

RP04/05/06

HEAD ALIGN VERIFICATION
MD-11-DZRJC-A

EP-DZRJC-A-DL-A
COPYRIGHT © 1976
FICHE 1 OF 1

NOV 1976
digital
MADE IN USA

DZRJC
SEQ

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

A large, complex, black and white fractal-like pattern resembling a stylized letter 'E' or a comb. The pattern is composed of many small, repeating geometric shapes, creating a dense, textured appearance. The overall shape is roughly rectangular with a central vertical bar and horizontal bars extending from it, forming a comb-like structure.

[illegible]

.REM

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZRJC-A-D
PRODUCT NAME: RPO4 5.6 HEAD ALIGNMENT VERIFICATION PROGRAM
DATE CREATED: MAY 1976
MAINTAINER: DIAGNOSTIC ENGINEERING
AUTHOR: C. HESS

COPYRIGHT (C) 1976 DIGITAL EQUIPMENT CORP., MAYNARD, MASS.

THE INFORMATION IN THIS STATEMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION.

ACTUAL DISTRIBUTION OF THE SOFTWARE DESCRIBED IN THIS DOCUMENT WILL BE SUBJECT TO TERMS AND CONDITIONS TO BE ANNOUNCED ON SOME FUTURE DATE BY DIGITAL EQUIPMENT CORPORATION.

DEC ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.

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

CONTENTS

1. ABSTRACT
2. REQUIREMENTS
 - 2.1 EQUIPMENT
 - 2.2 PRELIMINARY PROGRAMS
3. LOADING PROCEDURES
4. STARTING PROCEDURES
 - 4.1 STARTING ADDRESSES
 - 4.2 PROGRAM & OPERATOR ACTION
 - 4.2 DRIVE SELECTION
 - 4.4 RH11 - RH73 UNIBUS ADDRESSES
 - 4.5 OTHER UNIBUS ADDRESSES
5. SWITCH REGISTER SETTINGS
6. ERRORS
 - 6.1 TEST ERRORS
 - 6.2 ENTRY ERRORS
 - 6.3 SUBSYSTEM/DEVICE ERROR MESSAGES
7. RESTRICTIONS
8. PROGRAM DESCRIPTION
 - 8.1 VERIFICATION MODE
 - 8.1.1 RPO4/S OPERATION
 - 8.1.2 RPO6 OPERATION
 - 8.2 ALIGNMENT MODE
 - 8.3 RANDOM SEEK UTILITY
9. PROGRAM LISTING

1. ABSTRACT

THE RPO4 5 6 HEAD ALIGNMENT VERIFICATION PROGRAM CHECKS RPO4, RPO5, OR RPO6 DISK DRIVE HEAD ALIGNMENT OR ALLOWS THE OPERATOR TO ALIGN HEADS WITH THE APPROPRIATE DRIVE'S TEST BOX. THE PROGRAM ALSO CONTAINS A UTILITY ROUTINE WHICH PERFORMS 5,000 RANDOM SEEKS WITHOUT DATA TRANSFERS. THE RANDOM SEEK UTILITY ALLOWS THE OPERATOR TO EXERCISE THE DRIVE WITH THE ALIGNMENT PACK IN PLACE AND THEN RE-VERIFY HEAD ALIGNMENT.

2. REQUIREMENTS

2.1 EQUIPMENT

PDP-11 PROCESSOR
8K MEMORY
TELETYPE
PROGRAM LOAD DEVICE
KW11-L OR KW11-P CLOCK
RH11 OR RH70 WITH RPO4'S, RPO5'S, OR RPO6'S
ALIGNMENT PACK ('CE' PACK)
'DDU' OR 'DEDU' FOR RPO4'S
'PERCH' FOR RPO5'S

2.2 PRELIMINARY PROGRAMS

THE RPO4/5/6 DISK SUBSYSTEM MUST BE OPERATION AND ALL OTHER PROGRAMS IN THE SET FOR THE SUBSYSTEM MUST RUN SUCCESSFULLY.

