004062	000005		.WORD 5 ;SEK ERROR THRESHOLD
1587	004064	177770	177777	.WORD 177770,177777 ;MAXIMUM NUMBER OF COMMANDS TO DRIVE
1588	004070	177770	177777	.WORD 177770,177777 ;MAXIMUM NUMBER OF WORDS TRANSFERRED TO DRIVE
1589	004074	000		.BYTE 0 ;SAMPLE COMPARE FLAG
1590	004075	000		.BYTE 0 ;ECC COMPARE FLAG
1591	004076	000		.BYTE 0 ;INHIBIT FRRCD CORRECTION
1592	004077	000		.BYTE 0 ;DATA PATTERN USED
1593	004100	000		.BYTE 0 ;CURRENT GENERATED DATA PATTERN
1594	004101	000		.BYTE 0 ;CURRENT RANDOMLY GENERATED COMMAND
1595	004102	000000		.WORD 0 ;CURRENT RANDOMLY GENERATED CYLINDER
1596	004104	000		.BYTE 0 ;CURRENT RANDOMLY GENERATED SECTOR
1597	004105	000		.BYTE 0 ;CURRENT RANDOMLY GENERATED TRACK
1598	004106	000000		.WORD 0 ;CURRENT RANDOMLY GENERATED WORD COUNT

1599	004110	000000			.WORD	0	;CURRENT PANCONLY GENERATED BUS ADDRESS
1600	004112	000			.BYTE	0	
1601	004113	000			.BYTE	0	;LAST COMMAND
1602	004114	000000			.WORD	0	;LAST CYLINDER
1603	004116	000			.BYTE	0	;LAST SECTOR
1604	004117	000			.BYTE	0	;LAST TRACK
1605	004120	000000			.WORD	0	;LAST WORD COUNT
1606	004122	000			.BYTE	0	;LAST DATA PATTERN USED
1607	004123	000			.BYTE	0	;POPPED ALLOCATED
1608	004124	000			.BYTE	0	;RETRY COUNT
1609	004125	000			.BYTE	0	;REREAD STATUS
1610	004126	000000			.WORD	0	;DRIVE STATUS FLAGS
1611	004130	000000	000000		.WORD	0,0	;NUMBER OF CIPHERS
1612	004134	000000	000000	000000	.WORD	0,0,0	;NUMBER OF WORDS WRITTEN
1613	004142	000000	000000	000000	.WORD	0,0,0	;NUMBER OF WORDS READ
1614	004150	000000			.WORD	0	;NUMBER OF SOFT ERRORS (REREAD CORRECTABLE)
1615	004152	000000			.WORD	0	;NUMBER OF SOFT ERRORS (OFFSET CORRECTABLE)
1616	004154	000000			.WORD	0	;NUMBER OF SOFT ERRORS (ECC CORRECTABLE)
1617	004156	000000			.WORD	0	;NUMBER OF HARD DATA ERRORS
1618	004160	000000			.WORD	0	;NUMBER OF SEEK INCOMPLETES
1619	004162	000000			.WORD	0	;NUMBER OF MISPOSITIONING ERRORS
1620	004164	000000			.WORD	0	;NUMBER OF ALL OTHER ERRORS
1621	004166	000			.BYTE	0	;ASSIGNMENT COMMAND SEQUENCE
1622	004167	000			.BYTE	0	;SEK TO PREVIOUS COMMAND FLAG
1623	004170	000000			.WORD	0	;ERROR STATUS INFORMATION
1624	004172	000000			.WORD	0	;HEADER ADDRESS OF FIRST MISMATCH ERROR
1625	004174	000			.BYTE	0	
1626	004175	000			.BYTE	0	;OFFSET ADDRESS FROM HEADER ADDRESS
1627	004176	000000			.WORD	0	;COMPARISON LENGTH
1628	004200	000000			.WORD	0	;BUFFER COMPARISON LENGTH
1629	004202	007777			.WORD	007777	;DRIVE SERIAL NUMBER
1630	004204	000000	000000		.WORD	0,0	;CARIAGE SERIAL NUMBER
1631	004210	000000	000000		.WORD	0,0	;FIRST EXCLUSION AREA
1632	004214	000000	000000		.WORD	0,0	;SECOND EXCLUSION AREA
1633	004220	000000	000000		.WORD	0,0	;THIRD EXCLUSION AREA
1634	004224	000000	000000		.WORD	0,0	;FOURTH EXCLUSION AREA
1635	004230	000000	000000		.WORD	0,0	;FIFTH EXCLUSION AREA

1636					
1637				.SBTTL	PARAMETER BLOCK FOR DRIVE 7
1638					
1639	004234	007		PARM7:	.RYTE 7
1640	004235	000			.RYTE 0
1641	004236	000000			.WORD 0
1642	004240	000			.BYTE 0
1643	004241	000			.RYTE 0
1644	004242	000			.BYTE 0
1645	004243	000			.RYTE 0
1646	004244	000000			.WORD 0
1647	004246	000000			.WORD 0
1648	004250	000000			.WORD 0
1649	004252	000000			.WORD 0
1650	004254	000000			.WORD 0
1651	004256	000000			.WORD 0
1652	004260	000000			.WORD 0
1653	004262	000000			.WORD 0
1654	004264	000000			.WORD 0
1655	004266	000000			.WORD 0
1656	004270	000000			.WORD 0
1657	004272	000000			.WORD 0
1658	004274	000000			.WORD 0
1659	004276	000000			.WORD 0
1660	004300	000000			.WORD 0
1661	004302	000000			.WORD 0
1662	004304	000000			.WORD 0
1663	004306	000000			.WORD 0
1664	004310	000000			.WORD 0
1665	004312	000000			.WORD 0
1666	004314	000000			.WORD 0
1667	004316	000000			.WORD 0
1668	004320	000000			.WORD 0
1669	004322	000000			.WORD 0
1670	004324	000632			.WORD 410.
1671	004326	000			.BYTE 0
1672	004327	002			.BYTE 2
1673	004330	000			.BYTE 0
1674	004331	025			.RYTE 21.
1675	004332	003			.BYTE 3
1676	004333	000			.BYTE 0
1677	004334	000012			.WORD 10.
1678	004336	000005			.WORD 5
1679	004340	000005			.WORD 5
1680	004342	177770	177777		.WORD 177770,177777
1681	004346	177770	177777		.WORD 177770,177777
1682	004352	000			.BYTE 0
1683	004353	000			.BYTE 0
1684	004354	000			.BYTE 0
1685	004355	000			.RYTE 0
1686	004356	000			.BYTE 0
1687	004357	000			.BYTE 0
1688	004360	000000			.WORD 0
1689	004362	000			.RYTE 0
1690	004363	000			.BYTE 0
1691	004364	000000			.WORD 0

1692	004366	000000			.WORD	0	);CURRENT RANDOMLY GENERATED BUS ADDRESS
1693	004370	000			.BYTE	0	
1694	004371	000			.BYTE	0	);LAST COMMAND
1695	004372	000000			.WORD	0	);LAST CYLINDER
1696	004374	000			.BYTE	0	);LAST SECTOR
1697	004375	000			.BYTE	0	);LAST TRACK
1698	004376	000000			.WORD	0	);LAST WORD COUNT
1699	004400	000			.BYTE	0	);LAST DATA PATTERN USED
1700	004401	000			.BYTE	0	);BUFFER ALLOCATED
1701	004402	000			.BYTE	0	);RETRY COUNT
1702	004403	000			.BYTE	0	);REREAD STATUS
1703	004404	000000			.WORD	0	);DRIVE STATUS FLAGS
1704	004406	000000	000000		.WORD	0,0	);NUMBER OF ORDERS
1705	004412	000000	000000	000000	.WORD	0,0,0	);NUMBER OF WORDS WRITTEN
1706	004420	000000	000000	000000	.WORD	0,0,0	);NUMBER OF WORDS READ
1707	004426	000000			.WORD	0	);NUMBER OF SPT ERRORS (REREAD CORRECTABLE)
1708	004430	000000			.WORD	0	);NUMBER OF SPT ERRORS (OFFSET CORRECTABLE)
1709	004432	000000			.WORD	0	);NUMBER OF SPT ERRORS (ECC CORRECTABLE)
1710	004434	000000			.WORD	0	);NUMBER OF HARD DATA ERRORS
1711	004436	000000			.WORD	0	);NUMBER OF SEEK INCOMPLETES
1712	004440	000000			.WORD	0	);NUMBER OF MISPOSITIONING ERRORS
1713	004442	000000			.WORD	0	);NUMBER OF ALL OTHER ERRORS
1714	004444	000			.BYTE	0	);ASSIGNMENT COMMAND SEQUENCE
1715	004445	000			.BYTE	0	);SEEK TO PREVIOUS COMMAND FLAG
1716	004446	000000			.WORD	0	);ERROR STATUS INFORMATION
1717	004450	000000			.WORD	0	);HEADER ADDRESS OF FIRST MISCOMPARE ERROR
1718	004452	000			.BYTE	0	
1719	004453	000			.BYTE	0	);OFFSET ADDRESS FROM HEADER ADDRESS
1720	004454	000000			.WORD	0	);COMPARISON LENGTH
1721	004456	000000			.WORD	0	);BUFFER COMPARISON LENGTH
1722	004460	007777			.WORD	007777	);DRIVE SERIAL NUMBER
1723	004462	000000	000000		.WORD	0,0	);CARTRIDGE SERIAL NUMBER
1724	004466	000000	000000		.WORD	0,0	);FIRST EXCLUSION AREA
1725	004472	000000	000000		.WORD	0,0	);SECOND EXCLUSION AREA
1726	004476	000000	000000		.WORD	0,0	);THIRD EXCLUSION AREA
1727	004507	000000	000000		.WORD	0,0	);FOURTH EXCLUSION AREA
1728	004506	000000	000000		.WORD	0,0	);FIFTH EXCLUSION AREA

1729			.SBTTL	---	RETRY PARAMETER BLOCK	---
1730						
1731			.SBTTL	PARAMETER BLOCK 10 FOR DRIVE		
1732						
1733	004512	000	PARM10:	.BYTE	0	;DRIVE NUMBER
1734	004513	000		.BYTE	0	;COMMAND
1735	004514	000000		.WORD	0	;CYLINDER ADDRESS
1736	004516	000		.BYTE	0	;SECTOR ADDRESS
1737	004517	000		.BYTE	0	;TRACK ADDRESS
1738	004520	000		.BYTE	0	;OFFSET VALUE
1739	004521	000		.BYTE	0	;BUS ADDRESS (BITS 16 AND 17)
1740	004522	000000		.WORD	0	;BUS ADDRESS (BITS 0 - 15)
1741	004524	000000		.WORD	0	;WORD COUNT (2'S COMPLEMENT)
1742	004526	000000		.WORD	0	;PROGRAM DRIVE STATUS INFORMATION
1743	004530	000000		.WORD	0	;COMMAND AND STATUS REGISTER 1
1744	004532	000000		.WORD	0	;COMMAND AND STATUS REGISTER 2
1745	004534	000000		.WORD	0	;WORD COUNT REGISTER
1746	004536	000000		.WORD	0	;BUS ADDRESS REGISTER
1747	004540	000000		.WORD	0	;DESIRED TRACK AND SECTOR REGISTER
1748	004542	000000		.WORD	0	;DESIRED CYLINDER REGISTER
1749	004544	000000		.WORD	0	;ATTENTION SUMMARY/OFFSET REGISTER
1750	004546	000000		.WORD	0	;ERROR REGISTER
1751	004550	000000		.WORD	0	;STATUS REGISTER
1752	004552	000000		.WORD	0	;MESSAGE LINE A STATUS BYTE 00
1753	004554	000000		.WORD	0	;MESSAGE LINE B STATUS BYTE 00
1754	004556	000000		.WORD	0	;MESSAGE LINE A STATUS BYTE 01
1755	004560	000000		.WORD	0	;MESSAGE LINE B STATUS BYTE 01
1756	004562	000000		.WORD	0	;MESSAGE LINE A STATUS BYTE 10
1757	004564	000000		.WORD	0	;MESSAGE LINE B STATUS BYTE 10
1758	004566	000000		.WORD	0	;MESSAGE LINE A STATUS BYTE 11
1759	004570	000000		.WORD	0	;MESSAGE LINE B STATUS BYTE 11
1760	004572	000000		.WORD	0	;FCC POSITION INFORMATION
1761	004574	000000		.WORD	0	;FCC PATTERN INFORMATION

.SBTTL DATA PATTERNS AND DATA PATTERN TABLE

PATTAB: PATO

PAT1
PAT2
PAT3
PAT4
PAT5
PAT6
PAT7
PAT10
PAT11
PAT12
PAT13
PAT14
PAT15
PAT16
PAT17

PATO:

.WORD	165555
.WORD	133333
.WOPD	165555
.WOPD	133333
.WORD	165555
.WORD	133333
.WORD	165555
.WORD	133333
.WOR^	165555
.WORD	133333
.WORD	165555
.WORD	133333
.WORD	165555
.WORD	133333
.WCRO	165555
.WOPD	133333
.WORD	165555
.WORD	133333
.WORD	165555
.WORD	133333
.WORD	165555
.WORD	133333
.WORD	165555
.WORD	133333
.WORD	165555
.WORD	133333
.WORD	165555
.WORD	133333
.WOPD	165555
.WOPD	133333
.WCPO	165555
.WORD	133333

PAT1:

```
.WORD    026455
.WCPD    151322
.WORD    026455
.WORD    026455
```

1818	004746	151322	.WORD	151322
1819	004750	151322	.WORD	151322
1820	004752	026455	.WORD	026455
1821	004754	026455	.WORD	026455
1822	004756	026455	.WORD	026455
1823	004760	151322	.WORD	151322
1824	004762	151322	.WORD	151322
1825	004764	151322	.WORD	151322
1826	004766	026455	.WORD	026455
1827	004770	026455	.WORD	026455
1828	004772	026455	.WORD	026455
1829	004774	026455	.WORD	026455
1830	004776	151322	.WORD	151322
1831	005000	151322	.WORD	151322
1832	005002	151322	.WORD	151322
1833	005004	151322	.WORD	151322
1834	005006	026455	.WORD	026455
1835	005010	026455	.WORD	026455
1836	005012	026455	.WORD	026455
1837	005014	151322	.WORD	151322
1838	005016	151322	.WORD	151322
1839	005020	151322	.WORD	151322
1840	005022	026455	.WORD	026455
1841	005024	026455	.WORD	026455
1842	005026	151322	.WORD	151322
1843	005030	151322	.WORD	151322
1844	005032	026455	.WORD	026455
1845	005034	151322	.WORD	151322
1846				
1847	005036	007417	PAT2: .WORD	007417
1848	005040	170360	.WORD	170360
1849	005042	007417	.WORD	007417
1850	005044	007417	.WORD	007417
1851	005046	170360	.WORD	170360
1852	005050	170360	.WORD	170360
1853	005052	007417	.WORD	007417
1854	005054	007417	.WORD	007417
1855	005056	007417	.WORD	007417
1856	005060	170360	.WORD	170360
1857	005062	170360	.WORD	170360
1858	005064	170360	.WORD	170360
1859	005066	007417	.WORD	007417
1860	005070	007417	.WORD	007417
1861	005072	007417	.WORD	007417
1862	005074	007417	.WORD	007417
1863	005076	170360	.WORD	170360
1864	005100	170360	.WORD	170360
1865	005102	170360	.WORD	170360
1866	005104	170360	.WORD	170360
1867	005106	007417	.WORD	007417
1868	005110	007417	.WORD	007417
1869	005112	007417	.WORD	007417
1870	005114	170360	.WORD	170360
1871	005116	170360	.WORD	170360
1872	005120	170360	.WORD	170360
1873	005122	007417	.WORD	007417

1874	005124	007417	.WORD	007417
1875	005126	170360	.WCPD	170360
1876	005130	170360	.WOPD	170360
1877	005132	007417	.WORD	007417
1878	005134	170360	.WORD	170360
1879				
1880	005136	052525	PAT3: .WOPD	052525
1881	005140	125252	.WORD	125252
1882	005142	052525	.WORD	052525
1883	005144	052525	.WORD	052525
1884	005146	125252	.WORD	125252
1885	005150	125252	.WORD	125252
1886	005152	052525	.WORD	052525
1887	005154	052525	.WCPD	052525
1888	005156	052525	.WOPD	052525
1889	005160	125252	.WORD	125252
1890	005162	125252	.WOPD	125252
1891	005164	125252	.WORD	125252
1892	005166	052525	.WORD	052525
1893	005170	052525	.WORD	052525
1894	005172	052525	.WOPD	052525
1895	005174	052525	.WCPD	052525
1896	005176	125252	.WOPD	125252
1897	005200	125252	.WORD	125252
1898	005202	125252	.WOPD	125252
1899	005204	125252	.WORD	125252
1900	005206	052525	.WORD	052525
1901	005210	052525	.WORD	052525
1902	005212	052525	.WORD	052525
1903	005214	125252	.WCPD	125252
1904	005216	125252	.WORD	125252
1905	005220	125252	.WORD	125252
1906	005222	052525	.WOPD	052525
1907	005224	052525	.WCPD	052525
1908	005226	125252	.WOPD	125252
1909	005230	125252	.WORD	125252
1910	005232	052525	.WORD	052525
1911	005234	125252	.WCPD	125252
1912				
1913	005236	000000	PAT4: .WORD	000000
1914	005240	010421	.WORD	010421
1915	005242	021042	.WORD	021042
1916	005244	031463	.WORD	031463
1917	005246	042104	.WORD	042104
1918	005250	052525	.WOPD	052525
1919	005252	063146	.WCPD	063146
1920	005254	073567	.WORD	073567
1921	005256	104210	.WOPD	104210
1922	005260	114631	.WORD	114631
1923	005262	125252	.WCPD	125252
1924	005264	135673	.WORD	135673
1925	005266	146314	.WORD	146314
1926	005270	156735	.WOPD	156735
1927	005272	167356	.WCPD	167356
1928	005274	177777	.WOPD	177777
1929	005276	177777	.WORD	177777

1930	005300	167356	.WORD	167356
1931	005302	156735	.WORD	156735
1932	005304	146314	.WORD	146314
1933	005306	135673	.WORD	135673
1934	005310	125252	.WORD	125252
1935	005312	114631	.WORD	114631
1936	005314	104210	.WORD	104210
1937	005316	073567	.WORD	073567
1938	005320	063146	.WORD	063146
1939	005322	052525	.WORD	052525
1940	005324	042104	.WORD	042104
1941	005326	031463	.WORD	031463
1942	005330	021042	.WORD	021042
1943	005332	010421	.WORD	010421
1944	005334	000000	.WORD	000000
1945				
1946	005336	000000	PAT5: .WORD	000000
1947	005340	177777	.WORD	177777
1948	005342	000000	.WORD	000000
1949	005344	000000	.WORD	000000
1950	005346	177777	.WORD	177777
1951	005350	177777	.WORD	177777
1952	005352	000000	.WORD	000000
1953	005354	000000	.WORD	000000
1954	005356	000000	.WORD	000000
1955	005360	177777	.WORD	177777
1956	005362	177777	.WORD	177777
1957	005364	177777	.WORD	177777
1958	005366	000000	.WORD	000000
1959	005370	000000	.WORD	000000
1960	005372	000000	.WORD	000000
1961	005374	000000	.WORD	000000
1962	005376	177777	.WORD	177777
1963	005400	177777	.WORD	177777
1964	005402	177777	.WORD	177777
1965	005404	177777	.WORD	177777
1966	005406	000000	.WORD	000000
1967	005410	000000	.WORD	000000
1968	005412	000000	.WORD	000000
1969	005414	177777	.WORD	177777
1970	005416	177777	.WORD	177777
1971	005420	177777	.WORD	177777
1972	005422	000000	.WORD	000000
1973	005424	000000	.WORD	000000
1974	005426	177777	.WORD	177777
1975	005430	177777	.WORD	177777
1976	005432	000000	.WORD	000000
1977	005434	177777	.WORD	177777
1978				
1979	005436	000001	PAT6: .WORD	000001
1980	005440	000003	.WORD	000003
1981	005442	000007	.WORD	000007
1982	005444	000017	.WORD	000017
1983	005446	000037	.WORD	000037
1984	005450	000077	.WORD	000077
1985	005452	000177	.WORD	000177

M5

1986	005454	000777	.WORD	000777
1987	005456	000777	.WORD	000777
1988	005460	001777	.WORD	001777
1989	005462	003777	.WORD	003777
1990	005464	007777	.WORD	007777
1991	005466	017777	.WORD	017777
1992	005470	037777	.WORD	037777
1993	005472	077777	.WORD	077777
1994	005474	177777	.WORD	177777
1995	005476	177776	.WORD	177776
1996	005500	177774	.WORD	177774
1997	005502	177770	.WORD	177770
1998	005504	177760	.WORD	177760
1999	005506	177740	.WORD	177740
2000	005510	177700	.WORD	177700
2001	005512	177600	.WORD	177600
2002	005514	177400	.WORD	177400
2003	005516	177000	.WORD	177000
2004	005520	176000	.WORD	176000
2005	005522	174000	.WORD	174000
2006	005524	170000	.WORD	170000
2007	005526	160000	.WORD	160000
2008	005530	140000	.WORD	140000
2009	005532	100000	.WORD	100000
2010	005534	000000	.WORD	000000
2011				
2012	005536	000001	PAT7: .WORD	000001
2013	005540	000002	.WORD	000002
2014	005542	000004	.WORD	000004
2015	005544	000010	.WORD	000010
2016	005546	000020	.WORD	000020
2017	005550	000040	.WORD	000040
2018	005552	000100	.WORD	000100
2019	005554	000200	.WORD	000200
2020	005556	000400	.WORD	000400
2021	005560	001000	.WORD	001000
2022	005562	002000	.WORD	002000
2023	005564	004000	.WORD	004000
2024	005566	010000	.WORD	010000
2025	005570	020000	.WORD	020000
2026	005572	040000	.WORD	040000
2027	005574	100000	.WORD	100000
2028	005576	100000	.WORD	100000
2029	005600	040000	.WORD	040000
2030	005602	020000	.WORD	020000
2031	005604	010000	.WORD	010000
2032	005606	004000	.WORD	004000
2033	005610	002000	.WORD	002000
2034	005612	001000	.WORD	001000
2035	005614	000400	.WORD	000400
2036	005616	000200	.WORD	000200
2037	005620	000100	.WORD	000100
2038	005622	000040	.WORD	000040
2039	005624	000020	.WORD	000020
2040	005626	000010	.WORD	000010
2041	005630	000004	.WORD	000004

2042	005632	000002	.WORD	000002
2043	005634	000001	.WORD	000001
2044				
2045	005636	177776	PAT10: .WORD	177776
2046	005640	177775	.WORD	177775
2047	005642	177773	.WORD	177773
2048	005644	177767	.WORD	177767
2049	005646	177757	.WORD	177757
2050	005650	177737	.WORD	177737
2051	005652	177677	.WORD	177677
2052	005654	177577	.WORD	177577
2053	005656	177377	.WORD	177377
2054	005660	176777	.WORD	176777
2055	005662	175777	.WORD	175777
2056	005664	173777	.WORD	173777
2057	005666	167777	.WORD	167777
2058	005670	157777	.WORD	157777
2059	005672	137777	.WORD	137777
2060	005674	077777	.WORD	077777
2061	005676	077777	.WORD	077777
2062	005700	137777	.WORD	137777
2063	005702	157777	.WORD	157777
2064	005704	167777	.WORD	167777
2065	005706	173777	.WORD	173777
2066	005710	175777	.WORD	175777
2067	005712	176777	.WORD	176777
2068	005714	177377	.WORD	177377
2069	005716	177577	.WORD	177577
2070	005720	177677	.WORD	177677
2071	005722	177737	.WORD	177737
2072	005724	177757	.WORD	177757
2073	005726	177767	.WORD	177767
2074	005730	177773	.WORD	177773
2075	005732	177775	.WORD	177775
2076	005734	177776	.WORD	177776
2077				
2078	005736	172666	PAT11: .WORD	172666
2079	005740	155555	.WORD	155555
2080	005742	172666	.WORD	172666
2081	005744	155555	.WORD	155555
2082	005746	172666	.WORD	172666
2083	005750	155555	.WORD	155555
2084	005752	172666	.WORD	172666
2085	005754	155555	.WORD	155555
2086	005756	172666	.WORD	172666
2087	005760	155555	.WORD	155555
2088	005762	172666	.WORD	172666
2089	005764	155555	.WORD	155555
2090	005766	172666	.WORD	172666
2091	005770	155555	.WORD	155555
2092	005772	172666	.WORD	172666
2093	005774	155555	.WORD	155555
2094	005776	172666	.WORD	172666
2095	006000	155555	.WORD	155555
2096	006002	172666	.WORD	172666
2097	006004	155555	.WORD	155555

2098	006006	172666	.WORD	172666
2099	006010	155555	.WORD	155555
2100	006012	172666	.WORD	172666
2101	006014	155555	.WORD	155555
2102	006016	172666	.WORD	172666
2103	006020	155555	.WORD	155555
2104	006022	172666	.WORD	172666
2105	006024	155555	.WORD	155555
2106	006026	172666	.WORD	172666
2107	006030	155555	.WORD	155555
2108	006032	172666	.WORD	172666
2109	006034	155555	.WORD	155555
2110				
2111	006036	153333	PAT12: .WORD	153333
2112	006040	066667	.WORD	066667
2113	006042	153333	.WORD	153333
2114	006044	066667	.WORD	066667
2115	006046	153333	.WORD	153333
2116	006050	066667	.WORD	066667
2117	006052	153333	.WORD	153333
2118	006054	066667	.WORD	066667
2119	006056	153333	.WORD	153333
2120	006060	066667	.WORD	066667
2121	006062	153333	.WORD	153333
2122	006064	066667	.WORD	066667
2123	006066	153333	.WORD	153333
2124	006070	066667	.WORD	066667
2125	006072	153333	.WORD	153333
2126	006074	066667	.WORD	066667
2127	006076	153333	.WORD	153333
2128	006100	066667	.WORD	066667
2129	006102	153333	.WORD	153333
2130	006104	066667	.WORD	066667
2131	006106	153333	.WORD	153333
2132	006110	066667	.WORD	066667
2133	006112	153333	.WORD	153333
2134	006114	066667	.WORD	066667
2135	006116	153333	.WORD	153333
2136	006120	066667	.WORD	066667
2137	006122	153333	.WORD	153333
2138	006124	066667	.WORD	066667
2139	006126	153333	.WORD	153333
2140	006130	066667	.WORD	066667
2141	006132	153333	.WORD	153333
2142	006134	066667	.WORD	066667
2143				
2144	006136	000000	PAT13: .WORD	000000
2145	006140	177777	.WORD	177777
2146	006142	177777	.WORD	177777
2147	006144	177777	.WORD	177777
2148	006146	177777	.WORD	177777
2149	006150	177777	.WORD	177777
2150	006152	177777	.WORD	177777
2151	006154	177777	.WORD	177777
2152	006156	177777	.WORD	177777
2153	006160	177777	.WORD	177777

2154	006162	177777	.WORD	177777
2155	006164	177777	.WCPD	177777
2156	006166	177777	.WOPD	177777
2157	006170	177777	.WORD	177777
2158	006172	177777	.WORD	177777
2159	006174	177777	.WCRD	177777
2160	006176	177777	.WOPD	177777
2161	006200	177777	.WORD	177777
2162	006202	177777	.WORD	177777
2163	006204	177777	.WCRD	177777
2164	006206	177777	.WORD	177777
2165	006210	177777	.WORD	177777
2166	006212	177777	.WORD	177777
2167	006214	177777	.WORD	177777
2168	006216	177777	.WORD	177777
2169	006220	177777	.WORD	177777
2170	006222	177777	.WOPD	177777
2171	006224	177777	.WCRD	177777
2172	006226	177777	.WORD	177777
2173	006230	177777	.WORD	177777
2174	006232	177777	.WORD	177777
2175	006234	177777	.WCPD	177777
2176				
2177	006236	177777	PAT14: .WORD	177777
2178	006240	000000	.WOPD	000000
2179	006242	000000	.WORD	000000
2180	006244	000000	.WORD	000000
2181	006246	000000	.WORD	000000
2182	006250	000000	.WORD	000000
2183	006252	000000	.WCRD	000000
2184	006254	000000	.WORD	000000
2185	006256	000000	.WORD	000000
2186	006260	000000	.WOPD	000000
2187	006262	000000	.WOPD	000000
2188	006264	000000	.WORD	000000
2189	006266	000000	.WORD	000000
2190	006270	000000	.WORD	000000
2191	006272	000000	.WCRD	000000
2192	006274	000000	.WORD	000000
2193	006276	000000	.WORD	000000
2194	006300	000000	.WOPD	000000
2195	006302	000000	.WCPD	000000
2196	006304	000000	.WORD	000000
2197	006306	000000	.WORD	000000
2198	006310	000000	.WORD	000000
2199	006312	000000	.WCPD	000000
2200	006314	000000	.WORD	000000
2201	006316	000000	.WOPD	000000
2202	006320	000000	.WORD	000000
2203	006322	000000	.WCRD	000000
2204	006324	000000	.WOPD	000000
2205	006326	000000	.WORD	000000
2206	006330	000000	.WORD	000000
2207	006332	000000	.WCPD	000000
2208	006334	000000	.WORD	000000
2209				

2210	006336	172304	PAT15:	.WORD	172304
2211	006340	172304		.WORD	172304
2212	006342	172304		.WORD	172304
2213	006344	172304		.WORD	172304
2214	006346	172304		.WORD	172304
2215	006350	172304		.WORD	172304
2216	006352	172304		.WORD	172304
2217	006354	172304		.WORD	172304
2218	006356	172304		.WORD	172304
2219	006360	172304		.WORD	172304
2220	006362	172304		.WORD	172304
2221	006364	172304		.WORD	172304
2222	006366	172304		.WORD	172304
2223	006370	172304		.WORD	172304
2224	006372	172304		.WORD	172304
2225	006374	172304		.WORD	172304
2226	006376	172304		.WORD	172304
2227	006400	172304		.WORD	172304
2228	006402	172304		.WORD	172304
2229	006404	172304		.WORD	172304
2230	006406	172304		.WORD	172304
2231	006410	172304		.WORD	172304
2232	006412	172304		.WORD	172304
2233	006414	172304		.WORD	172304
2234	006416	172304		.WORD	172304
2235	006420	172304		.WORD	172304
2236	006422	172304		.WORD	172304
2237	006424	172304		.WORD	172304
2238	006426	172304		.WORD	172304
2239	006430	172304		.WORD	172304
2240	006432	172304		.WORD	172304
2241	006434	172304		.WORD	172304
2242					
2243	006436	070627	PAT16:	.WORD	070627
2244	006440	113431		.WORD	113431
2245	006442	014561		.WORD	014561
2246	006444	070627		.WORD	070627
2247	006446	113431		.WORD	113431
2248	006450	014561		.WORD	014561
2249	006452	070627		.WORD	070627
2250	006454	113431		.WORD	113431
2251	006456	014561		.WORD	014561
2252	006460	070627		.WORD	070627
2253	006462	113431		.WORD	113431
2254	006464	014561		.WORD	014561
2255	006466	070627		.WORD	070627
2256	006470	113431		.WORD	113431
2257	006472	014561		.WORD	014561
2258	006474	070627		.WORD	070627
2259	006476	113431		.WORD	113431
2260	006500	014561		.WORD	014561
2261	006502	070627		.WORD	070627
2262	006504	113431		.WORD	113431
2263	006506	014561		.WORD	014561
2264	006510	070627		.WORD	070627
2265	006512	113431		.WORD	113431

2266	006514	014561	.WORD	014561
2267	006516	070627	.WCPD	070627
2268	006520	113431	.WORD	113431
2269	006522	014561	.WORD	014561
2270	006524	070627	.WOPD	070627
2271	006526	113431	.WCRD	113431
2272	006530	014561	.WOPD	014561
2273	006532	070627	.WOPD	070627
2274	006534	113431	.WOPD	113431
2275				
2276	006536	133467	PAT17: .WORD	133467
2277	006540	133467	.WORD	133467
2278	006542	133467	.WORD	133467
2279	006544	133467	.WCPD	133467
2280	006546	133467	.WOPD	133467
2281	006550	133467	.WORD	133467
2282	006552	133467	.WORD	133467
2283	006554	133467	.WCPD	133467
2284	006556	133467	.WOPD	133467
2285	006560	133467	.WORD	133467
2286	006562	133467	.WOPD	133467
2287	006564	133467	.WCPD	133467
2288	006566	133467	.WOPD	133467
2289	006570	133467	.WORD	133467
2290	006572	133467	.WORD	133467
2291	006574	133467	.WCPD	133467
2292	006576	133467	.WOPD	133467
2293	006600	133467	.WORD	133467
2294	006602	133467	.WORD	133467
2295	006604	133467	.WCRD	133467
2296	006606	133467	.WORD	133467
2297	006610	133467	.WORD	133467
2298	006612	133467	.WOPD	133467
2299	006614	133467	.WCPD	133467
2300	006616	133467	.WOPD	133467
2301	006620	133467	.WCRD	133467
2302	006622	133467	.WOPD	133467
2303	006624	133467	.WCPD	133467
2304	006626	133467	.WOPD	133467
2305	006630	133467	.WOPD	133467
2306	006632	133467	.WOPD	133467
2307	006634	133467	.WCPD	133467

```

2308 .SBTTL PROGRAM SETUP
2309
2310 006636 112737 000001 001361 PARSRT: MOVB #1,FLAG ;SET FLAG FOR PARAMETER ENTRY
2311 006644 000406 BR PRGSRT ;START PROGRAM
2312
2313 006646 112737 177777 001361 RESTART: MOVB #1,FLAG ;SET FLAG FOR RETRY
2314 006654 000402 BR PRGSRT ;START PROGRAM
2315
2316 006656 105037 001361 START: CLRR FLAG ;SET FOR NORMAL PROGRAM START
2317 006662 012737 000340 177776 PRGSRT: MOV #PR7,PS ;LOCK OUT ALL INTERRUPTS
2318 .SBTTL INITIALIZE THE COMMON TAGS
2319 ;;CLEAR THE COMMON TAGS (SCMTAG) AREA
2320 006670 012706 001100 MOV #SCMTAG,R6 ;;FIRST LOCATION TO BE CLEARED
2321 006674 005026 CLR (R6)+ ;;CLEAR MEMORY LOCATION
2322 006676 022706 001140 CMP #SWR,P6 ;;DONE?
2323 006702 001374 BNE #-6 ;;LOOP BACK IF NO
2324 006704 012706 001100 MOV #STACK,SP ;;SETUP THE STACK POINTER
2325 ;;INITIALIZE A FEW VECTORS
2326 006710 012737 055034 000034 MOV #STRAP,#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
2327 006716 012737 000340 000036 MOV #340,#TRAPVEC+2;LEVEL 7
2328 006724 012737 053722 000024 MOV #SPWRDN,#PWRVEC ;;POWER FAILURE VECTOR
2329 006732 012737 000340 000026 MOV #340,#PWRVEC+2 ;;LEVEL 7
2330 ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
2331 ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
2332 006740 013746 000004 MOV #ERRVEC,-(SP) ;;SAVE ERROR VECTOR
2333 006744 012737 007000 000004 MOV #645,#ERRVEC ;;SET UP ERROR VECTOR
2334 006752 012737 177570 001140 MOV #DSWR,SWR ;;SETUP FOR A HARDWARE SWITCH REGISTER
2335 006760 012737 177570 001142 MOV #DDISP,DISPLAY ;;AND A HARDWARE DISPLAY REGISTER
2336 006766 022777 177777 172144 CMP #-1,#SWP ;;TRY TO REFERENCE HARDWARE SWR
2337 006774 001012 BNE 66$ ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
2338 ;;AND THE HARDWARE SWR IS NOT = -1
2339 006776 000403 BR 65$ ;;BRANCH IF NO TIMEOUT
2340 007000 012716 007006 64$: MOV #65$,(SP) ;;SET UP FOR TRAP RETURN
2341 007004 000002 RTI
2342 007006 012737 000176 001140 65$: MOV #SWREG,SWP ;;POINT TO SOFTWARE SWR
2343 007014 012737 000174 001142 MOV #DISPREG,DISPLAY
2344 007022 012637 000004 66$: MOV (SP)+,#ERRVEC ;;RESTORE ERROR VECTOR
2345
2346 007026 005037 001176 CLR $PASS ;;CLEAR PASS COUNT
2347 007032 132737 000200 001211 BITF #APTSIZE,$ENV ;;IFST USER SIZE UNDER APT
2348 007040 001403 BRQ 67$ ;;YES,USE MCM-APT SWITCH
2349 007042 012737 001212 001140 MOV #SSWRFC,SWR ;;NO,USE APT SWITCH REGISTER
2350 007050 67$:
2351 .SBTTL TYPE PROGRAM NAME
2352 ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
2353 007050 005227 177777 INC #-1 ;;FIRST TIME?
2354 007054 001035 BNE 68$ ;;BRANCH IF AC
2355 007056 104401 007064 TYPE ,69$ ;;TYPE ASCII STRING
2356 007062 000432 BR 68$ ;;GET OVER THE ASCII
2357 ;;69$: .ASCII <CRLF>"RK611/RK06 PERFORMANCE EXERCISER: MAINDEC DZR6P-B"<CRLF>
2358 68$:
2359 007150 000005 PMSRT: RESET ;RESET SYSTEM
2360 007152 012737 007224 000004 MOV #205,FRRVEC ;SET VECTOR FOR MEMORY PARITY CHECK
2361 007160 012737 000340 000006 MOV #PR7,FRRVEC+2
2362 007166 012703 172100 MOV #MEMBAS,R3 ;LOAD REGISTER TO DTEPMIN IF
2363 ; MEMORY CHECK ENABLE AVAILABLE

```

2364	007172	012704	000014		MOV	#12.,R4	;LOAD COUNT
2365	007176	012723	000001		16\$: MOV	#PAR.EM,(R3)+	;ENABLE MEMORY CHECK VECTOR
2366	007202	012737	051642	000114	MOV	#MEMERR,MENVEC	;LOAD MEMORY CHECK VECTOR
2367	007210	012737	000340	000116	MOV	#PK7,MENVEC+2	
2368	007216	005304			DEC	R4	;CHECK IF FINISHED
2369	007220	001366			BNE	16\$	;NG, SET UP NEXT MEMORY PARITY PCULE
2370	007222	000401			BR	22\$	;RESTORE TRAP VECTOR
2371							
2372	007224	022626			20\$: CMP	(SP)+,(SP)+	;ADJUST STACK
2373	007226	012737	000006	000004	22\$: MOV	#ERRVEC+2,ERRVEC	;RESTORE TRAP CATCHER
2374	007234	005037	000006		CLR	ERRVEC+2	
2375	007240	004737	053624		STARTS: JSR	PC,\$SIZE	;GET MAXIMUM MEMORY ADDRESS
2376	007244	012737	000001	001664	MOV	#1,FBLKC	;INITIALIZE FREE BLOCK COUNT
2377	007252	012703	001666		MOV	#FBLKT,R3	;STORE FREE BLOCK TABLE ADDRESS
2378	007256	012704	000022		MOV	#18.,R4	;LOAD COUNT FOR FREE BLOCK CLEAR
2379	007262	005023			25: CLR	(R3)+	;CLEAR FREE BLOCK TABLE
2380	007264	005304			DEC	R4	;DECREMENT COUNT
2381	007266	001375			BNE	2\$	;CHECK IF FINISHED
2382	007270	012737	065202	001666	MOV	#BUPADD,FBLKT	;LOAD FIRST ENTRY IN FREE BLOCK TABLE
2383	007276	105037	001640		CLRB	RTYSCN	;CLEAR RETRY SECTOR COUNT
2384	007307	005037	001567		CLR	LRRPRO	;CLEAR ADDRESS OF PARAMETER BLOCK
2385							; PROCESSING ERROR
2386	007306	005037	001460		CLR	E.CONT	;CLEAR CONTROLLER ERROR FLAGS
2387	007312	005037	001560		CLR	ERCONT	;CLEAR RECOVERABLE CONTROLLER ERROR FLAGS
2388	007316	005037	001622		CLR	DLTCNT	;CLEAR DATA LATE COUNT
2389	007322	005037	001624		CLR	CNTCNT	;CLEAR OTHER ERRORS COUNT
2390	007326	105037	001627		CLRB	STATIS	;CLEAR INTERVAL STATISTICS FLAG
2391	007332	112737	177777	001507	MOVR	#-1,0.OVER	;SET ALL DRIVES FOR IMPLIED SEKS
2392	007340	012703	001314		MOV	#AVAILQ,R3	;STORE QUEUE BASE ADDRESS
2393	007344	012704	000010		MOV	#8.,R4	;LOAD QUEUE COUNT
2394	007350	005023			35: CLR	(R3)+	;CLEAR QUEUE HEADS AND TAILS
2395	007352	005304			DEC	R4	;DECREMENT COUNT
2396	007354	001375			BNE	3\$	;CHECK IF FINISHED
2397	007356	012704	001540		MOV	#M.DRV,R4	;LOAD ADDRESS OF TIME OUT COUNTS
2398	007362	012703	000010		MOV	#8.,R3	;LOAD COUNT
2399	007366	005024			45: CLR	(R4)+	;CLEAR DRIVE TIME OUT COUNTS
2400	007370	005303			DEC	R3	;DECREMENT COUNT
2401	007372	001375			BNE	4\$	;CHECK IF FINISHED
2402	007374	004737	042574		JSR	PC,CHKCLK	;CHECK IF CLOCK IS PRESENT
2403	007400	105037	001506		CLRB	W.TIME	;CLEAR DRIVES BEING TIMED
2404	007404	013737	001466	001464	MOV	W.MILT,W.MTIM	;INITIALIZE MILT-SECOND TIMER
2405	007412	005037	001462		CLR	O.WAIT	;CLEAR PARAMETER PLOCK FOR DRIVE
2406							; WAITING COMMAND COMPLETION
2407	007416	105037	001502		CLRB	I.ISHL	;RESET RELEASE OR CONTROLLER CLEAR ISSUEC
2408	007422	105737	001361		TSTR	FLAG	;CHECK IF RESTART
2409	007426	100072			BPL	RESTO	;NG, CONTINUE
2410							
2411	007430	013737	053720	001670	MOV	\$LSTAD,FBLKT+2	;LOAD MAX. ACD IF FREE BLOCK TABLE
2412	007436	105737	001365		TSTR	OVLYLN	;CHECK IF OVERLAY LOADER
2413	007442	001012			BNE	20\$	;YES, LOAD RK06 VECTOR ADDRESS
2414	007444	105737	000041		TSTB	41	;SEE WHO LOADED THE PROBLEM
2415	007450	001004			BNE	10\$	;BRANCH IF XXDP
2416	007452	162737	000300	001670	SUB	#96.*7.,FBLKT+2	;SUBTRACT APS LOCATE SIZE
2417	007460	000403			BR	20\$	;LOAD VECTOR ADDRESS
2418							
2419	007462	162737	004004	001670	10\$: SUB	#1026.*7.,FBLKT+2	;SUBTRACT XXDP LOADER

TYPE PROGRAM NAME

```
2420 007470 013704 001446      20$:  MOV      RRVFC,R4          ;STORE VECTOR ADDRESS
2421 007474 012724 043470      MOV      #1.INTR,(R4)+      ;LOAD RK06 VECTOR ADDRESS
2422 007500 013714 001450      MOV      RRPRI,(R4)          ; AND INTERRUPT PRIORITY
2423 007504 012703 001520      MOV      #PRLKT,R3          ;LOAD PARAMETER BLOCK TABLE ADDRESS
2424 007510 012704 000011      MOV      #9.,R4           ;LOAD COUNT
2425
2426 007514 005304      21$:  DFC      R4              ;DECREMENT COUNT AND CHECK IF DONE
2427 007516 001434      BFC      30$              ;YES, SET UP TTY INTERRUPT
2428 007520 012305      MOV      (R3)+,R5          ;GET PARAMETER BLOCK ADDRESS
2429 007522 105065 000145      CLRB     P.BUFF(R5)          ;CLEAR BUFFER ALLOCATED
2430 007526 005065 000212      CLR      P.ERR(R5)          ;CLEAR ERROR FLAGS AND RETRY COUNTS
2431 007532 005065 000150      CLR      P.DSTT(R5)
2432 007536 005065 000146      CLR      P.RECT(R5)
2433
2434
2435      ;      RESET THE FOLLOWING DRIVER FLAGS
2436      ;      DRIVE POSITIONING
2437      ;      DRIVE POSITIONING FOR DATA TRANSFER
2438      ;      DRIVE SEIZED
2439      ;      DRIVE UNLOADED DUE TO ERROR
2440      ;      PARAMETER BLOCK ENQUEUED IN INITIATION QUEUE
2441 007542 042765 070006 000014      BIC      #DRVPOS|DRVPT|DRVSZD|E.UNLD|Q.INIT,P.PRST(15)
2442
2443 007550 032765 000001 000014      BIT      #DRVUSE,P.PRST(R5) ;CHECK IF DRIVE IN USE
2444 007556 001756      BFC      21$              ;NO, GET NEXT PARAMETER BLOCK
2445 007560 105765 000210      TSTB     P.ASSN(R5)          ;CHECK IF DRIVE IS BEING ASSIGNED
2446 007564 001003      BNE      22$              ;YES, RESTART ASSIGNMENT SEQUENCE
2447 007566 004737 013324      JSR      PC,TEST1          ;ISSUE A START SPINDLE IF NECESSARY
2448 007572 000750      BR       21$              ;GET NEXT PARAMETER BLOCK
2449
2450 007574 142765 177757 000210 22$:  BICB     #C<BIT4>,P.ASSN(R5) ;KEEP WRITE BACK BIT
2451 007602 004737 013250      JSR      PC,TEST          ;INITIATE DRIVE ASSIGNMENT SEQUENCE
2452 007606 000742      BR       21$              ;GET NEXT PARAMETER BLOCK
2453
2454 007610 000137 011230      30$:  JMP      REST2          ;SET TTY INTERRUPT
2455
2456 007614 012737 176543 054276 REST0: MOV      #176543,$SHNUM      ;INITIALIZE EABCCP NUMBER GENERATOR
2457 007622 012737 123456 054300      MOV      #123456,$LONUM
2458 007630 005037 001372      CLR      SECOND          ;INITIALIZE INTERNAL CLOCK
2459 007634 005037 001370      CLR      MINUTE
2460 007640 005037 001366      CLR      HOUR
2461 007644 012703 001520      MOV      #PRLKT,P3          ;LOAD PARAMETER BLOCK TABLE ADDRESS
2462 007650 012704 000010      MOV      #9.,R4           ;LOAD COUNT
2463 007654 012305      4$:  MOV      (R3)+,R5          ;GET PARAMETER BLOCK ADDRESS
2464 007656 005065 000014      CLR      P.PRST(R5)          ;CLEAR RK06 PROGRAM DEVICE STATUS BPG.
2465 007662 005065 000212      CLR      P.ERR(R5)          ;CLEAR DRIVE ERROR FLAGS
2466 007666 105065 000210      CLRB     P.ASSN(R5)          ;CLEAR DRIVE ASSIGNMENT CODE
2467 007672 010500      MOV      R5,R0            ;STORE R0 FOR TRACK EXCLUSIONS
2468 007674 062700 000232      ADD      #P.EXAR,R0          ;DETERMINE TRACK EXCLUSION BLOCK
2469 007700 012701 000012      MOV      #10.,R1          ;LOAD COUNT FOR CLEANING TRACK EXCLUSION
2470      ;      PLOCK
2471 007704 005020      5$:  CLR      (R0)+          ;CLEAR TRACK EXCLUSION ENTRY
2472 007706 005301      DEC      R1              ;DECREMENT COUNT
2473 007710 001375      BNE      5$              ;CHECK IF WHOLE BLOCK CLEARED
2474 007712 005304      DFC      R4              ;DECREMENT COUNT
2475 007714 001357      BNE      4$              ;TEST IF DONE
```

2476	007716	122737	000001	001361		CMPP	#1,FLAG	)CHECK IF PARAMETER START
2477	007724	001402				BEQ	10\$	)YES, ASK FOR PARAMETERS
2478	007726	000137	011060			JMP	51\$	)USP DEFAULT VALUES
2479								
2480	007732	104401	056250		10\$:	TYPE	,SYS000	)TYPE "CPU ID"
2481	007736	004037	020116			JSR	R0,TSTDEF	)CHECK FOR DEFAULT PARAMETERS
2482	007742	007772				11\$		)COMMA DETECTED
2483	007744	010010				13\$		)CARRIAGE RETURN DETECTED
2484	007746	011060				51\$		)CONTROL Z <"Z"> DETECTED
2485	007750	007732				10\$		)CONTROL C <"C"> DETECTED
2486	007752	010346				MOV	R3,-(SP)	)LOAD BUFFER ADDRESS ON STACK
2487	007754	004737	053316			JSR	PC,OCTBIN	)CONVERT TO BINARY
2488	007760	007772				11\$		)ERROR RETURN
2489	007762	012604				MOV	(SP)+,R4	)STORE INPUT VALUE
2490	007764	022704	000400			CMP	#400,R4	)CHECK IF 0-377
2491	007770	101010				BRI	14\$	)YES, LOAD CPU ID
2492	007772				11\$:			
2493	007772	010337	010000			MOV	R3,64\$	)LOAD BUFFER ADDRESS
2494	007776	104401				TYPE		)TYPE RECEIVED INPUT
2495	010000	000000			64\$:	.WORD	0	)BUFFER ADDRESS
2496	010002	104401	001164			TYPE	,SQUES	)TYPE QUESTION MARK
2497	010006	000751				BR	10\$	)TRY AGAIN
2498								
2499	010010	005004			13\$:	CLR	R4	)CLEAR CPU ID (R4)
2500	010012	110437	001364		14\$:	MOVB	R4,CPUID	)LOAD CPU ID
2501								
2502	010016	013737	053720	001670	15\$:	MOV	\$LSTAD,FBLKT+2	)LOAD LAST ADDRESS IN FIRST
2503								) FREE BLOCK ENTRY
2504	010024	104401	056260			TYPE	,SYS001	)TYPE "OVERLAY LOADER"
2505	010030	004037	020116			JSR	R0,TSTDEF	)CHECK FOR DEFAULT PARAMETERS
2506	010034	010104				11\$		)COMMA DETECTED
2507	010036	010052				16\$		)CARRIAGE RETURN DETECTED
2508	010040	011072				52\$		)CONTROL Z <"Z"> DETECTED
2509	010042	007732				10\$		)CONTROL C <"C"> DETECTED
2510	010044	122713	000131			CMPP	#"V,(R3)	)CHECK IF YES
2511	010050	001424				BEQ	19\$	)YES, INDICATE THAT LOADER HAS BEEN OVERLAYED
2512	010052	105037	001365		16\$:	CLRB	OVLYLD	)INDICATE THAT LOADER HAS NOT BEEN OVERLAYED
2513	010056	105737	000041			TSTR	41	)SEE WHO LOADED THE PROGRAM
2514	010062	001004				BNE	17\$	)BRANCH IF XXOP
2515	010064	162737	000300	001670		SUB	#96.*2.,FBLKT+2	)SUBTRACT ABS LOCDEF SIZE
2516	010072	000416				BR	20\$	)ASK FOR MAXIMUM TRANSFER SIZE
2517								
2518	010074	162737	006000	001670	17\$:	SUB	#1536.*7.,FBLKT+2	)SUBTRACT XXOP LOADER SIZE
2519	010102	000412				BR	20\$	)ASK FOR TRANSFER SIZE
2520								
2521	010104				18\$:			
2522	010104	010337	010112			MOV	R3,65\$	)LOAD BUFFER ADDRESS
2523	010110	104401				TYPE		)TYPE RECEIVED INPUT
2524	010112	000000			65\$:	.WORD	0	)BUFFER ADDRESS
2525	010114	104401	001164			TYPE	,SQUES	)TYPE QUESTION MARK
2526	010120	000736				BR	15\$	)TRY AGAIN
2527								
2528	010122	112737	177777	001365	19\$:	MOVB	#-1,OVLYLD	)INDICATE THAT LOADER HAS BEEN OVERLAYED
2529								
2530	010130	104401	056300		20\$:	TYPE	,SYS002	)TYPE "MAXIMUM TRANSFER"
2531	010134	013746	001670			MOV	FBLKT+2,-(SP)	)STORE LAST USABLE MEMORY LOCATION

```
2532 010140 163716 001666 SUB FBLKT,(SP) ;CALCULATE MEMORY LEFT
2533 010144 022716 027000 CMP #256.*23.*2.,(SP) ;CHECK IF GREATER THAN OF EQUAL TO 5000
2534 010150 101402 BLOS 21$ ;YES, USE 5000
2535 010152 006216 ASR (SP) ;DETERMINE NUMBER OF WORDS
2536 010154 000402 BR 27$ ;TYPE VALUE
2537
2538 010156 012716 013400 21$: MOV #256.*23.,(SP) ;LOAD 5000
2539
2540 010162 012637 001564 22$: MOV (SP)+,TMPCNT ;SAVE TRANSFER COUNT
2541 010166 013746 001564 MOV TMPCNT,-(SP) ;SAVE TRANSFER COUNT FOR TYPE GOT
2542 010172 004737 052144 JSR PC,BINOCY ;CONVERT TO OCTAL
2543 010176 104401 056332 TYPE ,SYS003
2544 010202 004037 020116 JSR RO,TSTDEF ;CHECK FOR DEFAULT PARAMETERS
2545 010206 010250 24$ ;COMMA DETECTED
2546 010210 010266 26$ ;CARRIAGE RETURN DETECTED
2547 010212 011122 S3$ ;CONTROL Z <^Z> DETECTED
2548 010214 007732 10$ ;CONTROL C <^C> DETECTED
2549 010216 010346 MOV R3,-(SP) ;STORE PUPPFA ADDRESS ON STACK
2550 010220 004737 053316 JSR PC,OCTBIN ;CONVERT TO BINARY
2551 010224 010246 23$ ;ERROR RETURN
2552 010226 012604 MOV (SP)+,R4 ;STORE INPUT
2553 010230 001407 BFG 24$ ;CHECK IF ZERO
2554 010232 020437 001564 CMP R4,TMPCNT ;CHECK IF LEGAL ENTRY
2555 010236 101004 BHI 24$ ;NO, TRY AGAIN
2556 010240 010437 001354 MOV R4,MAXBUF ;LOAD MAXIMUM TRANSFER
2557 010244 000422 BR 27$ ;GET NO. OF SECT COMPARES
2558
2559 010246 005726 23$: TST (SP)+ ;ADJUST STACK
2560 010250 24$:
2561 010250 010337 010256 MOV R3,66$ ;LOCAL BUFFER ADDRESS
2562 010254 104401 TYPE ;TYPE RECEIVED INPUT
2563 010256 000000 66$: .WORD 0 ;BUFFER ADDRESS
2564 010260 104401 001164 TYPE ,SQUES ;TYPE QUESTION MARK
2565 010264 000721 BP 20$ ;TRY AGAIN
2566
2567 010266 013737 001564 001354 26$: MOV TMPCNT,MAXBUF ;LOAD MAXIMUM TRANSFER
2568 010274 022737 013400 001354 CMP #256.*23.,MAXBUF ;CHECK IF LESS THAN OR EQUAL TO 23 SECTORS
2569 010302 103003 BHS 27$ ;YES, USE CALCULATED VALUE
2570 010304 012737 013400 001354 MOV #256.*23.,MAXBUF ;USE 23 SECTORS
2571
2572 010312 104401 056336 27$: TYPE ,SYS004 ;TYPE "NO. OF SOFTWARE COMPARISONS"
2573 010316 004037 020116 JSR RO,TSTDEF ;CHECK FOR DEFAULT PARAMETERS
2574 010322 010360 29$ ;COMMA DETECTED
2575 010324 010376 30$ ;CARRIAGE RETURN DETECTED
2576 010326 011160 S4$ ;CONTROL Z <^Z> DETECTED
2577 010330 007732 10$ ;CONTROL C <^C> DETECTED
2578 010332 010346 MOV R3,-(SP) ;STORE PARAMETER
2579 010334 004737 053452 JSR PC,DECBIN ;CONVERT TO BINARY
2580 010340 010360 29$ ;ERROR RETURN
2581 010342 023716 001354 CMP MAXBUF,(SP) ;CHECK IF LEGAL INPUT
2582 010346 103403 BLO 28$ ;TYPE VALUE AND RETRY
2583 010350 012637 001362 MOV (SP)+,SCFCMP ;LOAD NUMBER OF SOFTWARE COMPARES
2584 010354 000413 BR 31$ ;DETERMINE RK06 UNIBUS ADDRESS
2585
2586 010356 005726 28$: TST (SP)+ ;ADJUST STACK
2587 010360 29$:
```

2588	010360	010337	010366		MOV	R3,67\$	;LOAD BUFFER ADDRESS
2589	010364	104401			TYPE		;TYPE RECEIVED INPUT
2590	010366	000000		67\$:	.WORD	0	;BUFFER ADDRESS
2591	010370	104401	001164		TYPE	,SQUES	;TYPE QUESTION MARK
2592	010374	000746			BR	27\$	;TRY AGAIN
2593							
2594	010376	012737	000003	001362	30\$:	MOV	#3,SOPCMP
2595							;LOAD DEFAULT VALUE
2596	010404	105737	001374		31\$:	TSTB	CLKPLC
2597	010410	001434			BEQ	41\$	;CHECK IF CLOCK ON SYSTEM
2598	010412	104401	056517		TYPE	,SYS009	;NO, CONTINUE
2599	010416	004037	020116		JSR	R0,TSTDEF	;TYPE "STATISTIC INTERVAL="
2600	010422	010460			34\$		;CHECK FOR DEFAULT PARAMETERS
2601	010424	010476			35\$		;COMMA DETECTED
2602	010426	011166			55\$		;CARRIAGE RETURN DETECTED
2603	010430	007732			10\$		;CONTROL Z <"Z"> DETECTED
2604	010432	010346			MOV	R3,-(SP)	;CONTROL C <"C"> DETECTED
2605	010434	004737	053452		JSR	PC,DECBIN	;STORE NUMBER FOR CONVERSION
2606	010440	010460			34\$		;CONVERT TO BINARY
2607	010442	012604			MOV	(SP)+,R4	;ILLEGAL NUMBER
2608	010444	022704	000377		CMP	#255-,R4	;STORE NUMBER
2609	010450	103403			BLO	34\$	;CHECK IF GREATER THAN 255
2610	010452	110437	001360		MOVP	R4,PERINV	;NO, TRY AGAIN
2611	010456	000411			BR	41\$	;LOAD PERFORMANCE INTERVAL
2612							;CONTINUE
2613	010460			34\$:			
2614	010460	010337	010466		MOV	R3,68\$	;LOAD BUFFER ADDRESS
2615	010464	104401			TYPE		;TYPE RECEIVED INPUT
2616	010466	000000		68\$:	.WORD	0	;BUFFER ADDRESS
2617	010470	104401	001164		TYPE	,SQUES	;TYPE QUESTION MARK
2618	010474	000743			BR	31\$	;TRY AGAIN
2619							
2620	010476	105037	001360		35\$:	CLRM	PERINV
2621							;SET UP FOR NO INTERVAL STATISTICS
2622	010502	104401	056370		41\$:	TYPE	,SYS005
2623	010506	004037	020116		JSR	R0,TSTDEF	;TYPE "RK06 BUS ACD."
2624	010512	010626			44\$		;CHECK FOR DEFAULT PARAMETERS
2625	010514	010644			45\$		;COMMA DETECTED
2626	010516	011177			56\$		;CARRIAGE RETURN DETECTED
2627	010520	007732			10\$		;CONTROL Z <"Z"> DETECTED
2628	010522	010346			MOV	R3,-(SP)	;CONTROL C <"C"> DETECTED
2629	010524	004737	053316		JSR	PC,OCTBIN	;STORE PARAMETER
2630	010530	010626			44\$		;CONVERT TO BINARY
2631	010532	012604			MOV	(SP)+,R4	;ERRCP RETURN
2632	010534	022704	160000		CMP	#160000,R4	;STORE INPUT
2633	010540	101032			BHI	44\$	;CHECK IF I/C PAGE
2634	010542	012737	010602	000004	MOV	#42\$,ERRVEC	;NO, TRY AGAIN
2635	010550	013737	000340	000006	MOV	PR7,ERRVEC+2	;LOAD ERRVEC TO NON-EXISTENT MEMORY
2636	010556	005714			TST	(P4)	; PRIORITY 7
2637	010560	012737	000000	000006	MOV	#<HALT>,ERRVEC+2	;CHECK FOR NON-EXISTENT MEMORY
2638	010566	012737	000006	000004	MOV	#ERRVEC+2,ERRVEC	;REINSTATE TRAP CATCHER
2639	010574	010437	001444		MOV	R4,RKBAS	;LEGAL I/O ADDRESS, LOAD RK06 BASE
2640	010600	000424			BR	48\$	;GET VECTOR ADDRESS
2641							
2642	010602	012737	000000	000006	47\$:	MOV	#<HALT>,ERRVEC+2
2643	010610	012737	000006	000004	MOV	#ERRVEC+2,ERRVEC	;REINSTATE TRAP CATCHER

2644	010616	062706	000004			ADD	#4,SP	;ADJUST STACK
2645	010622	104401	056424			TYPE	,SYS007	;TYPE "NON-EXISTENT MEMORY"
2646	010626			44\$:				
2647	010626	010337	010634			MOV	R3,69\$	;LOAD BUFFER ADDRESS
2648	010632	104401				TYPE		;TYPE RECEIVED INPUT
2649	010634	000000		69\$:		.WORD	0	;BUFFER ADDRESS
2650	010636	104401	001164			TYPE	,SQUES	;TYPE QUESTION MARK
2651	010642	000717				BR	41\$	;TRY AGAIN
2652								
2653	010644	012737	177440	001444	45\$:	MOV	#177440,RKBAS	;LOAD DEFAULT BUS ADDRESS
2654								
2655	010652	104401	056406		48\$:	TYPE	,SYS006	;TYPE "RK06 VEC ADD"
2656	010656	004037	020116			JSR	R0,TSTDEF	;CHECK FOR DEFAULT PARAMETERS
2657	010662	010720				50\$		;COMMA DETECTED
2658	010664	010736				51\$		;CARRIAGE RETURN DETECTED
2659	010666	011200				57\$		;CONTROL Z <"Z"> DETECTED
2660	010670	007732				10\$		;CONTROL C <"C"> DETECTED
2661	010672	010346				MOV	R3,-(SP)	;STORE PARAMETER
2662	010674	004737	053316			JSR	PC,OCTBIN	;CONVERT TO BINARY
2663	010700	010720				50\$		;ERROR RETURN
2664	010702	022716	001000			CMP	#1000,(SP)	;CHECK IF VECTOR SPACE
2665	010706	101403				BLOS	49\$	;NO, TRY AGAIN
2666	010710	012637	001446			MOV	(SP)+,RKVEC	;LOAD VECTOR ADDRESS
2667	010714	000413				BR	53\$	;GET PRIORITY
2668								
2669	010716	005726			49\$:	TST	(SP)+	;ADJUST STACK
2670	010720				50\$:			
2671	010720	010337	010726			MOV	R3,70\$	;LOAD BUFFER ADDRESS
2672	010724	104401				TYPE		;TYPE RECEIVED INPUT
2673	010726	000000			70\$:	.WORD	0	;BUFFER ADDRESS
2674	010730	104401	001164			TYPE	,SQUES	;TYPE QUESTION MARK
2675	010734	000746				BR	48\$	;TRY AGAIN
2676								
2677	010736	012737	000210	001446	51\$:	MOV	#210,RKVEC	;LOAD DEFAULT RK06 VECTOR
2678	010744	104401	056543		53\$:	TYPE	,SYS010	;TYPE "RK06 PRIC="
2679	010750	004037	020116			JSR	R0,TSTDEF	;CHECK FOR DEFAULT PARAMETERS
2680	010754	011037				56\$		;COMMA DETECTED
2681	010756	011050				57\$		;CARRIAGE RETURN DETECTED
2682	010760	011206				58\$		;CONTROL Z <"Z"> DETECTED
2683	010762	007732				10\$		;CONTROL C <"C"> DETECTED
2684	010764	010346				MOV	R3,-(SP)	;STORE PARAMETER
2685	010766	004737	053316			JSR	PC,OCTBIN	;CONVERT TO BINARY
2686	010772	011032				56\$		;ERROR RETURN
2687	010774	022716	000007			CMP	#7,(SP)	;CHECK IF OCTAL
2688	011000	103413				BLO	55\$	
2689	011002	022716	000004			CMP	#4,(SP)	
2690	011006	101010				BHI	55\$	
2691	011010	006316				ASL	(SP)	;SHIFT 5 BITS LEFT
2692	011012	006316				ASL	(SP)	
2693	011014	006316				ASL	(SP)	
2694	011016	006316				ASL	(SP)	
2695	011020	006316				ASL	(SP)	
2696	011022	012637	001450			MOV	(SP)+,RKPRI	;LOAD PRIORITY
2697	011026	000472				BR	REST1	;LOAD RK06 VECTOR AND TTY VECTOR
2698								
2699	011030	005726			55\$:	TST	(SP)+	;ADJUST STACK

2700	011032				56\$:			
2701	011032	010337	011040			MOV	R3,715	;LOCAL BUFFER ADDRESS
2702	011036	104401				TYPE		;TYPE RECEIVED INPUT
2703	011040	000000			71\$:	.WORD	0	;BUFFER ADDRESS
2704	011047	104401	001164			TYPE	,SQUES	;TYPE QUESTION MARK
2705	011046	000736				BP	53\$	;TRY AGAIN
2706								
2707	011050	012737	000240	001450	57\$:	MOV	#PR5,PKPRI	;LOAD PRIORITY
2708	011056	000456				BP	REST1	;LOAD RK06 VECTOR AND ITY VECTOR
2709								
2710	011060	105037	001364		S1\$:	CLRP	CPUID	;LOAD DEFAULT CPU ID
2711	011064	013737	053720	001670		MOV	\$LSTAD,FBLKT+2	;LOAD LAST ADDRESS
2712	011072	105037	001365		S2\$:	CLRP	OVLYLD	;INDICATE THAT THE LOADER IS NOT OVERLAPED
2713	011076	105737	000041			TSTP	41	;SET WHO LOADED PROGRAM
2714	011102	001004				BNE	1\$	;BRANCH IF XDP
2715	011104	162737	000700	001670		SUB	#96.*2.,FBLKT+2	;SUBTRACT AHS LOADER SIZE
2716	011112	000403				BR	S3\$	;GO DETERMINE MAX BUFFER SIZE
2717								
2718	011114	162737	006000	001670	1\$:	SUB	#1536.*2.,FBLKT+2	;SUBTRACT XDP LOADER SIZE
2719	011127	013746	001670		S3\$:	MOV	FBLKT+2,-(SP)	;STORE MAXIMUM ADDRESS
2720	011126	163716	001666			SUB	FBLKT,(SP)	;DETERMINE MEMORY AVAILABLE
2721	011132	022716	027000			CMP	#256.*23.*2.,(SP)	;CHECK IF ENOUGH ROOM
2722	011136	101005				BHI	1\$	;NO,LOAD MEMORY AVAILABLE
2723	011140	005726				TST	(SP)+	;ADJUST STACK
2724	011142	012737	013400	001354		MOV	#256.*23.,MAXBUF	;LOAD MAXIMUM BUFFER SIZE
2725	011150	000403				BP	S4\$	;LOAD NUMBER OF SOFTWARE COMPARES
2726								
2727	011152	006716			1\$:	ASR	(SP)	;CHANGE TO WORDS
2728	011154	012637	001354			MOV	(SP)+,MAXBUF	;LOAD MAXIMUM WORDS TRANSFERRED
2729	011160	012737	000003	001362	S4\$:	MOV	#3,SOFCEP	;LOAD NO. OF SOFTWARE COMPARES
2730	011166	105037	001760		S5\$:	CLRP	PERINV	;SET UP FOR 60 INTERVAL STATISTICS
2731	011172	012737	177440	001444	S6\$:	MOV	#177440,RFBAS	;LOCAL DEFAULT BUS ADDRESS
2732	011200	012737	000210	001446	S7\$:	MOV	#710,PKVEC	;LOAD DEFAULT VECTOR ADDRESS
2733	011206	012737	000240	001450	S8\$:	MOV	#PR5,PKPRI	;LOAD RK06 PRIORITY
2734								
2735	011214	013704	001446		REST1:	MOV	RKVEC,R4	;STORE VECTOR ADDRESS
2736	011220	012724	043470			MOV	#1.INTR,(R4)+	;LOAD RK06 VECTOR ADDRESS
2737	011224	013714	001450			MOV	RKPRI,(R4)	; AND INTERUPT PRIORITY
2738	011230	004737	042666		RFST2:	JSR	PC,CLKINT	;INITIALIZE CLOCK VECTOR ADDRESS
2739	011234	104401	056451			TYPE	,SYS008	;TYPE "READY TO BEGIN TESTING"
2740	011240	005227	177777			INC	#-1	;TYPE HELP MESSAGE ON FIRST PASS
2741	011244	001002				BNE	5\$	
2742	011246	104401	064776			TYPE	,HELP	
2743	011252	004737	052712		5\$:	JSR	PC,ST*INT	;SET UP ITY INTERRUPT
2744	011256	013737	001450	000036		MOV	RKPRI,TRAPVEC+2	;ALLOW L AND P CLOCK INTERRUPTS
2745	011264	005037	177776			CLR	PS	;ALLOW RK06 INTERRUPTS
2746								

```

2747      .SBTTL  MAIN IDLE LOOP
2748
2749 011270 022706 001100      MAIN:  CMP      #STACK,SP      ;CHECK IF STACK PROBLEM
2750 011274 001401              BEQ      1$              ;NO, CONTINUE
2751 011276 000000              HALT                      ; *** PROGRAM PROBLEM
2752 011300 004737 051240      1$:   JSR      PC,C.CPT      ;CALL COMMAND OPTIMIZER
2753 011304 004737 043314      JSR      PC,W.WTCH      ;CALL WATCH DOG TIMER
2754 011310 005737 001560      TST      ERCONT      ;CHECK IF IN SPECIAL ERROR SEQUENCE
2755 011314 100765              BMI      MAIN      ;YES, DO NOT PRINT INTERVAL STATISTICS CF
2756                                ; ACD DRIVES TO TESTING SEQUENCE
2757 011316 032777 060000 167614      BIT      #SW14,SW13,MSWR ;CHECK IF LOOP ON OPERATION OR
2758                                ; INHIBIT PRINT CUT
2759                                BNE      11$      ;YES, DO NOT PRINT CUT INTERVAL STATISTICS
2760 011326 105737 001627      TSTB     STATIS      ;CHECK FOR INTERVAL STATISTICS
2761 011332 001417              BEQ      11$      ;NO, CONTINUE SCAN
2762 011334 005004              CLR      R4              ;START WITH DRIVE 0
2763 011336 016405 001520      2$:   MOV      PPLKT(R4),R5 ;GET PARAMETER BLOCK ADDRESS
2764 011342 032765 000001 000014      BIT      #DRVUSE,P.PRST(R5) ;CHECK IF DRIVE IS IN USE
2765 011350 001402              BEQ      3$              ;NO, LOOK AT NEXT PARAMETER BLOCK
2766 011352 004737 012772      JSR      PC,STAT      ;GET STATISTICS
2767 011356 005724      3$:   TST      (R4)+      ;INCREMENT DRIVE INDEX
2768 011360 022704 000020      CMP      #20,R4      ;CHECK IF ALL DRIVES SCANNED
2769 011364 101364              BHI      2$              ;NO, DO NEXT DRIVE
2770 011366 105037 001627      CLMR     STATIS      ;RESET FLAG
2771
2772 011372 004037 050766      11$:  JSR      R0,Q.POP      ;GET FIRST DRIVE FROM DRIVE
2773 011376 001314              AVAILQ     ; AVAILABLE QUEUE
2774 011400 012605              MOV      (SP)+,R5      ;STACK PARAMETER BLOCK ADDRESS
2775 011402 001732              BEQ      MAIN      ;IF QUEUE EMPTY WAIT
2776 011404 032765 002000 000014      BIT      #DRPDRV,P.PRST(R5) ;CHECK IF DROP DRIVE
2777 011412 001403              BEQ      12$      ;NO, GENERATE NEW COMMAND
2778 011414 004737 012716      JSR      PC,DROP      ;DROP DRIVE
2779 011420 000723              BR       MAIN      ;GET NEXT AVAILABLE DRIVE
2780
2781 011422 032777 040000 167510 12$:  BIT      #SW14,MSWR      ;CHECK IF LOOP ON OPERATION
2782 011430 001002              BNE      1$              ;YES, DO NOT CLEAR SEEK TO PREVIOUS
2783                                ; CYLINDER FLAG
2784 011432 105065 000211              CLRP     P.SEEK(R5) ;CLEAR SEEK TO PREVIOUS CYLINDER
2785 011436 004737 020640      1$:   JSR      PC,GENERT ;GENERATE NEW COMMAND
2786 011442 105765 000211              TSTP     P.SEEK(R5) ;CHECK IF SEEK TO PREVIOUS CYLINDER
2787 011446 001310              BNE      MAIN      ;YES, GET NEXT AVAILABLE DRIVE
2788 011450 013737 001450 177776      MOV      RFPRI,PS ;LOCK OUT RK06 INTERRUPTS
2789 011456 005737 001320              TST      BWAITQ ;CHECK IF BUFFER WAIT QUEUE EMPTY
2790 011462 001013              BNE      25$      ;NO, ENQUEUE PARAMETER BLOCK
2791 011464 004037 022114              JSR      R0,BUFFAL ;ALLOCATE BUFFER
2792 011470 011512              25$:  CLRP     PS ;NOT ENOUGH ROOM RETURN
2793 011472 005037 177776              CLR      PS ;ALLOW RK06 INTERRUPTS
2794 011476 004737 022532              JSR      PC,LDDATA ;LOAD BUFFER
2795 011502 004037 050706              JSR      R0,Q.PUSH ;PUT PARAMETER BLOCK ON COMMAND
2796 011506 001324              CINITQ     ; INITIATION QUEUE
2797 011510 000667              BR       MAIN ;GET NEXT AVAILABLE DRIVE
2798
2799 011512 004037 050706      25$:  JSR      R0,Q.PUSH ;PUT PARAMETER BLOCK ON BUFFER
2800 011516 001320              BWAITQ     ; WAIT QUEUE
2801 011520 005037 177776              CLR      PS ;ALLOW RK06 INTERRUPTS
2802 011524 000661              BR       MAIN ;GET NEXT AVAILABLE DRIVE

```

```

2803 .SBTTL OPERATOR COMMAND DECODER
2804
2805 ;*****
2806 ;*
2807 ;* THE LEGAL COMMANDS ARE:
2808 ;* TN- INITIATE TESTING ON DRIVE N
2809 ;* PN- CHANGE PARAMETER AND INITIATE TEST ON
2810 ;* DRIVE N
2811 ;* DN- DROP DRIVE FROM TESTING SEQUENCE
2812 ;* SN- DEMAND PERFORMANCE SUMMARY ON DRIVE N
2813 ;* WN- WRITE RANDOM DATA PATTERNS ON DRIVE N
2814 ;* AND INITIATE TESTING.
2815 ;*
2816 ;*CALL JSR PC,OPPCMD
2817 ;* RETURN
2818 ;*
2819 ;*
2820 ;* COMMAND ROUTINE
2821 ;* -----
2822 ;*
2823 ;* TN TESTDV
2824 ;* PN PARMDV
2825 ;* DN DROPDV
2826 ;* SN STATDV
2827 ;* WN WRTEPR
2828 ;*
2829 ;*****
2830
2831 011526 010446 OPRCMD: MOV R4,-(SP) ;STORE R4 ON STACK
2832 011530 104401 055317 TYPE ,OPR007 ;TYPE "PLEASE ENTER COMMAND"
2833 011534 104403 RDLIN ;READ LINE FROM TTY
2834 011536 012604 MOV (SP)+,R4 ;STORE ADDRESS OF INPUT STRING
2835 011540 116437 000001 055154 MOVR 1(R4),OPR001 ;MOVE SECOND CHARACTER TO DRIVE
2836 ; REQUEST STORAGE
2837 011546 122737 000101 055154 CMPB #'A,OPR001 ;CHECK IF ALL DRIVES
2838 011554 001421 BEQ 4$ ;YES, PROCESS ALL DRIVES
2839 011556 122737 000060 055154 CMPB #'0,OPR001 ;CHECK IF LEGAL DRIVE NUMBER
2840 011564 101004 BHI 2$ ;NO, PRINT ERROR
2841 011566 122737 000067 055154 CMPB #'7,OPR001 ;CHECK IF 0-7
2842 011574 103006 BHI 3$ ;YES, PROCESS DRIVE COMMAND
2843 011576 104401 055137 2$: TYPE ,OPR000 ;TYPE "DRIVE NUMBER N ILLEGAL?"
2844 011602 104401 055157 TYPE ,OPR002
2845 011606 012604 MOV (SP)+,R4 ;RESTORE R4
2846 011610 000207 RTS PC ;RETURN
2847
2848 011612 142737 000370 055154 3$: BICB #'70,OPR001 ;CLEAR UNIMPORTANT BITS
2849 011620 010546 4$: MOV R5,-(SP) ;STORE R5 ON STACK
2850 011622 122714 000124 CMPB #'T,(P4) ;CHECK IF TEST DRIVE
2851 011626 001427 BFC TESTDV ;YES, GO TO TEST DRIVE
2852 011630 122714 000120 CMPB #'P,(R4) ;CHECK IF TEST AND CHANGE PARAMETERS
2853 011634 001512 BEQ PARMDV ;YES, GO TEST AND CHANGE PARAMETERS
2854 011636 122714 000104 CMPB #'D,(R4) ;CHECK IF DROP DRIVE
2855 011642 001553 BFC DROPDV ;YES, GO DROP DRIVE
2856 011644 122714 000123 CMPB #'S,(P4) ;CHECK IF DEMAND STATISTICS
2857 011650 001002 BNE 5$
2858 011652 000137 012352 JMP STATDV ;YES, GO TO DEMAND STATISTICS

```

2A59						
2A60	011656	122714	000127	55:	CMPP	W,M,(P4)
2A61	011662	001002			BNE	6S
2A62	011664	000137	012526		JPP	WRTEPF
2A63						
2A64	011670	111437	055131	6S:	MOVP	(P4),ILLCPD
2A65	011674	104401	055100		TYPE	,ILLCOM
2A66	011700	012605			MOV	(SP)+,R5
2A67	011702	012604			MOV	(SP)+,R4
2A68	011704	000207			RTS	PC

;CHECK IF WRITE BACK AND TEST
;YES, GO WRITE BACK AND TEST
;STORE COMMAND FOR PRINT OUT
;TYPE ERROR MESSAGE
;RESTORE R5
;RESTORE R4
;RETURN

```

2869 .SBTTL TEST DRIVE COMMAND
2870
2871 011706 122737 000101 055154 TESTDV: CMPP      #*A,OPR001      ;CHECK IF ALL DRIVES ARE TO BE TESTED
2872 011714 001027          BNE          10$          ;NO, NO SPECIFIC DRIVE
2873 011716 105037 001626          CLR          DRVCNT      ;CLEAR DRIVE COUNT
2874 011722 005004          CLR          R4          ;CLEAR DRIVE COUNT
2875 011724 016405 001520          1$: MOV      PBLKT(R4),R5 ;GET PARAMETER BLOCK ADDRESS
2876 011730 032765 000001 000014 BIT      #DRVUSE,P.PRST(R5) ;CHECK IF DRIVE IN USE
2877 011736 001004          BNE          2$          ;YES, GO TO NEXT DRIVE
2878 011740 004737 013250          JSR      PC,TEST      ;GO TEST DRIVE
2879 011744 105237 001626          INCR      DRVCNT      ;INCREMENT DRIVE COUNT
2880 011750 005724          2$: TST      (R4)+          ;ADDRESS NEXT DRIVE (ADD 2)
2881 011752 022704 000020          CMP      #20,R4      ;CHECK IF FINISHED
2882 011756 001362          BNE          1$          ;NO, SET UP NEXT DRIVE
2883 011760 105737 001626          TSTB     DRVCNT      ;CHECK IF ANY DRIVE IN ASSIGNMENT SEQUENCE
2884 011764 001033          BNE          15$         ;YES, RETURN
2885 011766 104401 055476          TYPF     ,OPR013      ;TYPE "ALL DRIVES CURRENTLY UNDER TEST?"
2886 011772 000430          BR          15$         ;RETURN
2887
2888 011774 113704 055154          10$: MOVP     OPR001,R4      ;STORE DRIVE NUMBER
2889 012000 006304          ASL          R4          ;MULTIPLY DRIVE NUMBER BY 2
2890 012002 016405 001520          MOV      PBLKT(R4),R5 ;GET PARAMETER ADDRESS
2891 012006 032765 000001 000014 BIT      #DRVUSE,P.PRST(R5) ;CHECK IF DRIVE ASSIGNED
2892 012014 001003          BNE          11$         ;YES, PRINT DRIVE ALREADY ASSIGNED
2893 012016 004737 013250          JSR      PC,TEST      ;GO TEST DRIVE
2894 012022 000414          BR          15$         ;RETURN
2895
2896 012024 013737 001450 177776 11$: MOV      RFPRI,PS      ;LOCK OUT RK06 INTERRUPTS
2897 012032 152737 000060 055154 BISH     #60,OPR001 ;MAKE NUMBER ASCII
2898 012040 104401 055137          TYPE     ,OPR000      ;TYPE "DRIVE N ALREADY ASSIGNED?"
2899 012044 104401 055205          TYPE     ,OPR004
2900 012050 005037 177776          CLR      PS          ;ALLOW RK06 INTERRUPTS
2901 012054 012605          15$: MOV      (SP)+,R5      ;RESTORE R5
2902 012056 012604          MOV      (SP)+,R4      ;RESTORE R4
2903 012060 000207          RTS      PC          ;RETURN
2904
2905 .SBTTL CHANGE PARAMETERS COMMAND
2906
2907 012062 122737 000101 055154 PARMDV: CMPP      #*A,OPR001      ;CHECK IF PARAMETERS FOR ALL DRIVES
2908 012070 001020          BNE          10$          ;NO, NO SPECIFIC DRIVE
2909 012072 005004          CLR          R4          ;CLEAR DRIVE COUNT
2910 012074 016405 001520          1$: MOV      PBLKT(R4),R5 ;GET PARAMETER BLOCK ADDRESS
2911 012100 004737 015756          JSR      PC,PAPM      ;ASK FOR PARAMETERS
2912 012104 032765 000001 000014 BIT      #DRVUSE,P.PRST(R5) ;CHECK IF DRIVE IN USE
2913 012112 001002          BNE          2$          ;YES, GO TO NEXT DRIVE
2914 012114 004737 013250          JSR      PC,TEST      ;GO TEST DRIVE
2915 012120 005724          2$: TST      (R4)+          ;ADDRESS NEXT DRIVE (ADD 2)
2916 012122 022704 000020          CMP      #20,R4      ;CHECK IF FINISHED
2917 012126 001362          BNE          1$          ;NO, SET UP NEXT DRIVE
2918 012130 000415          BR          15$         ;RETURN
2919
2920 012132 113704 055154          10$: MOVP     LPR001,R4      ;STORE DRIVE NUMBER
2921 012136 006304          ASL          R4          ;MULTIPLY DRIVE NUMBER BY 2
2922 012140 016405 001520          MOV      PBLKT(R4),R5 ;GET PARAMETER ADDRESS
2923 012144 004737 015756          JSR      PC,PAPM      ;GET NEW PARAMETERS
2924 012150 032765 000001 000014 BIT      #DRVUSE,P.PRST(R5) ;CHECK IF DRIVE IN USE

```

2925	012156	001002			BNE	15\$		;YES, RETURN
2926	012160	004737	013750		JSR	PC,TEST		;GO TEST DRIVE
2927	012164	012605			15\$: MOV	(SP)+,R5		;RESTORE R5
2928	012166	012604			MOV	(SP)+,R4		;RESTORE R4
2929	012170	000207			RTS	PC		;RETURN
2930								
2931					.SBTTL	DROP DRIVE COMMAND		
2932								
2933	012172	122737	000101	055154	DROPDV: CMPR	#*A,OPR001		;CHECK IF ALL DRIVES ARE TO BE DROPPED
2934	012200	001030			BNE	10\$		;NO, DO SPECIFIC DRIVE
2935	012202	105037	001626		CLRP	DRVCNT		;CLEAR DRIVE COUNT
2936	012206	005004			CLR	R4		;CLEAR DRIVE COUNT
2937	012210	016405	001520		15\$: MOV	PRLKT(R4),R5		;GET PARAMETER PLCKR ADDRESS
2938	012214	032765	000001	000014	BIT	#DRVUSE,P.PRST(R5)		;CHECK IF DRIVE IN USE
2939	012222	001405			BEQ	2\$		;NO, GO TO NEXT DRIVE
2940	012224	052765	002000	000014	BIS	#DRPDV,P.PRST(R5)		;INDICATE DRIVE IS TO BE DROPPED
2941	012232	105237	001626		INCR	DRVCNT		;INCREMENT DRIVE COUNT
2942	012236	005724			25\$: TST	(R4)+		;ADDRESS NEXT DRIVE (ADD 2)
2943	012240	022704	000020		CMP	#20,R4		;CHECK IF FINISHED
2944	012244	001361			BNE	1\$		;NO, SET UP NEXT DRIVE
2945	012246	105737	001626		TSTB	DRVCNT		;CHECK IF ANY DRIVES AVAILABLE
2946	012252	001034			BNE	15\$		;YES, RETURN
2947	012254	104401	055540		TYPE	,OPR014		;TYPE "NO DRIVES IN USE?"
2948	012260	000431			BR	15\$		;RETURN
2949								
2950	012262	113704	055154		10\$: MOVB	OPR001,R4		;STORE DRIVE NUMBER
2951	012266	006304			ASL	R4		;MULTIPLY DRIVE NUMBER BY 2
2952	012270	016405	001520		MOV	PRLKT(R4),R5		;GET PARAMETER ADDRESS
2953	012274	032765	000001	000014	BIT	#DRVUSE,P.PRST(R5)		;CHECK IF DRIVE ASSIGNED
2954	012302	001404			BFC	11\$		;NO, PRINT DRIVE N NOT ASSIGNED
2955	012304	052765	002000	000014	BIS	#DRPDV,P.PRST(R5)		;INDICATE DRIVE IS TO BE DROPPED
2956	012312	000414			BR	15\$		;RETURN
2957								
2958	012314	013737	001450	177776	11\$: MOV	KPRI,PS		;LOCK OUT RK06 INTERRUPTS
2959	012322	152737	000060	055154	BISB	#60,OPR001		;MAKE NUMBER ASCII
2960	012330	104401	055137		TYPE	,OPR000		;TYPE "DRIVE N NOT ASSIGNED?"
2961	012334	104401	055231		TYPE	,OPR005		
2962	012340	005037	177776		CLR	PS		;ALLOW RK06 INTERRUPTS
2963	012344	012605			15\$: MOV	(SP)+,R5		;RESTORE R5
2964	012346	012604			MOV	(SP)+,R4		;RESTORE R4
2965	012350	000207			RTS	PC		;RETURN
2966								
2967					.SBTTL	GET DRIVE STATISTICS COMMAND		
2968								
2969	012357	122737	000101	055154	STATDV: CMPR	#*A,OPR001		;CHECK IF STATISTICS FOR ALL DRIVES
2970	012360	001027			BNE	10\$		;NO, DO SPECIFIC DRIVE
2971	012362	105037	001626		CLRP	DRVCNT		;CLEAR DRIVE COUNT
2972	012366	005004			CLR	R4		;CLEAR DRIVE COUNT
2973	012370	016405	001520		15\$: MOV	PRLKT(R4),R5		;GET PARAMETER PLCKR ADDRESS
2974	012374	032765	000001	000014	BIT	#DRVUSE,P.PRST(R5)		;CHECK IF DRIVE IN USE
2975	012402	001404			BFC	2\$		;NO, GO TO NEXT DRIVE
2976	012404	004737	012772		JSR	PC,STAT		;GATHER STATISTICS
2977	012410	105237	001626		INCR	DRVCNT		;INCREMENT DRIVE COUNT
2978	012414	005724			25\$: TST	(R4)+		;ADDRESS NEXT DRIVE (ADD 2)
2979	012416	022704	000020		CMP	#20,R4		;CHECK IF FINISHED
2980	012422	001362			BNE	1\$		;NO, SET UP NEXT DRIVE

```

2981 012424 105737 001626          TSTB  DRVCNT      ;CHECK IF ANY DRIVES AVAILABLE
2982 012430 001033          BNE  15$      ;YES, RETURN
2983 012432 104401 055540          TYPF  ,OPR014    ;TYPE "NO DRIVES IN USE?"
2984 012436 000430          BR    15$      ;RETURN
2985
2986 012440 113704 055154          10$: MOVB  OPR001,P4    ;STORE DRIVE NUMBER
2987 012444 006304          ASL  R4        ;MULTIPLY DRIVE NUMBER BY 2
2988 012446 016405 001520          MOV  PBLKT(R4),R5    ;GET PARAMETER ADDRESS
2989 012452 032765 000001 000014  BIT  #DRVUSE,P.PRST(5) ;CHECK IF DRIVE ASSIGNED
2990 012460 001403          BEQ  11$      ;PRINT DRIVE # NOT ASSIGNED
2991 012462 004737 012777          JSR  PC,STAT    ;GO GATHER DRIVE STATISTICS
2992 012466 000414          BR    15$      ;RETURN
2993
2994 012470 013737 001450 177776 11$: MOV  RKPRI,PS    ;LOCK OUT RK06 INTERRUPTS
2995 012476 152737 000060 055154  BISH  #60,OPR001  ;MAKE NUMBER ASCII
2996 012504 104401 055137          TYPE ,OPR000    ;TYPE "DRIVE # NOT ASSIGNED?"
2997 012510 104401 055231          TYPF  ,OPR005
2998 012514 005037 177776          CLR  PS        ;ALLOW RK06 INTERRUPTS
2999 012520 012605          15$: MOV  (SP)+,R5    ;RESTORE R5
3000 012522 012604          MOV  (SP)+,R4    ;RESTORE R4
3001 012524 000207          RTS  PC        ;RETURN
3002
3003          .SBTTL  WRITE PACK AND TEST COMMAND
3004
3005 012526 122737 000101 055154  WRTEPK: CMPB  #'A,OPR001    ;CHECK IF ALL DRIVES ARE TO BE TESTED
3006 012534 001032          BNE  10$      ;NO, DO SPECIFIC DRIVE
3007 012536 105037 001626          CLRP  DRVCNT    ;CLEAR DRIVE COUNT
3008 012542 005004          CLR  R4        ;CLEAR DRIVE COUNT
3009 012544 016405 001520          1$: MOV  PBLKT(R4),R5    ;GET PARAMETER BLOCK ADDRESS
3010 012550 032765 000001 000014  BIT  #DRVUSE,P.PRST(5) ;CHECK IF DRIVE IN USE
3011 012556 001007          BNE  2$      ;YES, GO TO NEXT DRIVE
3012 012560 112765 000020 000210  MOVB  #BIT4,P.ASSN(R5) ;SET FLAG FOR WRITE PACK
3013 012566 004737 013750          JSR  PC,TEST    ;GO ASSIGN DRIVE
3014 012572 105237 001626          INCR  DRVCNT    ;INCREMENT DRIVE COUNT
3015 012576 005724          2$: TST  (R4)+    ;ADDRESS NEXT DRIVE (ADD 2)
3016 012600 022704 000020          CMP  #20,R4    ;CHECK IF FINISHED
3017 012604 001757          BNE  1$      ;NO, SET UP NEXT COMMAND
3018 012606 105737 001626          TSTB  DRVCNT    ;CHECK IF ANY DRIVES NOT IN USE
3019 012612 001436          BEQ  15$      ;NO, RETURN
3020 012614 104401 055476          TYPE  ,OPR013    ;TYPE "ALL DRIVES CURRENTLY UNDER TEST"
3021 012620 000437          BR    15$      ;RETURN
3022
3023 012622 113704 055154          10$: MOVB  OPR001,P4    ;STORE DRIVE NUMBER
3024 012626 006304          ASL  R4        ;MULTIPLY DRIVE NUMBER BY 2
3025 012630 016405 001520          MOV  PBLKT(R4),R5    ;GET PARAMETER ADDRESS
3026 012634 032765 000001 000014  BIT  #DRVUSE,P.PRST(5) ;CHECK IF DRIVE ASSIGNED
3027 012642 001006          BNE  11$      ;YES, PRINT DRIVE ALREADY ASSIGNED
3028 012644 112765 000020 000210  MOVB  #BIT4,P.ASSN(R5) ;SET FLAG FOR WRITE PACK
3029 012652 004737 013250          JSR  PC,TEST    ;GO ASSIGN DRIVE
3030 012656 000414          BR    15$      ;RETURN
3031
3032 012660 013737 001450 177776 11$: MOV  RKPRI,PS    ;LOCK OUT RK06 INTERRUPTS
3033 012666 152737 000060 055154  BISH  #60,OPR001  ;MAKE NUMBER ASCII
3034 012674 104401 055137          TYPE ,OPR000    ;TYPE "DRIVE # ALREADY ASSIGNED?"
3035 012700 104401 055205          TYPE  ,OPR004
3036 012704 005037 177776          CLR  PS        ;ALLOW RK06 INTERRUPTS

```

3037	012710	012605	155:	MOV	(SP)+,R5	;PESTORE R5
3038	012712	012604		MOV	(SP)+,R4	;RESTORE R4
3039	012714	000207		RTS	PC	;RETURN
3040						

.SBTTL DPOP DRIVE AND GATHER STATISTICS ROUTINES

3041					
3042					
3043	012716	013746	177776		
3044	012722	013737	001450	177776	
3045	012730	005065	000014		
3046	012734	105065	000210		
3047	012740	005065	000212		
3048	012744	116537	000000	055446	
3049	012752	152737	000060	055446	
3050	012760	104401	055427		
3051	012764	104401	055251		
3052	012770	000405			
3053					
3054	012772	013746	177776		
3055	012776	013737	001450	177776	
3056	013004	104401	055615		
3057	013010	116537	000000	055446	
3058	013016	152737	000060	055446	
3059	013024	104401	055427		
3060	013030	004737	027060		
3061	013034	004737	043072		
3062	013040	104401	055673		
3063	013044	010546			
3064	013046	062716	000152		
3065	013052	004737	054344		
3066	013056	104401	055715		
3067	013062	010546			
3068	013064	062716	000160		
3069	013070	004737	054344		
3070	013074	104401	055742		
3071	013100	010546			
3072	013102	062716	000166		
3073	013106	004737	054344		
3074	013112	104401	055764		
3075					
3076	013116	016546	000176		
3077	013122	004737	054302		
3078	013126	104401	056017		
3079	013132	016546	000177		
3080	013136	004737	054302		
3081	013142	104401	056033		
3082	013146	016546	000174		
3083	013152	004737	054302		
3084	013156	104401	056047		
3085	013162	016546	000200		
3086	013166	004737	054302		
3087	013172	104401	056073		
3088	013176	016546	000202		
3089	013202	004737	054302		
3090	013206	104401	056117		
3091	013212	016546	000204		
3092	013216	004737	054302		
3093	013222	104401	056165		
3094	013226	016546	000206		
3095	013232	004737	054302		
3096	013236	104401	001165		

DROP:	MOV	PS,-(SP)	;STORE PSW
	MOV	RKPPI,PS	;LOCK OUT TTY AND RK06 INTERRUPTS
	CLR	P.PRST(R5)	;CLEAR PROGRAM STATUS REGISTER
	CLRB	P.ASSN(R5)	;CLEAR ASSIGNMENT CODE
	CLR	P.ERR(R5)	;CLEAR ERROR FLAGS
	MOVB	P.DRVN(R5),OPR011	;LOAD DRIVE NUMBER
	BISB	#60,OPR011	;MAKE IT ASCII
	TYPE	,OPR010	;TYPE "DRIVE # HAS BEEN DROPPED
	TYPE	,OPR006	; FROM TEST SEQUENCE"
	BR	STAT00	;GO PRINT STATISTICS
STAT:	MOV	PS,-(SP)	;STORE PSW
	MOV	RKPPI,PS	;LOCK OUT TTY AND RK06 INTERRUPTS
STAT00:	TYPE	,DRVSTT	;**DRIVE STATISTICS
	MOVB	P.DRVN(R5),OPR011	;LOAD DRIVE NUMBER FOR PRINT OUT
	BISB	#60,OPR011	;MAKE IT ASCII
	TYPE	,OPR010	;TYPE "DRIVE #"
	JSR	PC,PRISER	;PRINT DRIVE AND PACK SERIAL NUMBERS
	JSR	PC,PRITIM	;PRINT TIME IF CLOCK AVAILABLE
	TYPE	,STT001	;TYPE "ORDERS PERFORMED"
	MOV	R5,-(SP)	;GET PARAMETER BLOCK ADDRESS
	ADD	#P.NODR,(SP)	;CALCULATE ADDRESS FOR PRINT OUT
	JSR	PC,\$DB2D	;CONVERT TO ASCII
	TYPE	,STT002	;TYPE "WORDS WRITTEN*65K"
	MOV	R5,-(SP)	;GET PARAMETER BLOCK ADDRESS
	ADD	#P.WWPT+2,(SP)	;CALCULATE ADDRESS FOR PRINT OUT
	JSR	PC,\$DB2D	;CONVERT TO ASCII
	TYPE	,STT003	;TYPE "WORDS READ*65K"
	MOV	R5,-(SP)	;GET PARAMETER BLOCK ADDRESS
	ADD	#P.NRD+2,(SP)	;CALCULATE ADDRESS FOR PRINT OUT
	JSR	PC,\$DB2D	;CONVERT TO ASCII
	TYPE	,STT004	;TYPE "SOFT DATA ERRORS"
			; "ECC"
	MOV	P.SFCC(R5),-(SP)	;STORE SOFT ECC ERROR COUNT
	JSR	PC,\$SB2D	;CONVERT TO DECIMAL
	TYPE	,STT005	;TYPE "WEREAD"
	MOV	P.SRRD(R5),-(SP)	;SAVE SOFT REREAD CORRECTABLE
	JSR	PC,\$SB2D	;CONVERT TO DECIMAL
	TYPE	,STT006	;TYPE "OFFSET"
	MOV	P.SOFF(R5),-(SP)	;SAVE SOFT OFFSET CORRECTABLE
	JSR	PC,\$SB2D	;CONVERT TO DECIMAL
	TYPE	,STT007	;TYPE "HARD DATA ERRORS"
	MOV	P.HARD(R5),-(SP)	;SAVE HARD DATA ERRORS
	JSR	PC,\$SB2D	;CONVERT TO DECIMAL
	TYPE	,STT008	;TYPE "SEEK INCOMPLETES"
	MOV	P.NSKI(R5),-(SP)	;TYPE NUMBER OF SEEK INCOMPLETES
	JSR	PC,\$SB2D	;CONVERT TO DECIMAL
	TYPE	,STT009	;TYPE "OPERATION INCOMPLETES"
	MOV	P.NOPI(R5),-(SP)	;SAVE NUMBER OF OPERATION INCOMPLETES
	JSR	PC,\$SB2D	;CONVERT TO DECIMAL
	TYPE	,STT010	;TYPE "OTHER ERRORS"
	MOV	P.NFR(R5),-(SP)	;SAVE NUMBER OF OTHER ERRORS
	JSR	PC,\$SB2D	;CONVERT TO DECIMAL
	TYPE	,SCRLF	;TYPE <CR><LF>

3097	013242	012637	177776	MOV	(SP)+,PS	;RESTORE PSW
3098	013746	000207		RTS	PC	;RETURN
3099						

```

3100      .SBTTL  INITIATE DRIVE ASSIGNMENT SEQUENCE
3101
3102 013250 116546 000710      TEST:  MOVW  P.ASSN(R5),-(SP) ;SAVE ASSIGNMENT CODE
3103 013254 010446           MOV    R4,-(SP)      ;STORE R4 ON STACK
3104 013256 010346           MOV    R3,-(SP)      ;STORE R3 ON STACK
3105 013260 010503           MOV    R5,R3        ;LOAD PARAMETER BLOCK ADDRESS INTO R3
3106 013262 062703 000122     ADD    #P.RDPT,R3   ;CALCULATE BEGINNING OF STATISTICS AND
3107                                     ; ERROR AREA
3108 013266 010504           MOV    R5,R4        ;LOAD PARAMETER BLOCK ADDRESS INTO R4
3109 013270 062704 000237     ADD    #P.EXAR,R4   ;CALCULATE END OF STATISTICS AND
3110                                     ; PERGR AREA
3111 013274 005023           1S:   CLR    (R3)+    ;CLEAR STATISTIC AND ERROR AREA
3112 013276 020304           CMP    R3,R4        ;CHECK IF FINISHED
3113 013300 001375           BNE    1S            ;NO, CONTINUE
3114 013302 116504 000000     MOVW  P.DRVN(R5),R4 ;STORE DRIVE NUMBER
3115 013306 156437 001510 001507 BISH  INTMSK(P4),0,OVER ;SET UP FOR OVERLAPPING SEKS
3116 013314 012603           MOV    (SP)+,R3    ;RESTORE R3
3117 013316 012604           MOV    (SP)+,R4    ;RESTORE R4
3118 013320 112665 000210     MOVW  (SP)+,P.ASSN(P5) ;RESTORE ASSIGNMENT CODE
3119
3120 013324 105065 000117     TEST1: CLRB  P.ECMP(R5) ;CLEAR ECC COMPARE FLAG
3121 013330 152765 000001 000210 BISH  #BIT0,P.ASSN(R5) ;SET READ STATUS COMMAND FLAG
3122 013336 052765 000001 000014 BIS   #DRVUSE,P.PRST(R5) ;SET DRIVE IN USE
3123 013344 112765 000141 000001 MOVW  #RDSTAT,P.CMND(R5) ;READ ALL DRIVE STATUS
3124 013352 112765 000141 000123 MOVW  #RDSTAT,P.RCMD(R5)
3125 013360 004037 050706     JSR    R0,Q.PUSH  ;ENQUEUE PARAMETER BLOCK IN
3126 013364 001324           CINITQ ; COMMAND INITIATION QUEUE
3127 013366 000207           RTS    PC            ;RETURN

```

.SBTTL DRIVE ASSIGNMENT SEQUENCE

```

3128
3129
3130 013370 132765 000001 000210 ASNORM: BITH #BIT0,P.ASSN(R5) ;CHECK IF READ STATUS
3131 013376 001460 BEQ ASN4$ ;NO, CHECK IF START SPINDLE
3132 013400 142765 000001 000210 BICR #BIT0,P.ASSN(P5) ;CLEAR READ STATUS COMMAND ISSUED
3133 013406 016565 000054 000224 MOV P.A11(R5),P.SERL(R5) ;STORE SERIAL NUMBER
3134 013414 042765 100007 000224 BIC #C<N.SER>,P.SERL(R5) ;KEEP SERIAL NUMBER
3135 013422 006265 000224 ASR P.SERL(R5) ;SHIFT 3 BITS RIGHT
3136 013426 006265 000224 ASR P.SERL(R5)
3137 013432 006265 000224 ASR P.SERL(R5)
3138 013436 032765 000200 000036 BIT #DRDY,P.DS(R5) ;CHECK IF DRIVE READY
3139 013444 001415 BEQ ASN2$ ;NO, ISSUE START SPINDLE
3140 013446 116565 000123 000135 ASN1$: MOVR P.RCMD(R5),P.LCMD(R5) ;STORE PREVIOUS COMMAND
3141 013454 112765 000103 000001 MOVR #PACK,P.CMND(R5) ;ISSUE PACK ACKNOWLEDGE
3142 013462 112765 000103 000123 MOVB #PACK,P.RCMD(R5)
3143 013470 152765 000004 000210 BISR #BIT2,P.ASSN(R5) ;SET PACK ACKNOWLEDGE FLAG
3144 013476 000414 BR ASN3$ ;ISSUE PACK ACKNOWLEDGE AND WAIT
3145
3146 013500 116565 000123 000135 ASN2$: MOVR P.RCMD(R5),P.LCMD(R5) ;STORE PREVIOUS COMMAND
3147 013506 112765 000111 000001 MOVB #SRTSPL,P.CMND(P5) ;ISSUE START SPINDLE
3148 013514 112765 000111 000123 MOVB #SRTSPL,P.RCMD(R5)
3149 013522 152765 000002 000210 BISR #BIT1,P.ASSN(R5) ;SET START SPINDLE FLAG
3150 013530 004037 050706 ASN3$: JSR RO,Q.PUSH ;ENQUEUE PARAMETER BLOCK IN
3151 013534 001324 CINITQ ; COMMAND INITIATION QUEUE
3152 013536 000207 RTS PC ;RETURN
3153
3154 013540 132765 000002 000210 ASN4$: BITH #BIT1,P.ASSN(R5) ;CHECK IF START SPINDLE
3155 013546 001414 BEQ 15$ ;NO, CHECK IF PACK ACKNOWLEDGE
3156 013550 142765 000002 000210 BICR #BIT1,P.ASSN(R5) ;CLEAR START SPINDLE FLAG
3157 013556 032765 000200 000036 BIT #DRDY,P.DS(R5) ;CHECK IF DRIVE READY
3158 013564 001330 BNE ASN1$ ;YES, ISSUE PACK ACKNOWLEDGE
3159 013566 052765 002000 000212 BIS #BIT10,P.ERR(R5) ;SET DRIVE NOT READY AFTER START SPINDLE
3160 013574 000137 024766 JMP ABNORM ;RECEIPT ERROR
3161
3162 013600 132765 000004 000210 15$: BITH #BIT2,P.ASSN(R5) ;CHECK IF PACK ACKNOWLEDGE
3163 013606 001431 BEQ 16$ ;NO, CHECK IF RECAL
3164 013610 142765 000004 000210 BICR #BIT2,P.ASSN(R5) ;CLEAR PACK ACKNOWLEDGE FLAG
3165 013616 032765 000100 000036 BIT #VV,P.DS(R5) ;CHECK IF VOLUME VALID IS SET
3166 013624 001005 BNE 16$ ;YES, CHECK IF PACK IS TO BE WRITTEN
3167 013626 052765 004000 000212 BIS #BIT11,P.FRR(R5) ;SET VOLUME VALID NOT SET AFTER PACK
3168 ; ACKNOWLEDGE
3169 013634 000137 024766 JMP ABNORM ;RECEIPT ERROR
3170
3171 013640 116565 000123 000135 16$: MOVB P.RCMD(R5),P.LCMD(R5) ;STORE PREVIOUS COMMAND
3172 013646 112765 000113 000001 MOVR #RECAL,P.CMND(R5) ;ISSUE RECAL
3173 013654 112765 000113 000123 MOVB #RECAL,P.RCMD(R5)
3174 013662 152765 000100 000210 BISR #BIT6,P.ASSN(R5) ;SET RECAL ISSUED
3175 013670 000717 BR ASN7$
3176
3177 013672 132765 000100 000210 18$: BITH #BIT6,P.ASSN(R5) ;CHECK IF RECAL
3178 013700 001455 BEQ 23$ ;NO, CHECK IF RECAL PACK SERIAL NUMBER
3179 013702 142765 000100 000210 BICR #BIT6,P.ASSN(R5) ;RESET RECAL ISSUED
3180 013710 152765 000010 000210 BISR #BIT3,P.ASSN(R5) ;SET ASSIGNMENT CODE FOR PACK
3181 ; SERIAL NUMBER READ
3182 013716 012765 001000 000004 MOV #1000,P.SECT(R5) ; TRACK 2 SECTOR 0
3183 013724 116565 000123 000135 MOVR P.RCMD(R5),P.LCMD(R5) ;STORE PREVIOUS COMMAND

```

3184	013732	112765	000121	000001	20\$:	MOVB	#RDATA,P.CMND(R5)	;LOAD READ COMMAND FOR PACK
3185	013740	112765	000121	000123		MOVB	#RDATA,P.RCMD(R5)	; SERIAL NUMBER
3186	013746	012765	177774	000012		MOV	#-4,P.WC(R5)	;LOAD WORD COUNT FOR THE FIRST FOUR WORDS
3187	013754	012765	000632	000002		MOV	#410.,P.CYL(R5)	;SET CYLINDER TO 410
3188	013762	012765	000632	000124		MOV	#410.,P.RCYL(R5)	
3189	013770	016565	000004	000126		MOV	P.SECT(R5),P.RSEC(R5)	;STORE TRACK AND SECTOR
3190	013776	005737	001320			TST	BWAITQ	;CHECK IF BUFFER WAIT QUEUE IS EMPTY
3191	014002	001010				BNE	22\$	;NO, ENQUEUE PARAMETER BLOCK IN BUFFER
3192								; WAIT QUEUE
3193	014004	004037	022114			JSR	RO,BUFFAL	;ALLOCATE BUFFER
3194	014010	014024				22\$		;NOT ENOUGH ROOM RETURN
3195	014012	005037	177776			CLR	PS	;ALLOW RK06 INTERRUPTS
3196	014016	004737	022532			JSR	PC,LODATA	;CLEAR BUFFER
3197	014022	000642				BP	ASH3\$	;ISSUE COMMAND
3198								
3199	014024	004037	050706		22\$:	JSR	RO,Q.PUSH	;PUT PARAMETER BLOCK ON BUFFER
3200	014030	001320				BWAITQ		; WAIT QUEUE
3201	014032	000207				RTS	PC	;RETURN
3202								
3203	014034	032777	040000	165076	23\$:	BIT	#SW14,#SHR	;CHECK IS CYCLE ON OPERATION
3204	014042	001542				BEQ	43\$	;NO, CONTINUE WITH ASSIGNMENT PROCESS
3205	014044	105765	000211			TSTB	P.SEEK(R5)	;CHECK IF SEEK TO PREVIOUS POSITION
3206	014050	001444				BEQ	28\$	;NO, SERVICE INTERRUPT
3207	014052	105065	000211			CLRB	P.SEEK(R5)	;CLEAR FLAG
3208	014056	116565	000123	000001		MOVB	P.RCMD(R5),P.CMND(R5)	;REISSUE LAST COMMAND
3209	014064	016565	000124	000002		MOV	P.RCYL(R5),P.CYL(R5)	;LOAD CYLINDER ADDRESS
3210	014072	016565	000126	000004		MOV	P.RSEC(R5),P.SECT(R5)	;LOAD PREVIOUS TRACK AND SECTOR
3211	014100	016565	000130	000012		MOV	P.RWC(R5),P.WC(R5)	;LOAD PREVIOUS WORD COUNT
3212	014106	116565	000122	000121		MOVB	P.RDPT(R5),P.DPAT(R5)	;LOAD PREVIOUS PATTERN
3213	014114	132765	000010	000210		BITB	#BIT3,P.ASSN(R5)	;CHECK IF READ PACK SERIAL NUMBER
3214	014122	001003				BNE	25\$	;YES, DO NOT SET WRITE BEFORE WRITE CHECK
3215	014124	052765	000200	000014		BIS	#W.WCK,P.PRST(R5)	;SET WRITE BEFORE WRITE CHECK
3216	014132	005737	001320		25\$:	TST	BWAITQ	;CHECK IF ANY DRIVE WAITING FOR BUFFER
3217	014136	001332				BNE	22\$	;YES, ENQUEUE PARAMETER BLOCK
3218	014140	004037	022114			JSR	RO,BUFFAL	;GO ALLOCATE BUFFER
3219	014144	014024				22\$		;NOT ENOUGH ROOM RETURN
3220	014146	005037	177776			CLR	PS	;ALLOW RK06 INTERRUPTS
3221	014152	004737	022532			JSR	PC,LODATA	;GO LOAD DATA
3222	014156	000137	013530			JMP	ASH3\$	;GO ISSUE COMMAND
3223								
3224	014162	004037	024506		28\$:	JSR	RO,CHKADD	;CHECK SECTOR, TRACK, CYLINDER,
3225								; WORD COUNT, AND BUS ADDRESS
3226	014166	015074				60\$		;FRROR RETURN
3227	014170	132765	000010	000210		BITB	#BIT3,P.ASSN(R5)	;CHECK IF READ PACK SERIAL NUMBER
3228	014176	001424				BEQ	35\$	;NO, RELEASE BUFFER
3229	014200	010446				MOV	R4,-(SP)	;STORE R4 ON STACK
3230	014202	016504	000010			MOV	P.BALC(R5),R4	;LOAD R4 TO GET PACK SERIAL NUMBER
3231	014206	012465	000226			MOV	(R4)+,P.PKSR(R5)	;GET PACK SERIAL NUMBER
3232	014212	012465	000230			MOV	(R4)+,P.PKSR+7(R5)	
3233	014216	005724				TST	(R4)+	;ADJUST R4 TO ADDRESS ALIGNMENT INDICATION
3234	014220	005714				TST	(R4)	;CHECK IF DATA PACK
3235	014222	001423				BEQ	37\$	;YES, CONTINUE
3236	014224	012604			30\$:	MOV	(SP)+,R4	;RESTORE R4
3237	014226	004737	022316			JSR	PC,PUPREL	;RELEASE BUFFER
3238	014232	104401	055562			TYPE	,OPR015	;TYPE "ALIGNMENT PACK IN DRIVE"
3239	014236	004737	012716			JSR	PC,DROP	;GO DROP DRIVE

3240	014242	004737	022016			JSR	PC,GETBUF	;GO GET BUFFER FOR NEXT COMMAND
3241	014246	000207				RTS	PC	;RETURN
3242								
3243	014250	032765	000700	000014	35\$:	BIT	#W.WCF,P.PRST(R5)	;CHECK IF WRITE CHECK HAS BEEN ISSUED
3244	014256	001406				BFG	38\$	;YES, GC RELEASE BUFFER
3245	014260	042765	000200	000014		BIC	#W.WCF,P.PRST(R5)	;CLEAR WRITE BEFORE WRITE CHECK
3246	014266	000137	013530			JMP	ASN3\$	;GO ISSUE COMMAND
3247								
3248	014272	012604			37\$:	MOV	(SP)+,R4	;RESTORE R4
3249	014274	004737	022316		38\$:	JSR	PC,PUTRFL	;RELEASE BUFFER
3250	014300	032765	002000	000014		BIT	#DRPDV,P.PRST(R5)	;CHECK IF DROP DRIVE
3251	014306	001402				BFG	40\$	;NO, SEEK TO PREVIOUS POSITION
3252	014310	000137	015626			JMP	WV1\$	;GO DROP DRIVE
3253								
3254	014314	112765	177777	000211	40\$:	MOVB	#-1,P.SEEK(R5)	;SET FLAG
3255	014322	016565	000136	000002		MOV	P.LCYL(R5),P.CYL(R5)	;LOAD LAST CYLINDER
3256	014330	016565	000140	000004		MOV	P.LSEC(R5),P.SECT(R5)	;LOAD LAST SECTOR AND TRACK
3257	014336	112765	000117	000001		MOV	#SEEK,P.CMD(R5)	;ISSUE SEEK
3258	014344	000137	013530			JMP	ASN3\$	;GO SEEK TO PREVIOUS POSITION
3259								
3260	014350	132765	000010	000210	43\$:	BITB	#BIT3,P.ASSN(R5)	;CHECK IF READ PACK SERIAL NUMBER
3261	014356	001002				BNE	45\$	;YES, SET UP FOR READ
3262	014360	000137	015076			JMP	WRITER	;CHECK IF WRITE PACK
3263								
3264	014364	132765	000200	000210	45\$:	BITB	#BIT7,P.ASSN(R5)	;CHECK IF RECAL AND RETRY
3265								; READ OF PACK SERIAL NUMBER
3266	014372	001427				BEQ	48\$	;NO, SERVICE READ OF PACK SERIAL NUMBER
3267	014374	142765	000200	000210		BICB	#BIT7,P.ASSN(R5)	;RESET RECAL FOR RECAL OF PACK SERIAL NO.
3268	014402	116565	000123	000135		MOVB	P.RCMD(R5),P.LCMD(R5)	;STORE LAST COMMAND
3269	014410	016565	000124	000136		MOV	P.RCYL(R5),P.LCYL(R5)	;STORE LAST CYLINDER
3270	014416	016565	000126	000140		MOV	P.RSEC(R5),P.LSEC(R5)	;STORE LAST SECTOR AND TRACK
3271	014424	016565	000130	000142		MOV	P.RWC(R5),P.LWC(R5)	;STORE LAST WORD COUNT
3272	014432	062765	000002	000126		ADD	#2,P.RSEC(R5)	;TRY NEXT 16-BIT FORMAT SECTOR
3273	014440	016565	000126	000004		MOV	P.RSEC(R5),P.SECT(R5)	;STORE SECTOR TO BE READ
3274	014446	000137	013732			JMP	20\$	;GO ISSUE RPAR OF PACK SERIAL NUMBER
3275								
3276	014452	142765	000010	000210	48\$:	BICB	#BIT3,P.ASSN(R5)	;CLEAR FLAG
3277	014460	004037	024506			JSR	R0,CHKADD	;CHECK IF DATA TRANSFER CORRECT
3278	014464	015074				60\$		;ERROR RETURN
3279	014466	010446				MOV	R4,-(SP)	;STORE R4
3280	014470	016504	000132			MOV	P.RPAL(R5),R4	;LOAD R4 TO ADDRESS SERIAL NUMBER
3281	014474	012465	000226			MOV	(R4)+,P.PRSP(R5)	;LOAD LEAST SIGNIFICANT BITS
3282								; OF SERIAL NUMBER
3283	014500	012465	000230			MOV	(R4)+,P.PRSP+2(R5)	;LOAD MOST SIGNIFICANT BITS
3284								; OF SERIAL NUMBER
3285	014504	005724				TST	(R4)+	;ADJUST R4 TO ADDRESS ALIGNMENT INDICATION
3286	014506	005714				TST	(R4)	;CHECK IF ALIGNMENT PACK
3287	014510	001245				BNE	30\$	;YES, DROP DRIVE
3288	014512	012604				MOV	(SP)+,R4	;RESTORE R4
3289	014514	004737	022316			JSR	PC,PUTRFL	;RELEASE BUFFER
3290	014520	132765	000020	000210		BITB	#BIT4,P.ASSN(R5)	;CHECK IF WRITE PACK
3291	014526	001042				BNE	55\$	;YES, SET UP PATTERN AND ALLOCATE BUFFER
3292	014530	105765	000076			TSTB	P.RATE(R5)	;CHECK IF READ ONLY MODE
3293	014534	001413				BFG	52\$	;YES, START TESTING
3294	014536	032765	004000	000036		BIT	#WRL,P.DS(R5)	;CHECK IF WRITE LOCKED
3295	014544	001407				BFG	52\$	;NO, START TESTING

```
3296
3297 014546 104401 055451          50$: TYPE ,OPR012 ;TYPE "DRIVE WRITE LOCKED"
3298 014552 004737 012716          JSR PC,DROP ;DROP DRIVE FROM TEST SEQUENCE
3299 014556 004737 022016          JSR PC,GETBUF ;GET BUFFER FOR NEXT COMMAND
3300 014562 000207                  RTS PC ;RETURN
3301
3302 014564 116537 000000 055446 52$: MOVN P.DRW(R5),OPR011 ;LOAD CRIVE NUMBER
3303 014572 152737 000060 055446 BITN #60,OPR011 ;MAKE IT ASCII
3304 014600 104401 055427          TYPE ,OPR010 ;TYPE "DRIVE NUMBER & UNDER TEST"
3305 014604 104401 055172          TYPE ,OPR003
3306 014610 004737 027060          JSR PC,PRISEL ;PRINT DRIVE AND PACK SERIAL NUMBERS
3307 014614 004737 043072          JSR PC,PRITIM ;PRINT TIME
3308 014620 104401 001165          TYPE ,SCRLF
3309 014624 004037 050706          JSR RO,Q.PUSH ;PUT PARAMETER BLOCK IN DRIVE
3310 014630 001314              AVAILQ ; AVAILABLE QUEUE
3311 014632 000207              RTS PC ;RETURN
3312
3313          .SBTTL WRITE WHOLE PACK WITH RANDOM DATA
3314
3315 014634 032765 004000 000036 55$: BIT #WRL,P.DS(R5) ;CHECK IF WRITE LOCKED
3316 014642 001341              BNE 50$ ;YES, DROP DRIVE
3317 014644 005037 177776          CLR PS ;ALLOW RK06 INTERRUPTS
3318 014650 116565 000123 000135 MOVN P.RCMD(R5),P.LCMD(R5) ;STORE LAST COMMAND
3319 014656 016565 000124 000136 MOV P.RCYL(R5),P.LCYL(R5) ;STORE LAST CYLINDER
3320 014664 016565 000126 000140 MOV P.RSEC(R5),P.LSEC(R5) ;STORE LAST TRACK AND SECTOR
3321 014672 016565 000130 000142 MOV P.RWC(R5),P.LWC(R5) ;STORE LAST WORD COUNT
3322 014700 112765 000131 000001 MOVN #WRTCHK,P.CMD(R5) ;LOAD WRITE COMMAND
3323 014706 112765 000131 000123 MOVN #WRTCHK,P.RCMD(R5)
3324 014714 052765 000200 000014 BIS #W.WCK,P.PRST(R5) ;SET WRITE BEFORE WRITE CHECK
3325 014722 005065 000002          CLR P.CYL(R5) ;CLEAR CYLINDER ADDRESS
3326 014726 005065 000124          CLR P.RCYL(R5)
3327 014732 005065 000004          CLR P.SECT(R5) ;CLEAR TRACK AND SECTOR
3328 014736 005065 000126          CLR P.RSEC(R5)
3329 014742 105065 000121          CLRN P.DPAT(R5) ;LOAD RANDOM PATTERN
3330 014746 013765 001354 000012 MOV MAXBUF,P.WC(R5) ;LOAD WORD COUNT
3331 014754 042765 000377 000012 BIT #377,P.WC(R5) ;MAKE IT AN EVEN NUMBER OF SECTORS
3332 014762 001003              BWE 57$ ;IF NOT ZERO USE THIS AS WORD COUNT
3333 014764 012765 000400 000012 MOV #256.,P.WC(R5) ;TRANSFER AT LEAST 256 WORDS
3334 014772 062765 000002 000152 57$: ADD #2,P.NORD(R5) ;INCREMENT NUMBER OF ORDERS
3335 015000 066565 000012 000156 ADD P.WC(R5),P.NWRT(R5) ;ADD NUMBER OF WORDS WRITTEN
3336 015006 005565 000160          ADC P.NWRT+2(R5)
3337 015012 066565 000012 000164 ADD P.WC(R5),P.NRD(R5) ;ADD NUMBER OF WORDS READ
3338 015020 005565 000166          ADC P.NRD+2(R5)
3339 015024 005465 000012          NEG P.WC(R5) ;MAKE THE WORD COUNT NEGATIVE
3340 015030 013737 001450 177776 MOV RKPRI,PS ;LOCK RK06 INTERRUPTS
3341 015036 005737 001320          TST BWAITQ ;CHECK IF COMMANDS WAITING FOR BUFFER
3342 015042 001011              BNE 59$ ;YES, ENQUEUE COMMAND IN BUFFER WAIT QUEUE
3343 015044 004037 022114          JSR RO,PUFFAL ;GO ALLOCATE PUFFAL
3344 015050 015066              59$ ;NOT ENOUGH ROOM, ENQUEUE IN BUFFER WAIT QUEUE
3345 015052 005037 177776          CLR PS ;ALLOW RK06 INTERRUPTS
3346 015056 004737 022532          JSR PC,LDDATA ;LOAD BUFFER
3347 015062 000137 013530          JMP ASN3$ ;GO ISSUE COMMAND
3348
3349 015066 004037 050706          59$: JSR RO,Q.PUSH ;PUT PARAMETER BLOCK IN BUFFER
3350 015072 001320              BWAITQ ; WAIT QUEUE
3351 015074 000207          60$: RTS PC ;RETURN
```

```

3352
3353 015076 132765 000020 000210 WRTVER: BITB #BIT4,P.ASSN(R5) ;CHECK IF WRITE PACK IN PROGRESS
3354 015104 001001 BNE 15 ;YES, CONTINUE
3355 015106 000000 HALT ;*** PROGRAM PROBLEM
3356 015110 004037 024506 15: JSR R0,CHKADD ;CHECK SECTOR, TRACK, CYLINDER,
; WORD COUNT, AND BUS ADDRESS
3357 ;PRPR RETURN ADDRESS
3358 015114 015754 WV9$
3359 015116 032765 000200 000014 BIT #W.WCK,P.PRST(R5) ;CHECK IF WRITE CHECK ISSUED
3360 015124 001405 BEQ WV0$ ;YES, CHECK FOR DROP DRIVE
3361 015126 042765 000200 000014 BIT #W.WCK,P.PRST(R5) ;CLEAR ISSUE WRITE BEFORE WRITE CHECK
3362 015134 000137 013530 JMP ASN3$ ;GO ISSUE WRITE CHECK
3363
3364 015140 032765 002000 000014 WV0$: BIT #DRPDV,P.PRST(R5) ;CHECK IF DROP DRIVE
3365 015146 001402 BEQ 10$ ;NO, CONTINUE
3366 015150 000137 015616 JMP 30$ ;YES, TERMINATE PACK WRITING
3367
3368 015154 132765 000040 000210 10$: BITB #BIT5,P.ASSN(P5) ;CHECK IF END OF PACK
3369 015162 001402 BEQ 15$ ;NO, CONTINUE
3370 015164 000137 015640 JMP WV2$ ;YES, RETURN BUFFER
3371
3372 015170 112737 177777 001502 15$: MOVB #-1,I.ISRL ;INDICATE THAT AN INTERRUPT HAS BEEN ISSUED
3373 015176 012762 000300 000000 MOV #INTR,RKCS1(R2) ;GC SCAN FOR DRIVE INTERRUPTS
3374 015204 005037 177776 CLR PS ;ALLOW RK06 INTERRUPTS
3375 015210 116565 000123 000135 MOVB P.RCMD(R5),P.LCMD(R5) ;STORE PREVIOUS COMMAND
3376 015216 016565 000124 000136 MOV P.RCYL(R5),P.LCYL(R5) ;STORE PREVIOUS CYLINDER
3377 015224 016565 000126 000140 MOV P.RSEC(R5),P.LSEC(R5) ;STORE PREVIOUS TRACK AND SECTOR
3378 015232 116565 000122 000144 MOVB P.RDPT(R5),P.LDPT(R5) ;STORE PREVIOUS DATA PATTERN
3379 015240 016565 000130 000142 MOV P.RWC(R5),P.LWC(R5) ;STORE PREVIOUS WORD COUNT
3380 015246 016565 000026 000004 MOV P.DTS(R5),P.SECT(R5) ;LOAD NEW SECTOR AND TRACK
3381 015254 016565 000026 000126 MOV P.DTS(R5),P.RSEC(P5)
3382 015262 016565 000030 000002 MOV P.DCYL(R5),P.CYLN(R5) ;LOAD NEW CYLINDER
3383 015270 016565 000030 000124 MOV P.DCYL(R5),P.RCYL(R5)
3384 015276 016565 000132 000010 MOV P.RPAL(R5),P.PALC(R5) ;REINITIALIZE BUS ADDRESS FOR POSSIBLE
; BAD SECTOR RECOVERY ON LAST COMMAND
3385
3386 015304 052765 000200 000014 BIT #W.WCK,P.PRST(R5) ;SET ISSUE WRITE BEFORE WRITE CHECK
3387 015312 016546 000002 MOV P.CYLN(R5),-(SP) ;STORE CYLINDER ADDRESS ON STACK
3388 015316 006316 ASL (SP) ;MULTIPLY CYLINDER BY 3
3389 015320 066516 000002 ADD P.CYLN(R5),(SP)
3390 015324 005046 CLR -(SP) ;MAKE ROOM ON STACK FOR TRACK
3391 015326 116516 000005 MOVB P.TPCF(R5),(SP) ;STORE TRACK ON STACK
3392 015332 061666 000007 ADD (SP),2(SP) ;ADD TRACK
3393 015336 012716 000026 MOV #22,(SP) ;STORE 22 ON STACK (MULTIPLIER)
3394 015342 004737 054064 JSR PC,MULT ;CALCULATE NUMBER OF SECTORS TRANSFERRED
3395 015346 012616 MOV (SP)+,(SP) ;THROW AWAY MOST SIGNIFICANT BITS
3396 015350 005046 CLR -(SP) ;CLEAR LOCATION FOR SECTOR
3397 015352 116516 000004 MOVB P.SECT(R5),(SP) ;STORE SECTOR COUNT
3398 015356 061666 000002 ADD (SP),2(SP) ;DETERMINE TOTAL NUMBER OF SECTORS TRANSFERRED
3399 015362 012716 064740 MOV #<410.*22.*2>+<27.*2>,(SP) ;STORE NUMBER OF SECTORS ON DISK
3400 015366 166616 000002 SUB 2(SP),(SP) ;DETERMINE NUMBER OF SECTORS LEFT TO
; BE WRITTEN
3401
3402 015372 016546 000130 MOV P.WWC(R5),-(SP) ;STORE WORD COUNT
3403 015376 011665 000012 MOV (SP),P.WC(R5)
3404 015402 005416 NEG (SP) ;MAKE IT POSITIVE
3405 015404 116616 000001 MOVB 1(SP),(SP) ;KEEP SECTOR COUNT
3406 015410 105066 000001 CLWR 1(SP) ;CLEAR MOST SIGNIFICANT BITS
3407 015414 021666 000007 CMP (SP),7(SP) ;CHECK IF PACK CVERRUN

```

3408	015420	103413			BLC	20\$		;MC, GC ISSUE COMMAND
3409	015422	001407			BEG	19\$		;CHECK IF MC NEW WORD COUNT NEEDED
3410	015424	005016			CLR	(SP)		;MAKE ROOM FOR NEW WORD COUNT
3411	015426	116666	000007	000001	MOVR	2(SP),1(SP)		;MOVE IN NUMBER OF SECTORS LEFT
3412	015434	005416			NFC	(SP)		;MAKE WORD COUNT NEGATIVE
3413	015436	011665	000012		MOV	(SP),P.WC(R5)		;LOAD NEW WORD COUNT
3414	015442	152765	000040	000210	BISF	#R15,P.ASSN(R5)		;SET END OF PACK TRANSFER
3415								
3416	015450	062706	000006		20\$: ADD	#6,SP		;ADJUST STACK
3417	015454	016546	000012		MOV	P.WC(R5),-(SP)		;STORE PRESENT WORD COUNT
3418	015460	005416			NFC	(SP)		;MAKE IT POSITIVE
3419	015462	062765	000002	000157	ADD	#7,P.NODR(R5)		;INCREMENT NUMBER OF ORDERS
3420	015470	061665	000156		ADD	(SP),P.NWRT(R5)		;ADD NUMBER OF WORDS WRITTEN
3421	015474	005565	000160		ADC	P.NWRT+2(P5)		
3422	015500	062665	000164		ADD	(SP)+,P.NRD(R5)		;ADD NUMBER OF WORDS READ
3423	015504	005565	000166		ADC	P.NRD+2(R5)		
3424	015510	105265	000122		INCR	P.RDPT(R5)		;GENERATE NEXT DATA PATTERN TO BE WRITTEN
3425	015514	142765	000360	000122	BICB	#360,P.RDPT(R5)		;KEEP 4 LEAST SIGNIFICANT BITS
3426	015522	116565	000122	000121	MOVR	P.RDPT(R5),P.DPAT(R5)		;STORE DATA PATTERN USED
3427	015530	013737	001450	177776	MOV	RKPP1,PS		;LOCK OUT RK66 INTERRUPTS
3428	015536	132765	000040	000210	BITF	#R15,P.ASSN(R5)		;CHECK IF WHITE TO END OF PACK
3429	015544	001014			BNE	25\$		;YES, RELEASE BUFFER
3430	015546	005737	001320		TST	BWAIT0		;CHECK IF BUFFER WAIT QUEUE EMPTY
3431	015552	001011			BNE	25\$		;NO, RELEASE BUFFER
3432	015554	016565	000012	000130	MOV	P.WC(R5),P.PWC(R5)		;STORE CURRENT GENERATED WORD COUNT
3433	015562	005037	177776		CLW	PS		;ALLOW RK66 INTERRUPTS
3434	015566	004737	022532		JSR	PC,LDDATA		;LOAD NEW RANDOM DATA
3435	015572	000137	013530		JMP	ASN3\$		;GO ISSUE COMMAND
3436								
3437	015576	004737	022316		25\$: JSR	PC,PUPREL		;RELEASE BUFFER FOR OTHER DRIVES
3438	015602	004037	050706		JSR	RO,Q.PUSH		;PUT PARAMETER PLCK ON BUFFER WAIT QUEUE
3439	015606	001320			BWAIT0			
3440	015610	004737	022016		JSR	PC,GETBUF		;GET NEW BUFFER
3441	015614	000207			RTS	PC		;RETURN
3442								
3443	015616	105065	000210		30\$: CLRB	P.ASSN(R5)		;CLEAR ASSIGNMENT CODE
3444	015622	004737	022316		JSR	PC,PUPREL		;RELEASE BUFFER
3445	015626	104401	055747		WV1\$: TYPE	,OPR08		;TYPE "OPERATOR INITIATED DROP DRIVE"
3446	015632	004737	012716		JSR	PC,DROP		;GC DROP DRIVE
3447	015636	000444			BP	WV5\$		;GO ALLOCATE BUFFER
3448								
3449	015640	105065	000210		WV2\$: CLRB	P.ASSN(R5)		;CLEAR ASSIGNMENT FLAGS
3450	015644	116537	000000	055446	MOVB	P.DRVN(R5),OPR011		;GET DRIVE NUMBER
3451	015652	152737	000060	055446	BISR	#60,OPR011		;MAKE IT ASCII
3452	015660	104401	055427		TYPE	,OPR010		;TYPE "DRIVE NUMBER & UNDER TEST"
3453	015664	104401	055172		TYPE	,OPR003		
3454	015670	004737	027060		JSR	PC,PRISFH		;PRINT DRIVE AND PACK SERIAL NUMBERS
3455	015674	004737	043072		JSR	PC,PRITTF		;PRINT TIME
3456	015700	104401	001165		TYPE	,SCPLF		
3457	015704	010346			MOV	R3,-(SP)		;STORE R3 ON STACK
3458	015706	010446			MOV	R4,-(SP)		;STORE R4 ON STACK
3459	015710	010503			MOV	R5,R3		;LOAD PARAMETER BLOCK BASE
3460	015712	062703	000152		ADD	#P.NODR,R3		;CALCULATE BEGINNING OF STATISTICS
3461	015716	010504			MOV	R5,R4		;LOAD PARAMETER BLOCK BASE
3462	015720	062704	000210		ADD	#P.ASSN,R4		;CALCULATE END OF STATISTICS
3463	015724	005027			7\$: CLR	(R3)+		;CLEAR STATISTICS BEFORE STARTING TEST

3464	015726	020304		CMP	R3,R4	;CHPCW IF FINISHED
3465	015730	001375		BNE	7S	;NO, CONTINUE
3466	015732	012604		MOV	(SP)+,R4	;RESTORE R4
3467	015734	012603		MOV	(SP)+,R3	;RESTORE R3
3468	015736	004737	022316	JSR	PC,HUPRPL	;RELEASE BUFFFF
3469	015742	004037	050706	JSR	RO,O.PUSH	;ENQUEUE PARAMETER BLOCK IN DRIVE
3470	015746	001314		AVAILQ		; AVAILABLE QUEUE
3471	015750	004737	022016	WV5S: JSR	PC,GETBUF	;GET NEW BUFFFF
3472	015754	000207		WV9S: RTS	PC	;RETURN

```
3473 .SOTTL ALTER DRIVE PARAMETERS
3474
3475 015756 010446 PARM: MOV R4,-(SP) ;STORE R4 ON STACK
3476 015760 010346 MOV R3,-(SP) ;STORE R3 ON STACK
3477 015762 116537 000000 056603 POS: MOVB P.DPVN(R5),PAR001 ;LOAD DRIVE NUMBER FOR PRINT OUT
3478 015770 152737 000060 056603 BISR #60,PAR001 ;MAKE IT ASCII
3479 015776 104401 056557 TYPE ,PAR000 ;TYPE "PARAMETERS FOR DRIVE N"
3480 016002 104401 056610 2S: TYPE ,PAR002 ;TYPE "CYLINDER MAX,MIN"
3481 016006 004037 020116 JSR R0,TSTDEF ;CHECK FOR DEFAULT PARAMETERS
3482 016012 016124 5S ;COMMA DETECTED
3483 016014 016140 6S ;CARRIAGE RETURN DETECTED
3484 016016 017756 P1S ;CONTROL Z <"Z"> DETECTED
3485 016020 015762 POS ;CONTROL C <"C"> DETECTED
3486 016022 010346 MOV R7,-(SP) ;STORE BUFFER ADDRESS
3487 016024 004737 053316 JSR PC,OCTBIN ;CONVERT TO BINARY
3488 016030 016106 4S ;ERROR RETURN
3489 016032 012637 001576 MOV (SP)+,PARCYL ;STORE MAX CYLINDER ENTRY
3490 016036 022737 000632 001576 CMP #D.MXCL,PARCYL ;CHECK IF LESS OR EQUAL TO
3491 ; MAXIMUM CYLINDER ADDRESS
3492 016044 103420 BLO 4S ;NO, TRY AGAIN
3493 016046 010304 MOV R3,R4 ;STORE R3 IN R4 FOR LINE SCAN
3494 016050 105714 64S: TSTB (R4) ;CHECK FOR CARRIAGE RETURN
3495 016052 001435 BEQ 7S ;YES, EXIT
3496 016054 122427 000054 CMP# (R4)+,8", ;CHECK FOR COMMA
3497 016060 001373 BNE 64S ;NO, GO TEST NEXT CHARACTER
3498 016062 010446 3S: MOV R4,-(SP) ;STORE BUFFER ADDRESS
3499 016064 004737 053316 JSR PC,OCTBIN ;CONVERT TO BINARY
3500 016070 016106 4S ;ERROR RETURN
3501 016072 012637 001600 MOV (SP)+,PARCYL+2 ;STORE MIN CYLINDER ENTRY
3502 016076 023737 001576 001600 CMP PARCYL,PARCYL+2 ;CHECK IF LEGAL ENTRY
3503 016104 103022 BNIS 8S ;YES, LOAD MAX AND MIN CYLINDER
3504 016106 4S:
3505 016106 010337 016114 MOV R7,65S ;LOAD BUFFER ADDRESS
3506 016112 104401 TYPE ;TYPE RECEIVED INPUT
3507 016114 000000 65S: .WORD 0 ;BUFFER ADDRESS
3508 016116 104401 001164 TYPE ,SQUES ;TYPE QUESTION MARK
3509 016122 000727 BR 2S ;TRY AGAIN
3510
3511 016124 012737 000632 001576 5S: MOV #D.MXCL,PARCYL ;LOAD MAX CYLINDER = 410
3512 016132 010304 MOV R3,R4 ;STORE R3 IN R4 FOR LINE SCAN
3513 016134 105724 TSTB (R4)+ ;ADJUST R4
3514 016136 000751 BR 3S ;GET MINIMUM CYLINDER
3515
3516 016140 012737 000632 001576 6S: MOV #D.MXCL,PARCYL ;LOAD MAX CYLINDER = 410
3517 016146 005037 001600 7S: CLR PARCYL+2 ;LOAD MIN CYLINDER = 0
3518 016152 013765 001576 000070 8S: MOV PARCYL,P.MXCL(R5) ;LOAD MAX CYLINDER
3519 016160 013765 001600 000066 MOV PARCYL+2,P.MNCL(R5) ;LOAD MIN CYLINDER
3520
3521 016166 104401 056633 10S: TYPE ,PAR003 ;TYPE "TRACE MAX,MIN"
3522 016172 004037 020116 JSR R0,TSTDEF ;CHECK FOR DEFAULT PARAMETERS
3523 016176 016310 13S ;COMMA DETECTED
3524 016200 016324 14S ;CARRIAGE RETURN DETECTED
3525 016202 017770 P2S ;CONTROL Z <"Z"> DETECTED
3526 016204 015762 POS ;CONTROL C <"C"> DETECTED
3527 016206 010346 MOV R7,-(SP) ;STORE BUFFER ADDRESS
3528 016210 004737 053316 JSR PC,OCTBIN ;CONVERT TO BINARY
```

Line	Address	Offset	Label	Op	Op2	Op3	Op4	Op5	Op6	Op7	Op8	Op9	Op10	Op11	Op12	Op13	Op14	Op15	Op16	Op17	Op18	Op19	Op20	Op21	Op22	Op23	Op24	Op25	Op26	Op27	Op28	Op29	Op30	Op31	Op32	Op33	Op34	Op35	Op36	Op37	Op38	Op39	Op40	Op41	Op42	Op43	Op44	Op45	Op46	Op47	Op48	Op49	Op50	Op51	Op52	Op53	Op54	Op55	Op56	Op57	Op58	Op59	Op60	Op61	Op62	Op63	Op64	Op65	Op66	Op67	Op68	Op69	Op70	Op71	Op72	Op73	Op74	Op75	Op76	Op77	Op78	Op79	Op80	Op81	Op82	Op83	Op84	Op85	Op86	Op87	Op88	Op89	Op90	Op91	Op92	Op93	Op94	Op95	Op96	Op97	Op98	Op99	Op100	Op101	Op102	Op103	Op104	Op105	Op106	Op107	Op108	Op109	Op110	Op111	Op112	Op113	Op114	Op115	Op116	Op117	Op118	Op119	Op120	Op121	Op122	Op123	Op124	Op125	Op126	Op127	Op128	Op129	Op130	Op131	Op132	Op133	Op134	Op135	Op136	Op137	Op138	Op139	Op140	Op141	Op142	Op143	Op144	Op145	Op146	Op147	Op148	Op149	Op150	Op151	Op152	Op153	Op154	Op155	Op156	Op157	Op158	Op159	Op160	Op161	Op162	Op163	Op164	Op165	Op166	Op167	Op168	Op169	Op170	Op171	Op172	Op173	Op174	Op175	Op176	Op177	Op178	Op179	Op180	Op181	Op182	Op183	Op184	Op185	Op186	Op187	Op188	Op189	Op190	Op191	Op192	Op193	Op194	Op195	Op196	Op197	Op198	Op199	Op200	Op201	Op202	Op203	Op204	Op205	Op206	Op207	Op208	Op209	Op210	Op211	Op212	Op213	Op214	Op215	Op216	Op217	Op218	Op219	Op220	Op221	Op222	Op223	Op224	Op225	Op226	Op227	Op228	Op229	Op230	Op231	Op232	Op233	Op234	Op235	Op236	Op237	Op238	Op239	Op240	Op241	Op242	Op243	Op244	Op245	Op246	Op247	Op248	Op249	Op250	Op251	Op252	Op253	Op254	Op255	Op256	Op257	Op258	Op259	Op260	Op261	Op262	Op263	Op264	Op265	Op266	Op267	Op268	Op269	Op270	Op271	Op272	Op273	Op274	Op275	Op276	Op277	Op278	Op279	Op280	Op281	Op282	Op283	Op284	Op285	Op286	Op287	Op288	Op289	Op290	Op291	Op292	Op293	Op294	Op295	Op296	Op297	Op298	Op299	Op300	Op301	Op302	Op303	Op304	Op305	Op306	Op307	Op308	Op309	Op310	Op311	Op312	Op313	Op314	Op315	Op316	Op317	Op318	Op319	Op320	Op321	Op322	Op323	Op324	Op325	Op326	Op327	Op328	Op329	Op330	Op331	Op332	Op333	Op334	Op335	Op336	Op337	Op338	Op339	Op340	Op341	Op342	Op343	Op344	Op345	Op346	Op347	Op348	Op349	Op350	Op351	Op352	Op353	Op354	Op355	Op356	Op357	Op358	Op359	Op360	Op361	Op362	Op363	Op364	Op365	Op366	Op367	Op368	Op369	Op370	Op371	Op372	Op373	Op374	Op375	Op376	Op377	Op378	Op379	Op380	Op381	Op382	Op383	Op384	Op385	Op386	Op387	Op388	Op389	Op390	Op391	Op392	Op393	Op394	Op395	Op396	Op397	Op398	Op399	Op400	Op401	Op402	Op403	Op404	Op405	Op406	Op407	Op408	Op409	Op410	Op411	Op412	Op413	Op414	Op415	Op416	Op417	Op418	Op419	Op420	Op421	Op422	Op423	Op424	Op425	Op426	Op427	Op428	Op429	Op430	Op431	Op432	Op433	Op434	Op435	Op436	Op437	Op438	Op439	Op440	Op441	Op442	Op443	Op444	Op445	Op446	Op447	Op448	Op449	Op450	Op451	Op452	Op453	Op454	Op455	Op456	Op457	Op458	Op459	Op460	Op461	Op462	Op463	
------	---------	--------	-------	----	-----	-----	-----	-----	-----	-----	-----	-----	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	--

3585	016454	103022				BRIS	24\$	;YES, LOAD MAX AND MIN SECTOR
3586	016456				20\$:			
3587	016456	010337	016464			MOV	R3,69\$	;LOAD BUFFER ADDRESS
3588	016462	104401				TYPE		;TYPE RECEIVED INPUT
3589	016464	000000			69\$:	.WCRD	0	;BUFFER ADDRESS
3590	016466	104401	001164			TYPE	,SQUES	;TYPE QUESTION MARK
3591	016472	000727				BR	1R\$	;TRY AGAIN
3592								
3593	016474	012737	000025	001606	21\$:	MOV	#D.MXSC,PARSEC	;LOAD MAX SECTOR = 21
3594	016502	010304				MOV	R3,R4	;STORE R3 IN R4 FOR LINE SCAN
3595	016504	105724				TSTB	(R4)+	;ADJUST R4
3596	016506	000751				BR	1R\$	;GET MINIMUM SECTOR
3597								
3598	016510	012737	000025	001606	22\$:	MOV	#D.MXSC,PARSEC	;LOAD MAX SECTOR = 21
3599	016516	005037	001610		23\$:	CLR	PARSEC+2	;LOAD MIN SECTOR = 0
3600	016522	113765	001606	000075	24\$:	MOVB	PARSEC,P.MXSC(R5)	;LOAD MAX SECTOR
3601	016530	113765	001610	000074		MOVB	PARSEC+2,P.MXSC(R5)	;LOAD MIN SECTOR
3602								
3603	016536	104401	056674		26\$:	TYPE	,PAR005	;TYPE "READ/WRITE RATIO ="
3604	016542	004037	020116			JSR	RO,TSTDEF	;CHECK FOR DEFAULT PARAMETERS
3605	016546	016576				27\$		;CCMA DETECTED
3606	016550	016622				29\$		;CARRIAGE RETURN DETECTED
3607	016552	020014				P4\$		;CONTROL Z <"Z"> DETECTED
3608	016554	015762				P0\$		;CONTROL C <"C"> DETECTED
3609	016556	010346				MOV	R3,-(SP)	;STORE BUFFER ADDRESS
3610	016560	004737	053316			JSR	PC,OCTBIN	;CONVERT BINARY
3611	016564	016576				27\$		;FRRCR RETURN
3612	016566	012504				MOV	(SP)+,R4	;STORE INPUT
3613	016570	022704	000010			CMP	#10,R4	;CHECK IF 0-7
3614	016574	101007				BHI	28\$	;YES, LOAD READ/WRITE RATIO
3615	016576				27\$:			
3616	016576	010337	016604			MOV	R3,70\$	;LOAD BUFFER ADDRESS
3617	016602	104401				TYPE		;TYPE RECEIVED INPUT
3618	016604	000000			70\$:	.WCRD	0	;BUFFER ADDRESS
3619	016606	104401	001164			TYPE	,SQUES	;TYPE QUESTION MARK
3620	016612	000751				BR	26\$	;TRY AGAIN
3621								
3622	016614	110465	000076		28\$:	MOVB	R4,P.RATE(R5)	;LOAD READ/WRITE RATIO
3623	016620	000403				BR	30\$	;ASK "WRITE CHECK AFTER WRITE ="
3624								
3625	016622	112765	000003	000076	29\$:	MOVB	#D.RATE,P.RATE(R5)	;LOAD DEFAULT READ/WRITE
3626								
3627	016630	104401	056717		30\$:	TYPE	,PAR006	;TYPE "WRITE CHECK AFTER WRITE ="
3628	016634	004037	020116			JSR	RO,TSTDEF	;CHECK FOR DEFAULT PARAMETERS
3629	016640	016666				31\$		;CCMA DETECTED
3630	016642	016704				32\$		;CARRIAGE RETURN DETECTED
3631	016644	020022				P5\$		;CONTROL Z <"Z"> DETECTED
3632	016646	015762				P0\$		;CONTROL C <"C"> DETECTED
3633	016650	122713	000131			CMP	#Y,(R3)	;CHECK IF YES
3634	016654	001013				BNE	32\$	;NO, LOAD DEFAULT
3635	016656	112765	177777	000077		MOVB	#-1,P.AMCK(R5)	;SET WRITE CHECK AFTER WRITE
3636	016664	000411				BR	34\$	;GET CORRECTABLE READ ERROR
3637								; THRESHOLD
3638								
3639	016666				31\$:			
3640	016666	010337	016674			MOV	R3,71\$	;LOAD BUFFER ADDRESS

3641	016672	104401				TYPE		;TYPE RECEIVED INPUT
3642	016674	000000		71\$:		.WORD	0	;BUFFER ADDRESS
3643	016676	104401	001164			TYPE	,SQUES	;TYPE QUESTION MARK
3644	016702	000752				BR	30\$	;TRY AGAIN
3645								
3646	016704	105065	000077	32\$:		CLRB	P.AWCK(R5)	;CLEAR WRITE CHECK AFTER WRITE
3647								
3648	016710	104401	056751	34\$:		TYPE	,PAR007	;TYPE *CORRECTABLE READ ERROR
3649								; THRESHOLD
3650	016714	004037	020116			JSR	R0,TSTDEF	;CHECK FOR DEFAULT PARAMETERS
3651	016720	016746				35\$		;COMMA DETECTED
3652	016722	016764				36\$		;CARRIAGE RETURN DETECTED
3653	016724	020026				P6\$		;CONTROL Z <^Z> DETECTED
3654	016726	015762				P0\$		;CONTROL C <^C> DETECTED
3655	016730	010346				MOV	R3,-(SP)	;STORE BUFFER ADDRESS
3656	016732	004737	053452			JSR	PC,DECBIN	;CONVERT TO BINARY
3657	016736	016746				35\$		;FARCR RETURN
3658	016740	012665	000100			MOV	(SP)+,P.CERT(R5)	;LOAD CORRECTABLE READ ERROR
3659								; THRESHOLD
3660	016744	100012				BPL	38\$	;IF NOT NEGATIVE, GET UNCORRECTABLE
3661								; READ ERROR THRESHOLD
3662								
3663	016746			35\$:				
3664	016746	010337	016754			MOV	R3,72\$	;LOAD BUFFER ADDRESS
3665	016752	104401				TYPE		;TYPE RECEIVED INPUT
3666	016754	000000		72\$:		.WORD	0	;BUFFER ADDRESS
3667	016756	104401	001164			TYPE	,SQUES	;TYPE QUESTION MARK
3668	016762	000752				BR	34\$	;TRY AGAIN
3669								
3670	016764	012765	000012 000100	36\$:		MOV	#D.CERT,P.CERT(R5)	;LOAD DEFAULT CORRECTABLE READ
3671								; ERROR THRESHOLD
3672								
3673	016772	104401	057014	38\$:		TYPE	,PAR008	;TYPE *UNCORRECTABLE READ ERROR
3674								; THRESHOLD =
3675	016776	004037	020116			JSR	R0,TSTDEF	;CHECK FOR DEFAULT PARAMETERS
3676	017002	017030				39\$		;COMMA DETECTED
3677	017004	017046				40\$		;CARRIAGE RETURN DETECTED
3678	017006	020034				P7\$		;CONTROL Z <^Z> DETECTED
3679	017010	015762				P0\$		;CONTROL C <^C> DETECTED
3680	017012	010346				MOV	R3,-(SP)	;STORE BUFFER ADDRESS
3681	017014	004737	053452			JSR	PC,DECBIN	;CONVERT TO BINARY
3682	017020	017030				39\$		;FARCR RETURN
3683	017022	012665	000102			MOV	(SP)+,P.UERT(R5)	;LOAD UNCORRECTABLE READ ERROR
3684								; THRESHOLD
3685	017026	100012				BPL	42\$	;IF NOT NEGATIVE, GET SEEK
3686								; ERROR THRESHOLD
3687	017030			39\$:				
3688	017030	010337	017036			MOV	R3,73\$	;LOAD BUFFER ADDRESS
3689	017034	104401				TYPE		;TYPE RECEIVED INPUT
3690	017036	000000		73\$:		.WORD	0	;BUFFER ADDRESS
3691	017040	104401	001164			TYPE	,SQUES	;TYPE QUESTION MARK
3692	017044	000752				BR	38\$	;TRY AGAIN
3693								
3694	017046	012765	000005 000102	40\$:		MOV	#D.UERT,P.UERT(R5)	;LOAD DEFAULT UNCORRECTABLE READ
3695								; ERROR THRESHOLD
3696								

3697	017054	104401	057061	42\$:	TYPE	,PAR009	;TYPE "SEEK ERROR THRESHOLD ="
3698	017060	004037	020116		JSR	RO,TSTDEF	;CHECK FOR DEFAULT PARAMETERS
3699	017064	017112			43\$		;COMMA DETECTED
3700	017066	017130			44\$		;CARRIAGE RETURN DETECTED
3701	017070	020042			P8\$		;CONTROL Z <"Z"> DETECTED
3702	017072	015762			P0\$		;CONTROL C <"C"> DETECTED
3703	017074	010346			MOV	R3,-(SP)	;STORE BUFFER ADDRESS
3704	017076	004737	053452		JSR	PC,DECBIN	;CONVERT TO BINARY
3705	017102	017112			43\$		;ERROR RETURN
3706	017104	012665	000104		MOV	(SP)+,P.SKET(R5)	;LOAD SEEK ERROR THRESHOLD
3707	017110	100012			BPL	46\$	;CHECK IF INHIBIT ERROR CORRECTION
3708							; AND RETRY
3709	017112			43\$:			
3710	017112	010337	017120		MOV	R3,74\$	;LOAD BUFFER ADDRESS
3711	017116	104401			TYPE		;TYPE RECEIVED INPUT
3712	017120	000000		74\$:	.WOPD	0	;BUFFER ADDRESS
3713	017122	104401	001164		TYPE	,SQUES	;TYPE QUESTION MARK
3714	017126	000752			BR	42\$	;TRY AGAIN
3715							
3716	017130	012765	000005 000104	44\$:	MOV	#D.SKET,P.SKET(R5)	;LOAD DEFAULT SEEK ERROR THRESHOLD
3717							
3718	017136	104401	057110	46\$:	TYPE	,PAR010	;TYPE "INHIBIT ERROR CORRECTION
3719							; AND RETRY
3720	017142	004037	020116		JSR	RO,TSTDEF	;CHECK FOR DEFAULT PARAMETERS
3721	017146	017174			47\$		;COMMA DETECTED
3722	017150	017212			48\$		;CARRIAGE RETURN DETECTED
3723	017152	020050			P8\$		;CONTROL Z <"Z"> DETECTED
3724	017154	015762			P0\$		;CONTROL C <"C"> DETECTED
3725	017156	122713	000131		CMPR	#Y,(R3)	;CHECK IF YFS
3726	017162	001013			BNE	48\$	;LOAD DEFAULT
3727	017164	112765	177777 000120		MOVB	#-1,P.IFRC(R5)	;SET INHIBIT ERROR CORRECTION
3728	017172	000411			BR	50\$	;GET COMMAND COUNT THRESHOLD
3729							
3730	017174			47\$:			
3731	017174	010337	017202		MOV	R3,75\$	;LOAD BUFFER ADDRESS
3732	017200	104401			TYPE		;TYPE RECEIVED INPUT
3733	017202	000000		75\$:	.WOPD	0	;BUFFER ADDRESS
3734	017204	104401	001164		TYPE	,SQUES	;TYPE QUESTION MARK
3735	017210	000752			BR	46\$	;TRY AGAIN
3736							
3737	017212	105065	000120	48\$:	CLRP	P.IFRC(R5)	;CLEAR INHIBIT ERROR CORRECTION
3738							
3739	017216	104401	057155	50\$:	TYPE	,PAR011	;TYPE "OPERATION COUNT THRESHOLD *65K ="
3740	017222	004037	020116		JSR	RO,TSTDEF	;CHECK FOR DEFAULT PARAMETERS
3741	017226	017272			51\$		;COMMA DETECTED
3742	017230	017310			52\$		;CARRIAGE RETURN DETECTED
3743	017232	020054			P10\$		;CONTROL Z <"Z"> DETECTED
3744	017234	015762			P0\$		;CONTROL C <"C"> DETECTED
3745	017236	010346			MOV	R3,-(SP)	;STORE BUFFER ADDRESS
3746	017240	004737	053452		JSR	PC,DECBIN	;CONVERT TO BINARY
3747	017244	017277			51\$		;ERROR RETURN
3748	017246	012665	000110		MOV	(SP)+,P.MXCD+2(R5)	;STORE NUMBER OF 65K BLOCKS
3749	017252	001407			BEQ	51\$	;IF ZFRO, TRY AGAIN
3750	017254	100406			BMI	51\$	;IF MINUS, TRY AGAIN
3751	017256	005365	000110		DFC	P.MXCD+2(R5)	;DECREMENT BY 1
3752	017262	012765	177777 000106		MOV	#-1,P.MXCD(R5)	

3753	017270	000415		BR	54\$		;GET WORD TRANSFERREC THRESHOLD
3754							
3755	017272			51\$:			
3756	017272	010337	017300	MOV	R3,76\$		;LOAD BUFFER ADDRESS
3757	017276	104401		TYPE			;TYPE RECEIVED INPUT
3758	017300	000000		76\$:	.WORD	0	;BUFFER ADDRESS
3759	017302	104401	001164	TYPE	,SQUES		;TYPE QUESTION MARK
3760	017306	000743		BR	50\$		;TRY AGAIN
3761							
3762	017310	012765	077777	52\$:	MOV	#D.CTWH,P.MXCD+2(R5)	;LOAD DEFAULT COMMAND THRESHOLD
3763	017316	012765	177770		MOV	#D.CTPL,P.MXCD(R5)	
3764							
3765	017324	104401	057215	54\$:	TYPE	,PAR012	;TYPE "WORD TRANSFERRED THRESHOLD" 520K ="
3766	017330	004037	020116		JSR	RO,TSTDEF	;CHECK FOR DEFAULT PARAMETERS
3767	017334	017432			55\$		;COMMA DETECTED
3768	017336	017450			56\$		;CARRIAGE RETURN DETECTED
3769	017340	020070			P11\$		;CONTROL Z <"Z"> DETECTED
3770	017342	015762			POS		;CONTROL C <"C"> DETECTED
3771	017344	010346			MOV	R3,-(SP)	;STORE BUFFER ADDRESS
3772	017346	004737	053452		JSR	PC,DECBIN	;CONVERT TO BINARY
3773	017352	017432			55\$		;FRRCH RETURN
3774	017354	012665	000112		MOV	(SP)+,P.MXWT(R5)	;STORE NUMBER OF 520K BLOCKS
3775	017360	001424			55\$		;IF ZPRO, TRY AGAIN
3776	017362	100423			55\$		;IF NEGATIVE, TRY AGAIN
3777	017364	005065	000114		CLR	P.MXWT+2(R5)	;CLEAR HIGH ORDER BITS
3778	017370	006165	000112		ROL	P.MXWT(R5)	;SHIFT 4 BITS LEFT
3779	017374	006165	000114		ROL	P.MXWT+7(R5)	
3780	017400	006165	000112		ROL	P.MXWT(R5)	
3781	017404	006165	000114		ROL	P.MXWT+2(R5)	
3782	017410	006165	000112		ROL	P.MXWT(R5)	
3783	017414	006165	000114		ROL	P.MXWT+7(R5)	
3784	017420	006165	000112		ROL	P.MXWT(R5)	
3785	017424	006165	000114		ROL	P.MXWT+2(R5)	
3786	017430	000415			BP	60\$	;CHECK FOR SAMPLED COMPARES
3787							
3788	017432			55\$:			
3789	017432	010337	017440		MOV	R3,77\$	;LOAD BUFFER ADDRESS
3790	017436	104401			TYPE		;TYPE RECEIVED INPUT
3791	017440	000000		77\$:	.WORD	0	;BUFFER ADDRESS
3792	017442	104401	001164		TYPE	,SQUES	;TYPE QUESTION MARK
3793	017446	000726			BR	54\$	;TRY AGAIN
3794							
3795	017450	012765	077777	56\$:	MOV	#D.WTHI,P.MXWT+7(R5)	;LOAD DEFAULT WORD TRANSFER
3796	017456	012765	177770		MOV	#D.WTLO,P.MXWT(R5)	; THRESHOLD
3797							
3798	017464	005737	001362	60\$:	TST	SOFCMP	;CHECK IF ANY SOFTWARE COMPARES ARE TO BE MADE
3799	017470	001430			BEQ	PREXAP	;NO, DETERMINE TRACK EXCLUSION
3800	017472	104401	057513		TYPE	,PAR021	;TYPE "SAMPLED COMPARES ="
3801	017476	004037	020116		JSR	RO,TSTDEF	;CHECK FOR DEFAULT PARAMETERS
3802	017502	017530			61\$		;COMMA DETECTED
3803	017504	017546			62\$		;CARRIAGE RETURN DETECTED
3804	017506	020104			P12\$		;CONTROL Z <"Z"> DETECTED
3805	017510	015762			POS		;CONTROL C <"C"> DETECTED
3806	017512	122713	000116		CMPR	#N,(R3)	;CHECK IF NO
3807	017516	001013			BNE	62\$	;NO, MAKE SAMPLED COMPARES
3808	017520	112765	177777	000116	MOVR	#-1,P.SMPL(R5)	;DO NOT MAKE SAMPLED COMPARES

3809	017526	000411		BR	PREXAR	);DETERMINE TRACK EXCLUSION
3810						
3811	017530			61\$:		
3812	017530	010337	017536	MOV	R3,78\$	);LOAD BUFFER ADDRESS
3813	017534	104401		TYPE		);TYPE RECEIVED INPUT
3814	017536	000000		78\$:	.WORD 0	);BUFFER ADDRESS
3815	017540	104401	001164	TYPE	,SQUES	);TYPE QUESTION MARK
3816	017544	000747		BR	60\$	);TRY AGAIN
3817						
3818	017546	105065	000116	62\$:	CLRM P.SMPL(R5)	);WAVE SAMPLED COMPARES
3819						
3820	017552	104401	057350	PREXAR:	TYPE ,PAR019	);TYPE "CHANGE EXCLUDED PACK AREA ="
3821	017556	104401	057322	TYPE	,PAR018	
3822	017562	004037	020116	JSR	RO,TSTDEF	);CHECK FOR DEFAULT PARAMETERS
3823	017566	017610		IS		);COMMA DETECTED
3824	017570	017750		50\$		);CAPRIAGE RETURN DETECTED
3825	017572	017750		50\$		);CONTROL Z (^Z) DETECTED
3826	017574	015762		POS		);CONTROL C (^C) DETECTED
3827	017576	122713	000131	CMPB	#Y,(R3)	);CHECK IF YES
3828	017602	001411		BEQ	2\$	);YES, LOAD NEW TRACK EXCLUSIONS
3829	017604	000137	017750	JMP	50\$	);NO, RETURN
3830						
3831	017610			1\$:		
3832	017610	010337	017616	MOV	R3,64\$	);LOAD BUFFER ADDRESS
3833	017614	104401		TYPE		);TYPE RECEIVED INPUT
3834	017616	000000		64\$:	.WORD 0	);BUFFER ADDRESS
3835	017620	104401	001164	TYPE	,SQUES	);TYPE QUESTION MARK
3836	017624	000752		BR	PREXAR	);TRY AGAIN
3837						
3838	017626	010146		2\$:	MOV R1, -(SP)	);STORE R1 ON STACK
3839	017630	010501		MOV	R5, #1	);STORE PARAMETER BLOCK ADDRESS
3840	017632	062701	000232	ADD	#P.EXAR, R1	);CALCULATE EXCLUSION AREA
3841	017636	012704	000012	MOV	#10., P4	);LOAD COUNT
3842	017642	005021		5\$:	CLR (R1)+	);CLEAR EXCLUSION AREA
3843	017644	005304		DEC	R4	);DECREMENT COUNT
3844	017646	001375		BNE	5\$	);CHECK IF DONE
3845	017650	162701	000024	SUB	#20., R1	);REINITIALIZE PCOUNTER
3846	017654	004037	020166	JSR	RO, LDEXAR	);GET FIRST EXCLUDED AREA
3847	017660	057262		PAR013		
3848	017662	017746		45\$		);DEFAULT RETURN
3849	017664	017740		40\$		);CONTROL C (^C) RETURN
3850	017666	004037	020166	JSR	RO, LDEXAR	);GET SECOND EXCLUDED AREA
3851	017672	057270		PAR014		
3852	017674	017746		45\$		);DEFAULT RETURN
3853	017676	017740		40\$		);CONTROL C (^C) RETURN
3854	017700	004037	020166	JSR	RO, LDEXAR	);GET THIRD EXCLUDED AREA
3855	017704	057277		PAR015		
3856	017706	017746		45\$		);DEFAULT RETURN
3857	017710	017740		40\$		);CONTROL C (^C) RETURN
3858	017712	004037	020166	JSR	RO, LDEXAR	);GET FOURTH EXCLUDED AREA
3859	017716	057305		PAR016		
3860	017720	017746		45\$		);DEFAULT RETURN
3861	017722	017740		40\$		);CONTROL C (^C) RETURN
3862	017724	004037	020166	JSR	RO, LDEXAR	);GET FIFTH EXCLUDED AREA
3863	017730	057314		PAR017		
3864	017732	017746		45\$		);DEFAULT RETURN

```

3865 017734 017740          40$          ;CONTROL C <C> RETURN
3866 017736 000403          BP          45$          ;RETURN
3867
3868 017740 012601          40$: MOV      (SP)+,R1      ;RESTORE R1
3869 017742 000137 015762  JMP      POS          ;START AGAIN
3870
3871 017746 012601          45$: MOV      (SP)+,R1      ;RESTORE R1
3872 017750 012603          50$: MOV      (SP)+,R3      ;RESTORE R3
3873 017752 012604          MOV      (SP)+,R4      ;RESTORE R4
3874 017754 000207          RTS      PC          ;RETURN
3875
3876 017756 012765 000632 000070 P1$: MOV      #D.MXCL,P.MXCL(R5) ;LOAD DEFAULT MAXIMUM CYLINDER = 410
3877 017764 005065 000066          CLR      P.MNCL(R5)      ;LOAD DEFAULT MINIMUM CYLINDER
3878 017770 112765 000002 000073 P2$: MOV      #D.MXTR,P.MXTR(R5) ;LOAD DEFAULT MAXIMUM TRACK = 7
3879 017776 105065 000072          CLR      P.MNTR(R5)      ;LOAD DEFAULT MINIMUM TRACK
3880 020002 112765 000025 000075 P3$: MOV      #D.MXSC,P.MXSC(R5) ;LOAD DEFAULT MAXIMUM SECTOR = 21
3881 020010 105065 000074          CLR      P.MNSC(R5)      ;LOAD DEFAULT MINIMUM SECTOR
3882 020014 112765 000003 000076 P4$: MOV      #D.RATE,P.RATE(R5) ;LOAD READ/WRITE RATIO
3883 020022 105065 000077          CLR      P.AWCK(R5)      ;LOAD WRITE CHECK AFTER WRITE
3884 020026 012765 000012 000100 P6$: MOV      #D.CERT,P.CERT(R5) ;LOAD CORRECTABLE READ ERROR THRESHOLD
3885 020034 012765 000005 000102 P7$: MOV      #D.UERT,P.UERT(R5) ;LOAD UNCORRECTABLE READ ERROR THRESHOLD
3886 020042 012765 000005 000104 P8$: MOV      #D.SKET,P.SKET(R5) ;LOAD SEEK ERROR THRESHOLD
3887 020050 105065 000120          CLR      P.IERC(R5)      ;CLEAR INHIBIT ERROR CORRECTION
3888 020054 012765 077777 000110 P10$: MOV     #D.CTRH,P.MXCD+2(R5) ;LOAD COMMAND THRESHOLD
3889 020062 012765 177770 000106          MOV      #D.CTRL,P.MXCD(R5)
3890 020070 012765 077777 000114 P11$: MOV     #D.WTHI,P.MXWT+2(R5) ;LOAD WORD TRANSFERED THRESHOLD
3891 020076 012765 177770 000112          MOV      #D.WTLO,P.MXWT(R5)
3892 020104 105065 000116          P12$: CLR      P.SMPL(R5)      ;WAKE SAMPLED COMPARES
3893
3894 020110 012603          MOV      (SP)+,R3      ;RESTORE R3
3895 020112 012604          MOV      (SP)+,R4      ;RESTORE R4
3896 070114 000207          RTS      PC          ;RETURN

```

```

3997 .SBTTL TEST DEFAULT PARAMETERS ROUTINE
3998 ;*****
3999 ;*
3900 ;* THIS ROUTINE WILL READ A LINE OF ASCII TEXT AND CHECK
3901 ;* IF PARAMETER OR TEST OF PARAMETERS WILL BE DEFAULTED.
3902 ;* IT RETURNS WITH R3 POINTING TO THE ADDRESS OF THE BUFFER.
3903 ;*
3904 ;*
3905 ;*CALL JSR R0,TESTDEF
3906 ;* <ADDRESS OF COMMA DETECTION RETURN>
3907 ;* <ADDRESS OF CARRIAGE RETURN DETECTION RETURN>
3908 ;* <ADDRESS OF CONTROL Z (^Z) DETECTION RETURN>
3909 ;* <ADDRESS OF CONTROL C (^C) DETECTION RETURN>
3910 ;* RETURN
3911 ;*
3912 ;*****
3913
3914 020116 104403 TESTDEF: RDLIN ;GET RESPONSE
3915 020120 012603 MOV (SP)+,R3 ;STORE BUFFER ADDRESS
3916 070122 171727 000054 CMPR (R3),R0 ;CHECK IF COMMA
3917 020126 001415 BFC 10$ ;YES, RETURN
3918 020130 005720 TST (R0)+ ;ADJUST R0
3919 020132 105717 TSTR (R3) ;CHECK IF CARRIAGE RETURN
3920 020134 001412 BFC 10$ ;YES, RETURN
3921 020136 005720 TST (R0)+ ;ADJUST R0
3922 020140 122713 000032 CMPR #32,(R3) ;CHECK IF CONTROL Z (^Z)
3923 070144 001406 BEQ 10$ ;YES, RETURN
3924 020146 005720 TST (R0)+ ;ADJUST R0
3925 020150 122713 000003 CMPR #3,(R3) ;CHECK IF CONTROL C (^C)
3926 070154 001402 BFC 10$ ;YES, RETURN
3927 020156 005720 TST (R0)+ ;ADJUST R0
3928 020160 000200 RTS R0 ;RETURN
3929
3930 070162 011000 10$: MOV (R0),R0 ;STORE RETURN ADDRESS
3931 070164 000200 RTS R0 ;RETURN

```

```

3932 .SBTTL LOAD DISK EXCLUDED AREAS
3933 ;*****
3934 ;*
3935 ;* THIS ROUTINE WILL ACCEPT TTY INPUT FOR EXCLUDED AREAS AND
3936 ;* LOAD APPROPRIATE IDENTIFICATION NUMBERS IN THE EXCLUDED AREAS OF
3937 ;* THE PARAMETER BLOCK. THE FOLLOWING THREE FORMATS ARE
3938 ;* ACCEPTED BY THIS ROUTINE:
3939 ;*
3940 ;*
3941 ;*      CYL
3942 ;*      CYL,TRK
3943 ;*      CYL1,TRK1,CYL2,TRK2
3944 ;*
3945 ;* THE FOLLOWING FORMULA IS USED:
3946 ;*
3947 ;*      CYL1*66+TRK1*22+1 = BEGINNING OF EXCLUDED AREA
3948 ;*      CYL2*66+TRK2*22+1 = END OF EXCLUDED AREA
3949 ;*
3950 ;*      REGISTER      USE
3951 ;*      -----      ---
3952 ;*
3953 ;*      R1            ADDRESS IN EXCLUDED AREA
3954 ;*      R2            ADDRESS OF INPUT STRING
3955 ;*      R4            ADDRESS IN INPUT STRING
3956 ;*      R5            PARAMETER BLOCK ADDRESS
3957 ;*
3958 ;*CALL JSR      R0,LDEXAR
3959 ;*      <ADDRESS OF MESSAGE TO PRINT>
3960 ;*      <ADDRESS FOR CARRIAGE RETURN OR CONTROL Z <^Z> DETECTION>
3961 ;*      <ADDRESS FOR CONTROL C <^C> DETECTION>
3962 ;*      RETURN
3963 ;*
3964 ;*****
3965
3966 020166 012037 020174 LDEXAR: MOV      (R0)+,2S      ;LEGAL NUMBER OF EXCLUSION AREA
3967
3968 020172 104401 1S:      TYPE      ;TYPE WITH EXCLUDED AREA =
3969 020174 000000 2S:      .WORD      0
3970 020176 104401 057322 TYPE      ,PAR018
3971 020202 004037 020116 JSR      R0,TSTDEF      ;CHECK FOR DEFAULT PARAMETERS
3972 070706 020617 30$      ;COMMA DETECTED
3973 020210 020634 45$      ;CARRIAGE RETURN DETECTED
3974 020212 020634 45$      ;CONTROL Z <^Z> DETECTED
3975 020214 020632 40$      ;CONTROL C <^C> DETECTED
3976 020216 010346 MOV      R3,-(SP)      ;STORE BUFFER ADDRESS
3977 020220 004737 053316 JSR      PC,CCTBIN      ;CONVERT TO BINARY
3978 020224 020612 30$      ;ERROR RETURN
3979 020226 022716 000632 CMP      #410,-(SP)      ;CHECK IF LEGAL CYLINDER
3980 020232 103566 BLO      29$      ;NO, TRY AGAIN
3981 020234 011637 001612 MOV      (SP),FXARFA      ;MULTIPLY CYLINDER ADDRESS BY 3
3982 020240 006316 ASL      (SP)
3983 020242 062637 001612 ADD      (SP)+,EXAREA
3984 020246 010304 MOV      R3,P4      ;STORE BEGINNING OF COMMAND STRING
3985
3986 020250 105714 64$:     TSTR      (P4)      ;CHECK FOR CARRIAGE RETURN
3987 020252 001525 BFQ      10$      ;YES, EXIT
  