3. LOADING PROCEDURE

THE PROGRAM MAY BE LOADED FROM PAPER TAPE USING THE ABSOLUTE LOADER OR IT MAY BE LOADED FROM THE APPROPRIATE 'XXDP' MEDIA USING THE ASSOCIATED LOADER. THE PROGRAM MAY NOT BE INCLUDED IN A 'XXDP' CHAIN.

4. STARTING PROCEDURES

4.1 STARTING ADDRESSES

THE PROGRAM IS STARTED FROM LOCATION 200(9) IF THE ADDRESS OF THE RH11 OR RH70 WILL NOT BE CHANGED.

THE PROGRAM IS STARTED FROM LOCATION 204(9) IF THE ADDRESS

OF THE RH11 OR RH70 IS TO BE CHANGED FROM THE PRELOADED VALUE.
(SEE SECTION 4.4)

4.2 PROGRAM & OPERATION ACTION

1. LOAD THE PROGRAM INTO MEMORY (SEE SECTION 3)
2. LOAD ADDRESS 200 OR 204.
3. SET THE SWITCHES AND PRESS 'START'
4. PLACE AN ALIGNMENT DISK PACK ON THE DRIVE TO BE CHECKED AND SET THE DRIVE IN WRITE PROTECT MODE.
5. CYCLE UP THE DRIVE TO BE CHECKED. ALLOW SUFFICIENT TIME FOR THE ALIGNMENT PACK TO REACH OPERATING TEMPERATURE AND TO STABILIZE. REFER TO APPLICABLE MAINTENANCE PROCEDURES FOR THE REQUIRED AMOUNT OF TIME.
6. SELECT THE DRIVE TO BE CHECKED IN RESPONSE TO THE TYPEOUT ON THE TELETYPE.
7. THE PROGRAM WILL ASK FOR THE MODE OF OPERATION: ENTER AN 'A' IF THE PROGRAM IS TO BE USED FOR HEAD ALIGNMENT; ENTER A 'V' IF HEAD ALIGNMENT IS TO BE CHECKED; OR ENTER AN 'E' IF THE RANDOM SEEK UTILITY IS TO BE RUN.
8. IF AN 'A' IS ENTERED IN RESPONSE TO THE PROGRAM MODE MESSAGE, THE PROGRAM WILL POSITION THE DRIVE TO THE ALIGNMENT CYLINDER: RP04/5 - CYLINDER 245; RP06 - CYLINDER 496. THE PROGRAM WILL THEN ASK FOR A HEAD ADDRESS. THE HEAD ENTERED WILL BE SELECTED AND THE PROGRAM WILL REQUEST ANOTHER HEAD. UNTIL A NEW HEAD ADDRESS IS ENTERED, THE LAST ENTERED HEAD ADDRESS WILL REMAIN SELECTED. THE ALIGNMENT SEQUENCE MAY BE TERMINATED BY TYPING A 'CONTROL C'; THE PROGRAM WILL RETURN TO THE DRIVE SELECTION ROUTINE.
9. IF THE PROGRAM IS BEING RUN IN 'VERIFICATION' MODE AND SW<02> IS SET, THE PROGRAM WILL ASK FOR HEAD AND CYLINDER ADDRESSES. (HEAD AND CYLINDER ADDRESSES ARE ENTERED IN DECIMAL.) WITH SW<02>, ONLY THE ALIGNMENT OF THE SPECIFIED HEAD WILL BE CHECKED.
10. WHEN THE OPERATOR HAS COMPLETED THE PRESENT DRIVE, CYCLE THE DRIVE DOWN, AND TRANSFER THE TEST BOX AND ALIGNMENT PACK TO THE NEXT DRIVE TO BE CHECKED. WHEN THE ALIGNMENT PACK HAS STABILIZED, THE NEW DRIVE CAN BE SELECTED. IT IS NOT NECESSARY TO RESTART THE PROGRAM.

4.3 DRIVE SELECTION

ENTER A DRIVE NUMBER IN RESPONSE TO THE 'ENTER DRIVE NUMBER:' TYPEOUT FROM THE PROGRAM. ONLY VALID DRIVE NUMBERS (0 - 7) WILL BE ACCEPTED BY THE PROGRAM. NOTE THAT THE DRIVE SETUP

(TEST BOX CONNECTION AND ALIGNMENT PACK STABILIZATION) MUST HAVE BEEN COMPLETED BEFORE A DRIVE IS SELECTED.