```

3989	020754	122427	000054	CMPR	(R4)+,B",	;CHECK FOR COMMA
3989	020260	001373		BNE	64\$	;NO, GO TEST NEXT CHARACTER
3990	020262	105714		TSTB	(R4)	;CHECK IF CARRIAGE RETURN
3991	020264	001552		BEQ	30\$	;YES, TRY AGAIN
3992	020266	121427	000054	CMPR	(R4)+,B",	;CHECK IF COMMA
3993	020272	001547		BFO	30\$	;YES, TRY AGAIN
3994	020274	010446		MOV	R4,-(SP)	;STORE BUFFER ADDRESS
3995	020276	004737	053316	JSR	PC,OCTBIN	;CONVERT TO BINARY
3996	020302	020617		30\$		;ERRCR RETURN
3997	020304	022716	000002	CMP	#2,(SP)	;CHECK IF LEGAL TRACK
3998	020310	103537		BLO	29\$	;TRY AGAIN
3999	020312	063716	001612	ADD	EXAREA,(SP)	;PUT MULTIPLIER ON STACK
4000	020316	012746	000026	MOV	#22,-(SP)	;PUT MULTIPLIER ON STACK
4001	020322	004737	054064	JSR	PC,\$MULT	;DO MULTIPLICATION
4002	020326	005216		INC	(SP)	;ADD 1
4003	020330	012637	001612	MOV	(SP)+,EXAREA	;STORE VALUE
4004	020334	005726		TST	(SP)+	;THROW AWAY MOST SIGNIFICANT BITS
4005						
4006	020336	105714		65\$: TSTB	(P4)	;CHECK FOR CARRIAGE RETURN
4007	020340	001511		BEQ	20\$	;YES, EXIT
4008	020342	122427	000054	CMPB	(R4)+,B",	;CHECK FOR COMMA
4009	020346	001373		BNE	64\$	;NO, GO TEST NEXT CHARACTER
4010	020350	105714		TSTB	(P4)	;CHECK IF CARRIAGE RETURN
4011	020352	001517		BEQ	30\$	;YES, TRY AGAIN
4012	020354	121427	000054	CMPB	(R4)+,B",	;CHECK IF COMMA
4013	020360	001514		BEQ	30\$	;YES, TRY AGAIN
4014	020362	010446		MOV	R4,-(SP)	;STORE BUFFER ADDRESS
4015	020364	004737	053316	JSR	PC,OCTBIN	;CONVERT TO BINARY
4016	020370	020617		30\$		;ERRCR RETURN
4017	020372	022716	000632	CMP	#410,-(SP)	;CHECK IF LEGAL CYLINDER
4018	020376	103504		BLO	29\$	;NO, TRY AGAIN
4019	020400	011637	001614	MOV	(SP),FXARFA+2	;MULTIPLY CYLINDER ADDRESS BY 3
4020	020404	006316		ASL	(SP)	
4021	020406	062637	001614	ADD	(SP)+,EXAREA+2	
4022						
4023	020412	105714		66\$: TSTB	(R4)	;CHECK FOR CARRIAGE RETURN
4024	020414	001476		BEQ	30\$	;YES, EXIT
4025	020416	122427	000054	CMPB	(R4)+,B",	;CHECK FOR COMMA
4026	020422	001373		BNE	66\$	;NO, GO TEST NEXT CHARACTER
4027	020424	105714		TSTB	(R4)	;CHECK IF CARRIAGE RETURN
4028	020426	001471		BEQ	30\$	;YES, TRY AGAIN
4029	020430	121427	000054	CMPB	(R4)+,B",	;CHECK IF COMMA
4030	020434	001466		BEQ	30\$	;YES, TRY AGAIN
4031	020436	010446		MOV	R4,-(SP)	;STORE BUFFER ADDRESS
4032	020440	004737	053316	JSR	PC,OCTBIN	;CONVERT TO BINARY
4033	020444	020612		30\$		;ERRCR RETURN
4034	020446	022716	000002	CMP	#2,(SP)	;CHECK IF LEGAL TRACK
4035	020452	103456		BLO	29\$	;TRY AGAIN
4036	020454	063716	001614	ADD	EXAREA+2,(SP)	;PUT MULTIPLIER ON STACK
4037	020460	012746	000026	MOV	#22,-(SP)	;PUT MULTIPLIER ON STACK
4038	020464	004737	054064	JSR	PC,\$MULT	;DO MULTIPLICATION
4039	020470	005216		INC	(SP)	;ADD 1
4040	020472	012637	001614	MOV	(SP)+,EXAREA+2	;STORE VALUE
4041	020476	005726		TST	(SP)+	;THROW AWAY MOST SIGNIFICANT BITS
4042	020500	023737	001612 001614	CMP	EXAREA,EXAREA+2	;CHECK IF LEGAL RANGE
4043	020506	103041		BHIS	30\$	;NO, TRY AGAIN

4044	020510	013721	001612		MOV	EXAREA,(R1)+	;LOAD EXCLUDED AREA
4045	020514	013721	001614		MOV	EXAREA+2,(R1)+	
4046	020520	062700	000004		ADD	#4,R0	;ADJUST R0
4047	020524	000200			RTS	R0	;RETURN
4048							
4049	020526	013746	001612	10\$:	MOV	EXAREA,-(SP)	;LOAD MULTIPLIER
4050	020532	012746	000026		MOV	#22,-(SP)	;LOAD MULTICAND
4051	020536	004737	054064		JSR	PC,\$MULT	;CALCULATE VALUE
4052	020542	005216			INC	(SP)	;INCREMENT RESULT
4053	020544	011621			MOV	(SP),(R1)+	;STORE EXCLUDED AREA
4054	020546	062716	000102		ADD	#66,(SP)	
4055	020552	012621			MOV	(SP)+,(R1)+	
4056	020554	005726			TST	(SP)+	;THROW AWAY MOST SIGNIFICANT BITS
4057	020556	062700	000004		ADD	#4,R0	;ADJUST R0
4058	020562	000700			RTS	R0	;RETURN
4059							
4060	020564	013721	001612	20\$:	MOV	EXAREA,(R1)+	;STORE EXCLUDED AREA
4061	020570	062737	000026	001612	ADD	#22,EXAREA	
4062	020576	013721	001612		MOV	EXAREA,(R1)+	
4063	020602	062700	000004		ADD	#4,R0	;ADJUST R0
4064	020606	000200			RTS	R0	;RETURN
4065							
4066	020610	005726		29\$:	TST	(SP)+	;ADJUST STACK
4067	020612	010337	020620	30\$:	MOV	R3,31\$	;LOAD BUFFER ADDRESS
4068	020616	104401			TYPE		;TYPE RECEIVED INPUT
4069	020620	000000		31\$:	.WORD	0	;BUFFER ADDRESS
4070	020622	104401	001164		TYPE	,SQUES	;TYPE QUESTION MARK
4071	020626	000137	020172		JMP	1\$	;TRY AGAIN
4072							
4073	020632	005720		40\$:	TST	(R0)+	;ADJUST R0
4074	020634	011000		45\$:	MOV	(R0),R0	;STORE SPECIAL RETURN ADDRESS
4075	020636	000200			RTS	R0	;RETURN

```

4076 .SBTTL GENFRATE NEW RANDOM COMMAND
4077
4078 ;;*****
4079 ;*
4080 ;* IF SWITCH 14 IS RESET, THIS ROUTINE WILL STORE THE PREVIOUS
4081 ;* COMMAND AND GENERATE A NEW RANDOM COMMAND. A CHECK IS MADE
4082 ;* ON THE DISK ADDRESS AND THE DISK ADDRESS + THE NUMBER OF
4083 ;* SECTORS TO BE READ OR WRITTEN. IF THE DATA TRANSFER WILL
4084 ;* CAUSE DATA TO BE READ OR WRITTEN BEYOND THE MAXIMUM PACK
4085 ;* ADDRESS OR USES ANY PART OF AN EXCLUDED AREA, A NEW DISK
4086 ;* ADDRESS WILL BE RANDOMLY GENERATED.
4087 ;*
4088 ;* REGISTER USE
4089 ;* -----
4090 ;*
4091 ;* R1 EXCLUSION AREA COUNT
4092 ;* R3 EXCLUSION AREA INDEX
4093 ;* R4 SECTOR POSITION
4094 ;* R5 PARAMETER BLOCK ADDRESS
4095 ;* (SP) SECTORS TO BE TRANSFERRED
4096 ;*
4097 ;*
4098 ;* IF SWITCH 14 IS SET, THIS ROUTINE WILL SEEK TO THE PREVIOUS
4099 ;* DISK POSITION OR REISSUE THE LAST COMMAND DEPENDENT ON THE
4100 ;* P-SEEK FLAG IN THE PARAMETER BLOCK.
4101 ;*
4102 ;*
4103 ;* ROUTINES USED
4104 ;* -----
4105 ;*
4106 ;* Q.PUSH
4107 ;* SPAND
4108 ;* SMULT
4109 ;*
4110 ;*CALL JSR PC,GENERT
4111 ;* RETURN
4112 ;*
4113 ;;*****
4114
4115 020640 005065 000146 GENERT: CLR P.RECT(R5) ;CLEAR ERROR FLAGS AND
4116 020644 005065 000150 CLR P.DSTT(R5) ; RETRY COUNTS
4117 020650 005065 000212 CLR P.EPR(R5)
4118
4119 020654 032777 040000 160256 BIT #SW14,#SWR ;CHECK IF NEW COMMAND
4120 ; SHOULD BE GENERATED
4121 020662 001454 BFC 10$ ;YES, GENERATE NEW COMMAND
4122 020664 105765 000211 TSTB P.SEEK(R5) ;CHECK IF PREVIOUS CYLINDER ISSUED
4123 020670 001020 BNE 1$ ;YES, SET UP DATA TRANSFER
4124 020672 112765 177777 000211 MOVR #-1,P.SEEK(R5) ;INDICATE SEEK HAS BEEN ISSUED
4125 020700 112765 000117 000001 MOVR #SECK,P.CMND(R5) ;LOAD SEEK PARAMETERS
4126 020706 016565 000136 000002 MOV P.LCYL(R5),P.CYLN(R5) ;LOAD PRIOR CYLINDER
4127 020714 016565 000140 000004 MOV P.LSEC(R5),P.SECT(R5) ;LOAD PRIOR SECTION AND TRACK
4128 020722 004037 050706 JSR R0,Q.PUSH ;PUT PARAMETER BLOCK ON
4129 020726 001324 CINITQ ; COMMAND INITIATION QUEUE
4130 020730 000207 RTS PC ;RETURN
4131

```

4132	020732	105065	000211		15:	CLRP	P.SERR(R5)	;INDICATE DATA TRANSFER ISSUED
4133	020736	116565	000122	000121		MOVB	P.RDPT(R5),P.DPAT(R5)	;LOAD LAST DATA PATTERN
4134	020744	116565	000123	000001		MOVB	P.RCMD(R5),P.CMND(R5)	;LOAD LAST COMMAND
4135	020752	016565	000124	000002		MOV	P.RCYL(R5),P.CYLN(R5)	;LOAD LAST CYLINDER ADDRESS
4136	020760	122765	000131	000001		CMPB	#WRTCHK,P.CMND(R5)	;CHECK IF WRITE CHECK
4137	020766	001003				BNE	25	;NO, DO NOT SET FLAG
4138	020770	052765	000200	000014		BIS	#W.MCK,P.PRST(R5)	;SET ISSUE WRITE FOR WRITE CHECK
4139	020776	016565	000126	000004	25:	MOV	P.RSEC(R5),P.SECT(R5)	;LOAD LAST SECTOR AND TRACK
4140	021004	016565	000130	000012		MOV	P.RWC(R5),P.WC(R5)	;LOAD LAST WORD COUNT
4141	021012	000207				RTS	PC	;RETURN
4142								
4143	021014	005077	160124		10\$:	CLR	#STKS	;LOCK OUT TTY INTERRUPTS
4144	021020	010446				MOV	R4,-(SP)	;STORE R4 ON STACK
4145	021022	010346				MOV	R3,-(SP)	;STORE R3 ON STACK
4146	021024	010146				MOV	R1,-(SP)	;STORE R1 ON STACK
4147	021026	010046				MOV	R0,-(SP)	;STORE R0 ON STACK
4148	021030	116565	000123	000135		MOVB	P.RCMD(R5),P.LCMD(R5)	;STORE LAST COMMAND
4149	021036	016565	000124	000136		MOV	P.RCYL(R5),P.LCYL(R5)	;STORE LAST CYLINDER
4150	021044	016565	000126	000140		MOV	P.RSEC(R5),P.LSEC(R5)	;STORE LAST SECTOR AND TRACK
4151	021052	016565	000130	000142		MOV	P.RWC(R5),P.LWC(R5)	;STORE LAST WORD COUNT
4152	021060	116565	000122	000144		MOVB	P.RDPT(R5),P.LDPT(R5)	;STORE LAST PATTERN
4153	021066	004737	054200			JSR	PC,SRAND	;GENERATE RANDOM NUMBER
4154	021072	013746	054300			MOV	\$LONUM,-(SP)	;LOAD MULTIPLIER
4155	021076	013746	001354			MOV	MAXBUF,-(SP)	;LOAD MULTIPLICAND
4156	021102	005216				INC	(SP)	;INCREMENT WORD COUNT MAX.
4157	021104	004737	054064			JSR	PC,\$MULT	;CALCULATE WORD COUNT
4158	021110	005726				TST	(SP)+	;THROW AWAY LEAST SIGNIFICANT BITS
4159	021112	042716	100003			BIC	#100003,(SP)	;MAKE DIVISIBLE BY 4 AND < 32K
4160	021116	001002				BNE	11\$	;IF NOT ZERO USE WORD COUNT
4161	021120	012716	000004			MOV	#4,(SP)	;MAKE WORD COUNT 4
4162	021124	116604	000001		11\$:	MOVB	1(SP),R4	;CALCULATE NUMBER OF SECTORS TO
4163	021130	105716				TSTR	(SP)	; TRANSFERRED
4164	021132	001401				BEQ	12\$	;IF EVEN SECTORS DO NOT MODIFY
4165	021134	005204				INC	R4	; SECTOR COUNT
4166	021136	011646			12\$:	MOV	(SP),-(SP)	;SAVE WORD COUNT
4167	021140	005416				NEG	(SP)	;GENERATE NEGATIVE NUMBER
4168	021142	012665	000012			MOV	(SP)+,P.WC(R5)	;LOAD PRESENT WORD COUNT
4169	021146	062765	000001	000152		ADD	#1,P.WORD(R5)	;INCREMENT NUMBER OF WORDS
4170	021154	005565	000154			ADC	P.WORD+2(R5)	
4171	021160	113746	054276			MOVB	\$MINUM,-(SP)	;RANDOM NUMBER FOR COMMAND
4172	021164	042716	177770			BIC	#177770,(SP)	;KEEP LOW 3 BITS
4173	021170	126526	000076			CMPB	P.RATE(R5),(SP)+	;DETERMINE IF READ OR WRITE
4174	021174	101462				BLOS	15\$	;ISSUE READ
4175	021176	113746	054277			MOVB	\$MINUM+1,-(SP)	;RANDOM NUMBER FOR PATTERN
4176	021202	042716	177760			BIC	#177760,(SP)	;KEEP LOW 4 BITS
4177	021206	111665	000122			MOVB	(SP),P.RDPT(R5)	;STORE DATA PATTERN
4178	021212	112665	000121			MOVB	(SP)+,P.DPAT(R5)	
4179	021216	105765	000077			TSTR	P.AWCK(R5)	;CHECK IF AUTO WRITE CHECK
4180	021222	001017				BNE	13\$	;YES, ISSUE WRITE CHECK
4181	021224	113746	054276			MOVB	\$MINUM,-(SP)	;RANDOM NUMBER FOR WRITE CHECK
4182	021230	042726	177707			BIC	#177707,(SP)+	;CHECK IF WRITE COMMAND
4183	021234	001412				BEQ	13\$	;YES, ISSUE WRITE CHECK
4184	021236	112765	000123	000123		MOVB	#WRDATA,P.RCMD(R5)	;ISSUE WRITE DATA
4185	021244	062665	000156			ADD	(SP)+,P.WRT(R5)	;ADD NUMBER OF WORDS TO BE WRITTEN
4186	021250	005565	000160			ADC	P.WRT+2(R5)	
4187	021254	005565	000162			ADC	P.WRT+4(R5)	

```
4188 021260 000441 BR 18$
4189
4190 021262 112765 000131 000123 17$: MOVR #WRTCHK,P.RCMD(R5) ;ISSUE WRITE CHECK
4191 021270 052765 000200 000014 BTS #W.WCK,P.PRST(R5) ;SET NO WRITE FOR WRITE CHECK
4192 021276 062765 000001 000152 ADD #1,P.WORD(R5) ;INCREMENT NUMBER OF COMMAND ISSUED
4193 021304 005565 000154 ADC P.WORD+2(R5)
4194 021310 061665 000156 ADD (SP),P.WWRT(R5) ;ADD NUMBER OF WORDS TO BE WRITTEN
4195 021314 005565 000160 ADC P.WWRT+2(R5)
4196 021320 005565 000162 ADC P.WWRT+4(R5)
4197 021324 062665 000164 ADD (SP)+,P.NRD(R5) ;ADD NUMBER OF WORDS READ
4198 021330 005565 000166 ADC P.NRD+2(R5)
4199 021334 005565 000170 ADC P.NRD+4(R5)
4200 021340 000411 BR 18$
4201
4202 021342 112765 000121 000123 15$: MOVR #RDATA,P.RCMD(R5) ;ISSUE READ DATA
4203 021350 062665 000164 ADD (SP)+,P.NRD(R5) ;ADD NUMBER OF WORDS READ
4204 021354 005565 000166 ADC P.NRD+2(R5)
4205 021360 005565 000170 ADC P.NRD+4(R5)
4206 021364 116565 000123 000001 18$: MOVB P.RCMD(R5),P.CMND(R5) ;LOAD COMMAND
4207 021372 012700 000100 MOV #100,R0 ;LOCAL REGISTER COUNT
4208
4209 021376 010503 20$: MOV R5,R3 ;LOAD INDEX
4210 021400 062703 000232 ADD #P.FXAR,R3 ;CALCULATE EXCLUDED AREA INDEX
4211 021404 012701 000005 MOV #5,R1 ;LOCAL BAD TRACK COUNT
4212 021410 004737 054200 JSR PC,SRAND ;GENERATE RANDOM NUMBER
4213 021414 016546 000070 MOV P.MXCL(R5),-(SP) ;STORE MAXIMUM CYLINDER
4214 021420 166516 000066 SUB P.MNCL(R5),-(SP) ;CALCULATE DIFFERENCE
4215 021424 001406 BEQ 21$ ;IF 0, SKIP MULTIPLY
4216 021426 005216 INC (SP) ;INCREMENT DIFFERENCE
4217 021430 013746 054300 MOV $LOWM,-(SP) ;PUT MULTIPLIER ON STACK
4218 021434 004737 054064 JSR PC,$MULT ;CALCULATE CYLINDER ADDRESS
4219 021440 005726 TST (SP)+ ;THROW AWAY LEASE SIGNIFICANT BITS
4220
4221 021442 066516 000066 21$: ADD P.MNCL(R5),-(SP) ;CALCULATE CYLINDER
4222 021446 011665 000124 MOV (SP),P.PCYL(R5) ;STORE CYLINDER
4223 021452 116537 000073 001566 MOVB P.MXTR(R5),TMPTRK ;STORE MAXIMUM TRACK
4224 021460 116537 000072 001570 MOVB P.MNTR(R5),TMPTRK+2 ;STORE MINIMUM TRACK
4225 021466 163737 001570 001566 SUB TMPTRK+2,TMPTRK ;CALCULATE PREFERENCE
4226 021474 001415 BFC 22$ ;IF 0, SKIP MULTIPLY
4227 021476 005046 CLR -(SP) ;MAKE ROOM ON STACK
4228 021500 113716 054277 MOVB $MINUM+1,(SP) ;PUT IN MULTIPLIER ON STACK
4229 021504 013746 001566 MOV TMPTRK,-(SP) ;PUT TRACK NUMBER ON STACK
4230 021510 005216 INC (SP) ;INCREMENT DIFFERENCE
4231 021512 004737 054064 JSR PC,$MULT ;CALCULATE TRACK
4232 021516 116637 000001 001566 MOVB 1(SP),TMPTRK ;STORE TRACK
4233 021524 062706 000004 ADD #4,SP ;ADJUST STACK
4234
4235 021530 063737 001570 001566 22$: ADD TMPTRK+2,TMPTRK ;CALCULATE TRACK
4236 021536 113765 001566 000127 MOVB TMPTRK,P.PTRK(R5) ;STORE TRACK NUMBER
4237 021544 116537 000075 001572 MOVB P.MXSC(R5),TMPSEC ;STORE MAXIMUM SECTOR
4238 021552 116537 000074 001574 MOVB P.MNSC(R5),TMPSEC+2 ;STORE MINIMUM SECTOR
4239 021560 163737 001574 001572 SUB TMPSEC+2,TMPSEC ;CALCULATE DIFFERENCE
4240 021566 001415 BFC 23$ ;IF 0, SKIP MULTIPLY
4241 021570 005046 CLR -(SP) ;MAKE ROOM ON STACK
4242 021572 113716 054276 MOVB $MINUM,(SP) ;PUT IN MULTIPLIER
4243 021576 013746 001572 MOV TMPSEC,-(SP) ;PUT SECTOR NUMBER ON STACK
```

4244	021602	005216				INC	(SP)	;INCREMENT DIFFERENCE
4245	021604	004737	054064			JSR	PC,SMULT	;CALCULATE SECTOR
4246	021610	116637	000001	001572		MOVB	1(SP),TMPSEC	;STORE SECTOR
4247	021616	062706	000004			ADD	#4,SP	;ADJUST STACK
4248								
4249	021622	063737	001574	001572	23\$:	ADD	TMPSEC+7,TMPSEC	;CALCULATE SECTOR
4250	021630	113765	001572	000126		MOVB	TMPSEC,P.PSFC(45)	;STORE SECTOR NUMBER
4251	021636	006316				ASL	(SP)	;MULTIPLY CYLINDER BY 3
4252	021640	066516	000124			ADD	P.RCYL(R5),(SP)	
4253	021644	063716	001566			ADD	TMPTRK,(SP)	;ADD TRACK
4254	021650	012746	000026			MOV	#22,-(SP)	;MULTIPLICAND - 22 SECTORS/TRACK
4255	021654	004737	054064			JSR	PC,SMULT	
4256	021660	012616				MOV	(SP)+,(SP)	;THROW AWAY MOST SIGNIFICANT BITS
4257	021662	063716	001572			ADD	TMPSEC,(SP)	;ADD SECTOR
4258	021666	005216				INC	(SP)	;CONSTANT FOR BAD TRACKS
4259	021670	061604				ADD	(SP),R4	;ADD NUMBER OF SECTORS
4260	021672	022704	064741			CMP	#410.*2.*3.>+<22.*2.>+1,R4	;CHECK IF DATA TRANSFER EXCEEDS DISK
4261	021676	103421				BLO	40\$	;CALCULATE NEW DISK ADDRESS
4262								
4263	021700	005713			25\$:	TST	(R3)	;CHECK IF BAD TRACK INTERVAL
4264								;EXIST
4265	021702	001407				BEQ	30\$	;NO, ISSUE COMMAND
4266	021704	021623				CMP	(SP),(R3)+	;CHECK IF IN BAD TRACK INTERVAL
4267	021706	103402				BLO	26\$	;NO, CHECK NEXT BAD TRACK 1
4268	021710	020413				CMP	R4,(R3)	
4269	021712	103413				BLO	40\$	;YES, CALCULATE NEW DISK ADDRESS
4270	021714	005723			26\$:	TST	(R3)+	;ADJUST R3
4271	021716	005301				DEC	R1	;DECREMENT BAD BLOCK COUNT
4272	021720	001367				BNE	25\$	;IF NOT 0 GET NEXT BAD BLOCK
4273								
4274	021722	005726			30\$:	TST	(SP)+	;ADJUST STACK
4275	021724	016565	000124	000002		MOV	P.RCYL(R5),P.CYLN(R5)	;LOAD CYLINDER
4276	021732	016565	000126	000004		MOV	P.RSEC(R5),P.SECT(R5)	;LOAD TRACK AND SECTOR
4277	021740	000416				BP	45\$	;RETURN
4278								
4279	021742	162604			40\$:	SUB	(SP)+,R4	;GET NUMBER OF SECTOR TRANSFERRED
4280	021744	005300				DPC	R0	;DECREMENT REGENERATE COUNT
4281	021746	001407				BEQ	41\$	;DROP DRIVE CANNOT GENERATE
4282								;RANDOM DISK ADDRESS
4283	021750	000137	021376			JMP	20\$	;GET ANOTHER DISK ADDRESS
4284								
4285	021754	013737	001450	177776	41\$:	MOV	RFPRI,PS	;LOCK OUT RK06 AND TTY INTERRUPTS
4286	021762	104401	057357			TYPE	,PAF020	;TYPE "UNABLE TO GENERATE NEW COMMAND"
4287	021766	004737	012716			JSR	PC,DROP	;DROP DRIVE
4288	021772	005037	177776			CLR	PS	;ALLOW RK06 INTERRUPTS
4289								
4290	021776	012600			45\$:	MOV	(SP)+,R0	;RESTORE P0
4291	022000	012601				MOV	(SP)+,R1	;RESTORE P1
4292	022002	012603				MOV	(SP)+,R3	;RESTORE R3
4293	022004	012604				MOV	(SP)+,R4	;RESTORE R4
4294	022006	012777	000100	157130		MOV	#1E,#STKS	;ALLOW TTY INTERRUPTS
4295	022014	000207				RTS	PC	;RETURN

```

4296          .SBTTL  GET AVAILIBLE BUFFER ROUTINE
4297
4298 022016 013746 177776      GETBUF: MOV    PS, -(SP)      ;STORE PSW ON STACK
4299 022022 013737 001450 177776      MOV    RKPRI,PS      ;LOCK OUT RK06 INTERRUPTS
4300 022030 013705 001320      MOV    BWAITQ,R5      ;CHECK IF COMMAND WAITING FOR BUFFER
4301 022034 001424      BEQ    10$      ;RETURN IF QUEUE IS EMPTY
4302 022036 004037 022114      1$:   JSR    R0,BUFFAL      ;ALLOCATE BUFFER
4303 022042 022106      10$      JSR    R0,Q.POP      ;NOT ENOUGH MEMORY RETURN
4304 022044 004037 050766      JSR    R0,Q.POP      ;GET NEXT PARAMETER BLOCK FROM
4305 022050 001320      BWAITQ      ; BUFFER WAIT QUEUE
4306 022052 005726      TST    (SP)+      ;ADJUST STACK
4307 022054 005037 177776      CLR    PS      ;ALLOW RK06 INTERRUPTS
4308 022060 004737 022532      JSR    PC,LDDATA      ;LOAD BUFFER
4309 022064 004037 050706      JSR    R0,Q.PUSH      ;PUT PARAMETER BLOCK ON
4310 022070 001324      CINITQ      ; COMMAND INITIATION QUEUE
4311 022072 013737 001450 177776      MOV    RKPRI,PS      ;LOCK OUT RK06 INTERRUPTS
4312 022100 013705 001320      MOV    BWAITQ,R5      ;CHECK IF COMMAND WAITING FOR BUFFER
4313 022104 001354      BNE    1$      ;YES, TRY TO ALLOCATE BUFFER
4314
4315 022106 012637 177776      10$:  MOV    (SP)+,PS      ;RESTORE PSW
4316 022112 000207      RTS    PC      ;RETURN

```

```

4317 .SBTTL BUFFER ALLOCATION ROUTINE
4318
4319 ;*****
4320 ;*
4321 ;* THIS ROUTINE WILL TRY TO ALLOCATE A MAIN MEMORY BUFFER
4322 ;* IF MEMORY IS AVAILABLE. (CALLED BY FRICHTY 7)
4323 ;*
4324 ;* REGISTER USE
4325 ;* -----
4326 ;*
4327 ;* R1 WORD COUNT * 2
4328 ;* R3 FREE BLOCK TABLE INDEX
4329 ;* R4 FREE BLOCK TABLE COUNT
4330 ;* R5 PARAMETER BLOCK ADDRESS
4331 ;*
4332 ;*CALL JSR R0,BUFFAL
4333 ;* <ADDRESS OF BUFFER MEMORY NOT AVAILABLE ROUTINE>
4334 ;* RETURN <BUFFER ALLOCATED>
4335 ;*
4336 ;*****
4337
4338 022114 010446 BUFFAL: MOV R4,-(SP) ;STORE R4 ON STACK
4339 022116 010346 MOV R3,-(SP) ;STORE R3 ON STACK
4340 022120 010146 MOV R1,-(SP) ;STORE R1 ON STACK
4341 022122 016501 000012 MOV P.WC(R5),R1 ;STORE WORD COUNT
4342 022126 010165 000130 MOV R1,P.WC(R5) ;STORE WORD COUNT
4343 022132 005401 NEG R1 ;GET ACTUAL WORD COUNT
4344 022134 006301 ASL R1 ;MULTIPLY WORD COUNT BY 2
4345 022136 013704 001664 MOV FBLC,R4 ;STORE FREE BLOCK TABLE ADDRESS
4346 022142 001420 BEQ 25 ;RETURN, IF NO BLOCKS AVAILABLE
4347 022144 012703 001666 MOV FBLC,R3 ;LOAD ADDRESS OF FREE BLOCK TABLE
4348 022150 022704 000011 CMP #9,R4 ;HANG IF FREE BLOCK COUNT IS GREATER
4349 022154 101001 BHI 15 ; OF EQUAL TO 9
4350 022156 000000 HALT
4351
4352 022160 012346 15: MOV (R3)+,-(SP) ;STORE BEGINNING OF FREE BLOCK
4353 022162 060116 ADD R1,(SP) ;ADD WORD COUNT
4354 022164 103411 BCS 55 ;CHECK IF ADDRESS OVERFLOW
4355 022166 021316 CMP (R3),(SP) ;CHECK IF ENOUGH ROOM
4356 022170 001423 BEQ 15$ ;YES, ELIMINATE BLOCK
4357 022172 101010 BHI 10$ ;YES, SHORTEN BLOCK
4358 022174 005723 TST (R3)+ ;ADJUST R3
4359 022176 005726 TST (SP)+ ;ADJUST STACK
4360 022200 005304 DEC R4 ;DECREMENT FREE BLOCK COUNT
4361 022202 001366 BNE 15 ;EXAMINE NEXT FREE BLOCK
4362 022204 011000 25: MOV (R0),R0 ;LOAD RETURN ADDRESS
4363 022206 000437 BR 25$ ;RETURN
4364
4365 022210 005726 55: TST (SP)+ ;ADJUST STACK
4366 022212 000774 BR 25 ;RETURN
4367
4368 022214 014365 000132 10$: MOV -(R3),P.PAL(R5) ;LOAD BUS ADDRESS
4369 022220 011365 000010 MOV (R3),P.PAL(R5) ;LOAD BUS ADDRESS
4370 022224 012613 MOV (SP)+,(R3) ;LOAD NEW BASE
4371 022226 005720 TST (R0)+ ;ADJUST RETURN
4372 022230 112765 177777 000145 MOVB #1,P.BUFF(R5) ;INDICATE THAT BUFFER HAS BEEN ASSIGNED

```

```

4373 022236 000423      BR      25$      ;RETURN
4374
4375 022240 010301      15$:  MOV      R3,R1      ;STORE WORD COUNT
4376 022242 014365 000137  MOV      -(R3),P.RBAL(R5)
4377 022246 011365 000010  MOV      (R3),P.PALQ(R5) ;LCAC BUS ADDRESS
4378 022252 005726      TST      (SP)+      ;ADJUST STACK
4379 022254 005721      TST      (R1)+      ;ADJUST R1
4380 022256 005720      TST      (R0)+      ;ADJUST RETURN
4381 022260 112765 177777 000145  MOVB     @-1,P.BUFF(R5) ;INDICATE THAT BUFFER HAS BEEN ASSIGNED
4382 022266 005337 001664      DEC      PRLKC      ;DECREMENT FREE BLOCK COUNT
4383 022272 005304      DEC      R4      ;DECREMENT REMAIN BLOCK COUNT
4384 022274 001404      BEQ      25$      ;IF 0, RETURN
4385
4386 022276 012123      20$:  MOV      (R1)+,(R3)+      ;TRANSFER TWO WORDS
4387 022300 012123      MOV      (P1)+,(R3)+
4388 022302 005304      DEC      R4      ;DECREMENT REMAINING BLOCKS TO BE ADJUSTED
4389 022304 001374      BNE      20$      ;IF NOT 0, TRANSFER NEXT TWO WORDS
4390 022306 012601      25$:  MOV      (SP)+,R1      ;RESTORE R1
4391 022310 012603      MOV      (SP)+,R3      ;RESTORE R3
4392 022312 012604      MOV      (SP)+,R4      ;RESTORE R4
4393 022314 000200      RTS      R0      ;RETURN
4394

```

```

4395      .SBTTL  BUFFER RELEASE ROUTINE
4396
4397      ;;*****
4398      ;*
4399      ;*      THIS ROUTINE WILL RELEASE BUFFER USED.
4400      ;*      (MUST BE CALLED IN PRIORITY 7)
4401      ;*
4402      ;*      REGISTER      USE
4403      ;*      -----      ---
4404      ;*
4405      ;*      R0      NEW END OF FREE BLOCK TABLE
4406      ;*      R1      LAST ADDRESS OF RELEASED BUFFER
4407      ;*      R3      FREE BLOCK INDEX
4408      ;*      R4      FREE BLOCK COUNT
4409      ;*      R5      PARAMETER BLOCK ADDRESS
4410      ;*      (SP)     BEGINNING ADDRESS OF RELEASED BLOCK
4411      ;*
4412      ;*CALL  JSR      PC,BUFREL
4413      ;*      RETURN
4414      ;*
4415      ;;*****
4416
4417      BUFREL:  MOV      R4,-(SP)      ;STORE R4 ON STACK
4418              MOV      R3,-(SP)      ;STORE R3 ON STACK
4419              MOV      R1,-(SP)      ;STORE R1 ON STACK
4420              MOV      R0,-(SP)      ;STORE R0 ON STACK
4421      022316 105765 000145      TSTB     P.BUFF(R5)      ;CHECK IF BUFFER ASSIGNED
4422      022332 001472              BEQ      25$              ;NO, RETURN
4423      022334 105065 000145      CLRB     P.BUFF(R5)      ;CLEAR BUFFER ALLOCATED
4424      022340 016501 000130      MOV      P.RWC(R5),R1      ;STORE ALLOCATED WORD COUNT
4425      022344 005401              MFC      R1              ;GET ACTUAL WORD COUNT
4426      022346 006301              ASL      R1              ;MULTIPLY BY 2
4427      022350 016546 000132      MOV      P.RBAL(R5),-(SP) ;STORE BUFFER ADDRESS
4428      022354 061601              ADD      (SP),R1      ;CALCULATE FINAL ADDRESS
4429      022356 012703 001666      MOV      #FBLKT,R3      ;LOCATE FREE BLOCK TABLE ADDRESS
4430      022362 013704 001664      MOV      FBLKC,R4      ;STORE FREE BLOCK COUNT
4431      022366 001424              BEQ      8$              ;ADD FREE BLOCK
4432
4433      022370 022301      1$:      CMP      (R3)+,R1      ;COMPARE END OF BUFFER AND FREE BLOCK
4434      022372 001427              BEQ      10$             ;EQUAL MERGE BLOCKS
4435      022374 101005              BNE     5$              ;LESS THEN ADD FREE BLOCK
4436      022376 022316              CMP      (R3)+,(SP)      ;COMPARE BEGINNING OF BUFFER
4437                                ; AND END OF FREE BLOCK
4438      022400 001426              BEQ      15$             ;EQUAL MERGE
4439      022402 005304              DEC      R4              ;DECREMENT FREE BLOCK COUNT
4440      022404 001371              BNE     1$              ;GO TO NEXT FREE BLOCK
4441      022406 000414              BR      8$              ;GO ADD BLOCK
4442
4443      022410 010446      5$:      MOV      R4,-(SP)      ;PUT REMAINING COUNT ON STACK
4444      022412 006316              ASL      (SP)      ;MULTIPLY BY 4
4445      022414 006316              ASL      (SP)
4446      022416 005743              TST      -(R3)      ;DECREMENT R3 BY 2
4447      022420 062603              ADD      (SP)+,R3      ;CALCULATE OLD END OF FREE BLOCK TABLE
4448      022422 010300              MOV      R3,R0
4449      022424 062700 000004      ADD      #4,R0      ;CALCULATE NEW END OF FREE BLOCK TABLE
4450      022430 014340      6$:      MOV      -(R3),-(R0)      ;MOVE TABLE

```

4451	022432	014340		MOV	-(R3),-(R0)	
4452	022434	005304		DEC	R4	;DECREMENT FREE BLOCK COUNT
4453	022436	001374		BNE	6\$	;IF NOT ZERO, COUNT
4454						
4455	022440	005237	001664	8\$: INC	FBLKC	;INCREMENT FREE BLOCK COUNT
4456	022444	012623		MOV	(SP)+,(R3)+	;STORE BEGINNING OF FREE BLOCK
4457	022446	010123		MOV	R1,(R3)+	;STORE END OF FREE BLOCK
4458	022450	000423		BR	25\$	;RETURN
4459						
4460	022452	012643		10\$: MOV	(SP)+,-(R3)	;LOAD BEGINNING OF FREE BLOCK
4461	022454	000421		BR	25\$	;RETURN
4462						
4463	022456	005726		15\$: TST	(SP)+	;ADJUST STACK
4464	022460	005304		DEC	R4	;DECREMENT FREE BLOCKS LEFT
4465	022462	001402		BEQ	16\$	;NO MORE FREE BLOCK ADJUST END
4466	022464	021301		CMP	(R3),R1	;CHECK IF RELEASED BLOCK END=
4467						; FREE BLOCK BEGINNING
4468	022466	001402		BEQ	20\$	;YES, MERGE BLOCKS
4469	022470	010143		16\$: MOV	R1,-(R3)	;LOAD END OF FREE BLOCK
4470	022472	000412		BR	25\$	;RETURN
4471						
4472	022474	010300		20\$: MOV	R3,R0	;STORE R3 FOR BUFFER MOVEMENT
4473	022476	005743		TST	-(R3)	;ADJUST R3
4474	022500	005720		TST	(R0)+	;ADJUST R0
4475	022502	006304		ASL	R4	;MULTIPLY COUNT BY TWO
4476	022504	005304		DEC	R4	;DECREMENT TRANSFER COUNT
4477	022506	012023		21\$: MOV	(R0)+,(R3)+	;MOVE TABLE ENTRY
4478	022510	005304		DEC	R4	;DECREMENT COUNT
4479	022512	001375		BNE	21\$	;IF NOT 0, CONTINUE MOVE
4480	022514	005337	001664	DEC	FBLKC	;DECREMENT FREE BLOCK COUNT
4481						
4482	022520	012600		25\$: MOV	(SP)+,R0	;RESTORE R0
4483	022522	012601		MOV	(SP)+,R1	;RESTORE R1
4484	022524	012603		MOV	(SP)+,R2	;RESTORE R2
4485	022526	012604		MOV	(SP)+,R4	;RESTORE R4
4486	022530	000207		RTS	PC	;RETURN
4487						

```

4488 .SBTTL LOAD DATA BUFFER
4489
4490 ;*****
4491 ;*
4492 ;* THE REGISTERS BE USED AS FOLLOWS:
4493 ;*
4494 ;* REGISTER USE
4495 ;* -----
4496 ;*
4497 ;* R5 ADDRESS OF PARAMETER BLOCK
4498 ;* R4 BUFFER ADDRESS
4499 ;* R3 PATTERN ADDRESS
4500 ;* R2 WORDS LEFT IN SECTOR
4501 ;* R1 WORD COUNT
4502 ;* R0 PATTERN COUNT
4503 ;*
4504 ;*CALL JSR PC,LDDATA
4505 ;* RETURN
4506 ;*
4507 ;*
4508 ;* THE DATA WILL LOOK AS FOLLOWS:
4509 ;*
4510 ;* -----
4511 ;* CPU ID PAT PAT WORD 0
4512 ;* NO. OF PAT. WORDS (N) NO. OF PAT. WORDS (N) WORD 1
4513 ;* RANDOM DATA WORDS 2
4514 ;* TO N+1
4515 ;* -----
4516 ;* ZFRC FILL WORDS N+2
4517 ;* TO 255
4518 ;* -----
4519 ;*
4520 ;*****
4521
4522 022532 010446 LDDATA: MOV R4,-(SP) ;STORE R4 ON STACK
4523 022534 010346 MOV R3,-(SP) ;STORE R3 ON STACK
4524 022536 010246 MOV R2,-(SP) ;STORE R2 ON STACK
4525 022540 010146 MOV R1,-(SP) ;STORE R1 ON STACK
4526 022542 010046 MOV R0,-(SP) ;STORE R0 ON STACK
4527 022544 016504 000132 MOV P.RPAL(R5),R4 ;GET BUS ADDRESS
4528 022550 016501 000130 MOV P.RWC(R5),R1 ;STORE WORD COUNT
4529 022554 005401 NEG K1 ;GET POSITIVE NUMBER FOR WORD COUNT
4530 022556 122765 000121 000001 CMP P.RDDATA,P.CMND(R5) ;CHECK IF READ DATA
4531 022564 001005 BNE SS ;NO, LOAD BUFFER
4532 022566 005024 3$: CLK (R4)+ ;CLEAR BUFFER BEFORE READ
4533 022570 005301 DEC K1 ;CHECK IF DONE
4534 022572 001375 BNE 3$ ;NO, CONTINUE
4535 022574 000137 022744 JMP 25$ ;RESTORE REGISTERS
4536
4537 022600 116500 000121 5$: MOVR P.DPAT(R5),R0 ;STORE PATTERN COUNT
4538 022604 010046 MOV R0,-(SP) ;STORE PATTERN FOR FIRST WORD
4539 022606 006316 ASL (SP) ;MULTIPLY PATTERN BY 4 FOR
4540 022610 006316 ASL (SP) ; PREDUNACY
4541 022612 006316 ASL (SP)
4542 022614 006316 ASL (SP)
4543 022616 050016 BIS R0,(SP) ;STORE FIRST WORD OF SECTOR

```

A10

4544	022620	113766	001364	000001		MOVB	CPUID,1(SP)		; ON STACK
4545	022626	006300				ASL	R0		;MULTIPLY DATA PATTERN BY 2
4546	022630	016003	004576			MOV	PATTAR(P0),P3		;LOAD DATA PATTERN ADDRESS
4547	022634	012700	000040			MOV	#32.,R0		;LOAD PATTERN COUNT
4548	022640	011624			11\$:	MOV	(SP),(R4)+		;STORE PATTERN AND CPU ID IN 1ST WORD
4549	022642	162701	000002			SUB	#2,P1		;DECREMENT WORD COUNT BY 2
4550	022646	022701	000376			CMP	#254.,R1		;CHECK IF REST OF SECTOR WILL BE
4551									; WRITER
4552	022652	101003				BRI	12\$		;NO, USE R1 AS WORD COUNT
4553	022654	012746	000376			MOV	#254.,-(SP)		;STORE SECTOR COUNT = 254
4554	022660	000401				BP	13\$		
4555									
4556	022662	010146			12\$:	MOV	R1,-(SP)		;STORE REMAINING WORD COUNT
4557	022664	111666	000001		13\$:	MOVB	(SP),1(SP)		;ADD REDUNDACY
4558	022670	012624				MOV	(SP)+,(P4)+		;STORE WORD COUNT IN 2ND WORD
4559	022672	012702	000376			MOV	#254.,R2		;LOAD SECTOR WORD COUNT
4560									
4561	022676	012324			15\$:	MOV	(P3)+,(R4)+		;STORE DATA PATTERN
4562	022700	005301				DFC	R1		;DECREMENT WORD COUNT
4563	022702	001417				BEQ	20\$		;BRANCH IF FINISHED
4564	022704	005302				DEC	R2		;DECREMENT SECTOR COUNT
4565	022706	005300				DEC	R0		;DECREMENT PATTERN COUNT
4566	022710	001407				BFG	17\$		;IF ZERO, INITIALIZE PATTERN
4567	022712	005702				TST	R2		;ARE WE AT END OF SECTOR
4568	022714	001370				BNE	15\$		;NO, CONTINUE TO LOAD DATA PATTERN
4569	022716	162703	000074			SUB	#60.,P3		;INITIALIZE PATTERN ADDRESS
4570	022722	012700	000040			MOV	#32.,P0		;INITIALIZE PATTERN COUNT
4571	022726	000744				BR	11\$		;START NEXT SECTOR
4572									
4573	022730	162703	000100		17\$:	SUB	#64.,P3		;INITIALIZE PATTERN ADDRESS
4574	022734	012700	000040			MOV	#32.,P0		;INITIALIZE PATTERN
4575	022740	000756				BR	15\$		;PEPEAT PATTERN
4576									
4577	022742	005726			20\$:	TST	(SP)+		;ADJUST STACK
4578	022744	012600			2\$:	MOV	(SP)+,R0		;RESTORE R0
4579	022746	012601				MOV	(SP)+,R1		;RESTORE R1
4580	022750	012602				MOV	(SP)+,R2		;RESTORE R2
4581	022752	012603				MOV	(SP)+,R3		;RESTORE R3
4582	022754	012604				MOV	(SP)+,R4		;RESTORE R4
4583	022756	000207				RTS	PC		;RETURN

```

4584 .SBTTL FIND FIRST DATA ERROR
4585
4586 022760 032777 000002 156152 BUFE1: BIT #SW1,0SWR ;CHECK IF INHIBIT SOFTWARE DATA
4587 ; CCMPARE
4588 022766 001017 BNE SS ;YES, RETURN
4589 022770 013765 001362 000220 MOV SOFCMP,P.CMLG(R5) ;LOAD COMPARISON LENGTH
4590 022776 001413 BEQ SS ;IF NO MORE COMPARISONS, RETURN
4591 023000 016565 000010 000214 MOV P.BALO(R5),P.FRHD(R5) ;LOAD ADDRESS OF START OF SECTOR
4592 023006 016565 000012 000222 MOV P.WC(R5),P.BFCP(R5) ;LOAD TRANSFER LENGTH
4593 023014 005465 000222 MFC P.BFCP(R5) ;MAKE IT POSITIVE IF POSITIVE
4594 023020 004737 023036 JSR PC,CMPBUF ;DO COMPARISON
4595 023024 023032 10$ ;FRCR RETURF
4596
4597 023026 005720 5$: TST (R0)+ ;ADJUST R0 FOR RETURN
4598 073030 000200 RTS R0 ;RETURN
4599
4600 023032 011000 10$: MOV (R0),R0 ;LOAD ERROR RETURN
4601 023034 000200 RTS R0 ;RETURN

```

```
4602 .SBTTL COMPARE DATA BUFFER
4603
4604 ;*****
4605 ;*
4606 ;* THE REGISTERS WILL BE USED AS FOLLOWS:
4607 ;*
4608 ;* REGISTER USE
4609 ;* -----
4610 ;*
4611 ;* R5 ADDRESS OF PARAMETER BLOCK
4612 ;* R4 BUFFER ADDRESS
4613 ;* R3 PATTERN ADDRESS
4614 ;* R2 SECTOR WORD COUNT
4615 ;* R1 WORDS ON SECTOR WRITTEN
4616 ;* R0 PATTERN COUNT
4617 ;*
4618 ;*CALL JSR PC,CMPBNF
4619 ;* <ADDRESS OF COMPARISON ERROR RETURN>
4620 ;* RETURN
4621 ;*
4622 ;* (SP) NUMBER OF SOFT DATA COMPARES
4623 ;* (SP+2) WORD COUNT
4624 ;*
4625 ;*****
4626
4627 023036 010446 CMPBUF: MOV R4,-(SP) ;STORE R4
4628 023040 010346 MOV R3,-(SP) ;STORE R3
4629 023042 010246 MOV R2,-(SP) ;STORE R2
4630 023044 010146 MOV R1,-(SP) ;STORE R1
4631 023046 010046 MOV R0,-(SP) ;STORE R0
4632 073050 016504 000214 MOV P.ERHD(R5),R4 ;GET START ADDRESS
4633 023054 016546 000222 MOV P.BFCP(R5),-(SP) ;GET COMPARE COUNT
4634 023060 016546 000220 MOV P.CMLC(R5),-(SP) ;LOAD NUMBER OF COMPARISON
4635 023064 001007 BNE 10$ ;COMPARE IF COMPARE COUNT NOT ZERO
4636 023066 000137 023322 JMP 40$ ;ELSE RETURN
4637
4638 023072 010465 000714 10$: MOV R4,P.ERHD(R5) ;LOAD SECTOR HEAD IN CASE OF ERROR
4639 023076 005002 CLF R2 ;CLEAR CFFSFT FROM HEAD
4640 023100 012400 MOV (R4)+,R0 ;STORE 1ST WORD (PATTERN INDEX)
4641 023102 010003 MOV R0,R3
4642 023104 042700 177760 BIC #177760,R0 ;ZERO LOW 4 BITS
4643 023110 042703 177417 BIC #177417,R3 ;MASK CPU ID
4644 023114 006203 ASH R3 ;SHIFT 4 BITS RIGHT
4645 073116 006203 ASR R3
4646 023120 006203 ASR R3
4647 023122 006203 ASR R3
4648 023124 020003 CMP R0,R3 ;CHECK FOR PATTERN MATCH
4649 023126 001066 BNE 30$ ;NO, REPORT ERROR
4650 023130 005202 INC R2 ;INCREMENT CFFSFT
4651 023132 005366 000002 DEC 2(SP) ;DECREMENT COMPARISON COUNT
4652
4653 073136 006300 13$: ASL R0 ;MULTIPLY INDEX BY 2
4654 023140 016003 00457, MOV PATTAR(R0),R3 ;STORE PATTERN ADDRESS
4655 023144 126414 0000,1 CMPH 1(R4),(R4) ;CHECK WORD COUNT
4656 023150 001055 BNE 30$ ;NO, REPORT ERROR
4657 023152 012401 MOV (R4)+,R1 ;STORE WORDS IN SECTOR WRITTEN
```

4658	023154	042701	177400		BIC	#177400,R1	;KEEP LOW BYTE
4659	023160	005202			INC	R2	;INCREMENT OFFSET VALUE
4660	023162	005366	000002		DFC	2(SP)	;DECREMENT WORD COUNT
4661	023166	005765	000116		TST	P.SAMPL(R5)	;CHECK IF SAMPLED COMPARE
4662	023172	001467			BFQ	50\$	;YES, START SAMPLED COMPARE SEQUENCE
4663	023174	005701			TST	R1	;CHECK IF SECTOR ALL ZEROS
4664	023176	001426			BEQ	20\$	;YES GO TO ZERO FILL LOOP
4665	023200	012700	000040		MOV	#32.,R0	;LOAD PATTERN COUNT
4666	023204	022324		15\$:	CMP	(R3)+,(R4)+	;CHECK PATTERN
4667	023206	001036			BNE	30\$	;MISCOMPARE ERROR
4668	023210	005366	000002		DFC	2(SP)	;DECREMENT WORD COUNT
4669	023214	001442			BEQ	40\$	;NO MORE COMPARISONS
4670	023216	005316			DEC	(SP)	;DECREMENT NUMBER OF SOFT
4671							; DATA COMPARE
4672	023220	001440			BEQ	40\$	;NO MORE COMPARISONS
4673	023222	005202			INC	R2	;INCREMENT OFFSET
4674	023224	022702	000400		CMP	#256.,R2	;CHECK IF COMPLETE SECTOR COMPARES
4675	023230	001720			BEQ	10\$	;YES, START NEXT SECTOR
4676	023232	005301			DEC	R1	;DECREMENT NUMBER OF WORDS WRITTEN
4677	023234	001407			BEQ	20\$	;IF ZERO, GO TO ZERO FILL LOOP
4678	023236	005300			DEC	R0	;DECREMENT PATTERN COUNT
4679	023240	001361			BNE	15\$	;NEXT WORD
4680	023242	012700	000040		MOV	#32.,R0	;REINITIALIZE PATTERN COUNT
4681	023246	162703	000100		SUB	#64.,R3	;REINITIALIZE PATTERN ADDRESS
4682	023252	000754			BR	15\$	;CHECK NEXT WORD
4683							
4684	023254	005724		20\$:	TST	(R4)+	;CHECK ZERO FILL OF PATTERN
4685	023256	001012			BNE	30\$	;MISCOMPARE ERROR
4686	023260	005366	000002		DFC	2(SP)	;DECREMENT WORD COUNT
4687	023264	001416			BEQ	40\$	;NO MORE COMPARISONS
4688	023266	005316			DFC	(SP)	;DECREMENT NUMBER OF SOFT
4689							; DATA COMPARES
4690	023270	001414			BEQ	40\$	;NO MORE COMPARISONS
4691	023272	005202			INC	R2	;INCREMENT OFFSET VALUE
4692	023274	022702	000400		CMP	#256.,R2	;CHECK IF COMPLETE SECTOR COMPARES
4693	023300	001674			BEQ	10\$	;YES, START NEXT SECTOR
4694	023302	000764			BR	20\$	;COMPARE NEXT WORD ON
4695							
4696	023304	110265	000217	30\$:	MOVP	R2,P.FRAG(R5)	;STORE OFFSET ADDRESS
4697	023310	022626			CMP	(SP)+,(SP)+	;ADJUST STACK
4698	023312	017666	000012 000012		MOV	#12(SP),12(SP)	;LOAD ERROR RETURN
4699	023320	000404			BR	4\$	;RETURN
4700							
4701	023322	022626		40\$:	CMP	(SP)+,(SP)+	;ADJUST STACK
4702	023324	062766	000002 000012		ADD	#2,12(SP)	;ADJUST RETURN
4703							
4704	023332	105065	000117	45\$:	CLRB	P.ECMP(R5)	;CLEAR ECC COMPARE FLAG
4705	023336	012600			MOV	(SP)+,R0	;RESTORE R0
4706	023340	012601			MOV	(SP)+,R1	;RESTORE R1
4707	023342	012602			MOV	(SP)+,R2	;RESTORE R2
4708	023344	012603			MOV	(SP)+,R3	;RESTORE R3
4709	023346	012604			MOV	(SP)+,R4	;RESTORE R4
4710	023350	000207			RTS	PC	;RETURN
4711							
4712	023352	005701		50\$:	TST	R1	;CHECK IF ALL ZEROS
4713	023354	001432			BFQ	52\$	;YES, TEST FOR ZEROES

4714	023356	021314			CMP	(R3),(R4)	;CHECK FIRST WORD OF PATTERN
4715	023360	001351			BNE	30\$	;MISCOMPARE, REPORT ERROR
4716	023362	076601	000002		CMP	2(SP),R1	;CHECK IF ENOUGH WORDS READ FOR
4717							; SECONDARY COMPARISON
4718	023366	103443			BLO	55\$	;NO, DO SAMPLE COMPARE
4719	023370	010146			MOV	R1,-(SP)	;SAVE SECTOR COUNT
4720	023372	005301			DEC	R1	;CALCULATE NUMBER OF WORDS FROM START
4721	023374	006301			ASL	R1	;CONVERT TO BYTES
4722	023376	060104			ADD	R1,R4	;CALCULATE MEMORY ADDRESS
4723	023400	042701	177700		BIC	#177700,R1	;GET PATTERN ADDRESS OFFSET
4724	023404	060103			ADD	R1,R3	;CALCULATE PATTERN ADDRESS
4725	023406	021324			CMP	(R3),(R4)+	;CHECK IF DATA WORD IS CORRECT
4726	023410	001403			BEQ	51\$	;YES, CHECK ZERO FILL
4727	023412	005216			INC	(SP)	;CALCULATE OFFSET
4728	023414	012602			MOV	(SP)+,R2	;STORE OFFSET
4729	023416	000732			BR	30\$	;REPORT ERROR
4730							
4731	023420	012602		51\$:	MOV	(SP)+,R2	;STORE OFFSET
4732	023422	062702	000002		ADD	#2,R2	;CALCULATE OFFSET FOR START OF ZERO FILL
4733	023426	022702	000377		CMP	#377,R2	;CHECK IF NO ZERO FILL
4734	023432	103421			BLO	55\$	;YES, DO SAMPLE COMPARE
4735	023434	026602	000002		CMP	2(SP),R2	;CHECK WORD WAS READ
4736	023440	103416			BLO	55\$	;NO, DO SAMPLE COMPARE
4737	023442	005714		52\$:	TST	(R4)	;CHECK START OF ZERO FILL
4738	023444	001317			BNE	30\$	;NO, REPORT ERROR
4739	023446	022766	000377 000002		CMP	#377,2(SP)	;CHECK IF LAST WORD READ
4740	023454	101010			BHI	55\$	;NO, DO SAMPLE COMPARE
4741	023456	012702	000377		MOV	#377,R2	;LOAD OFFSET COUNT
4742	023462	016504	000214		MOV	P.ERHD(R5),R4	;LOAD START OF SECTOR
4743	023466	062704	000776		ADD	#776,R4	;CALCULATE END OF SECTOR
4744	023472	005714			TST	(R4)	;CHECK LAST WORD OF SECTOR
4745	023474	001303			BNE	30\$	;REPORT ERROR
4746							
4747	023476	016504	000214	55\$:	MOV	P.ERHD(R5),R4	;SET START OF SECTOR
4748	023502	012400			MOV	(R4)+,R0	;GET PATTERN INDEX
4749	023504	042700	177760		BIC	#177760,R0	;MASK OFF UNNECESSARY BIT
4750	023510	006300			ASL	R0	;MULTIPLY BY 2
4751	023512	016003	004576		MOV	PATTAR(R0),R3	;GET PATTERN ADDRESS
4752	023516	012401			MOV	(R4)+,R1	;GET SECTOR WORD COUNT
4753	023520	042701	177400		BIC	#177400,R1	;MASK OFF UNNECESSARY BITS
4754	023524	012702	000002		MOV	#2,R2	;INITIALIZE OFFSET
4755	023530	012700	000007		MOV	#7,R0	;LOAD NUMBER OF SAMPLES
4756	023534	162701	000043	57\$:	SUB	#35.,R1	;SUBTRACT 35 FROM SECTOR WORD COUNT
4757	023540	101430			BLOS	60\$	;CHECK IF ZERO FILL
4758	023542	162766	000043 000002		SUB	#35.,2(SP)	;SUBTRACT 35 FROM WORD COUNT
4759	023550	101664			BLOS	40\$	;CHECK IF FINISHED
4760	023552	062702	000043		ADD	#35.,R2	;CALCULATE OFFSET
4761	023556	062704	000106		ADD	#106,R4	;DETERMINE ADDRESS OF COMPARE
4762	023562	062703	000006		ADD	#6,R3	;DETERMINE ADDRESS OF PATTERN
4763	023566	021413			CMP	(R4),(R3)	;CHECK IF DATA CORRECT
4764	023570	001245			BNE	30\$	;NO, REPORT ERROR
4765	023572	005316			DEC	(SP)	;DECREMENT NUMBER OF COMPARES
4766	023574	001652			BFC	40\$	;CHECK IF FINISHED
4767	023576	005300			DEC	R0	;CHECK IF IF TIME FOR NEXT SECTOR
4768	023600	001355			BNE	57\$	;NO, CHECK NEXT SAMPLE
4769	023602	162766	000011 000002		SUB	#11,2(SP)	;ADJUST WORD COUNT

4770	023610	101644			BLOS	40\$	;CHECK IF FINISHED
4771	023612	062704	000022		ADD	#22,R4	;CALCULATE START OF NEXT SECTOR
4772	023616	000137	023072		JMP	10\$	;GO SERVICE NEXT SECTOR
4773							
4774	023622	162766	000043	000002 60\$:	SUB	#35-,2(SP)	;SUBTRACT 3* FROM WORD COUNT
4775	023630	101634			BLOS	40\$	;CHECK IF FINISHED
4776	023632	062702	000043		ADD	#35-,R2	;DETERMINE OFFSET
4777	023636	062704	000106		ADD	#106,R4	;DETERMINE ADDRESS OF COMPARE
4778	023642	005714			TEST	(R4)	;CHECK IF DATA CORRECT
4779	023644	001217			BNE	30\$	;NO, REPORT ERROR
4780	023646	005316			DFC	(SP)	;DECREMENT NUMBER OF COMPARES
4781	023650	001624			BEQ	40\$	;BRANCH IF FINISHED
4782	023652	005300			DFC	R0	;CHECK IF SAMPLE COUNT EXHAUSTED
4783	023654	001362			BNE	60\$	;NO, CHECK NEXT SAMPLE
4784	023656	162766	000011	000002	SUB	#11,2(SP)	;ADJUST WORD COUNT
4785	023664	101616			BLOS	40\$	;CHECK IF FINISHED
4786	023666	062704	000022		ADD	#22,R4	;CALCULATE START OF NEXT SECTOR
4787	023672	000137	023072		JMP	10\$	;GO SERVICE NEXT SECTOR

```

4788 .SBTTL NORMAL COMMAND RETURN
4789
4790 ;*****
4791 ;*
4792 ;* THIS ROUTINE WILL HANDLE ALL NORMAL COMMAND TERMINATIONS
4793 ;*
4794 ;* CALLED ROUTINE USE
4795 ;* -----
4796 ;*
4797 ;* ASNORM ASSIGNMENT NORMAL TERMINATION
4798 ;* CNTRL1 CONTROLLER ERROR
4799 ;* ABNORM DRIVE ERROR
4800 ;* EVCVY1 NORMAL ERROR RECOVERY
4801 ;* DATERR INCORRECT DATA BUFFER
4802 ;*
4803 ;* ASSUMED REGISTERS
4804 ;*
4805 ;* REGISTER CONTENTS
4806 ;* -----
4807 ;*
4808 ;* R5 ADDRESS OF PARAMETER BLOCK
4809 ;* R2 ADDRESS OF RK06 REGISTERS
4810 ;*
4811 ;*****
4812
4813 023676 004737 046664 NORMAL: JSR PC,I.CSTS ;GATHER CONTROLLER STATUS
4814 023702 032765 024000 000016 BIT #SPARICTO,P.CS1(R5) ;CHECK IF DRIVE BUS PARITY OR CONTROLLER TYPE C01
4815 073710 001010 BNE 2$ ;YES, INDICATE CONTROLLER FAULT
4816
4817 ; CHECK IF ANY OF THE FOLLOWING ERRORS ARE SET IN CS2
4818 ; UNIT FIELD ERROR
4819 ; MULTIPLE DRIVE SELECT
4820 ; PROGRAMMING ERROR
4821 ; NON-EXISTENT MEMORY
4822 ; NON-EXISTENT DRIVE
4823 ; UNIBUS PARITY ERROR
4824 ; WRITE CHECK ERROR
4825 ; DATA LATE
4826 023712 032765 177400 000020 BIT #UFEINDSIPGEINEMINEDIUPEINCFIDLT,F.CS2(R5)
4827 023720 001004 BNE 2$ ;YES, INDICATE CONTROLLER ERROR
4828
4829 ; CHECK IF ANY OF THE FOLLOWING ERRORS ARE SET IN RRER
4830 ; ILLFCAL PUNCTION CODE
4831 ; FORMAT ERROR
4832 ; DRIVE TYPE ERROR
4833 ; ECC HARD
4834 ; BAD SFCTCP ERROR
4835 ; HEADER VRC ERROR
4836 ; CYLINDER ADDRESS OVERFLOW
4837 ; DRIVE TIMING ERROR
4838 ; OPERATION INCMPLTF
4839 ; DATA CHECK
4840 023722 032765 131761 000034 BIT #ILCIPMTEIDTYFIECHIRSEIRVRCICCEIDTEIDPIIDCK,P.ER(R5)
4841 023730 001405 BEQ 3$ ;NO, CHECK FOR UNREPORT DRIVE ERRORS
4842 023732 052737 000001 001560 2$: BIS #BIT0,ERCCNT ;SET CONTROLLER ERROR NOT FLAGGED
4843 023740 000137 026426 JMP CNTRL1 ;REPORT CONTROLLER ERROR

```

H10

```

4844
4845      ;      CHECK THE FOLLOWING ERROR BITS IN RKEP(UNREPORTED DRIVE ERROR)
4846      ;      SEEK INCOMPLETE
4847      ;      NON-EXECUTABLE DRIVE FUNCTION
4848      ;      DRIVE DETECTED DRIVE BUS PARITY ERROR
4849      ;      INVALID DISK ADDRESS
4850      ;      WRITE LOCK ERROR
4851      ;      UNSAFE
4852 023744 032765 046016 000034 3S:  BIT  #SKIINXFICRPARIDAEIABLEIUNS,P.EF(R5)
4853 023752 001004      BNE  4S      ;YES,SET UNREPORTED DRIVE ERROR
4854
4855      ;      CHECK THE FOLLOWING ERROR BITS IN RKDS
4856      ;      AC LOW
4857      ;      SPEED LOSS
4858      ;      DRIVE OFF TRACK
4859 023754 032765 000070 000036  BIT  #ACLOISPDLSIDRGT,P.EC(R5)
4860 023762 001405      BEQ  5S      ;NO, CHECK IF SPECIAL SEQUENCE
4861 023764 052765 000010 000212 4S:  BIS  #BIT3,P.ERR(R5) ;SET DRIVE ERROR NOT REPORTED
4862 023772 000137 024766      JMP  ARNORM ;PROCESS ERROR
4863
4864 023776 032765 000001 000014 5S:  BIT  #DRVUSE,P.PRST(R5) ;CHECK IF DRIVE IN USE
4865 024004 001005      BNE  6S      ;YES, CHECK IF SPECIAL SEQUENCE
4866 024006 052765 000200 000212  BIS  #BIT7,P.ERR(R5) ;SET UNSOLICITED ATTENTION
4867 024014 000137 024766      JMP  ARNORM ;PROCESS ERROR
4868
4869 024020 005737 001560      6S:  TST  ERCCNT ;CHECK IF CONTROLLER ERROR HAS OCCURRED
4870 024024 100017      BPL  11S     ;NO, CONTINUE NORMAL PROCESSING
4871 024026 122765 000121 000001  CMPB #RDDATA,P.CMND(R5) ;CHECK IF DATA TRANSFER COMMAND
4872 024034 101007      BFI  7S      ;NO, REISSUE COMMAND
4873 024036 132765 000040 000001  BITP #BIT5,P.CMND(R5) ;CHECK IF SPECIAL COMMAND
4874 024044 001003      BNE  7S      ;YES, REISSUE COMMAND
4875 024046 004037 024506      JSR  RO,CHKADD ;CHECK BUS AND DISK ADDRESS
4876 024052 024062      10S     ;ERROR RETURN
4877 024054 004037 050706      7S:  JSR  RO,Q.PUSH ;ENQUEUE PARAMETER BLOCK IN
4878 024060 001324      CINITQ ; COMMAND INITIATION QUEUE
4879 024062 000207      10S:  RTS   PC ;RETURN
4880
4881 024064 005765 000150      11S:  TST  P.DSTT(R5) ;CHECK IF RETRY SEQUENCE
4882 024070 001417      BFQ  32S     ;NO, CHECK IF DRIVE IS BEING ASSIGNED
4883 024072 004737 037464      JSR  PC,ERRCV1 ;GO THROUGH ERROR RECOVERY
4884 024076 005737 001562      20S:  TST  ERRPRO ;CHECK IF ERROR PROCESSING COMPLETE
4885 024102 001367      BNE  10S     ;NO, RETURN
4886 024104 004037 050766      JSR  RO,Q.POP ;GET NEXT ERROR TO BE PROCESSED
4887 024110 001330      ERRPRO
4888 024112 012605      MOV  (SP)+,R5 ;LOAD ADDRESS OF PARAMETER BLOCK FOR
4889      ; NEXT ERROR
4890 024114 001762      BFQ  10S     ;IF NO MORE ERRORS, RETURN
4891 024116 010537 001562      MOV  R5,FRRPRC ;INDICATE THAT ERROR IS BEING PROCESSED
4892 024122 004737 031224      JSR  PC,ERRCV ;GO THROUGH ERROR RECOVERY
4893 024126 000763      BP  20S     ;CHECK IF ERROR PROCESSING COMPLETE
4894
4895 024130 105765 000210      32S:  TSTB P.ASSN(R5) ;CHECK IF DRIVE IS BEING ASSIGNED
4896 024134 001402      BFQ  33S     ;NO, CHECK IF SEEK OR RELEASE
4897 024136 000137 013770      JMP  ARNORM ;GO CONTINUE ASSIGNMENT
4898
4899 024142 122765 000117 000001 33S:  CMPB #SEEK,P.CMND(R5) ;CHECK IF SEEK TO PREVIOUS POSITION

```


4956	024412	016546	000170		MOV	P.NRD+4(R5),-(SP)	
4957	024416	066566	000160	000002	ADD	P.NWRT+2(R5),2(SP)	;ADD NUMBER OF WORDS WRITTEN
4958	074424	005516			ADC	(SP)	
4959	024426	066516	000162		ADD	P.NWRT+4(R5),(SP)	
4960	024432	103410			BCS	30\$	;DROP DRIVE IF OVERFLOW
4961	024434	026526	000114		CMP	P.NWRT+2(R5),(SP)+	;CHECK IF MAXIMUM NUMBER OF WORDS
4962							; HAVE BEEN TRANSFERRED
4963	024440	101010			BHI	32\$	;NO, PUT PARAMETER BLOCK IN
4964							; AVAILABLE QUEUE
4965	024442	103405			BLO	31\$	;YES, DROP DRIVE
4966	024444	026526	000112		CMP	P.NWRT(R5),(SP)+	;CHECK IF MAXIMUM NUMBER OF WORDS
4967							; HAVE BEEN TRANSFERRED
4968	024450	103753			BLO	28\$	;YES, DROP DRIVE
4969	024452	000404			BR	33\$	;NO, PUT PARAMETER BLOCK IN
4970							; AVAILABLE QUEUE
4971							
4972	024454	005726			30\$: TST	(SP)+	;ADJUST STACK
4973	024456	005726			31\$: TST	(SP)+	;ADJUST STACK
4974	024460	000747			BR	28\$	;DROP DRIVE
4975							
4976	024462	005726			32\$: TST	(SP)+	;ADJUST STACK
4977	024464	112765	000140	000001	33\$: MOVR	#RELEAS,P.CMND(R5)	;PUT RELEASE COMMAND IN PARAMETER BLOCK
4978	024472	004037	050706		JSR	RO,Q.PUSH	;ENQUEUE PARAMETER BLOCK IN
4979	024476	001324			CINITO		; COMMAND INITIATION QUEUE
4980							
4981	024500	004737	022016		35\$: JSR	PC,GETBUF	;GET NEW BUFFER
4982	024504	000207			RTS	PC	;RETURN

```

4983          .SBTTL CHECK BUS ADDRESS, WORD COUNT, SECTOR, TRACK, AND CYLINDER
4984
4985 024506 005765 000022          CHKADD: TST      P.WCR(R5)      ;CHECK IF WORD COUNT ZERO
4986 024512 001404          BFG      6$              ;YES, CHECK BUS ADDRESS
4987 024514 052737 000002 001560  BTS      #BIT1,EPCCNT      ;SET WORD COUNT NOT EQUAL ZERO
4988 024522 000512          BR       24$              ;REPORT ERROR
4989
4990 024524 032765 001400 000016 6$:  BIT      #RA16|BA17,P.CS1(R5) ;CHECK IF HIGH BUS ADDRESS BITS SET
4991 024532 001011          BNE      10$              ;YES, REPORT ERROR
4992 024534 016546 000012          MOV      P.WC(R5),-(SP) ;STORE WORD COUNT
4993 024540 005416          NEG      (SP)              ;MAKE IT POSITIVE
4994 024542 006316          ASL      (SP)              ;DETERMINE NUMBER OF BITS
4995 024544 066516 000010          ADD      P.BALO(R5),(SP) ;DETERMINE EXPECTED BUS ADDRESS
4996 024550 026526 000024          CMP      P.BAR(R5),(SP)+ ;CHECK BUS ADDRESS
4997 024554 001404          BEQ      17$              ;BUS ADDRESS CORRECT--CHECK CYLINDER,
4998                                     ;TRACK, AND SECTOR
4999 024556 052737 000004 001560 10$:  BTS      #BIT2,EPCCNT      ;SET BUS ADDRESS INCORRECT
5000 024564 000471          BR       24$              ;REPORT ERROR
5001
5002 024566 016546 000012          17$:  MOV      P.WC(R5),-(SP) ;STORE WORD COUNT
5003 024572 005416          NEG      (SP)              ;MAKE IT POSITIVE
5004 024574 105716          TSTB     (SP)              ;CHECK IF EVEN NUMBER OF SECTORS
5005 024576 001402          BEQ      18$              ;YES, DO COMPARISON
5006 024600 105266 000001          INCR     1(SP)              ;INCREMENT SECTOR COUNT
5007 024604 005046          18$:  CLR      -(SP)              ;MAKE ROOM ON STACK
5008 024606 116616 000003          MOVB     3(SP),(SP)      ;STORE NUMBER OF SECTORS TRANSFERRED
5009 024612 005066 000002          CLR      2(SP)              ;CLEAR LOCATION ON STACK
5010 024616 116566 000004 000002  MOVR     P.SECT(R5),2(SP) ;STORE STARTING SECTOR
5011 024624 066616 000002          ADD      2(SP),(SP)      ;DETERMINE FINAL SECTOR ADDRESS
5012 024630 005066 000002          CLR      2(SP)              ;CLEAR NUMBER OF TRACKS TRANSFERRED
5013
5014 024634 022716 000026          19$:  CMP      #22,(SP)              ;CHECK FOR SECTOR OVERFLOW
5015 024640 101005          BHI      20$              ;NO, CHECK IF SECTOR CORRECT
5016 024642 162716 000026          SUB      #22,(SP)              ;DECREMENT SECTOR COUNT BY 22 AND
5017 024646 005266 000002          INC      2(SP)              ; INCREMENT TRACKS TRANSFERRED
5018 024652 000770          BR       19$              ;CHECK FOR SECTOR OVERFLOW
5019
5020 024654 176526 000026          20$:  CMPB     P.DTS(R5),(SP)+ ;CHECK IF FINAL SECTOR CORRECT
5021 024660 001027          BNE      23$              ;NO, REPORT ERROR
5022 024662 005046          CLR      -(SP)              ;MAKE ROOM FOR TRACKS TRANSFERRED
5023 024664 116516 000005          MOVR     P.TRCK(R5),(SP) ;STORE STARTING TRACK
5024 024670 066616 000002          ADD      2(SP),(SP)      ;DETERMINE FINAL TRACK ADDRESS
5025 024674 005066 000002          CLR      2(SP)              ;CLEAR FINAL CYLINDER
5026
5027 024700 122716 000003          21$:  CMPB     #3,(SP)              ;CHECK FOR TRACK OVERFLOW
5028 024704 101005          BHI      22$              ;NO, CHECK FINAL TRACK
5029 024706 162716 000003          SUB      #3,(SP)              ;DECREMENT TRACK COUNT BY 3 AND
5030 024712 005266 000002          INC      2(SP)              ; INCREMENT CYLINDER COUNT
5031 024716 000770          BR       21$              ;CHECK FOR TRACK OVERFLOW
5032
5033 024720 126526 000027          22$:  CMPB     P.DTS+1(R5),(SP)+ ;CHECK IF FINAL TRACK CORRECT
5034 024724 001005          BNE      23$              ;TRACK INCORRECT ERROR
5035 024726 066516 000002          ADD      P.CYL(R5),(SP) ;CALCULATE FINAL CYLINDER
5036 024732 026516 000030          CMP      P.DCYL(R5),(SP) ;CHECK IF CYLINDER CORRECT
5037 024736 001410          BFG      2$              ;CORRECT GO TO NORMAL RETURN
5038 024740 005726          23$:  TST      (SP)+ ;ADJUST STACK

```

5039	024742	052737	000010	001560		BIS	#BIT3,ERCONT	;SET INCORRECT SECTOR, TRACK, OR CYLINDER
5040	024750	004737	026426		24\$:	JSR	PC,CNTRL1	;REPORT ERROR
5041	024754	011000				MOV	(R0),R0	;LOAD R0 FOR ESRCH RETURN
5042	024756	000200				RTS	R0	;RETURN
5043								
5044	024760	005726			25\$:	TST	(SP)+	;ADJUST STACK
5045	024762	005720				TST	(R0)+	;ADJUST R0
5046	024764	000200				RTS	R0	;NORMAL RETURN

```

5047 .SDBTL ABNORMAL TERMINATION
5048
5049 074766 004737 051126 ABNORM: JSR PC,C.PMCV ;REMOVE PARAMETER BLOCK FROM COMMAND
5050 ; INITIATION QUEUE
5051 024772 032765 000100 000014 BIT #CMDTO,P.PRST(R5) ;CHECK IF COMMAND TIME OUT
5052 075000 001402 BEQ 3$ ;NO, PROCESS ABNORMAL TERMINATION
5053 025002 000137 025172 JMP TIMEOUT ;JUMP TO TIME OUT ROUTINE
5054
5055 025006 005737 001560 3$: TST ERCONT ;CHECK IF CONTROLLER ERROR HAS OCCURRED
5056 075017 100010 BPL 10$ ;NO, CONTINUE ERROR PROCESSING
5057 025014 032765 000001 000014 BIT #DRVUSE,P.PRST(R5) ;CHECK FOR UNSOLICATED INTERRUPT
5058 025022 001403 BEQ 5$ ;YES, DO NOT ENQUEUE PARAMETER BLOCK
5059 025024 004037 050706 JSR RO,Q.PUSH ;ENQUEUE PARAMETER BLOCK IN
5060 075030 001324 CINITO ; COMMAND INITIATION QUEUE
5061 025032 000207 5$: RTS PC ;RETURN
5062
5063 075034 032765 000001 000014 10$: BIT #DRVUSE,P.PRST(R5) ;CHECK IF DRIVE IN USE
5064 025042 001004 BNE ABN000 ;YES, CHECK IF DRIVE BEING ASSIGNED
5065 025044 052765 000200 000217 BIS #BIT7,P.LPR(R5) ;INDICATE UNSOLICATED INTERRUPT
5066 025052 000420 BP ABN001 ;CHECK IF ERROR IS BEING PROCESSED
5067
5068 025054 005765 000150 ABN000: TST P.DSTT(R5) ;CHECK IF CURRENTLY UNFLAGGING
5069 ; ERROR PROCESSING
5070 075060 001415 BEQ ABN001 ;NO, CHECK IF ERROR PRESENTLY BEING PROCESSED
5071 075067 100402 BMI 13$ ;ERROR ENQUEUED
5072 075064 000137 042262 JMP ERGVY2 ;ERROR RECOVERY ROUTINE
5073
5074 025070 052765 000100 000212 13$: BIS #BIT6,P.ERR(R5) ;SET ERROR WHILE ENQUEUED IN ERROR QUEUE
5075 075076 000207 RTS PC ;RETURN
5076
5077 025100 013737 001450 177776 DATERR: MOV RKPPI,PS ;LOCK RF06 INTERRUPTS
5078 075106 052765 000004 000212 BIS #BIT2,P.LPR(R5) ;SET SOFTWARE DETECTED DATA ERROR
5079
5080 075114 012765 100200 000150 ABN001: MOV #BIT1<BIT7,P.DSTT(R5) ;SET ERROR ENQUEUED AND FIRST ERROR
5081 025122 005737 001562 TST ERKPRO ;CHECK IF ERROR IS BEING PROCESSED
5082 025126 001015 BNE 5$ ;YES, GO ENQUEUE ERROR
5083 075130 010537 001567 2$: MOV R5,ERRPRO ;LOAD ERROR BEING PROCESSED
5084 025134 004737 031224 JSR PC,FRCVY ;START ERROR RECOVERY
5085 025140 005737 001562 TST ERKPRO ;CHECK IF ERROR PROCESSING THROUGH
5086 025144 001005 BNE 3$ ;NO, RETURN
5087 075146 004037 050766 JSR RO,Q.POP ;GET NEXT ERROR ENQUEUED
5088 025152 001330 ERKPRO
5089 025154 012605 MOV (SP)+,R5 ;LOAD R5 FOR ERROR PROCESSING
5090 025156 001364 BNE 2$ ;IF NOT ZERO PROCESS NEXT ERROR
5091 075160 000207 3$: RTS PC ;RETURN
5092
5093 025162 004037 050706 5$: JSR RO,Q.PUSH ;PUT PARAMETER BLOCK ON
5094 075166 001330 ERKPRO ; ERROR PROCESSING QUEUE
5095 075170 000207 RTS PC ;RETURN

```

```

5096          .SBTTL  COMMAND TIME OUT ROUTINE
5097
5098 025172 016246 000010          TIMEOUT: MOV      RKCS2(R2),-(SP) ;GET CS7 FOR DRIVE NUMBER
5099 025176 042716 177770          BIC      #C<DRVMSK>,(SP) ;KEEP DRIVE NUMBER
5100 025202 122665 000000          CMPR     (SP)+,P.DRVN(F5) ;CHECK IF TIME OUT IF FOR
5101                                     ; CURRENTLY SELECTED DRIVE
5102 025206 001014          BNE      SS      ;NO, TEST IF LOADING ON CONTROLLER ERROR
5103 025210 016237 000000 001406  MOV      RKCS1(R2),T.CS1 ;GET COMMAND AND STATUS REGISTER 1
5104 025216 032737 000001 001406  BIT      #GO,T.CS1 ;CHECK IF GO IS STILL SET
5105 025224 001405          BFC      SS      ;NO, TEST IF LOADING ON CONTROLLER ERROR
5106 025226 052737 040000 001460  BIS      #E.CMTO,E.COMT ;SET CONTROLLER TIMED OUT ON COMMAND
5107 025234 000137 026022          JMP      CONTRL ;GO REPORT CONTROLLER ERROR
5108
5109 025240 005737 001560          SS:      TST      ERCONT ;CHECK IF CYCLING ON CONTROLLER ERROR
5110 025244 100004          BPL      10$      ;NO, CHECK IF DRIVE IS CURRENTLY
5111                                     ; SEIZED BY CTREP PORT
5112 025246 004037 050706          JSR      RO,Q.PUSH ;REQUEUE COMMAND IN COMMAND
5113 025252 001324          CINITO ; INITIATION QUEUE
5114 025254 000207          RTS      PC      ;RETURN
5115
5116 025256 032765 010000 000014 10$: BIT      #DRVSZD,P.PRST(R5) ;CHECK IF DRIVE WAS SEIZED
5117 025264 001404          BEQ      12$      ;NO, CONTINUE
5118 025266 052765 000020 000212  BIS      #BIT4,P.ERR(R5) ;SET TIME OUT BECAUSE DRIVE SEIZED
5119 025274 000667          BR       ABN000 ;GO INITIATE ERROR PROCESSING
5120
5121 025276 032765 020000 000014 12$: BIT      #E.UNLD,P.PRST(R5) ;CHECK IF WAITING FOR HEADS TO RELOCAT
5122 025304 001404          BEQ      15$      ;NO, SET TIME OUT FOR POSITIONING
5123 025306 052765 000400 000212  BIS      #BIT8,P.ERR(R5) ;SET TIME OUT IN RELOADING HEADS
5124 025314 000657          BR       ABN000 ;GO INITIATE ERROR PROCESSING
5125
5126 025316 052765 000040 000212 15$: BIS      #BIT5,P.ERR(R5) ;SET TIME OUT DURING POSITIONING
5127 025324 000653          BR       ABN000 ;GO INITIATE ERROR PROCESSING

```

5128
5129
5130 025326 010346
5131 025330 010146
5132 025332 012703 001406
5133 025336 012701 001426
5134 025342 104401 063401
5135 025346 004737 025606
5136 025352 012701 001442
5137 025356 104401 063500
5138 025362 004737 025606
5139 025366 012601
5140 025370 012603
5141 025372 000207
5142
5143
5144
5145 025374 010346
5146 025376 104401 063560
5147 025402 016546 000036
5148 025406 004737 052144
5149 075412 104401 063567
5150 025416 010503
5151 025420 062703 000040
5152 025424 012346
5153 075426 004737 052144
5154 025432 104401 057551
5155 025436 012346
5156 025440 004737 052144
5157 025444 104401 063630
5158 025450 012346
5159 025452 004737 052144
5160 025456 104401 057551
5161 025462 012346
5162 025464 004737 052144
5163 025470 104401 063643
5164 025474 012346
5165 075476 004737 052144
5166 025502 104401 057551
5167 025506 012346
5168 025510 004737 052144
5169 075514 104401 063656
5170 025520 012346
5171 025522 004737 052144
5172 025526 104401 057551
5173 025532 012346
5174 025534 004737 052144
5175 025540 104401 001165
5176 025544 012603
5177 075546 000207

.SBTTL PRINT RK06 UNIBUS REGISTERS

PRIREG: MOV R3,-(SP) ;STORE R3 ON STACK
MOV R1,-(SP) ;STORE R1 ON STACK
MOV #T.CS1,R3 ;LOAD BEGINNING OF REGISTER STORAGE
MOV #T.DS,R1 ;LOAD END OF REGISTER STORAGE
TYPE ,ERR300 ;PRINT REGISTER NAMES
JSR PC,PRSTAT ;PRINT REGISTER CONTENTS
MOV #T.DB,R1 ;LOAD END OF REGISTER STORAGE
TYPE ,ERR301 ;PRINT REGISTER NAMES
JSR PC,PRSTAT ;PRINT REGISTER CONTENTS
MOV (SP)+,R1 ;RESTORE R1
MOV (SP)+,R3 ;RESTORE R3
RTS PC ;RETURN

.SBTTL PRINT DRIVE STATUS

PRDSTT: MOV R3,-(SP) ;STORE R3 ON STACK
TYPE ,ERR302 ;PRINT REGISTER NAMES
MOV P.DS(R5),-(SP) ;CONVERT DRIVE STATUS REG TO OCTAL
JSR PC,BINOCY
TYPE ,ERR303 ;PRINT HEADER
MOV #5,R3 ;DETERMINE START OF PRINT COT
ADD #P.A00,R3
MOV (R3)+,-(SP) ;PRINT MESSAGE A
JSR PC,BINOCY
TYPE ,BLANKS
MOV (R3)+,-(SP) ;PRINT MESSAGE B
JSR PC,BINOCY
TYPE ,ERR304 ;PRINT MESSAGE A
MOV (R3)+,-(SP) ;PRINT MESSAGE B
JSR PC,BINOCY
TYPE ,BLANKS
MOV (R3)+,-(SP) ;PRINT MESSAGE B
JSR PC,BINOCY
TYPE ,ERR305 ;PRINT MESSAGE A
MOV (R3)+,-(SP) ;PRINT MESSAGE B
JSR PC,BINOCY
TYPE ,BLANKS
MOV (R3)+,-(SP) ;PRINT MESSAGE B
JSR PC,BINOCY
TYPE ,ERR306 ;PRINT MESSAGE A
MOV (R3)+,-(SP) ;PRINT MESSAGE B
JSR PC,BINOCY
TYPE ,BLANKS
MOV (R3)+,-(SP) ;PRINT MESSAGE B
JSR PC,BINOCY
TYPE ,SCRLF
MOV (SP)+,R3 ;RESTORE R3
RTS PC ;RETURN

```

517*      .SBTTL PRINT CONTROLLER STATUS
5179
5180 025550 010346 PRISTT: MOV R3,-(SP) ;STORE R3 ON STACK
5181 025557 010146      MOV R1,-(SP) ;STORE R1 ON STACK
5182 025554 010503      MOV R5,R3 ;STORE PARAMETER BLOCK ADDRESS
5183 025556 010501      MOV R5,R1 ; FOR INDEX CALCULATIONS
5184 025560 062703 000016      ADD @P.CS1,R3 ;CALCULATE BEGINNING OF OUTPUT OCTAL
5185 025564 062701 000036      ADD @P.DS,R1 ;CALCULATE END OF OUTPUT OCTAL
5186 025570 104401 063401      TYPE ,ERP300 ;PRINT REGISTER NAMES
5187 025574 004737 025606      JSR PC,PRSTAT ;PRINT REGISTER CONTENTS
5188 025600 012601      MOV (SP)+,R1 ;RESTORE R1
5189 025602 012603      MOV (SP)+,R3 ;RESTORE R3
5190 025604 000207      RTS PC ;RETURN
5191
5192      .SBTTL PRINT REGISTER CONTENTS
5193
5194 025606 012346 PRSTAT: MOV (R3)+,-(SP) ;SAVE REGISTER VALUE FOR TYPE OUT
5195 025610 004737 052144      JSR PC,RINOCT ;CONVERT TO OCTAL
5196 025614 020103      CMP R1,R3 ;CHECK IF FINISHED
5197 025616 001403      BEQ 10$ ;YES, RETURN
5198 025620 104401 057551      TYPE ,BLANKS ;TYPE 2 BLANKS
5199 025624 000770      BR PRSTAT ;TYPE NEXT REGISTER
5200
5201 025626 104401 001165 10$: TYPE ,<CRLF> ;TYPE <CR><LF>
5202 025632 000207      RTS PC ;RETURN

```

```

5203          .SBTTL  ECC CORRECTION ROUTINE
5204
5205 025634 010446      ECCOR: MOV      R4,-(SP)      ;STORE R4 ON STACK
5206 025636 010346      MOV      R3,-(SP)      ;STORE R3 ON STACK
5207 025640 010146      MOV      R1,-(SP)      ;STORE R1 ON STACK
5208 025642 010046      MOV      R0,-(SP)      ;STORE R0 ON STACK
5209 025644 013703 001630  MOV      RTYBA,R3      ;LOAD BEGINNING OF SECTOR ADDRESS
5210 025650 013704 001632  MOV      RTYWC,R4      ;LOAD WORDS READ FROM SECTOR
5211 025654 006304      ASL      R4              ;MAKE IT BYTES
5212 025656 010346      MOV      R3,-(SP)      ;STORE ADDRESS OF START OF SECTOR
5213 025660 060416      ADD      R4,(SP)      ;CALCULATE END OF SECTOR
5214 025662 016501 000060  MOV      P.EPOS(R5),R1    ;SAVE POSITION INFORMATION
5215 025666 005301      DEC      R1              ;DECREMENT BIT COUNT BY ONE TO GET
5216                                     ; BIT ADDRESS
5217 025670 010100      MOV      R1,R0
5218 025672 006201      ASR      R1              ;DETERMINE WORDS THAT ARE BAD
5219 025674 006201      ASR      R1              ; FOR DATA READ (SHIFT 3 BITS
5220 025676 006201      ASR      R1              ; RIGHT)
5221 025700 042701 000001  BIC      #1,R1      ;CLEAR BYTE INDICATOR
5222 025704 020104      CMP      R1,R4      ;CHECK IF EACH WITHIN TRANSFER
5223 025706 103037      BHS      10$          ;NO, RETURN
5224 025710 060103      ADD      R1,R3      ;DETERMINE ADDRESS OF ERROR
5225 025712 016537 000062 001616  MOV      P.EPAT(R5),ECCPAT ;LOAD PATTERN
5226 025720 005037 001620      CLR      ECCPAT+2
5227 025724 042700 177760      BIC      #177760,R0    ;DETERMINE BIT POSITION FOR START OF ECC
5228                                     ; CORRECTION
5229 025730 001406      BEQ      5$            ;CORRECTION STARTS ON
5230                                     ; WORD BCUNLAFV
5231
5232 025732 006337 001616      3$:  ASL      ECCPAT      ;SHIFT PATTERN 1 BIT LEFT
5233 025736 006137 001620      ROL      ECCPAT+2
5234 025742 005300      DEC      R0              ;DECREMENT COUNT
5235 025744 001372      BNE      3$            ;CHECK IF IN POSITION
5236
5237 025746 011300      5$:  MOV      (R3),R0      ;CORRECT FIRST WORD WITH EXCLUSIVE
5238 025750 013701 001616      MOV      ECCPAT,R1      ; OF BITS IN ERROR
5239 025754 043713 001616      BIC      ECCPAT,(R3)
5240 025760 040001      BIC      R0,R1
5241 025762 050123      BIS      R1,(R3)+
5242 025764 021603      CMP      (SP),R3      ;CHECK IF SECCAD WORD IN TRANSFER
5243 025766 001407      BEQ      10$          ;NO, RETURN
5244 025770 011300      MOV      (R3),R0      ;CORRECT SECOND WORD WITH EXCLUSIVE
5245 025772 013701 001620      MOV      ECCPAT+2,R1      ; OF BITS IN ERROR
5246 025776 043713 001620      BIC      ECCPAT+2,(R3)
5247 026002 040001      BIC      R0,R1
5248 026004 050113      BIS      R1,(R3)
5249
5250 026006 005726      10$: TST      (SP)+      ;ADJUST STACK
5251 026010 012600      MOV      (SP)+,R0      ;RESTORE R0
5252 026012 012601      MOV      (SP)+,R1      ;RESTORE R1
5253 026014 012603      MOV      (SP)+,R3      ;RESTORE R3
5254 026016 012604      MOV      (SP)+,R4      ;RESTORE R4
5255 026020 000207      RTS      PC              ;RETURN
  
```

.SBTTL CONTROLLER ERROR DETECTED BY RK06 DRIVER

5256									
5257									
5258	026022	010446			CONTRL: MOV	R4, -(SP)		;STORE R4 ON STACK	
5259	026024	004737	046536		JSR	PC, I.STOR		;GET CONTROLLER STATUS	
5260	026030	032777	002000	153102	BIT	#SW10, #SWP		;CHECK IF BELL CH ERROR	
5261	026036	001402			BEQ	1\$		;NO, CONTINUE	
5262	026040	104401	001160		TYPE	, \$BELL		;RING BELL	
5263	026044	013704	001460		1\$: MOV	E.CONT, R4		;STORE CONTROLLER INDICATORS	
5264	026050	032704	100000		BIT	#E.WDS, R4		;CHECK IF DUPLICATE DRIVE SELECT	
5265	026054	001403			BEQ	2\$		;NO, CHECK FOR OTHER ERRORS	
5266	026056	104401	060023		TYPE	, ERR006		;TYPE "MULTIPLE DRIVE SELECT"	
5267	026062	000457			BP	10\$		;DISPLAY REGISTERS	
5268									
5269	026064	032704	000001		2\$: BIT	#E.CCLR, R4		;CHECK IF "CLEAR CONTROLLER DID NOT	
5270								; CLEAR ERROR"	
5271	026070	001403			BEQ	3\$		;NO, CHECK OTHER CONTROLLER ERRORS	
5272	026072	104401	057601		TYPE	, ERR001			
5273	026076	000451			BP	10\$		;DISPLAY REGISTERS	
5274									
5275	026100	032704	000002		3\$: BIT	#E.WOAT, R4		;CHECK IF "NO ATTENTION IN REASON"	
5276	026104	001403			BEQ	4\$		;NO, CHECK THE OTHER CONTROLLER ERRORS	
5277	026106	104401	057646		TYPE	, ERR002			
5278	026112	000443			BP	10\$		;DISPLAY REGISTERS	
5279									
5280	026114	032704	000010		4\$: BIT	#E.UDAT, R4		;CHECK IF "UNEXPECTED DATA TYPE ERROR"	
5281	026120	001403			BEQ	5\$		;NO, CHECK OTHER CONTROLLER ERRORS	
5282	026122	104401	057675		TYPE	, ERR003			
5283	026126	000435			BR	10\$		;DISPLAY REGISTERS	
5284									
5285	026130	032704	000020		5\$: BIT	#E.CLAT, R4		;CHECK IF "ATTENTION DID NOT RESET	
5286								; WITH DRIVE CLEAR"	
5287	026134	001403			BEQ	6\$		;NO, CHECK OTHER CONTROLLER ERRORS	
5288	026136	104401	057730		TYPE	, ERR004			
5289	026142	000427			BR	10\$		;DISPLAY REGISTERS	
5290									
5291	026144	032704	000100		6\$: BIT	#E.ILLD, R4		;CHECK IF "ILLEGAL DRIVER COMMAND"	
5292	026150	001403			BEQ	7\$		;NO, CHECK IF DATA LATE WHEN UNLOADING HEADER	
5293	026152	104401	057774		TYPE	, ERR005			
5294	026156	000421			BR	10\$		;DISPLAY REGISTERS	
5295									
5296	026160	032704	000400		7\$: BIT	#E.DLT, R4		;CHECK IF "DATA LATE WHEN UNLOADING HEADER"	
5297	026164	001403			BEQ	8\$		;NO, CHECK IF "CONTROLLER ERROR WHILE GATEING	
5298								; STATUS"	
5299	026166	104401	062602		TYPE	, ERR200			
5300	026172	000413			BP	10\$		;DISPLAY REGISTERS	
5301									
5302	026174	032704	001000		8\$: BIT	#E.CERR, R4		;CHECK IF "CONTROLLER ERROR DURING	
5303								; DRIVE SERVICING"	
5304	026200	001403			BEQ	9\$		;NO, CHECK IF CONTROLLER TIME OUT	
5305	026202	104401	062642		TYPE	, ERR201			
5306	026206	000405			BP	10\$		;DISPLAY REGISTERS	
5307									
5308	026210	032704	040000		9\$: BIT	#E.CMTO, R4		;CHECK IF CONTROLLER TIME OUT	
5309	026214	001414			BEQ	20\$		;NO, CHECK IF DRIVE DETECTED DRIVE BUS PARITY ERROR	
5310	026216	104401	063127		TYPE	, ERR207			
5311									

```
5312 026222 104401 001165      10$: TYPE ,SCRLP ;TYPE <CH><LF>
5313 026226 004737 043072      JSR PC,PRITM ;PRINT TIME
5314 026232 004737 025326      JSR PC,PRIMEG ;PRINT CONTROLLER REGISTERS
5315 026236 104401 057554      TYPE ,ERR000 ;TYPE FATAL ERROR
5316 026242 000000             HALT ;HANG ON FATAL ERROR
5317 026244 000776             BR -2
5318
5319 026246 032704 002000      20$: BIT @E.DPAR,R4 ;CHECK IF "DRIVE DETECTED DRIVE BUS PARITY ERROR"
5320 026252 001001             BNE 30$ ;YES, SET DPAR IN ERROR REGISTER
5321 026254 000000             HALT
5322
5323 026256 012604             30$: MOV (SP)+,R4 ;RESTORE R4
5324 026260 052765 000010 000034 BIT @DRPAR,P.ER(R5) ;SET DRIVE DETECTED DRIVE BUS/OTTY PARITY
5325 026266 010346             MOV R3,-(SP) ;STORE R3 ON THE STACK
5326 026270 005037 001462      CLR 0,WAIT ;CLEAR WAITING FOR COMMAND COMPLETION
5327 026274 116503 000000      MOV@ P.DRVN(R5),R3 ;GET DRIVE NUMBER
5328 026300 005062 000026      CLR RKMR1(R2) ;CLEAR MAINTENANCE REGISTER 1
5329 026304 010762 000010      MOV R3,RKCS2(R2) ;LOAD DRIVE NUMBER
5330 026310 012762 000105 000000 MOV @CLEAR,RKCS1(R2) ;LOAD DRIVE CLEAR
5331 026316 105762 000000      35$: TSTR RKCS1(R2) ;WAIT FOR READY
5332 026322 001775             BEQ 35$
5333 026324 004737 046536      JSR PC,I.STCF ;STORE REGISTERS
5334 026330 032737 100000 001406 BIT @CERR,T.CS1 ;CHECK IF CONTROLLER ERROR
5335 026336 001413             BEQ 37$ ;NO, CHECK IF ATTENTION CLEARED
5336
5337 ; CHECK FOR CONTROLLER TYPE ERROR
5338 026340 032737 137400 001410 BIT @UPFINDSIPGEINEMINEDIOPFIDLT,T.CS2
5339 026346 001004             BNE 36$
5340 026350 032737 130761 001424 BIT @ILCIDTYEIPFTEIFCHIPS*INVRICDTEICPIIDCK,T.ER
5341 026356 001412             BEQ 40$
5342
5343 026360 104401 062642      36$: TYPE ,ERR201 ;TYPE CONTROLLER ERROR DURING DRIVE SERVICING
5344 026364 000716             BR 10$ ;GO REPORT ERROR
5345
5346 026366 136337 001510 001423 37$: BIT@ INTMSK(R3),T.ASOP+1 ;CHECK IF ATTENTION CLEARED
5347 026374 001403             BEQ 40$ ;YES, REPORT ERROR
5348 026376 104401 057730      TYPE ,ERR004 ;TYPE "DRIVE CLEAR DID NOT RESET ATTENTION"
5349 026402 000707             BR 10$ ;GO REPORT ERROR
5350
5351 026404 146337 001510 001506 40$: BICP INTMSK(R3),W.TIME ;CLEAR TIMING CURRENTLY ON THIS DRIVE
5352 026412 006303             ASL R3 ;MULTIPLY BY 2
5353 026414 005063 001540      CLR W.DRV(R3) ;CLEAR WATCH DOG TIME
5354 026420 012603             MOV (SP)+,R3 ;RESTORE R3
5355 026422 000137 024766      JMP ABNORM ;GO TO ABNORMAL RETURN
```

.SBTTL CONTROLLER ERROR DUE TO REGISTER INCONSISTANCIES

```
5356
5357
5358 026426 004737 051126          CNTRL1: JSR      PC,Q.PMOV      ;REMOVE PARAMETER BLOCK FROM THE
5359                                     ; COMMAND INITIATION QUEUE
5360 026432 032777 002000 152500    BIT      $SW10,$SWR      ;CHECK IF BELL CB ERROR
5361 026440 001402                   BEQ      1$              ;NO, CONTINUE
5362 026442 104401 001160           TYPE     ,SBPLL          ;RING BELL
5363 026446 032777 020000 152464 1$: BIT      $SW13,$SWR      ;CHECK IF INHIBIT PRINT OUT
5364 026454 001404                   BEQ      2$              ;NO PRINT ERROR MESSAGE
5365 026456 032777 040000 152454    BIT      $SW14,$SWR      ;CHECK IF CYCLE ON OPERATION
5366 026464 001050                   BNE     10$             ;YES, CHECK IF LCCP ON OPERATION
5367 026466 032737 000001 001560 2$: BIT      $BIT0,ERCONT    ;CHECK IF "CONTROLLER ERROR NOT FLAGGED"
5368 026474 001402                   BFO      3$
5369 026476 104401 060332           TYPE     ,ERP013
5370 026502 032737 000002 001560 3$: BIT      $BIT1,ERCONT    ;CHECK IF "WORD COUNT NOT 0"
5371 026510 001402                   BEQ      4$
5372 026512 104401 060371           TYPE     ,ERP014
5373 026516 032737 000004 001560 4$: BIT      $BIT2,ERCONT    ;CHECK IF "BUS ADDRESS INCORRECT"
5374 026524 001402                   BEQ      5$
5375 026526 104401 060422           TYPE     ,ERP015
5376 026532 032737 000010 001560 5$: BIT      $BIT3,ERCONT    ;CHECK IF "CYLINDER, TRACK, SECTOR
5377                                     ; INCORRECT"
5378 026540 001402                   BEQ      7$
5379 026542 104401 060446           TYPE     ,ERP016
5380 026546 032777 020000 152364 7$: BIT      $SW13,$SWR      ;CHECK IF INHIBIT PRINT OUT
5381 026554 001014                   BNE     10$             ;YES, CHECK IF HALT ON ERROR
5382 026556 004737 043072           JSR      PC,PRITIM        ;PRINT TIME
5383 026562 004737 025550           JSR      PC,PRISTT        ;PRINT RK06 REGISTERS
5384 026566 004737 027476           JSR      PC,PRIO02        ;PRINT CURRENT SUPPLIED COMMAND
5385 026572 032737 000001 001560    BIT      $BIT0,ERCONT    ;CHECK IF CONTROLLER ERROR
5386                                     ; NOT FLAGGED
5387 026600 001402                   BEQ      10$
5388 026602 004737 027320           JSR      PC,PRIO01        ;PRINT PREVIOUS COMMAND
5389 026606 032777 100000 152324 10$: BIT     $SW15,$SWR      ;CHECK IF HALT ON ERROR
5390 026614 001010                   BNE     11$
5391 026616 032777 040000 152314    BIT      $SW14,$SWR      ;CHECK IF LCCP OPERATION
5392 026624 001005                   BNE     15$
5393 026626 104401 057554           TYPE     ,ERP000          ;YES, CLEAR ERROR
5394 026632 104401 060051           TYPE     ,ERN007          ;PRINT "SET SWITCH 14 TO CYCLE CB ERROR"
5395 026636 000000                   HALT
5396 026640 012737 100000 001560 11$: MOV      $BIT15,ERCONT    ;SET CONTROLLER ERROR PRESENT
5397 026646 012762 100000 000000    MOV      $CLR,RKCS1(R2) ;CLEAR CONTROLLER
5398 026654 005062 000026           CLR      RKMR1(R2)        ;CLEAR MAINTENANCE REGISTER 1
5399 026660 112737 000005 001406    MOVB     $DR.CLR,T.CS1 ;LOAD COMMAND INTO TEMPORARY CS1
5400 026666 004037 046314           JSR      R0,I.ISSU        ;GC ISSUE DRIVE CLEAR
5401 026672 026702                   20$
5402 026674 004037 050706           JSR      R0,Q.PUSH        ;PUT PARAMETER BLOCK ON COMMAND
5403 026700 001324                   CINITQ   ; INITIATION QUEUE
5404 026702 000207                   20$: RTS      PC          ;RETURN
```

```

5405          .SBTTL PRINT WHOLE DATA BUFFER
5406
5407 026704 122765 000121 000123 PRIBUF: CMPB  #RDDATA,P.RCMD(R5) ;CHECK IF READ DATA
5408 076712 001017          BNE  5$          ;NO, RETURN
5409 026714 032765 000010 000150          BIT  #BIT3,P.DSTT(R5) ;CHECK IF TIME TO PRINT BUFFER
5410 026722 001013          BNE  5$          ;NO, RETURN
5411 026724 052765 000010 000150          BIS  #BIT3,P.DSTT(R5) ;SET BUFFER ATTEMPTED TO BE PRINTED
5412 026732 032777 020000 152200          BIT  #SW13,#SWR      ;CHECK IF INHIBIT PRINT OUT
5413 026740 001004          BNE  5$          ;YES, RETURN
5414 026742 032777 000010 152170          BIT  #SW3,#SWR      ;CHECK IF PRINT SECTOR BEFORE BEGINNING
5415          ; ERROR RECOVERY
5416 076750 001001          BNE  10$         ;YES, PRINT SECTOR IN ERROR
5417 026752 000207          5$: RTS  PC      ;RETURN
5418
5419 026754 010146          10$: MOV  R1,-(SP) ;STORE R1 ON THE STACK
5420 026756 010346          MOV  R3,-(SP) ;STORE R3 ON THE STACK
5421 026760 010446          MOV  R4,-(SP) ;STORE R4 ON THE STACK
5422 026762 104401 063312          TYPE ,ERR213 ;TYPE ADDRESS OF BEGINNING OF
5423          ; SECTOR IN ERROR
5424 026766 013703 001630          MOV  RTYBA,R3 ;GET ADDRESS OF START OF SECTOR
5425 026772 013704 001632          MOV  RTYWC,R4 ;GET WORD COUNT
5426 026776 010346          MOV  R3,-(SP) ;STORE STARTING ADDRESS FOR PRINT OUT
5427 027000 004737 052144          JSR  PC,RINOCY ;CONVERT OF CCTL
5428 027004 104401 063363          TYPE ,ERR214 ;TYPE DATA READ
5429 027010 012701 000010          15$: MOV  #8,R1 ;SET 8 COLUMNS ON PAGE
5430 027014 104401 001165          TYPE ,SCPLP ;TYPE <CR><LF>
5431 027020 012346          20$: MOV  (R3)+,-(SP) ;GET NEXT WORD OF SECTOR
5432 027022 004737 052144          JSR  PC,RINOCY ;CONVERT TO CCTL
5433 027026 005304          DEC  R4 ;CHECK IF FINISHED
5434 027030 001405          BEQ  30$         ;YES, RESTORE REGISTERS
5435 027032 005301          DEC  R1 ;CHECK IF END OF LINE
5436 027034 001765          BEQ  15$         ;YES, INITIALIZE COLUMN COUNT
5437 027036 104401 057551          TYPE ,BLANKS ;TYPE 2 BLANKS
5438 027042 000766          BR   20$         ;GET NEXT WORD
5439
5440 027044 104401 001165          30$: TYPE ,SCPLP ;TYPE <CR><LF>
5441 027050 012604          MOV  (SP)+,R4 ;RESTORE R4
5442 027052 012603          MOV  (SP)+,R3 ;RESTORE R3
5443 027054 012601          MOV  (SP)+,R1 ;RESTORE R1
5444 027056 000207          RTS  PC      ;RETURN

```

```

5445          .SBTTL PRINT DRIVE AND PACK SERIAL NUMBERS
5446
5447 027060 010146 PRISER: MOV R1,-(SP) ;STORE R1 ON STACK
5448 027062 010346      MOV R3,-(SP) ;STORE R3 ON STACK
5449 027064 010446      MOV R4,-(SP) ;STORE R4 ON STACK
5450 027066 012701 055667 MOV #STT100,R1 ;GET ADDRESS OF ASCII STRING
5451 027072 016503 000224 MOV P.SERL(R5),R3 ;GET DRIVE SERIAL NUMBER
5452 027076 006103      ROL R3 ;SHIFT DRIVE SERIAL NUMBER 4 BITS LEFT
5453 027100 006103      ROL R3
5454 027102 006103      ROL R3
5455 027104 006103      ROL R3
5456 027106 012704 000003 MOV #3,R4 ;LOAD NUMBER OF DIGITS OF SERIAL NUMBER
5457 027112 105011 25: CLRB (R1) ;CLEAR NEXT BYTE FOR PRINT OUT
5458 027114 006103      ROL R3 ;SHIFT IN NEXT 4 BITS
5459 027116 106111      ROLB (R1)
5460 027120 006103      ROL R3
5461 027122 106111      ROLB (R1)
5462 027124 006103      ROL R3
5463 027126 106111      ROLP (R1)
5464 027130 006103      ROL R3
5465 027132 106111      ROLB (R1)
5466 027134 152721 000060 BISR #60,(R1)+ ;CHANGE BCD TO DECIMAL
5467 027140 005304      DEC R4 ;CHECK IF FINISHED
5468 027142 001363      BNE 2* ;CONVERT NEXT DIGIT
5469 027144 104401 055644 TYPE ,STT000 ;TYPE "DRIVE SERIAL NUMBER"
5470 027150 016503 000226 MOV P.PKSR(P5),P3 ;GET LEAST SIGNIFICANT BITS OF
5471 ; PACK SERIAL NUMBER
5472 027154 016504 000230 MOV P.PKSP+2(R5),R4 ;GET MOST SIGNIFICANT BITS OF
5473 ; PACK SERIAL NUMBER
5474 027160 012701 056232 MOV #STT012,R1 ;GET ADDRESS OF ASCII STRING
5475 027164 105011      CLRB (R1) ;CLEAR FIRST BYTE OF PRINT OUT
5476 027166 000404      BR 75 ;START CONVERSION TO OCTAL
5477
5478 027170 105011 55: CLRB (P1) ;CLEAR NEXT BYTE OF PRINT OUT
5479 027172 006103      ROL R3 ;SHIFT IN NEXT THREE BITS
5480 027174 006104      ROL R4
5481 027176 106111      ROLB (R1)
5482 027200 006103 75: ROL R3
5483 027202 006104      ROL R4
5484 027204 106111      ROLP (R1)
5485 027206 006103      ROL R3
5486 027210 006104      ROL R4
5487 027212 106111      ROLP (R1)
5488 027214 152721 000060 BISR #60,(R1)+ ;MAKE IT ASCII
5489 027220 022701 056745 CMP #STT013,R1 ;CHECK IF FINISHED
5490 027224 101361      BHI 55 ;NO, CONTINUE CONVERSION
5491 027226 104401 056205 TYPE ,STT011 ;TYPE PACK SERIAL NUMBER
5492 027232 012604      MOV (SP)+,R4 ;RESTORE R4
5493 027234 012603      MOV (SP)+,R3 ;RESTORE R3
5494 027236 012601      MOV (SP)+,R1 ;RESTORE R1
5495 027240 000207      RTS PC ;RETURN

```

```

5496          .SBTTL PRINT BEGINNING OF ERROR HEADER
5497
5498 027242 032777 002000 151670 PRI000: BIT    #SW10,@SWP    ;CHECK IF BELL ON ERROR
5499 027250 001402          BEQ      1$          ;NO, CHECK IF INHIBIT PRINT OUT
5500 027252 104401 001160          TYPE    ,SBELL    ;RING BELL
5501 027256 032777 020000 151654 1$: BIT    #SW13,@SWP    ;CHECK IF INHIBIT PRINT OUT
5502 027264 001014          BNE      20$          ;YES, INCREMENT CONTROLLED ERROR
5503 027266 116537 000000 055446        MOVB    P.DRVN(R5),OPR011 ;STORE DRIVE NUMBER
5504 027274 152737 000060 055446        BISR    #60,OPR011    ;MAKE IT ASCII
5505 027302 104401 055427          TYPF    ,OPR010    ;TYPE "DRIVE NUMBER"
5506 027306 004737 027060          JSR      PC,PRISER    ;PRINT DRIVE AND PACK SERIAL NUMBERS
5507 027312 004737 043072          JSR      PC,PRITIM    ;PRINT TIME
5508 027316 000207          20$: RTS      PC          ;RETURN

```

```

5509          .SBTTL PRINT PREVIOUS COMMAND
5510
5511 027320 104401 062453 PRI001: TYPE ,ERR101 ;TYPE "PREVIOUS CMND CYL TRF SEC WRD CT"
5512 027324 122765 000121 000135 CMPB ,RDATA,P.LCMD(R5) ;CHECK IF READ DATA
5513 027332 001402 BEQ 5$ ;YES, DO NOT PRINT PATTERN NOW
5514 027334 104401 062535 TYPE ,ERR102 ;PRINT "PAT"
5515 027340 104401 001165 5$: TYPE ,SCRLF ;PRINT <CR><LF>
5516 027344 005046 CLR -(SP) ;MAKE ROOM ON STACK
5517 027346 116516 000135 MOVB P.LCMD(R5),(SP) ;GET LAST COMMAND
5518 027352 004737 052144 JSR PC,BINOC1 ;CONVERT TO OCTAL
5519 027356 104401 057551 TYPE ,BLANKS ;TYPE 2 BLANKS
5520 027362 016546 000136 MOV P.LCYL(R5),-(SP) ;GET LAST CYLINDER USED
5521 027366 004737 052144 JSR PC,BINOC1 ;CONVERT TO OCTAL
5522 027372 104401 057551 TYPE ,BLANKS ;TYPE 2 BLANKS
5523 027376 005046 CLR -(SP) ;MAKE ROOM ON STACK
5524 027400 116516 000141 MOVB P.LTRF(R5),(SP) ;GET LAST TRACK USED
5525 027404 004737 052144 JSR PC,BINOC1 ;CONVERT TO OCTAL
5526 027410 104401 057551 TYPE ,BLANKS ;TYPE 2 BLANKS
5527 027414 005046 CLR -(SP) ;MAKE ROOM ON STACK
5528 027416 116516 000140 MOVB P.LSEC(R5),(SP) ;GET LAST SECTOR USED
5529 027422 004737 052144 JSR PC,BINOC1 ;CONVERT TO OCTAL
5530 027426 104401 057551 TYPE ,BLANKS ;TYPE 2 BLANKS
5531 027432 016546 000142 MOV P.LWC(R5),-(SP) ;STORE LAST WORD COUNT
5532 027436 004737 052144 JSR PC,BINOC1 ;CONVERT TO OCTAL
5533 027442 122765 000121 000135 CMPB ,RDATA,P.LCMD(R5) ;CHECK IF PAT # VALID
5534 027450 001407 BEQ 10$ ;NO, DO NOT TYPE
5535 027452 104401 057551 TYPE ,BLANKS ;TYPE 2 BLANKS
5536 027456 005046 CLR -(SP) ;MAKE ROOM ON STACK
5537 027460 116516 000144 MOVB P.LDPT(R5),(SP) ;GET LAST DATA PATTERN
5538 027464 004737 052144 JSR PC,BINOC1 ;CONVERT TO OCTAL
5539 027470 104401 001165 10$: TYPE ,SCRLF ;TYPE <CR><LF>
5540 027474 000207 RTS PC ;RETURN

```

```

5541 .SBTTL PPINT CURRENT GENERATED COMMAND
5542
5543 077476 104401 062342 PRI002: TYPE ,ERR100 ;PRINT "CURRENT SUPPLIED COMMAND"
5544 ; CYL TRK SEC CFFSET
5545 ; BUS AD WRC CT"
5546 027502 122765 000121 000123 CMPB #RDATA,P.RCMD(R5) ;CHECK IF PAT # IS TO BE PRINTED
5547 027510 001402 BEQ 55 ;NO, DO NOT PRINT HEADER
5548 027512 104401 062535 TYPE ,ERR102 ;PRINT "PAT"
5549 027516 104401 001165 55: TYPE ,SCRLF ;PRINT <CR><LF>
5550 027522 005046 CLR -(SP) ;MAKE ROOM ON STACK
5551 027524 116516 000001 MOVB P.CMND(R5),(SP) ;STORE COMMAND FOR PRINT OUT
5552 027530 004737 052144 JSR PC,BINOCY ;CONVERT TO OCTAL
5553 027534 104401 057551 TYPE ,BLANKS ;TYPE 2 BLANKS
5554 027540 016546 000002 MOV P.CYLW(R5),-(SP) ;STORE CYLINDER
5555 027544 004737 052144 JSR PC,BINOCY ;CONVERT TO OCTAL
5556 027550 104401 057551 TYPE ,BLANKS ;TYPE 2 BLANKS
5557 027554 005046 CLR -(SP) ;MAKE ROOM ON STACK
5558 027556 116516 000005 MOVB P.TRCK(R5),(SP) ;STORE TRACK
5559 027562 004737 052144 JSR PC,BINOCY ;CONVERT TO OCTAL
5560 027566 104401 057551 TYPE ,BLANKS ;TYPE 2 BLANKS
5561 027572 005046 CLR -(SP) ;MAKE ROOM ON STACK
5562 027574 116516 000004 MOVB P.SECT(R5),(SP) ;STORE SECTOR
5563 027600 004737 052144 JSR PC,BINOCY ;CONVERT TO OCTAL
5564 027604 104401 057551 TYPE ,BLANKS ;TYPE 2 BLANKS
5565 027610 005046 CLR -(SP) ;MAKE ROOM ON THE STACK
5566 027612 122765 000021 000147 CMPB #21,P.RERD(R5) ;CHECK IF OFFSET
5567 027620 101002 BHI 105 ;NO, INDICATE ZPAC CFFSET
5568 027622 116516 000006 MOVB P.OPST(R5),(SP) ;STORE OFFSET FOR PRINT OUT
5569 027626 004737 052144 105: JSR PC,BINOCY ;CONVERT TO OCTAL
5570 027632 104401 057551 TYPE ,BLANKS ;TYPE 2 BLANKS
5571 027636 016546 000010 MOV P.BALC(R5),-(SP) ;STORE BUS ADDRESS
5572 027642 004737 052144 JSR PC,BINOCY ;CONVERT TO OCTAL
5573 027646 104401 057551 TYPE ,BLANKS ;TYPE 2 BLANKS
5574 027652 016546 000012 MOV P.WC(R5),-(SP) ;STORE WORD COUNT
5575 027656 004737 052144 JSR PC,BINOCY ;CONVERT TO OCTAL
5576 027662 122765 000121 000123 CMPB #RDATA,P.RCMD(R5) ;CHECK IF PAT # IS TO BE PRINTED
5577 027670 001402 BEQ 155 ;NO, RETURN
5578 027672 104401 057551 TYPE ,BLANKS ;TYPE 2 BLANKS
5579 027676 005046 CLR -(SP) ;MAKE ROOM ON THE STACK
5580 027700 116516 000121 MOVB P.DPAT(R5),(SP) ;STORE DATA PATTERN FOR PRINT OUT
5581 027704 004737 052144 JSR PC,BINOCY ;CONVERT TO OCTAL
5582 027710 104401 001165 155: TYPE ,SCRLF ;TYPE <CR><LF>
5583 027714 000207 RTS PC ;RETURN

```

```

5584 .SBTTL PRINT ERROR MESSAGE AND DROP DRIVE
5585 ;*****
5586 ;*
5587 ;*
5588 ;* EXPECTED PRINT CODES
5589 ;* -----
5590 ;*
5591 ;* BIT 1 PRINT DRIVE STATUS
5592 ;* BIT 2 PRINT PREVIOUS POSITION
5593 ;* BIT 5 QUESTIONABLE DRIVE STATUS
5594 ;* BIT 6 CURRENT GENERATED COMMAND
5595 ;*
5596 ;*****
5597
5598 027716 010446 PRI100: MOV R4, -(SP) ;STORE R4 ON STACK
5599 027720 012004 MOV (R0)+, R4 ;STORE ADDRESS OF PRINT CODE
5600 027722 005204 INC R4 ;CALCULATE START OF MESSAGE
5601 027724 010437 027736 MOV R4, 25 ;STORE ERROR MESSAGE ADDRESS
5602 027730 004737 027242 JSR PC, PRI000 ;PRINT HEADER
5603 027734 104401 TYPE ;TYPE ERROR MESSAGE
5604 027736 000000 25: .WORD 0
5605 027740 104401 001165 TYPE ,SCRLF ;TYPE <CR><LF>
5606 027744 032777 070000 151166 BIT #513, @51F ;CHECK IF INHIBIT PRINTOUT
5607 027752 001033 BNE 10$ ;YES, RETURN
5608 027754 105765 000150 TSTB P.DSTT(R5) ;CHECK IF FIRST ERROR
5609 027760 100402 BMI 3$ ;YES, GO PRINT CONTROLLER REGISTERS
5610 027762 104401 062712 TYPE ,ERR202 ;TYPE ERROR DURING ERROR RECOVERY
5611 027766 004737 025550 3$: JSR PC, PRISTT ;PRINT CONTROLLER REGISTERS
5612 027772 132744 000002 BITB #BIT1, -(R4) ;CHECK IF PRINT DRIVE STATUS
5613 027776 001407 BEQ 5$ ;NO, CONTINUE
5614 030000 132714 000040 BITB #BIT5, (R4) ;CHECK IF QUESTIONABLE STATUS
5615 030004 001402 BEQ 4$ ;NO, DO NOT PRINT MESSAGE
5616 030006 104401 063073 TYPE ,ERR206 ;PRINT "QUESTIONABLE DRIVE STATUS"
5617 030012 004737 025374 4$: JSR PC, PRDSTT ;PRINT DRIVE STATUS
5618 030016 132714 000100 5$: BITB #BIT6, (R4) ;CHECK IF PRINT CURRENT GENERATED COMMAND
5619 030022 001402 BEQ 8$ ;NO, DO NOT PRINT CURRENT COMMAND
5620 030024 004737 027476 JSR PC, PRI002 ;PRINT CURRENT GENERATED COMMAND
5621 030030 132714 000004 8$: BITB #BIT2, (R4) ;CHECK IF PRINT PREVIOUS COMMAND
5622 030034 001402 BFC 10$ ;NO, CHECK IF HALT ON ERROR
5623 030036 004737 027320 JSR PC, PRI001 ;PRINT PREVIOUS COMMAND
5624 030042 004737 030472 10$: JSR PC, PLT000 ;TEST IF HALT ON ERROR
5625 030046 005265 000206 INC P.NER(R5) ;INCREMENT OTHER ERROR COUNT
5626 030052 004737 022316 JSR PC, PUFREL ;RELEASE PUFFER
5627 030056 004737 012716 JSR PC, DROP ;DROP DRIVE FROM TEST SEQUENCE
5628 030062 004737 022016 JSR PC, GET90F ;GET NEW DRIVE, COMMAND, AND PUFFER
5629 030066 005037 001562 CLR ERKPRO ;CLEAR ERROR CURRENTLY BEING PROCESSED
5630 030072 012604 MOV (SP)+, R4 ;RESTORE R4
5631 030074 000200 RTS R0 ;RETURN

```

```

5632 .SBTTL PRINT ERROR MESSAGE
5633
5634 ;*****
5635 ;*
5636 ;* EXPECTED PRINT CODES
5637 ;* -----
5638 ;*
5639 ;* BIT 0 DATA TYPE ERROR = 1
5640 ;* NON-DATA TYPE ERROR = 0
5641 ;* BIT 1 PRINT DRIVE STATUS
5642 ;* BIT 2 PRINT PREVIOUS POSITION
5643 ;* BIT 3 PRINT SECTOR IN ERROR
5644 ;* BIT 4 PRINT HEADER
5645 ;* BIT 5 QUESTIONABLE DRIVE STATUS
5646 ;* BIT 6 CURRENT GENERATED COMMAND
5647 ;*
5648 ;*****
5649
5650 030076 010446 PRI200: MOV R4,-(SP) ;STORE R4 ON STACK
5651 030100 012004 MOV (R0)+,R4 ;GET PRINT CODE ERROR FLAGS
5652 030102 132724 000001 BITB #BIT0,(P4)+ ;CHECK IF DATA TYPE ERROR
5653 030106 001416 BFQ SS ;NO, NON-DATA TYPE ERROR
5654 030110 032765 000002 000150 BIT #BIT1,P.DSTT(R5) ;CHECK IF DATA TYPE ERROR
5655 ; BEING PROCESSED
5656 030116 001402 BEQ 1$ ;NO, PRINT MESSAGE
5657 030120 012604 MOV (SP)+,R4 ;RESTORE R4
5658 030122 000200 RTS R0 ;RETURN
5659
5660 030124 052765 000002 000150 1$: BIS #BIT1,P.DSTT(R5) ;SET DATA TRANSFER ERROR OCCURRED
5661 030132 100015 BPL 10$ ;CHECK IF START OF ERROR RECOVERY
5662 030134 052765 000001 000150 BIS #BIT0,P.DSTT(R5) ;YES, SET SERVICING DATA ERROR
5663 030142 000411 BP 10$ ;GO REPORT ERROR
5664
5665 030144 032765 000004 000150 5$: BIT #BIT2,P.DSTT(R5) ;CHECK IF NON-DATA TYPE ERROR OCCURRED
5666 030152 001402 BEQ 6$ ;NO, REPORT ERROR
5667 030154 012604 MOV (SP)+,R4 ;RESTORE R4
5668 030156 000200 RTS R0 ;RETURN
5669
5670 030160 052765 000004 000150 6$: BIS #BIT2,P.DSTT(R5) ;SET NON-DATA ERROR OCCURRED
5671 030166 004737 027242 10$: JSR PC,PRI000 ;TYPE ERROR HEADER
5672 030172 032777 020000 150740 BIT #SW13,@SWP ;CHECK IF INHIBIT PRINT OUT
5673 030200 001115 BFE 30$ ;YES, INHIBIT PRINT OUT
5674 030202 010437 030222 MOV R4,12$ ;STORE ERROR MESSAGE
5675 030206 105765 000150 TSTB P.DSTT(P5) ;CHECK IF ERROR DURING RECOVERY
5676 030212 100402 BMI 11$ ;NO, PRINT ERROR
5677 030214 104401 062717 TYPE ,ERR202 ;PRINT "ERROR DURING ERROR RECOVERY"
5678 030220 104401 11$: TYPE ;PRINT ERROR MESSAGE
5679 030222 000000 12$: .WOPD 0
5680 030224 104401 001165 TYPE ,SCPLF ;TYPE <CR><LF>
5681 030230 004737 025550 JSR PC,PRI000 ;PRINT CONTROLLER REGISTERS
5682 030234 132744 000002 BITB #BIT1,-(R4) ;CHECK IF PRINT DRIVE STATUS
5683 030240 001407 BEQ 22$ ;NO, CONTINUE
5684 030242 132714 000040 BITB #BIT5,(P4) ;CHECK IF QUESTIONABLE DRIVE STATUS
5685 030246 001402 BFQ 20$ ;NO, DO NOT PRINT MESSAGE
5686 030250 104401 063073 TYPE ,ERR206 ;TYPE "QUESTIONABLE DRIVE STATUS"
5687 030254 004737 025374 20$: JSR PC,PRDSTT ;PRINT DRIVE STATUS

```

5688	030260	132714	000100	27\$:	BITF	#BIT6,(R4)	;CHECK IF PRINT CURRENT COMMAND
5689	030264	001402			BEQ	23\$	;NO, CONTINUE
5690	030266	004737	027476		JSR	PC,PRIO02	;PRINT CURRENTLY ISSUED COMMAND
5691	030272	132714	000004	23\$:	BITF	#BIT2,(R4)	;CHECK IF PRINT PREVIOUS COMMAND
5697	030276	001402			BEQ	25\$	;NO, CHECK IF PRINT HEADER
5693	030300	004737	027320		JSR	PC,PRIO01	;PRINT PREVIOUS COMMAND
5694	030304	132714	000001	25\$:	BITF	#BIT0,(R4)	;CHECK IF DATA TYPE ERROR
5695	030310	001422			BEQ	27\$	;NO, DO NOT PRINT ECC
5696	030312	032765	100000 000034		BIT	#DCK,P.ER(R5)	;CHECK IF DATA CHECK
5697	030320	001416			BEQ	27\$	;NO, DO NOT PRINT ECC
5698	030322	104401	062141		TYPE	,ERR067	;TYPE ECC PAT, ECC POS
5699	030326	016546	000062		MOV	P.EPAT(R5),-(SP)	;GET PATTERN
5700	030332	004737	052144		JSR	PC,BINOC1	;CONVERT TO OCTAL
5701	030336	104401	057551		TYPE	,BLANKS	
5702	030342	016546	000060		MOV	P.EPOS(P5),-(SP)	;GET POSITION
5703	030346	004737	052144		JSR	PC,BINOC1	;CONVERT TO OCTAL
5704	030352	104401	001165		TYPE	,SCRLF	
5705	030356	132724	000020	27\$:	BITF	#BIT4,(R4)+	;CHECK IF PRINT HEADER
5706	030362	001424			BEQ	30\$	;NO, CHECK IF HALT ON ERROR
5707	030364	104401	062554		TYPE	,ERR111	;TYPE HEADER DESCRIPTOR
5708	030370	013746	001642		MOV	HEAD1,-(SP)	;STORE FIRST WORD OF THE HEADER
5709	030374	004737	052144		JSR	PC,BINOC1	;CONVERT TO OCTAL
5710	030400	104401	057551		TYPE	,BLANKS	;TYPE 2 BLANKS
5711	030404	013746	001644		MOV	HEAD2,-(SP)	;STORE SECOND WORD OF THE HEADER
5712	030410	004737	052144		JSR	PC,BINOC1	;CONVERT TO OCTAL
5713	030414	104401	057551		TYPE	,BLANKS	;TYPE 2 BLANKS
5714	030420	013746	001646		MOV	HEAD3,-(SP)	;STORE THIRD WORD OF THE HEADER
5715	030424	004737	052144		JSR	PC,BINOC1	;CONVERT TO OCTAL
5716	030430	104401	001165		TYPE	,SCRLF	;TYPE <CR><LF>
5717	030434	132744	000010	30\$:	BITF	#BIT3,-(R4)	;CHECK IF PRINT WHOLE SECTOR
5718	030440	001402			BEQ	40\$	;NO, CHECK IF HALT ON ERROR
5719	030442	004737	026704		JSR	PC,PRIBUF	;GO PRINT BUFFER
5720	030446	132714	000001	40\$:	BITF	#BIT0,(R4)	;CHECK IF DATA TYPE ERROR
5721	030452	001403			BEQ	45\$	;NO, CHECK IF HALT ON ERROR
5722	030454	042765	100200 000150		BIC	#BIT15 BIT7,P.DST(P5)	;CLEAR FIRST ERROR AND ERGCH
5723							; ENQUEUED
5724	030462	004737	030472	45\$:	JSR	PC,HLT000	;CHECK IF HALT ON ERROR
5725	030466	012604			MOV	(SP)+,R4	;RESTORE R4
5726	030470	000200			RTS	R0	;RETURN

```
5727 .SBTTL HALT ON ERROR CHECK
5728
5729 030472 032777 100000 150440 HLT000: BIT #SW15, @SWR ;CHECK IF HALT ON ERROR
5730 030500 001407 BFQ 10$ ;NO, RETURN
5731 030502 005737 001462 TST 0.WAIT ;CHECK IF WAITING FOR COMMAND
5732 ; COMPLETION
5733 030506 001403 BEQ 5$ ;NO, HALT
5734 030510 105762 000000 1$: TSTB RRC51(R2) ;WAIT FOR READY
5735 030514 001775 BEQ 1$
5736 030516 000000 5$: HALT
5737 030520 000207 10$: RTS PC ;RETURN
```

```

5738 .SBTTL OTHER ERROR STATISTICS GATHERING
5739
5740 030522 105765 000150 MERSTT: TSTB P.DSTT(R5) ;CHECK IF FIRST ERROR
5741 030526 100057 BPL STT5$ ;NO, CONTINUE ERROR RECOVERY
5742 030530 042765 100200 000150 BIC @BIT15IBIT7,P.DSTT(R5) ;RESET FIRST ERROR AND ERROR ENQUEUED
5743 030536 005265 000206 INC P.MER(R5) ;INCREMENT OTHER ERRORS COUNT
5744 030542 022765 000020 000206 CMP @T.MER,P.MER(R5) ;CHECK IF ERROR THRESHOLD EXCEEDED
5745 030550 103426 BLO STT2$ ;YES, CHECK IF DROP DRIVE
5746 030552 000445 BR STT5$ ;NO, START ERROR RECOVERY SEQUENCE
5747
5748 .SBTTL SEEK INCOMPLETE STATISTICS GATHERING
5749
5750 030554 105765 000150 SKISTT: TSTB P.DSTT(R5) ;CHECK IF FIRST ERROR
5751 030560 100047 BPL STT5$ ;NO, CONTINUE ERROR RECOVERY
5752 030562 005265 000202 INC P.NSKI(R5) ;INCREMENT NUMBER OF SEEK INCOMPLETES
5753 030566 000405 BR STT1$ ;GO CHECK IF ERROR THRESHOLD EXCEEDED
5754
5755 .SBTTL OPERATION INCOMPLETE STATISTICS GATHERING
5756
5757 030570 105765 000150 OPISTT: TSTB P.DSTT(R5) ;CHECK IF FIRST ERROR
5758 030574 100034 BPL STT5$ ;NO, CONTINUE ERROR PROCESSING
5759 030576 005265 000204 INC P.NCPI(R5) ;INCREMENT NUMBER OF OPERATION INCOMPLETES
5760 030602 042765 100200 000150 STT1$: BIC @BIT15IBIT7,P.DSTT(R5) ;RESET FIRST ERROR AND ERROR ENQUEUED
5761 030610 016546 000204 MOV P.NOPI(R5),-(SP) ;STORE NUMBER OF OPERATION INCOMPLETES
5762 030614 066516 000202 ADD P.NSKI(R5),(SP) ;ADD SEEK INCOMPLETES
5763 030620 026526 000104 CMP P.SKET(R5),(SP)+ ;CHECK IF ERROR THRESHOLD EXCEEDED
5764 030624 103020 BHS STT5$ ;NO, START ERROR RECOVERY
5765 030626 032777 040020 150304 STT2$: BIT @SM4ISM14,@SMR ;CHECK IF INHIBIT DRIVE DESELECTION
5766 030634 001014 BNE STT5$ ;YES, START RECOVERY SEQUENCE
5767 030636 104401 063040 TYPE ,PKR205 ;PAINT "ERROR THRESHOLD EXCEEDED"
5768 030642 004737 022316 JSR PC,PUPRFL ;RELEASE BUFFER
5769 030646 004737 012716 JSR PC,DROP ;DROP DRIVE
5770 030652 004737 022016 JSR PC,GETBUF ;GET NEW COMMAND, DRIVE, AND BUFFER
5771 030656 005037 001562 CLR ERRPRO ;CLEAR ERROR CURRENT BEING PROCESSED
5772 030662 011000 MOV (R0),R0 ;INDICATE NO RECOVERY SEQUENCE IN PROGRESS
5773 030664 000200 RTS R0 ;RETURN
5774
5775 030666 105765 000120 STT5$: TSTB P.IERC(R5) ;CHECK IF INHIBIT ERROR RECOVERY
5776 030672 001010 BNE SS ;YES, INDICATE UNRECOVERABLE RETURN
5777 030674 105765 000146 INCR P.RECT(R5) ;INCREMENT RETRY COUNT
5778 030700 122765 000010 000146 CMPR @R.,P.RECT(R5) ;CHECK IF RETRY FINISHED
5779 030706 103402 BLO SS ;YES, CLEAN UP AFTER UNSUCCESSFUL RETRY
5780 030710 005720 TST (R0)+ ;INDICATE RECOVERY SEQUENCE IN PROGRESS
5781 030712 000200 RTS R0 ;RETURN
5782
5783 030714 004737 030724 SS: JSR PC,UNRECY ;CLEAN UP FOR UNSUCCESSFUL RECOVERY
5784 030720 011000 MOV (R0),R0 ;INDICATE NO RECOVERY SEQUENCE IN PROGRESS
5785 030722 000200 RTS R0 ;RETURN

```

```
5786 .SBTTL UNSUCCESSFUL RECOVERY CLEANUP
5787
5788 030724 105037 001640 UNRECV: CLRR RTVSCN ;CLRR RETRY SECTOR COUNT
5789 030730 004737 022316 JSR PC,BUPREL ;RELEASE BUFFER
5790 030734 032765 000001 000150 BIT #BIT0,P.DSTT(R5) ;CHECK IF DATA HARD ERROR
5791 030742 001415 BEQ 15 ;NO, CHECK IF DRIVE IS BEING ASSIGNED
5792 030744 005265 000200 INC P.HARD(R5) ;INCREMENT HARD ERROR COUNT
5793 030750 026565 000102 000200 CMP P.UERT(R5),P.HARD(R5) ;CHECK IF HARD ERROR THRESHOLD EXCEEDED
5794 030756 103007 BHS 15 ;NO, CHECK IF DRIVE IS BEING ASSIGNED
5795 030760 032777 040020 150152 BIT #SW41SW14,0SWP ;INHIBIT DRIVE DESELECTION
5796 030766 001003 BNE 15 ;YES, CHECK IF DRIVE IS BEING ASSIGNED
5797 030770 104401 063040 TYPE ,ERR205 ;PRINT ERROR THRESHOLD EXCEEDED
5798 030774 000413 BP 25 ;GO DROP DRIVE
5799
5800 030776 105765 000210 15: TSTB P.ASSN(P5) ;CHECK IF DRIVE IS BEING ASSIGNED
5801 031002 001421 BEQ 55 ;NO, CLEAN UP ERROR
5802 031004 132765 000010 000210 BITR #BIT3,P.ASSN(R5) ;CHECK IF READ BACK SERIAL NUMBER
5803 031012 001404 BEQ 25 ;NO, DROP DRIVE
5804 031014 132765 000200 000210 BITR #BIT7,P.ASSN(R5) ;CHECK IF RECAL FOR READ OF
5805 ; PACK SERIAL NUMBER
5806 031022 001441 BEQ 105 ;NO, ISSUE RECALIBRATE AND TRY AGAIN
5807 031024 104401 063003 25: TYPE ,ERR204 ;TYPE "ERROR RECOVERY UNSUCCESSFUL"
5808 031030 004737 012716 JSR PC,DROP ;DROP DRIVE FROM TEST SEQUENCE
5809 031034 004737 022016 JSR PC,GETBUF ;GET NEW DRIVE, COMMAND, AND BUFFER
5810 031040 005037 001562 CLR EPRPRO ;CLEAR ERROR CURRENT BEING PROCESSED
5811 031044 000207 RTS PC ;RETURN
5812
5813 031046 105765 000120 55: TSTB P.IERC(R5) ;CHECK IF INHIBIT RETRY
5814 031052 001006 BNE 65 ;YES, DO NOT PRINT MESSAGE
5815 031054 032777 020000 150056 BIT #SW13,0SWP ;CHECK IF INHIBIT PRINT OUT
5816 031062 001002 BNE 65 ;YES, CONTINUE
5817 031064 104401 063003 TYPE ,ERR204 ;TYPE "ERROR RECOVERY UNSUCCESSFUL"
5818 031070 010446 65: MOV R4,-(SP) ;STORE R4 ON STACK
5819 031072 116504 000000 MOVR P.DRVN(R5),R4 ;GET DRIVE NUMBER
5820 031076 156437 001510 001507 BISH INTMSK(R4),0,CVER ;INDICATE IMPLIED SEKS ARE ALLOWED
5821 031104 012604 MOV (SP)+,R4 ;RESTORE R4
5822 031106 004037 050706 JSR RO,Q.PUSH ;PUT PARAMETER BLOCK IN DRIVE
5823 031112 001314 AVATLQ ; AVAILABLE QUEUE
5824 031114 004737 022016 JSR PC,GETBUF ;GET NEW DRIVE, COMMAND, AND BUFFER
5825 031120 005037 001562 CLR EPRPRO ;CLEAR ERROR CURRENT BEING PROCESSED
5826 031124 000207 RTS PC ;RETURN
5827
5828 031126 122765 000010 000126 105: CMPB #R.,P.RSEC(R5) ;CHECK IF ALL 16-PIE MODE SECTORS HAVE BEEN TRIED
5829 031134 101733 BLOS 25 ;YES, DROP DRIVE
5830 031136 010446 MOV R4,-(SP) ;STORE R4 ON STACK
5831 031140 116504 000000 MOVR P.DRVN(R5),R4 ;GET DRIVE NUMBER FOR INDEX
5832 031144 156437 001510 001507 BISH INTMSK(R4),0,CVER ;INDICATE THAT IMPLIED SEKS ARE ALLOWED
5833 031152 012604 MOV (SP)+,R4 ;RESTORE R4
5834 031154 152765 000200 000210 BISH #BIT7,P.ASSN(R5) ;INDICATE THAT RECALIBRATE HAS BEEN ISSUED
5835 031162 005065 000146 CLR P.RECT(R5) ;CLEAR ERROR FLAGS
5836 031166 005065 000150 CLR P.DSTT(R5)
5837 031172 005065 000212 CLR P.EPR(R5)
5838 031176 112765 000113 000001 MOVR #RECAL,P.COMND(R5) ;LOAD RECALIBRATE COMMAND
5839 031204 004037 050706 JSR RO,Q.PUSH ;PUT PARAMETER BLOCK IN COMMAND
5840 031210 001324 CINITQ ; INITIATION QUEUE
5841 031212 004737 022016 JSR PC,GETBUF ;GO ALLOCATE MEMORY BUFFER
```

RR611/RR06 PERFORMANCE EXERCISE MAINDEC DZR6P-B MACY11 30(1046) 01-DEC-77 14:15 PAGE 128
DZR6PP.P11 11-AUG-76 00:00 UNSUCCESSFUL RECOVERY CLFANUP

SEQ 0150

5842 031216 005037 001562
5843 031222 000207

CLR
RTS

ENRPRC
PC

;CLFAP FORCE PROCESSING IN PROGRESS
;RETURN

G12

```

5844 .SBTTL ERROR RECOVERY SEQUENCE
5845
5846 031224 032765 000200 000212 EPCVY: BIT #BIT7,P.ERR(R5) ;CHECK IF UNSOLICITED INTERRUPT
5847 031232 001441 BEQ 35 ;NO, CHECK IF EPACR INFORMATION STILL VALID
5848 031234 032777 002000 147676 BIT #SW10,0SWR ;CHECK IF BELL ON ERROR
5849 031242 001402 BFQ 15 ;NO, CONTINUE PROCESSING
5850 031244 104401 001160 TYPE ,SBELL ;RING BELL
5851 031250 032777 020000 147662 15: BIT #SW13,0SWR ;CHECK IF INHIBIT PRINT OUT
5852 031256 001020 BNE 25 ;YES, CHECK IF HALT ON ERROR
5853 031260 116537 000000 055446 MOVR P.DRVN(R5),OPR011 ;LOAD DRIVE NUMBER FOR PRINT OUT
5854 031266 152737 000060 055446 BISH #60,OPR011 ;MAKE IT ASCII
5855 031274 104401 055427 TYPE ,OPR010 ;TYPE "DRIVE N"
5856 031300 004737 043072 JSR PC,PRITIM ;PRINT TIME
5857 031304 104401 060704 TYPE ,ERR026 ;TYPE "INT FROM DRV NOT UNDER TEST"
5858 031310 004737 025550 JSR PC,PRISTT ;PRINT CONTROLLER REGISTERS
5859 031314 004737 025774 JSR PC,PRDSTT ;PRINT DRIVE STATUS
5860 031320 004737 030472 25: JSR PC,HLT000 ;CHECK IF HALT ON ERROR
5861 031324 005065 000212 CLR P.ERR(R5) ;CLEAR ERROR FLAGS
5862 031330 005037 001562 CLR ERRPRO ;CLEAR ERROR BEING PROCESSED
5863 031334 000207 RTS PC ;RETURN
5864
5865 031336 032765 000020 000212 35: BIT #BIT4,P.ERR(R5) ;CHECK IF SERVICING DRIVE SEIZED TIME OUT
5866 031344 001427 BEQ 55 ;NO, CONTINUE
5867 031346 004737 027242 JSR PC,PR1000 ;PRINT HEADER
5868 031352 104401 061744 TYPE ,ERR059 ;PRINT "TIME OUT BECAUSE DRIVE SEIZED"
5869 031356 032777 020000 147554 BIT #SW13,0SWR ;CHECK IF INHIBIT PRINT OUT
5870 031364 001002 BNE 45 ;YES, DO NOT PRINT CURRENT GENERATED COMMAND
5871 031366 004737 027476 JSR PC,PR1002 ;PRINT CURRENT GENERATED COMMAND
5872 031372 005265 000206 45: INC P.NFR(R5) ;INCREMENT OTHER ERRORS
5873 031376 004737 030472 JSR PC,HLT000 ;CHECK IF HALT ON ERROR
5874 031402 004737 022316 JSR PC,PUPRFL ;RELEASE BUFFER
5875 031406 004737 012716 JSR PC,DROP ;DROP DRIVE
5876 031412 004737 022016 JSR PC,GETBUF ;GET NEW DRIVE, COMMAND, AND BUFFER
5877 031416 005037 001562 CLR ERRPRO ;CLEAR ERROR BEING PROCESSED
5878 031422 000207 RTS PC ;RETURN
5879
5880 031424 032765 000400 000212 55: BIT #BIT9,P.ERR(R5) ;CHECK IF TIME OUT BECAUSE HEADS DID NOT RELEASE
5881 031432 001404 BEQ 65 ;NO, CONTINUE
5882 031434 004037 027716 JSR RO,PR1100 ;PRINT TIME OUT BECAUSE HEADS DID NOT RELEASE
5883 031440 062277 ERN074
5884 031442 000207 RTS PC ;RETURN
5885
5886 031444 032765 000040 000212 65: BIT #BIT5,P.ERR(R5) ;CHECK IF TIME OUT DURING DRIVE POSITIONING
5887 031452 001425 BEQ 75 ;NO, CONTINUE
5888 031454 042765 100000 000150 BIT #BIT15,P.DSTT(R5) ;RESET ERROR FLAGUED
5889 031462 052765 000100 000150 BIS #BIT6,P.DSTT(R5) ;SET READ STATUS ISSUED
5890 031470 116537 000001 001662 MOVR P.CMND(R5),POSCMD ;STORE COMMAND IN ERNCR
5891 031476 042765 000002 000014 BIT #DRVPOS,P.PRST(R5) ;RESET DRIVE POSITIONING
5892 031504 004737 036150 JSR PC,GETCOR ;STORE PRESENT STATUS FOR PRINT OUT
5893 031510 112765 000141 000001 MOVR #RDSTAT,P.CMND(R5) ;GET DRIVE STATUS
5894 031516 004037 050706 JSR RO,Q.PUSH ;ENQUEUE COMMAND IN COMMAND
5895 031522 001324 CINITQ ; INITIATION QUEUE
5896 031524 000207 RTS PC ;RETURN
5897
5898 031526 032765 000100 000212 75: BIT #BIT6,P.ERR(R5) ;CHECK IF ERROR WHILE ENQUEUED
5899 031534 001404 BEQ 85 ;NO, CONTINUE

```

```
5900 031536 004037 027716 JSR RO,PRI100 ;PRINT HEADPR
5901 031542 060564 EPR022 ;TYPE "ERROR WHILE WAITING TO
5902 ; REPORT ERROR"
5903 031544 000207 RTS PC ;RETURN
5904
5905 031546 032765 004000 000016 8$: BIT #CTO,P.CS1(R5) ;CHECK IF CONTROLLER TIME OUT
5906 031554 001404 BEQ 9$ ;NO, CONTINUE
5907 031556 004037 036524 JSR RO,CONERR ;GO REPORT ERROR
5908 031562 060476 EPR017
5909 031564 000207 RTS PC ;RETURN
5910
5911 031566 032765 002000 000020 9$: BIT #PGE,P.CS2(R5) ;CHECK IF PROGRAMMING ERROR
5912 031574 001404 BEQ 10$ ;NO, CONTINUE
5913 031576 004037 036524 JSR RO,CONEPR ;PRINT HEADER
5914 031602 060532 EPR020 ;TYPE "PROC ERROR"
5915 031604 000207 RTS PC ;RETURN
5916
5917 031606 032765 000001 000034 10$: BIT #ILC,P.ER(R5) ;CHECK IF ILLEGAL FUNCTION CODE
5918 031614 001404 BEQ 11$ ;NO, CONTINUE
5919 031616 004037 036524 JSR RO,CONEPR ;PRINT HEADER
5920 031622 060545 EPR021 ;TYPE "ILLEGAL FUNCTION CODE"
5921 031624 000207 RTS PC ;RETURN
5922
5923 031626 032765 010000 000020 11$: BIT #NED,P.CS2(R5) ;CHECK FOR NON-EXISTENT DRIVE
5924 031634 001404 BEQ 15$ ;NO, CONTINUE
5925 031636 004037 027716 JSR RO,PRI100 ;PRINT HEADER
5926 031642 060631 EPR023 ;TYPE "NON-EXISTENT DRIVE"
5927 031644 000207 RTS PC ;RETURN
5928
5929 031646 032765 000400 000020 15$: BIT #UFE,P.CS2(R5) ;CHECK IF UNIT FIELD ERROR
5930 031654 001404 BEQ 18$ ;NO, CONTINUE
5931 031656 004037 027716 JSR RO,PRI100 ;PRINT HEADER
5932 031662 060664 EPR025 ;TYPE "UNIT FIELD ERROR"
5933 031664 000207 RTS PC ;RETURN
5934
5935 031666 032765 020000 000016 18$: BIT #SPAR,P.CS1(R5) ;CHECK IF CONTROLLER DETECTED BAD PARITY
5936 031674 001004 BNE 19$ ;YES, PRINT MESSAGE
5937 031676 032765 000010 000034 BIT #DRPAR,P.ER(R5) ;CHECK IF DRIVE DETECTED BAD PARITY
5938 031704 001437 BEQ 25$ ;NO, CONTINUE
5939 031706 004037 030076 19$: JSR RO,PRI200 ;PRINT HEADPR
5940 031712 060650 EPR024 ;PRINT "SERCON PARITY"
5941 031714 004037 030522 JSR NO,NERSTT ;TAKE STATISTICS FOR OTHER ERRORS
5942 031720 032166 60$ ;ERROR RECOVERY SEQUENCE FINISHED RETURN
5943 031722 032765 020000 000016 BIT #SPAR,P.CS1(R5) ;CHECK IF CONTROLLER DETECTED SERCON PARITY ERROR
5944 031730 001021 BNE 22$ ;YES, GO RETRY COMMAND
5945 031732 032765 003400 000150 BIT #BIT8|BIT9|BIT10,P.DSTT(R5) ;CHECK IF NO POSITIONING
5946 031740 001015 BNE 22$ ;YES, RETRY COMMAND
5947 031742 032765 000007 000210 BITB #BIT0|BIT1|BIT2,P.ASSN(R5) ;CHECK IF NO POSITION
5948 ; DURING DRIVE ASSIGNMENT
5949 031750 001011 BNE 22$ ;YES, RETRY COMMAND
5950 031752 042765 030000 000150 BIT #BIT12|BIT13,P.DSTT(R5) ;RESET RE-SER AND RE-OFFSET
5951 031760 052765 004000 000150 BIS #BIT11,P.DSTT(R5) ;SET RECALIBRATE DRIVE
5952 031766 112765 000113 000001 MOVB #PECAL,P.CMND(R5) ;LOAD RECALIBRATE COMMAND
5953 031774 004037 050706 22$: JSR RO,Q.PUSH ;PUT PARAMETER BLOCK IN COMMAND
5954 032000 001324 CINITO ; INITIATION QUEUE
5955 032002 000207 RTS PC ;RETURN
```

5956									
5957	032004	032765	004000	000020	25\$:	BIT	#MEM,P.CS2(R5)	;CHECK IF MEM-EXISTENT MEMORY	
5958	032012	001404				BEQ	30\$	;NO, CONTINUE	
5959	032014	004037	036524			JSR	RO, CONERR	;PRINT HEADER	
5960	032020	061270				ERR044		;PRINT "NON-EXISTENT MEMORY"	
5961	032022	000207				RTS	PC	;RETURN	
5962									
5963	032024	032765	020000	000020	30\$:	BIT	#UPE,P.CS2(R5)	;CHECK IF UNIBUS PARITY	
5964	032032	001404				BEQ	35\$	;NO, CONTINUE	
5965	032034	004037	036524			JSR	RO, CONERR	;PRINT HEADER	
5966	032040	061306				ERR045		;PRINT "UNIBUS PARITY"	
5967	032042	000207				RTS	PC	;RETURN	
5968									
5969	032044	004037	036714		35\$:	JSR	RO, TSTDRP	;CHECK IF DPCP DRIVE ERROR	
5970	032050	032166				60\$		;YES, ERROR RETURN	
5971	032052	032765	000002	000034		BIT	#SKI,P.ER(R5)	;CHECK IF SEEK INCOMPLETE	
5972	032060	001407				BEQ	45\$	;NO, CONTINUE	
5973	032062	004037	030076			JSR	RO, PRI200	;PRINT HEADER	
5974	032066	061172				ERR038		;PRINT "SEEK INCOMPLETE"	
5975	032070	004037	030554			JSR	RO, SKISTT	;TAKE SEEK INCOMPLETE STATISTICS	
5976	032074	032166				60\$		;ERROR RECOVERY FINISHED RETURN	
5977	032076	000412				BR	48\$	;GC ISSUE RECALIBRATE	
5978									
5979	032100	032765	000040	000036	45\$:	BIT	#DROT,P.DS(R5)	;CHECK IF DRIVE OFF TRACK	
5980	032106	001420				BEQ	50\$	;NO, CONTINUE	
5981	032110	004037	030076			JSR	RO, PRI200	;PRINT HEADER	
5982	032114	061207				ERR039		;PRINT "DRIVE OFF TRACK"	
5983	032116	004037	030522			JSR	RO, MERSTT	;TAKE STATISTICS ON OTHER ERRORS	
5984	032122	032166				60\$		;END OF ERROR RECOVERY RETURN	
5985	032124	042765	034000	000150	48\$:	BIC	#PIT11,PIT12,PIT13,P.DSTT(R5)	;RESET STATUS	
5986	032132	052765	004000	000150		BIS	#BIT11,P.DSTT(R5)	;SET RECALIBRATE ISSUED	
5987	032140	112765	000113	000001		MOVP	#RECAL,P.CMND(R5)	;LOAD RECALIBRATE	
5988	032146	000712				BR	22\$	;ENQUEUF COMMAND	
5989									
5990	032150	032765	001000	000034	50\$:	BIT	#COE,P.ER(R5)	;CHECK IF CYLINDER ADDRESS OVERFLOW	
5991	032156	001404				BEQ	60\$	;NO, CONTINUE	
5992	032160	004037	036524			JSR	RO, CONERR	;PRINT HEADER	
5993	032164	061406				ERR051		;PRINT "CYLINDER ADDRESS OVERFLOW"	
5994	032166	000707			60\$:	RTS	PC	;RETURN	
5995									
5996	032170	005765	000022		60\$:	TST	P.WCR(R5)	;CHECK IF FINAL WORD COUNT IS ZERO	
5997	032174	001410				BEQ	5\$	;YES, CONTINUE	
5998	032176	026565	000022	000012		CMP	P.WCR(R5),P.WC(R5)	;CHECK THAT FINAL WORD COUNT IS LESS	
5999								; THAN OR EQUAL TO INITIAL WORD COUNT	
6000	032204	103004				BHIS	5\$	;YES, CONTINUE	
6001	032206	004037	036524			JSR	RO, CONERR	;TYPE "WORD COUNT INVALID"	
6002	032212	062162				ERR070			
6003	032214	000207				RTS	PC	;RETURN	
6004									
6005	032216	032765	001400	000016	5\$:	BIT	#PA16,BA17,P.CS1(R5)	;CHECK IF BUS ADDRESS BITS 16 OR 17 SET	
6006	032224	001015				BNE	8\$	;YES, REPORT ERROR	
6007	032226	026565	000024	000010		CMP	P.BAR(R5),P.BALO(P5)	;CHECK THAT FINAL BUS ADDRESS IS GREATER	
6008								; THAN OR EQUAL TO INITIAL BUS ADDRESS	
6009	032234	103411				BLO	8\$	;NO, REPORT ERROR	
6010	032236	016546	000012			MOV	P.WC(R5),-(SP)	;PUT INITIAL WORD COUNT ON STACK	
6011	032242	005416				NEG	(SP)	;MAKE IT POSITIVE	

6012	032244	006316				ASL	(SP)	;MULTIPLY BY 2
6013	032246	066516	000010			ADD	P.BALD(R5),(SP)	;CALCULATE MAXIMUM FINAL ADDRESS
6014	032252	026526	000024			CMP	P.BAR(R5),(SP)	;MAKE SURE IT IS NOT TOO LARGE
6015	032256	101404				BLOS	10\$	;GOOD ADDRESS CONTINUE
6016	032260	004037	036524		8\$:	JSR	RO,CONERR	;TYPE "BUS ADDRESS INVALID"
6017	032264	062205				ERR071		
6018	032266	000207				RTS	PC	;RETURN
6019								
6020	032270	026565	000030	000002	10\$:	CMP	P.DCYL(R5),P.CYLN(R5)	;CHECK THAT FINAL CYLINDER IS GREATER
6021								; THAN OR EQUAL TO INITIAL CYLINDER
6022	032276	001405				BEQ	15\$	;IF EQUAL CHECK TRACK/SECTOR
6023	032300	101014				BHI	E1\$	;IF GREATER THAN, DETERMINE ERROR
6024	032302	004037	036524			JSR	RO,CONERR	;TYPE "CYLIN ADDRESS INVALID"
6025	032306	062231				ERR072		
6026	032310	000207				RTS	PC	;RETURN
6027								
6028	032312	026565	000026	000004	15\$:	CMP	P.DTS(R5),P.SECT(P5)	;CHECK TRACK/SECTOR GREATER THAN OF
6029								; EQUAL TO INITIAL VALUES
6030	032320	103004				BHIS	E1\$	;YES, GO INTO ERROR RECOVERY
6031	032322	004037	036524			JSR	RO,CONERR	;TYPE "SECTOR/TRACK INVALID"
6032	032326	062252				ERR073		
6033	032330	000207				RTS	PC	
6034								
6035	032332	032765	010000	000034	E1\$:	BIT	#DTE,P.ER(R5)	;CHECK FOR DRIVE TIMING ERROR
6036	032340	001002				BNE	2\$	;YES, PROCESS DRIVE TIMING ERROR
6037	032342	000137	033034			JMP	E2\$	;NO, CONTINUE
6038								
6039	032346	105765	000147		2\$:	TSTB	P.RERD(R5)	;CHECK IF FIRST DATA ERROR IN
6040								; RETRY SEQUENCE
6041	032352	001117				BNE	30\$	;NO, REPORT ERROR
6042	032354	016537	000010	001630		MOV	P.BALD(R5),RTYBA	;STORE SUPPLIED POS ADDRESS
6043	032362	016537	000012	001632		MOV	P.WC(R5),RTYWC	;STORE SUPPLIED WORD COUNT
6044	032370	005437	001632			NEG	RTYWC	;MAKE IT POSITIVE
6045	032374	022737	000400	001632		CMP	#256.,RTYWC	;CHECK IF LESS THAN OR EQUAL TO A SECTOR
6046	032402	103013				BHIS	21\$	;YES, WORK WITH ONE SECTOR
6047	032404	026565	000026	000004		CMP	P.DTS(R5),P.SECT(R5)	;CHECK IF STARTING TRACK AND SECTOR
6048	032412	001021				BNE	25\$	;NOT RETRY 2 SECTORS
6049	032414	076565	000030	000002		CMP	P.DCYL(R5),P.CYLN(R5)	;CHECK IF STARTING CYLINDER
6050	032422	001015				BNE	25\$	;NO, RETRY 2 SECTORS
6051	032424	012737	000400	001632		MOV	#256.,RTYWC	;SET WORD COUNT = 1 SECTOR
6052	032432	112737	000001	001640	21\$:	MOVB	#1,RTYSCN	;SET SECTOR COUNT = 1
6053	032440	016537	000004	001634		MOV	P.SECT(R5),RTYSEC	;LOAD SECTOR AND TRACK
6054	032446	016537	000002	001636		MOV	P.CYLN(R5),RTVCYL	;LOAD CYLINDER
6055	032454	000456				BR	30\$	;GO REPORT ERROR
6056								
6057	032456	016537	000026	001634	25\$:	MOV	P.DTS(R5),RTYSEC	;SET SECTOR AND TRACK
6058	032464	016537	000030	001636		MOV	P.DCYL(R5),RTVCYL	;GET CYLINDER
6059	032472	105737	001634			TSTB	RTYSEC	;CHECK IF SECTOR = 0
6060	032476	001403				BEQ	26\$	;YES, GO DECREMENT TRACK
6061	032500	105337	001634			DECB	RTYSEC	;DECREMENT SECTOR
6062	032504	000416				BP	28\$	;DETERMINE WORD COUNT
6063								
6064	032506	112737	000025	001634	26\$:	MOVB	#21.,RTYSEC	;LOAD SECTOR = 21
6065	032514	105737	001635			TSTB	RTYTRK	;CHECK IF TRACK 0
6066	032520	001403				BFG	27\$	;YES, GO DECREMENT CYLINDER
6067	032522	105337	001635			DECB	RTYTRK	;DECREMENT TRACK

6068	032526	000405			BR	2R\$		; DETERMINE WORD COUNT
6069								
6070	032530	112737	000002	001635	27\$:	MOVW	#2,RTYTRK	; LOAD TRACK = 2
6071	032536	005337	001636			DEC	RTYCYL	; DECREMENT CYLINDER
6072	032542	004737	036412		2R\$:	JSR	PC,CALMC	; CALCULATE RETRY WORD COUNT
6073	032546	112737	000002	001640		MOVW	#2,RTYSCN	; LOAD SECTOR COUNT = 2
6074	032554	022737	001000	001632		CMP	#512.,RTYWC	; CHECK IF LESS THAN OR EQUAL
6075	032562	101004				BNI	29\$	; TWO SECTORS
6076	032564	012737	001000	001632		MOV	#512.,RTYWC	; MAKE IT 512
6077	032572	000407				BR	30\$	; GO BEFORE RECOVER
6078								
6079	032574	022737	000400	001632	29\$:	CMP	#256.,RTYWC	; CHECK IF ONLY ONE SECTOR
6080	032602	103403				BLO	30\$	; NO, GO REPORT ERROR
6081	032604	112737	000001	001640		MOVW	#1,RTYSCN	; LOAD SECTOR COUNT = 1
6082								
6083	032612	004037	030076		30\$:	JSR	NO,PR1200	; PRINT HEADER
6084	032616	061321				ERR046		; PRINT DRIVE TIMING ERROR
6085	032620	105765	000120			TSTB	P.IERC(R5)	; CHECK IF INHIBIT ERROR RECOVERY
6086	032624	001016				BNE	35\$	; YES, GO CLEAN UP FOR NEXT COMMAND
6087	032626	105265	000147			INCB	P.RERD(R5)	; INCREMENT REREAD COUNT
6088	032632	122765	000043	000147		CMPB	#43,P.RERD(R5)	; CHECK IF RETRY UNSUCCESSFUL
6089	032640	101410				BLGS	35\$	; YES, GO INDICATE UNSUCCESSFUL
6090								; RECOVERY
6091	032642	122765	000121	000123		CMPB	#RDATA,P.RCMD(R5)	; CHECK IF READ DATA
6092	032650	001407				BEQ	37\$	; YES, GO TO RECOVERY SEQUENCE
6093	032652	122765	000021	000147		CMPB	#21,P.RERD(R5)	; CHECK IF RETRY UNSUCCESSFUL
6094	032660	101026				BNI	40\$	; NO, GO TO RETRY SEQUENCE
6095	032662	004737	030724		35\$:	JSR	PC,UNRECV	; CLEAN UP FOR UNSUCCESSFUL RECOVERY
6096	032666	000207				RTS	PC	; RETURN
6097								
6098	032670	122765	000021	000147	37\$:	CMPB	#21,P.RERD(R5)	; CHECK IF OFFSET +400 MICRO-INCHES
6099	032676	001017				BNE	40\$	; NO, CONTINUE RECOVERY
6100	032700	122737	000001	001640		CMPB	#1,RTYSCN	; CHECK IF NUMBER OF SECTORS = 1
6101	032706	001404				BEQ	38\$	; YES, DO OFFSET
6102	032710	122737	000025	001634		CMPB	#21.,RTYSEC	; CHECK IF MID-TRANSFER SEEK
6103								; IS REQUIRED
6104	032716	001761				BEQ	35\$	; YES, INDICATE UNSUCCESSFUL RETRY
6105	032720	010446			38\$:	MOV	R4,-(SP)	; STORE R4 ON THE STACK
6106	032722	116504	000000			MOVW	P.DRVN(R5),R4	; GET DRIVE NUMBER FOR INDEX
6107	032726	146437	001510	001507		BICB	INTMSK(R4),O.OVER	; DO NOT SEEK BEFORE DATA
6108								; TRANSFER (RESETTING OFFSET)
6109	032734	012604				MOV	(SP)+,R4	; RESTORE R4
6110								
6111	032736	004037	036230		40\$:	JSR	RO,DETOFF	; DETERMINE IF OFFSET INSTRUCTION
6112	032742	033032				50\$		; OFFSET ISSUED RETURN
6113	032744	116565	000123	000001		MOVW	P.RCMD(R5),P.CMND(R5)	; LOAD COMMAND
6114	032752	013765	001634	000004		MOV	RTYSEC,P.SECT(R5)	; LOAD SECTOR AND TRACK
6115	032760	013765	001636	000002		MOV	RTYCYL,P.CYLN(R5)	; LOAD CYLINDER
6116	032766	013765	001630	000010		MOV	RTYPA,P.BALO(R5)	; LOAD BUS ADDRESS
6117	032774	013765	001632	000012		MOV	RTYWC,P.WC(R5)	; LOAD WORD COUNT
6118	033002	005465	000012			NEG	P.WC(R5)	; MAKE IT NEGATIVE
6119	033006	122765	000131	000001		CMPB	#WRTCHK,P.CMND(R5)	; CHECK IF WRITE CHECK
6120	033014	001003				BNE	45\$	; NO, ISSUE COMMAND
6121	033016	052765	000200	000014		BIS	#W.WCK,P.PKST(R5)	; YES, SET ISSUE WRITE COMMAND
6122	033024	004037	050706		45\$:	JSR	RO,Q.PUSH	; ENQUEUE PARAMETER BLOCK IN
6123	033030	001324				CINITO		; COMMAND INITIATION QUEUE

```
6124 033032 000207          50$: RTS PC ;RETURN
6125
6126 033034 032765 020400 000034 E2$: BIT #OPTIHVRC,P.ER(R5) ;CHECK IF HEADER TYPE ERROR
6127 033042 001002          BNE 2$ ;YES, PROCESS ERROR
6128 033044 000137 033372          JMP E3$ ;NO, CONTINUE
6129
6130 033050 004737 036150          2$: JSR PC,GETCUR ;GET CURRENT REGISTER STATUS
6131 033054 112737 000001 001640 MOVN #1,RTYSCN ;LOAD RETRY SECTOR COUNT TO 1
6132 033062 016537 000026 001634 MOV P.DTS(R5),RTYSEC ;STORE CURRENT SECTOR AND TRACK
6133 033070 016537 000030 001636 MOV P.DCYL(R5),RTYCYL ;STORE CURRENT SECTOR
6134 033076 016537 000010 001630 MOV P.BALO(R5),RTYBA ;GET INITIAL BUS ADDRESS
6135 033104 016537 000012 001632 MOV P.WC(R5),RTYWC ;GET INITIAL WORD COUNT
6136 033112 005437 001632 MFC RTYWC ;MAKE IT POSITIVE
6137 033116 022737 000400 001632 CMP #256.,RTYWC ;CHECK IF LESS THAN OR EQUAL TO ONE SECTOR
6138 033124 103021          BHS 20$ ;RETRY COMMAND
6139 033126 026565 000026 000004 CMP P.DTS(R5),P.SECT(R5) ;CHECK IF CURRENT SECTOR AND TRACK EQUALS
6140          ; INITIAL TRACK AND SECTOR
6141 033134 001004          BNE 15$ ;NO, CALCULATE DESIRED SECTOR
6142 033136 026565 000030 000002 CMP P.DCYL(R5),P.CYL(R5) ;CHECK IF CURRENT CYLINDER EQUALS
6143          ; INITIAL CYLINDER
6144 033144 001406          BEQ 18$ ;YES, USE INITIAL CYLINDER, TRACK, AND SECTOR
6145 033146 004737 036412          15$: JSR PC,CALWC ;CALCULATE WORD COUNT
6146 033152 022737 000400 001632 CMP #256.,RTYWC ;CHECK IF WORD COUNT LESS THAN OR EQUAL
6147          ; TO ONE SECTOR
6148 033160 103003          BHS 20$ ;YES, USE CALCULATED WORD COUNT
6149 033162 012737 000400 001632 18$: MOV #256.,RTYWC ;USE 256 WORDS
6150 033170 016546 000052          20$: MOV P.B10(R5),-(SP) ;STORE DRIVE CYLINDER ADDRESS
6151 033174 042716 160017          BIC #C<M.CADD>,(SP) ;THROW AWAY PARITY AND MESSAGE TYPE
6152 033200 006216          ASR (SP) ;SHIFT 4 BITS RIGHT
6153 033202 006216          ASR (SP)
6154 033204 006216          ASR (SP)
6155 033206 006216          ASR (SP)
6156 033210 022637 001636          CMP (SP)+,RTYCYL ;CHECK IF CYLINDER IS CORRECT
6157 033214 001420          BEQ 30$ ;YES, ISSUE READ HEADER
6158 033216 004037 030076          JSR R0,PR1200 ;REPORT ERROR
6159 033222 061224          ERR040 ;TYPE "MISPOS"
6160 033224 004037 030570          JSR R0,OPTSTT ;GET CPI/MISPOSITIONING STATISTICS
6161 033230 033370          60$ ;ERROR THRESHOLD EXCEEDED RETURN
6162 033232 052765 004000 000150 BIS #BIT11,P.DSTT(R5) ;SET RECALIBRATE ISSUED
6163 033240 112765 000113 000001 MOVN #RECAL,P.CMND(R5) ;LOAD RECALIBRATE IN PARAMETER BLOCK
6164 033246 004037 050706          JSR R0,Q.PUSH ;PUT PARAMETER BLOCK IN COMMAND
6165 033252 001324          CINITQ ; INITIATION QUEUE
6166 033254 000207          RTS PC ;RETURN
6167
6168 033256 010446          30$: MOV R4, -(SP) ;STORE R4 ON THE STACK
6169 033260 113704 001635          MOVN RTYTRF,R4 ;GET TRACK FOR INDEX
6170 033264 136465 001503 000057 BITB H.HEAD(R4),P.P11+1(R5) ;CHECK IF CORRECT HEAD IS SELECTED
6171 033272 001013          BNE 35$ ;YES, CONTINUE
6172 033274 012604          MOV (SP)+,R4 ;RESTORE R4
6173 033276 004037 030076          JSR R0,PR1200 ;PRINT HEAD SELECT EMPLR
6174 033302 062076          EPR065
6175 033304 004037 030570          JSR R0,OPTSTT ;RECORD AS AN OPI
6176 033310 033370          60$ ;ERROR THRESHOLD EXCEEDED RETURN
6177 033312 004037 050706          JSR R0,Q.PUSH ;REISSUE COMMAND
6178 033316 001324          CINITQ
6179 033320 000207          RTS PC ;RETURN
```