4.4 RH11 - RH70 UNIBUS ADDRESSES

THE PROGRAM ASSUMES THAT THE RH11 OR RH70 ADDRESSES START AT 176700 AND THAT THE VECTOR ADDRESS IS 254. THESE ADDRESSES MAY BE CHANGED WHEN THE PROGRAM IS STARTED FROM LOCATION 204. ENTER THE RH11/RH70 ADDRESS IN RESPONSE TO THE REQUEST FROM THE PROGRAM.

4.5 OTHER UNIBUS ADDRESSES

LOC	TAG	CONTENTS	FUNCTION
---	---	-----	-----
1144	STKS	177560	TTY KEYBOARD STATUS REGISTER
1146	STKB	177562	TTY KEYBOARD BUFFER REGISTER
1150	STPS	177564	TTY PRINTER STATUS REGISTER
1152	STPB	177566	TTY PRINTER BUFFER REGISTER
1244	PKV	104	KW11-P CLOCK VECTOR ADDRESS
1246	PKCS	172540	KW11-P CONTROL REGISTER
1250	PKB	172542	KW11-P COUNT SET REGISTER
1252	PKC	172544	KW11-P COUNTER REGISTER
1254	LKV	100	KW11-L CLOCK VECTOR ADDRESS
1256	LKS	177546	KW11-L STATUS REGISTER

5. SWITCH REGISTER SETTINGS

SW<15>=1...HALT ON ERROR
 SW<13>=1...INHIBIT ERROR TIMEOUTS
 SW<09>=1...LOOP ON ERROR
 SW<02>=1...CHECK ALIGNMENT OF THE SPECIFIED HEAD
 SW<01>=1...LOOP ON THE CURRENT HEAD
 SW<00>=1...TIMEOUT ALL TRACK CENTER VALUES

IF THE PROGRAM IS BEING RUN ON A SWITCHLESS PROCESSOR (I.E. AN 11-34) THE PROGRAM WILL DETERMINE THAT THE HARDWARE SWITCH REGISTER IS NOT PRESENT AND WILL USE A 'SOFTWARE' SWITCH REGISTER. THE 'SOFTWARE' SWITCH REGISTER IS LOCATED AT LOCATION 176 (8). THE SETTINGS OF THE 'SOFTWARE' SWITCHES ARE CONTROLLED THROUGH A KEYBOARD ROUTINE WHICH IS CALLED BY TYPING A 'CONTROL G'. THE PROGRAM WILL RECOGNIZE THE 'CONTROL G' AT ANY TIME EXCEPT WHEN THE PROGRAM IS AT A HIGHER PRIORITY PROCESSING AN RPD4/5/6 INTERRUPT. THE 'SOFTWARE' SWITCH VALUES ARE ENTERED AS AN OCTAL NUMBER IN RESPONSE TO THE PROMPT FROM THE SWITCH ENTRY ROUTINE:

'SWR = NNNNNN NEW ='

EACH TIME SWITCH SETTING ARE ENTERED, THE ENTIRE SWITCH REGISTER IMAGE MUST BE ENTERED. LEADING ZEROS ARE NOT REQUIRED. 'RUBOUT' AND 'CONTROL U' FUNCTIONS MAY BE USED TO CORRECT TYPING ERRORS

DURING SWITCH ENTRY.

ON PROCESSORS WITH HARDWARE SWITCH REGISTERS, THE 'SOFTWARE' SWITCH REGISTER MAY BE USED. IF THE PROGRAM FINDS ALL 16 SWITCHES IN THE 'UP' POSITION, ALL SWITCH REGISTER REFERENCES WILL BE TO THE 'SOFTWARE' REGISTER AND THE PROCEDURES DESCRIBED ABOVE MUST BE FOLLOWED.

5. ERRORS

5.1 TEST ERRORS

RP04/5:

THE PROGRAM CHECKS THAT EACH HEAD IS WITHIN + OR - 150 MICRO-INCHES FROM THE TRACK CENTERLINE FOR CYLINDER 245 AND IS WITHIN + OR - 350 MICRO-INCHES FOR CYLINDERS 4 AND 400. IF A HEAD IS FOUND THAT IS NOT WITHIN THESE TOLERANCES, AN ERROR MESSAGE IS TYPED. THE ERROR MESSAGE IDENTIFIES THE CYLINDER, HEAD, AND THE ACTUAL POSITION OF THE HEAD RELATIVE TO THE CENTERLINE. (IF SW<00>=1, THE ACTUAL ALIGNMENT OF EACH HEAD WILL BE TYPED OUT; HEADS OUT OF TOLERANCE WILL NOT BE IDENTIFIED.)

RP06:

THE PROGRAM CHECKS THAT EACH HEAD IS WITHIN + OR - 75 MICRO-INCHES FROM THE TRACK CENTERLINE FOR CYLINDER 496 AND IS WITHIN + OR - 175 MICRO-INCHES FOR CYLINDERS 8 AND 800. IF A HEAD IS FOUND THAT IS NOT WITHIN THESE TOLERANCES, AN ERROR MESSAGE IS TYPED. THE ERROR MESSAGE IDENTIFIES THE CYLINDER, HEAD, AND THE ACTUAL POSITION OF THE HEAD RELATIVE TO THE CENTERLINE. (IF SW<00>=1, THE ACTUAL ALIGNMENT OF EACH HEAD WILL BE TYPED OUT; HEADS OUT OF TOLERANCE WILL NOT BE IDENTIFIED.)

5.2 ENTRY ERRORS

THE PROGRAM WILL NOT ACCEPT DRIVE NUMBERS GREATER THAN 7. IF AN INVALID DRIVE NUMBER IS ENTERED IN RESPONSE TO THE DRIVE NUMBER REQUEST, THE PROGRAM WILL TYPE A '?'; THE OPERATOR MAY ENTER A VALID NUMBER.

THE PROGRAM WILL NOT ACCEPT HEAD ADDRESSES GREATER THAN 19 (DECIMAL) OR CYLINDER ADDRESSES OTHER THAN 245(10) FOR RP04/5'S OR 496(10) FOR RP06'S. IF AN INVALID CYLINDER OR HEAD ADDRESS IS ENTERED, THE PROGRAM WILL REJECT THE ENTRY AND RETURN TO THE DRIVE SELECTION ROUTINE.

THE PROGRAM WILL NOT ALLOW A DRIVE TO BE ASSIGNED IF IT IS NOT 'WRITE PROTECTED'. IF THIS IS ATTEMPTED, THE OPERATOR WILL BE NOTIFIED.

DRIVES ASSIGNED ARE CHECKED TO ENSURE THAT THE DRIVE ASSIGNED IS PRESENT, ONLINE, AND IS AN RP04, RP05, OR RP06. THE PROGRAM WILL REJECT ASSIGNMENTS FOR DRIVES WHICH ARE NOT PRESENT OR ARE NOT ONLINE.