6180								
6181	033322	012604			35\$:	MOV	(SP)+,R4	;RESTORE R4
6182	033324	052765	000020	000150		BIS	#R14,P.DST(R5)	;SET R4 ALL HEADERS ISSUED
6183	033332	013765	001634	000004		MOV	RTYSEC,P.SECT(R5)	;LOAD SECTOR AND TRACK
6184	033340	013765	001634	000002		MOV	RTYCYL,P.CYL(R5)	;LOAD CYLINDER
6185	033346	012765	064776	000010		MOV	#HDRUFF,P.BALC(R5)	;LOAD ADDRESS OF HEADER BUFFER
6186	033354	112765	000164	000001		MOV	#RDALND,P.CMNC(R5)	;LOAD COMMAND
6187	033362	004037	050706			JSR	RO,Q.PUSH	;PUT PARAMETER BLOCK IN COMMAND
6188	033366	001324				CINITO		; INITIATION QUEUE
6189	033370	000207			60\$:	RTS	PC	;RETURN
6190								
6191	033372	032765	000200	000034	E3\$:	BIT	#BSE,P.ER(R5)	;CHECK FOR BAD SECTOR ERROR
6192	033400	001403				BFC	SS	;NO, CONTINUE
6193	033402	004737	041560			JSR	PC,P.DSECT	;SERVICE BAD SECTOR
6194	033406	000207				RTS	PC	;RETURN
6195								
6196	033410	032765	100000	000020	5\$:	BIT	#DLT,P.CS2(R5)	;CHECK IF DATA LATE
6197	033416	001444				BFC	E4\$	;NO, CONTINUE
6198	033420	004737	027242			JSR	PC,PR1000	;TYPE HEADER
6199	033424	032777	020000	145506		BIT	#SW13,@SWP	;CHECK IF INHIBIT PRINT OUT
6200	033432	001004				BNE	10\$	;YES, CHECK IF HALT ON ERROR
6201	033434	104401	061341			TYPE	,ERP047	;TYPE "DATA LATE"
6202	033440	004737	025550			JSR	PC,PR1000	;PRINT CONTROLLER STATUS
6203	033444	004737	030472		10\$:	JSR	PC,PLT000	;CHECK IF HALT ON ERROR
6204	033450	005237	001622			INC	DLTCNT	;INCREMENT DATA LATE COUNT
6205	033454	022737	001000	001627		CMP	#T.DLT,DLTCNT	;CHECK IF THRESHOLD EXCEEDED
6206	033462	103006				BHS	15\$	;NO, REISSUE COMMAND
6207	033464	104401	057554			TYPE	,ERP000	;TYPE FATAL ERROR
6208	033470	104401	060157			TYPE	,FNR009	;TYPE DATA LATE THRESHOLD EXCEEDED
6209	033474	000000				HALT		;HANG
6210	033476	000776				BP	.-2	
6211								
6212	033500	105765	000150		15\$:	TSTP	P.DST(R5)	;CHECK IF FIRST ERROR
6213	033504	100005				BPL	20\$	;NO, REISSUE COMMAND
6214	033506	042765	100200	000150		PLC	#PIT15IR17,P.DST(R5)	;CLEAR ERROR ENQUEUED AND FIRST PEEK
6215	033514	005037	001562			CLR	ERRPR0	;CLEAR ERROR RECOVERY IN PROGRESS
6216	033520	004037	050706		20\$:	JSR	RO,Q.PUSH	;ENQUEUE PARAMETER BLOCK IN COMMAND
6217	033524	001324				CINITO		; INITIATION QUEUE
6218	033526	000207				RTS	PC	;RETURN
6219								
6220	033530	032765	100000	000034	E4\$:	BIT	#DCK,P.ER(R5)	;CHECK IF DATA CHECK
6221	033536	001002				BNE	2\$	;YES, START RECOVERY
6222	033540	000137	034340			JMP	E4\$	;NO, CONTINUE
6223								
6224	033544	112737	000001	001640	2\$:	MOV	#1,RTYSCN	;SET SECTOR COUNT = 1
6225	033552	016537	000004	001634		MOV	P.SECT(R5),RTYSEC	;STORE RETRY SECTOR AND TRACK
6226	033560	016537	000002	001636		MOV	P.CYL(R5),RTYCYL	;STORE RETRY CYLINDER
6227	033566	016537	000010	001630		MOV	P.BALC(R5),RTYBA	;STORE RETRY BUS ADDRESS
6228	033574	016537	000012	001632		MOV	P.WC(R5),RTYWC	;STORE RETRY WORD COUNT
6229	033602	005437	001632			NEG	RTYWC	;MAKE IT POSITIVE
6230	033606	022737	000400	001632		CMP	#256.,RTYWC	;CHECK IF LESS THAN OR EQUAL TO A SECTOR
6231	033614	103043				BHS	15\$	;YES, CHECK IF ECC CORRECTION
6232	033616	016537	000026	001634		MOV	P.DTS(R5),RTYSEC	;STORE PRESENT TRACK AND SECTOR
6233	033624	016537	000030	001636		MOV	P.DCYL(R5),RTYCYL	;STORE PRESENT CYLINDER
6234	033632	105737	001634			TSTB	RTYSEC	;CHECK IF SECTOR = 0
6235	033636	001403				BFC	11\$	;YES, DECREMENT TRACK

6236	033640	105337	001634			DECR	RTVSEC	;DECREMENT SECTOR
6237	033644	000416				BP	13\$	;GO CALCULATE WORD COUNT
6238								
6239	033646	112737	000025	001634	11\$:	MOVR	#21.,RTVSFC	;SET SECTOR = 21
6240	033654	105737	001635			TSTR	RTVTRK	;CHECK IF TRACK = 0
6241	033660	001403				BEQ	12\$	;YES, DECREMENT CYLINDER
6242	033662	105337	001635			DECR	RTVTRK	;DECREMENT TRACK
6243	033666	000405				BP	13\$	;GO CALCULATE WORD COUNT
6244								
6245	033670	112737	000002	001635	12\$:	MOVB	#2,RTVTRK	;SET TRACK = 2
6246	033676	005337	001636			DPC	RTVCYL	;DECREMENT CYLINDER
6247	033702	004737	036412		13\$:	JSR	PC,CALWC	;CALCULATE WORD COUNT
6248	033706	022737	000400	001632		CMP	#256.,RTYWC	;CHECK IF LESS THAN OR EQUAL TO ONE SECTOR
6249	033714	103003				BNIS	15\$	;YES, PERFORM ECC CORRECTION
6250	033716	012737	000400	001637		MOV	#256.,RTYWC	;MAKE IT ONE SECTOR
6251	033724	004037	030076		15\$:	JSR	RO,PR1200	;PRINT HEADER
6252	033730	061355				ERR04H		;PRINT DATA CHECK
6253	033732	105765	000120			TSTB	P.IENC(R5)	;CHECK IF INHIBIT ERROR RECOVERY
6254	033736	001123				BNE	40\$	;YES, INDICATE UNSUCCESSFUL RECOVERY
6255	033740	122765	000121	000123		CMPH	#PDATA,P.RCMD(R5)	;CHECK IF READ DATA
6256	033746	001101				BNE	30\$	;NO, DO NOT DO ECC CORRECTION
6257	033750	032765	000100	000034		BIT	#ECC,P.ER(R5)	;CHECK IF CORRECTABLE
6258	033756	001075				BNE	30\$	;NO, DO NOT DO ECC CORRECTION
6259	033760	022765	010040	000060		CMP	#10040,P.EPOS(R5)	;CHECK IF ECC POSITION TOO LARGE
6260	033766	101406				BLOS	16\$	;YES, REPORT ERROR
6261	033770	005765	000060			TST	P.EPOS(R5)	;CHECK IF ECC POSITION = 0
6262	033774	001403				BEQ	16\$	;YES, REPORT ERROR
6263	033776	005765	000062			TST	P.EPAT(R5)	;CHECK IF ECC PATTERN = 0
6264	034002	001004				BNE	17\$	;NO, DO ECC CORRECTION
6265	034004	004037	036524		16\$:	JSR	RO,CONERR	;TYPE "ECC LOGIC ERROR"
6266	034010	062121				ERR066		
6267	034012	000207				RTS	PC	;RETURN
6268								
6269	034014	004737	025634		17\$:	JSR	PC,ECCCOR	;DO ECC CORRECTION
6270	034020	105765	000210			TSTP	P.ASSN(R5)	;CHECK IF IN ASSIGNMENT SEQUENCE
6271	034024	001017				BNE	18\$	;YES, DO NOT CHECK ECC CORRECTION
6272	034026	013765	001630	000214		MOV	RTYRA,P.ERHD(R5)	;LOAD START OF SECTOR FOR
6273								; DATA COMPARISON
6274	034034	013765	001632	000222		MOV	RTYWC,P.BFCP(R5)	;LOAD NUMBER OF WORDS OF
6275	034042	013765	001632	000220		MOV	RTYWC,P.CMLG(R5)	; SECTOR PRESENT
6276	034050	112765	177777	000117		MOVR	#-1,P.ECMP(R5)	;SET ECC COMPARE FLAG
6277	034056	004737	023036			JSR	PC,CMPHUF	;CHECK ECC CORRECTION
6278	034062	034072				20\$		;INDICATE ECC MISCORRECTION
6279	034064	004737	035500		18\$:	JSR	PC,SUPRCY	;INDICATE SUCCESSFUL RECOVERY
6280	034070	000207				RTS	PC	;RETURN
6281								
6282	034072	032777	020000	145040	20\$:	BIT	#SW13,#SMF	;CHECK IF INHIBIT ERROR PRINT OUT
6283	034100	001024				BNE	30\$	;YES, GO TO RECOVERY SEQUENCE
6284	034102	104401	062015			TYPE	,ERR060	;TYPE "ECC MISCORRECTION"
6285	034106	104401	063274			TYPE	,ERR211	;TYPE "RKPC"
6286	034112	016546	000060			MOV	P.EPOS(R5),-(SP)	;STORE RFPPOS FOR PRINT OUT
6287	034116	004737	052144			JSR	PC,RINOCT	;CONVERT TO OCTAL
6288	034122	104401	001165			TYPE	,SCRLF	;TYPE <CR><LF>
6289	034126	104401	063303			TYPE	,ERR212	;TYPE "RKPAT"
6290	034132	016546	000062			MOV	P.EPAT(R5),-(SP)	;STORE RFPAT FOR PRINT OUT
6291	034136	004737	052144			JSR	PC,BINOCT	;CONVERT TO OCTAL

ADDRESS	INSTR	OPCODE	DISP	DATA	COMMENT
6292	034147	104401	001165		TYPE <CR><LF>
6293	034146	004737	035226		PRINT WORD IN ERROR
6294	034152	105265	000147	30\$:	INCB P.RERD(R5) ;INCREMENT RETRY COUNT
6295	034156	122765	000043	000147	CMPR #43,P.RERD(R5) ;CHECK IF UNSUCCESSFUL RETRY
6296	034164	101410			BLOS 40\$;YES, INDICATE UNSUCCESSFUL RETRY
6297	034166	122765	000121	000123	CMPB #PDATA,P.RCMD(R5) ;CHECK IF READ DATA
6298	034174	001407			BFC 45\$;YES, CHECK FOR OFFSET
6299	034176	122765	000021	000147	CMPB #21,P.RERD(R5) ;CHECK IF ABOUT TO ENTER OFFSET OPERATIONS
6300	034204	101021			BHI 50\$;NO, DO REPAIR
6301	034206	004737	030724	40\$:	JSR PC,UNRECV ;INDICATE UNSUCCESSFUL RECOVERY
6302	034212	000207			RTS PC ;RETURN
6303					
6304	034214	122765	000021	000147	45\$:
6305	034222	001007			CMPB #21,P.RERD(R5) ;CHECK IF OFFSET +400 MICRONS-INCHES
6306					BNE 48\$;NO, GO DETERMINE IF OFFSET IS TO
6307	034224	010446			MOV R4,-(SP) ;STORE R4 ON STACK
6308	034226	116504	000000		MOVB P.DRVN(R5),R4 ;GET DRIVE NUMBER AS INDEX
6309	034232	146437	001510	001507	BITB INTMSK(R4),O.OVER ;RESET ISSUE IMPLIED SEEK
6310	034240	012604			MOV (SP)+,R4 ;RESTORE R4
6311	034242	004037	036230	48\$:	JSR R0,DETOFF ;DETERMINE IF OFFSET IS TO BE ISSUED
6312	034246	034336			BOS ;OFFSET ISSUED RETURN
6313	034250	116565	000123	000001	50\$:
6314	034256	013765	001634	000004	MOVB P.RCMD(R5),P.CMND(R5) ;REISSUE LAST COMMAND
6315	034264	013765	001636	000002	MOV RTYSEC,P.SECT(R5) ;LOAD SECTOR AND TRACK
6316	034272	013765	001630	000010	MOV RTYCYL,P.CYLN(R5) ;LOAD CYLINDER
6317	034300	013765	001632	000012	MOV RTYBA,P.BALO(R5) ;LOAD BUS ADDRESS
6318	034306	005465	000012		MOV RTYWC,P.WC(R5) ;LOAD WORD COUNT
6319	034312	122765	000131	000001	NEG P.WC(R5) ;MAKE IT NEGATIVE
6320	034320	001003			CMPB #WRTCHK,P.CMND(R5) ;CHECK IF WRITE CHECK
6321	034322	052765	000200	000014	BNE 55\$;NO, GO ISSUE COMMAND
6322	034330	004037	050706	55\$:	BIS #W.WCK,P.PRST(R5) ;ISSUE WRITE BEFORE WRITE CHECK
6323	034334	001324			JSR R0,Q.PUSH ;PUT PARAMETER BLOCK ON COMMAND
6324	034336	000207		60\$:	CINITQ ;COMMAND INITIATION QUEUE
6325					RTS PC ;RETURN
6326	034340	032765	040000	000020	25\$:
6327	034346	001002			BIT #WCE,P.CS2(R5) ;CHECK IF WRITE CHECK ERROR
6328	034350	000137	034512		BNE 1\$;YES, PROCESS WRITE CHECK ERROR
6329					JMP E10\$;NO, CONTINUE
6330	034354	004037	030076	1\$:	JSR R0,PR1200 ;PRINT WRITE CHECK ERROR
6331	034360	061367			EPR050
6332	034362	105765	000120		TSTR P.IFRC(R5) ;CHECK IF INHIBIT ERROR CORRECTION AND RETRY
6333	034366	001006			BNE 5\$;YES, INITIATE NEXT COMMAND
6334	034370	105265	000147		INCB P.RFRD(R5) ;INCREMENT RETRY COUNT
6335	034374	122765	000021	000147	CMPB #21,P.RERD(R5) ;CHECK IF 16 REREADS OCCURRED
6336	034402	101003			BHI 10\$;NO, START RECOVERY SEQUENCE
6337	034404	004737	030724	5\$:	JSR PC,UNRECV ;INDICATE UNSUCCESSFUL RECOVERY
6338	034410	000207			RTS PC ;RETURN
6339					
6340	034412	016537	000024	001630	10\$:
6341	034420	162737	000002	001630	MOV P.BAR(R5),RTYBA ;GET ADDRESS OF WORD IN ERROR
6342	034426	004737	035014		SUB #2,RTYBA ;ADJUST ADDRESS
6343	034432	116565	000123	000001	JSR PC,ALSEC ;CALCULATE SECTOR, TRACK AND CYLINDER
6344	034440	013765	001634	000004	MOVB P.RCMD(R5),P.CMND(R5) ;GET PREVIOUS COMMAND
6345	034446	013765	001636	000002	MOV RTYSEC,P.SECT(R5) ;GET SECTOR AND TRACK
6346	034454	013765	001630	000010	MOV RTYCYL,P.CYLN(R5) ;GET CYLINDER
6347	034462	013765	001632	000012	MOV RTYBA,P.BALO(R5) ;GET BUS ADDRESS
					MOV RTYWC,P.WC(R5) ;GET WORD COUNT

6348	034470	005465	000012			NEG	P.WC(R5)	;MAKE IT NEGATIVE
6349	034474	052765	000200	000014		BIS	#W.WCF,P.PAST(R5)	;SET ISSUP WAIT BEFORE WHITE CHECK
6350	034502	004037	050706			JSR	RO,Q.PUSH	;ENQUEOF PARAMETER IN
6351	034506	001324				CYINITQ		; COMMAND INITIATION QUEUE
6352	034510	000207				RTS	PC	;RETURN
6353								
6354								
6355	034512	032765	000004	000712	E10\$:	BIT	#BIT2,P.ERR(R5)	;CHECK IF DATA COMPARE ERROR
6356	034520	001001				BNE	2\$	;YES, CONTINUE PROCESSING ERROR
6357	034522	000000				HALT		; *** UNFNCN ENTRY
6358	034524	005046			2\$:	CLR	-(SP)	;MAKE ROOM ON STACK
6359	034526	116516	000217			MOVW	P.ERAD(R5),(SP)	;GET WORD IN ERROR
6360	034532	006316				ASL	(SP)	;MULTIPLY BY 2
6361	034534	066516	000214			ADD	P.EPHD(R5),(SP)	;CALCULATE ADDRESS OF MISCOMPARE WORD
6362	034540	012637	001630			MOV	(SP)+,RTYBA	;LOAD ADDRESS OF RAD WORD
6363	034544	004737	035014			JSR	PC,CALSPC	;CALCULATE CYLINDER, TRACK, AND SECTOR
6364	034550	032765	000002	000150		BIT	#BIT1,P.DSTT(R5)	;CHECK IF DATA TYPE ERROR HAS ALREADY OCCURRED
6365	034556	001037				BNE	10\$	;YES, GO TO RETRY SEQUENCE
6366	034560	052765	000002	000150		BIS	#BIT1,P.DSTT(R5)	;SET DATA TYPE ERROR OCCURRED
6367	034566	100003				BPL	3\$	;CHECK IF FIRST ERROR
6368	034570	052765	000001	000150		BIS	#BIT0,P.DSTT(R5)	;SET INITIAL DATA ERROR
6369	034576	004737	027242		3\$:	JSR	PC,PR1000	;PRINT HEADER
6370	034602	032777	020000	144330		BIT	#SW13,#SWP	;CHECK IF INHIBIT PRINT COT
6371	034610	001013				BNE	5\$	;YES, CHECK IF HALT ON ERROR
6372	034612	104401	062041			TYPE	,ERR062	;TYPE DATA COMPARE ERROR
6373	034616	005765	000150			TST	P.DSTT(P5)	;CHECK IF FIRST ERROR
6374	034622	100402				BMI	4\$	;YES, GO PRINT ERROR WORD
6375	034624	104401	062712			TYPE	,ERR202	;TYPE ERROR DURING ERROR RECOVERY
6376	034630	004737	035226		4\$:	JSR	PC,PRERWD	;PRINT WORD IN ERROR
6377	034634	004737	027476			JSR	PC,PR1002	;PRINT CURRENTLY SUPPLIED COMMAND
6378	034640	042765	100200	000150	5\$:	BYC	#BIT15#BIT7,P.DSTT(P5)	;CLEAR FIRST ERROR AND ERROR ENQUEOF
6379	034646	004737	026704			JSR	PC,PRIBUF	;GO PRINT BUFFER
6380	034652	004737	030472			JSR	PC,PLT000	;CHECK IF HALT ON ERROR
6381	034656	105765	000120		10\$:	TSTP	P.IERC(R5)	;CHECK IF INHIBIT ERROR CORRECTION AND RECOVERY
6382	034662	001006				BNE	15\$	;YES, INDICATE UNSUCCESSFUL RECOVERY
6383	034664	105265	000147			INCP	P.RERD(R5)	;INCREMENT RETRY COUNT
6384	034670	172765	000043	000147		CMPP	#43,P.RERD(R5)	;CHECK IF ERROR RECOVERY UNSUCCESSFUL
6385	034676	101003				BHI	20\$	;NO, GO TO RECOVERY SEQUENCE
6386	034700	004737	030724		15\$:	JSR	PC,UNPECY	;INDICATE UNSUCCESSFUL RECOVERY
6387	034704	000207				RTS	PC	;RETURN
6388								
6389	034706	122765	000021	000147	20\$:	CMPP	#21,P.RERD(R5)	;CHECK IF EXISTING OFFSET
6390	034714	001007				BNE	25\$	;NO, DETERMINE IF TIME TO CHANGE OFFSET
6391	034716	010446				MOV	R4,-(SP)	;STORE R4 ON STACK
6392	034720	116504	000000			MOVW	P.DRVN(R5),R4	;GET DRIVE NUMBER FOR INDEX
6393	034724	146437	001510	001507		BICB	INTMSK(R4),O.OVER	;RESET SEEK BEFORE DATA TRANSFER
6394	034732	012604				MOV	(SP)+,R4	;RESTORE R4
6395	034734	004037	036230		25\$:	JSR	RO,DET0FF	;DETERMINE IF OFFSET TO BE ISSUED
6396	034740	035012				50\$		;OFFSET HAS BEEN ISSUED
6397	034742	116565	000123	000001		MOVW	P.RCMD(R5),P.CMND(R5)	;GET LAST ISSUED COMMAND
6398	034750	013765	001634	000004		MOV	RTYSEC,P.SECT(R5)	;LOAD SECTOR AND TRACK ADDRESS
6399	034756	013765	001636	000002		MOV	RTYCYL,P.CYLN(R5)	;LOAD CYLINDER ADDRESS
6400	034764	013765	001630	000010		MOV	RTYPA,P.BALO(R5)	;LOAD PUS ADDRESS
6401	034772	013765	001632	000012		MOV	RTYWC,P.WC(R5)	;GET WORD COUNT
6402	035000	005465	000012			NEG	P.WC(R5)	;MAKE IT NEGATIVE
6403	035004	004037	050706			JSR	RO,Q.PUSH	;ENQUEOF PARAMETER BLOCK IN COMMAND

RF611/RK06 PERFORMANCE EXERCISER MAINDEC DZR6P-B MACV11 30(1046) 01-DEC-77 14:15 PAGE 139
DZR6P8.P11 11-AUG-76 00:00 ERROR RECOVERY SEQUENCE

SEQ 0161

6404 035010 001324
6405 035012 000207

SOS: CINITO
RTS PC

; INITIATION QUEUE
;RETURN

E13

```

6406 .SBTTL CALCULATE CYLINDER, TRACK, AND SECTOR FOR RETRY
6407
6408 035014 010146 CALSEC: MOV R1, -(SP) ;STORE R1 ON STACK
6409 035016 010346 MOV R3, -(SP) ;STORE R3 ON STACK
6410 035020 013703 001630 MOV RTYBA, R3 ;GET ADDRESS OF ERROR WORD
6411 035024 166503 000010 SUB P.BALC(R5), R3 ;CALCULATE BYTES CORRECTLY TRANSFERRED
6412 035030 042703 000777 BIC #777, R3 ;MAKE IT ON SECTOR BOUNDARY
6413 035034 010337 001630 MOV R3, RTYBA ;STORE CALCULATED BUS ADDRESS
6414 035040 066537 000010 001630 ADD P.BALC(R5), RTYBA ;CALCULATE START OF SECTOR WITH ERROR
6415 035046 016501 000012 MOV P.WC(R5), P1 ;GET INITIAL WCRC COUNT
6416 035052 005401 NEG R1 ;MAKE IT POSITIVE
6417 035054 006203 ASK R3 ;DETERMINE WORDS BEFORE ERROR
6418 035056 160301 SUB R3, R1 ;CALCULATE NUMBER OF WORDS LEFT
6419 035060 022701 000400 CMP #256., R1 ;CHECK IF LESS THAN OR EQUAL TO ONE SECTOR
6420 035064 103002 BHS 5$ ;LOAD RETRY WORD COUNT
6421 035066 012701 000400 MOV #256., R1 ;NO, MAKE IT ONE SECTOR
6422 035072 010137 001632 5$: MOV R1, RTYWC ;LOAD RETRY WORD COUNT
6423 035076 116537 000005 001635 MOV P.TPC(R5), RTYTRK ;STORE RETRY TRACK
6424 035104 016537 000002 001636 MOV P.CYL(R5), PTYCYL ;STORE RETRY CYLINDER
6425 035112 010346 MOV R3, -(SP) ;STORE WORDS TRANSFERRED
6426 035114 116616 000001 MOV R1, (SP), (SP) ;STORE NUMBER OF SECTORS
6427 035120 105066 000001 CLR 1(SP) ;CLEAR HIGH ORDER BYTE
6428 035124 005046 CLR -(SP) ;MAKE ROOM ON STACK
6429 035126 116516 000004 MOV P.SFCT(R5), (SP) ;GET INITIAL SECTOR
6430 035132 062616 ADD (SP)+, (SP) ;ADD SECTORS TRANSFERRED
6431 035134 022716 000025 10$: CMP #21., (SP) ;CHECK IF SECTOR COUNT GREATER THAN 21
6432 035140 103005 BHS 15$ ;NO, CHECK TRACK
6433 035142 162716 000026 SUB #22., (SP) ;SUBTRACT 22 SECTORS/TRACK
6434 035146 105237 001635 INC RTYTRK ;INCREMENT TRACK
6435 035152 000770 BP 10$ ;CHECK SECTOR COUNT
6436
6437 035154 112637 001634 15$: MOV (SP)+, RTYSEC ;STORE RETRY SECTOR
6438 035160 005046 CLR -(SP) ;MAKE ROOM ON THE STACK
6439 035162 113716 001635 MOV RTYTRK, (SP) ;STORE RETRY TRACK
6440 035166 122716 000002 17$: CMP #2., (SP) ;CHECK TRACK GREATER THAN 2
6441 035172 103005 BHS 20$ ;NO, RETURN
6442 035174 162716 000003 SUB #3., (SP) ;SUBTRACT 3 TRACKS/CYLINDER
6443 035200 005237 001636 INC RTYCYL ;INCREMENT CYLINDER
6444 035204 000770 BP 17$ ;CHECK TRACK COUNT
6445
6446 035206 112637 001635 20$: MOV (SP)+, RTYTRK ;LOAD RETRY COUNT
6447 035212 112737 000001 001640 MOV #1, PTYSCN ;INDICATE ONE SECTOR IN RETRY COUNT
6448 035220 012603 MOV (SP)+, R3 ;RESTORE R3
6449 035222 012601 MOV (SP)+, R1 ;RESTORE R1
6450 035224 000207 RTS PC ;RETURN
  
```

6451					.SBTTL PRINT FIRST ERROR WORD IN BUFFER
6452					
6453	035226	010046			PRERWD: MOV R0,-(SP) ;STORE R0 ON STACK
6454	035230	010146			MOV R1,-(SP) ;STORE R1 ON STACK
6455	035232	010346			MOV R3,-(SP) ;STORE R3 ON STACK
6456	035234	010446			MOV R4,-(SP) ;STORE R4 ON STACK
6457	035236	104401	060214		TYPE ,FRR012 ;TYPE ERROR HEADER
6458	035242	016503	000214		MOV P.ERHD(P5),R3 ;GET ADDRESS OF START OF SECTOR
6459	035246	005046			CLR -(SP) ;MAKE ROOM ON STACK
6460	035250	116516	000217		MOVR P.ERAD(P5),(SP) ;GET WORD IN PERCP OF SECTOR
6461	035254	006316			ASL (SP) ;MULTIPLY IT BY 2
6462	035256	011601			MOV (SP),R1 ;STORE IT IN R1
6463	035260	062603			ADD (SP)+,R3 ;GET ADDRESS OF BAD WORD
6464	035262	010346			MOV R3,-(SP) ;STORE ADDRESS FOR PRINT OUT
6465	035264	004737	052144		JSR PC,RINOC ;CONVERT TO CCTL
6466	035270	122765	000001	000217	CMR #1,P.ERAD(P5) ;CHECK IF IN DESCRIPTOR AREA
6467	035276	103403			BLO SS ;NO, CALCULATE WORD
6468	035300	104401	057536		TYPE ,QUFSMK ;TYPE QUESTION MARKS
6469	035304	000436			BR 10\$;GO TYPE WORD READ
6470					
6471	035306	016504	000214		SS: MOV P.ERHD(P5),R4 ;GET HEADER FOR PATTERN INDEX
6472	035312	016400	000002		MOV 2(R4),R0 ;GET NUMBER OF WORDS WRITTEN IN SECTOR
6473	035316	042700	177400		BIC #177400,R0 ;FREE LOW BYTE
6474	035322	005200			INC R0 ;INCREMENT R0 TO MAKE OFFSET FROM START
6475					; OF SECTOR
6476	035324	126500	000217		CMR P.ERAD(P5),R0 ;CHECK IF IN ZERO FILL AREA
6477	035330	101402			BLOS 7\$;NO, CALCULATE ADDRESS OF
6478					; WORD IN DATA PATTERN
6479	035332	005046			CLR -(SP) ;LOAD ZERO FOR PRINT OUT
6480	035334	000414			BR 9\$;GO PRINT EXPECTED CONTENTS
6481					
6482	035336	011404			7\$: MOV (R4),R4 ;GET DATA PATTERN NUMBER FOR INDEX
6483	035340	042704	177760		BIC #177760,R4 ;MASK OFF INSIGNIFICANT BITS
6484	035344	006304			ASL R4 ;MULTIPLY BY 2
6485	035346	016404	004576		MOV PATTAB(R4),R4 ;GET PATTERN BASE
6486	035352	162701	000004		SUB #4,R1 ;SUBTRACT 4 FROM SECTOR OFFSET
6487	035356	042701	177700		BIC #177700,R1 ;MAKE IT MOD 32 WORDS
6488	035362	060104			ADD R1,R4 ;CALCULATE PATTERN ADDRESS
6489	035364	011446			MOV (R4),-(SP) ;STORE EXPECTED CONTENTS FOR PRINT OUT
6490	035366	104401	057551		9\$: TYPE ,BLANKS ;TYPE BLANKS
6491	035372	004737	052144		JSR PC,BINOC ;CONVERT TO CCTL
6492	035376	104401	057551		TYPE ,BLANKS ;TYPE BLANKS
6493	035402	011346			10\$: MOV (R3),-(SP) ;STORE WORD READ
6494	035404	004737	052144		JSR PC,RINOC ;CONVERT TO CCTL
6495	035410	104401	057551		TYPE ,BLANKS ;TYPE BLANKS
6496	035414	005046			CLR -(SP) ;MAKE ROOM ON STACK
6497	035416	116516	000217		MOVR P.ERAD(P5),(SP) ;STORE SECTOR WORD NUMBER FOR PRINT OUT
6498	035422	004737	052144		JSR PC,RINOC ;CONVERT TO CCTL
6499	035426	105765	000217		TST P.ERAD(P5) ;CHECK IF ERAD IS IN PATTERN WORD
6500	035432	001003			BNE 15\$;NO, PRINT PATTERN NUMBER
6501	035434	104401	057536		TYPE ,QUFSMK ;TYPE QUESTION MARKS
6502	035440	000410			BR 20\$;GO RESTORE REGISTERS
6503					
6504	035442	104401	057551		15\$: TYPE ,BLANKS ;TYPE BLANKS
6505	035446	017546	000214		MOV @P.ERHD(P5),-(SP) ;GET FIRST WORD OF SECTOR HEAD
6506	035452	042716	177760		BIC #177760,(SP) ;CLEAR INSIGNIFICANT BITS

6507	035456	004737	052144		JSR	PC,PINOC	;CONVERT TO CCTL
6508	035462	104401	001165	205:	TYPE	,SCRLF	;TYPE <CR><LF>
6509	035466	012604			MOV	(SP)+,R4	;RESTORE R4
6510	035470	012603			MOV	(SP)+,R3	;RESTORE R3
6511	035472	012601			MOV	(SP)+,R1	;RESTORE R1
6512	035474	012600			MOV	(SP)+,R0	;RESTORE R0
6513	035476	000207			RTS	PC	;RETURN

```

6514 .SBTTL SUCCESSFUL RECOVERY CLEAN UP
6515
6516 035500 032765 000001 000150 SURECY: BIT #BIT0,P.DSTT(R5) ;CHECK IF DATA TYPE ERROR
6517 035506 001466 BEQ 10$ ;NO, CLEAN UP FOR NEXT COMMAND
6518 035510 105765 000147 TSTP P.RERD(R5) ;CHECK IF FIRST ERROR
6519 035514 001003 BNE 5$ ;NO, CHECK IF OFFSET
6520 035516 005265 000176 INC P.SFCC(R5) ;INCREMENT NO. OF ECC RECOVERABLE ERRORS
6521 035522 000426 BR 9$ ;GO CLEAN UP FOR NEXT COMMAND
6522
6523 035524 122765 000020 000147 5$: CMPB #20,P.RERD(R5) ;CHECK IF OFFSET
6524 035532 103403 BLO 8$ ;YES, INCREMENT OFFSET RECOVERABLE
6525 035534 005265 000172 INC P.SRRD(R5) ;INCREMENT REEAD RECOVERABLE
6526 035540 000417 BP 9$ ;GO CLEAN UP FOR NEXT COMMAND
6527
6528 035542 005265 000174 8$: INC P.SOFF(R5) ;INCREMENT OFFSET RECOVERABLE
6529 035546 032777 020000 143364 BIT #SW13,0SWR ;CHECK IF INHIBIT PRINT OUT
6530 035554 001011 BNE 9$ ;YES, CLEAN UP FOR NEXT COMMAND
6531 035556 104401 062543 TYPF ,ERR108 ;TYPE "OFFSET"
6532 035562 005046 CLR -(SP) ;MAKE ROOM ON STACK
6533 035564 116516 000006 MOVB P.CPST(R5),(SP) ;STORE CPFSPT FOR PRINT OUT
6534 035570 004737 052144 JSR PC,BINOC ;CONVERT TO OCTAL
6535 035574 104401 001165 TYPE ,SCRLF ;TYPE <CR><LF>
6536 035600 105037 001640 9$: CLRB RTYSCN ;CLEAR SECTOR RETRY FLAG
6537 035604 016546 000176 MOV P.SFCC(R5),-(SP) ;STORE NUMBER OF ECC RECOVERABLE ERRORS
6538 035610 066516 000172 ADD P.SRRD(R5),(SP) ;ADD NUMBER OF REEAD RECOVERABLE ERRORS
6539 035614 066516 000174 ADD P.SOFF(R5),(SP) ;ADD NUMBER OF OFFSET RECOVERABLE ERRORS
6540 035620 022665 000100 CMP (SP)+,P.CPST(R5) ;CHECK IF SCPT ERROR RATE EXCEEDED
6541 035624 101417 BLOS 10$ ;NO, GO ISSUE NEXT COMMAND
6542 035626 032777 046020 143304 BIT #SW41SW14,0SWR ;CHECK IF INHIBIT AUTOMATIC DRIVE DROPPING
6543 035634 001013 BNE 10$ ;YES, GO ISSUE NEXT COMMAND
6544 035636 004737 02231C JSR PC,BUFREL ;GO RELEASE BUFFER
6545 035642 104401 063040 TYPE ,ERR205 ;TYPE "ERROR THRESHOLD EXCEEDED"
6546 035646 004737 012716 JSR PC,DROP ;GO DROP DRIVE
6547 035652 005037 001562 CLR ERRPRO ;CLEAR ERROR CURRENTLY BEING PROCESSED
6548 035656 004737 022016 JSR PC,GETBUF ;GET BUFFER FOR NEXT COMMAND
6549 035662 000207 RTS PC ;RETURN
6550
6551 035664 032777 020000 143246 10$: BIT #SW13,0SWR ;CHECK IF INHIBIT PRINT OUT
6552 035672 001002 BNE 15$ ;YES, CHECK IF READ BACK SERIAL NUMBER
6553 035674 104401 062750 TYPE ,ERR203 ;TYPE RECOVERABLE ERROR
6554 035700 105037 001640 15$: CLRB RTYSCN ;CLEAR SECTOR RETRY COUNT
6555 035704 005065 000146 CLR P.RFCT(R5) ;CLEAR ERROR FLAGS AND RETRY COUNTS
6556 035710 005065 000150 CLR P.DSTT(R5)
6557 035714 005065 000212 CLR P.ERR(R5)
6558 035720 005037 001562 CLR ERRPRO ;CLEAR ERROR CURRENTLY BEING PROCESSED
6559 035724 010446 MOVB R4,-(SP) ;STORE R4 ON STACK
6560 035726 116504 000000 MOVB P.DRVN(R5),R4 ;STORE DRIVE NUMBER FOR INDEX
6561 035732 156437 001510 001507 BITSP INTMSF(R4),0.CVER ;ALLOW OVERLAPPED OPERATIONS ON THIS DRIVE
6562 035740 012604 MOV (SP)+,R4 ;RESTORE R4
6563 035742 105765 000210 TSTB P.ASSI(R5) ;CHECK IF IN ASSIGNMENT SEQUENCE
6564 035746 001025 BNE 20$ ;YES, TERMINATE SEQUENCE
6565 035750 122765 000117 000001 CMPB #SEEK,P.CMND(R5) ;CHECK IF SEEK TO PREVIOUS CYLINDER
6566 035756 001410 BEQ 16$ ;YES, GO ISSUE RELEASE
6567 035760 122765 000140 000001 CMPB #RELEASE,P.CMND(R5) ;CHECK IF RELEASE
6568 035766 001013 BNE 17$ ;NO, CARRY OUT NORMAL RECOVERY
6569 035770 004037 050706 JSR R0,Q.PUSH ;ENQUEUE PARAMETER BLOCK IN DRIVE

```

6570	035774	001314				AVAILQ		; AVAILIABLE QUEUE
6571	035776	000207				RTS	PC	;RETURN
6572								
6573	036000	112765	000140	000001	16\$:	MOVB	#RELEAS,P.CMND(R5)	;ISSUE RELEASE COMMAND
6574	036006	004037	050706			JSR	RO,Q.PUSH	;ENQUEUE PARAMETER BLOCK IN COMMAND
6575	036012	001324				CINITQ		; INITIATION QUEUE
6576	036014	000207				RTS	PC	;RETURN
6577								
6578	036016	000137	024272		17\$:	JMP	NORM1	;CAREY OUT NORMAL DATA TRANSFER TERMINATION
6579								
6580								CHECK IF ANY OF THE FOLLOWING ASSIGNMENT STAGES
6581								READ DRIVE STATUS
6582								START SPINDLE
6583								SET VOLUME VALID
6584								RECAL
6585								READ PACK SERIAL NUMBER
6586	036022	132765	000117	000210	20\$:	BITB	#BIT0BIT1BIT2BIT3BIT6,P.ASSN(R5)	
6587	036030	001402				BFG	25\$	;NO, PROCESS WRITE PACK
6588	036032	000137	013370			JMP	ASNM	;PROCESS NORMAL TERMINATION
6589								
6590	036036	032765	002000	000014	25\$:	BIT	#DRPDIV,P.PRST(R5)	;CHECK IF DROP DRIVE
6591	036044	001030				BNE	30\$	;YES, GO DROP DRIVE
6592	036046	105065	000211			CLRB	P.SEEK(R5)	;CLEAR SEEK TO PREVIOUS LOCATION
6593	036052	116565	000123	000001		MOVB	P.RCMD(R5),P.CMND(R5)	;REISSUE LAST WRITE/WRITE CHECK COMMAND
6594	036060	052765	000200	000014		BIS	#W.MCK,P.PRST(R5)	;SET ISSUE WRITE BEFORE WRITE CHECK
6595	036066	016565	000126	000004		MOV	P.RSEC(R5),P.SECT(R5)	;LOAD PREVIOUS TRACK AND SECTOR
6596	036074	016565	000124	000002		MOV	P.RCYL(R5),P.CYLN(R5)	;LOAD PREVIOUS CYLINDER
6597	036102	016565	000132	000010		MOV	P.RBAL(R5),P.PALG(R5)	;LOAD PREVIOUS RUS ADDRESS
6598	036110	016565	000130	000012		MOV	P.RWC(R5),P.WC(R5)	;LOAD PREVIOUS WORD COUNT
6599	036116	004037	050706			JSR	RO,Q.PUSH	;ENQUEUE PARAMETER BLOCK IN COMMAND
6600	036122	001324				CINITQ		; INITIATION QUEUE
6601	036124	000207				RTS	PC	;RETURN
6602								
6603	036126	004737	022316		30\$:	JSR	PC,BUFREL	;GO RELEASE BUFFER
6604	036132	104401	055347			TYPE	,OPR008	;TYPE "OPERATOR INITIATED DROP DRIVE
6605								; DURING PACK WRITING"
6606	036136	004737	012716			JSR	PC,DROP	;GO DROP DRIVE
6607	036142	004737	022016			JSR	PC,GETBUF	;GET BUFFER FOR NEXT COMMAND
6608	036146	000207				RTS	PC	;RETURN

```

6609          .SBTTL  GET CURRENT STATUS INFORMATION
6610
6611 036150 010446   GETCUR: MOV    R4,-(SP)      ;STORE R4 ON STACK
6612 036152 010346   MOV    R3,-(SP)      ;STORE R3 ON STACK
6613 036154 010504   MOV    R5,R4        ;LOAD START OF CURRENT PARAMETER BLOCK
6614 036156 012703 004512  MOV    @PARM10,R3    ;LOAD START OF RETRY PARAMETER BLOCK
6615
6616 036162 012423   SS:    MOV    (R4)+,(R3)+    ;COPY PARAMETER BLOCK
6617 036164 022703 004576  CMP    @PARM10+P.EPAT+2,R3 ;CHECK IF FINISHED
6618 036170 001374   BNE     SS              ;NO, GET NEXT WORD
6619 036172 012603   MOV    (SP)+,R3        ;RESTORE R3
6620 036174 012604   MOV    (SP)+,R4        ;RESTORE R4
6621 036176 000207   RTS     PC              ;RETURN
6622
6623          .SBTTL  RSTORE STATUS
6624
6625 036200 010446   RSTAT: MOV    R4,-(SP)      ;STORE R4 ON STACK
6626 036202 010346   MOV    R3,-(SP)      ;STORE R3 ON STACK
6627 036204 010504   MOV    R5,R4        ;LOAD START OF CURRENT PARAMETER BLOCK
6628 036206 012703 004512  MOV    @PARM10,R3    ;LOAD START OF RETRY PARAMETER BLOCK
6629
6630 036212 012324   SS:    MOV    (R3)+,(R4)+    ;COPY PARAMETER BLOCK
6631 036214 022703 004576  CMP    @PARM10+P.EPAT+2,R3 ;CHECK IF FINISHED
6632 036220 001374   BNE     SS              ;NO, GET NEXT WORD
6633 036222 012603   MOV    (SP)+,R3        ;RESTORE R3
6634 036224 012604   MOV    (SP)+,R4        ;RESTORE R4
6635 036226 000207   RTS     PC              ;RETURN

```

```

6636          .SBTTL DETERMINE OFFSET FOR RETRY
6637
6638 036230 122765 000021 000147 DETOFF: CMPR    #21,P.READ(R5) ;CHECK IF OFFSET = +400 MICRO-INCHES
6639 036236 001004          BNE      10$          ;NO, CONTINUE
6640 036240 112765 000020 000006          MOVR    #OFFP4,P.OFST(R5) ;LOAD OFFSET VALUE = +400 MICRC-INCHES
6641 036246 000447          BR       15$          ;ISSUE OFFSET
6642
6643 036250 122765 000024 000147 10$:  CMPR    #24,P.READ(R5) ;CHECK IF OFFSET = -400 MICRO-INCHES
6644 036256 001004          BNE      11$          ;NO, CONTINUE
6645 036260 112765 000220 000006          MOVR    #OFFM4,P.OFST(R5) ;LOAD OFFSET VALUE = -400 MICRC-INCHES
6646 036266 000437          BR       15$          ;ISSUE OFFSET
6647
6648 036270 122765 000027 000147 11$:  CMPR    #27,P.READ(R5) ;CHECK IF OFFSET = +800 MICRO-INCHES
6649 036276 001004          BNE      12$          ;NO, CONTINUE
6650 036300 112765 000040 000006          MOVR    #OFFP8,P.OFST(R5) ;LOAD OFFSET VALUE = +800 MICRC-INCHES
6651 036306 000427          BR       15$          ;ISSUE OFFSET
6652
6653 036310 122765 000032 000147 12$:  CMPR    #32,P.READ(R5) ;CHECK IF OFFSET = -800 MICRO-INCHES
6654 036316 001004          BNE      13$          ;NO, CONTINUE
6655 036320 112765 000240 000006          MOVR    #OFFM8,P.OFST(R5) ;LOAD OFFSET VALUE = -800 MICRC-INCHES
6656 036326 000417          BR       15$          ;ISSUE OFFSET
6657
6658 036330 122765 000035 000147 13$:  CMPR    #35,P.READ(R5) ;CHECK IF OFFSET = +1200 MICRO-INCHES
6659 036336 001004          BNE      14$          ;NO, CONTINUE
6660 036340 112765 000060 000006          MOVR    #OFFP12,P.OFST(R5) ;LOAD OFFSET VALUE = +1200 MICRC-INCHES
6661 036346 000407          BR       15$          ;ISSUE OFFSET
6662
6663 036350 122765 000040 000147 14$:  CMPR    #40,P.READ(R5) ;CHECK IF OFFSET = -1200 MICRO-INCHES
6664 036356 001013          BNE      30$          ;NO, NO RETURN
6665 036360 112765 000260 000006          MOVR    #OFFM12,P.OFST(R5) ;LOAD OFFSET VALUE = -1200 MICRO-INCHES
6666 036366 112765 000115 000001 15$:  MOVR    #OFFSET,P.COMD(R5) ;LOAD OFFSET COMMAND
6667 036374 004037 050706          JSR      R0,Q.PUSH      ;ENQUEUE PARAMETER BLOCK IN COMMAND
6668 036400 001324          CTRITQ          ; INITIATE QUEUE
6669 036402 011000          MOV      (R0),R0      ;LOAD OFFSET ISSUED RETURN
6670 036404 000200          RTS      R0          ;RETURN
6671
6672 036406 005720          30$:  TST      (R0)+      ;ADJUST RETURN PCF RC OFFSET
6673 036410 000200          RTS      R0          ;RETURN
  
```

6674				.SBTTL	CALCULATE RETRY WORD COUNT
6675					
6676	036412	013746	001636	CALWC:	MOV RTYCYL,-(SP) ;STORE PETRY CYLINDER
6677	036416	166516	000002		SUB P.CYLN(R5),(SP) ;SUBTRACT INITIAL CYLINDER
6678	036422	011646			MOV (SP),-(SP) ;STORE CYLINDER DIFFERENCE
6679	036424	006316			ASL (SP) ;MULTIPLY DIFFERENCE BY 2
6680	036426	062616			ADD (SP)+,(SP) ;MULTIPLY DIFFERENCE BY 3
6681	036430	005046			CLR -(SP) ;MAKE ROOM ON STACK
6682	036432	113716	001635		MOVB RTYTRK,(SP) ;GET PETRY TRACK
6683	036436	062616			ADD (SP)+,(SP) ;ADD PETRY TRACK
6684	036440	005046			CLR -(SP) ;MAKE ROOM ON THE STACK
6685	036442	116516	000005		MOVB P.TRCK(R5),(SP) ;GET INITIAL TRACK
6686	036446	162616			SUB (SP)+,(SP) ;SUBTRACT INITIAL TRACK
6687	036450	012746	000026		MOV #22,-(SP) ;MULTIPLY TRACK BY 22
6688	036454	004737	054064		JSR PC,\$MULT
6689	036460	012616			MOV (SP)+,(SP) ;THROW AWAY MOST SIGNIFICANT BITS
6690	036462	005046			CLR -(SP) ;MAKE ROOM ON THE STACK
6691	036464	113716	001634		MOVB RTYSEC,(SP) ;GET PETRY SECTOR
6692	036470	062616			ADD (SP)+,(SP) ;ADD PRESENT SECTOR
6693	036472	005046			CLR -(SP) ;MAKE ROOM ON STACK
6694	036474	116516	000004		MOVB P.SECT(R5),(SP) ;GET INITIAL SECTOR
6695	036500	162616			SUB (SP)+,(SP) ;CALCULATE NUMBER OF SECTORS UNTIL ERROR
6696	036502	111666	000001		MOVB (SP),1(SP) ;MULTIPLY BY 256 WORDS/SECTOR
6697	036506	105016			CLRB (SP)
6698	036510	161637	001632		SUB (SP),RTYWC ;SUBTRACT FROM INITIAL WORD COUNT
6699	036514	006316			ASL (SP) ;DETERMINE NUMBER OF BYTES
6700	036516	012637	001630		ADD (SP)+,RTYBA ;CALCULATE STARTING BUFFER ADDRESS
6701	036522	000207			RTS PC ;RETURN

```

6702          .SBTTL RECOVERABLE CONTROLLER ERROR TERMINATION
6703
6704 036524 012037 036546      CONFER: MOV      (R0)+,25      ;STORE ERROR MESSAGE ADDRESS
6705 036530 004737 027242      JSR      PC,PRT000      ;PRINT MESSAGE
6706 036534 032777 026000 142376 BIT      BSM13,BSMP      ;CHECK IF INHIBIT PRINT OUT
6707 036542 001037              BNE      10$              ;YES, CHECK IF HALT ON ERROR
6708 036544 104401              TYPE     ,ERR067          ;TYPE ERROR MESSAGE
6709 036546 000000      2$:   .WOPD      0              ;
6710 036550 104401 001165      TYPE     ,SCRLF          ;TYPE <CR><LF>
6711 036554 004737 025550      JSR      PC,PRTSTT        ;PRINT CONTROLLER REGISTERS
6712 036560 004737 027476      JSR      PC,PRT002        ;PRINT CURRENT EXECUTED COMMAND
6713 036564 022737 062121 036546 CMP      BERR066,25      ;CHECK IF ECC LOGIC ERROR
6714 036572 001016              BNE      10$              ;NO, CHECK IF HALT ON ERROR
6715 036574 104401 062141      TYPE     ,ERR067          ;TYPE "RECEPI WRECP"
6716 036600 016546 000062      MOV      P.EPAT(R5),-(SP) ;STORE ECC PATTERN FOR PRINT OUT
6717 036604 004737 052144      JSR      PC,BINOC1        ;CONVERT TO OCTAL
6718 036610 104401 057551      TYPE     ,BLANKS          ;TYPE 2 SPACES
6719 036614 016546 000060      MOV      P.EPOS(R5),-(SP) ;STORE ECC POSITION FOR PRINT OUT
6720 036620 004737 052144      JSR      PC,BINOC1        ;CONVERT TO OCTAL
6721 036624 104401 001165      TYPE     ,SCRLF          ;TYPE <CR><LF>
6722 036630 004737 030472      10$:  JSR      PC,HLT000      ;CHECK IF HALT ON ERROR
6723 036634 005237 001624      INC      CNTCNT          ;INCREMENT CONTROLLER ERROR COUNT
6724 036640 022737 000012 001624 CMP      BT.CNT,CNTCNT ;CHECK IF THRESHOLD EXCEEDED
6725 036646 103006              BHS      15$              ;NO, CONTINUE
6726 036650 104401 057554      TYPE     ,ERR000          ;TYPE "FATAL ERROR"
6727 036654 104401 060117      TYPE     ,ERR008          ;TYPE "CONTROLLER ERROR THRESHOLD EXCEEDED"
6728 036660 000000      HALT
6729 036662 000776      BR      -2
6730
6731 036664 105765 000150      15$:  TSTB      P.DSTT(R5)      ;CHECK IF FIRST ERROR
6732 036670 100005              BPL      20$              ;NO, REQUEST COMMAND
6733 036672 042765 100200 000150 BIT      BBIT15IRIT7,P.DSTT(R5) ;CLEAR ERROR ENQUEUED AND FIRST ERROR
6734 036700 005037 001562      CLW      ERRPRO          ;CLEAR ERROR BEING PROCESSED
6735 036704 004037 050706      20$:  JSR      R0,Q.PUSH      ;REQUEST COMMAND IN COMMAND
6736 036710 001324              CINITQ
6737 036712 000200      RTS      R0                      ;RETURN

```

```

6738 .SBTTL TEST FOR DROP DRIVE ERRORS
6739
6740 036714 032765 000030 000036 TSTD RP: BIT #ACLOISPDLSS,P.DS(R5) ;CHECK IF AC LOW OR SPEED LOSS
6741 036722 001016 BNE 10$ ;YES REPORT ERROR
6742 ; CHECK IF ANY OF THE FOLLOWING ERRORS ARE SET
6743 ; UNSAFE
6744 ; ILLEGAL DISK ADDRESS
6745 ; WRITE LOCK ERROR
6746 ; DRIVE TYPE ERROR
6747 ; FORMAT ERROR
6748 ; NON-EXECUTABLE DRIVE FUNCTION
6749 036724 032765 046064 000034 BIT #UNSIIDAEIMLEIDTYFIPMTEIMXP,P.ER(R5)
6750 BNE 12$ ;YES, REPORT ERROR
6751 036732 001026 ; CHECK IF ANY OF THE FOLLOWING FLAGS ARE SET
6752 ; DRIVE ERROR NOT REPORTED
6753 ; DRIVE READY NOT SET AFTER START SPINDLE
6754 ; VOLUME VALID NOT SET AFTER PACK ACKNOWLEDGE
6755 036734 032765 006010 000212 BIT #BIT3IBIT10IBIT11,P.ERR(R5)
6756 036742 001072 BNE 18$ ;YES, GC REPORT ERROR
6757 ; CHECK IF ANY OF THE FOLLOWING FLAGS ARE SET
6758 ; DRIVE HARD ERROR
6759 ; DRIVE STATUS CHANGE CTD NOT CLEAR
6760 ; UNEXPECTED ATTN FROM DRIVE COMMAND
6761 ; ATTENTION SET BUT DSC AND FAULT RESET
6762 036744 032765 004070 000014 BIT #DRVHRDIDRVDSICUEXATTIMODSC,F.FIRST(R5)
6763 BNE 32$ ;YES REPORT ERROR
6764 036752 001116 TST (R0)+ ;GET NORMAL RETURN ADDRESS
6765 036754 005720 RTS R0 ;RETURN
6766 036756 000200
6767 036760 004737 027242 10$: JSR PC,PR1000 ;PRINT ERROR HEADER
6768 036764 032765 000010 000036 BIT #ACLO,P.DS(R5) ;CHECK IF AC LOW
6769 036772 001403 BEQ 11$ ;NO, CONTINUE
6770 036774 104401 061074 TYPE ,ERR033 ;TYPE "AC LOW"
6771 037000 000477 BR 30$ ;GO GET ERROR QUALIFICATION
6772 037002 104401 061103 11$: TYPE ,ERR034 ;TYPE "SPEED LOSS"
6773 037006 000467 BR 30$ ;GET ERROR QUALIFICATION
6774 037010 004737 027242 12$: JSR PC,PR1000 ;PRINT ERROR HEADER
6775 037014 032765 040000 000034 BIT #UNS,P.ER(R5) ;CHECK IF DRIVE UNSAFE
6776 037022 001403 BEQ 13$ ;NO, CONTINUE
6777 037024 104401 061613 TYPE ,ERR036 ;TYPE "DRIVE UNSAFE"
6778 037030 000456 BR 30$ ;GET ERROR QUALIFICATION
6779 037032 032765 000040 000034 13$: BIT #DTYPE,P.ER(R5) ;CHECK IF DRIVE TYPE ERROR
6780 037040 001403 BEQ 14$ ;NO, CONTINUE
6781 037042 104401 061116 TYPE ,ERR035 ;TYPE "DRIVE TYPE ERROR"
6782 037046 000447 BR 30$ ;GET ERROR QUALIFIER BITS
6783 037050 032765 000020 000034 14$: BIT #FMTE,P.ER(R5) ;CHECK IF FORMAT ERROR
6784 037056 001403 BEQ 15$ ;NO, CONTINUE
6785 037060 104401 061133 TYPE ,ERR036 ;TYPE FORMAT ERROR

```

BIF

6794	037064	000440				BP	30\$	;GET ERROR QUALIFIER BITS
6795								
6796	037066	032765	000004	000034	15\$:	BIT	#NXP,P.ER(R5)	;CHECK IF NON-EXECUTABLE DRIVE FUNCTION
6797	037074	001403				BFQ	16\$	;NO, CONTINUE
6798	037076	104401	061146			TYPE	,ERP037	;TYPE ILLEGAL DRIVE FUNCTION
6799	037102	000431				BR	30\$	;GET ERROR QUALIFIER BITS
6800								
6801	037104	032765	002000	000034	16\$:	BIT	#IDAE,P.ER(R5)	;CHECK IF ILLEGAL DISK ADDRESS
6802	037112	001403				BEQ	17\$	;NO, CONTINUE
6803	037114	104401	061043			TYPE	,FNR031	;TYPE "ILLEGAL DISK ADDRESS"
6804	037120	000427				BP	30\$	;GET ERROR QUALIFIER BITS
6805								
6806	037122	104401	061057		17\$:	TYPE	,ERP032	;TYPE "WRITE LOCK ERROR"
6807	037126	000417				BR	30\$	;GET ERROR QUALIFIER BITS
6808								
6809	037130	004737	027242		18\$:	JSR	PC,PR1000	;PRINT ERROR HEADER
6810	037134	032765	002000	000212		BIT	#BIT10,P.ER(R5)	;CHECK IF DRIVE NOT READY AFTER START SPINDLE
6811	037142	001403				BFQ	19\$	;NO, CONTINUE
6812	037144	104401	061505			TYPE	,FR053+1	;PRINT "DRIVE NOT READY AFTER START SPINDLE"
6813	037150	000406				BP	30\$	;GET ERROR QUALIFIER BITS
6814								
6815	037152	032765	004000	000212	19\$:	BIT	#BIT11,P.ER(R5)	;CHECK IF VOLUME VALID DID NOT SET
6816								; AFTER PACK ACKNOWLEDGE
6817	037160	001404				BFQ	31\$	;NO, CONTINUE
6818	037162	104401	061546			TYPE	,FR054+1	;PRINT "VOLUME VALID NOT SET AFTER
6819								; PACK ACKNOWLEDGE INSTRUCTION"
6820	037166	104401	001165		30\$:	TYPE	,SCRLF	;TYPE <CR><LF>
6821	037172	032765	000010	000212	31\$:	BIT	#BIT3,P.ER(R5)	;CHECK IF ERROR NOT INDICATED BY FAULT
6822	037200	001405				BFQ	42\$	;NO, CHECK OTHER QUALIFIER BITS
6823	037202	104401	061626			TYPE	,ERR057	;TYPE "ERROR NOT INDICATED BY FAULT"
6824	037206	000402				BR	42\$	;CHECK OTHER QUALIFIER BITS
6825								
6826	037210	004737	027242		32\$:	JSR	PC,PR1000	;PRINT ERROR HEADER
6827	037214	032765	000020	000014	42\$:	BIT	#DRVHRD,P.PRST(R5)	;CHECK IF HARD DRIVE ERROR
6828	037222	001402				BFQ	43\$	;NO, CHECK OTHER QUALIFIER BITS
6829	037224	104401	060744			TYPE	,ERR027	;TYPE "DRIVE HARD ERROR"
6830	037230	032765	004000	000014	43\$:	BIT	#MODSC,P.PRST(R5)	;CHECK IF ATTENTION BOT
6831								; NO DRIVE STATUS CHANGE OR FAULT
6832	037236	001402				BEQ	44\$	;NO, CHECK OTHER QUALIFIER BITS
6833	037240	104401	061667			TYPE	,ERR058	;TYPE "ATTN BOT NO CSC CR FAULT"
6834	037244	032765	000040	000014	44\$:	BIT	#DRVDSC,P.PRST(R5)	;CHECK IF DRIVE STATUS CHANGE DID
6835								; NOT CLEAR
6836	037252	001402				BFQ	45\$	;NO, CHECK OTHER ERROR QUALIFIER BITS
6837	037254	104401	060763			TYPE	,ERR028	;TYPE "DRIVE STATUS CHANGE DID NOT CLEAR"
6838	037260	032765	000010	000014	45\$:	BIT	#UEXATT,P.PRST(R5)	;CHECK IF UNEXPECTED ATTENTION FROM
6839								; DRIVE COMMAND
6840	037266	001402				BEQ	46\$	;NO, CHECK OTHER QUALIFIER BITS
6841	037270	104401	061023			TYPE	,ERP030	;TYPE "UNEXPECTED ATTN FROM DRIVE COMMAND"
6842	037274	005265	000206		46\$:	INC	P.ER(R5)	;INCREMENT OTHER ERROR COUNT
6843	037300	032777	020000	141632		BIT	#SW13,@SWF	;CHECK IF INHIBIT PRINT OUT
6844	037306	001015				BNE	60\$	;YES, CHECK IF HALT ON ERROR
6845	037310	005765	000150			TST	P.DSTT(R5)	;CHECK IF ERROR DURING ERROR RECOVERY
6846	037314	100402				BMI	51\$	;NO, PRINT STATUS
6847	037316	104401	062712			TYPE	,FRP202	;TYPE "ERROR DURING ERROR RECOVERY"
6848	037322	004737	025550		51\$:	JSR	PC,PR1STT	;PRINT CONTROLLER REGISTERS
6849	037326	004737	025374			JSR	PC,PRDSTT	;PRINT DRIVE STATUS

6850	037332	004737	027476		JSR	PC,PRI002	;PRINT CURRENT COMMAND
6851	037336	004737	027320		JSR	PC,PRI001	;PRINT PREVIOUS COMMAND
6852	037342	004737	030472	60\$:	JSR	PC,HLT000	;CHECK IF HALT ON ERROR
6853	037346	032777	000100	141564	BIT	#SW6,#SMR	;CHECK PCR SWITCH 6
6854	037354	001431			BEQ	63\$	;MC DROP DRIVE
6855	037356	032765	000030	000036	BIT	#ACLOISPULSS,P.DS(R5)	;CHECK IF UNSAFE, ACLOW AND SPEED LOSS
6856	037364	001004			BNE	61\$	
6857	037366	032765	040000	000034	BIT	#UNS,P.FR(R5)	
6858	037374	001421			BFG	63\$	
6859	037376	105765	000120	61\$:	TSTR	P.IERC(R5)	;CHECK IF INHIBIT ERROR RECOVERY
6860	037402	001012			BNE	62\$	
6861	037404	105265	000146		INCP	P.RECT(R5)	;INCREMENT ERROR REPLY COUNT
6862	037410	122765	000010	000146	CMPB	#B.,P.RECT(R5)	;CHECK IF REPLY THRESHOLD EXCEEDED
6863	037416	103404			BLO	62\$	;YES, INDICATE UNSUCCESSFUL RECOVERY
6864	037420	004037	050706		JSR	RO,Q.PUSH	;ENQUEUE PARAMETER BLOCK IN COMMAND
6865	037424	001324			CINITQ		; INITIATION QUEUE
6866	037426	000402			BR	57\$	;RETURN
6867							
6868	037430	004737	030724	62\$:	JSR	PC,UNRECY	;INDICATE UNSUCCESSFUL RECOVERY
6869	037434	011000		57\$:	MOV	(R0),R0	
6870	037436	000200			RTS	R0	
6871							
6872	037440	004737	022316	63\$:	JSR	PC,PUPRFL	;RELEASE PUPRFL
6873	037444	004737	012716		JSR	PC,DROP	;DROP DRIVE
6874	037450	004737	022016		JSR	PC,GETBUF	;GET NEW BUFFER
6875	037454	005037	001562		CLR	ERRPRO	;CLEAR ERROR BEING PROCESSED
6876	037460	011000			MOV	(R0),R0	;LOAD ERROR RETURN
6877	037462	000200			RTS	R0	;RETURN

```

6878 .SBTTL ERROR RECOVERY SEQUENCE NORMAL RETURN
6879
6880 037464 032765 004000 000150 ERCVY1: BIT #BIT11,P.DSTT(R5) ;CHECK IF RPCALIBRATE ISSUED
6881 037472 001415 BFC EC3$ ;NO, CONTINUE
6882 037474 042765 004000 000150 BIT #BIT11,P.DSTT(R5) ;CLEAR RECALIBRATE ISSUED
6883 037502 052765 010000 000150 15$: BIS #BIT12,P.DSTT(R5) ;SET SEEK ISSUED
6884 037510 112765 000117 000001 MOVN #SEFK,P.CMND(R5) ;ISSUE SEEK INSTRUCTION
6885 037516 004037 050706 JSR R0,Q.PUSH ;PUT PARAMETER BLOCK IN COMMAND
6886 037522 001324 CINITQ ; INITIATION QUEUE
6887 037524 000207 RTS PC ;RETURN
6888
6889 037526 032765 010000 000150 EC3$: BIT #BIT17,P.DSTT(R5) ;CHECK IF SEEK ISSUED
6890 037534 001427 BFC 10$ ;NO, CONTINUE
6891 037536 042765 010000 000150 BIC #BIT12,P.DSTT(R5) ;CLEAR SEEK ISSUED
6892 037544 105765 000211 TSTN P.SFK(R5) ;CHECK IF SEEK TO PREVIOUS POSITION
6893 037550 001403 BFC 5$ ;NO, PROCESS DATA TRANSFER
6894 037552 004737 035500 JSR PC,SUHCY ;INDICATE SUCCESSFUL RECOVERY
6895 037556 000207 RTS PC ;RETURN
6896
6897 037560 122765 000021 000147 5$: CMPR #21,P.RFND(R5) ;CHECK IF OFFSET
6898 037566 101053 BHI 20$ ;NO, REISSUE COMMAND
6899 037570 052765 020000 000150 BIS #BIT13,P.DSTT(R5) ;SET OFFSET ISSUED
6900 037576 112765 000115 000001 MOVN #OFFSET,P.CMND(R5) ;ISSUE OFFSET
6901 037604 004037 050706 JSR R0,Q.PUSH ;PUT PARAMETER BLOCK IN COMMAND
6902 037610 001324 CINITQ ; INITIATION QUEUE
6903 037612 000207 RTS PC ;RETURN
6904
6905 037614 032765 020000 000150 10$: BIT #BIT13,P.DSTT(R5) ;CHECK IF CC PREVIOUS OFFSET
6906 037622 001453 BEQ EC4$ ;NO, CONTINUE
6907 037624 042765 020000 000150 BIC #BIT13,P.DSTT(R5) ;CLEAR OFFSET ISSUED
6908 037632 122765 000021 000147 CMPR #21,P.RFRD(P5) ;CHECK IF OFFSET +400 MICRO-INCHES
6909 037640 001424 BFC 15$ ;YES, GO ISSUE READ
6910 037642 122765 000024 000147 CMPP #24,P.WEWD(R5) ;CHECK IF OFFSET -400 MICRO-INCHES
6911 037650 001420 BEQ 15$ ;YES, GO ISSUE READ
6912 037652 122765 000027 000147 CMPP #27,P.RFRD(R5) ;CHECK IF OFFSET +800 MICRO-INCHES
6913 037660 001414 BEQ 15$ ;YES, GO ISSUE READ
6914 037662 122765 000032 000147 CMPR #32,P.RFRD(R5) ;CHECK IF OFFSET -800 MICRO-INCHES
6915 037670 001410 BFC 15$ ;YES, GO ISSUE READ
6916 037672 122765 000035 000147 CMPR #35,P.RFRD(R5) ;CHECK IF OFFSET +1200 MICRO-INCHES
6917 037700 001404 BFC 15$ ;YES, GO ISSUE READ
6918 037702 122765 000040 000147 CMPP #40,P.RFRD(P5) ;CHECK IF OFFSET -1200 MICRO-INCHES
6919 037710 001002 BNE 20$ ;NO, DO NOT INCREMENT REREAD COUNT
6920 037712 105265 000147 15$: INCR P.RFRD(R5) ;INCREMENT REREAD COUNT
6921 037716 116565 000127 000001 20$: MOVN P.RCMD(P5),P.CMND(R5) ;LOAD COMMAND
6922 037724 122765 000131 000001 CMPP #WRTCHK,P.CMND(R5) ;CHECK IF WRITE CHECK
6923 037732 001003 BNE 25$ ;NO, ISSUE COMMAND
6924 037734 052765 000700 000014 BIS #W.MCK,P.PRST(R5) ;YES, ISSUE WRITE BEFORE WRITE CHECK
6925 037742 004037 050706 25$: JSR R0,Q.PUSH ;PAQUE PARAMETER BLOCK IN COMMAND
6926 037746 001324 CINITQ ; INITIATION QUEUE
6927 037750 000207 RTS PC ;RETURN
6928
6929 037752 032765 000100 000150 EC4$: BIT #BIT6,P.DSTT(R5) ;CHECK IF WAITING FOR READ STATUS
6930 037760 001436 BFC EC5$ ;NO, CONTINUE
6931 037762 042765 000100 000150 BIC #BIT6,P.DSTT(P5) ;CLEAR WAITING FOR READ STATUS
6932 037770 113765 001662 000001 MOVN POSCMD,P.CMND(R5) ;RESTORE COMMAND IF ERROR

```

```
6934 037776 032765 000200 000036      BIT      #DRDY,P.DS(R5) ;CHECK IF DRIVE READY
6935 040004 001004      BNE      10$      ;YES, REISSUE COMMAND
6936 040006 004037 027716      JSR      RO,PRI100 ;PRINT DRIVE TIME OUT DURING POSITIONING
6937 040012 061426      EPR052      ; AND DRCP CRIVE
6938 040014 000207      RTS      PC      ;RETURN
6939                                     SS:
6940 040016 004037 030076      10$:      JSR      RO,PRI200 ;PRINT DRIVE TIME OUT DURING
6941 040022 061426      EPR052      ; POSITIONING
6942 040024 004037 030522      JSR      RO,NERSTT ;TAKE OTHER ERROR STATICS
6943 040030 040014      5$          ;NO RECOVERY RETURN
6944 040032 052765 004000 000150      BIT      #BIT11,P.DSTT(R5) ;SET RECALIBRATE ISSUED
6945 040040 112765 000113 000001      MOVP      #RECAL,P.CMND(R5) ;ISSUE RECALIBRATE
6946 040046 004037 050706      JSR      RO,Q.PUSH ;ENQUEUE PARAMETER BLOCK IN COMMAND
6947 040052 001324      CINITQ      ; INITIATION QUEUE
6948 040054 000207      RTS      PC      ;RETURN
6949
6950 040056 032765 000020 000150 EC5$:      BIT      #BIT4,P.DSTT(R5) ;CHECK IF HEADERS ERROR
6951 040064 001002      BNE      1$          ;YES, GO PROCESS FWRUP
6952 040066 000137 040602      JMP      EC6$      ;NO, CONTINUE
6953
6954 040072 042765 000020 000150 1$:      BIC      #BIT4,P.DSTT(R5) ;CLEAR READ ALL HEADERS ISSUED
6955 040100 010346      MOV      R3,-(SP) ;STORE R3 ON THE STACK
6956 040102 016546 000056      MOV      P.B11(R5),-(SP) ;STORE SECTOR HEAD
6957 040106 042716 177017      BIC      #C<M.SECT>,(SP) ;KEEP ONLY SECTOR COUNT
6958 040112 006216      ASR      (SP) ;SHIFT 4 BITS RIGHT
6959 040114 006216      ASR      (SP)
6960 040116 006216      ASR      (SP)
6961 040120 006216      ASR      (SP) ;GET SECTOR COUNT
6962 040122 005046      CLR      -(SP) ;MAKE ROOM ON THE STACK
6963 040124 116516 000004      MOVP      P.SECT(R5),(SP) ;GET DESIRED SECTOR
6964 040130 162616      SUB      (SP)+,(SP) ;CALCULATE DIFFERENCE
6965 040132 100410      BMI      2$          ;BRANCH IF IN LOWER PART OF BUFFER
6966 040134 012703 065174      MOV      #RUPADD-6,R3 ;LOAD BASE OF HIGHER PART OF BUFFER
6967 040140 006316      ASL      (SP) ;MULTIPLY SECTOR COUNT BY 6
6968 040142 011646      MOV      (SP),-(SP)
6969 040144 006316      ASL      (SP)
6970 040146 062616      ADD      (SP)+,(SP)
6971 040150 162603      SUB      (SP)+,R3 ;CALCULATE HEADER ADDRESS
6972 040152 000407      BP      3$          ;CHECK HEADERS
6973
6974 040154 005416      2$:      MFC      (SP) ;MAKE COUNT POSITIVE
6975 040156 006316      ASL      (SP) ;MULTIPLY DIFFERENCE BY 6
6976 040160 011603      MOVP      (SP),R3
6977 040162 006316      ASL      (SP)
6978 040164 062603      ADD      (SP)+,R3
6979 040166 062703 064770      ADD      #RUPUFF-6,R3 ;CALCULATE HEADER ADDRESS
6980 040172 012337 001642      3$:      MOVP      (R3)+,HEAD1 ;GET FIRST WORD OF HEADER
6981 040176 012337 001644      MOV      (R3)+,HEAD2 ;GET SECOND WORD OF HEADER
6982 040202 011337 001646      MOV      (R3),HEAD3 ;GET THIRD WORD OF HEADER
6983 040206 012603      MOVP      (SP)+,R3 ;RESTORE R3
6984 040210 004737 036200      JSR      PC,RSTAT ;RESTORE STATUS
6985 040214 013746 001642      MOV      HEAD1,-(SP) ;STORE FIRST WORD OF THE HEADER
6986 040220 043716 001644      BIC      HEAD2,(SP) ;HD1 & HD2
6987 040224 013746 001644      MOV      HEAD2,-(SP) ;STORE SECOND WORD OF HEADER
6988 040230 043716 001647      BIC      HEAD1,(SP) ;HD1 & HD2
6989 040234 052616      RTS      (SP)+,(SP) ;GENERATE EXPECTED HEADER WORD
```

6990	040236	023726	001646			CMP	HEAD3,(SP)+	;CHECK IF HEADER VRC IS CORRECT
6991	040242	001501				BEQ	15\$	;NO, CONTINUE
6997	040244	004037	030076			JSR	RO,PRI200	;PRINT HEADER VRC ERROR
6993	040250	061253				EPK043		
6994	040252	105765	000120			TSTB	P.IERC(P5)	;CHECK IF INHIBIT ERROR RECOVERY
6995	040256	001016				BNE	5\$	;YES, INDICATE UNSUCCESSFUL RECOVERY
6996	040260	105265	000147	4\$:		INCR	P.PERC(P5)	;INCREMENT PEREC COUNT
6997	040264	122765	000043	000147		CMPB	#43,P.PERC(P5)	;CHECK IF UNRECOVERABLE
6998	040272	101410				BLOS	5\$	;YES, INDICATE UNRECOVERABLE ERROR
6999	040274	122765	000121	000121		CMPB	#PDATA,P.RCMD(P5)	;CHECK IF READ DATA COMMAND
7000	040302	001407				BEQ	6\$	;YES, START ERROR RECOVERY
7001	040304	122765	000021	000147		CMPB	#21,P.PERC(P5)	;CHECK IF UNRECOVERABLE WRITE
7002	040312	101021				BHI	8\$	;NO, START ERROR RECOVERY
7003	040314	004737	030724	5\$:		JSR	PC,UNRECV	;INDICATE UNRECOVERABLE ERROR
7004	040320	000207				RTS	PC	;RETURN
7005								
7006	040322	122765	000021	000147	6\$:	CMPB	#21,P.PERC(P5)	;CHECK IF BEGINNING OFFSETTING OPERATION
7007	040330	001007				BNE	7\$	;NO, DETERMINE PROPER OFFSET
7008	040332	010446				MOV	R4,-(SP)	;STORE R4 ON THE STACK
7009	040334	116504	000000			MOV	P.DRVN(R5),R4	;GET DRIVE NUMBER FOR INDEX
7010	040340	146437	001510	001507		BICR	INTMSK(R4),O.OVER	;DO NOT DO IMPLIED SEES
7011	040346	012604				MOV	(SP)+,R4	;RESTORE R4
7012	040350	004037	036230	7\$:		JSR	RO,DETOPF	;DETERMINE OFFSET VALUE
7013	040354	040444				12\$		;OFFSET ISSUED RETURN
7014	040356	116565	000121	000001	8\$:	MOV	P.RCMD(R5),P.CMD(R5)	;STORE COMMAND
7015	040364	013765	001636	000002		MOV	RTYCYL,P.CYL(R5)	;GET CYLINDER
7016	040372	013765	001634	000004		MOV	RTYSEC,P.SECT(R5)	;GET TRACK AND SECTOR
7017	040400	013765	001630	000010		MOV	RTYMA,P.BALO(R5)	;GET BUS ADDRESS
7018	040406	013765	001632	000017		MOV	RTYMC,P.MC(R5)	;GET WORD COUNT
7019	040414	005465	000012			MFC	P.MC(R5)	;MAKE IT POSITIVE
7020	040420	122765	000131	000001		CMPB	#WRTCHK,P.CMD(R5)	;CHECK IF WRITE CHECK COMMAND
7021	040426	001003				BNE	10\$	;NO, GO ISSUE COMMAND
7022	040430	052765	000700	000014		BIS	#W.MCK,P.PHST(R5)	;ISSUE WRITE BEFORE WRITE CHECK
7023	040436	004037	050706	10\$:		JSR	RO,Q.PUSH	;ENQUEUE PARAMETER BLOCK ON COMMAND
7024	040442	001324				CINITQ		; INITIATION QUEUE
7025	040444	000207		12\$:		RTS	PC	;RETURN
7026								
7027	040446	023765	001642	000030	15\$:	CMP	HEAD1,P.DCYL(R5)	;CHECK IF CYLINDER CORRECT
7028	040454	001420				BEQ	20\$	;YES, CHECK SECTOR TRACK AND FORMAT
7029	040456	004037	030076			JSR	RO,PRI200	;PRINT MISPOSITIONING
7030	040462	062066				EPK064		
7031	040464	004037	030570			JSR	RO,OPISTT	;GO TAKE STATISTICS
7032	040470	040514				17\$		;UNRECOVERABLE ERROR RETURN
7033	040472	052765	004000	000150		BIS	#BIT11,P.DSTT(R5)	;SET RECALIBRATE ISSUED
7034	040500	112765	000113	000001		MOV	#RECAL,P.CMD(R5)	;ISSUE RECALIBRATE
7035	040506	004037	050706			JSR	RO,Q.PUSH	;ENQUEUE PARAMETER BLOCK IN
7036	040512	001324				CINITQ		; COMMAND INITIATION QUEUE
7037	040514	000207		17\$:		RTS	PC	;RETURN
7038								
7039	040516	005046		20\$:		CLR	-(SP)	;MAKE ROOM ON STACK
7040	040520	113766	001635	000001		MOV	RTYTRK,1(SP)	;GET TRACK
7041	040526	006216				ASR	(SP)	;SHIFT RIGHT 3 BITS
7042	040530	006216				ASR	(SP)	
7043	040532	006216				ASR	(SP)	
7044	040534	153716	001634			BISR	RTYSEC,(SP)	;GENERATE EXPECTED SECOND WORD
7045	040540	052716	140000			BIS	#140000,(SP)	;PUT IN GC TRACK BITS

7046	040544	022637	001644			CMP	(SP)+,HEAD2	;CHECK IF SECOND WORD IS CORRECT
7047	040550	001407				BEQ	25\$	;CHECK FOR BAD SECTOR
7048	040552	004037	030076			JSR	RO,PRI200	;INDICATE OPERATION INCOMPLETE
7049	040556	061234				EPR042		
7050	040560	004037	030570			JSR	RO,OPISTT	;GC TAKE STATISTICS
7051	040564	040514				17\$		;UNRECOVERABLE RETURN
7052	040566	000673				BR	8\$	;GO ISSUE LAST COMMAND
7053								
7054	040570	004037	030076		25\$:	JSR	RO,PRI200	;TYPE HEADER FRCH
7055	040574	063223				EPR209		
7056	040576	000137	040260			JMP	4\$	;GO TRY TO RETRY COMMAND
7057								
7058	040602	122765	000021	000147	EC6\$:	CMPR	#21,P.RFHD(R5)	;CHECK IF REISSUE DATA TRANSFER
7059	040610	101036				BRI	10\$	;YES, INDICATE SUCCESSFUL RECOVERY
7060	040612	001424				BEQ	5\$	;NO, OFFSET = +400 MICRO-INCHES
7061								;ISSUE DATA TRANSFER
7062	040614	122765	000024	000147		CMPB	#24,P.RERD(R5)	;CHECK IF OFFSET -400 MICRO-INCHES
7063	040622	001420				BEQ	5\$	;YES, ISSUE DATA TRANSFER
7064	040624	122765	000027	000147		CMPB	#77,P.RERD(R5)	;CHECK IF OFFSET +800 MICRO-INCHES
7065	040632	001414				BEQ	5\$	;YES, ISSUE DATA TRANSFER
7066	040634	122765	000032	000147		CMPB	#32,P.REHD(R5)	;CHECK IF OFFSET -800 MICRO-INCHES
7067	040642	001410				BEQ	5\$	;YES, ISSUE DATA TRANSFER
7068	040644	122765	000035	000147		CMPB	#35,P.RFHD(R5)	;CHECK IF OFFSET +1200 MICRO-INCHES
7069	040652	001404				BEQ	5\$	;YES, ISSUE DATA TRANSFER
7070	040654	122765	000040	000147		CMPB	#40,P.RFHD(R5)	;CHECK IF OFFSET -1200 MICRO-INCHES
7071	040662	001011				BNE	10\$	;NO, INDICATE SUCCESSFUL RECOVERY
7072	040664	105265	000147		5\$:	INCB	P.RERD(R5)	;INCREMENT REHEAD COUNT
7073	040670	116565	000123	000001		MOVB	P.RCMD(R5),P.CMND(R5)	;REISSUE COMMAND
7074	040676	004037	050706			JSR	RO,G.PUSH	;ENQUEUE PARAMETER BLOCK IN COMMAND
7075	040702	001324				CINITQ		;INITIATION CODE
7076	040704	000207				RTS	PC	;RETURN
7077								
7078	040706	032765	040000	000150	10\$:	BIT	#BIT14,P.DSTT(M)	;CHECK IF SERVICING BAD SECTOR
7079	040714	001403				BEQ	11\$	;NO, CONTINUE
7080	040716	005765	000146			TST	P.RFCT(R5)	;CHECK IF IN RETRY SEQUENCE
7081	040722	001477				BEQ	EC7\$	;NO, GO TO BAD SECTOR SERVICE ROUTINE
7082	040724	105765	000210		11\$:	TSTR	P.ASSN(R5)	;CHECK IF DRIVE IS BEING ASSIGNED
7083	040730	001403				BEQ	12\$	;NO, CHECK IF SEEK TO PREVIOUS CYLINDER
7084	040737	004737	035500			JSR	PC,SURECY	;INDICATE SUCCESSFUL RECOVERY
7085	040736	000207				RTS	PC	;RETURN
7086								
7087	040740	122765	000117	000001	12\$:	CMPB	#SEEK,P.CMND(R5)	;CHECK IF SEEK TO PREVIOUS CYLINDER
7088	040746	001404				BEQ	14\$	;YES, INDICATE SUCCESSFUL RECOVERY
7089	040750	122765	000140	000001		CMPB	#RELEASE,P.CMND(R5)	;CHECK IF RELEASE
7090	040756	001003				BNE	16\$	;NO, PROCESS DATA TRANSFER
7091	040760	004737	035500		14\$:	JSR	PC,SURECY	;INDICATE SUCCESSFUL RECOVERY
7092	040764	000207				RTS	PC	;RETURN
7093								
7094	040766	004037	024506		16\$:	JSR	RO,CHKADD	;CHECK SECTOR, TRACK, CYLINDER, ECS ADDRESS
7095								;AND WORD COUNT
7096	040772	041030				20\$		;FARER RETURN
7097	040774	005037	177776			CLR	PS	;ALLOW RK06 INTERRUPTS
7098	041000	122765	000121	000001		CMPB	#RDATA,P.CMND(R5)	;CHECK IF READ DATA
7099	041006	001021				BNE	30\$	;NO, CHECK IF WRITE CHECK
7100	041010	004037	022760			JSR	RO,PUPFP1	;FIND FIRST DATA FRMP
7101	041014	041032				25\$		;REPCR RETURN

7102	041016	013737	001450	177776	10\$:	MOV	RFPRI,PS	;LOCK OUT RK06 INTERRUPTS
7103	041024	004737	035500			JSR	PC,SURECY	;INDICATE SUCCESSFUL RECOVERY
7104	041030	000207			20\$:	RTS	PC	;RETURN
7105								
7106	041032	013737	001450	177776	25\$:	MOV	RFPRI,PS	;LOCK OUT RK06 INTERRUPTS
7107	041040	052765	000004	000212		BIS	#BIT2,P.ERR(R5)	;INDICATE DATA COMPARE ERROR
7108	041046	000137	034512			JMP	E10\$	;PROCESS DATA COMPARE ERROR
7109								
7110	041052	122765	000131	000001	30\$:	CMPP	#WRTCHK,P.CMND(R5)	;CHECK IF WRITE CHECK
7111	041060	001356				BNE	10\$	;NO, INDICATE SUCCESSFUL RECOVERY
7112	041062	032765	000200	000014		BIT	#W.WCK,P.PRST(R5)	;CHECK IF WRITE CHECK ISSUED
7113	041070	001752				BEG	10\$	;YES, INDICATE SUCCESSFUL RECOVERY
7114	041072	042765	000200	000014		BIC	#W.WCK,P.PRST(R5)	;CLEAR WRITE BEFORE WRITE CHECK
7115	041100	004037	050706			JSR	RO,Q.PUSH	;ENQUEUE COMMAND IN COMMAND INITIATION QUEUE
7116	041104	001324				CINITO		
7117	041106	000207				RTS	PC	;RETURN
7118								
7119	041110	004037	024506		EC7\$:	JSR	RO,CHKADD	;CHECK IF BUS ADD, WORD COUNT, SECTOR,
7120								; TRACK AND CYLINDER ARE CORRECT
7121	041114	041202				9\$		;ERROR RETURN
7122	041116	042765	040000	000150		BIC	#BIT14,P.DSTT(R5)	;CLEAR SERVICING PAD SECTOR
7123	041124	004737	041344			JSR	PC,BSTRAN	;CALCULATE NUMBER OF WORDS TRANSFERRED
7124	041130	013765	001654	000012		MOV	BSWC,P.WC(R5)	;LOAD BUS ADDRESS FOR NEXT DATA TRANSFER
7125	041136	100022				BPL	10\$	;CHECK IF DATA TRANSFER FINISHED
7126	041140	013765	001650	000004		MOV	BSSECT,P.SECT(R5)	;LOAD SECTOR AND TRACK FOR NEXT COMPARE
7127	041146	013765	001652	000002		MOV	BSCYLN,P.CYLN(R5)	;LOAD CYLINDER FOR NEXT COMPARE
7128	041154	013765	001656	000010		MOV	BSBA,P.BALO(R5)	;LOAD BUS ADDRESS FOR NEXT DATA TRANSFER
7129	041162	052765	000200	000014		BIS	#W.WCK,P.PRST(R5)	;GC ISSUE WRITE
7130	041170	004037	050706		5\$:	JSR	RO,Q.PUSH	;ENQUEUE COMMAND IN COMMAND INITIATION
7131	041174	001324				CINITO		; QUEUE
7132	041176	005037	001562			CLR	ERRPRO	;CLEAR ERROR PROCESSING FLAG
7133	041202	000207			9\$:	RTS	PC	;RETURN
7134								
7135	041204	105765	000210		10\$:	TSTR	P.ASSN(R5)	;CHECK IF PACK IS BEING WRITTEN
7136	041210	001012				BNE	13\$	;YES, CHECK IF SEEK TO NEXT CYLINDER
7137	041212	004737	022316			JSR	PC,BUFFFL	;RELEASE BUFFER
7138	041216	004037	050706			JSR	RO,Q.PUSH	;ENQUEUE PARAMETER BLOCK IN DRIVE
7139	041222	001314				AVATLO		; AVAILABLE QUEUE
7140	041224	004737	022016			JSR	PC,GETBUF	;GET NEXT BUFFER FOR DATA TRANSFER
7141	041230	005037	001562			CLR	ERRPRO	;CLEAR ERROR RETNC PROCESSING
7142	041234	000207				RTS	PC	;RETURN
7143								
7144	041236	005037	001562		13\$:	CLR	ERRPRO	;CLEAR ERROR BEING PROCESSED
7145	041242	013765	001650	000026		MOV	BSSECT,P.DTS(R5)	;LOAD NEXT SECTOR AND TRACK
7146	041250	013765	001652	000030		MOV	BSCYLN,P.DCYL(R5)	;LOAD NEXT CYLINDER
7147	041256	032777	040000	137654		BIT	#SW14,#SWR	;CHECK IF LOCK ON CURRENT OPERATION
7148	041264	001002				BNE	14\$	;YES, ISSUE SEEK TO PREVIOUS CYLINDER
7149	041266	000137	015140			JMP	WV0\$	;NO, CONTINUE TO WRITE PACK
7150								
7151	041272	004737	022316		14\$:	JSR	PC,BUFFREL	;RELEASE BUFFER
7152	041276	032765	002000	000014		BIT	#DRPDV,P.PRST(R5)	;CHECK IF DRIVE DRIVE
7153	041304	001402				BEG	14\$	;NO ISSUE SEEK TO PREVIOUS CYLINDER
7154	041306	000137	015626			JMP	WV1\$	;DROP DRIVE
7155								
7156	041312	112765	177777	000211	15\$:	MOVR	#-1,P.SEEK(R5)	;SET SEEK TO PREVIOUS CYLINDER
7157	041320	016565	000136	000002		MOV	P.LCYL(R5),P.CYLN(R5)	;GET LAST CYLINDER

RK611/RK06 PERFORMANCE EXERCISER MAINDEC DZR6P-B
DZR6PB.P11 11-AUG-76 00:00

MACV11 30(1046) 01-DEC-77 14:15 PAGE 157
ERROR RECOVERY SEQUENCE NORMAL RETURN

SEC 0179

7158 041326 016565 000140 000004
7159 041334 112765 000117 000001
7160 041347 000712

MOV P.LSEC(R5),P.SECT(R5) ;GET LAST TRACK AND SECTOR
MOVR #SEEK,P.CMND(R5) ;LOAD COMMAND
BP 55 ;GO ISSUE SEEK COMMAND

```

7161          .SBTTL  CALCULATE WORDS TRANSFERRED UNTIL BAD SECTOR
7162
7163 041344 016537 000026 001650 BSTRAN: MOV      P.DTS(R5),BSSFCT ;STORE ADDRESS OF BAD SECTOR
7164 041352 016537 000030 001657      MOV      P.DCYL(R5),BSCYLW
7165 041360 105237 001650          INCP      BSSFCT      ;ADVANCE TO NEXT SECTOR ON BACK
7166 041364 122737 000025 001650      CMPB     #21.,BSSECT    ;CHECK FOR SECTOR OVERFLOW
7167 041372 103014          BPLS      SS      ;NO, CONTINUE
7168 041374 105037 001650          CLRB     BSSFCT      ;CLEAR SECTOR
7169 041400 105237 001651          INCB     BSTRCK      ;INCRMPMT TRACK
7170 041404 122737 000002 001651      CMPB     #2,BSTRCK    ;CHECK FOR TRACK OVERFLOW
7171 041412 103004          BPLS      SS      ;NO, CONTINUE
7172 041414 105037 001651          CLRB     BSTRCK      ;CLEAR TRACK
7173 041420 005237 001652          INC      BSCYLW      ;INCRMPMT CYLINDER
7174 041424 013746 001652          SS:  MOV      BSCYLW,-(SP) ;STORE CYLINDER ON STACK
7175 041430 166516 000002          SUB      P.CYLW(R5),(SP) ;SUBTRACT STARTING CYLINDER
7176 041434 011646          MOV      (SP),-(SP) ;SAVE DIFFERENCE
7177 041436 006316          ASL      (SP)      ;MULTIPLY CYLINDER BY 3
7178 041440 062616          ADD      (SP)+,(SP)
7179 041442 005046          CLR      -(SP)      ;MAKE ROOM ON STACK
7180 041444 113716 001651          MOVB     BSTRCK,(SP) ;PLACE TRACK ON STACK
7181 041450 062616          ADD      (SP)+,(SP) ;ADD TRACK
7182 041452 005046          CLR      -(SP)      ;MAKE ROOM ON THE STACK
7183 041454 116516 000005          MOVB     P.TRCK(R5),(SP) ;STORE TRACK
7184 041460 162616          SUB      (SP)+,(SP) ;SUBTRACT INITIAL TRACK
7185 041462 012746 000026          MOV      #22,-(SP) ;STORE 22 ON STACK FOR MULTIPLICATION
7186 041466 004737 054064          JSR      PC,SMULT ;MULTIPLY BY 22
7187 041472 005046          CLR      -(SP)      ;MAKE ROOM ON THE STACK
7188 041474 113716 001650          MOVB     BSSECT,(SP) ;STORE BAD SECTOR
7189 041500 062616          ADD      (SP)+,(SP) ;ADD BAD SECTOR ADDRESS
7190 041502 005046          CLR      -(SP)      ;MAKE ROOM ON THE STACK
7191 041504 116516 000004          MOVB     P.SPCT(R5),(SP) ;STORE ORIGINAL SECTOR ADDRESS
7192 041510 162616          SUB      (SP)+,(SP) ;SUBTRACT ORIGINAL SPCTCF ADDRESS
7193 041512 111666 000001          MOVB     (SP),1(SP) ;KEEP NUMBER OF SECTORS TRANSFERRED
7194 041516 105016          CLRP      (SP)
7195 041520 011637 001660          MOV      (SP),RWORD ;LOAD WORD COUNT
7196 041524 006316          ASL      (SP)      ;MULTIPLY BY 2
7197 041526 066516 000010          ADD      P.BALO(R5),(SP) ;GET STARTING ADDRESS FOR NEW DATA
7198 041532 011637 001656          MOV      (SP),BSBA ;GET NEW BUS ADDRESS
7199 041536 166516 000132          SUB      P.BPAL(R5),(SP) ;GET NUMBER OF BYTES TRANSFERRED
7200 041542 006216          ASR      (SP)      ;CONVERT TO WORDS (DIVIDE BY 2)
7201 041544 066516 000130          ADD      P.RWC(R5),(SP) ;DETERMINE REMAINING WORD COUNT
7202 041550 012637 001654          MOV      (SP)+,BSWC ;STORE CALCULATED WORD COUNT
7203 041554 005726          TST      (SP)+ ;THROW AWAY MOST SIGNIFICANT BITS
7204 041556 000207          RTS      PC ;RETURN

```

```

7205          .SBTTL  BAD SECTOR HANDLING
7206
7207 041560 032765 040000 000150 BDSECT: BIT  #BIT14,P.DSTT(R5) ;CHECK IF PRESENTLY SERVICING BAD SECTOR
7208 041566 001401 BEQ 5$ ;NO, CONTINUE
7209 041570 000000 HALT
7210 041572 132765 000010 000210 5$: BIT# #BIT3,P.ASSN(R5) ;CHECK IF READING PACK SERIAL NUMBER
7211 041600 001157 BNE 40$ ;YES, DETERMINE NEXT SECTOR
7212 041602 004737 041344 JSR PC,RSTRAN ;CALCULATE WORDS TRANSFERRED
7213 041606 122765 000121 000001 CMPB #RDATA,P.CMND(R5) ;CHECK IF READ COMMAND
7214 041614 001076 BNE 25$ ;NO, DO NO CHECK BUS ADDRESS AND WORD COUNT
7215 041616 013746 001656 MOV BSBA,-(SP) ;STORE BUS ADDRESS
7216 041622 162716 001000 SUB #256,*2,(SP) ;SUBTRACT ADDITIONAL SECTOR
7217 041626 026526 000024 CMP P.BAR(R5),(SP)+ ;CHECK IF BUS ADDRESS CORRECT
7218 041632 001404 BEQ 10$ ;YES, CONTINUE
7219 041634 004037 036524 JSR R0,CONERR ;PRINT "BUS ADDRESS INVALID"
7220 041640 062705 EPR071
7221 041642 000207 RTS PC ;RETURN
7222
7223 041644 016546 000022 10$: MOV P.WCR(R5),-(SP) ;STORE WORD COUNT FOR COMPARISON
7224 041650 062716 000400 ADD #256,-(SP) ;SUBTRACT OFF BAD SECTOR
7225 041654 023726 001654 CMP BSMC,(SP)+ ;CHECK IF WORD COUNT CORRECT
7226 041660 001404 BEQ 15$ ;YES, CONTINUE
7227 041662 004037 036524 JSR R0,CONERR ;PRINT "WORD COUNT INVALID"
7228 041666 062162 ERR070
7229 041670 000207 RTS PC ;RETURN
7230
7231 041672 042765 100200 000150 15$: BIT #BIT15|BIT7,P.DSTT(R5) ;CLEAR FIRST ERROR AND ERROR ENQUEUED
7232 041700 022737 000400 001660 CMP #256-,BSWORD ;CHECK IF NO WORDS TRANSFERRED
7233 041706 001413 BEQ 20$ ;YES, DO NOT CHECK DATA
7234 041710 004037 022760 JSR R0,PUFER1 ;CHECK DATA HEAD
7235 041714 042250 SOS ;ERROR RETURN
7236 041716 063765 001660 000012 ADD BSWORD,P.WC(R5) ;CALCULATE NEW WORD COUNT
7237 041724 100410 BNE 22$ ;CHECK IF DATA TRANSFER NOT COMPLETE
7238 041726 000464 BP 30$ ;DATA TRANSFER COMPLETE
7239
7240 041730 052765 000200 000014 19$: BIS #W.WCF,P.PHST(R5) ;ISSUE WRITE BEFORE WRITE CHECK
7241 041736 013765 001654 000012 20$: MOV BSMC,P.WC(R5) ;GET NUMBER OF WORDS TO BE TRANSFERRED
7242 041744 100055 BPL 30$ ;CHECK IF DATA TRANSFER COMPLETE
7243 041746 013765 001652 000002 22$: MOV BSCYLN,P.CYLN(R5) ;GET NEW CYLINDER
7244 041754 013765 001650 000004 MOV BSSECT,P.SECT(R5) ;GET NEW SECTOR
7245 041762 013765 001656 000010 MOV BSBA,P.PALC(R5) ;GET NEW BUS ADDRESS
7246 041770 004037 050706 JSR R0,Q.PUSH ;PUT PARAMETER IN COMMAND
7247 041774 001324 CINITQ ; INITIATION QUEUE
7248 041776 005765 000146 TST P.RECT(R5) ;CHECK IF ERROR RECOVERY IN PROGRESS
7249 042002 001002 BNE 23$ ;YES, DO NOT RESET ERROR PROCESSING IN PROGRESS
7250 042004 005037 001562 CLR ERRPRD ;RESET ERROR PROCESSING IN PROGRESS
7251 042010 000207 23$: RTS PC ;RETURN
7252
7253 042012 042765 100200 000150 25$: BIT #BIT15|BIT7,P.DSTT(R5) ;CLEAR FIRST ERROR AND ERROR ENQUEUED
7254 042020 122765 000131 000001 CMPB #WCHK,P.CMND(R5) ;CHECK IF WRITE CHECK COMMAND
7255 042026 001343 BNE 20$ ;NO, CONTINUE
7256 042030 022737 000400 001660 CMP #256-,BSWORD ;CHECK IF NO WORDS TRANSFERRED
7257 042036 001734 BEQ 19$ ;YES, ISSUE NEXT DATA TRANSFER
7258 042040 012746 000400 MOV #256,-(SP) ;STORE EXCESS WORD COUNT ON STACK
7259 042044 163716 001660 SUB BSWORD,(SP) ;CALCULATE NEW WORD COUNT
7260 042050 012665 000012 MOV (SP)+,P.WC(R5) ;STORE NEW WORD COUNT

```

7261	042054	052765	040000	000150		BTS	#BIT14,P.DSTT(R5)	;INDICATE THAT PAD
7262								; SECTOR IS BEING PROCESSED
7263	042062	042765	000200	000014		BTC	#W.MCF,P.PEST(R5)	;DC WRITE CHECK PART OF COMMAND
7264	042070	004037	050706			JSR	RO,Q.PUSH	;PUT PARAMETER BLOCK IN COMMAND
7265	042074	001324				CINITQ		; INITIATION QUEUE
7266	042076	000707				RTS	PC	;RETURN
7267								
7268	042100	004737	022316		30\$:	JSR	PC,BUPREL	;RELEASE BUFFER
7269	042104	005765	000146			TST	P.RECT(R5)	;DETERMINE IF CURRENTLY UNDERGOING ERROR
7270								; RECOVERY
7271	042110	001403				BFC	35\$	;YES, PUT IN AVAILABLE QUEUE
7272	042112	004737	035500			JSR	PC,SURECY	;INDICATE SUCCESSFUL RECOVERY
7273	042116	000207				RTS	PC	;RETURN
7274								
7275	042120	004037	050706		35\$:	JSR	RO,Q.PUSH	;ENQUEUE PARAMETER BLOCK IN
7276	042124	001314				AVAILQ		; AVAILABLE QUEUE
7277	042126	004737	022016			JSR	PC,GETBUF	;GET NEXT BUFFER
7278	042132	005037	001562			CLR	ERRPRO	;CLEAR ERROR BEING PROCESSED
7279	042136	000207				RTS	PC	;RETURN
7280								
7281	042140	042765	100200	000150	40\$:	BTC	#BIT15BIT7,P.DSTT(R5)	;CLEAR FIRST ERROR AND ERROR ENQUEUED
7282	042146	122737	000010	000126		CMPP	#P.,P.RSEC	;CHECK IF ALL SECTORS HAVE BEEN READ
7283	042154	101422				BLOS	45\$	;YES, DROP DRIVE FROM TEST SEQUENCE
7284	042156	116565	000123	000135		MOV	P.RCMD(R5),P.LCMD(R5)	;STORE PREVIOUS COMMAND
7285	042164	016565	000124	000136		MOV	P.RCYL(R5),P.LCYL(R5)	
7286	042172	016565	000126	000140		MOV	P.RSEC(R5),P.LSEC(R5)	
7287	042200	016537	000130	000142		MOV	P.RVC(R5),P.LVC	
7288	042206	004037	050706			JSR	RO,Q.PUSH	;ENQUEUE PARAMETER BLOCK IN COMMAND
7289	042212	001324				CINITQ		; INITIATION QUEUE
7290	042214	005037	001562			CLR	ERRPRO	;CLEAR ERROR RECOVERY IN PROGRESS
7291	042220	000207				RTS	PC	;RETURN
7292								
7293	042222	004737	022316		45\$:	JSR	PC,BUPREL	;GC RELEASE BUFFER
7294	042226	104401	063003			TYPE	,ERP204	
7295	042232	004737	012716			JSR	PC,DROP	;DROP DRIVE FROM TEST SEQUENCE
7296	042236	004737	022016			JSR	PC,GETBUF	;GET NEW BUFFER
7297	042242	005037	001562			CLR	ERRPRO	;CLEAR ERROR RECOVERY IN PROGRESS
7298	042246	000207				RTS	PC	;RETURN
7299								
7300	042250	052765	000004	000712	50\$:	BTS	#BIT2,P.ERR(R5)	;SET DATA TYPE ERROR
7301	042256	000137	034512			JMP	E105	;GO PROCESS DATA ERROR

```

7302 .SRTTL  ERROR RECOVERY SEQUENCE ABNORMAL RETURN
7303
7304 042262 032765 000120 000150 ERCHV7: BIT 0BIT4BIT6,P.DSTT(R5) ;CHECK IF DIAGNOSTIC SEQUENCE
7305 042270 001524 MEQ 35$ ;NO, GO TO ERROR RECOVERY SEQUENCE
7306 042272 004737 027247 JSR PC,PR1000 ;PRINT HEADFF
7307 042276 104401 063172 TYPF ,FRF208 ;TYPE "ERRCH WHILE DIAGNOSING ---"
7308 042302 032765 000020 000150 BIT 0BIT4,P.DSTT(R5) ;CHECK IF HEADFF TYPE ERCH
7309 042310 001005 BNE 2$ ;YES, PRINT MESSAGE
7310 042312 104401 061427 TYPF ,FRF052+1 ;TYPE "TIME LOT WHILE DRIVE POSITIONING"
7311 042316 104401 001165 TYPE ,SCPLF ;TYPE <CR><LF>
7312 042322 000402 BP 3$ ;GO PRINT STATUS
7313
7314 042324 104401 063224 2$: TYPF ,FRF209+1 ;PRINT HEADFF TYPE ERCH
7315 042330 032777 020000 136602 3$: BIT 0SM13,0SWP ;CHECK IF INHIBIT PRINT OUT
7316 042336 001066 BNE 30$ ;YES, DO NOT PRINT STATUS
7317 042340 032765 000020 000217 BIT 0BIT4,P.ERR(R5) ;CHECK IF DRIVE SEIZED TIME OUT
7318 042346 001402 BEQ 4$
7319 042350 104401 061744 TYPE ,ERR059
7320 042354 032765 000040 000212 4$: BIT 0BIT5,P.ERR(R5) ;CHECK IF TIME OUT WHILE DRIVE POSITIONING
7321 042362 001402 BEQ 5$
7322 042364 104401 061427 TYPE ,FRF052+1
7323 042370 032765 000010 000014 5$: BIT 0UEVATT,P.POST(R5) ;CHECK IF UNEXPECTED ATTENTION
7324 042376 001402 BEQ 6$
7325 042400 104401 061027 TYPF ,FRF070
7326 042404 032765 000020 000014 6$: BIT 0DEVMMD,P.FIRST(R5) ;CHECK IF DRIVE HAND ERROR
7327 042412 001402 BEQ 7$
7328 042414 104401 060744 TYPF ,FRF027
7329 042420 032765 000040 000014 7$: BIT 0DRVUSC,P.POST(R5) ;CHECK IF DSC CTR NOT CLEAR
7330 042426 001402 BEQ 8$
7331 042430 104401 060763 TYPE ,FRF028
7332 042434 032765 004000 000014 8$: BIT 0MODSC,P.FIRST(R5) ;CHECK IF ATTN BUT NO DSC OR FAULT
7333 042442 001402 BEQ 11$
7334 042444 104401 061667 TYPF ,FRF058
7335 042450 004737 025550 11$: JSR PC,PR1STT ;PRINT CONTROLLER STATUS
7336 042454 004737 025374 JSR PC,PRDSTT ;PRINT DRIVE STATUS
7337 042460 004737 027476 JSR PC,PR1002 ;PRINT RETRY COMMAND
7338 042464 010546 MOV R5,-(SP) ;STORE R5 ON STACK
7339 042466 012705 004512 MOV 0PAPH10,R5 ;LOAD R5 FOR PREVIOUS STATUS
7340 042472 104401 063243 TYPF ,FRF210 ;TYPE "FIRST ERCH INFO"
7341 042476 004737 025550 JSR PC,PR1STT ;PRINT CONTROLLER STATUS
7342 042502 004737 025374 JSR PC,PRDSTT ;PRINT DRIVE STATUS
7343 042506 004737 027476 JSR PC,PR1002 ;PRINT INITIAL COMMAND
7344 042512 012605 MOV (SP)+,R5 ;RESTORE R5
7345 042514 005765 000206 30$: INC P.ERR(R5) ;INCREMENT OTHER ERRORS
7346 042520 004737 030472 JSR PC,HLT000 ;CHECK IF HALT ON ERROR
7347 042524 004737 022316 JSR PC,BUFREL ;RELEASE BUFFERS
7348 042530 004737 012716 JSR PC,DROP ;DROP DRIVE
7349 042534 004737 022016 JSR PC,GETBUF ;GO ALLOCATE BUFFERS
7350 042540 000405 BP 40$ ;CHECK IF ERROR QUEUE EMPTY
7351
7352 042542 004737 031224 35$: JSR PC,ERCHV7 ;GO DO ERROR RECOVERY
7353 042546 005737 001567 TST ERPRO ;CHECK IF ERROR RECOVERY ON LAST DRIVE
7354 ; IS FINISHED
7355 042552 001007 BNE 45$ ;NO, RETURN
7356 042554 004037 050766 40$: JSR RO,Q.POP ;GET NEXT ELEMENT FROM
7357 042560 001330 ERPRO ; ERCH PROCESSING QUEUE

```

HK611/RK06 PERFORMANCE EXERCISER MAINDEC DZR6P-B
DZR6PP.P11 11-AUG-76 00:00

MACY11 30(1046) 01-DEC-77 14:15 PAGE 162
ERROR RECOVERY SEQUENCE ABNORMAL RETURN

SEQ 0184

7358	042562	012605		MOV	(SP)+,R5	;LOAD PARAMETER BLOCK
7359	042564	010537	001562	MOV	R5,ERRPPG	;LOAD CURRENT PROCESSING ERROR
7360	042570	001364		BNE	355	;IF ERROR PROCESSING QUEUE NOT EMPTY GO
7361						; PROCESS NEXT ERROR
7362	042572	000207		455:	RTS	;RETURN

```

7363 .SBTTL DETERMINE IF CLOCK IS PRESENT
7364
7365 042574 012737 042624 000004 CHKCLK: MOV    $5$,ERRVEC    ;SET UP ERROR VECTOR FOR NON-EXISTENT MEMCHV
7366 042602 012737 000340 000006      MOV    $PR7,PRRVEC+2 ;LOCK ALL INTERRUPTS
7367 042610 005777 136522      TST    $SLKCSB    ;CHECK FOR P CLOCK
7368 042614 112737 177777 001374      MOVR   $-1,CLKPLG    ;INDICATE RW11-P PRESENT
7369 042622 000413      BR      20$      ;LOAD COMMON CLOCK PARAMETERS
7370
7371 042624 022626      SS:    CMP    (SP)+,(SP)+    ;ADJUST STACK
7372 042626 012737 042650 000004      MOV    $15$,ERRVEC    ;SET UP ERROR VECTOR FOR NON-EXISTENT MEMCHV
7373 042634 005777 136502      TST    $SLKS      ;CHECK FOR L CLOCK
7374 042640 112737 000001 001374      MOVR   $1,CLKPLG    ;SET RW11-L PRESENT
7375 042646 000401      BR      20$      ;REINSTATE TRAP CATCHER
7376
7377 042650 022626      15$:    CMP    (SP)+,(SP)+    ;ADJUST STACK
7378 042652 012737 000006 000004      20$:    MOV    $PRRVEC+2,ERRVEC ;REINSTATE TRAP CATCHER
7379 042660 005037 000006      CLR     ERRVEC+7
7380 042664 000207      RTS      PC      ;RETURN

```

```

7381                      .SBTTL SFT UP CLOCK INTERRUPT
7382
7383 042666 105737 001374 CLKINT: TSTP CLKPLG ;CHECK IF A CLOCK IS ON SYSTEM
7384 042672 001431          BFQ 20$ ;NO, RETURN
7385 042674 100011          BPL 5$ ;CHECK IF RM11-L
7386 042676 013701 001340          MOV $LPVEC,R1 ;LOAD VFCYCR ADDRESS
7387 042702 012777 177777 136426          MOV #1,$SLKCSB ;LOAD COUNT BUFFER WITH 1'S
7388 042710 012777 000135 136416          MOV #135,$SLKCSR ;SET COUNT UP, 16 MS, CNT
7389 042716 000405          BR 10$ ;LOAD COMMON CLOCK PARAMETERS
7390
7391 042720 013701 001344 5$: MOV $LLVEC,R1 ;LOAD VFCYCR ADDRESS
7392 042724 012777 000100 136410          MOV #100,$SLKS ;SET UP L CLOCK FOR INTERRUPT MODE
7393 042732 012721 042760 10$: MOV #CLOCK,(R1)+ ;LOAD ADDRESS OF INTERRUPT ROUTINE
7394 042736 012711 000300          MOV #PH6,(R1) ;SET UP FOR PRIORITY 6
7395 042742 113737 001352 001357          MOVR HZ,CLKFRQ ;SET UP CLOCK FREQUENCY
7396 042750 113737 001360 001375          MOVP PERINV,PERICD ;SET UP COUNTUP FOR INITIAL STATISTIC SUMMARY
7397 042756 000207          20$: RTS PC ;RETURN
  
```

```

7398          .SBTTL  KW11-L AND FW11-P INTERRUPT HANDLER
7399
7400 042760 105337 001353      CLOCK:  DFCR  CLKFRQ      ;CHECK IF ONE SECOND PASTED
7401 042764 001041              BNE  10$      ;NO, RETURN
7402 042766 113737 001352 001353      MOVR  H7,CLFRQ    ;REINITIALIZE SYSTEM CLCCR
7403 042774 005237 001372              INC  SECOND     ;INCREMENT SECOND COUNT
7404 043000 022737 000074 001372      CMP   @60.,SECOND ;CHECK IF MINUTE HAS OCCURRED
7405 043006 001030              BNE  10$      ;NO, RETURN
7406 043010 005037 001372              CLR   SPCOND     ;CLEAR SECCNT
7407 043014 105737 001360              TSTB  PERINV     ;CHECK IF INTERVAL STATISTICS
7408 043020 001411              BEQ   SS          ;NO, UPDATE MINUTE COUNT
7409 043022 105337 001375              DECB  PERIOD     ;DECREMENT PERIOD COUNT
7410 043026 001006              BNE  SS          ;CHECK IF END OF PERIOD, IF NOT UPDATE MINUTE COUNT
7411 043030 113737 001360 001375      MOVB  PERINV,PERIOD ;INITIALIZE PERIOD
7412 043036 112737 177777 001627      MOVR  @-1,STATIS  ;INDICATE TIME FOR STATISTIC TYPE OUT
7413 043044 005237 001370              SS:  INC  MINUTE   ;INCREMENT MINUTE COUNTER
7414 043050 022737 000074 001370      CMP   @60.,MINUTE ;CHECK IF HOUR HAS OCCURED
7415 043056 001004              BNE  10$      ;NO, RETURN
7416 043060 005037 001370              CLR   MINUTE     ;CLEAR MINUTE COUNT
7417 043064 005237 001366              INC  HOUR        ;INCREMENT HOUR COUNT
7418 043070 000002              10$:  RTI          ;RETURN

```

```

7419 .SBTTL PPINT TIME ROUTINE
7420
7421 043072 105737 001374 PRITIM: TSTR CLKPLG ;CHECK IF CLOCK ON SYSTEM
7422 043076 001455 BEQ 50$ ;NO, RETURN
7423 043100 010046 MOV R0,-(SP) ;SAVE R0
7424 043102 013746 177776 MOV PS,-(SP) ;SAVE PSW
7425 043106 012737 000340 177776 MOV #PR7,PS ;LOCK GOT CLOCK
7426 043114 013737 001366 001376 MCV HOUR,TIMHR ;SAVE TIME FOR PRINT OUT
7427 043122 013737 001370 001400 MOV MINUTE,TIMMIN
7428 043130 013737 001372 001402 MOV SFCOND,TIMSEC
7429 043136 012637 177776 MOV (SP)+,PS ;RESTORE PSW
7430 043142 012700 063734 MOV #TIM001,R0 ;LOAD R0 FOR INDEX
7431 043146 105010 CLRB (R0) ;DETERMINE HUNDREDS OF HOURS
7432 043150 013746 001376 MOV TIMHR,-(SP) ;STORE HOURS FOR CONVERSION
7433 043154 162716 000144 1$: SUB #100,-(SP) ;CONVERT HUNDREDS
7434 043160 100402 BMI 2$ ;CHECK IF TOO SMALL
7435 043162 105210 INCB (R0)
7436 043164 000773 BR 1$
7437
7438 043166 062716 000144 2$: ADD #100,-(SP)
7439 043172 152720 000060 BISR #60,(R0)+ ;MAKE DIGIT ASCII
7440 043176 004737 043234 JSR PC,CONHUN
7441 043202 013716 001400 MCV TIMMIN,(SP) ;LOAD STACK FOR MINUTES
7442 043206 004737 043234 JSR PC,CONHUN
7443 043212 013716 001402 MOV TIMSEC,(SP) ;LOAD STACK FOR SECONDS
7444 043216 004737 043234 JSR PC,CONHUN
7445 043222 005726 TST (SP)+ ;ADJUST STACK
7446 043224 104401 063727 TYPE ,TIM000 ;TYPE TIME:
7447 043230 012600 MOV (SP)+,R0 ;RESTORE R0
7448 043232 000207 50$: RTS PC ;RETURN
7449
7450 .SBTTL CONVERT DECIMAL LESS THAN 100
7451
7452 043234 105010 CONHUN: CLRB (R0) ;ZERO DIGIT
7453 043236 162766 000012 000002 1$: SUB #10,-2(SP) ;CONVERT TENS
7454 043244 100402 BMI 5$
7455 043246 105210 INCB (R0)
7456 043250 000773 BR 1$
7457
7458 043252 062766 000012 000002 5$: ADD #10,-2(SP)
7459 043260 152720 000060 BISR #60,(R0)+ ;MAKE ASCII
7460 043264 105010 CLRB (R0) ;ZERO DIGIT
7461 043266 005766 000002 TST 2(SP)
7462 043272 001404 BEQ 7$
7463 043274 105210 6$: INCB (R0) ;CONVERT ONES
7464 043276 005766 000002 DEC 2(SP)
7465 043302 001374 BNE 6$
7466 043304 152720 000060 7$: BISR #60,(R0)+ ;MAKE ASCII
7467 043310 105720 TSTR (R0)+ ;SKIP OVER COLUMN
7468 043312 000207 RTS PC

```

```

7469 .SBTTL RK611/RK06 UNIBUS DRIVER FOR QUEUED OPERATIONS (REV. 0.08)
7470
7471 ;*COPYRIGHT (C) 1975
7472 ;*DIGITAL EQUIPMENT CORP.
7473 ;*MAYNARD, MA. 01754
7474 ;*AUTHOR: ROY SPITZER
7475
7476 .SBTTL *WATCH-DOG TIMER
7477
7478 ;;*****
7479 ;*
7480 ;* THE WATCH-DOG TIMER DOES A PSPUDC-TIMING OF RK06 UNIBUS
7481 ;* SUBSYSTEM COMMAND. SINCE ONE CAN NOT GUARANTEE THAT A
7482 ;* REAL-TIME CLOCK (RM11-P OR RM11-L) IS ON THE SYSTEM
7483 ;* THE RK06 DRIVER WILL USE THE LOCATION W.MTIM FOR
7484 ;* MILLI-SECOND TIMING. WHEN W.MTIM REACHES ZERO THE
7485 ;* WATCH-DOG TIMER WILL SCAN THE DRIVES IN USE AS
7486 ;* DETERMINED BY THE LOCATION W.TIME. THE TIMER COUNTS
7487 ;* (ONE FOR EACH DRIVE) ARE KEPT IN THE TABLE W.DRV.
7488 ;* IF ANY COUNT IN THE TABLE W.DRV REACHES ZERO A COMMAND
7489 ;* TIME-OUT WILL BE DESIGNATED IN THE PCCFAP DEVICE STATUS
7490 ;* REGISTER OF THAT DRIVE'S PARAMETER BLOCK.
7491 ;*
7492 ;* THE DRIVER WILL USE THE LOCATION W.WIN AS THE NUMBER
7493 ;* OF MILLISECONDS FOR AN UNLOAD OR START SPINDLE COMMAND.
7494 ;* THE DRIVER WILL USE THE LOCATION W.SEC AS THE TIME
7495 ;* LIMIT FOR ALL OTHER COMMANDS.
7496 ;*
7497 ;* FOR QUEUED OPERATIONS THE WATCH-DOG TIMER WILL
7498 ;* WATCH UP TO 8 OPERATIONS SIMULTANEOUSLY. FOR SEQUENTIAL
7499 ;* OPERATIONS ONLY ONE OPERATION WILL BE WATCHED.
7500 ;*
7501 ;*CALL JSR PC,W.WTCH
7502 ;* RETURN IF NO DRIVE CPOPR EXCEEDED ITS TIME LIMIT
7503 ;*
7504 ;* OTHERWISE AN ABNORMAL RETURN TO THE ROUTINE ADDRESS
7505 ;* BY LOCATION A.ABNL WILL OCCUR AND THE CMCTC FLAG
7506 ;* IN THE PROGRAM DEVICE STATUS REGISTER OF THE
7507 ;* APPROPRIATE PARAMETER BLOCK WILL BE SET.
7508 ;*
7509 ;;*****
7510
7511 043314 010546 W.WTCH: MOV R5,-(SP) ;SAVE R5 ON THE STACK
7512 043316 010446 MOV R4,-(SP) ;SAVE R4 ON THE STACK
7513 043320 010346 MOV R3,-(SP) ;SAVE R3 ON THE STACK
7514 043322 010246 MOV R2,-(SP) ;SAVE R2 ON THE STACK
7515 043324 013746 177776 MOV PS,-(SP) ;SAVE PROGRAM STATUS WORD ON STACK
7516 043330 005337 001464 DEC W.MTIM ;DECREMENT MILLISECOND TIMER
7517 043334 001046 BNE 20$ ;IF NOT ZERO RETURN
7518 043336 013737 001466 001464 MOV W.MILI,W.MTIM ;REINITIALIZE MILLISECOND TIMER
7519 043344 013737 001450 177776 MOV RFPRI,PS ;LOCK OUT RK06 INTERRUPTS
7520 043352 013702 001444 MOV RFBAS,R2 ;LOAD BASE OF RK06 REGISTERS
7521 043356 005004 CLR R4 ;CLEAR INDEX OF DRIVE MASKS
7522 043360 005003 CLR R? ;CLEAR INDEX OF DRIVE TIMES
7523 ; AND PARAMETER BLOCK ADDRESSES
7524 043362 136437 001510 001506 1$: BITB INTMSR(R4),W.TIME ;CHECK IF DRIVE IS BEING TIMED

```

7525	043370	001422			BFG	6S		;NO, GO TO NEXT DRIVE
7526	043372	005363	001540		DEC	W.DPV(R ²)		;DECREMENT DRIVE COUNT
7527	043376	001017			BNE	6S		;IF NOT TIME CUT, GO TO
7528								; NEXT DRIVE
7529	043400	016305	001520		MOV	PRLKT(R3),R5		;LOAD ADDRESS OF PARAMETER
7530								; BLOCK TABLE FOR INDEXING
7531	043404	146437	001510	001506	BTCP	INTWST(R4),W.TIME		;RESET TIMING INDICATOR FOR DRIVE
7532	043412	052765	000100	000014	BIS	#CMDT0,P.PRST(R5)		;SET COMMAND TIME OUT
7533	043420	020537	001467		CMP	R5,O.WAIT		;CHECK IF DRIVER IS WAITING FOR
7534								; COMMAND COMPLETION
7535	043424	001002			BNE	5S		;NO, DO NOT ALTER WAITING FOR
7536								; COMMAND COMPLETION
7537	043426	005037	001462		CLR	O.WAIT		;CLEAR WAIT FOR COMMAND COMPLETION
7538	043432	004777	136016		JSR	PC,RA.ARM		;BRANCH TO ENHCR PCOTIME
7539	043436	062703	000002		ADD	#2,R3		;ADD 2 TO INDEX OF DRIVE TIMES
7540								; AND PARAMETER BLOCK ADDRESSES
7541	043442	005204			INC	R4		;INCRMENT INDEX OF DRIVE NAME
7542	043444	022704	000010		CMP	#10,R4		;CHECK IF ALL DRIVES ARE TESTED
7543	043450	001344			BNE	1S		;NO, CHECK NEXT DRIVE
7544	043452	012637	177776		MOV	(SP)+,PS		;RESTORE PSM
7545	043456	012602			MOV	(SP)+,R2		;RESTORE R2
7546	043460	012603			MOV	(SP)+,R3		;RESTORE R3
7547	043462	012604			MOV	(SP)+,R4		;RESTORE R4
7548	043464	012605			MOV	(SP)+,R5		;RESTORE R5
7549	043466	000207			RTS	PC		;RETURN

5S:
6S:

20S:

```

7550 .SBTTL *RK06 INTERRUPT SERVICE ROUTINE
7551
7552 ;*****
7553 ;*
7554 ;* THIS ROUTINE WILL SERVICE ALL RK06 INTERRUPTS.
7555 ;*
7556 ;* UPON RECEIVING AN INTERRUPT, THIS ROUTINE WILL
7557 ;* PERFORM ONE OF THE FOLLOWING SERVICES:
7558 ;*
7559 ;* 1.) SERVICE PORT WAS SEIZED BY OTHER PORT
7560 ;* 2.) SERVICE DRIVER IS WAIT FOR COMMAND COMPLETION
7561 ;* 3.) SERVICE POSITIONING COMPLETION
7562 ;* 4.) REQUEUE COMMAND IF DRIVE WAS RELEASED
7563 ;* FOR THE QUEUED RK06 DRIVER.
7564 ;* 5.) IF NO SERVICE IS REQUIRED, THE COMMAND WILL BE ISSUED
7565 ;* FOR THE QUEUED RK06 DRIVER.
7566 ;*
7567 ;* THREE LINKS ARE PROVIDED TO THE DRIVING PROGRAM.
7568 ;* THEY ARE:
7569 ;*
7570 ;* 1.) A.NORM ADDRESS OF NORMAL RETURN (SUCCESSFUL COMPLETION OF COMMAND)
7571 ;* 2.) A.ABNL ADDRESS OF ABNORMAL RETURN (UNSUCCESSFUL COMPLETION OF COMMAND)
7572 ;* 3.) A.CONT ADDRESS OF CONTROL ERROR RETURN
7573 ;*
7574 ;* FOR NORMAL AND ABNORMAL RETURNS, THE ADDRESS OF THE APPROPRIATE
7575 ;* PARAMETER BLOCK WILL BE IN R5.
7576 ;*
7577 ;* FOR THE CONTROLLER ERROR RETURN, THE LOCATION E.CONT CONTAINS
7578 ;* THE REASON FOR THE CONTROLLER ERROR.
7579 ;*
7580 ;* ROUTINES USED:
7581 ;* C.OPT (QUEUED ONLY)
7582 ;* Q.PUSH (QUEUED ONLY)
7583 ;* Q.REOV (QUEUED ONLY)
7584 ;* R.CONT (SEQUENTIAL ONLY)
7585 ;* R.NORM (SEQUENTIAL ONLY)
7586 ;* R.ABNL (SEQUENTIAL ONLY)
7587 ;* I.CSTS
7588 ;* I.STAT
7589 ;* I.ISSU
7590 ;* I.CCLR
7591 ;*
7592 ;*****
7593
7594 043470 010546 I.INTR: MOV R5,-(SP) ;STORE R5 ON THE STACK
7595 043472 010446 MOV R4,-(SP) ;STORE R4 ON THE STACK
7596 043474 010346 MOV R3,-(SP) ;STORE R3 ON THE STACK
7597 043476 010246 MOV R2,-(SP) ;STORE R2 ON THE STACK
7598 043500 010146 MOV R1,-(SP) ;STORE R1 ON THE STACK
7599 043502 010046 MOV R0,-(SP) ;STORE R0 ON THE STACK
7600 043504 013702 001444 MOV RKBAS,R2 ;LOAD R2 TO ADDRESS RK06 REGISTER
7601 043510 016737 000010 001410 MOV RKCS2(R?),T.CS2 ;STORE CS2
7602 043516 032737 001000 001410 BIT #MDS,T.CS2 ;CHECK IF MULTIPLE DRIVE SELECT
7603 043524 001407 BFC 15 ;NO, CONTINUE PROCESSING
7604 043526 052737 100000 001460 BIS #E.MDS,F.CONT ;SET MULTIPLE DRIVE SELECT
7605 043534 004777 135716 JSR PC,RA.CONT ;REPORT ERROR

```

```
7606 043540 000137 046214      JMP      I.RTRN      ;RETURN
7607
7608 043544 105737 001502      1S:    TSTB     I.ISRL      ;CHECK IF INTERRUPT OR RELEASE
7609 043550 001410              BEQ      6S          ;NO, CHECK IF DRIVE AVAILABLE
7610 043552 100403              BMI     5S          ;CHECK IF RELEASE COMMAND
7611 043554 105037 001502      CLRP     I.ISRL      ;YES, CLPAR FLAG
7612 043560 000504              BP      I.I00        ;CONTINUE PROCESSING INTERRUPT
7613
7614 043562 105037 001502      5S:    CLRP     I.ISRL      ;CLEAR FLAG
7615 043566 000137 044740      JMP      I.ATTN      ;GO PROCESS DRIVE ATTENTIONS
7616
7617 043572 032737 010400 001410 6S:    BIT      @NEDIOPE,T.CS2    ;CHECK FOR NON-EXISTENT DRIVE OF
7618                                ; UNIT FIELD ERROR
7619 043600 001415              BEQ      7S          ;NO, WAIT FOR DUAL ACCESS INTERRUPT
7620 043602 013704 001410      MOV      T.CS2,R4      ;LOAD R4 FOR DRIVE NUMBER
7621 043606 042704 177770      BIC      @C<DRVMSK>,R4    ;FREE DRIVE BITS
7622 043612 010403              MOV      R4,P3        ;STORE DRIVE NUMBER FOR INDEX
7623 043614 006303              ASL      R3          ;MULTIPLY BY 2
7624 043616 016305 001520      MOV      PBLKT(R3),R5    ;STORE PARAMETER BLOCK ADDRESS
7625 043622 016237 000000 001406      MOV      RKCS1(R2),T.CS1    ;LOAD TEMPORARY CS1 FOR STATUS REPORT
7626 043630 000137 044142      JMP      I.ERMC      ;REPORT ERROR
7627
7628 043634 016237 000012 001426 7S:    MOV      RKDS(R2),T.DS    ;STORE STATUS REGISTER FOR COMPARISON
7629 043642 032737 000001 001426      BIT      @DRA,T.DS      ;CHECK IF DRIVE SEIZED BY OTHER
7630                                ; PORT
7631 043650 001050              BNE     I.I00        ;NO, CONTINUE PROCESSING INTERRUPT
7632
7633                                ;CHECK IF ANY DATA TRANSFER ERROR EXISTS
7634 043652 032737 164000 001410      BIT      @DLTIMEIOPEINEM,T.CS2
7635
7636 043660 001007              BNE     10S         ;INDICATE ERROR
7637 043662 016237 000014 001424      MOV      RKER(R2),T.ER    ;STORE ERROR REGISTER
7638
7639                                ; CHECK FOR DATA TRANSFER ERROR TYPE PEECF
7640 043670 032737 125700 001424      BIT      @DCKIOPTIMLEICOEIRVRCIBSEIECH,T.ER
7641
7642 043676 001407              BEQ      11S         ;NO, WAIT FOR RELEASE OF RK06 DRIVE
7643
7644 043700 052737 000010 001460 10S:   BIS      @E.UDAT,E.CONT    ;SET UNEXPECTED DATA TYPE ERROR
7645 043706 004777 135544              JSR      PC,PA.COMT    ;REPORT ERROR
7646 043712 000137 046214              JMP      I.RTRN      ;RESTORE REGISTERS
7647
7648 043716 013703 001410      11S:   MOV      T.CS2,R3      ;STORE RKCS2 FOR DRIVE NUMBER
7649 043722 042703 177770              BIC      @C<DRVMSK>,R3    ;STRIP OFF JUNK
7650 043726 006303              ASL      R3          ;MULTIPLY INCPX BY 2
7651 043730 013763 001474 001540      MOV      W.MIN,W.DPV(R3)    ;WAIT ONE MINUTE FOR RELEASE
7652 043736 016305 001520      MOV      PBLKT(R3),R5    ;LOAD R5 WITH PARAMETER
7653                                ; BLOCK ADDRESS
7654 043742 004737 051126              JSR      PC,Q.WMCV      ;REMOVE PARAMETER BLOCK FROM
7655                                ; COMMAND INITIATION QUEUE
7656 043746 052765 010000 000014      BIS      @DRVS7D,P.PPST(R5) ;SET DRIVE SPIED IN THE
7657                                ; PROGRAM DRIVE STATUS REGISTER
7658 043754 005037 001462              CLR      Q.WAIT      ;CLEAR WAIT FOR COMMAND COMPLETION
7659 043760 004037 046232              JSR      RO,I.CCLR      ;CLEAR CONTROLLED
7660 043764 046714              I.RTRN      ;ERROR RETURN
7661 043766 000137 046214              JMP      I.RTRN      ;GO RESTORE REGISTERS
```

```

7662
7663 043777 013705 001462      I.I00: MOV    O.WAIT,R5      ;LOAD PARAMETER BLOCK ADDRESS INTO R5
7664 043776 001002              BNE     2$              ;IS COMMAND WAITING PROCESSING
7665                                ; YES, DO PROCESSING
7666 044000 000137 044740              JMP     I.ATTN      ;NO, PROCESS ATTENTION
7667
7668 044004 013704 001410      2$:  MOV    T.CS2,R4      ;STORE RKCS2 FOR DRIVE NUMBER
7669 044010 042704 177770              BTC     @C<DRVMSK>,R4 ;MASK OUT UNNECESSARY BITS
7670
7671
7672 044014 010407              MOV    R4,R3      ;STORE DRIVE NUMBER FOR INDEX
7673 044016 006303              ASL     R3      ;MULTIPLY BY 2 FOR INDEX
7674
7675 044020 126504 000000              CMPB   P.DRVN(R5),R4 ;CHECK IF DRIVE NUMBER IS EXPECTED
7676 044024 001401              BEQ     3$              ;YES, CONTINUE
7677 044026 000000              HALT                    ;NO, DRIVER ERROR
7678 044030 122765 000164 000001 3$:  CMPB   @PDALND,P.CMND(R5) ;CHECK IF READ ALL HEADERS
7679 044036 001002              BNE     10$             ;NO, EXECUTE NORMAL DATA TRANSFER
7680 044040 000137 044400              JMP     I.HDAL      ;GO EXECUTE SPECIAL HEADER SEQUENCE
7681
7682 044044 005037 001462      10$:  CLR     O.WAIT      ;CLEAR WAIT FOR COMMAND COMPLETION
7683 044050 005063 001540              CLR     W.DRV(R?)    ;CLEAR WATCH-DOG TIME
7684 044054 146437 001510 001506      BITB   INTMSK(R4),W.TIME ;RESET TIMING ON THIS DRIVE
7685 044062 016237 000000 001406      MOV    RKCS1(R2),T.CS1 ;STORE COMMAND AND STATUS REGISTER 1
7686 044070 032737 100000 001406      BIT    @CERR,T.CS1    ;CHECK IF CONTROLLER ERROR
7687 044076 001021              BNE     I.ERRC      ;YES, PROCESS ERROR
7688 044100 016237 000016 001422      MOV    RKASOF(R2),T.ASOF ;STORE ATTENTION SUMMARY
7689 044106 136437 001510 001423      BITB   INTMSK(R4),T.ASOF+1 ;CHECK IF DRIVE ATTENTION SET
7690 044114 001004              BNE     15$             ;YES, REPORT ERROR
7691 044116 004777 135330              JSR    PC,RA.NORM    ;INDICATE NORMAL RETURN
7692 044122 000137 046214              JMP     I.RTRN      ;RESTORE REGISTERS
7693
7694 044126 052765 000010 000014 15$:  BIS     @UEXATT,P.FRST(R5) ;SET UNEXPECTED ATTENTION
7695
7696 044134 004737 046664      I.ERRA: JSR    PC,I.CSTS      ;STORE CONTROLLER STATUS
7697 044140 000405              BP     I.ERN      ;STORE PATTERN AND POSITION INFORMATION
7698
7699 044142 013765 001406 000016  I.ERRC: MOV    T.CS1,P.CS1(R?) ;GET PERCR RKCS1
7700 044150 004737 046706              JSR    PC,I.CST1      ;GET REST OF CONTROLLER STATUS
7701 044154 016765 000037 000067  I.ERN:  MOV    RKECPT(R2),P.PPAT(R5) ;STORE ECC PATTERN
7702 044162 016765 000030 000060      MOV    RKECPS(R2),P.FPCS(R?) ;STORE ECC POSITION
7703 044170 004037 046232              JSR    RO,I.CCLR      ;CLEAR CONTROLLER
7704 044174 046214              I.RTRN      ;ERROR RETURN
7705 044176 032765 010400 000020      BIT    @MEDIUPE,P.CS2(R5) ;CHECK IF IT WAS NON-EXISTENT DRIVE CH
7706                                ; UNIT FIELD ERROR
7707                                BNE     5$              ;YES, REPORT ERROR
7708 044206 004037 046770              JSR    RO,I.STAT      ;GATHER DRIVE STATUS
7709 044212 046214              I.RTRN      ;ERROR RETURN
7710 044214 112737 000005 001406      MOVB   @DR.CLR,T.CS1  ;LOAD COMMAND
7711 044222 004037 046314              JSR    RO,I.TSSU      ;ISSUE DRIVE CLEAR
7712 044226 046214              I.RTRN      ;ERROR RETURN
7713 044230 136437 001510 001423      BITB   INTMSK(R4),T.ASOF+1 ;CHECK IF ATTENTION RESET
7714 044236 001407              BEQ     2$              ;NO, INDICATE DRIVE ERROR
7715 044240 052737 000020 001460      BIS     @E.CLAT,E.CMNT ;SET ATTENTION DID NOT RESET
7716                                ; WITH CLEAR
7717 044246 004777 135204              JSR    PC,RA.COMT     ;REPORT CONTROLLER ERROR

```

```
7718 044252 000137 046214      JMP      I.RTRN      ;GO RESTORE REGISTERS
7719
7720 044256 032737 040000 001432 2S:  BIT      %S.DSC,T.MR2    ;CHECK IF DRIVE STATUS CHANGE CLEARED
7721 044264 001403                BEQ      3$              ;YES, CHECK FAULT
7722 044266 052765 000040 000014      BIS      %DRVDSR,P.PRST(R5) ;SET DSC DID NOT CLEAR
7723 044274 032737 001000 001434 3S:  BIT      %S.PAR,T.MR3    ;CHECK IF DRIVE PARITY ERROR
7724 044302 001407                BEQ      5$              ;NO, INDICATE ABNORMAL TERMINATION
7725 044304 052737 002000 001460      BIS      %E.DPAR,E.CONT    ;SET DRIVE PARITY ERROR
7726 044312 004777 135140                JSR      PC,PA.CONT    ;INDICATE CONTROLLER ERROR
7727 044316 000137 046214      JMP      I.RTRN      ;RETURN
7728
7729 044322 032765 000020 000014 5S:  BIT      %DRVHND,P.PRST(R5) ;CHECK IF HARD DRIVE ERROR
7730 044330 001017                BNE      10$             ;YES, GO REPORT ERROR
7731 044332 032737 020000 001432      BIT      %S.PIP,T.MR2    ;CHECK IF DRIVE IS CYCLING DOWN
7732 044340 001413                BEQ      10$             ;NO, REPORT ERROR
7733 044342 052765 020000 000014      BIS      %E.UNLD,P.PRST(R5) ;SET DRIVE UNLOADING
7734 044350 156437 001510 001506      BISB   INTMSK(R4),W.TIME ;SET UP 8 SECONDS FOR
7735 044356 013763 001472 001540      MOV      W.8SEC,W.DRV(R3) ; DRIVE TO CYCLE UP
7736 044364 000137 046214      JMP      I.RTRN      ;GO RESTORE REGISTERS
7737
7738 044370 004777 135060                10$: JSR      PC,PA.ABNL    ;GO REPORT ERROR
7739 044374 000137 046214      JMP      I.RTRN      ;GO RESTORE REGISTERS
7740
7741                .SBTTL    *READ ALL HEADERS INTERRUPT SEQUENCE
7742
7743 044400 016237 000000 001406 I.HDAL: MOV      RKCS1(R2),T.CS1 ;STORE CS1 TO CHECK CONTROLLER
7744                                ; ERROR
7745 044406 032737 100000 001406      BIT      %CERR,T.CS1    ;CHECK IF CONTROLLER ERROR
7746 044414 001423                BEQ      5$              ;NO, CHECK FOR ATTENTION
7747
7748 044416 005037 001462                CLR      D.WAIT        ;CLEAR WAITING FOR COMMAND COMPLETE
7749 044422 146437 001510 001506      BISB   INTMSK(R4),W.TIME ;RESET TIMING ON DRIVE
7750 044430 005063 001540                CLR      W.DRV(R3)      ;CLEAR TIME OUT COUNT
7751 044434 013765 001406 000016      MOV      T.CS1,P.CS1(R5) ;STORE ERROR RKCS1
7752 044442 004737 046706                JSR      PC,I.CST1    ;STORE CONTROLLER REGISTERS
7753 044446 004037 046232                JSR      RO,I.CCLR    ;CLEAR CONTROLLER
7754 044452 046214                I.RTRN      ;ERROR RETURN
7755 044454 004777 134774                JSR      PC,PA.ABNL    ;INDICATE ERROR RETURN
7756 044460 000137 046214      JMP      I.RTRN      ;RESTORE REGISTERS
7757
7758 044464 016537 000016 001422 5S:  MOV      RKASOF(R5),T.ASOF ;STORE ATTENTION SUMMARY
7759 044472 136437 001510 001422      BISB   INTMSK(R4),T.ASOF+1 ;CHECK IF DRIVE ATTENTION IS SET
7760 044500 001411                BEQ      7$              ;NO, CHECK IF REAT ALL HEADERS
7761 044502 005037 001462                CLR      D.WAIT        ;CLEAR WAITING FOR COMMAND COMPLETION
7762 044506 146437 001510 001506      BISB   INTMSK(R4),W.TIME ;RESET TIMING ON DRIVE
7763 044514 005063 001540                CLR      W.DRV(R3)      ;CLEAR TIME OUT COUNT
7764 044520 000137 044134      JMP      I.ERRA      ;GO REPORT ERROR
7765
7766 044524 013701 001476                7$: MOV      HDR.AD,R1      ;GET MAIN MEMORY ADDRESS
7767 044530 016221 000024                MOV      RKDP(R2),(R1)+ ;GET FIRST WORD OF HEADER
7768 044534 016221 000024                MOV      RKDB(R2),(R1)+ ;GET SECOND WORD OF HEADER
7769 044540 016221 000024                MOV      RKDB(R2),(R1)+ ;GET THIRD WORD OF HEADER
7770 044544 010137 001476                MOV      R1,HDR.AD      ;STORE ADDRESS FOR NEXT HEADER
7771 044550 016237 000010 001410      MOV      RKCS2(R2),T.CS2 ;STORE CS2 TO CHECK FOR DATA LATE
7772 044556 032737 100000 001410      BIT      %DLT,T.CS2    ;CHECK FOR DATA LATE
7773 044564 001056                BNE      35$             ;YES, REPORT ERROR
```

7774	044566	005337	001500		DFC	HDR.CT	;DECREMENT NUMBER OF HEADERS YET TO READ
7775	044572	001027			BNE	255	;IF NON-ZERO, GO ISSUE NEXT READ HEADER
7776	044574	005037	001462		CLR	O.WAIT	;CLEAR DRIVE WAITING FOR COMMAND COMPLETION
7777	044600	005063	001540		CLR	W.DRV(R3)	;CLEAR TIME OUT COUNT FOR THIS DRIVE
7778	044604	146437	001510	001506	BICB	INTMSK(R4),W.TIME	;CLEAR WATCH DOG TIME ON THIS DRIVE
7779	044612	012762	000003	000026	MOV	#3,RKMR1(R2)	;LOAD MAINTENANCE REGISTER FOR SECTOR COUNT
7780	044620	112737	000001	001406	MOVB	#DR.SEL,T.CS1	;LOAD SELECT COMMAND
7781	044626	004037	046314		JSR	FO,I.ISSU	;GET SECTOR COUNT
7782	044632	046214			I.RTRN		;ERROR RETURN
7783	044634	013765	001434	000056	MOV	T.MR3,P.B11(R5)	;LOAD SECTOR COUNT
7784	044642	004777	134604		JSR	PC,RA.NORM	;INDICATE NORMAL TERMINATION
7785	044646	000137	046214		JMP	I.RTRN	;RESTORE REGISTERS
7786							
7787	044652	016567	000002	000020	25\$: MOV	P.CYL(R5),PKDCYL(R2)	;LOAD CYLINDER ADDRESS REGISTER
7788	044660	016567	000004	000006	MOV	P.SECT(R5),PKDA(R2)	;LOAD SECTOR AND TRACK
7789	044666	116565	000007	000017	MOVB	P.CS1H(R5),P.CS1+1(R5)	;STORE BITS 0-15 OF CS1
7790	044674	042765	165777	000016	BIC	#C<CDTICFMT>,P.CS1(R5)	;CLEAR ALL BITS EXCEPT FORMAT AND
7791							; DATA TYPE
7792	044702	112765	000125	000016	MOVB	#PDHEAD,P.CS1(R5)	;STORE COMMAND ISSUED
7793	044710	016562	000016	000000	MOV	P.CS1(R5),RKCS1(R2)	;ISSUE READ HEADER
7794	044716	000137	046214		JMP	I.RTRN	;RESTORE REGISTERS
7795							
7796	044722	052737	000400	001460	35\$: BIS	#E.DLT,E.CCNT	;SET DATA LATE WHILE UNLOADING HEADER
7797	044730	004777	134522		JSR	PC,RA.COMT	;PERCENT ERROR
7798	044734	000137	046214		JMP	I.RTRN	;RESTORE REGISTERS
7799							
7800					.SBTTL	*DRIVE ATTENTION SCANNER	
7801							
7802	044740	016237	000000	001406	I.ATTN: MOV	RKCS1(R2),T.CS1	;STORE COMMAND AND STATUS
7803							; REGISTER 1 FOR COMPARISON
7804	044746	032737	100000	001406	BIT	#CERR,T.CS1	;CHECK IF CONTROLLER ERROR OCCURRED
7805	044754	001444			BEQ	55	;NO, CHECK IF ATTENTION
7806							
7807							;CHECK IF ANY DATA TRANSFER TYPE ERROR EXISTS
7808	044756	032737	164000	001410	BIT	#DLTIMECIEIPEINEM,T.CS2	
7809							
7810	044764	001007			BNE	15	;INDICATE ERROR
7811	044766	016237	000014	001424	MOV	WFER(R2),T.EH	;STORE ERROR REGISTER
7812							
7813							CHECK FOR DATA TRANSFER ERROR TYPE
7814	044774	032737	125700	001424	BIT	#DCKIOPTIWLICOFIHVPCIBSEIECH,T.EP	
7815							
7816	045002	001467			BEQ	25	;NO DATA TRANSFER ERROR
7817							
7818	045004	052737	000010	001460	15: BIS	#E.UDAT,E.COMT	;SET UNEXPECTED DATA TYPE ERROR
7819	045012	004777	134440		JSR	PC,RA.COMT	;PERCENT ERROR
7820	045016	000137	046214		JMP	I.RTRN	;RESTORE REGISTERS
7821							
7822	045022	013704	001410		25: MOV	T.CS2,R4	;SAVE CS2 FOR REGISTER NUMBER
7823	045026	042704	177770		BIC	#C<DRVMSF>,R4	;STRIP OFF JUNK
7824	045032	010403			MOV	R4,R3	;STORE DRIVE NUMBER IN R3
7825	045034	006307			ASL	R3	;MULTIPLY DRIVE NUMBER BY 2
7826	045036	146437	001510	001506	BICB	INTMSK(R4),W.TIME	;CLEAR WATCH-DOG TIME FOR DRIVE
7827	045044	005063	001540		CLR	W.DRV(R3)	;RESET WATCH DOG TIME
7828	045050	016305	001520		MOV	PLVLT(R3),R5	;STORE PARAMETER BLOCK ADDRESS IN R5
7829							

Address	Op Code	Op 1	Op 2	Op 3	Op 4	Op 5	Op 6	Op 7	Op 8	Op 9	Op 10	Op 11	Op 12	Op 13	Op 14	Op 15	Op 16	Op 17	Op 18	Op 19	Op 20	Op 21	Op 22	Op 23	Op 24	Op 25	Op 26	Op 27	Op 28	Op 29	Op 30	Op 31	Op 32	Op 33	Op 34	Op 35	Op 36	Op 37	Op 38	Op 39	Op 40	Op 41	Op 42	Op 43	Op 44	Op 45	Op 46	Op 47	Op 48	Op 49	Op 50	Op 51	Op 52	Op 53	Op 54	Op 55	Op 56	Op 57	Op 58	Op 59	Op 60	Op 61	Op 62	Op 63	Op 64	Op 65	Op 66	Op 67	Op 68	Op 69	Op 70	Op 71	Op 72	Op 73	Op 74	Op 75	Op 76	Op 77	Op 78	Op 79	Op 80	Op 81	Op 82	Op 83	Op 84	Op 85	Op 86	Op 87	Op 88	Op 89	Op 90	Op 91	Op 92	Op 93	Op 94	Op 95	Op 96	Op 97	Op 98	Op 99	Op 100	Op 101	Op 102	Op 103	Op 104	Op 105	Op 106	Op 107	Op 108	Op 109	Op 110	Op 111	Op 112	Op 113	Op 114	Op 115	Op 116	Op 117	Op 118	Op 119	Op 120	Op 121	Op 122	Op 123	Op 124	Op 125	Op 126	Op 127	Op 128	Op 129	Op 130	Op 131	Op 132	Op 133	Op 134	Op 135	Op 136	Op 137	Op 138	Op 139	Op 140	Op 141	Op 142	Op 143	Op 144	Op 145	Op 146	Op 147	Op 148	Op 149	Op 150	Op 151	Op 152	Op 153	Op 154	Op 155	Op 156	Op 157	Op 158	Op 159	Op 160	Op 161	Op 162	Op 163	Op 164	Op 165	Op 166	Op 167	Op 168	Op 169	Op 170	Op 171	Op 172	Op 173	Op 174	Op 175	Op 176	Op 177	Op 178	Op 179	Op 180	Op 181	Op 182	Op 183	Op 184	Op 185	Op 186	Op 187	Op 188	Op 189	Op 190	Op 191	Op 192	Op 193	Op 194	Op 195	Op 196	Op 197	Op 198	Op 199	Op 200	Op 201	Op 202	Op 203	Op 204	Op 205	Op 206	Op 207	Op 208	Op 209	Op 210	Op 211	Op 212	Op 213	Op 214	Op 215	Op 216	Op 217	Op 218	Op 219	Op 220	Op 221	Op 222	Op 223	Op 224	Op 225	Op 226	Op 227	Op 228	Op 229	Op 230	Op 231	Op 232	Op 233	Op 234	Op 235	Op 236	Op 237	Op 238	Op 239	Op 240	Op 241	Op 242	Op 243	Op 244	Op 245	Op 246	Op 247	Op 248	Op 249	Op 250	Op 251	Op 252	Op 253	Op 254	Op 255	Op 256	Op 257	Op 258	Op 259	Op 260	Op 261	Op 262	Op 263	Op 264	Op 265	Op 266	Op 267	Op 268	Op 269	Op 270	Op 271	Op 272	Op 273	Op 274	Op 275	Op 276	Op 277	Op 278	Op 279	Op 280	Op 281	Op 282	Op 283	Op 284	Op 285	Op 286	Op 287	Op 288	Op 289	Op 290	Op 291	Op 292	Op 293	Op 294	Op 295	Op 296	Op 297	Op 298	Op 299	Op 300	Op 301	Op 302	Op 303	Op 304	Op 305	Op 306	Op 307	Op 308	Op 309	Op 310	Op 311	Op 312	Op 313	Op 314	Op 315	Op 316	Op 317	Op 318	Op 319	Op 320	Op 321	Op 322	Op 323	Op 324	Op 325	Op 326	Op 327	Op 328	Op 329	Op 330	Op 331	Op 332	Op 333	Op 334	Op 335	Op 336	Op 337	Op 338	Op 339	Op 340	Op 341	Op 342	Op 343	Op 344	Op 345	Op 346	Op 347	Op 348	Op 349	Op 350	Op 351	Op 352	Op 353	Op 354	Op 355	Op 356	Op 357	Op 358	Op 359	Op 360	Op 361	Op 362	Op 363	Op 364	Op 365	Op 366	Op 367	Op 368	Op 369	Op 370	Op 371	Op 372	Op 373	Op 374	Op 375	Op 376	Op 377	Op 378	Op 379	Op 380</
---------	---------	------	------	------	------	------	------	------	------	------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	----------

110

```

7886 045334 136465 001510 000033      BITF    INTMSK(P4),P.ASCF+1(65) ;CHECK IF ATTENTION RESET
7887 045342 001407                      BEG      155                      ;YES, CONTINUE INTERRUPT PROCESSING
7888 045344 052737 000020 001460      BIS      BF.CLAT,E.COMT ;SET ATTENTION DID NOT RESET
7889                                     ; WITH DRIVE CLEAR
7890 045352 004777 134100                      JSR      PC,QA.COMT ;FLAG ERROR
7891 045356 000137 046214                      JMP      I.RTN      ;RESTORE REGISTERS
7892
7893 045367 013765 001432 000040 155:    MOV      T.WR2,P.A00(R5) ;STORE MAINTENANCE REGISTER 2
7894 045370 032765 040000 000040      BIT      BS.DSC,P.A00(R5) ;CHECK IF DRIVE STATUS CHANGE
7895                                     ; RESET
7896 045376 001404                      BFC      165                      ;YES, CONTINUE INTERRUPT PROCESSING
7897 045400 052765 000040 000014      BIS      BDRVDSC,P.PRST(W5) ;SET DRIVE STATUS CHANGE DID NOT CLEAR
7898 045406 000421                      BP        I.AERR      ;GO PROCESS ERROR
7899
7900 045410 146437 001510 001506 165:    BITF    INTMSK(P4),W.TIME ;RESET TIMING ON THIS DRIVE
7901 045416 005063 001540                      CLR      W.DRV(R3) ;CLEAR DRIVE TIMING COUNT
7902 045422 032765 000004 000014      BIT      BDRVPDT,P.PRST(P5) ;CHECK IF DRIVE IS POSITIONING
7903                                     ; FOR DATA TRANSFER
7904 045430 001004                      BNE      175                      ;YES, CALL COMMAND OPTIMIZER
7905 045432 004777 134014                      JSR      PC,QA.NCMP ;REPORT SUCCESSFUL COMMAND
7906                                     ; COMPLETION
7907 045436 000137 046214                      JMP      I.RTN      ;RESTORE REGISTERS
7908
7909 045442 004737 051740 175:    JSR      PC,C.CPT ;CALL COMMAND OPTIMIZER
7910 045446 000137 046214                      JMP      I.RTN      ;RESTORE REGISTERS
7911
7912                                     .SBTTL  *ATTENTION ERROR HANDLER
7913
7914 045452 042765 000004 000014  I.AERR: BIC      BDRVPDT,P.PRST(P5) ;RESET POSITIONING IN PROGRESS BECAUSE
7915                                     ; OF DATA TRANSFER
7916 045460 146437 001510 001506      BITF    INTMSK(P4),W.TIME ;CLEAR TIMING FOR THIS DRIVE
7917 045466 005063 001540                      CLR      W.DRV(R3) ;RESET WATCH-DOG TIME
7918 045472 042765 177741 000016      BIC      B177741,P.CS1(R5) ;KEEP COMMAND ISSUED
7919 045500 042737 000036 001406      BIC      B16,T.CS1 ;KEEP CURRENT CONTROLLER STATUS
7920 045506 053765 001406 000016      BIS      T.CS1,P.CS1(R5) ;MAKE GOOD MESSAGE
7921 045514 013765 001410 000020      MOV      T.CS2,P.CS2(R5) ;STORE CONTROLLER REGISTERS
7922 045522 013765 001412 000022      MOV      T.WCR,P.WCR(R5)
7923 045530 013765 001414 000024      MOV      T.BA,P.PAP(P5)
7924 045536 013765 001416 000026      MOV      T.CA,P.PYS(R5)
7925 045544 013765 001420 000030      MOV      T.DC,P.DCVL(R5)
7926 045552 013765 001422 000032      MOV      T.ASOF,P.ASOF(W5)
7927 045560 013765 001424 000034      MOV      T.ER,P.ER(R5)
7928 045566 013765 001426 000036      MOV      T.DS,P.DS(R5)
7929 045574 004037 046770                      JSR      RO,I.STAT ;GATHER DRIVE STATUS
7930 045600 046714                      I.RTN      ;PARCE RETURN
7931 045607 112737 000005 001406      MOVW    BDRV.CLRT,T.CS1 ;LOAD COMMAND
7932 045610 004037 046314                      JSR      RO,I.ISSU ;CLEAR DRIVE FENCES
7933 045614 046214                      I.RTN      ;PARCE RETURN
7934 045616 136437 001510 001423      BITF    INTMSK(P4),T.ASCF+1 ;CHECK IF ATTENTION RESET
7935 045624 001407                      BFC      25                      ;YES, FLAG DRIVE PARCE
7936 045626 052737 000020 001460      BIS      BF.CLAT,E.COMT ;SET ATTENTION DID NOT RESET
7937 045634 004777 133616                      JSR      PC,QA.COMT ;REPORT ERROR
7938 045640 000137 046714                      JMP      I.RTN      ;RESTORE REGISTERS
7939
7940 045644 032765 000020 000014 25:    BIT      BDRVHND,P.PRST(W5) ;CHECK IF ANOTHER DRIVE PARCE
7941 045652 001017                      BNE      105                      ;YES, REPORT PARCE

```

```

7942 045654 032737 020000 001432      BIT      #S.PIP,T.WR2      ;CHECK IF DRIVE IS UNLOADING
7943 045662 001413                      BEQ      10$              ;NO, REPORT ERROR
7944 045664 052765 020000 000014      BIS      #E.UNLD,P.PRST(R5) ;SET DRIVE UNLOADING DUE TO PABCP
7945 045672 156437 001510 001506      BISR      INTMSK(R4),W.TIME ;SET TIMING ON THIS DRIVE
7946 045700 013763 001472 001540      MOV      W.8SEC,W.DRV(R3) ;LOAD # SECONDS FOR CYCLE UP
7947 045706 000137 046214                      JMP      I.RTRN              ;RESTORE REGISTERS
7948
7949 045712 004777 133536      10$:      JSR      PC,PA.ABNL      ;REPORT ERROR
7950 045716 000137 046214                      JMP      I.RTRN              ;RESTORE REGISTERS
7951
7952                      .SBTTL      *ERROR CAUSING DRIVE TO UNLOAD
7953
7954 045722 052765 020000 000014      I.UNLD:  BIS      #F.UNLD,P.PRST(R5) ;CLEAR DRIVE UNLOADING BECAUSE OF ENPCF
7955 045730 112737 000005 001406      MOV      #DR.CLR,T.CS1      ;LOAD IN DRIVE CLEAR
7956 045736 004037 046314                      JSR      RO,T.ISSU          ;CC ISSUE DRIVE CLEAR
7957 045742 046214                      I.RTRN              ;ERROR RETURN
7958 045744 136437 001510 001423      BIT      INTMSK(R4),T.ASOF+1 ;CHECK IF ATTENTION CLEARED
7959 045752 001406                      BFC      15$              ;YES, CONTINUE
7960 045754 012737 000020 001460      MOV      #F.CLAT,E.CONT      ;SET ATTENTION DID NOT RESET
7961 045762 004777 133470                      JSR      PC,PA.CONT      ;REPORT ERROR
7962 045766 000512                      BR      I.RTRN              ;RESTORE REGISTERS
7963
7964 045770 032737 040000 001432      15$:      BIT      #S.DSC,T.WR2      ;CHECK IF DRIVE STATUS CHANGE RESET
7965 045776 001403                      BEQ      20$              ;YES, CONTINUE
7966 046000 052765 000040 000014      BIS      #DRVDSC,P.PRST(R5) ;SET DRIVE STATUS CHANGE DID NOT CLEAR
7967 046006 146437 001510 001506      20$:      BISR      INTMSK(R4),W.TIME ;RESET TIMING ON THIS DRIVE
7968 046014 005063 001540                      CLR      W.DRV(R3)          ;CLEAR TIME COUNT
7969 046020 004777 133.30                      JSR      PC,PA.ABNL      ;REPORT ERROR
7970 046024 000473                      BR      I.RTRN              ;RESTORE REGISTERS
7971
7972                      .SBTTL      *DUAL ACCESS INTERRUPT HANDLER
7973
7974 046026 112737 000001 001406      I.DUAL:  MOV      #DR.SEL,T.CS1      ;LOAD COMMAND
7975 046034 004037 046314                      JSR      RO,T.ISSU          ;SELECT DRIVE
7976 046040 046214                      I.RTRN              ;ERROR RETURN
7977 046042 032737 000200 001434      BIT      #S.FLT,T.WR3      ;CHECK IF DRIVE FAULT
7978 046050 001402                      BFC      10$              ;NO, PROCESS INTERRUPT
7979 046052 000137 045457                      JMP      I.AERP            ;PROCESS ATTENTION ERROR
7980
7981 046056 032737 040000 001432      10$:      BIT      #S.DSC,T.WR2      ;CHECK IF DRIVE STATUS CHANGE SET
7982 046064 001005                      BNE      11$              ;YES, PROCESS INTERRUPT
7983 046066 052765 004000 000014      BIS      #MODSC,P.PRST(R5) ;SET NO DRIVE STATUS CHANGE
7984 046074 000137 045452                      JMP      I.AERP            ;PROCESS ATTENTION ERROR
7985
7986 046100 042765 010000 000014      11$:      BIT      #DKVSZD,P.PRST(R5) ;RESET DRIVE SPIED
7987 046106 112737 000005 001406      MOV      #DR.CLR,T.CS1      ;LOAD COMMAND
7988 046114 004037 046314                      JSR      RO,T.ISSU          ;CLEAR DRIVE CSC
7989 046120 046214                      I.RTRN              ;ERROR RETURN
7990 046122 136437 001510 001423      BIT      INTMSK(R4),T.ASOF+1 ;CHECK IF ATTENTION RESET
7991 046130 001406                      BFC      15$              ;YES, CONTINUE
7992 046132 052737 000020 001460      BIS      #F.CLAT,E.CONT      ;SET ATTENTION DID NOT RESET
7993 046140 004777 133312                      JSR      PC,PA.CONT      ;INDICATE CONTROLLER ERROR
7994 046144 000423                      BR      I.RTRN              ;RESTORE REGISTERS
7995
7996 046146 032737 040000 001432      15$:      BIT      #S.DSC,T.WR2      ;CHECK IF DRIVE STATUS CHANGE IS STILL SET
7997 046154 001405                      BFC      20$              ;NO, GO ENQUEUE COMMAND

```

7999	046156	052765	000040	000014		BIS	DDRVDSC,P.PRST(R5)	;SPT DRIVE STATUS CHANGE DID NOT CLEAR
7999	046164	000137	045457			JMP	I.AERR	;PERCPT ATTENTION ERROR
8000								
8001	046170	146437	001510	001506	20\$:	BICE	INTMSK(R4),W.TIME	;STOP TIMING ON THIS DRIVE
8002	046176	005063	001540			CLR	W.DRV(R3)	;CLEAR WATCH DOG TIME OF THIS DRIVE
8003	046202	004037	050706			JSR	RO,Q.PUSH	;PUT PARAMETER BLOCK ADDRESS ON
8004	046206	001324				CYMITO		; COMMAND INITIATION QUEUE
8005	046210	004737	051240			JSR	PC,C.OPT	;CALL COMMAND OPTIMIZER
8006								
8007	046214	012600				I.RTRN: MOV	(SP)+,R0	;RESTORE R0
8008	046216	012601				MOV	(SP)+,R1	;RESTORE R1
8009	046220	012602				MOV	(SP)+,R2	;RESTORE R2
8010	046222	012603				MOV	(SP)+,R3	;RESTORE R3
8011	046224	012604				MOV	(SP)+,R4	;RESTORE R4
8012	046226	012605				MOV	(SP)+,R5	;RESTORE R5
8013	046230	000007				RTI		;RETURN
8014								

```

8015      .SBTTL      *CONTROLLER CLEAR ROUTINE
8016
8017      ;;*****
8018      ;*
8019      ;*      THIS ROUTINE WILL BE USED BY THE DRIVER TO CLEAR THE CONTROLLER
8020      ;*      AND CHECK IF THE CONTROLLER ERRORS ARE RESET. IF THE ERROR IS NOT
8021      ;*      CLEARED, THE ROUTINE AS SPECIFIED IN A.CONT WILL BE CALLED WITH
8022      ;*      E.CCLR SET IN E.CCNT.
8023      ;*
8024      ;*      REGISTER      USE
8025      ;*      -----      ---
8026      ;*
8027      ;*      R2              ADDRESS OF RK06 REGISTERS
8028      ;*      R5              ADDRESS OF PARAMETER BLOCK
8029      ;*
8030      ;*CALL JSR      R0,I.CCLR
8031      ;*      <ADDRESS OF ERROR RETURN>
8032      ;*      RETURN
8033      ;*
8034      ;;*****
8035
8036      046232 012762 100000 000000 I.CCLR: MOV      @CCLR,RFC51(R2) ;CLEAR CONTROLLER
8037      046240 016237 000000 001406      MOV      RFC51(R2),I.C51 ;STORE COMMAND AND STATUS REGISTER 1
8038      046246 032737 100000 001406      BIT      @CERR,I.C51 ;CHECK IF CONTROLLER CLEAR DID
8039      ;*      ; CLEAR ERROR
8040      046254 001407      BEQ      SS ;YES, RETURN TO DRIVER PROCESSING
8041      046256 052737 000001 001460      BIS      @F.CCLR,E.CONT ;SET CLEAR CONTROLLER DID NOT CLEAR ERROR
8042      046264 004777 133166      JSR      PC,RA.CONT ;REPORT CONTROLLER ERROR
8043      046270 011000      MOV      (R6),R0 ;SET UP ERROR RETURN
8044      046272 000200      RTS      R0 ;RETURN
8045
8046      046274 012762 000100 000000 SS:      MOV      @IE,RFC51(R2) ;SET INTERRUPT ENABLE
8047      046302 112737 177777 001402      MOVB     #1,I.I5F1 ;SET INTERRUPT ENABLE ISSUED
8048      046310 005720      TST      (PC)+ ;ADJUST FOR NORMAL RETURN
8049      046312 000200      RTS      R0 ;RETURN

```

```

8050 .SBTTL *COMMAND ISSUED BY DRIVER SERVICE ROUTINE
8051
8052 ;*****
8053 ;*
8054 ;* THIS ROUTINE WILL ISSUE THE COMMAND AS SPECIFIED IN T.CS1
8055 ;* AND CHECK IF A CONTROLLER ERROR OCCURRED. IF A CONTROLLER
8056 ;* ERROR OCCURRED, E.CERR WILL BE SET IN E.CONT AND
8057 ;* CONTROL WILL BE TURN OVER TO THE ROUTINE SPECIFIED BY THE
8058 ;* ADDRESS IN A.CONT.
8059 ;*
8060 ;* REGISTER USE
8061 ;* -----
8062 ;*
8063 ;* R7 ADDRESS OF RK06 REGISTERS
8064 ;* R6 ADDRESS OF PARAMETER BLOCK
8065 ;*
8066 ;*CALL JSR NO,T.ISSU
8067 ;* <ADDRESS OF ERROR RETURN>
8068 ;* RETURN
8069 ;*
8070 ;* ROUTINES USED:
8071 ;* -----
8072 ;*
8073 ;* I.CCLR
8074 ;* I.STOP
8075 ;*
8076 ;*****
8077
8078 046714 013746 001406 I-ISSU: MOV T.CS1, -(SP) ;STORE COMMAND ISSUED
8079 046320 005037 001410 CLR T.CS2 ;CLEAR TEMPORARY CS2
8080 046324 116537 000000 001410 MOVW P.DRVN(P5),T.CS2 ;LOAD IN DRIVE NUMBER
8081 046332 013762 001410 000010 MOV T.CS2,RKCS2(R7) ;LOAD DRIVE NUMBER FOR COMMAND
8082 046340 116537 000007 001407 MOVB P.CS1H(P5),T.CS1+1 ;STORE BITS 8-15 OF CS1
8083 046346 142737 177753 001407 BICB #C<B.CDTIB.CPMT>,T.CS1+1 ;CLEAR ALL BITS EXCEPT
8084 ; FORMAT AND DRIVE TYPE
8085 046354 013762 001406 000000 MOV T.CS1,RKCS1(R7) ;ISSUE COMMAND
8086 046362 105762 000000 1S: TSTB RKCS1(R7) ;WAIT FOR READY
8087 046366 100375 BPL 1S
8088 046770 004737 046536 JSR PC,T.STOR ;GO STORE REGISTERS
8089 046374 032737 100000 001406 BIT #CERR,T.CS1 ;CHECK IF CONTROLLER ERROR OCCURRED
8089 046402 001437 BEQ 5S ;NO, RETURN
8091 046404 032737 001000 001410 BIT #MDS,T.CS2 ;CHECK IF MULTIPLE DRIVE SELECT
8092 046412 001406 BEQ 2S ;NO, CHECK FOR OTHER CONTROLLER ERRORS
8093 046414 052737 100000 001460 BIS #F.MDS,F.CCMT ;SET MULTIPLE DRIVE SELECT FLAG
8094 046422 004777 133030 JSR PC,RA.CONT ;REPORT CONTROLLER ERROR
8095 046426 000440 BR 10S ;RETURN
8096
8097 ;CHECK IF ANY CONTROLLER ERROR IS SET
8098 046430 032737 024000 001406 2S: BIT #CTOISPAR,T.CS1
8099 046436 001027 BNE 7S
8100 046440 032737 176400 001410 BIT #UFFIPGFINEWINECIUPFIMCFIDLT,T.CS2
8101 046446 001023 BNE 7S
8102 046450 032737 131761 001424 BIT #ILCIDTYEIFMTFIECHIPSIFVRCICCEIDTEIOPIIDCK,T.ER
8103 046456 001017 BNE 7S
8104
8105 046460 122716 000005 CMPB #DR.CLR,(SP) ;CHECK IF CLEAR DRIVE

```

8106	046464	001003				BNE	35		;NO, DO NOT SET DRIVE HARD ERROR
8107	046466	052765	000020	000014		BIS	#DRVHDD,P.PRST(R5)		;SET HARD DRIVE ERROR
8108	046474	004037	046232		35:	JSR	RO,I.CCLR		;GC ISSUE A CONTROLLER CLEAR
8109	046500	046530				LOS			;ERROR RETURN
8110	046502	012767	000100	000000	55:	MOV	#IE,RKCS1(R2)		;SET INTERRUPT ENABLE
8111	046510	005726				TST	(SP)+		;ADJUST STACK
8112	046512	005720				TST	(RO)+		;ADJUST RO FOR NORMAL RETURN
8113	046514	000200				RTS	RO		;RETURN
8114									
8115	046516	052737	001000	001460	75:	BIS	#E.CERR,E.CONT		;SET CONTROLLER ERROR DURING
8116									; DRIVER SERVICING
8117	046524	004777	132726			JSR	PC,PA.CONT		;REPORT ERROR
8118	046530	005726			105:	TST	(SP)+		;ADJUST STACK
8119	046532	011000				MOV	(RO),RO		;ADJUST RO FOR ERROR RETURN
8120	046534	000200				RTS	RO		;RETURN

8121
8122
8123
8124
8125
8126
8127
8128
8129
8130
8131
8132
8133
8134
8135
8136
8137
8138 046536 016237 000000 001406
8139 046544 016237 000010 001410
8140 046552 016237 000002 001412
8141 046560 016237 000004 001414
8142 046566 016237 000006 001416
8143 046574 016237 000012 001426
8144 046602 016237 000014 001424
8145 046610 016237 000016 001427
8146 046616 016237 000020 001420
8147 046624 016237 000026 001430
8148 046632 016237 000034 001432
8149 046640 016237 000036 001434
8150 046646 016237 000030 001436
8151 046654 016237 000032 001440
8152 046662 000207

.SBTTL *STORE RK611 UNIPUS REGISTERS

```

;*****
;*
;*      THIS SUBROUTINE IS CALLED BY THE RK06 DRIVER TO STORE ALL
;*      RK611 REGISTER IN TEMPORARY LOCATIONS.
;*
;*CALL   JSR      PC,T.STOR
;*      RETURN
;*
;*      REGISTER      USE
;*      -----      ---
;*
;*      R2              ADDRESS OF RK611 REGISTERS
;*
;*****

```

```

I.STOR: MOV      RKCS1(R2),T.CS1 ;STORE ALL CONTROLLED REGISTERS
        MOV      RKCS2(R2),T.CS2 ; EXCEPT DATA BUFFER
        MOV      RKWC(R2),T.WCR
        MOV      RKBA(R2),T.BA
        MOV      RKDA(R2),T.DA
        MOV      RKDS(R2),T.DS
        MOV      RKER(R2),T.ER
        MOV      RKASOP(P2),T.ASCP
        MOV      RKDCYL(R2),T.DC
        MOV      RKMR1(R2),T.MR1
        MOV      RKMR2(R2),T.MR2
        MOV      RKMR3(R2),T.MR3
        MOV      RKECPS(R2),T.PCS
        MOV      RKECPT(R2),T.PAT
        RTS      PC              ;RETURN

```

```

8153 .SBTTL *STORE CONTROLLER STATUS
8154
8155 ;*****
8156 ;*
8157 ;* THIS SUBROUTINE IS CALLED BY THE RK06 DRIVER AT PRIORITY 7.
8158 ;* THE FOLLOWING REGISTERS WILL BE STORED:
8159 ;*
8160 ;* COMMAND AND STATUS REGISTER 2
8161 ;* WORD COUNT REGISTER
8162 ;* BUS ADDRESS REGISTER
8163 ;* DESIRED TRACK AND SECTOR
8164 ;* STATUS REGISTER
8165 ;* ERROR REGISTER
8166 ;* ATTENTION SUMMARY/OFFSET REGISTER
8167 ;* CYLINDER ADDRESS REGISTER
8168 ;*
8169 ;*CALL JSR PC,I.CSTS
8170 ;* RETURN
8171 ;*
8172 ;* THIS ROUTINE ASSUMES THE FOLLOWING REGISTERS CONTAIN:
8173 ;*
8174 ;*
8175 ;* REGISTER CONTENTS
8176 ;* -----
8177 ;*
8178 ;* R2 RK06 BASE ADDRESS
8179 ;* R5 ADDRESS OF PARAMETER BLOCK
8180 ;*
8181 ;*****
8182
8183 046664 042765 177741 000016 I.CSTS: BIC #177741,P.CS1(R5) ;CLEAR ALL BITS EXCEPT FUNCTION
8184 ;OF LAST COMMAND ISSUED
8185 046672 042737 000036 001406 BIC #36,T.CS1 ;CLEAR FUNCTION OF CS1 STATUS
8186 046700 053765 001406 000016 BIS T.CS1,P.CS1(R5) ;GENERATE CS1 STATUS INFORMATION
8187 046706 016265 000010 000020 I.CST1: MOV RKCS2(R2),P.CS2(R5) ;STORE COMMAND AND STATUS REGISTER 2
8188 046714 016265 000002 000022 MOV RWMC(R2),P.WCR(R5) ;STORE WORD COUNT REGISTER
8189 046722 016265 000004 000024 MOV RWBA(R2),P.BAR(P5) ;STORE BUS ADDRESS REGISTER
8190 046730 016265 000006 000026 MOV RKDA(R2),P.DTS(R5) ;STORE DESIRED TRACK AND SECTOR
8191 046736 016765 000012 000036 MOV RKDS(R2),P.DS(R5) ;STORE DRIVE STATUS REGISTER
8192 046744 016265 000014 000034 MOV RWER(R2),P.EE(R5) ;STORE ERROR REGISTER
8193 046752 016265 000016 000032 MOV RWASOP(R2),P.ASOP(R5) ;STORE ATTENTION SUMMARY AND
8194 ; OFFSET
8195 046760 016765 000020 000030 MOV RKDCYL(R2),P.DCYL(R5) ;STORE CYLINDER ADDRESS
8196 046766 000207 RTS PC ;RETURN
8197