6.3 SUBSYSTEM/DEVICE ERROR MESSAGES

1. 'RH11 INTERRUPT OCCURRED (RPAS=0)' - AN INTERRUPT OCCURRED, BUT NOTHING ON THE MASSBUS IS INDICATING AN ATTENTION.
2. 'UNEXPECTED ATTENTION OCCURRED' - THE INDICATED DRIVE INTERRUPTED, BUT NO INTERRUPT WAS EXPECTED FROM THE INDICATED DRIVE.
3. 'MASSBUS PARITY ERROR (MCPE=1)' - A CONTROL BUS PARITY ERROR WAS DETECTED BY THE RH11 WHEN THE INDICATED REGISTER WAS READ.
4. 'MASSBUS PARITY ERROR (PAR=1)' - A CONTROL BUS PARITY ERROR OCCURRED WHEN THE INDICATED REGISTER WAS WRITTEN.
5. 'ADDRESS PLUG CHANGE BIT SET' - THE PROGRAM FOUND THE 'OPE' BIT SET FOR THE INDICATED DRIVE.
6. 'RH11 DIDN'T RESPOND TO ADDRESSING' - THE PROGRAM ACCESSED THE RH11 AT THE INDICATED ADDRESS AND RECEIVED NO RESPONSE.
7. 'DRIVE OR DATA ERROR' - THE INDICATED DRIVE COMPLETED AN OPERATION WITH THE INDICATED ERROR CONDITIONS SET.
8. 'DRIVE UNSAFE ERROR' - THE INDICATED DRIVE SIGNALLED AN UNSAFE CONDITION.
9. 'HEAD OUT OF ALIGNMENT' - THE INDICATED HEAD OF NOT WITHIN TOLERANCE.
10. 'HEAD TOO FAR OUT OF ALIGNMENT' - THE INDICATED HEAD IS TOO FAR OUT OF ALIGNMENT FOR THE PROGRAM TO FIND THE TRACK CENTERLINE FOR THAT HEAD.
11. 'SOFTWARE TIMEOUT' - THE INDICATED DID NOT COMPLETE THE OPERATION WITH 1 SECOND.
12. 'UNCORRECTABLE MASSBUS PARITY ERROR' - THE PROGRAM ATTEMPTED TO PERFORM AN OPERATION AND DETECTED 3 SUCESSIVE MASSBUS PARITY ERRORS OR THE PROGRAM ATTEMPTED TO CLEAR A 'PAR' ERROR IN THE DRIVE AND A PARITY ERROR OCCURRED.
13. 'DRIVE IS UNAVAILABLE' - THE INDICATED DRIVE HAS GONE OFFLINE OR HAS BECOME NON-EXISITENT.

7. RESTRICTIONS

BECAUSE AN ALIGNMENT DISK PACK IS USED ON DRIVES TESTED BY THIS

PROGRAM. NO DATA TRANSFER OPERATIONS ARE PERFORMED.

9. PROGRAM DESCRIPTION

9.1 VERIFICATION MODE

9.1.1 RPO4/5 OPERATIONS

THE PROGRAM CHECKS HEAD ALIGNMENT AT CYLINDER 245, HEADS 0 - 18, AT CYLINDERS 400 AND 4, HEADS 0 AND 18, AND REVERIFIES ALIGNMENT AT CYLINDER 245, HEADS 0 - 18. THE OPERATOR WILL BE NOTIFIED IF ANY HEAD IS OUT OF ALIGNMENT BY MORE THAN THE SPECIFIED AMOUNT.

HEAD ALIGNMENT IS CHECKED IN THE FOLLOWING MANNER:

1. OFFSET THE POSITIONER TO +1200 MICRO-INCHES.
2. STORE THE SIGN CHANGE BIT.
3. MOVE THE POSITIONER IN THE OPPOSITE DIRECTION IN 25 MICRO-INCH INCREMENTS UNTIL THE SIGN CHANGE BIT CHANGES VALUE. STORE THE OFFSET VALUE.
4. OFFSET THE POSITIONER TO -1200 MICRO-INCHES AND REPEAT STEPS 2 AND 3 ABOVE.
5. AVERAGE THE TWO SIGN CHANGE OFFSET VALUES AND REPORT IF THE SELECTED HEAD IS MISALIGNED BY MORE THAN + OR - 150 MICRO-INCHES FOR CYLINDER 245 OR + OR - 350 MICRO-INCHES FOR CYLINDERS 4 AND 400.

REPEAT THE ABOVE SEQUENCE FOR ALL HEADS AT CYLINDER 245 AND FOR HEADS 0 AND 18 AT CYLINDERS 4 AND 400.

THE OPERATOR MAY TERMINATE THE OPERATION AT ANY TIME BY TYPING A 'CONTROL C'.

9.1.2 RPO6 OPERATIONS

THE PROGRAM CHECKS HEAD ALIGNMENT AT CYLINDER 496, HEADS 0 - 18, AT CYLINDERS 800 AND 8, HEADS 0, 1, 10, 17, AND 18, AND RE-VERIFIES ALIGNMENT AT CYLINDER 496, HEADS 0 - 18. THE OPERATOR WILL BE NOTIFIED IF A HEAD IS OUT OF ALIGNMENT BY MORE THAN THE SPECIFIED AMOUNT.