```

```

8199      .SBTTL  *GATHER DRIVE STATUS
8199
8200      ;;*****
8201      ;*
8202      ;*      THIS SUBROUTINE WILL BE USED TO GATHER DRIVE STATUS
8203      ;*      BYTE 01, 10, AND 11. IT IS ASSUMED THAT THE DRIVE
8204      ;*      HAS PREVIOUSLY BEEN SEIZED. IT RUNS AT PRIORITY 7.
8205      ;*
8206      ;*CALL  JSR      RO,I.STAT
8207      ;*      <ADDRESS OF EPROR RETURN>
8208      ;*      RETURN
8209      ;*
8210      ;*      THIS ROUTINE ASSUMES THE FOLLOWING REGISTERS CONTAIN:
8211      ;*
8212      ;*
8213      ;*      REGISTER      CONTENTS
8214      ;*      -----
8215      ;*
8216      ;*      R2      RK06 BASE ADDRESS
8217      ;*      R5      ADDRESS OF PARAMETER BLOCK
8218      ;*
8219      ;*      ROUTINES USED:
8220      ;*      I.ISSU
8221      ;*
8222      ;;*****
8223
8224      046770 012762 000001 000026 I.STAT: MOV      #1,RKMR1(R2)      ;LOAD MAINTENANCE REGISTER 1
8225      ;PCR STATUS BYTE 01
8226      046776 112737 000001 001406      MOVB      #DR.SEL,T.CS1      ;LOAD COMMAND
8227      047004 004037 046314      JSR      RO,I.ISSU      ;GET STATUS BYTES 01
8228      047010 047200      3S      ;ERRCR RETURN
8229      047017 013765 001432 000044      MOV      T.MR2,P.A01(R5)      ;STORE STATUS BYTE 01 MESS A
8230      047020 013765 001434 000046      MOV      T.MP3,P.B01(R5)      ;STORE STATUS BYTE 01 MESS B
8231      047026 012762 000002 000026      MOV      #2,RKMR1(P2)      ;LOAD MAINTENANCE REGISTER 1
8232      ;PCR STATUS BYTE 10
8233      047034 112737 000001 001406      MOVB      #DR.SPL,T.CS1      ;LOAD COMMAND
8234      047042 004037 046314      JSR      RO,I.ISSU      ;GET STATUS BYTES 10
8235      047046 047200      3S      ;ERRCR RETURN
8236      047050 013765 001437 000050      MOV      T.MR2,P.A10(R5)      ;STORE STATUS BYTE 10 MESS A
8237      047056 013765 001434 000052      MOV      T.MR3,P.B10(R5)      ;STORE STATUS BYTE 10 MESS B
8238      047064 012762 000003 000026      MOV      #3,RKMR1(R2)      ;LOAD MAINTENANCE REGISTER
8239      ;PCR STATUS BYTE 11
8240      047072 112737 000001 001406      MOVB      #DR.SEL,T.CS1      ;LOAD COMMAND
8241      047100 004037 046314      JSR      RO,I.ISSU      ;GET STATUS BYTES 11
8242      047104 047200      3S      ;ERRCR RETURN
8243      047106 013765 001432 000054      MOV      T.MR2,P.A11(R5)      ;STORE STATUS BYTE 11 MESS A
8244      047114 013765 001434 000056      MOV      T.MP3,P.B11(R5)      ;STORE STATUS BYTE 11 MESS B
8245      047122 005062 000026      CLR      RKMR1(R?)      ;LOAD MAINTENANCE REGISTER 1
8246      ;PCR STATUS BYTE 00
8247      047126 112737 000001 001406      MOVB      #DR.SEL,T.CS1      ;LOAD COMMAND
8248      047134 004037 046314      JSR      RO,I.ISSU      ;GET STATUS BYTES 00
8249      047140 047200      3S      ;ERRCR RETURN
8250      047142 013765 001432 000040      MOV      T.MR2,P.A00(R5)      ;STORE STATUS BYTE 00 MESS A
8251      047150 013765 001434 000042      MOV      T.MP3,P.B00(R5)      ;STORE STATUS BYTE 00 MESS B
8252      047156 032737 001000 001434      BIT      #S.PAR,T.MR3      ;CHECK IF BAD PARITY DETECTED BY DRIVE
8253      047164 001407      BFG      SS      ;NO, RETURN NORMALLY

```

8254	047166	052737	002000	001460		BIS	#E.DPAR,E.CONT	;INDICATE BAD PARITY DETECTED BY DRIVE
8255	047174	004777	132256			JSR	PC,PA.CONT	;REPORT ERROR
8256	047200	011000			3S:	MOV	(R0),R0	;LOAD R0 FOR ERROR RETURN
8257	047202	000200				RTS	R0	;RETURN
8258								
8259	047204	052765	001000	000014	5S:	BIS	#PBSVAL,P.PRST(R5)	;SET PARAMETER BLOCK STATUS VALID
8260	047212	005720				TST	(R0)+	;ADJUST EC FOR NORMAL RETURN
8261	047214	000200				RTS	R0	;RETURN
8262								

.SBTTL *COMMAND INITATOR

```

;*****
;*
;* THIS SUBROUTINE WILL INITIATE ALL COMMANDS AS SPECIFIED
;* BY THE COMMAND FIELD OF THE PARAMETER BLOCK. THE FOLLOWING
;* SPECIAL COMMAND ARE ALSO EXECUTED:
;*
;* RELEASE
;* CONTROLLER CLEAR
;* SUBSYSTEM CLEAR
;* READ ALL DRIVE STATUS
;* READ SPECIFIED HEADER
;*
;* THE ABOVE COMMANDS ARE TRANSLATED INTO A SEQUENCE OF COMMANDS
;*
;*CALL JSR PC,C.INIT
;* <ADDRESS OF PARAMETER BLOCK>
;* RETURN
;*
;* FOR THE SEQUENTIAL OPERATIONS, THE CPUFFS WILL LOAD THE
;* LOCATIONS, PBLKT AND INTMSK.
;*
;* ROUTINES USED:
;* M.WTCH
;* I.CSTS
;* I.STAT
;* I.CCLR
;*****

```

8294	047216	010546		
8295	047220	010446		
8296	047222	010346		
8297	047224	010246		
8298	047226	010146		
8299	047230	010046		
8300	047232	013746	177776	
8301	047236	013737	001450	177776
8302	047244	017605	000016	
8303	047250	062766	000002	000016
8304	047256	016504	000000	
8305	047262	042704	177770	
8306	047266	156437	001510	001506
8307	047274	010403		
8308	047276	006303		
8309	047300	013763	001470	001540
8310				
8311	047306	013702	001444	
8312				
8313				
8314				
8315				
8316				
8317				
8318				

```

C.INIT: MOV R5, -(SP) ;STORE R5 ON STACK
        MOV R4, -(SP) ;STORE R4 ON STACK
        MOV R3, -(SP) ;STORE R3 ON STACK
        MOV R2, -(SP) ;STORE R2 ON STACK
        MOV R1, -(SP) ;STORE R1 ON STACK
        MOV R0, -(SP) ;STORE R0 ON STACK
        MOV PS, -(SP) ;STORE PSW ON STACK
        MOV RKPRI, PS ;LOCK OUT RK06 INTERRUPTS
        MOV @16(SP), R5 ;STORE PARAMETER BLOCK ADDRESS
        ADD @2, 16(SP) ;ADJUST RETURN
        MOV P.DRVN(R5), P4 ;STORE DRIVE NUMBER
        BTC #C<DRVMASK>, R4 ;MASK OUT JUNK
        BTSP INTMSK(P4), W.TIME ;SET WATCH DOG TIME
        MOV R4, R3 ;STORE DRIVE NUMBER
        ASL R3 ;MULTIPLY DRIVE NUMBER BY 2
        MOV W.SEC, W.DRV(R3) ;LOAD WATCH-DOG TIME

        MOV RFBAS, R7 ;LOAD R2 WITH RK06 ADDRESS BASE

;
; RFSFT ALL BITS IN PROGRAM DEVICE STATUS REGISTER EXCEPT
; DRIVE IN USE
; WRITE FOR WRITE CHECK
; NO CHECK
; DROP DRIVE FROM TEST SEQUENCE
; INHIBIT BUS ADDRESS INCREMENT

```

8319	047312	042765	075176	000014		BIC	#C<DRVPOS, WCKINCHKIDRDPDRVIDTBAIL>, P.PRST(P5)
8320							
8321	047320	010500				MOV	R5, R0 ;STORE PARAMETER BLOCK ADDRESS
8322	047322	062700	000016			ADD	#P.CS1, R0 ;CALCULATE FIRST LOCATION TO BE CLEARED
8323	047326	010501				MOV	R5, P1 ;STORE PARAMETER BLOCK ADDRESS
8324	047330	062701	000062			ADD	#P.PPAT, R1 ;CALCULATE LAST LOCATION TO BE CLEARED
8325							
8326	047334	005020			1\$:	CLR	(R0)+ ;CLEAR RETURN PARAMETER
8327	047336	020001				CMP	R0, R1 ;CHECK IF FINISHED
8328	047340	101775				BLOS	1\$;NO, CLEAR NEXT RETURN PARAMETER
8329	047342	105037	001502			CLRB	I.ISRL ;CLEAR RELEASE OR INTERRUPT ISSUEC
8330	047346	010465	000020			MOV	R4, P.CS2(R5) ;STORE DRIVE NUMBER
8331	047352	005062	000026			CLR	RMR1(R2) ;CLEAR PK06 MAINTENANCE REGISTER 1
8332	047356	132765	000040	000001		BITB	#R15, P.CMND(R5) ;CHECK IF SPECIAL COMMAND
8333	047364	001402				BEQ	3\$;NC, PACCPS
8334	047366	000137	047746			JMP	C.SPEC ;JUMP TO SPECIAL COMMAND PACCPS
8335							
8336	047372	122765	000107	000001	3\$:	CMPB	#UNLOAD, P.CMND(R5) ;CHECK IF POSITIONING COMMAND
8337							; START SPINDLE
8338							; RECALIBRATE
8339							; OFFSET
8340							; SEEK
8341							; UNLOAD
8342							
8343	047400	101125				BRI	25\$;NO, DRIVE COMMAND
8344							; SELECT DRIVE
8345							; PACK ACKNOWLEDGE
8346							; CLEAR
8347							
8348	047402	122765	000117	000001		CMPB	#SEEK, P.CMND(R5) ;CHECK IF DATA TRANSFER
8349	047410	103471				BLO	20\$;YES, DATA TRANSFER COMMAND
8350							; READ DATA
8351							; WRITE DATA
8352							; READ HEADER
8353							; WRITE HEADER
8354							; WRITE CHECK
8355	047412	016562	000020	000010		MOV	P.CS2(R5), RKC2(R2) ;LOAD DRIVE NUMBER
8356	047420	052765	000007	000014		BIS	#DRVPOS, P.PRST(R5) ;SET DRIVE POSITIONING
8357	047426	005037	001462			CLH	0.WAIT ;CLEAR WAIT FOR COMMAND
8358	047432	122765	000117	000001		CMPB	#SEEK, P.CMND(R5) ;CHECK IF SEEK
8359	047440	001007				BNE	5\$;NO, CHECK FOR OFFSET
8360	047442	016562	000002	000020		MOV	P.CYLW(R5), PKDCYL(R2) ;LOAD CYLINDER ADDRESS
8361	047450	016562	000004	000006		MOV	P.SECT(R5), PKDA(R2) ;LOAD SECTOR AND TRACK
8362	047456	000431				BR	8\$;GO ISSUE COMMAND
8363							
8364	047460	122765	000115	000001	5\$:	CMPB	#OFFSET, P.CMND(R5) ;CHECK IF OFFSET
8365	047466	001007				BNE	6\$;NC, CHECK FOR UNLOAD
8366	047470	116565	000006	000032		MOV	P.OFST(P5), P.ASOF(R5) ;STORE OFFSET
8367	047476	016562	000032	000016		MOV	P.ASOF(P5), PKASOF(R2) ;LOAD OFFSET REGISTER
8368	047504	000416				BR	8\$;GO ISSUE COMMAND
8369							
8370	047506	122765	000111	000001	6\$:	CMPB	#RTSPL, P.CMND(R5) ;CHECK IF START SPINDLE
8371	047514	001003				BNE	7\$;NO, CHECK IF WFCAL
8372	047516	013763	001474	001540		MOV	W.WIN, W.DRV(R3) ;LOAD WATCH LOG TIME FOR 1 MINUTE
8373	047524	122765	000113	000001	7\$:	CMPB	#RECAL, P.CMND(R5) ;CHECK IF RECAL
8374	047532	001003				BNE	8\$;NO, CONTINUE

8375	047534	013763	001472	001540		MOV	W.8SEC,W.DRV(R3) ;LOAD RECAL TIME TO 8 SECONDS
8376	047542	116565	000007	000017	85:	MOV	P.CS1H(R5),P.CS1+1(R5) ;STORE BITS P-15 OF CS1
8377	047550	042765	165777	000016		BIC	#C<CFMTICDT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT PCFMT
8378							; AND DRIVE TYPE
8379	047556	116565	000011	000016		MOV	P.CMND(R5),P.CS1(R5) ;MOVE COMMAND INTO CS1
8380	047564	042765	000200	000014		BIC	#W.WCF,P.PRST(R5) ;RESET WRITE PCP WRITE CHECK
8381	047572	000460				BR	305 ;GO, ISSUE COMMAND
8382							
8383	047574	016562	000010	000004	205:	MOV	P.BALR(P5),PKPA(R2) ;LOAD BUS ADDRESS REGISTER
8384	047602	016562	000017	000002		MOV	P.WC(P5),PKWC(R2) ;LOAD WORD COUNT REGISTER
8385	047610	016562	000002	000020		MOV	P.CYLH(R5),PKCYL(R2) ;LOAD CYLINDER ADDRESS REGISTER
8386	047616	016562	000004	000006		MOV	P.SECT(P5),PKDA(R2) ;LOAD SECTOR AND TRACK NUMBERS
8387	047624	122765	000131	000001		CMPR	#WRTCHK,P.CMND(R5) ;CHECK IF WRITE CHECK COMMAND
8388	047632	001010				BNE	255 ;NO, GO ISSUE THE COMMAND
8389	047634	032765	000700	000014		BIT	#W.WCF,P.PRST(R5) ;CHECK IF WRITE COMMAND SHOULD BE ISSUED
8390	047642	001404				BFG	255 ;NO, GO ISSUE THE COMMAND
8391	047644	012765	000123	000016		MOV	#WRTATA,P.CS1(R5) ;ISSUE WRITE COMMAND
8392	047657	000406				BR	265 ;GO ISSUE COMMAND
8393							
8394	047654	116565	000001	000016	255:	MOV	P.CMND(R5),P.CS1(R5) ;MOVE COMMAND INTO CS1
8395	047662	042765	000200	000014		BIC	#W.WCF,P.PRST(R5) ;RESET WRITE PCP WRITE CHECK
8396	047670	116565	000007	000017	265:	MOV	P.CS1H(R5),P.CS1+1(R5) ;STORE BITS P-15 OF CS1
8397	047676	142765	177750	000017		BICR	#C<B.CFMTIP.CDTIP.PA16IB.BA17>,P.CS1+1(R5) ;CLEAR ALL BITS EXCEPT
8398							; PCFMT, DRIVE TYPE, AND BUS ADDRESS
8399							; BITS 16-17
8400	047704	010537	001467			MOV	R5,0.WAIT ;LOAD WAITING FOR COMMAND
8401	047710	032765	100000	000014		BIT	#DTRAIL,P.PRST(P5) ;CHECK IF INHIBIT BUS ADDRESS INCREMENT
8402	047716	001403				BFG	275 ;NO, LOAD CS2
8403	047720	052765	000020	000020		BIS	#BAI,P.CS2(R5) ;SET INHIBIT BUS ADDRESS INCREMENT
8404	047726	016562	000020	000010	275:	MOV	P.CS2(R5),R5CS2(R2) ;LOAD CS2
8405	047734	016562	000016	000000	305:	MOV	P.CS1(R5),R5CS1(R2) ;ISSUE COMMAND
8406	047742	000137	050664			JMP	C.WTRN ;RESTORE REGISTERS
8407							
8408							
8409							
8410	047746	172765	000141	000001	C.SPEC:	CMPR	#PDSTAT,P.CMND(R5) ;CHECK IF READ DRIVE STATUS
8411	047754	001132				BNE	105 ;NO, PROCESS OTHER COMMANDS
8412	047756	016562	000020	000010		MOV	P.CS2(R5),R5CS2(R2) ;LOAD CS2 FOR COMMAND
8413	047764	116565	000007	000017		MOV	P.CS1H(R5),P.CS1+1(R5) ;STORE BITS P-15 OF CS1
8414	047772	042765	165777	000016		BIC	#C<CFMTICDT>,P.CS1(R5) ;CLEAR ALL BITS EXCEPT PCFMT
8415							; AND DRIVE TYPE
8416	050000	112765	000001	000016		MOV	#DN.SFL,P.CS1(R5) ;STORE COMMAND
8417	050006	016562	000016	000000		MOV	P.CS1(R5),R5CS1(R2) ;ISSUE COMMAND
8418	050014	004737	043314		25:	JSR	PC,W.WTCH ;CALL WATCH-DOG TIMER
8419	050020	016265	000000	000016		MOV	R5CS1(R2),P.CS1(R5) ;STORE COMMAND AND STATUS REG. 1
8420	050026	032765	000700	000016		BIT	#PDY,P.CS1(R5) ;WAIT FOR READY
8421	050034	001767				BFG	25
8422	050036	004737	046706			JSR	PC,I.CST1 ;STORE CONTROLLER REGISTERS
8423	050042	016265	000034	000040		MOV	R5MP2(R2),P.A00(R5) ;STORE STATUS BYTE 00 MESSAGE A
8424	050050	016265	000036	000047		MOV	R5MP3(R2),P.B00(R5) ;STORE STATUS BYTE 00 MESSAGE B
8425	050056	032765	100000	000016		BIT	#CERK,P.CS1(R5) ;CHECK IF CONTROLLER ERROR
8426	050064	001443				BFG	65 ;NO, GATHER DRIVE STATUS
8427	050066	146437	001510	001506		BITR	INTMSK(R4),W.TIME ;RESET WATCH DOG TIMING ON THIS DRIVE
8428	050074	005063	001540			CLR	W.DRV(R3) ;CLEAR WATCH DOG COUNT
8429	050100	032765	001000	000020		BIT	#MDS,P.CS2(R5) ;CHECK IF MULTIPLE DRIVE SELECT
8430	050106	001046				BNE	95 ;YES, INDICATE CONTROLLER ERROR

.SBTTL *SPECIAL COMMAND PROCESSING

8431	050110	004037	046732			JSR	RO,I.CCLR	;CLEAR ERROR
8432	050114	050664				C.RTRN		;ERROR RETURN
8433	050116	032765	010400	000020		BIT	#MEDIUF,P.CS2(R5)	;CHECK IF NON-EXISTENT DRIVE OR UNIT FIELD ERROR
8434	050124	001017				BNE	SS	;REPORT ERROR
8435	050126	032765	000001	000036		BIT	#DRA,P.DS(R5)	;CHECK IF DRIVE AVAILABLE
8436	050134	001013				BNE	SS	;YES, REPORT ERROR
8437	050136	156437	001510	001506		BTRN	INTMSK(R4),W.TIME	;SET TIMING FOR THIS DRIVE
8438	050144	013763	001474	001540		MOV	W.MIN,W.DRV(R3)	;WAIT 1 MINUTE FOR RELEASE
8439	050152	052765	010000	000014		BIS	#DRVSZD,P.PRST(R5)	;SET DRIVE SEIZED
8440	050160	000137	050664			JMP	C.RTRN	;RESTORE REGISTERS
8441								
8442	050164	004777	131264		55:	JSR	PC,RA.ABNL	;REPORT ERROR
8443	050170	000137	050664			JMP	C.RTRN	;RESTORE REGISTERS
8444								
8445	050174	004037	046770		65:	JSR	RO,I.STAT	;GATHER DRIVE STATUS
8446	050200	050664				C.RTRN		;ERROR RETURN
8447	050202	146437	001510	001506		BTRN	INTMSK(R4),W.TIME	;STOP WATCH-DOG TIMING ON DRIVE
8448	050210	005063	001540			CLR	W.DRV(R3)	;RESET WATCH-DOG TIME
8449	050214	004777	131232			JSR	PC,RA.NORM	;REPORT COMMAND COMPLETE
8450	050220	000137	050664			JMP	C.RTRN	;RESTORE REGISTERS
8451								
8452	050224	052737	100000	0015	95:	BIS	#E.MDS,F.CONT	;SET MULTIPLE DRIVE SELECT
8453	050232	004777	131220			JSR	PC,RA.COMT	;INDICATE CONTROLLER ERROR
8454	050236	000137	050664			JMP	C.RTRN	
8455								
8456	050242	122765	000140	000001	105:	CMPP	#PELEAS,P.CMND(R5)	;CHECK IF RELEASE COMMAND
8457	050250	001031				BNE	135	;NO, CHECK IF READ ALL HEADERS
8458	050252	010537	001462			MOV	R5,O.WAIT	;STORE PARAMETER *LOCK ADDRESS IN
8459								; WAIT FOR COMMAND
8460	050256	052765	000010	000020		BIS	#RLS,P.CS2(P5)	;SET RELEASE PIT
8461	050264	016562	000020	000010		MOV	P.CS2(R5),RRC52(R2)	;LOAD CS2 FOR DESELECT
8462	050272	112737	000001	001507		MOV	#1,I.ISPL	;SET FLAG FOR RELEASE COMMAND
8463	050300	116565	000007	000017		MOV	P.CS1H(P5),P.CS1+1(R5)	;STORE BITS 8-15 OF CS1
8464	050306	042765	165777	000016		BIC	#C<CPMTICD>,P.CS1(R5)	;CLEAR ALL BITS EXCEPT PCFMT
8465								; AND DRIVE TYPE
8466	050314	112765	000101	000016		MOV	#SELDPR,P.CS1(R5)	;STORE COMMAND
8467	050322	016562	000016	000000	115:	MOV	P.CS1(R5),RRC51(R2)	;ISSUE COMMAND
8468	050330	000137	050664			JMP	C.RTRN	;RESTORE REGISTERS
8469								
8470	050334	122765	000164	000001	135:	CMPP	#PDALHD,P.CMND(R5)	;CHECK IF READ ALL HEADERS
8471	050342	001047				BNE	305	;NO, CHECK IF CONTROLLER CLEAR
8472	050344	010537	001462			MOV	R5,O.WAIT	;SET WAITING FOR COMMAND COMPLETION
8473	050350	016537	000010	001476		MOV	P.BALO(P5),HDR.AD	;LOAD HEADER ADDRESS
8474	050356	132765	000020	000007		BIT	#R.CFMT,P.CS1H(R5)	;CHECK IF 22 SECTOR FORMAT
8475	050364	001404				BFG	145	;YES, LOAD 22 IN HEADER COUNT
8476	050366	012737	000024	001500		MOV	#20.,HDR.CT	;LOAD 20 IN SECTOR COUNT
8477	050374	000407				BP	225	;GO ISSUE READ HEADER COMMAND
8478								
8479	050376	012737	000026	001500	145:	MOV	#22.,HDR.CT	;LOAD 22 IN SECTOR COUNT
8480	050404	016562	000002	000020	225:	MOV	P.CYLH(R5),RKDCYL(R2)	;LOAD CYLINDER ADDRESS
8481	050412	016562	000004	000006		MOV	P.SECT(R5),RKDA(R2)	;LOAD TRACK NUMBER
8482	050420	016562	000020	000010		MOV	P.CS2(R5),RRC52(R2)	;LOAD DRIVE NUMBER
8483	050426	116565	000007	000017		MOV	P.CS1H(P5),P.CS1+1(R5)	;STORE BITS 8-15 OF CS1
8484	050434	042765	165777	000016		BIC	#C<CPMTICD>,P.CS1(R5)	;CLEAR ALL BITS EXCEPT DRIVE TYPE
8485								; AND FORMAT
8486	050442	112765	000125	000016		MOV	#RDHEAD,P.CS1(R5)	;STORE READ HEADER COMMAND

8487	050450	016567	000016	000000		MOV	P.CS1(R5),RKCS1(R7)	;ISSUE READ HEADER
8488	050456	000137	050664			JMP	C.RTRN	;RESTORE REGISTERS
8489								
8490	050462	122765	000176	000001	30\$:	CMPF	#CONCLR,P.CMND(R5)	;CHECK IF CONTROLLER CLEAR
8491	050470	001004				BNE	37\$	;NO, CHECK IF SUBSYSTEM CLEAR
8492	050472	004037	046232			JSR	FO,I.CCLR	;CLEAR CONTROLLER
8493	050476	050664				C.RTRN		;REPORT ERROR
8494	050500	000467				BR	40\$	;INDICATE NORMAL RETURN
8495								
8496	050502	122765	000177	000001	32\$:	CMPF	#SUPCLR,P.CMND(R5)	;CHECK IF SUBSYSTEM CLEAR
8497	050510	001406				BEG	36\$	;YES, CLEAR SUBSYSTEM
8498	050512	052737	000100	001460	34\$:	BIS	#F.IILD,E.COMT	;SET ILLEGAL DRIVER COMMAND
8499	050520	004777	130737			JSR	PC,RA.COMT	;REPORT ERROR
8500	050524	000457				BR	C.RTRN	;RESTORE REGISTERS
8501								
8502	050526	012762	000040	000010	36\$:	MOV	#SCLR,RKCS2(R2)	;ISSUE SUBSYSTEM CLEAR
8503	050534	016765	000000	000016		MOV	RKCS1(R7),P.CS1(R5)	;STORE COMMAND AND STATUS REGISTER 1
8504	050542	032765	100000	000016		BIT	#CEFR,P.CS1(R5)	;CLEAR IF CONTROLLER ERROR RESET
8505	050550	001406				BFG	37\$	;NO, FINISH COMMAND
8506	050552	052737	000001	001460		BIS	#BIT0,E.COMT	;SET CLEAR SUBSYSTEM DID NOT CLEAR
8507								; CONTROLLER ERROR
8508	050560	004777	130672			JSR	PC,RA.COMT	;REPORT ERROR
8509	050564	000437				BR	C.RTRN	;RESTORE REGISTERS
8510								
8511	050566	013746	001466		37\$:	MOV	W.MILT,-(SP)	;LOCAL 16 MILLI-SECOND COUNT FOR ATTENTION
8512								; TC DISAPPEAR
8513	050572	016765	000000	000016	38\$:	MOV	RKCS1(R7),P.CS1(R5)	;STORE CS1
8514	050600	032765	040000	000016		BIT	#DI,P.CS1(R5)	;CHECK IF ATTENTIONS CLEARED
8515	050606	001411				BEG	39\$	;YES, FINISH COMMAND
8516	050610	005716				DEC	(SP)	;DECREMENT 16 MILLI-SECOND COUNT
8517	050612	001367				BNE	38\$	;CHECK DRIVE INTERRUPT AGAIN
8518	050614	005726				TST	(SP)+	;ADJUST STACK
8519	050616	052737	000040	001460		BIS	#F.SCLR,E.COMT	;SET SUBSYSTEM CLEAR DID NOT CLEAR
8520								; DRIVE ATTENTIONS
8521	050624	004777	130626			JSR	PC,RA.COMT	;REPORT ERROR
8522	050630	000415				BR	C.RTRN	;RESTORE REGISTER
8523								
8524	050637	005726			39\$:	TST	(SP)+	;ADJUST STACK
8525	050634	032765	000400	000014		BIT	#NCCH,P.PST(R5)	;CHECK IF AC CHECK MODE
8526	050642	001010				BNE	C.RTRN	;YES, RESTORE REGISTERS
8527	050644	112737	177777	001502		MOVF	#-1,I.ISKL	;SET INTERRUPT ENABLE SET
8528	050652	012762	000100	000000		MOV	#IE,RKCS1(R2)	;SET INTERRUPT ENABLE
8529	050660	004777	130566		40\$:	JSR	PC,RA.NCFM	;INDICATE NORMAL TERMINATION
8530								
8531	050664	012637	177776			C.RTRN: MOV	(SP)+,PS	;RESTORE PSW
8532	050670	012600				MOV	(SP)+,R0	;RESTORE R0
8533	050672	012601				MOV	(SP)+,R1	;RESTORE R1
8534	050674	012602				MOV	(SP)+,R2	;RESTORE R2
8535	050676	012603				MOV	(SP)+,R3	;RESTORE R3
8536	050700	012604				MOV	(SP)+,R4	;RESTORE R4
8537	050702	012605				MOV	(SP)+,R5	;RESTORE R5
8538	050704	000207				RTS	PC	;RETURN

```

0539      .SBTTI      *ENQUEUE ROUTINE
0540
0541      ;;*****
0542      ;*
0543      ;*      THIS ROUTINE WILL PUT THE ADDRESS OF THE PARAMETER IN THE
0544      ;*      QUEUE SPECIFIED BY THE LOCATION FOLLOWING THE CALL.
0545      ;*
0546      ;*CALL      JSR      17,Q.PUSH
0547      ;*      <ADDRESS OF QUEUE>
0548      ;*      RETURN
0549      ;*
0550      ;;*****
0551
0552      050706 010446      Q.PUSH: MOV      R4,-(SP)      ;STORE R4 ON STACK
0553      050710 010746      MOV      R3,-(SP)      ;STORE R3 ON STACK
0554      050712 013746 177776      MOV      PS,-(SP)      ;STORE PSW ON STACK
0555      050716 013737 001450 177776      MOV      RFPRI,PS      ;LOCK OUT RK06 INTERRUPTS
0556      050724 012004      MOV      (R0)+,R4      ;STORE QUEUE ADDRESS
0557      050726 005064 000064      CLR      P.QLNK(R5)      ;CLEAR QUEUE LINK
0558      050732 005724      TST      (R4)+      ;CHECK IF QUEUE IS EMPTY
0559      050734 001405      BFG      1$      ;YES, PUT PARAMETER BLOCK ADDRESS
0560      ; AT HEAD AND TAIL OF THE QUEUE
0561      050736 011403      MOV      (R4),R3      ;STORE TAIL OF QUEUE
0562      050740 010563 000064      MOV      R5,P.QLNK(R3)      ;UPDATE QUEUE LINK
0563      050744 010514      MOV      R5,(R4)      ;UPDATE QUEUE TAIL
0564      050746 000402      BR      2$
0565
0566      050750 010514      1$:      MOV      R5,(R4)      ;LOAD TAIL OF QUEUE
0567      050752 010544      MOV      R5,-(R4)      ;LOAD HEAD OF QUEUE
0568
0569      050754 012637 177776      2$:      MOV      (SP)+,PS      ;RESTORE PSW
0570      050760 012603      MOV      (SP)+,R3      ;RESTORE R3
0571      050762 012604      MOV      (SP)+,R4      ;RESTORE R4
0572      050764 000200      RTS      R0      ;RETURN

```

```

0573 .SBTTL *POP QUEUE ROUTINE
0574
0575 ;*****
0576 ;*
0577 ;* THIS ROUTINE WILL PUT THE ADDRESS OF THE FIRST PARAMTER
0578 ;* BLOCK ADDRESS OF THIS QUEUE ON THE STACK AND REMOVE IT
0579 ;* FROM THE QUEUE DESIGNATED BY THE CALL. A RETURN OF ZERO
0580 ;* ON THE STACK INDICATES THAT THE QUEUE WAS EMPTY. OTHERWISE,
0581 ;* THE STACK WILL CONTAIN THE ADDRESS OF THE FIRST
0582 ;* PARAMETER BLOCK ON THE QUEUE.
0583 ;*
0584 ;*CALL JSR R0,Q.POP
0585 ;* <ADDRESS OF QUEUE>
0586 ;* RETURN
0587 ;*
0588 ;*****
0589
0590 Q.POP: MOV (SP),-(SP) ;CREATE SPACE ON THE STACK
0591 MOV R5, -(SP) ;STORE R5 ON THE STACK
0592 MOV R4, -(SP) ;STORE R4 ON THE STACK
0593 MOV PS, -(SP) ;STORE PSW ON THE STACK
0594 MOV RKPRI, PS ;LOCK OUT RK06 INTERRUPTS
0595 MOV (R0)+, R4 ;STORE QUEUE ADDRESS
0596 MOV (R4), R5 ;STORE PARAMETER BLOCK ADDRESS
0597 MOV R5, 10(SP) ;LOAD ADDRESS ON STACK
0598 BFG 15 ;IF QUEUE IS EMPTY, RETURN
0599 MOV P.LINK(P5), (R4)+ ;READJUST HEAD OF THE QUEUE
0600 BNE 15 ;IF NOT LAST ELEMENT OF THE QUEUE, RETURN
0601 CLR (R4) ;CLEAR TAIL POINTER IF LAST ELEMENT OF QUEUE
0602
0603 15: MOV (SP)+, PS ;RESTORE PSW
0604 MOV (SP)+, R4 ;RESTORE R4
0605 MOV (SP)+, R5 ;RESTORE R5
0606 RTS R0 ;RETURN

```

```

8607      .SBTTL  *SEARCH QUEUE FOR DRIVE NOT POSITIONING
8608
8609      ;;*****
8610      ;*
8611      ;*      THIS ROUTINE WILL SEARCH THROUGH THE COMMAND INITIATION
8612      ;*      QUEUE FOR THE FIRST PARAMETER BLOCK WHOSE DRIVE IS
8613      ;*      NOT POSITIONING.  IF THE QUEUE IS EMPTY OF ALL DRIVES
8614      ;*      ARE POSITIONING ZERO IS PLACED IN R5.  OTHERWISE,
8615      ;*      THE ADDRESS OF THE PARAMETER BLOCK IS PLACED IN R5.
8616      ;*
8617      ;*CALL JSR      PC,Q.SRCH
8618      ;*      RETURN
8619      ;*
8620      ;*NOTE: THIS ROUTINE DESTROYS R4.
8621      ;*
8622      ;;*****
8623
8624      051042  012704  001324      Q.SRCH: MOV      #CINITQ,R4      ;LOAD ADDRESS OF COMMAND INITIATION
8625      ;*      ; QUEUE IN R4
8626      051046  011405      1$:      MOV      (R4),R5      ;GET NEXT QUEUE ELEMENT
8627      051050  001425      BEQ      5$      ;IF END OF QUEUE, RETURN
8628      051052  032765  000002  000014      BIT      #DRVPOS,P.PPST(R5) ;CHECK IF DRIVE IS POSITIONING
8629      051060  001404      BFG      2$      ;NO, RETURN PARAMETER BLOCK ADDRESS
8630      051062  010504      MOV      R5,R4      ;UPDATE LINK ADDRESS POINTER
8631      051064  062704  000064      ADD      #P.QLNK,R4      ;CALCULATE LINK ADDRESS
8632      051070  000766      BR       1$      ;GET NEXT ELEMENT OF QUEUE
8633
8634      051072  016514  000064      2$:      MOV      P.QLNK(R5),(R4) ;UPDATE QUEUE LINK
8635      051076  001012      BNE      5$      ;CHECK IF LAST ELEMENT OF QUEUE
8636      ;*      ; IF NOT, DO NOT UPDATE QUEUE TAIL
8637      051100  022704  001324      CMP      #CINITQ,R4      ;CHECK IF ONLY ELEMENT IN QUEUE
8638      051104  001405      BFG      4$      ;YES, CLEAR HEAD AND TAIL
8639      051106  162704  000064      SUB      #P.QLNK,R4      ;ADJUST R4 FOR TAIL POINTER
8640      051112  010437  001326      MOV      R4,CINITQ+2      ;LOAD TAIL
8641      051116  000207      RTS      PC      ;RETURN
8642
8643      051120  005024      4$:      CLR      (R4)+      ;CLEAR HEAD OF QUEUE
8644      051122  005014      CLR      (R4)      ;CLEAR TAIL OF QUEUE
8645      051124  000207      5$:      RTS      PC      ;RETURN

```

```

8646 .SBTTL *REMOVE PARAMETER BLOCK FROM COMMAND INITIATION QUEUE
8647
8648 ;;*****
8649 ;*
8650 ;* THIS ROUTINE WILL CHECK IF THE CURRENT PARAMETER BLOCK
8651 ;* IS IN THE COMMAND INITIATION QUEUE. IF IT IS, THIS
8652 ;* ROUTINE WILL REMOVE IT FROM THE COMMAND INITIATION QUEUE.
8653 ;*
8654 ;*CALL JSR PC,Q.PMOV
8655 ;* RETURN
8656 ;*
8657 ;*NOTE: R5 CONTAINS ADDRESS OF CURRENT PARAMETER BLOCK.
8658 ;*
8659 ;;*****
8660
8661 051126 032765 040000 000014 Q.RMOV: BIT #Q.INIT,P.PRST(R5) ;CHECK IF PARAMETER BLOCK ENQUEUED IN
8662 ; INITIATION QUEUE
8663 051134 001001 BNE SS ;YES, REMOVE PARAMETER BLOCK
8664 ; FROM COMMAND INITIATION QUEUE
8665 051136 000207 RTS PC ;RETURN
8666
8667 051140 010446 SS: MOV R4,-(SP) ;STORE R4 ON STACK
8668 051142 010346 MOV R3,-(SP) ;STORE R3 ON THE STACK
8669 051144 012704 001324 MOV #CINITQ,R4 ;LOAD ADDRESS OF COMMAND INITIATION
8670 ; QUEUE IN R4
8671 051150 011403 10$: MOV (P4),P3 ;GET NEXT QUEUE ELEMENT
8672 051152 001001 BNE 11$ ;CHECK IF NO MORE ELEMENTS
8673 051154 000000 HALT ; *** FLAGS DO NOT AGREE WITH
8674 ; COMMAND INITIATION QUEUE
8675 051156 020305 11$: CMP R3,P5 ;CHECK IF CORRECT PARAMETER BLOCK
8676 051160 001404 BEQ 12$ ;YES, ADJUST QUEUE
8677 051162 010304 MOV R3,R4 ;UPDATE LINK ADDRESS POINTER
8678 051164 062704 000064 ADD #P.QLNK,R4 ;CALCULATE LINK ADDRESS
8679 051170 000767 BR 10$ ;GET NEXT ELEMENT OF QUEUE
8680
8681 051172 016314 000064 12$: MOV P.QLNK(R3),(R4) ;UPDATE QUEUE LINK
8682 051176 001012 BNE 15$ ;CHECK IF LAST ELEMENT OF QUEUE
8683 ; IF NOT, DO NOT UPDATE QUEUE TAIL
8684 051200 022704 001324 CMP #CINITQ,R4 ;CHECK IF ONLY ELEMENT IN QUEUE
8685 051204 001405 BEQ 14$ ;YES, CLEAR HEAD AND TAIL
8686 051206 162704 000064 SUB #P.QLNK,R4 ;ADJUST R4 FOR TAIL POINTER
8687 051212 010437 001226 MOV R4,CINITQ+2 ;LOAD TAIL
8688 051216 000407 BR 15$ ;RETURN
8689
8690 051220 005024 14$: CLR (R4)+ ;CLEAR HEAD OF QUEUE
8691 051222 005014 CLR (P4) ;CLEAR TAIL OF QUEUE
8692 051224 042765 040006 000014 15$: BTL #DRVPDT+DHVPCSIG,INIT,P.PRST(R5) ;CLEAR DRIVE POSITIONING,
8693 ; DRIVE POSITIONING FOR DATA TRANSFER, AND
8694 ; PARAMETER BLOCK ENQUEUED IN INITIATION QUEUE
8695
8696 051232 012603 MOV (SP)+,R3 ;RESTORE R3
8697 051234 012604 MOV (SP)+,R4 ;RESTORE R4
8698 051236 000707 RTS PC ;RETURN

```

```

8699      .SBTTL  *COMMAND OPTIMIZER
8700
8701      ;;*****
8702      ;*
8703      ;*      THIS ROUTINE WILL INITIATE THE COMMAND AS SPECIFIED IN THE
8704      ;*      PARAMETER BLOCK OF THE FIRST DRIVE WHICH IS NOT POSITIONING
8705      ;*      IN THE COMMAND INITIATION QUEUE.
8706      ;*
8707      ;*CALL  JSR      PC,C.OPT
8708      ;*      RETURN
8709      ;*
8710      ;*      ROUTINES USED:
8711      ;*              C.INIT
8712      ;*              Q.PUSH
8713      ;*              G.SPCN
8714      ;*
8715      ;;*****
8716
8717      C.OPT:  MOV      R5,-(SP)      ;STORE R5 ON STACK
8718             MOV      R4,-(SP)      ;STORE R4 ON STACK
8719             MOV      R3,-(SP)      ;STORE R3 ON STACK
8720             MOV      R2,-(SP)      ;STORE R2 ON STACK
8721             MOV      R1,-(SP)      ;STORE R1 ON STACK
8722             MOV      R0,-(SP)      ;STORE R0 ON STACK
8723             MOV      PS,-(SP)      ;STORE PSW ON STACK
8724             MOV      RKPRI,PS      ;LOCK OUT RK06 INTERRUPTS
8725             TST      O.WAIT        ;CHECK IF WAITING FOR COMMAND COMPLETION
8726             BNE      O.RTRN        ;YES, RESTORE REGISTERS
8727             MOV      RFBAS,R2      ;LOAD R2 WITH BASE ADDRESS OF THE RK06
8728             ; REGISTERS
8729             MOV      RKCSI(R2),T.CSI ;STORE COMMAND AND STATUS REGISTER 1
8730             BIT      #RDY,T.CSI    ;CHECK IF READY SET
8731             BEQ      O.RTRN        ;NO, WAIT FOR COMMAND COMPLETION
8732             JSR      PC,O.SRCH      ;SEARCH FOR PARAMETER BLOCK ADDRESS
8733             TST      R5            ;CHECK IF NO COMMAND CAN BE ISSUED
8734             BNE      IS            ;YES, ISSUE COMMAND
8735             BIT      #DI,T.CSI     ;CHECK IF DRIVE INTERRUPT WAITING
8736             BEQ      O.RTRN        ;NO, RETURN
8737             MOVR     #-1,I.ISRL    ;SET FLAG THAT INTERRUPT WAS FORCED
8738             MOV      #INTR,RKCSI(R2) ;FORCE INTERRUPT TO SCAN DRIVE ATTENTIONS
8739             BR       O.RTRN        ;RESTORE REGISTERS
8740
8741             1S:  CLR      P.ERR(R5) ;CLEAR PROOP STATUS INFORMATION
8742             CMPR     #RDALHD,P.CMND(R5) ;CHECK IF READ ALL HEADERS
8743             BEQ      2S            ;YES, CHECK IF IMPLIED SEEK
8744             CMPR     #PDDATA,P.CMND(R5) ;CHECK IF DATA TRANSFER
8745             BHI      3S            ;NO, GO TO THE COMMAND INITIATOR
8746             CMPR     #WRTCHK,P.CMND(R5) ;CHECK IF SPECIAL COMMAND
8747             BLO      3S            ;YES, GO TO THE COMMAND INITIATOR
8748             2S:  MOVR     P.DRVN(R5),P4 ;STORE DRIVE NUMBER
8749             BITR     INTVSF(R4),C.CVER ;CHECK IF OVERLAPPED OPERATIONS FOR THIS DRIVE
8750             BEQ      3S            ;NO, GO TO THE COMMAND INITIATOR
8751             BIT      #DRVPDT,P.PRST(R5) ;CHECK IF POSITIONING FOR
8752             ; DATA TRANSFER OCCURED
8753             BEQ      5S            ;NO, ISSUE SEEK
8754             BIC      #DRVPDTIQ.INIT,P.PRST(R5) ;CLEAR POSITIONING IN PROGRESS FOR

```

```

8755                                     ; DATA TRANSFER AND PARAMETER BLOCK ENQUEUED
8756                                     ; IN INITIATION QUEUE
8757
8758 051442 010537 051452      3S:  MOV    R5,45      ;LOAD PARAMETER BLOCK ADDRESS
8759 051446 004737 047216      JSR    PC,C.INIT  ;ISSUE COMMAND
8760 051452 000000      4S:    .WOPD    0          ;ADDRESS OF PARAMETER BLOCK
8761                                     ; PLACED HERE
8762 051454 000461      BR      0.RTRN          ;RESTORE REGISTERS
8763
8764 051456 156437 001510 001506 5S:  BISR    INTMSK(R4),W.TIME ;SET WATCH DCG TIMING
8765 051464 010403      MOV    R4,R3          ;STORE DRIVE NUMBER
8766 051466 006303      ASL     R3            ;MULTIPLY DRIVE NUMBER BY 2
8767 051470 013763 001470 001540      MOV    W.SEC,W.DRV(R3) ;LOAD WATCH-DCG TIME
8768 051476 005037 001462      CLR     C.WAIT   ;CLEAR WAIT FOR COMMAND
8769 051502 105037 001502      CLRR   I.ISRL   ;CLEAR RELEASE OF INTERRUPT ISSUED
8770
8771                                     ;
8772                                     ;
8773                                     ;
8774                                     ;
8775                                     ;
8776                                     ;
8777 051506 042765 075176 000014      BIC     #C<DRVOSFIN.WCKINCHKIDRPDRVIDTBAIL>,P.PRST(R5)
8778
8779                                     ;
8780                                     ;
8781 051514 052765 040006 000014      BIS     #DRVPOSIDRVPDTIQ.INIT,P.PRST(R5)
8782
8783 051522 010500      MOV    R5,R0          ;STORE PARAMETER BLOCK ADDRESS
8784 051524 062700 000016      ADD     #P.CS1,R0   ;CALCULATE FIRST LOCATION TO BE CLEARED
8785 051530 010501      MOV    R5,R1          ;STORE PARAMETER BLOCK ADDRESS
8786 051532 062701 000062      ADD     #P.EPAT,R1  ;CALCULATE LAST LOCATION TO BE CLEARED
8787
8788 051536 005020      6S:  CLR     (R0)+      ;CLEAR RETURN PARAMETERS
8789 051540 020001      CMP     R0,R1          ;CHECK IF FINISHED
8790 051542 101775      BLOS    6S            ;NO, CLEAR NEXT RETURN PARAMETER
8791 051544 005067 000026      CLR     RKNR1(R7) ;CLEAR MAINTENANCE REGISTER 1
8792 051550 016562 000002 000020      MOV    P.CYLW(R5),RKDCYL(R2) ;LOAD CYLINDER ADDRESS
8793 051556 010462 000010      MOV    R4,PKCS2(R2) ;LOAD DEVICE NUMBER
8794 051562 116565 000007 000017      MOVW   P.CS1H(R5),P.CS1+1(R5) ;STORE BITS 0-15 OF CS1
8795 051570 042765 165777 000016      BIC     #C<CPMTICDT>,P.CS1(R5) ;CLEAR BITS EXCEPT FORMAT AND
8796                                     ; DRIVE TYPE
8797 051576 112765 000117 000016      MOVW   #SEFK,P.CS1(R5) ;STORE COMMAND
8798 051604 016562 000016 000000      MOV    P.CS1(R5),RKCS1(R7) ;ISSUE SEFK
8799 051612 004037 050706      JSR     R0,Q.PUSH ;QUEUEUP COMMAND IN COMMAND
8800 051616 001324      CINITQ          ; INITIATION QUEUE
8801
8802 051620 012637 177776      0.RTRN: MOV    (SP)+,PS   ;RESTORE PSW
8803 051624 012600      MOV    (SP)+,R0      ;RESTORE R0
8804 051626 012601      MOV    (SP)+,R1      ;RESTORE R1
8805 051630 012602      MOV    (SP)+,R2      ;RESTORE R2
8806 051632 012603      MOV    (SP)+,R3      ;RESTORE R3
8807 051634 012604      MOV    (SP)+,R4      ;RESTORE R4
8808 051636 012605      MOV    (SP)+,R5      ;RESTORE R5
8809 051640 000207      RTS     PC            ;RETURN

```

```
0810      .SBTTL  MEMORY CHECK ENABLE TEST
0811
0812 051642 104401 063671      MEMERR: TYPE      ,ERR400      ;TYPE UNEXPECTED MEMORY PARITY TRAP
0813 051646 011646      MOV      (SP),-(SP)      ;TYPE PC VALUE
0814 051650 004737 052144      JSR      PC,RINDCT
0815 051654 104401 001165      TYPE      ,SCRLF
0816 051660 000007      RTI      ;RETURN
```

```

8817 .SBTTL TYPE ROUTINE
8818
8819 ;;*****
8820 ;;ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
8821 ;;THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
8822 ;;NOTE1: SNUL contains the character to be used as the filler character.
8823 ;;NOTE2: SPILLS contains the number of filler characters required.
8824 ;;NOTE3: SPILLC contains the character to fill after.
8825 ;;
8826 ;;CALL:
8827 ;;1) USINC A TRAP INSTRUCTION
8828 ;; TYPE ,MESADR ;;MFSADR IS FIRST ADDRESS OF AN ASCIZ STRING
8829 ;;OP
8830 ;; TYPE
8831 ;; MFSADR
8832 ;;
8833
8834 051667 105737 001157 $TYPE: TSTB STPFLC ;;IS THERE A TERMINAL?
8835 051666 100002 BPL 1$ ;;BR IF YES
8836 051670 000000 HALT ;;HALT HERE IF NO TERMINAL
8837 051672 000430 BR 3$ ;;LEAVE
8838 051674 010046 1$: MOV R0,-(SP) ;;SAVE R0
8839 051676 017600 000002 MOV 02(SP),R0 ;;GET ADDRESS OF ASCIZ STRING
8840 051702 122737 000001 001210 CMPR #APTENV,SENV ;;RUNNING IN APT MODE
8841 051710 001011 BNE 67$ ;;NO,GO CHECK FOR APT CONSOLE
8842 051712 132737 000100 001211 BITR #APTSPOOL,SENV ;;SPOOL MESSAGE TO APT
8843 051720 001405 BEQ 62$ ;;NO,GO CHECK FOR CONSOLE
8844 051722 010037 051732 MOV R0,61$ ;;SETUP MESSAGE ADDRESS FOR APT
8845 051726 004737 054574 JSR PC,$ATY? ;;SPOOL MESSAGE TO APT
8846 051732 000000 61$: .WOPD 0 ;;MESSAGE ADDRESS
8847 051734 132737 000040 001211 62$: BITB #APTCSP,SENV ;;APT CONSOLE SUPPRESSED
8848 051742 001003 BNE 60$ ;;YES,SKIP TYPE OUT
8849 051744 112046 2$: MOVB (R0)+,-(SP) ;;PUSH CHARACTER TO BE TYPED ONTC STACK
8850 051746 001005 BNE 4$ ;;BR IF IT ISN'T THE TERMINATOR
8851 051750 005726 TST (SP)+ ;;IF TERMINATOR POP IT OFF THE STACK
8852 051752 012600 60$: MOV (SP)+,R0 ;;RESTORE R0
8853 051754 062716 000002 3$: ADD #2,(SP) ;;ADJUST RETURN PC
8854 051760 000002 RTI ;;RETURN
8855 051762 122716 000011 4$: CMPR #RT,(SP) ;;BRANCH IF <RT>
8856 051766 001430 BFQ 8$ ;;BRANCH IF ACT <CALLF>
8857 051770 122716 000200 CMPR #CRLF,(SP)
8858 051774 001006 BNE 5$
8859 051776 005726 TST (SP)+ ;;POP <CR><LF> EQUIV
8860 052000 104401 TYPE ;;TYPE A CR AND LF
8861 052002 001165 SCRLF
8862 052004 105037 052140 CLRB $CHARCNT ;;CLEAR CHARACTER COUNT
8863 052010 000755 BR 2$ ;;GET NEXT CHARACTER
8864 052012 004737 052074 5$: JSR PC,$TYPEC ;;GO TYPE THIS CHARACTER
8865 052016 123726 001156 6$: CMPR $FILLC,(SP)+ ;;IS IT TIME FOR FILLER CHARS.?
8866 052022 001350 BNE 2$ ;;IF NO GO GET NEXT CHAR.
8867 052024 013746 001154 MOV $NULL,-(SP) ;;GET # OF FILLER CHARS. NEEDED
8868 ;;AND THE NULL CHAR.
8869 052030 105366 000001 7$: DECB 1(SP) ;;DOES A NULL NEED TO BE TYPED?
8870 052034 002770 BLT 6$ ;;BR IF NO--GO POP THE NULL OFF OF STACK
8871 052036 004737 052074 JSR PC,$TYPEC ;;GO TYPE A NULL
8872 052042 105337 052140 DFCB $CHARCNT ;;DO NOT COUNT AS A COUNT

```

```

8873 052046 000770          BR      75          ;;LOOP
8874
8875          ;HORIZONTAL TAB PROCESSOR
8876
8877 052050 112716 000040      85:      MOVR      R*,(SP)          ;;REPLACE TAB WITH SPACE
8878 052054 004737 052074      95:      JSR      PC,STPEC          ;;TYPE A SPACE
8879 052060 132737 000007 052140      BITB      R7,$CHARCNT          ;;BRANCH IF ACT AT
8880 052066 001372          BNE      95          ;;TAB STOP
8881 052070 005726          TST      (SP)+          ;;POP SPACE OFF STACK
8882 052072 000724          BP      25          ;;GET NEXT CHARACTER
8883 052074 105777 127050      STPEC: TSTB      0STPS          ;;WAIT UNTIL PRINTER IS READY
8884 052100 100375          BPL      STPEC
8885 052102 116677 000002 127042      MOVB      2(SP),0STPB          ;;LOAD CHAR TO BE TYPED INTO DATA REG.
8886 052110 122766 000015 000002      CMPB      BCR,2(SP)          ;;IS CHARACTER A CARRIAGE RETURN?
8887 052116 001003          BNE      1$          ;;BRANCH IF NC
8888 052120 105037 052140      CLRB      $CHARCNT          ;;YES--CLEAR CHARACTER COUNT
8889 052124 000406          BR      STPEC          ;;EXIT
8890 052126 122766 000012 000002 1$:      CMPB      0LF,2(SP)          ;;IS CHARACTER A LINE FEED?
8891 052134 001402          BEQ      STPEC          ;;BRANCH IF YES
8892 052136 105227          INCB      (PC)+          ;;COUNT THE CHARACTER
8893 052140 000000      $CHARCNT:=WORD 0          ;;CHARACTER COUNT STORAGE
8894 052142 000207      STPEC: RTS      PC
8895
8896
8897          .SBTTL  BINARY TO OCTAL (ASCII) CONVERSION
8898
8899          ;;*****
8900          ;*
8901          ;*      THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A
8902          ;*      6-DIGIT OCTAL (ASCII) NUMBER.
8903          ;*
8904          ;*CALL  MOV      B0N,(SP)
8905          ;*      JSR      PC,BINOCT
8906          ;*      RETURN          ;NUMBER IN OCTSTG
8907          ;*
8908          ;;*****
8909
8910      BINOCT:
8911 052144          MOV      R3,-(SP)          ;;PUSH R3 ON STACK
8912 052146          MOV      R4,-(SP)          ;;PUSH R4 ON STACK
8913 052150          MOV      R5,-(SP)          ;;PUSH R5 ON STACK
8914 052152 012704 000006          MOV      R6,R4          ;;STORE COUNT
8915 052156 016605 000010          MOV      10(SP),R5          ;;PICKUP INPUT NUMBER
8916 052162 012703 052240          MOV      0OCTSTG,R3          ;;LOAD ADDRESS OF CONVERTED STRING
8917 052166 005013          CLR      (R3)          ;;CLEAR OUTPUT BYTE
8918 052170 006105          BCL      R4          ;;PCTATE MSB INTO "C"
8919 052172 000404          BR      3$          ;;GO TO PSR
8920
8921 052174 006105          2$:      ROL      R4          ;;PCRB THIS DIGIT
8922 052176 006105          ROL      R5
8923 052200 006105          ROL      R5
8924 052202 110513          MOVR      R5,(R3)
8925 052204 106113          3$:      RCLP      (R3)          ;;GET LSP OF THIS DIGIT
8926 052206 142713 177770          BICB      0177770,(R3)          ;;GET PID OF JUMP
8927 052212 152723 000060          BISM      0'0,(R3)+          ;;MAKE IT ASCII
8928 052216 005304          DFC      R4          ;;CHECK IF DONE

```

```

8929 052220 001365      BNE      2$          ;N/C CONVERT NEXT DIGIT
8930 052222 012605      MOV      (SP)+,R5      ;;POP STACK INTO R5
8931 052224 012604      MOV      (SP)+,R4      ;;POP STACK INTO R4
8932 052226 012603      MOV      (SP)+,R3      ;;POP STACK INTO R3
8933 052230 012616      MOV      (SP)+,(SP)    ;;SETUP STACK FOR RETURN
8934 052232 104401 052240  TYPF      ,OCTSTG    ;TYPE NUMBER
8935 052736 000207      RTS      PC          ;RETURN
8936
8937 052240 000006      OCTSTG: .BLFB      6
8938 052246      000      .BYTE      0,0
8939
8940      .SBTTL  ITY INPUT ROUTINE
8941
8942      ;;*****
8943      .FNABL  LSB
8944
8945      .DSABL  LSB
8946
8947
8948      ;;*****
8949      ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE ITY
8950      ;*CALL:
8951      ;*      RDCHR          ;;INPUT A SINGLE CHARACTER FROM THE ITY
8952      ;*      RETURN HERE    ;;CHARACTER IS ON THE STACK
8953      ;*                    ;;WITH PARITY BIT STRIPPED OFF
8954      ;
8955
8956 052250 011646      $PDCNR: MOV      (SP),-(SP)    ;;PUSH DOWN THE PC
8957 052252 016666 000004 000002  MOV      4(SP),2(SP)    ;;SAVE THE PS
8958 052260 105777 126660 1$:      TSTB      @STKS      ;;WAIT FOR
8959 052764 100375      BPL      1$          ;;A CHARACTER
8960 052266 117766 126654 000004  MOVR      @STFB,4(SP)    ;;READ THE ITY
8961 052274 042766 177600 000004  BIC      @C<177>,4(SP)    ;;GET RID OF JUNK IF ANY
8962 052302 076627 000004 00002$  CMP      4(SP),@73      ;;IS IT A CONTROL-S?
8963 052310 001013      BNE      3$          ;;BRANCH IF NO
8964 052312 105777 126626 2$:      TSTB      @STKS      ;;WAIT FOR A CHARACTER
8965 052316 100375      BPL      2$          ;;LOOP UNTIL ITS THERE
8966 052320 117746 126622      MOVR      @STFB,-(SP)    ;;GET CHARACTER
8967 052324 042716 177600      BIC      @C177,(SP)    ;;MAKE IT 7-BIT ASCII
8968 052330 022627 000021      CMP      (SP)+,@71      ;;IS IT A CONTROL-Q?
8969 052334 001366      BNE      2$          ;;IF NOT DISCARD IT
8970 052336 000750      BP      1$          ;;YES, RESUME
8971 052340 026627 000004 000140 3$:      CMP      4(SP),@140    ;;IS IT UPPER CASE?
8972 052346 002407      BLT      4$          ;;BRANCH IF YES
8973 052350 026627 000004 000175      CMP      4(SP),@175    ;;IS IT A SPECIAL CHAR?
8974 052356 003003      BGT      4$          ;;BRANCH IF YES
8975 052360 042766 000040 000C04  BIC      @40,4(SP)    ;;MAKE IT UPPER CASE
8976 052366 000002 4$:      RTI          ;;GO BACK TO USER
8977      ;;*****
8978      ;*THIS ROUTINE WILL INPUT A STRING FROM THE ITY
8979      ;*CALL:
8980      ;*      RDLIN          ;;INPUT A STRING FROM THE ITY
8981      ;*      RETURN HERE    ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
8982      ;*                    ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
8983
8984 052370 010346      $PDLIN: MOV      R3,-(SP)    ;;SAVE R3

```

8985	052372	005046			CLR	-(SP)	;;CLEAR THE RUBOUT KEY
8986	052374	012703	052624	15:	MOV	#STTYIN,R3	;;GET ADDRESS
8987	052400	022703	052655	25:	CMP	#STTYIN+25,,R3	;;BUFFER FULL?
8988	052404	101456			BLOS	45	;;BR IF YFS
8989	052406	104402			ROCHR		;;GO READ ONE CHARACTER FROM THE TTY
8990	052410	112613			MOVB	(SP)+,(P3)	;;GET CHARACTER
8991	052412	122713	000177	105:	CMPP	#177,(R3)	;;IS IT A RUBOUT
8992	052416	001022			BNE	55	;;BR IF NO
8993	052420	005716			TST	(SP)	;;IS THIS THE FIRST RUBOUT?
8994	052422	001007			BNE	65	;;BR IF NO
8995	052424	112737	000134	052627	MOVB	#'\,95	;;TYPE A BACK SLASH
8996	052432	104401	052622		TYPE	,95	
8997	052436	012716	177777		MOV	#-1,(SP)	;;SET THE RUBOUT KEY
8998	052442	005303		65:	DFC	R3	;;BACKUP BY ONE
8999	052444	020327	052624		CMP	R3,#STTYIN	;;STACK EMPTY?
9000	052450	103434			BLC	45	;;BR IF YES
9001	052452	111337	052622		MOVB	(R3),95	;;SETUP TO TYPEOUT THE DELETED CHAR.
9002	052456	104401	052622		TYPE	,95	;;GO TYPE
9003	052462	000746			BR	25	;;GO READ ANOTHER CHAR.
9004	052464	005716		55:	TST	(SP)	;;RUBOUT KEY SET?
9005	052466	001406			BFO	75	;;BR IF NO
9006	052470	112737	000134	052627	MOVB	#'\,95	;;TYPE A BACK SLASH
9007	052476	104401	052622		TYPE	,95	
9008	052502	005016			CLR	(SP)	;;CLEAR THE RUBOUT KEY
9009	052504	122713	000025	75:	CMPP	#25,(R3)	;;IS CHARACTER A CTRL U?
9010	052510	001003			BNE	65	;;BR IF NO
9011	052512	104401	052655		TYPE	,SCNTLU	;;TYPE A CONTROL "U"
9012	052516	000726			BR	15	;;GO START OVER
9013	052520	122713	000022	85:	CMPP	#22,(R3)	;;IS CHARACTER A "RM"?
9014	052524	001011			BNE	35	;;BRANCH IF AC
9015	052526	105013			CLRP	(R3)	;;CLEAR THE CHARACTER
9016	052530	104401	001165		TYPE	,SCPLF	;;TYPE A "CR" & "LF"
9017	052534	104401	052624		TYPE	,STTYIN	;;TYPE THE INPUT STRING
9018	052540	000717			BR	25	;;GO PICKUP ANOTHER CHARACTER
9019	052542	104401	001164	45:	TYPE	,SQUES	;;TYPE A "?"
9020	052546	000712			BR	15	;;CLEAR THE BUFFER AND LOCK
9021	052550	111337	052622	35:	MOVB	(R3),95	;;ECHO THE CHARACTER
9022	052554	104401	052622		TYPE	,95	
9023	052560	122723	000015		CMPP	#15,(R3)+	;;CHECK FOR RETURN
9024	052564	001305			BNE	25	;;LOOP IF NOT RETURN
9025	052566	105063	177777		CLRP	-1(P3)	;;CLEAR RETURN (TYPE 15)
9026	052572	104401	001166		TYPE	,SLF	;;TYPE A LINE FEED
9027	052576	005726			TST	(SP)+	;;CLEAR RUBOUT KEY FROM THE STACK
9028	052600	012603			MOV	(SP)+,R3	;;RESTORE R3
9029	052602	011646			MOV	(SP),-(SP)	;;ADJUST THE STACK AND PUT ADDRESS OF THE
9030	052604	016666	000004	000002	MOV	4(SP),2(SP)	;; FIRST ASCII CHARACTER IN IT
9031	052612	012766	052624	000004	MOV	#STTYIN,4(SP)	
9032	052620	000002			RTI		;;RETURN
9033	052622	000		95:	.BYTE	0	;;STORAGE FOR ASCII CHAR. TO TYPE
9034	052627	000			.BYTE	0	;;TERMINATOR
9035	052624	000031		STTYIN:	.RLVB	25.	;;RESERVE 25. BYTES FOR TTY INPUT
9036	052655	136	006525	000012	SCNTLU:	.ASCII7 /"U/<15><12>	;;CONTROL "U"
9037	052662	043536	005015	000	SCNTLG:	.ASCII2 /"G/<15><12>	;;CONTROL "G"
9038	052667	015	051412	051127	\$SWR:	.ASCII7 <15><12>/SWR = /	
9039	052674	036440	000040				
9040	052700	020040	042516	020127	\$MNEW:	.ASCII7 / NEW = /	

```

9041 052706 020075 000
9042 052712
9043
9044 .SBTTL TK INITIALIZE ROUTINE
9045
9046 ;;*****
9047 ;*
9048 ;* THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD TO ALL THE
9049 ;* OPERATOR TO COMMUNICATE WITH THE PROGRAM.
9050 ;*
9051 ;*CALL
9052 ;* JSR PC,STKINT
9053 ;* RETURN
9054 ;*
9055 ;;*****
9056
9057 052712 012737 052742 000060 STKINT: MOV #STKSRV,TRVFC ;INITIALIZE THE KEY BOARD VECTOR
9058 052720 013737 001450 000062 MOV RKPPI,TRVEC+2 ;LOCK RK06 AND TTY INTERRUPTS
9059 052726 005777 126714 TST @STKB ;CLEAR DONE FLAG
9060 052732 012777 000100 126204 MOV #IE,@STKS ;ENABLE INTERRUPT
9061 052740 000207 RTS PC ;RETURN
9062
9063 .SBTTL TK SERVICE ROUTINE (ONLY CONTROL C IS REGNIZED)
9064
9065 052742 117737 126200 053304 STKSRV: MOVB @STKB,25 ;PICK UP CHARACTER AND STORE POP
9066 ; PRINT OUT
9067 052750 142737 000200 053304 BITB #BIT7,25 ;STRIP THE JUNK
9068 052756 122737 000003 053304 CMPP #3,25 ;IS IT A CONTROL-C
9069 052764 001022 BNE 15 ;NO, PRINT <CHAR>?<15><12>
9070 052766 104401 053311 TYPE ,SCNTLC ;TYPE A CONTROL-C
9071 052772 005077 126146 CLR @STFS ;CLEAR INTERRUPT ENABLE
9072 052776 016637 000002 177776 MOV 2(SP),PS ;RESTORE PREVIOUS PSM
9073 053004 004737 011526 JSR PC,OPPCMD ;PROCESS OPERATOR COMMAND
9074 053010 013737 001450 177776 MOV RKPPI,PS ;LOAD PSM TO LOCK OUT TTY AND RKCE
9075 053016 005777 126124 TST @STKB ;CLEAR DONE FLAG
9076 053022 012777 000100 126114 MOV #IE,@STFS ;ENABLE INTERRUPTS
9077 053030 000002 RTI ;RETURN
9078
9079 053032 122737 000007 053304 15: CMPP #7,25 ;CHECK IF CONTROL C
9080 053040 001403 BFQ 55 ;YES, GET NEW SWITCH REGISTER VALUE
9081 053042 104401 053304 TYPE ,25 ;PRINT CHARACTER ?<15><12>
9082 053046 000002 35: RTI ;RETURN
9083
9084 053050 022737 000176 001140 55: CMP #SWREG,SWP ;CHECK IF SOFTWARE SWITCH REG
9085 053056 001373 BNE 35 ;NO, RETURN
9086 053060 122737 000001 001134 CMPB #1,SAUTCB ;CHECK IF IN AUTO ACCEPT MODE
9087 053066 001767 BEQ 35 ;YES, RETURN
9088 053070 104401 052662 TYPE ,SCNTLC ;TYPE CONTROL C
9089 053074 104401 052667 TYPE ,SMSWP ;TYPE OLD SWITCH REGISTER
9090 053100 013746 000176 MOV SWREG,-(SP)
9091 053104 004737 052144 JSR PC,RINUCT
9092 053110 104401 052700 TYPE ,SMNEW
9093 053114 005077 126024 CLR @STFS ;GO INTO FLAG MODE
9094 053120 005046 195: CLR -(SP) ;CLEAR COUNTER
9095 053122 005046 CLR -(SP) ;THE NEW SWR
9096 053124 105777 126014 75: TSTP @STFS ;CHAR THERE?

```

```

9097 053130 100375          BPL      75          ;;IF NOT TRY AGAIN
9098
9099 053132 117746 126010      MOVB    @STFB,-(SP)      ;;PICK UP CHAP
9100 053136 042716 177600      BIC     @^C177,(SP)      ;;MAKE IT 7-BIT ASCII
9101
9102 053142 021627 000025      95:     CMP     (SP),#25      ;;IS IT A CONTROL-U?
9103 053146 001005              BNE     10$              ;;BRANCH IF NOT
9104 053150 104401 052655      TYPF    ,SCNTLU      ;;YES, ECHO CONTROL-U (^U)
9105 053154 062706 000006      20$:    ADD     @6,SP      ;;IGNORE PREVIOUS INPUT
9106 053160 000757              BP      19$              ;;LET'S TRY IT AGAIN
9107
9108
9109 053162 021627 000015      10$:    CMP     (SP),#15      ;;IS IT A <CR>?
9110 053166 001016              BNE     16$              ;;BRANCH IF NOT
9111 053170 005766 000004      TST     4(SP)          ;;YES, IS IT THE FIRST CHAP?
9112 053174 001403              BFG     11$              ;;BRANCH IF YES
9113 053176 016677 000002 175734      MOV     2(SP),@SWR      ;;SAVE NEW SWR
9114 053204 062706 000006      11$:    ADD     @6,SP      ;;CLEAR UP STACK
9115 053210 104401 001165      14$:    TYPE    ,SCPLF      ;;ECHO <CR> AND <LF>
9116 053214 012777 000100 125722      MOV     @100,@STKS      ;;RE-ENABLE TTY KEY INTERRUPTS
9117 053222 000002              15$:    RTI              ;;RETURN
9118 053224 004737 052074      16$:    JSR     PC,STYPEC      ;;ECHO CHAP
9119 053230 021627 000060      CMP     (SP),@60          ;;CHAR < 0?
9120 053234 002420              BLT     18$              ;;BRANCH IF YES
9121 053236 021627 000067      CMP     (SP),@67          ;;CHAR > 7?
9122 053242 003015              BGT     18$              ;;BRANCH IF YES
9123 053244 042726 000060      BIC     @60,(SP)+      ;;STRIP-OFF ASCII
9124 053250 005766 000002      TST     2(SP)          ;;IS THIS THE FIRST CHAP?
9125 053254 001403              BFG     17$              ;;BRANCH IF YES
9126 053256 006316              ASL     (SP)          ;;NO, SHIFT PRESENT
9127 053260 006316              ASL     (SP)          ;;  CHAP OVER TO MAKE
9128 053262 006316              ASL     (SP)          ;;  ROOM FOR NEW ONE.
9129 053264 005266 000002      17$:    INC     2(SP)          ;;KEEP COUNT OF CHAP
9130 053270 056616 177776      BIS     -2(SP),(SP)      ;;SET IN NEW CHAP
9131 053274 000713              BR      7$              ;;GET THE NEXT ONE
9132 053276 104401 001164      18$:    TYPE    ,SQUES      ;;TYPE ?<CR><LF>
9133 053302 000724              BR      20$              ;;SIMULATE CONTROL-U
9134
9135 053304      000          2$:     .BYTE    0          ;STORAGE FOR QUESTIONABLE INPUT
9136 053305      077 005015      .ASCIZ    /?/<15><17>
9137 053311      136 006503 000012 SCNTLC: .ASCIZ    /^C/<15><12>      ;CONTROL-C
9138
9139      .SBTTL   OCTAL TO BINARY CONVERSION ROUTINE
9140
9141      ;;*****
9142      ;;*
9143      ;;*   THIS ROUTINE WILL CHECK A STRING OF ASCII CHARACTERS TERMINATED
9144      ;;*   WITH A NULL <000> OR CCMA. IF THE CHARACTERS ARE LEGAL
9145      ;;*   IT WILL GENERATE TWO BINARY WORDS PLACING THE LOW 16 BITS
9146      ;;*   ON THE STACK AND THE HIGH 16 BITS IN LOCATION $HI0CT.
9147      ;;*
9148      ;;*CALL
9149      ;;*   MOV     <ADDRESS OF ASCII STRING>,-(SP)
9150      ;;*   JSR     PC,OCTBIN
9151      ;;*   <ADDRESS OF EPOCH RETURN>
9152      ;;*   RETURN

```

```

9153 ;*
9154 ;*****
9155
9156 OCTBIN: MOV R0,-(SP) ;SAVE R0
9157 MOV R1,-(SP) ;SAVE R1
9158 MOV R2,-(SP) ;SAVE R2
9159 MOV 10(SP),R0 ;GET ADDRESS OF ASCII STRING
9160 CLR R1 ;CLEAR DATA WORDS
9161 CLR R2
9162 25: MOVN (R0)+,-(SP) ;PICK THIS CHARACTER
9163 BEC 35 ;IF ZPRC GET OUT
9164 CMPR (SP),R' ;CHECK IF COMMA
9165 BPC 35 ;IF COMMA GET OUT
9166 CMPR R'0,(SP) ;MAKE SURE THIS CHARACTER IS
9167 BGT 45 ; AN OCTAL DIGIT
9168 CMPR R'7,(SP)
9169 BLT 45
9170 ASL R1 ; *2
9171 ROL R2
9172 ASL R1 ; *4
9173 ROL R2
9174 ASL R1 ; *8
9175 ROL R2
9176 BIC R'07,(SP) ;STRIP THE ASCII JUNK
9177 ADD (SP)+,R1 ;ADD THIS DIGIT
9178 BR 25 ;LOOP
9179 35: TST (SP)+ ;CLEAN PARTIAL FROM STACK
9180 MOV R1,10(SP) ;SAVE RESULT
9181 MOV R2,SHIOCT
9182 MOV (SP)+,R2 ;RESTORE R2
9183 MOV (SP)+,R1 ;RESTORE R1
9184 MOV (SP)+,R0 ;RESTORE R0
9185 ADD R2,(SP) ;ADJUST RETURN
9186 RTS PC ;RETURN
9187
9188 45: TST (SP)+ ;CLEAN UP PARTIAL FROM STACK
9189 MOV (SP)+,R2 ;RESTORE R2
9190 MOV (SP)+,R1 ;RESTORE R1
9191 MOV (SP)+,R0 ;RESTORE R0
9192 MOV R(SP)+,(SP) ;PUT ADDRESS OF ERROR ROUTINE ON STACK
9193 RTS PC ;GO PROCESS ERROR
9194 SHIOCT: .WOPD 0 ;HIGH ORDER BITS GO HERE
9195 .SBTTL OCTAL TO BINARY CONVERSION ROUTINE
9196
9197 ;*****
9198 ;*
9199 ;* THIS ROUTINE WILL CHECK A STRING OF ASCII CHARACTERS TERMINATED
9200 ;* WITH A NULL <000> OR COMMA. IF THE CHARACTERS ARE LEGAL,
9201 ;* IT WILL GENERATE A BINARY WORD PLACING IT ON THE STACK.
9202 ;*
9203 ;*CALL
9204 ;* MOV <ADDRESS OF ASCII STRING>,-(SP)
9205 ;* JSR PC,DECBIN
9206 ;* <ADDRESS OF ERROR RETURN>
9207 ;* RETURN
9208 ;*

```

```

9209      ;*****
9210
9211 053452 010046      DECBIN: MOV      R0,-(SP)      ;SAVE R0
9212 053454 010146      MOV      R1,-(SP)      ;SAVE R1
9213 053456 010246      MOV      R2,-(SP)      ;SAVE R2
9214 053460 016600 000010      MOV      10(SP),R0      ;GET ADDRESS OF ASCII STRING
9215 053464 005046      CLR      -(SP)      ;CLEAR DATA WORD
9216 053466 005002      CLR      R2      ;SIGN SET POSITIVE
9217 053470 122710 000055      CMPB    #'-(R0)      ;SEE IF A MINUS SIGN
9218 053474 001001      BNE      2S      ;BRANCH IF NO MINUS SIGN
9219 053476 112002      MOVB    (R0)+,R2      ;SAVE FOR LATER USE
9220 053500 112001      2S:      MOVB    (R0)+,R1      ;PICKUP THIS CHARACTER
9221 053502 001427      BFG      3S      ;GET OUT IF ZERO
9222 053504 120127 000054      CMPB    R1,#',      ;CHECK IF COMMA
9223 053510 001424      BEG      3S      ;GET OUT IF COMMA
9224 053512 122701 000060      CMPB    #'0,R1      ;MAKE SURE THIS CHARACTER IS
9225 053516 003034      BGT      5S      ; A DIGIT BETWEEN 0 & 9
9226 053520 122701 000071      CMPB    #'9,R1
9227 053524 002431      BLT      5S
9228 053526 032716 170000      BIT      #170000,(SP)      ;DON'T LET NUMBER GET TO PIC
9229 053532 001026      BNE      5S      ;BRANCH IF NUMBER WOULD OVERFLOW
9230 053534 006316      ASL      (SP)      ; *2
9231 053536 011646      MOV      (SP),-(SP)      ;SAVE FOR LATER
9232 053540 006316      ASL      (SP)      ; *4
9233 053542 006316      ASL      (SP)      ; *8
9234 053544 062616      ADD      (SP)+,(SP)      ; *16
9235 053546 102420      BVS      5S      ;OVERFLOW ISN'T ALLOWED
9236 053550 162701 000060      SUB      #'0,R1      ;STRIP AWAY THE ASCII JUNK
9237 053554 060116      ADD      R1,(SP)      ;ADD IN THIS DIGIT
9238 053556 102414      BVS      5S      ;OVERFLOW ISN'T ALLOWED
9239 053560 000747      BR       2S      ;LOOP
9240 053562 005707      3S:      TST      R2      ;CHECK IF NUMBER IS NEGATIVE
9241 053564 001401      BFG      4S      ;BRANCH IF NC
9242 053566 005416      NEG      (SP)      ;YES--NEGATE THE NUMBER
9243 053570 012666 000010      4S:      MOV      (SP)+,10(SP)      ;SAVE RESULT
9244 053574 012607      MOV      (SP)+,R2      ;RESTORE R2
9245 053576 012601      MOV      (SP)+,R1      ;RESTORE R1
9246 053600 012600      MOV      (SP)+,R0      ;RESTORE R0
9247 053602 062716 000007      ADD      #7,(SP)      ;ADJUST RETURN
9248 053606 000707      RTS      PC      ;RETURN
9249
9250 053610 005726      5S:      TST      (SP)+      ;CLEAN PARTIAL NUMBER FROM STACK
9251 053612 012607      MOV      (SP)+,R2      ;RESTORE R2
9252 053614 012601      MOV      (SP)+,R1      ;RESTORE R1
9253 053616 012600      MOV      (SP)+,R0      ;RESTORE R0
9254 053620 013616      MOV      @((SP)+,(SP)      ;PUT ADDRESS OF BRANCH ON STACK
9255 053622 000207      RTS      PC      ;CC PROCESS BRANCH
9256      .SBTTL ROUTINE TO SIZE MEMORY
9257
9258      ;*****
9259      ;*CALL:
9260      ;*      JSR      PC,$SIZE
9261      ;*      RETURN
9262      ;*$LSTAD WILL CONTAIN THE LAST AVAILABLE MEMORY LOCATION
9263
9264 053624 010046      $SIZE: MOV      R0,-(SP)      ;SAVE R0 ON THE STACK

```

9765	053626	010146			MOV	R1,-(SP)	;;SAVE R1 ON THE STACK
9766	053630	013746	000004		MOV	00ERRVEC,-(SP)	;;SAVE PRESENT ERROR VECTOR PS & PC
9767	053634	013746	000006		MOV	00ERRVEC+2,-(SP)	
9768	053640	010600			MOV	SP,PO	;;SAVE THE STACK POINTER
9769							;;SET THE ERRVEC PS TO THE PRESENT PS
9770	053642	104400			TRAP		;;PUSH OLD PSW AND PC ON STACK
9771	053644	012637	000006		MOV	(SP)+,00ERRVEC+2	;;SAVE THE PSW IN 00ERRVEC+2
9772	053650	012737	053670	000004	MOV	#7\$,00ERRVEC	;;SET FOR TIMECUT
9773	053656	012701	020000		MOV	#20000,R1	;;FIRST ADDRESS
9774	053662	005711			1S: TST	(R1)	;;TEST THIS ADDRESS
9775	053664	005721			TST	(R1)+	;;STEP TO NEXT ADDRESS
9776	053666	000775			BR	1S	;;TRY ANOTHER
9777	053670	162701	000002		2S: SUB	#7,R1	;;DROP BACK
9778	053674	010006			MOV	RO,SP	;;RESTORE THE STACK
9779	053676	012637	000006		MOV	(SP)+,00ERRVEC+2	;;RESTORE ERROR VECTOR
9780	053702	012637	000004		MOV	(SP)+,00ERRVEC	
9781	053706	010137	053720		MOV	R1,SLSTAD	;;LAST ADDRESS
9782	053712	012601			MOV	(SP)+,R1	;;RESTORE R1
9783	053714	012600			MOV	(SP)+,RO	;;RESTORE RO
9784	053716	000207			RTS	PC	
9785	053720	000000			SLSTAD: .WORD	0	;;CONTAINS THE LAST ADDRESS
9786							;;*****
9787							
9788							.SBTTL POWER DOWN AND UP ROUTINES
9789							
9790							;POWER DOWN ROUTINE
9791	053722	017737	125212	001404	SPWRDN: MOV	0SWR,SAVSWR	;;SAVE SWITCH REGISTER
9792	053730	012737	053742	000024	MOV	0SPWRUP,PWRVEC	;;SET UP VECTOR
9793	053736	000000			HALT		
9794	053740	000776			BR	--2	;;WANG UP
9795							
9796							;POWER UP ROUTINE
9797	053742	005037	054046		SPWRUP: CLR	SPWRCT	;;WAIT LOOP FOR THE TTY
9798	053746	012737	000144	054050	MOV	#100.,SPWRCT+2	
9799	053754	005237	054046		1S: INC	SPWRCT	;;WAIT FOR THE INCREMENT
9300	053760	001375			BNE	1S	;; OF WORD
9301	053762	005337	054050		DFC	SPWRCT+2	
9302	053766	001372			BNE	1S	
9303	053770	012737	053722	000024	MOV	0SPWRDN,PWRVEC	;;SET POWER DOWN VECTOR
9304	053776	012737	000340	000026	MOV	0PR7,PWRVEC+2	;;PRIORITY 7
9305	054004	012737	000340	000036	MOV	0PR7,TRAPVEC+2	;;LOCK OUT ALL INTERRUPTS FOR TRAPS
9306	054012	012706	001100		MOV	0STACK,SP	;;INITIALIZE STACK
9307	054016	104401	054052		TYPE	,SPCMFR	;;REPORT POWER FAIL
9308	054022	004737	043072		JSR	PC,PRITIM	;;PRINT TIME
9309	054026	112737	177777	001361	MOV	#-1,FLAG	;;SET FLAG FOR RESTART SEQUENCE
9310	054034	013777	001404	125076	MOV	SAVSWR,0SWR	;;RESTORE SWITCH REGISTER
9311	054042	000137	007150		JMP	PWRST	;;RESTART PROGRAM
9312							
9313	054046	000000	000000		SPWRCT: .WORD	0,0	;;WAIT COUNT FOR TTY
9314	054052	005015	047520	042527	SPWRUP: .ASCIZ	<15><12>/POWER/<15><12>	
9315	054060	006522	000012				
9316							.EVEN
9317							
9318							.SBTTL INTEGER MULTIPLY ROUTINE
9319							
9320							;;*****

```

9321      ;*
9322      ;*CALL
9323      ;*      MOV      MULTIPLIER, -(SP)
9324      ;*      MOV      MULTIPLICAND, -(SP)
9325      ;*      JSR      PC, $MULT
9326      ;*      RETURN
9327      ;*
9328      ;*      STACK      PRODUCT
9329      ;*      -----
9330      ;*
9331      ;*      TOP      LSB'S
9332      ;*      +2      MSB'S
9333      ;*
9334      ;******
9335
9336      054064      SMULT:
9337      054064      010046      MOV      R0, -(SP)      ;PUSH R0 ON STACK
9338      054066      010146      MOV      R1, -(SP)      ;PUSH R1 ON STACK
9339      054070      010246      MOV      R2, -(SP)      ;PUSH R2 ON STACK
9340      054072      010346      MOV      R3, -(SP)      ;PUSH R3 ON STACK
9341      054074      010446      MOV      R4, -(SP)      ;PUSH R4 ON STACK
9342      054076      010546      MOV      R5, -(SP)      ;PUSH R5 ON STACK
9343      054100      016605      000016      MOV      16(SP), R5      ;STORE MULTIPLICAND
9344      054104      016603      000020      MOV      20(SP), R3      ;STORE MULTIPLIER
9345      054110      005000      CLR      R0      ;CLEAR HIGH ORDER WORDS
9346      054112      005001      CLR      R1
9347      054114      005002      CLR      R2
9348      054116      005004      CLR      R4
9349      054120      012746      000041      MOV      #41, -(SP)      ;MOVE 33 DEC TC COUNTER
9350      054124      006000      1S:      ROR      R0
9351      054126      006001      ROR      R1
9352      054130      006002      ROR      R2      ;SHIFT TC ACC
9353      054132      006003      ROR      R3
9354      054134      103003      BCC      2S      ;NO CARRY NO ACC
9355      054136      060501      ADD      R5, R1
9356      054140      005500      ADC      R0      ;ADD DOUBLE PRECISION TC OBTAIN NEW PARTIAL
9357      054142      060400      ADD      R4, R0      ;PRODUCT
9358      054144      005316      2S:      DEC      (SP)      ;DECREMENT COUNTER
9359      054146      001366      BNE      1S
9360      054150      005726      TST      (SP)+
9361      054152      010766      000020      MOV      R2, 20(SP)      ;GET LEAST SIGNIFICANT BITS
9362      054156      010366      000016      MOV      R3, 16(SP)      ;GET MOST SIGNIFICANT BITS
9363      054162      012605      MOV      (SP)+, R5      ;POP STACK INTO R5
9364      054164      012604      MOV      (SP)+, R4      ;POP STACK INTO R4
9365      054166      012603      MOV      (SP)+, R3      ;POP STACK INTO R3
9366      054170      012602      MOV      (SP)+, R2      ;POP STACK INTO R2
9367      054172      012601      MOV      (SP)+, R1      ;POP STACK INTO R1
9368      054174      012600      MOV      (SP)+, R0      ;POP STACK INTO R0
9369      054176      000207      RTS      PC      ;RETURN
9370
9371
9372      .SBTTL      RANDOM NUMBR GENERATOR ROUTINE
9373
9374      ;******
9375      ;*THIS ROUTINE IS A DOUBLE PRECISION PSEUDO RANDOM NUMBR GENERATOR
9376      ;*WITH A RANGE OF 0 TO 2(+33)-1.

```

```

9377 ;*CALL:
9378 ;* JSR PC,SRAND ;*CALL THE ROUTINE
9379 ;* RETURN ;*RETURN HERE THE RANDOM
9380 ;* ;*NUMBER WILL BE IN
9381 ;* ;*SWINUM,SLONUM
9382
9383 $RAND:
9384 MOV R0,-(SP) ;*PUSH R0 ON STACK
9385 MOV R1,-(SP) ;*PUSH R1 ON STACK
9386 MOV R2,-(SP) ;*PUSH R2 ON STACK
9387 MOV $LONUM,R0 ;*SET R0 WITH LOW
9388 MOV $WINUM,R1 ;*SET R1 WITH HIGH
9389 MOV #7,R2 ;*SET SHIFT COUNT
9390 1S: ASL R0 ;*SHIFT R0 LEFT AND
9391 ROL R1 ;*ROTATE CARRY INTO R1 AND
9392 INC R2 ;*CHECK FOR CARRY
9393 BNE 1S ;*CONTINUE SHIFT LOOP
9394 ADD $LONUM,R0 ;*ADD NUMBER TO MAKE X 129
9395 ADC R1 ;*PROPAGATE CARRY
9396 ADD $WINUM,R1 ;*ADD NUMBER TO MAKE X 129
9397 ADC #1057,R0 ;*ADD LOW CONSTANT
9398 ADC R1 ;*PROPAGATE CARRY
9399 ADD #47401,R1 ;*ADD HIGH CONSTANT
9400 MOV R0,$LONUM ;*SAVE R0
9401 MOV R1,$WINUM ;*SAVE R1
9402 MOV (SP)+,R2 ;*POP STACK INTO R2
9403 MOV (SP)+,R1 ;*POP STACK INTO R1
9404 MOV (SP)+,R0 ;*POP STACK INTO R0
9405 RTS PC ;*RETURN
9406 $WINUM: .WORD 176543
9407 $LONUM: .WORD 123456
9408
9409 .SBTTL SINGLE/DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINES
9410
9411 ;*****
9412 ;*
9413 ;* THESE ROUTINE WILL CONVERT A 16-BIT OR 32-BIT BINARY NUMBER
9414 ;* TO AN UNSIGNED DECIMAL (ASCII) NUMBER.
9415 ;*
9416 ;*CALL MOV $PNTR,-(SP) ;*POINTER TO LOW WORD OF BINARY NUMBER
9417 ;* JSR PC,$DN2D
9418 ;* RETURN ;*DECIMAL NUMBER IN $DECVL
9419 ;*
9420 ;*CALL MOV $NUM,-(SP)
9421 ;* JSR PC,$SN2C
9422 ;* RETURN ;*DECIMAL NUMBER IN $SECVL
9423 ;*
9424 ;*****
9425
9426 $SN2C:
9427 MOV R0,-(SP) ;*PUSH R0 ON STACK
9428 MOV R1,-(SP) ;*PUSH R1 ON STACK
9429 MOV R2,-(SP) ;*PUSH R2 ON STACK
9430 MOV R3,-(SP) ;*PUSH R3 ON STACK
9431 MOV R4,-(SP) ;*PUSH R4 ON STACK
9432 MOV R5,-(SP) ;*PUSH R5 ON STACK

```

9433	054316	016601	000016		MOV	16(SP),R1	;STORE NUMBER IN R1
9434	054322	005002			CLR	R2	;CLEAR MOST SIGNIFICANT BITS
9435	054324	012737	000005	054450	MOV	#5,SDECNT	;SET UP FOR 5 CONVERSIONS
9436	054332	012704	054526		MOV	#STNPM,R4	;ADDRESS FOR 5 POWER
9437	054336	012705	054530		MOV	#STNPM+7,R5	
9438	054342	000421			BR	\$P2C	;START CONVERSION
9439							
9440	054344				\$DB2D:		
9441	054344	010046			MOV	R0,-(SP)	;PUSH R0 ON STACK
9442	054346	010146			MOV	R1,-(SP)	;PUSH R1 ON STACK
9443	054350	010246			MOV	R2,-(SP)	;PUSH R2 ON STACK
9444	054352	010346			MOV	R3,-(SP)	;PUSH R3 ON STACK
9445	054354	010446			MOV	R4,-(SP)	;PUSH R4 ON STACK
9446	054356	010546			MOV	R5,-(SP)	;PUSH R5 ON STACK
9447	054360	016602	000016		MOV	16(SP),R2	;PICKUP DATA POINTER
9448	054364	012701			MOV	(R2)+,R1	;PICKUP BINARY NUMBER
9449	054366	011202			MOV	(R2),R2	
9450	054370	012737	000012	054450	MOV	#10,SDECNT	;SET UP TO DO 10 CONVERSIONS
9451	054376	012704	054502		MOV	#STNPM,R4	;ADDRESS OF TEN POWER
9452	054402	012705	054504		MOV	#STNPM+2,R5	
9453							
9454	054406	012700	054552		\$P2D: MOV	#SDECVL,R0	;GET ADDRESS OF "SDECVL" STRING
9455	054412	005003			\$2DEC: CLR	R3	;CLEAR PARTIAL
9456	054414	161401			2S: SUB	(R4),R1	;SUBTRACT TEN POWER
9457	054416	005602				R2	
9458	054420	161502				(R5),R2	
9459	054422	002402			BLT	3S	;BRANCH IF TEN POWER TOO LARGE
9460	054424	005203			INC	R3	;ADD 1 TO PARTIAL
9461	054426	000772			BR	2S	;LOOP
9462							
9463	054430	062401			3S: ADD	(R4)+,R1	;RESTORE SUBTRACTED VALUE
9464	054432	005502				R2	
9465	054434	062402				(R4)+,R2	
9466	054436	022525				(R5)+,(R5)+	;MOVE TO NEXT TEN POWER
9467	054440	052703	000060			#0,R3	;CHANGE IT TO ASCII
9468	054444	110320				R3,(R0)+	;SAVE IT
9469	054446	005327				(PC)+	;DONE?
9470	054450	000000			\$DECNT: .WCPD	0	
9471	054452	001357				BNE	\$2DEC
9472	054454	105020				CLR	(R0)+
9473	054456	012605				MOV	(SP)+,R5
9474	054460	012604				MOV	(SP)+,R4
9475	054462	012603				MOV	(SP)+,R3
9476	054464	012602				MOV	(SP)+,R2
9477	054466	012601				MOV	(SP)+,R1
9478	054470	012600				MOV	(SP)+,R0
9479	054472	012616				MOV	(SP)+,(SP)
9480	054474	104401	054552			TYPE	,SDECVL
9481	054500	000207				RTS	PC
9482							;RETURN
9483	054502	145000			\$TNPMR: 145000		;1.0E09
9484	054504	035632				35632	
9485	054506	160400				160400	;1.0E08
9486	054510	002765				2765	
9487	054512	113200				113200	;1.0E07
9488	054514	000230				230	

9489	054516	041100			041100		;1.0E06
9490	054520	000017			17		
9491	054522	103240			103240		;1.0E05
9492	054524	000001			1		
9493	054526	023420		\$TMPN:	23420		;1.0E04
9494	054530	000000			0		
9495	054532	001750			1750		;1.0E03
9496	054534	000000			0		
9497	054536	000144			144		;1.0E02
9498	054540	000000			0		
9499	054542	000012			12		;1.0E01
9500	054544	000000			0		
9501	054546	000001			1		;1.0E00
9502	054550	000000			0		
9503	054552	000014		\$DECVL:	.PLFB 12.		;RESERVE STORAGE FOR ASCII STRING
9504				.SBTTL	APT COMMUNICATIONS ROUTINE		
9505							
9506							
9507	054566	112737	000001	055032	\$ATY1: MOV	#1,\$PFLC	;TO REPORT FATAL ERROR
9508	054574	112737	000001	055030	\$ATY3: MOV	#1,\$MFLC	;TO TYPE A MESSAGE
9509	054602	000403			BR	\$ATYC	
9510	054604	112737	000001	055032	\$ATY4: MOV	#1,\$PFLC	;TO ONLY REPORT FATAL ERROR
9511	054612				\$ATYC:		
9512	054612	010046			MOV	R0,-(SP)	;PUSH R0 ON STACK
9513	054614	010146			MOV	R1,-(SP)	;PUSH R1 ON STACK
9514	054616	105737	055030		TST	\$MFLC	;SHOULD TYPE A MESSAGE?
9515	054622	001450			BFC	55	;IF NOT: BR
9516	054624	122737	000001	001210	CMPB	\$APTENV,\$ENV	;OPERATING UNDER APT?
9517	054632	001031			BNE	35	;IF NOT: BR
9518	054634	132737	000100	001211	BITR	\$APTSPOOL,\$ENV	;SHOULD SPOOL MESSAGES?
9519	054642	001425			BFC	35	;IF NOT: BR
9520	054644	017600	000004		MOV	#4(SP),R0	;GET MESSAGE ADDR.
9521	054650	062766	000002	000004	ADD	#2,4(SP)	;BUMP RETURN ADDR.
9522	054656	005737	001170		15:	TST	\$MSGTYPE
9523	054662	001375			BNE	15	;IF NOT: WAIT
9524	054664	010037	001204		MOV	R0,\$MSGAD	;PUT ADDR IN MAILBOX
9525	054670	105720			25:	TSTR	(R0)+
9526	054672	001376			BNE	25	;FIND END OF MESSAGE
9527	054674	163700	001204		SUB	\$MSGAD,R0	;SUE START OF MESSAGE
9528	054700	006200			ASR	R0	;GET MESSAGE LENGTH IN WORDS
9529	054702	010037	001206		MOV	R0,\$MSGLC	;PUT LENGTH IN MAILBOX
9530	054706	012737	000004	001170	MOV	#4,\$MSGTYPE	;TELL APT TO TAKE MSG.
9531	054714	000413			BR	55	
9532	054716	017637	000004	054742	35:	MOV	#4(SP),45
9533	054724	062766	000002	000004	ADD	#2,4(SP)	;BUMP RETURN ADDRESS
9534	054732	013746	177776		MOV	177776,-(SP)	;PUSH 177776 ON STACK
9535	054736	004737	051662		JSR	PC,\$TYPE	;CALL TYPE MACRO
9536	054742	000000			45:	.WORD	0
9537	054744				55:		
9538	054744	105737	055032		105:	TSTR	\$PFLC
9539	054750	001416			BEQ	125	;SHOULD REPORT FATAL ERROR?
9540	054752	005737	001210		TST	\$ENV	;IF NOT: BR
9541	054756	001413			BFC	125	;RUNNING UNDER APT?
9542	054760	005737	001170		115:	TST	\$MSGTYPE
9543	054764	001375			BNE	115	;FINISHED LAST MESSAGE?
9544	054766	017637	000004	001172	MOV	#4(SP),\$FATAL	;IF NOT: WAIT

```

9545 054774 062766 000002 000004      ADD      #2,4(SP)          ;;BUMP RETURN ADDR.
9546 055002 005237 001170              INC      $MSGTYPE        ;;TFL APT TO TAKF ERROR
9547 055006 105037 055037      12$:  CLRB     $PFLC          ;;CLEAR FATAL FLAG
9548 055012 105037 055031          CLRB     $LFLC          ;;CLEAR LOG FLAG
9549 055016 105037 055030          CLRB     $MFLC          ;;CLEAR MESSAGE FLAG
9550 055022 012601              MOV      (SP)+,R1        ;;POP STACK INTO R1
9551 055024 012600              MOV      (SP)+,R0        ;;POP STACK INTO R0
9552 055026 000207              RTS      PC              ;;RETURN
9553 055030      000          $MFLC: .BYTE 0          ;;MESSAGE FLAG
9554 055031      000          $LFLC: .BYTE 0          ;;LOG FLAG
9555 055032      000          $PFLC: .BYTE 0          ;;FATAL FLAG
9556              055034          .EVEN
9557              000200      APTSIZE=200
9558              000001      APTENV=001
9559              000100      APTSPCL=100
9560              000040      APTCSUP=040
9561              .SBTTL  TRAP DECODER
9562
9563      ;;*****
9564      ;;THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
9565      ;;AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
9566      ;;OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
9567      ;;GO TO THAT ROUTINE.
9568
9569 055034 010046      $TRAP:  MOV      R0,-(SP)          ;;SAVE R0
9570 055036 016600 000002      MOV      2(SP),R0        ;;GET TRAP ADDRESS
9571 055042 005740      TST      -(R0)                ;;BACKUP BY 2
9572 055044 111000      MOVB     (R0),R0              ;;GET RIGHT BYTE OF TRAP
9573 055046 006300      ASL      R0                    ;;POSITION FOR INDEXING
9574 055050 016000 055070      MOV      $TRPAD(PC),R0    ;;INDEX TO TABLE
9575 055054 000200      RTS      R0                    ;;GO TO ROUTINE
9576
9577
9578      ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
9579
9580 055056 011646      $TRAP2: MOV      (SP),-(SP)        ;;MOVE THE PC DOWN
9581 055060 016666 000004 000002      MOV      4(SP),2(SP)    ;;MOVE THE PSW DOWN
9582 055066 000002      RTI                          ;;RESTORE THE PSW
9583
9584      .SBTTL  TRAP TABLE
9585
9586      ;;THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
9587      ;;BY THE "TRAP" INSTRUCTION.
9588
9589      ;      ROUTINE
9590      ;      -----
9591 055070 055056      $TRPAD: .WORD  $TRAP2              ;;CALL=TYPE      TRAP+1(104401)  TTY TYPEOUT ROUTINE
9592 055072 051667      STYPE      ;;CALL=TYPE      TRAP+1(104401)  TTY TYPEOUT ROUTINE
9593
9594
9595 055074 052750      $PDCHR  ;;CALL=PDCHR      TRAP+2(104402)  TTY TYPEIN CHARACTER ROUTINE
9596 055076 052370      $PDLIN  ;;CALL=PDLIN      TRAP+3(104403)  TTY TYPEIN STRING ROUTINE

```

```

9597          .SBTTL  ASCII MESSAGES
9598
9599 055100 046111 042514 040507 ILLCGM: .ASCII  /ILLEGAL OPERATOR COMMAND /
9600 055106 020114 050117 051105
9601 055114 052101 051117 041440
9602 055127 046517 040515 042116
9603 055130      040
9604 055131      040 037440 005015 ILLCMD: .ASCIZ  / ?/<15><12>
9605 055136      000
9606 055137      104 044522 042526 OPR000: .ASCII  /DRIVE NUMBER /
9607 055144 047040 046525 042502
9608 055152 020122
9609 055154 020060      000 OPR001: .ASCIZ  /0 /
9610 055157      111 046114 043505 OPR002: .ASCIZ  /ILLEGAL?/<15><12>
9611 055164 046101 006477 000017
9612 055172 047125 042504 020122 OPR003: .ASCIZ  /UNDER TEST/
9613 055200 042524 052123      000
9614 055205      101 051114 040505 OPR004: .ASCIZ  /ALREADY ASSIGNED?/<15><12>
9615 055212 054504 040440 051523
9616 055220 043511 042516 037504
9617 055226 005015      000
9618 055231      116 052117 040440 OPR005: .ASCIZ  /NOT ASSIGNED?/<15><12>
9619 055236 051523 043511 042516
9620 055244 037504 005015      000
9621 055251      110 051501 041040 OPR006: .ASCIZ  /HAS BEEN DROPPED FROM TEST SEQUENCE/<15><12>
9622 055256 042505 020116 051104
9623 055264 050117 042520 020104
9624 055272 051106 046517 052040
9625 055300 051505 020124 042523
9626 055306 052521 047105 042503
9627 055314 005015      000
9628 055317      120 042514 051501 OPR007: .ASCIZ  /PLEASE ENTER COMMAND/<15><12>/**
9629 055324 020105 047105 042524
9630 055337 020122 047503 046515
9631 055340 047101 006504 025012
9632 055346      000
9633 055347      117 042520 040522 OPR008: .ASCIZ  /OPERATOR INITIATED DROP DRIVE DURING WRITE PACK/
9634 055354 047524 020127 047111
9635 055362 052111 040511 042524
9636 055370 020104 051104 050117
9637 055376 042040 044522 042526
9638 055404 042040 051125 047111
9639 055412 020107 051127 052111
9640 055420 020105 046520 045503
9641 055426      000
9642 055427      015 042012 044527 OPR010: .ASCII  <15><12>/DRIVE NUMBER /
9643 055434 042526 047040 046525
9644 055442 042502 020122
9645 055446 020060      000 OPR011: .ASCIZ  /0 /
9646 055451      104 044522 042526 OPR012: .ASCIZ  /DRIVE WRITE LOCKED/<15><12>
9647 055456 053440 044522 042524
9648 055464 046040 041517 042513
9649 055472 006504 000012
9650 055476 046101 020114 051104 OPR013: .ASCIZ  /ALL DRIVES CURRENTLY UNDER TEST/<15><12>
9651 055504 053111 051505 041440
9652 055512 051125 042522 052116

```

9653	055520	054514	052440	042116	
9654	055526	051105	052040	051505	
9655	055534	006524	000012		
9656	055540	047516	042040	044522	OPR014: .ASCII /NO DRIVE IN USE/<15><12>
9657	055546	042526	044440	020116	
9658	055554	051525	006505	000012	
9659	055562	046101	044514	047107	OPR015: .ASCIIZ /ALIGNMENT PACK IN DRIVE/<15><12>
9660	055570	042515	052116	050040	
9661	055576	041501	020113	047111	
9662	055604	042040	044522	042526	
9663	055612	005015	000		
9664	055615	052	042052	044522	DRVSTT: .ASCIIZ /**DRIVE STATISTICS**/<15><12>
9665	055622	042526	051440	040524	
9666	055630	044524	052123	041511	
9667	055636	025123	006452	000012	
9668	055644	005015	051104	053111	STT000: .ASCII <15><12>/DRIVE SERIAL NO. /
9669	055652	020105	042523	044522	
9670	055660	046101	047040	077117	
9671	055666	040			
9672	055667	000003			STT100: .BLFB 3
9673	055672	000			.RYTE 0
9674	055673	117	042122	051105	STT001: .ASCIIZ /ORDERS PERFORMED /
9675	055700	020123	042520	043122	
9676	055706	051117	042515	020104	
9677	055714	000			
9678	055715	015	053412	051117	STT002: .ASCIIZ <15><12>/WORDS WRITTEN*65K /
9679	055722	051504	053440	044522	
9680	055730	052124	047105	033052	
9681	055736	045465	000040		
9682	055742	005015	047527	042122	STT003: .ASCIIZ <15><12>/WORDS READ*65K /
9683	055750	020123	042522	042101	
9684	055756	033052	045465	000040	
9685	055764	005015	047523	052106	STT004: .ASCIIZ <15><12>/SOFT DATA ERRORS/<15><12>/ FCC /
9686	055772	042040	052101	020101	
9687	056000	051105	047522	051522	
9688	056006	005015	020040	041505	
9689	056014	020103	000		
9690	056017	015	020012	051040	STT005: .ASCIIZ <15><12>/ REREAD /
9691	056024	051105	040505	020104	
9692	056032	000			
9693	056033	015	020012	047440	STT006: .ASCIIZ <15><12>/ OFFSET /
9694	056040	043106	042523	020124	
9695	056046	000			
9696	056047	015	044012	051101	STT007: .ASCIIZ <15><12>/HARD DATA ERRORS /
9697	056054	020104	040504	040524	
9698	056062	042440	051122	051117	
9699	056070	020123	000		
9700	056073	015	051412	042505	STT008: .ASCIIZ <15><12>/SEEK INCOMPLETES /
9701	056100	020113	047111	047503	
9702	056106	050115	042514	042524	
9703	056114	020123	000		
9704	056117	015	047412	042520	STT009: .ASCIIZ <15><12>/OPERATION INCOMPLETES/MISPOSITIONS "
9705	056124	040522	044524	047117	
9706	056132	044440	041516	046517	
9707	056140	046120	052105	051505	
9708	056146	046457	051511	047520	

9709	056154	044523	044524	047117	
9710	056167	020127	000		
9711	056165	015	047412	044124	STT010: .ASCII7 <15><12>/CTHEM FRPCPS /
9712	056172	051105	042440	051127	
9713	056200	051117	020127	000	
9714	056205	015	050017	041501	STT011: .ASCII7 <15><12>/PACK SERIAL NUMBER /
9715	056212	020113	042523	044522	
9716	056220	046101	047040	046525	
9717	056226	042502	020122		
9718	056237	000017			STT017: .ASCII7 11.
9719	056245	015	000012		STT013: .ASCII7 <15><12>
9720	056250	050103	020125	042111	SYS000: .ASCII7 /CPU ID= /
9721	056256	000075			
9722	056260	053117	051105	040514	SYS001: .ASCII7 /OVERLAY LOADER= /
9723	056266	020131	047514	042101	
9724	056274	051105	000075		
9725	056300	040515	020130	051124	SYS002: .ASCII7 /MAX TRANSFER SIZE (MAX: /
9726	056306	047101	043127	051105	
9727	056314	051440	055111	020105	
9728	056322	020050	040515	035130	
9729	056330	000040			
9730	056337	024440	000075		SYS003: .ASCII7 /)= /
9731	056336	047516	020056	043117	SYS004: .ASCII7 /MC. OF SOFTWARE COMPARIS= /
9732	056344	051440	043117	053524	
9733	056352	051101	020105	047503	
9734	056360	050115	051101	051505	
9735	056366	000075			
9736	056370	045522	033060	041040	SYS005: .ASCII7 /PK06 BPS ADD= /
9737	056376	051525	040440	042104	
9738	056404	000075			
9739	056406	045522	033060	053040	SYS006: .ASCII7 /RK06 VFC ADD= /
9740	056414	041505	040440	042104	
9741	056422	000075			
9742	056424	047516	042516	044530	SYS007: .ASCII7 /NONEXISTENT MEMCOPY/<15><12>
9743	056432	052123	047105	020124	
9744	056440	042515	047515	054522	
9745	056446	005015	000		
9746	056451	127	040505	054504	SYS008: .ASCII7 /READY TO BEGIN PERFORMANCE TESTING/<15><12><12>
9747	056456	052040	020117	042502	
9748	056464	044507	020116	042520	
9749	056472	043127	051117	040515	
9750	056500	041516	020105	042524	
9751	056506	052123	047111	006507	
9752	056514	005012	000		
9753	056517	123	040524	044524	SYS009: .ASCII7 /STATISTIC INTERVAL= /
9754	056524	052127	041511	044440	
9755	056532	052116	051105	040526	
9756	056540	036514	000		
9757	056543	127	030113	020066	SYS010: .ASCII7 /PK06 PPICK= /
9758	056550	051120	047511	036527	
9759	056556	000			
9760	056557	120	051101	042515	PAR000: .ASCII7 /PARAMETERS FOR DRIVE /
9761	056564	042524	051522	043040	
9762	056572	051117	042040	044527	
9763	056600	042526	040		
9764	056603	060	005015	000012	PAR001: .ASCII7 /0/<15><12><12>

9765	056610	054503	044514	042116	PAR002: .ASCIZ	/CYLINDER MAX,MIN =/
9766	056616	051105	046440	054101		
9767	056624	046454	047111	036440		
9768	056632	000				
9769	056633	124	040522	045503	PAR003: .ASCIZ	/TRACK MAX,MIN =/
9770	056640	046440	054101	046454		
9771	056646	047111	036440	000		
9772	056653	123	041505	047524	PAR004: .ASCIZ	/SECTOR MAX,MIN =/
9773	056660	020122	040515	026130		
9774	056666	044515	020116	000075		
9775	056674	042522	042101	053457	PAR005: .ASCIZ	"HEAD/WRITE RATIO ="
9776	056702	044522	042524	051040		
9777	056710	052101	047511	036440		
9778	056716	000				
9779	056717	127	044522	042524	PAR006: .ASCIZ	/WRITE CHECK AFTER WRITE =/
9780	056724	041440	042510	045503		
9781	056732	040440	052106	051105		
9782	056740	053440	044522	042524		
9783	056746	036440	000			
9784	056751	103	051117	042522	PAR007: .ASCIZ	/CORRECTABLE READ ERROR THRESHOLD =/
9785	056756	052103	041101	042514		
9786	056764	051040	040505	020104		
9787	056772	051105	047522	020122		
9788	057000	044124	042522	044123		
9789	057006	046117	020104	000075		
9790	057014	047125	047503	051127	PAR008: .ASCIZ	/UNCORRECTABLE READ ERROR THRESHOLD =/
9791	057022	041505	040524	046102		
9792	057030	020105	042522	042101		
9793	057036	042440	051122	051117		
9794	057044	052040	051110	051505		
9795	057052	047510	042114	036440		
9796	057060	000				
9797	057061	123	042505	020113	PAR009: .ASCIZ	/SEEK ERROR THRESHOLD =/
9798	057066	051105	047522	020127		
9799	057074	044124	042522	044123		
9800	057102	046117	020104	000075		
9801	057110	047111	044510	044502	PAR010: .ASCIZ	/INHIBIT ERROR CORRECTION AND RETRY =/
9802	057116	020124	051105	047527		
9803	057124	020122	047503	051122		
9804	057132	041505	044524	047117		
9805	057140	040440	042116	051040		
9806	057146	052105	054522	036440		
9807	057154	000				
9808	057155	117	042520	040522	PAR011: .ASCIZ	/OPERATION COUNT THRESHOLD*65K =/
9809	057162	044524	047117	041440		
9810	057170	052517	052116	052040		
9811	057176	051110	051505	047510		
9812	057204	042114	033052	045465		
9813	057212	036440	000			
9814	057215	127	051117	051504	PAR012: .ASCIZ	/WORDS TRANSFERRED THRESHOLD*1,049K =/
9815	057222	052040	040522	051516		
9816	057230	042506	051122	042105		
9817	057236	052040	051110	051505		
9818	057244	047510	042114	030452		
9819	057252	030054	034464	020113		
9820	057260	000075				

9821	057262	044506	051522	000124	PAR013: .ASCIZ	/FIRST/
9822	057270	042523	047503	042116	PAR014: .ASCIZ	/SECOND/
9823	057276	000				
9824	057277	124	044510	042122	PAR015: .ASCIZ	/THIRD/
9825	057304	000				
9826	057305	106	052517	052122	PAR016: .ASCIZ	/FOURTH/
9827	057312	000110				
9828	057314	044506	052106	000110	PAR017: .ASCIZ	/FIFTH/
9829	057322	042440	041530	052514	PAR018: .ASCIZ	/ EXCLUDED PACK AREA =/
9830	057330	042504	020104	040520		
9831	057336	045503	040440	042522		
9832	057344	020101	000075			
9833	057350	044103	047101	042507	PAR019: .ASCIZ	/CHANGE/
9834	057356	000				
9835	057357	125	040516	046102	PAR020: .ASCIZ	/UNABLE TO GENERATE NEW COMMAND CODE TO/<15><12>
9836	057364	020105	047524	043440		
9837	057372	047105	051105	052101		
9838	057400	020105	042516	020127		
9839	057406	047503	046515	047101		
9840	057414	020104	052504	020105		
9841	057422	047524	005015			
9842	057426	054105	046103	042125	.ASCIZ	/EXCLUDED PACK AREAS AND MAX TRANSFER SIZE CONFLICT/<15><12>
9843	057434	042105	050040	041501		
9844	057442	020113	051101	040505		
9845	057450	020123	047101	020104		
9846	057456	040515	020130	051124		
9847	057464	047101	043123	051105		
9848	057472	051440	055111	020105		
9849	057500	047503	043116	044514		
9850	057506	052103	005015	000		
9851	057513	123	046501	046120	PAR021: .ASCIZ	/SAMPLED COMPARES =/
9852	057520	042105	041440	046517		
9853	057526	040520	042522	020123		
9854	057534	000075				
9855	057536	020040	037477	037477	QUESMK: .ASCIZ	/ ???? /
9856	057544	037477	020040	000		
9857	057551	040	000040		BLANKS: .ASCIZ	/ /
9858	057554	025057	043052	052101	ERR000: .ASCIZ	/***FATAL ERROR***/<15><12><12>
9859	057562	046101	042440	051122		
9860	057570	051117	025052	006452		
9861	057576	005012	000			
9862	057601	103	047117	051124	ERR001: .ASCIZ	/CONTROLLER CLEAR DTC NOT CLEAR ERROR/
9863	057606	046117	042514	020122		
9864	057614	046103	040505	020122		
9865	057622	044504	020104	047516		
9866	057630	020124	046103	040505		
9867	057636	020122	051105	047522		
9868	057644	000122				
9869	057646	047516	040440	052124	ERR002: .ASCIZ	/NO ATTENTION IN PKASOP/
9870	057654	047105	044524	047117		
9871	057662	044440	020116	045522		
9872	057670	051501	043117	000		
9873	057675	125	042516	050130	ERR003: .ASCIZ	/UNEXPECTED DATA TYPE ERROR/
9874	057702	041505	042524	020104		
9875	057710	040504	040524	052040		
9876	057716	050131	020105	051105		

9877	057724	047522	000122						
9878	057730	051104	053111	020105	ERR004: .ASCIZ	/DRIVE CLEAR DID NOT RESET ATTENTION/			
9879	057736	046103	040505	020122					
9880	057744	044504	020104	047516					
9881	057752	020124	042522	042523					
9882	057760	020124	052101	042524					
9883	057766	052116	047511	000116					
9884	057774	046111	042514	040507	EPR005: .ASCIZ	/ILLEGAL DRIVER COMMAND/			
9885	060002	020114	051104	053111					
9886	060010	051105	041440	046517					
9887	060016	040515	042116	000					
9888	060023	115	046125	044524	EPR006: .ASCIZ	/MULTIPLE DRIVE SELECT/			
9889	060030	046120	020105	051104					
9890	060036	053111	020105	042523					
9891	060044	042514	052103	000					
9892	060051	123	052105	051440	EPR007: .ASCIZ	/SET SWITCH 14 TO CYCLE ON EPRCR/<15><12>			
9893	060056	044527	041524	020110					
9894	060064	032061	052040	020117					
9895	060072	054503	046103	020105					
9896	060100	047117	042440	051122					
9897	060106	051117	005015	000					
9898	060112	103	047117	051124	ERR008: .ASCIZ	/CONTROLLER ERROR THRESHOLD EXCEEDED/			
9899	060120	046060	042514	020122					
9900	060126	051105	047522	020122					
9901	060134	044124	042522	044123					
9902	060142	046117	020104	054105					
9903	060150	042503	042105	042105					
9904	060156	000							
9905	060157	104	052101	020101	ERR009: .ASCIZ	/DATA LATE THRESHOLD EXCEEDED/			
9906	060164	040514	042524	052040					
9907	060172	051110	051505	047510					
9908	060200	042114	042440	041530					
9909	060206	042505	042504	000104					
9910	060214	042101	051104	051505	EPR012: .ASCIZ	/ADDRESS GOOD	RAD	SECTOR	DATA/<15><12>
9911	060222	020123	047507	042117					
9912	060230	020040	020040	040502					
9913	060236	020104	020040	020040					
9914	060244	042523	052103	051117					
9915	060252	020040	040504	040524					
9916	060260	005015							
9917	060267	020040	020040	020040	.ASCIZ	/	DATA	DATA	POS PAT #/<15><17>
9918	060270	070040	040504	040524					
9919	060276	020040	020040	040504					
9920	060304	040524	020040	020040					
9921	060312	047520	020123	020040					
9922	060320	020040	040520	020124					
9923	060326	006443	000012						
9924	060332	047503	052116	047522	ERR013: .ASCIZ	/CONTROLLER EPRCR NOT FLAGGED/<15><12>			
9925	060340	046114	051105	042440					
9926	060346	051122	051117	047040					
9927	060354	052117	043040	040514					
9928	060362	043507	042105	005015					
9929	060370	000							
9930	060371	127	051117	020104	ERR014: .ASCIZ	/WCRD COUNT NOT EQUAL 0/<15><12>			
9931	060376	047503	047125	020124					
9932	060404	047516	020124	050505					

9933	060412	040525	020114	006460		
9934	060420	000012				
9935	060422	052502	020123	042101	ERR015: .ASCIIZ	/BUS ADD INCORRECT/<15><12>
9936	060430	020104	047111	047503		
9937	060436	051122	041505	006524		
9938	060444	000012				
9939	060446	054503	026114	051124	ERR016: .ASCIIZ	/CYL,TRK,SEC INCORRECT/<15><12>
9940	060454	026113	042523	020103		
9941	060462	047111	047503	051122		
9942	060470	041505	006524	000012		
9943	060476	047503	052116	047522	ERR017: .ASCIIZ	/CONTROLLER COMMAND TIME OUT/
9944	060504	046114	051105	041440		
9945	060512	046517	040515	042116		
9946	060520	052040	046511	020105		
9947	060526	052517	000124			
9948	060532	051120	043517	042440	ERR020: .ASCIIZ	/PROC ERROR/
9949	060540	051122	051117	000		
9950	060545	111	046114	043040	ERR021: .ASCIIZ	/ILL FUNCT CODE/
9951	060552	047125	052103	041440		
9952	060560	042117	000105			
9953	060564	106			ERR022: .BYTE	106
9954	060565	105	051122	051117	.ASCIIZ	/ERPOP WHILE WAITING TO REPORT ERROR/
9955	060572	053440	044510	042514		
9956	060600	053440	044501	044524		
9957	060606	043516	052040	020117		
9958	060614	042522	047520	052122		
9959	060622	042440	051122	051117		
9960	060630	000				
9961	060631	106			ERR023: .BYTE	106
9962	060632	047516	026516	054105	.ASCIIZ	/NON-EXIST DRV/
9963	060640	051511	020124	051104		
9964	060646	000126				
9965	060650	106			ERR024: .BYTE	106
9966	060651	127	051105	047507	.ASCIIZ	/SEPCOM PAR/
9967	060656	020116	040520	000122		
9968	060664	106			ERR025: .BYTE	106
9969	060665	125	044516	020124	.ASCIIZ	/UNIT FIELD EPR/
9970	060672	044506	046105	020104		
9971	060700	051105	000122			
9972	060704	005015	047111	020124	ERR026: .ASCIIZ	<15><12>/INT FROM DRV NOT UNDER TEST/<15><12>
9973	060712	051106	046517	042040		
9974	060720	053122	047040	052117		
9975	060726	052440	042116	051105		
9976	060734	052040	051505	006524		
9977	060742	000012				
9978	060744	040510	042122	042040	ERR027: .ASCIIZ	/HARD DRV ERR/<15><12>
9979	060752	053122	042440	051122		
9980	060760	005015	000			
9981	060763	104	053122	051440	ERR028: .ASCIIZ	/DRV STATUS CHANGE DID NOT CLR/<15><12>
9982	060770	040524	052524	020127		
9983	060776	044103	047101	042507		
9984	061004	042040	042111	047040		
9985	061012	052117	041440	051114		
9986	061020	005015	000			
9987	061023	125	042516	050130	ERR030: .ASCIIZ	/UNEXPECTED ATTN/<15><12>
9988	061030	041505	020124	052101		

9989	061036	047124	005015	000		
9990	061047	111	046114	042040	EPR031: .ASCIZ	/ILL DSF ADD/
9991	061050	045523	040440	042104		
9992	061056	000				
9993	061057	127	052122	046040	ERR032: .ASCIZ	/WRT LOCK EPR/
9994	061064	041517	020113	051105		
9995	061072	000127				
9996	061074	041501	046040	053517	EPR033: .ASCIZ	/AC LCM/
9997	061102	000				
9998	061103	123	042520	042105	EPR034: .ASCIZ	/SPEED LOSS/
9999	061110	046040	051517	000123		
10000	061116	051104	020126	054524	ERR035: .ASCIZ	/DRV TYPE EPR/
10001	061124	042520	042440	051122		
10007	061137	000				
10003	061133	106	051117	040515	ERR036: .ASCIZ	/FORMAT ERR/
10004	061140	020124	051105	000122		
10005	061146	047516	026516	054105	ERR037: .ASCIZ	/NON-EXIST DRV FUNCT/
10006	061154	051511	020124	051104		
10007	061162	020126	052506	041516		
10008	061170	000124				
10009	061172	106			ERR038: .BYTE	106
10010	061177	123	042505	020113	.ASCIZ	/SEEK INCOMP/
10011	061200	047111	047503	050115		
10012	061206	000				
10013	061207	106			ERR039: .BYTE	106
10014	061210	051104	020126	043117	.ASCIZ	/DRV OFF TRK/
10015	061216	020106	051124	000113		
10016	061224	106			ERR040: .BYTE	106
10017	061225	115	051511	047520	.ASCIZ	/MISPOS/
10018	061232	000123				
10019	061234	126			ERR042: .BYTE	126
10020	061235	117	042520	040522	.ASCIZ	/OPERAT INCOMP/
10021	061242	020124	047111	047503		
10022	061250	050115	000			
10023	061253	127			ERR043: .BYTE	127
10024	061254	042110	020122	051126	.ASCIZ	/HDR VRC ERR/
10025	061262	020103	051105	000122		
10026	061270	047516	026516	054105	ERR044: .ASCIZ	/NON-EXIST MEM/
10027	061276	051511	020124	042515		
10028	061304	000115				
10029	061306	047125	041111	051525	ERR045: .ASCIZ	/UNIBUS PAK/
10030	061314	050040	051101	000		
10031	061321	117			ERR046: .BYTE	117
10032	061322	051104	020126	044524	.ASCIZ	/DRV TIMING ERR/
10033	061330	044515	043516	042440		
10034	061336	051122	000			
10035	061341	104	052101	020101	EPR047: .ASCIZ	/DATA LATF/<15><12>
10036	061346	040514	042524	005015		
10037	061354	000				
10038	061355	117			ERR049: .BYTE	117
10039	061356	040504	040524	041440	.ASCIZ	/DATA CHK/
10040	061364	045510	000			
10041	061367	107			ERR050: .BYTE	107
10042	061370	051127	052111	020105	.ASCIZ	/WRITE CHK ERR/
10043	061376	044103	020113	051105		
10044	061404	000122				

10045	061406	054503	020114	042101
10046	061414	051104	047440	051126
10047	061422	046106	000127	
10048	061426	106		
10049	061427	103	047115	020104
10050	061434	044524	042515	047440
10051	061442	052125	042040	051125
10052	061450	047111	020107	047520
10053	061456	044523	044524	047117
10054	061464	047111	020107	050117
10055	061472	051105	052101	047511
10056	061500	006516	000012	
10057	061504	106		
10058	061505	104	053122	047040
10059	061512	052117	051040	054504
10060	061520	040440	052106	051105
10061	061526	051440	040524	052122
10062	061534	051440	044520	042116
10063	061542	042514	000	
10064	061545	106		
10065	061546	047526	020114	040526
10066	061554	044514	020104	044504
10067	061562	020104	047516	020124
10068	061570	042523	020124	043101
10069	061576	042524	020122	040520
10070	061604	045503	040440	045503
10071	061612	000		
10072	061613	104	053122	052440
10073	061620	051516	043101	000105
10074	061626	051104	020126	051105
10075	061634	020122	047516	020124
10076	061642	047111	044504	040503
10077	061650	042524	020104	054502
10078	061656	043040	052501	052114
10079	061664	005015	000	
10080	061667	104	053122	040440
10081	061674	052124	020116	052502
10082	061707	020124	047516	043040
10083	061710	052501	052114	047440
10084	061716	020122	051104	020126
10085	061724	052123	052101	051525
10086	061737	041440	040510	043516
10087	061740	006505	000012	
10088	061744	046503	042116	052040
10089	061752	046511	020105	052517
10090	061760	020124	051104	020126
10091	061766	042523	055111	042105
10092	061774	041040	020131	052117
10093	062002	042510	020122	047520
10094	062010	052127	005015	000
10095	062015	105	041503	046440
10096	062022	051511	047503	051127
10097	062030	041505	044524	047117
10098	062036	005015	000	
10099	062041	104	052101	020101
10100	062046	047503	050115	051101

ERR051: .ASCII7 /CYL ADDR OVRFLW/

ERR052: .BYTE 106
.ASCIIZ /CMND TIME OUT DURING POSITIONING OPERATION/<15><12>

ERR053: .BYTE 106
.ASCIIZ /DRV NOT PDY AFTER START SPINDLE/

ERR054: .BYTE 106
.ASCIIZ /VOL VALID DID NOT SET AFTER PACK ACK/

ERR056: .ASCII7 /DRV UNSAFE/

ERR057: .ASCIIZ /DRV ERR NOT INDICATED BY FAULT/<15><12>

ERR058: .ASCIIZ /DRV ATTN BUT NO FAULT OR DRV STATUS CHANGE/<15><12>

ERR059: .ASCII7 /CMND TIME OUT DRV SEIZED BY OTHER PORT/<15><12>

ERR060: .ASCII7 /FCC MISCONFECTILN/<15><12>

ERR062: .ASCII7 /DATA COMPARE ERROR/<15><12>

10101	062054	020105	051105	047522	
10107	062067	006527	000012		
10103	062066	126			ERR064: .BYTE 126
10104	062067	115	051511	047520	.ASCIIZ /WISPOS/
10105	062074	000123			
10106	062076	106			ERR065: .BYTE 106
10107	062077	110	040505	020104	.ASCIIZ /HEAD SELECT ERROR/
10108	062104	042523	042514	052103	
10109	062112	042440	051122	051117	
10110	062120	000			
10111	062121	105	041503	046040	ERR066: .ASCIIZ /ECC LOGIC ERROR/
10112	062126	043517	041511	042440	
10113	062134	051122	051117	000	
10114	062141	122	042513	050103	ERR067: .ASCIIZ /PKECPT PKECPS/<15><12>
10115	062146	020124	051040	042513	
10116	062154	050103	006523	000012	
10117	062162	047527	042122	041440	ERR070: .ASCIIZ /WORD COUNT INVALID/
10118	062170	052517	052116	044440	
10119	062176	053116	046101	042111	
10120	062204	000			
10121	062205	102	051525	040440	ERR071: .ASCIIZ /BUS ADDRESS INVALID/
10122	062217	042104	042522	051523	
10123	062220	044440	053116	046101	
10124	062226	042111	000		
10125	062231	103	046131	047111	ERR072: .ASCIIZ /CYLINDER INVALID/
10126	062236	042504	020122	047111	
10127	062244	040526	044514	000104	
10128	062252	051124	041501	027513	ERR073: .ASCIIZ "TRACK/SECTOR INVALID"
10129	062260	042523	052103	051117	
10130	062266	044440	053116	046101	
10131	062274	042111	000		
10132	062277	106			ERR074: .BYTE 106
10133	062300	051105	047522	020122	.ASCIIZ /ERROR IN DRV DID NOT WFLDAD HEADS/
10134	062306	047111	042040	053122	
10135	062314	042040	042111	047040	
10136	062322	052117	051040	046105	
10137	062330	040517	020104	042510	
10138	062336	042101	000123		
10139	062342	052503	051122	047105	ERR100: .ASCIIZ /CURRENT SUPPLIED/<15><12>
10140	062350	020124	052523	050120	
10141	062356	044514	042105	005015	
10142	062364	046503	042116	020040	.ASCIIZ /CMND CYL TRF SEC OFFSET BUS AD WAD CT/
10143	062372	020040	054503	020114	
10144	062400	020040	020040	051124	
10145	062406	020113	020040	020040	
10146	062414	042523	020103	020040	
10147	062422	020040	043117	051506	
10148	062430	052105	020040	052502	
10149	062436	020123	042101	020040	
10150	062444	051127	020104	052103	
10151	062452	000			
10152	062453	120	042522	044526	ERR101: .ASCIIZ /PREVIOUS /<15><12>
10153	062460	052517	020123	005015	
10154	062466	046503	042116	020040	.ASCIIZ /CMND CYL TRF SEC WAD CT/
10155	062474	020040	054503	020114	
10156	062502	020040	020040	051124	

10157	062510	020113	020040	020040	
10158	062516	042523	020103	070040	
10159	062524	020040	051127	020104	
10160	062532	052103	000		
10161	062535	040	050040	052101	ERR102: .ASCIZ / PAT/
10162	062542	000			
10163	062543	117	043106	042523	ERR108: .ASCIZ /OFFSPT =/
10164	062550	020124	000075		
10165	062554	042110	020061	020040	ERR111: .ASCIZ /WD1 WD2 WD3/<15><12>
10166	062562	020040	042110	020062	
10167	062570	020040	020040	042110	
10168	062576	006463	000012		
10169	062602	040504	040524	046040	ERR200: .ASCIZ /DATA LATF WHEN UNLOADING HPADER/
10170	062610	052101	020105	044127	
10171	062616	047105	052440	046116	
10172	062624	040517	044504	043516	
10173	062632	044040	040505	042504	
10174	062640	000127			
10175	062642	047503	052116	047522	ERR201: .ASCIZ /CONTROLLER ERASE DURING DRIVE SERVICING/
10176	062650	046114	051105	042440	
10177	062656	051122	051117	042040	
10178	062664	051125	047111	020107	
10179	062672	051104	053111	020105	
10180	062700	042523	053122	041511	
10181	062706	047111	000107		
10182	062712	051105	047522	020122	ERR202: .ASCIZ /ERPOP DURING ERROR RECOVERY/<15><12>
10183	062720	052504	044522	043516	
10184	062726	042440	051122	051117	
10185	062734	051040	041505	053117	
10186	062742	051105	006531	000012	
10187	062750	051105	047522	020122	ERR203: .ASCIZ /ERPOP RECOVERY SUCCESSFUL/<15><12>
10188	062756	042522	047503	042526	
10189	062764	054522	051440	041525	
10190	062772	051505	043123	046125	
10191	063000	005015	000		
10192	063003	105	051122	051117	ERR204: .ASCIZ /ERPOP RECOVERY UNSUCCESSFUL/<15><12>
10193	063010	051040	041505	053117	
10194	063016	051105	070131	047125	
10195	063024	052523	042503	051523	
10196	063032	052506	006514	000012	
10197	063040	051105	047522	020122	ERR205: .ASCIZ /ERROR THRESHOLD EXCEEDED/<15><12>
10198	063046	044124	042522	044123	
10199	063054	046117	020104	054105	
10200	063062	042503	042105	042105	
10201	063070	005015	000		
10202	063073	121	042525	052123	ERR206: .ASCIZ /QUESTIONABLE DRIVE STATUS/<15><12>
10203	063100	047511	040516	046102	
10204	063106	020105	051104	053111	
10205	063114	020105	052123	052101	
10206	063122	051525	005015	000	
10207	063127	103	047117	051124	ERR207: .ASCIZ /CONTROLLER TIMED OUT WITH CL SPT/<15><12>
10208	063134	046117	042514	020122	
10209	063142	044524	042515	020104	
10210	063150	052517	020124	044527	
10211	063156	044124	043440	020117	
10212	063164	042523	006524	000012	

10213	063172	051105	047522	020122	ERR208: .ASCII	/ERROR WHILE DIAGNOSING/<15><12>
10714	063200	044127	046111	020105		
10215	063206	044504	043501	047516		
10216	063214	044523	043516	005015		
10217	063222	000				
10718	063223	127			ERR209: .BYTE	127
10219	063224	042510	042101	051105	.ASCII	/HEADER ERROR/<15><12>
10220	063232	042440	051122	051117		
10221	063240	005015	000			
10227	063243	052	075052	044506	ERR210: .ASCII	/***FIRST ERROR INFO***/<15><12>
10223	063250	051522	020124	051105		
10224	063256	047522	020122	047111		
10225	063264	047506	025052	006452		
10726	063777	000012				
10227	063274	045522	047520	020123	ERR211: .ASCII	/RKPOS /
10228	063302	000				
10229	063303	122	050113	052101	ERR212: .ASCII	/RKPAT /
10230	063310	000040				
10231	063312	042101	051104	051505	ERR213: .ASCII	/ADDRESS CP BEGINNING CP SECTION IN FORCE /
10232	063320	020123	043117	041040		
10233	063326	043505	047111	044516		
10234	063334	043516	047440	020106		
10235	063342	042523	052103	051117		
10236	063350	044440	020116	051105		
10237	063356	047522	020122	000		
10738	063361	015	042012	052101	ERR214: .ASCII	<15><12>/DATA READ/<15><12>
10239	063370	020101	042522	042101		
10240	063376	005015	000			
10241	063401	122	041513	030523	ERR300: .ASCII	/RKCS1 RKCS2 RKWCP RKPA RKCA RKDC RKASOP RKRE/<15><12>
10242	063406	070040	051040	041513		
10243	063414	031123	020040	051013		
10244	063422	053513	051103	020040		
10245	063430	051040	041113	020101		
10246	063436	020040	051040	042113		
10247	063444	020101	020040	051040		
10248	063452	042113	020103	020040		
10249	063460	051040	040513	047523		
10250	063466	020106	051040	042513		
10251	063474	006522	000012			
10252	063500	045522	051504	020040	ERR301: .ASCII	/RKDS RKMR1 RKMR2 RKMR3 RKFS RKPAT/<15><12>
10253	063506	020040	045522	051115		
10254	063514	020061	020040	045527		
10255	063522	051115	020062	020040		
10256	063530	045522	051115	020063		
10257	063536	020040	045522	047520		
10258	063544	020123	020040	045527		
10259	063552	040520	006524	000012		
10260	063560	045522	051504	005015	ERR302: .ASCII	/RKDS/<15><12>
10261	063566	000				
10262	063567	015	046412	051505	ERR303: .ASCII	<15><12>/MESS A B/<15><12>/ 00 /
10263	063574	020123	020040	020040		
10264	063602	020040	020101	020040		
10265	063610	020040	020040	006502		
10266	063616	020012	030060	020040		
10267	063624	020040	000040			
10268	063630	005015	030040	020061	ERR304: .ASCII	<15><12>/ 01 /

10269	063636	020040	020040	000	
10270	063643	015	020012	030061	ERR305: .ASCIZ <15><12>/ 10 /
10271	063650	020040	020040	000040	
10272	063656	005015	030440	020061	ERR306: .ASCIZ <15><12>/ 11 /
10273	063664	020040	020040	000	
10274	063671	125	042516	050130	ERR400: .ASCIZ /UNEXPECTED MEMORY PARITY TRAP/
10275	063676	041505	042524	020104	
10276	063704	042515	047515	054522	
10277	063712	050040	051101	052111	
10278	063720	020131	051124	050101	
10279	063726	000			
10280	063727	124	046511	035105	TIM000: .ASCII /TIME:/
10281	063734	020040	035040	020040	TIM001: .ASCIZ / : : /<15><17>
10282	063747	020072	006440	000012	
10283					.EVEN

```

10284          .SBTTL  *** DUMP MEMORY ***
10285
10286          .EVEN
10287 063750 000040          .BLKW 40          ;RESERVE SPACE FOR STACK
10288 064050 000000          ..STCK: .WORD 0          ;STORED STACK POINTER
10289
10290 064052 012737 000340 177776 ..DUMP: MOV #PR7,PS          ;LOCK OUT ALL INTERRUPTS
10291 064060 010637 064050          MOV SP,..STCK          ;STORE STACK POINTER
10292 064064 012706 064050          MOV #..STCK,SP          ;LOAD STACK POINTER
10293 064070 010546          MOV R5,-(SP)          ;;PUSH R5 ON STACK
10294 064072 010446          MOV R4,-(SP)          ;;PUSH R4 ON STACK
10295 064074 010346          MOV R3,-(SP)          ;;PUSH R3 ON STACK
10296 064076 010246          MOV R2,-(SP)          ;;PUSH R2 ON STACK
10297 064100 010146          MOV R1,-(SP)          ;;PUSH R1 ON STACK
10298 064102 010046          MOV R0,-(SP)          ;;PUSH R0 ON STACK
10299 064104 013705 000224          MOV ..LOW,R5          ;STORE FIRST ADDRESS
10300 064110 004737 064442          JSR PC,..CR          ;TYPE <CR><LF>
10301 064114 004037 064522          JSR R0,..ASCII          ;TYPE R0 =
10302 064120 064630          ..R0
10303 064122 011646          MOV (SP),-(SP)
10304 064124 004737 064546          JSR PC,..OCT
10305 064130 004737 064442          JSR PC,..CR
10306 064134 004037 064522          JSR R0,..ASCII          ;TYPE R1 =
10307 064140 064643          ..R1
10308 064142 016646 000002          MOV 2(SP),-(SP)
10309 064146 004737 064546          JSR PC,..OCT
10310 064152 004737 064442          JSR PC,..CR
10311 064156 004037 064522          JSR R0,..ASCII          ;TYPE R2 =
10312 064162 064656          ..R2
10313 064164 016646 000004          MOV 4(SP),-(SP)
10314 064170 004737 064546          JSR PC,..OCT
10315 064174 004737 064442          JSR PC,..CR
10316 064200 004037 064522          JSR R0,..ASCII          ;TYPE R3 =
10317 064204 064671          ..R3
10318 064206 016646 000006          MOV 6(SP),-(SP)
10319 064212 004737 064546          JSR PC,..OCT
10320 064216 004737 064442          JSR PC,..CR
10321 064222 004037 064522          JSR R0,..ASCII          ;TYPE R4 =
10322 064226 064704          ..R4
10323 064230 016646 000010          MOV 10(SP),-(SP)
10324 064234 004737 064546          JSR PC,..OCT
10325 064240 004737 064442          JSR PC,..CR
10326 064244 004037 064522          JSR R0,..ASCII          ;TYPE R5 =
10327 064250 064717          ..R5
10328 064252 016646 000012          MOV 12(SP),-(SP)
10329 064256 004737 064546          JSR PC,..OCT
10330 064262 004737 064442          JSR PC,..CR
10331 064266 004037 064522          JSR R0,..ASCII          ;TYPE SP =
10332 064272 064737          ..SP
10333 064274 013746 064050          MOV ..STCK,-(SP)
10334 064300 004737 064546          JSR PC,..OCT
10335 064304 004737 064442          JSR PC,..CR
10336 064310 004737 064442          JSR PC,..CR
10337 064314 004037 064522          JSR R0,..ASCII          ;TYPE HEADER
10338 064320 064745          ..ADD
10339 064322 004737 064442          JSR PC,..CR

```

10340	064326	004737	064442		JSR	PC,..CR	
10341							
10342	064332	010546		10\$:	MOV	R5,-(SP)	;TYPE ADDRESS
10343	064334	004737	064546		JSR	PC,..OCT	
10344	064340	012703	000004		MOV	#4,R3	;TYPE 4 LOCATIONS PPR LINE
10345	064344	004037	064522	12\$:	JSR	R0,..ASCII	
10346	064350	064770			..BLNK		
10347	064352	012546			MOV	(R5)+,-(SP)	;TYPE CONTENTS
10348	064354	004737	064546		JSR	PC,..OCT	
10349	064360	005303			DEC	R3	;CHECK IF START OF NEXT LINE
10350	064362	001370			BNE	12\$	
10351	064364	004737	064442		JSR	PC,..CR	
10352	064370	005777	114544		TST	@SMR	;CHECK IF STOP PRINT OUT
10353	064374	100001			BPL	15\$	
10354	064376	000000			HALT		
10355	064400	023705	000226	15\$:	CMP	..HIGH,R5	;CHECK IF FINISHED
10356	064404	103352			BNE	10\$	
10357	064406	004737	064442		JSR	PC,..CR	
10358	064412	004737	064442		JSR	PC,..CR	
10359	064416	012600			MOV	(SP)+,R0	;POP STACK INTO R0
10360	064420	012601			MOV	(SP)+,R1	;POP STACK INTO R1
10361	064422	012602			MOV	(SP)+,R2	;POP STACK INTO R2
10362	064424	012603			MOV	(SP)+,R3	;POP STACK INTO R3
10363	064426	012604			MOV	(SP)+,R4	;POP STACK INTO R4
10364	064430	012605			MOV	(SP)+,R5	;POP STACK INTO R5
10365	064432	013706	064050		MOV	..STCK,SP	;RESTORE STACK POINTER
10366	064436	000000			HALT		
10367	064440	000776			BR	..-2	
10368							
10369	064442	105777	114502	..CR:	TSTR	@STPS	;WAIT FOR READY
10370	064446	100375			BPL	..CR	
10371	064450	112777	000015 114474		MOVB	#15,@STPB	;LOAD <CR>
10372	064456	105777	114466	5\$:	TSTR	@STPS	;WAIT FOR READY
10373	064462	100375			BPL	5\$	
10374	064464	112777	000012 114460		MOVB	#12,@STPB	;LOAD <LF>
10375	064472	113701	001155		MOVB	\$FILLS,R1	;LOAD FILL COUNT
10376	064476	001410			BEQ	20\$	;RETURN IF NO FILLS
10377	064500	105777	114444	10\$:	TSTR	@STPS	;WAIT FOR READY
10378	064504	100375			BPL	10\$	
10379	064506	113777	001154 114436		MOVB	\$NULL,@STPB	;LOAD FILL CHARACTER
10380	064514	005301			DEC	R1	;CHECK IF FILL COMPLETE
10381	064516	001370			BNE	10\$	
10382	064520	000207		20\$:	RTS	PC	;RETURN
10383							
10384	064522	012001		..ASCII:	MOV	(R0)+,R1	;STORE ADDRESS OF ASCII
10385							
10386	064524	105711		1\$:	TSTR	(R1)	;CHECK IF FINISHED
10387	064526	001406			BEQ	20\$	
10388	064530	105777	114414	2\$:	TSTR	@STPS	;WAIT FOR READY
10389	064534	100375			BPL	2\$	
10390	064536	112177	114410		MOVB	(R1)+,@STPB	;TYPE CHARACTER
10391	064542	000770			BR	1\$	
10392							
10393	064544	000700		20\$:	RTS	R0	;RETURN
10394							
10395	064546	012701	000006	..OCT:	MOV	#6,R1	;LOAD DIGIT COUNT INTO R1

```

10396 064552 005002          CLK      R2          ;CLEAR R2 FOR OUTPUT
10397 064554 000407          BR       10$        ;GET LEADING DIGIT
10398
10399 064556 005002          5$:      CLR      R2          ;GET OCTAL DIGIT
10400 064560 006366 000002          ASL      2(SP)
10401 064564 006102          ROL      R2
10402 064566 006366 000002          ASL      2(SP)
10403 064577 006102          ROL      R2
10404 064574 006366 000002          10$:    ASL      2(SP)
10405 064600 006102          ROL      R2
10406 064602 052702 000060          RLS      #60,R2          ;MAKE IT ASCII
10407 064606 105777 114336          15$:    TSTB    #STPS          ;WAIT FOR READY
10408 064612 100775          BPL      15$
10409 064614 110277 114332          MOVB    R7,#STPB          ;TYPE OCTAL DIGIT
10410 064620 005301          DFC      R1          ;CHECK IF WHOLE WORD PRINTED
10411 064622 001355          BNE      5$
10412 064624 012616          MOV      (SP)+,(SP)          ;ADJUST RETURN
10413 064626 000207          RTS      PC          ;RETURN
10414
10415 064630 030122 036440 020040  ..R0:   .ASCIZ  /P0 =      /
10416 064636 020040 020040 000          .EVFN
10417 064642 122 070061 070075  ..R1:   .ASCIZ  /R1 =      /
10418 064650 020040 020040 000040
10419 064656 031122 036440 020040  ..R2:   .ASCIZ  /R2 =      /
10420 064664 020040 020040 000
10421 064671 122 070063 020075  ..R3:   .ASCIZ  /P3 =      /
10422 064676 020040 020040 000040
10423 064704 032122 036440 020040  ..R4:   .ASCIZ  /P4 =      /
10424 064712 020040 020040 000
10425 064717 122 020065 020075  ..R5:   .ASCIZ  /R5 =      /
10426 064724 020040 020040 000040
10427 064732 050123 036440 020040  ..SP:   .ASCIZ  /SP =      /
10428 064740 020040 020040 000
10429 064745 101 042104 042527  ..ADD:  .ASCIZ  /ADDRESS  CONTENTS/
10430 064752 051523 020040 041440
10431 064760 047117 042524 052116
10432 064766 000123
10433 064770 020040 070040 000  ..BLNK:  .ASCIZ  /      /
10434 064776
10435 064776
10436 065202
10437 064776 044124 020105 045527  ..EVFN
10438 065004 033060 045455 046440  HDBUFF=.
10439 065012 051525 020124 042502  BUFADD=+.137.
10440 065020 050040 047522 042520  HELP:   .ASCII  /THE PK06-K MUST BE PROPERLY FORMATTED 22 SECTIONS PER TRACE/<15><12>
10441 065026 046122 020131 047506
10442 065034 046522 052101 042105
10443 065042 031040 020062 042523
10444 065050 052103 051117 020127
10445 065056 042520 020122 051124
10446 065064 041501 006513 012
10447 065071 124 020117 042502          .ASCII  /TC BEGYN PERFORMANCE TESTING TYPE CONTROL-C <^C>/<15><12>
10448 065076 044507 020116 042520
10449 065104 043122 051117 040515
10450 065112 041516 020105 042524
10451 065120 052123 047111 020107

```

10457	065126	054524	042520	041440
10453	065134	047117	051124	046117
10454	065142	041455	036040	041536
10455	065150	006476	012	
10456	065153	114	043505	046101
10457	065160	041440	046517	040515
10458	065166	042116	020123	051101
10459	065174	035105	005015	012
10460	065201	124	020116	020055
10461	065206	052040	051505	020124
10462	065214	051104	053111	020105
10463	065222	006516	012	
10464	065225	127	020116	020055
10465	065232	053440	044522	042524
10466	065240	050040	041501	026113
10467	065246	042526	043122	020131
10468	065254	040520	045503	020054
10469	065262	047101	020104	042524
10470	065270	052123	042040	044522
10471	065276	042526	047040	005015
10472	065304	047123	026440	020040
10473	065312	042507	020124	052123
10474	065320	052101	051511	044524
10475	065326	051503	047440	020116
10476	065334	051104	053111	020105
10477	065342	006516	012	
10478	065345	104	020116	020055
10479	065352	042040	047522	020120
10480	065360	051104	053111	020105
10481	065366	020116	047101	020104
10482	065374	042507	020124	052123
10483	065402	052101	051511	044524
10484	065410	051503	005015	012
10485	065415	116	041440	047101
10486	065422	041040	020105	026460
10487	065430	020067	051117	040440
10488	065436	005015	012	
10489	065441	104	052101	020101
10490	065446	040520	052124	051105
10491	065454	051516	046440	051525
10492	065462	070124	042502	053440
10493	065470	044522	052124	047105
10494	065476	041040	020131	044124
10495	065504	020105	054105	051105
10496	065512	044503	042523	006527
10497	065520	012		
10498	065521	102	043105	051117
10499	065526	020105	042520	043122
10500	065534	051117	040515	041516
10501	065542	020105	042524	052123
10502	065550	047111	006507	005012
10503	065556	025052	053452	051101
10504	065564	044516	043516	025052
10505	065572	020052	043111	050040
10506	065600	047522	042503	051523
10507	065606	051117	044040	046101

.ASCII /LEGAL COMMANDS ARE:/<15><12><12>

.ASCII /TM - TEST DRIVE M/<15><12>

.ASCII /WM - WRITE PACK, VERIFY PACK, AND TEST DRIVE M/<15><12>

.ASCII /SM - GET STATISTICS ON DRIVE M/<15><12>

.ASCII /DM - DROP DRIVE M AND GET STATISTICS/<15><12><12>

.ASCII /M CAN BE 0-7 OR A/<15><12><12>

.ASCII /DATA PATTERNS MUST BE WRITTEN BY THE EXERCISER/<15><12>

.ASCII /BEFORE PERFORMANCE TESTING/<15><12><12>

.ASCII /***WARNING*** IF PROCESSOR HALT DURING DATA TRANSFER/<15><12>

10508	065614	020124	052504	044522
10509	065622	043516	042040	052101
10510	065630	020101	051124	047101
10511	065636	043123	051105	005015
10512	065644	020040	020040	052040
10513	065652	020117	051104	053111
10514	065660	026105	041040	042101
10515	065666	042440	041503	046440
10516	065674	054501	041040	020105
10517	065702	051127	052111	042524
10518	065710	020116	047117	050040
10519	065716	041501	027113	005015
10520	065724	020040	020040	042040
10521	065732	044522	042526	020123
10522	065740	052515	052123	041040
10523	065746	020105	051104	050117
10524	065754	042520	020104	054502
10525	065762	050040	047522	051107
10526	065770	046501	047440	020122
10527	065776	044527	044124	042040
10528	066004	047522	020120	047503
10529	066012	046515	047101	006504
10530	066020	005012	000	
10531		000001		

.ASCII / TO DRIVE, BAD ECC MAY BE WRITTEN ON PACK./<15><12>

.ASCIIZ / DRIVES MUST BE DROPPED BY PROGRAM OR WITH DRCP COMPARE/<15><12><12>

.END

ABASE = 177440
APNORM 024766
ABN000 025054
APN001 025114
ACDW1 = 000000
ACDW2 = 000000
ACLO = 000010
ACPUOP= 000000
ADDW0 = 000000
ADDW1 = 000000
ADDW10= 000000
ADDW11= 000000
ADDW12= 000000
ADDW13= 000000
ADDW14= 000000
ADDW15= 000000
ADDW2 = 000000
ADDW3 = 000000
ADDW4 = 000000
ADDW5 = 000000
ADDW6 = 000000
ADDW7 = 000000
ADDW8 = 000000
ADDW9 = 000000
ADEVCT= 000000
ADEVM = 000000
AENV = 000000
AENVH = 000000
AFATAL= 000000
AMADR1= 000000
AMADR2= 000000
AMADR3= 000000
AMADR4= 000000
AMAWS1= 000000
AMAWS2= 000000
AMAWS3= 000000
AMAWS4= 000000
AMSGAD= 000000
AMSGLG= 000000
AMSGTY= 000000
AMTYP1= 000000
AMTYP2= 000000
AMTYP3= 000000
AMTYP4= 000000
APASS = 000000
APRIOR= 000240
APTCSD= 000040
APTENV= 000001
APTSIZ= 000200
APTSPD= 000100
ASNORM 013370
ASN1\$ 013446
ASN2\$ 013500

ASN3\$ 013530
ASN4\$ 013540
ASMREG= 000000
ATESTN= 000000
AUNIT = 000000
AUSNR = 000000
AVATLQ 001314
AVECT1= 120210
AVECT2= 000000
A.ABNL 001454
A.COMT 001456
A.WORM 001452
BAI = 000020
BA16 = 000400
BA17 = 001000
BDSECT 041560
BINOCT 052144
BIT0 = 000001
BIT00 = 000001
BIT01 = 000002
BIT02 = 000004
BIT03 = 000010
BIT04 = 000020
BIT05 = 000040
BIT06 = 000100
BIT07 = 000200
BIT08 = 000400
BIT09 = 001000
BIT1 = 000002
BIT10 = 002000
BIT11 = 004000
BIT12 = 010000
BIT13 = 020000
BIT14 = 040000
BIT15 = 100000
BIT2 = 000004
BIT3 = 000010
BIT4 = 000020
BIT5 = 000040
BIT6 = 000100
BIT7 = 000200
BIT8 = 000400
BIT9 = 001000
BLANKS 057551
BPTVEC= 000014
BSBA 001656
BSCVLN 001652
BSE = 000200
BSSFCT 001650
BSTRAN 041344
BSTRCK 001651
BSWC 001654
BSWORD 001660

BUFADD= 065202
BUFER1 022760
BUFFAL 022114
BUPREL 022316
BWAITQ 001320
B.BA16= 000001
B.BA17= 000002
B.CDT = 000004
B.CPMT= 000020
CALSEC 035014
CALWC 036412
CCLR = 100000
CDT = 002000
CERR = 100000
CPMT = 010000
CHKADD 024566
CHKCLK 042574
CINITQ 001324
CLEAR = 000105
CLKFLC 001374
CLKFRQ 001353
CLKINT 042666
CLOCK 042760
CLR = 000040
CMDTO = 000100
CMPHUP 023036
CNTCNT 001624
CNTRL1 026426
CNE = 001000
CONCLR= 000176
CONERR 036524
CONHUN 043234
CONTKL 026022
CPUID 001364
CP = 000015
CPLP = 000200
CTO = 004000
C.INIT 047216
C.OPT 051240
C.RTRN 050664
C.SPEC 047746
DATERR 025100
DCK = 100000
DCLO = 000020
DDISP = 177570
DDT = 000400
DECRIN 053452
DESL = 000010
DETOFF 036730
DI = 040000
DISPLA 001142
DISPRE 000174
DLT = 100000

DLTCNT 001622
DND = 000040
DRA = 000001
DRDY = 000200
DROP 012716
DPCPDV 012172
DPOT = 000040
DRPAR = 000010
DRPDIV= 002000
DRVCHT 001626
DRVDSG= 000040
DRVHRD= 000020
DRVSR= 000007
DRVPTD= 000004
DRVPCS= 000002
DRVSTT 055615
DRVSZD= 010000
DRVUSE= 000001
DRY = 000200
DR.CLP= 000005
DR.SEL= 000001
DSC = 040000
DSNR = 177570
DTBAIT= 100000
DTE = 010000
DTYE = 000040
D.AWCF= 000000
D.CERT= 000012
D.CTRN= 077777
D.CTRL= 177770
D.MWCL= 000000
D.MWSC= 000000
D.MWTR= 000000
D.MXCL= 000632
D.MXSC= 000025
D.MXTR= 000007
D.FATE= 000003
D.SKEY= 000005
D.UENT= 000005
D.WTHI= 077777
D.WTLO= 177770
ECCCOR 025634
ECCPAT 001616
ECCW = 020000
ECH = 000100
EC3\$ 037526
EC4\$ 037752
EC5\$ 040056
EC6\$ 040607
EC7\$ 041110
EMTVEC= 000030
EPCONT 001560
EPCVY 031224

EPCVY1 037464
EPCVY2 042262
EPRPAC 001562
EPRPAC 001330
EPRVEC= 000004
E55000 057554
EPR001 057601
EPR002 057646
E55003 057675
E55004 057730
EPR005 057774
E55006 060023
EPR007 060051
EPR008 060112
E55009 060157
EPR012 060214
EPR013 060327
E55014 060371
EPR015 060422
EPR016 060446
EPR017 060476
EPR020 060532
EPR021 060545
EPR022 060564
EPR023 060631
EPR024 060650
EPR025 060664
E55026 060764
EPR027 060744
EPR028 060762
E55030 061023
E55031 061043
EPR032 061057
E55033 061074
EPR034 061103
EPR035 061116
E55036 061133
E55037 061146
E55038 061177
E55039 061267
E55040 061224
E55042 061234
E55043 061252
EPR044 061270
EPR045 061306
EPR046 061321
E55047 061341
E55048 061355
EPR050 061367
E55051 061406
EPR052 061426
EPR053 061564
E55054 061545

ERR056	061613	E.NOAT=	000002	I.HDAL	044400	OPISTT	030570	PAR014	057270
ERR057	061626	E.SCLR=	000040	I.INTP	043470	OPRCMD	011526	PAR015	057277
ERR058	061667	E.UATT=	000004	I.ISRL	001502	CPR000	055137	PAR016	057305
ERR059	061744	E.UDAT=	000010	I.ISSN	046314	OPR001	055154	PAR017	057314
ERR060	062015	E.UNLD=	020000	I.I00	043777	CPR002	055157	PAR018	057327
ERR062	062041	E0S	032170	I.RTRN	046214	CPR003	055172	PAR019	057350
ERR064	062066	E1S	032332	I.STAT	046770	OPR004	055205	PAR020	057357
ERR065	062076	E10S	034512	I.STOR	046536	OPR005	055231	PAR021	057357
ERR066	062121	E2S	033034	I.UNLD	045722	CPR006	055251	PAR022	057357
ERR067	062141	E3S	033372	LDDATA	022532	OPR007	055317	PAR023	057357
ERR070	062162	E4S	033530	LDEXAR	020166	OPR008	055347	PAR024	057357
ERR071	062205	E5S	034740	LF	= 000012	CPR010	055427	PAR025	057357
ERR072	062231	FBLKC	001664	MAIN	011270	OPR011	055446	PAR026	057357
ERR073	062252	FBLKT	001666	MAXBUF	001354	OPR012	055451	PAR027	057357
ERR074	062277	FLAG	001361	MCLK	= 000400	OPR013	055476	PAR028	057357
ERR100	062342	FMTE	= 000020	MDS	= 001060	OPR014	055540	PAR029	057357
ERR101	062453	GENERT	020640	MEMRAS	= 172100	OPR015	055562	PAR030	057357
ERR102	062535	GETBUF	022016	MEMERR	051642	OP	= 000200	PAR031	057357
ERR103	062543	GETCUR	036150	MEMVEC	= 000114	CVLYLD	001365	PAR032	057357
ERR111	062554	GO	= 000001	MERD	= 001000	O.QVER	001507	PAR033	057357
ERR200	062602	HCRC	= 000400	MESMSK	= 000017	O.RTRN	051620	PAR034	057357
ERR201	062642	HDBUFF=	064776	MEMD	= 002060	O.WAIT	001462	PAR035	057357
ERR202	062712	HDR.AD	001476	MIND	= 000200	PACF	= 000103	PAR036	057357
ERR203	062750	HDR.CT	001500	MINUTE	001370	PARCYL	001576	PAR037	057357
ERR204	063003	HEAD1	001642	MSP	= 000100	PARM	015756	PAR038	057357
ERR205	063040	HEAD2	001644	M.CADD=	017760	PARMDV	012062	PAR039	057357
ERR206	063073	HEAD3	001646	M.CDIF=	017760	PARM0	001732	PAR040	057357
ERR207	063127	HELP	064776	M.DRV	= 000007	PARM1	002210	PAR041	057357
ERR208	063172	HLT000	030472	M.HEAD=	007000	PARM10	004512	PAR042	057357
ERR209	063227	HOOR	001366	M.ID	= 000003	PARM2	002466	PAR043	057357
ERR210	063243	HT	= 000011	M.PAR	= 100000	PARM3	002744	PAR044	057357
ERR211	063274	HVRC	= 000400	M.SECT=	000760	PARM4	003222	PAR045	057357
ERR212	063303	HZ	001352	M.SER	= 077770	PARM5	003500	PAR046	057357
ERR213	063312	H.HEAD	001503	M.D	= 010060	PARM6	003756	PAR047	057357
ERR214	063363	IDAE	= 002000	MEN	= 004060	PARM7	004234	PAR048	057357
ERR300	063401	IE	= 000100	MERSTT	070522	PARSEC	001606	PAR049	057357
ERR301	063500	ILC	= 000001	MOCHK	= 000400	PARSET	006636	PAR050	057357
ERR302	063560	ILF	= 000004	MODSC	= 004000	PARTRK	001602	PAR051	057357
ERR303	063567	ILLCMD	055131	NORMAL	023676	PAR.EN=	000001	PAR052	057357
ERR304	063630	ILLCOM	055100	NORM1	024272	PAR000	056557	PAR053	057357
ERR305	063643	INTMSK	001510	NORM2	024330	PAR001	056603	PAR054	057357
ERR306	063656	INTR	= 000300	NXF	= 000004	PAR002	056610	PAR055	057357
ERR400	063671	IOTVEC=	000020	OCTRIN	053316	PAR003	056633	PAR056	057357
EXAPEA	001612	IP	= 000100	OCTSTG	052240	PAR004	056653	PAR057	057357
E.CCLP=	000001	I.AERP	045452	OFFM12=	000260	PAR005	056674	PAR058	057357
E.CERR=	001000	I.ATTN	044740	OFFM4	= 000220	PAR006	056717	PAR059	057357
E.CLAT=	000020	I.CCLP	046232	OFFM8	= 000240	PAR007	056751	PAR060	057357
E.CMTD=	040000	I.CSTS	046664	OFFP12=	000060	PAR008	057014	PAR061	057357
E.CONT	001460	I.CST1	046706	OFFP4	= 000020	PAR009	057061	PAR062	057357
E.DLT	= 000400	I.DUAL	046026	OFFP8	= 000040	PAR010	057110	PAR063	057357
E.DPAR=	002000	I.EPR	044154	OFFSET=	000115	PAR011	057155	PAR064	057357
E.ILLD=	000100	I.EPKA	044134	OPST	= 000004	PAR012	057215	PAR065	057357
E.MDS	= 100000	I.EPRC	044142	OPI	= 020000	PAR013	057262	PAR066	057357

PH0 = 000000	P.LDPT= 000144	P5\$ 020022	RTV3CH 001640	SW04 = 000020
PR1 = 000040	P.LSEC= 000140	P6\$ 020026	RTVSEC 001634	SW05 = 000040
PR2 = 000100	P.LTRK= 000141	P7\$ 020034	RTVTFW 001635	SW06 = 000100
PR3 = 000140	P.LWC = 000142	P8\$ 020042	RTVMC 001632	SW07 = 000200
PP4 = 000200	P.MNCL= 000066	P9\$ 020050	R6 = 000000	SW08 = 000400
PR5 = 000240	P.MWSC= 000074	QWESW= 000000	R7 = 000000	SW09 = 001000
PR6 = 000300	P.MNTR= 000072	QUESW= 057536	SAVSW= 001404	SW1 = 000002
PP7 = 000340	P.MXCD= 000106	Q.INIT= 040000	SCLR = 000040	SW10 = 002000
PS = 177776	P.MXCL= 000070	Q.POP 050766	SECOND 001372	SW11 = 004000
PSW = 177776	P.MXSC= 000075	Q.PUSH 050706	SEK = 000117	SW12 = 010000
PWRSRT 007150	P.MXTR= 000073	Q.RMOV 051126	SELDIV= 000101	SW13 = 020000
PWRVEC= 000024	P.MXWT= 000112	Q.SRCH 051042	SFI = 000002	SW14 = 040000
P.ASOF= 000032	P.NER = 000206	RDALHD= 000164	SFISTT 030554	SW15 = 100000
P.ASSN= 000210	P.NODR= 000152	RDCMR = 104402	SOPCMP 001362	SW2 = 000004
P.AWCF= 000077	P.NOPI= 000204	RDDATA= 000121	SPAR = 020000	SW3 = 000010
P.A00 = 000040	P.NRD = 000164	RDGATE= 100000	SPDLSE= 000020	SW4 = 000020
P.A01 = 000044	P.NSKI= 000202	RDHEAD= 000125	SPTSPL= 000111	SW5 = 000040
P.A10 = 000050	P.NWRT= 000156	RDLIN = 104402	STACK = 001100	SW6 = 000100
P.A11 = 000054	P.OFST= 000006	RNSTAT= 000141	START 006656	SW7 = 000200
P.BANI= 000007	P.PKSP= 000226	RDY = 000200	STARTS 007240	SW8 = 000400
P.BALO= 000010	P.PRST= 000014	RFCAL = 000112	STAT 012777	SW9 = 001000
P.BAR = 000024	P.QLNK= 000064	RPLEAS= 000140	STATDV 012352	SY000 056250
P.BFCP= 000222	P.RATE= 000076	RESTRY 006646	STATIS 001627	SY001 056260
P.BUFF= 000145	P.RRAH= 000134	RESTO 007614	STAT00 013004	SY002 056300
P.B00 = 000042	P.RRAL= 000132	REST1 011214	STELMT= 177774	SY003 056332
P.B01 = 000046	P.RCMD= 000123	REST2 011230	STT000 055644	SY004 056336
P.B10 = 000052	P.RCYL= 000124	RESVEC= 000010	STT001 055672	SY005 056370
P.B11 = 000056	P.RDPT= 000122	RKASOF= 000016	STT002 055715	SY006 056400
P.CERT= 000100	P.RECT= 000146	RKBA = 000004	STT003 055742	SY007 056424
P.CMLG= 000220	P.RERD= 000147	RKBAS 001444	STT004 055764	SY008 056451
P.CMND= 000001	P.RSEC= 000126	RVCSS1 = 000000	STT005 056017	SY009 056517
P.CS1 = 000016	P.RTRK= 000127	RVCSS2 = 000010	STT006 056032	SY010 056543
P.CS1H= 000007	P.RWC = 000130	RFDA = 000006	STT007 056047	S.ACLC= 000100
P.CS2 = 000020	P.SECC= 000176	RFDH = 000024	STT008 056072	S.BRHH= 000100
P.CVLM= 000002	P.SFCT= 000004	RFDG = 000020	STT009 056117	S.BRKE= 040000
P.DCYL= 000030	P.SEEK= 000211	RKDCYL= 000020	STT010 056165	S.CART= 000400
P.DPAT= 000121	P.SERL= 000224	RKDS = 000012	STT011 056205	S.DCLC= 010000
P.DRYN= 000000	P.SKET= 000104	RFECP5= 000030	STT012 056232	S.EID = 002000
P.DS = 000036	P.SMPL= 000116	RFECP7= 000032	STT013 056245	S.DOOD= 000200
P.DSTT= 000150	P.SOFF= 000174	RKER = 000014	STT15 056602	S.DRA = 000400
P.DTS = 000026	P.SRRD= 000172	RKMP1 = 000026	STT100 055667	S.EFCT= 020000
P.ECMP= 000117	P.TPCK= 000005	RKMP2 = 000034	STT25 030626	S.DRY = 000200
P.EPAT= 000062	P.UERT= 000102	RKMP3 = 000036	STT55 050666	S.ESC = 040000
P.EPOS= 000060	P.WC = 000012	RFPAT = 000032	SUBCLF= 000177	S.FLT = 000200
P.ER = 000034	P.WCR = 000022	RKPOS = 000030	SURECY 035500	S.FCRM= 001000
P.ERAD= 000217	P0\$ 015762	RKPRI 001450	SVAL = 100000	S.FWD = 002000
P.EPHD= 000214	P1\$ 017756	RKVEC 001446	SWR 001140	S.HDPL= 000200
P.ERR = 000212	P10\$ 020054	RKWC = 000002	SWREC 001176	S.HCHN= 000040
P.EXAR= 000232	P11\$ 020070	RLS = 000010	SW0 = 000001	S.ICYL= 000040
P.HARD= 000200	P12\$ 020104	RSTAT 036200	SW00 = 000001	S.IIF = 000400
P.IERC= 000120	P2\$ 017770	RTYBA 001630	SW01 = 000002	S.LIME= 020000
P.LCMD= 000135	P3\$ 020002	RTYCMD 001641	SW02 = 000004	S.LCAC= 010000
P.LCYL= 000136	P4\$ 020014	RTYCYL 001636	SW03 = 000010	S.PND = 000400

S.MMOV= 010000	T.CONT= 000012	SHASE 001244	SHIFTS 001000	SEB2D 054302
S.OFF = 002000	T.CS1 001406	SHDADR 001122	SHINUP 054276	SEETOP= 000014
S.PAR = 001000	T.CS2 001410	SHDDAT 001126	SHIOCT 053450	SEIZE 053624
S.PIP = 020000	T.DA 001416	SPELL 001160	SICNT 001104	SETUP = 177777
S.PLO = 004000	T.DB 001442	SB2D 054406	SINTAC 001135	SSNP = 162000
S.REV = 004000	T.DC 001420	SCDW1 001250	SYTEND 001114	SSNREC 001212
S.RTZ = 020000	T.DLT = 001000	SCDW2 001252	SLF 001166	STESTN 001174
S.SECT= 000020	T.DS 001426	SCHARC 052140	SLFLC 055031	STKR 001146
S.SKI = 002000	T.ER 001424	SCMTAG 001100	SLKCSB 001236	STKINT 052717
S.SPIW= 010000	T.MR1 001430	SCM3 = 000000	SLKCSB 001334	STKS 001144
S.SPLS= 010000	T.MR2 001432	SCNTLC 053311	SLKS 001342	STKSNV 052742
S.SPOV= 001000	T.MR3 001434	SCNTLC 052662	SLLVEC 001344	STM = 000001
S.TYPE= 000400	T.WER = 000020	SCNTLU 052655	SLCHUP 054300	STBPM 054526
S.UNLD= 040000	T.PAT 001440	SCPDOP 001216	SLPADR 001106	STBPMR 054562
S.UNS = 040000	T.POS 001436	SCRLF 001165	SLPFRN 001110	STPB 001152
S.VV = 000100	T.WCR 001412	SDB2D 054344	SLPVEC 001340	STPPLC 001157
S.WCLF= 000040	DEXATT= 000010	SDDW0 001254	SLSCS 001346	STPS 001150
S.WCAT= 000100	DFE = 000400	SDDW1 001256	SLSDR 001350	STRAP 055034
S.WLE = 004000	UNLOAD= 000107	SDDW10 001300	SLSTAD 053720	STRAP2 055056
S.WPL = 004000	UNRECY 030724	SDDW11 001302	SPADR1 001222	STRP = 000004
S.XDOK= 000020	UNS = 040000	SDDW12 001304	SPADR2 001226	STRPAD 055070
S.XERR= 001000	UPE = 070000	SDDW13 001306	SPADR3 001237	STSTM 001004
S1\$ 011060	VV = 000100	SDDW14 001310	SPADR4 001236	STSTN 001102
S2\$ 011072	WCE = 040000	SDDW15 001312	SWAIL 001170	STTVIN 052624
S3\$ 011122	WLE = 004000	SDDW2 001260	SWAVS1 001220	STYPE 051667
S4\$ 011160	WRDATA= 000123	SDDW3 001262	SWAVS2 001224	STYPEC 052074
S5\$ 011166	WRHEAD= 000127	SDDW4 001264	SWAVS3 001230	STYPEX 052142
S6\$ 011172	WRL = 004000	SDDW5 001266	SWAVS4 001234	SUBIT 001202
S7\$ 011200	WPTCHK= 000131	SDDW6 001270	SWBADF 001002	SUBITP 001010
S8\$ 011206	WRTEPK 012526	SDDW7 001272	SWFLC 055030	SUSNR 001214
TRITVF= 000014	WRTGAT= 040000	SDDW8 001274	SWHFN 052700	SVECT1 001240
TEST 013250	WPTVER 015076	SDDW9 001276	SWSCAC 001204	SVECT2 001242
TESTDV 011706	WVOS 015140	SDECNT 054450	SWSCLC 001206	SZDEC 054412
TEST1 013324	WV1\$ 015626	SDECVL 054552	SWSGTY 001170	. = 000023
TIMHR 001376	WV2\$ 015640	SDEVCT 001200	SWSWR 052667	..\$X = 001000
TIMMIN 001400	WV5\$ 015750	SEVM 001246	SWTVP1 001221	..ADD 004745
TIMOUT 025172	WV9\$ 015754	SENV 001210	SWTVP2 001225	..ASCII 004527
TIMSEC 001407	W.DRV 001540	SENVH 001211	SWTVP3 001231	..BLNK 004770
TIM000 063727	W.MILI 001466	SERPLC 001103	SWTVP4 001235	..CP 004442
TIM001 063734	W.MIN 001474	SERMAX 001115	SWULT 054064	..COMP 004052
TRVEC = 000060	W.MTIM 001464	SERPPC 001116	SWULL 001154	..HIGH 000226
TRPCNT 001564	W.SEC 001470	SERPTB 001314	SPASS 001176	..LCN 000224
TRPSEC 001572	W.TIME 001506	SERTTL 001112	SPASTP 001006	..OCT 004546
TRPTRF 001566	W.WCK = 000700	SETABL 001210	SPCNEP 054052	..RC 004630
TPVFC = 000064	W.WTCH 043314	SETFND 001314	SPBRCY 054046	..R1 004643
TPAPVF= 000034	W.8SEC 001472	SPATAL 001172	SPBRDN 053722	..R2 004656
TRTVEC= 000014	SAPTND 001000	SPFLC 055037	SPBRUP 053742	..R3 004671
TSTDEF 020116	SATYC 054612	SPILLC 001156	SQUES 001164	..R4 004704
TSTDRP 036714	SATY1 054566	SPILLS 001155	SRAND 054200	..R5 004717
TYPE = 104401	SATY3 054574	SGDADR 001120	SRDCHP 052250	..SP 004737
T.ASOF 001422	SATY4 054604	SGDDAT 001124	SRCLIN 052370	..STCK 004050
T.BA 001414	SAUTOB 001134	SPD = 000000	SPDSZ = 000031	

RK611/RFO6 PERFORMANCE EXERCISER MAINDEC DZR6P-R
DZR6PR.P11 11-AUG-76 00:00 SYMBOL TABLE

MACY11 30(1046) 01-DEC-77 14:15 PAGE 234

SEC 0255

ERPOPS DETECTED: 0

,DZR6PR/SOL/EQ:QNEWSM=DRIVE8,DUMP,DZR6PB
RUN-TIME: 30 29 1 SECONDS
RUN-TIME RATIO: 1457/61=23.6
CORE USED: 42K (83 PAGES)

H4