HEAD ALIGNMENT IS CHECKED IN THE FOLLOWING MANNER:

1. OFFSET THE POSITIONER TO +600 MICRO-INCHES.
2. STORE THE SIGN CHANGE BIT.
3. MOVE THE POSITIONER IN THE OPPOSITE DIRECTION IN 25 MICRO-INCH

INCREMENTS UNTIL THE SIGN CHANGE BIT CHANGES VALUE. STORE THE OFFSET VALUE.

4. OFFSET THE POSITIONER TO -600 MICRO-INCHES AND REPEAT STEPS 2 AND 3 ABOVE.
5. AVERAGE THE TWO SIGN CHANGE OFFSET VALUES AND REPORT IF THE SELECTED HEAD IS MISALIGNED BY MORE THAN + OR - 75 MICRO-INCHES FOR CYLINDER 496 OR + OR - 175 MICRO-INCHES FOR CYLINDERS 8 AND 800.

REPEAT THE ABOVE SEQUENCE FOR ALL HEADS AT CYLINDER 496 AND FOR HEADS 0, 1, 10, 17, 18 AT CYLINDERS 8 AND 800.

THE OPERATOR MAY TERMINATE THE OPERATION AT ANY TIME BY TYPING A 'CONTROL C'.

8.2

ALIGNMENT MODE

THE PROGRAM MAY ALSO BE USED TO PROVIDE HEAD SELECTION FOR DDU CONTROLLED HEAD ALIGNMENT. WHEN THIS MODE IS SELECTED, THE PROGRAM WILL PERFORM NO ALIGNMENT CHECKING - ONLY THE REQUESTED HEAD IS SELECTED IN THE DCL. THE ACTUAL ALIGNMENT MUST BE PERFORMED USING THE ALIGNMENT METER ON THE DDU, DEDU, OR PERCH. WHEN USED IN THE ALIGNMENT MODE, THE PROGRAM PROVIDES HEAD SELECTION WHICH IS NOT PRESENT IN THE DDU.

THE OPERATOR MAY TERMINATE THE OPERATION AT ANY TIME BY TYPING A 'CONTROL C'.

8.3

RANDOM SEEK UTILITY

THE PROGRAM CONTAINS A UTILITY ROUTINE WHICH PERFORMS 5,000 (10) RANDOM SEEK OPERATIONS ON THE DRIVE BEING CHECKED. THIS UTILITY ALLOWS THE OPERATOR TO EXERCISE THE HEAD ASSEMBLY ON THE DRIVE AFTER HEAD ALIGNMENT HAS BEEN PERFORMED.

THE UTILITY ROUTINE IS NORMALLY USED AFTER HEADS HAVE BEEN ALIGNED: THE RANDOM SEEK UTILITY IS CALLED AND HEAD ALIGNMENT IS RE-VERIFIED AT THE COMPLETION OF THE RANDOM SEEKS. USE OF THIS ROUTINE DOES NOT REQUIRE CYCLING DOWN THE DRIVE AND REPLACING THE ALIGNMENT PACK WITH A SCRATCH PACK FOR THE HEAD SHAKE DOWN.

THE OPERATOR MAY TERMINATE THE OPERATION AT ANY TIME BY TYPING A 'CONTROL C'.

9.

PROGRAM LISTING

L01

.MAIN. MACY11 27(655) 27-APR-76 16:51 PAGE 10
DZRJCA.DOC

SEQ 0010

%
.END

ERRORS DETECTED: 0

*.DZRJCA/SOL=DZRJCA.DOC
RUN-TIME: 1 2 0 SECONDS
CORE USED: 3K

11/11

16	OPERATIONAL SWITCH SETTINGS
26	BASIC DEFINITIONS
139	RH11 REGISTERS
194	RPO4/5/6 REGISTERS
377	TRAP CATCHER
387	PROGRAM STARTING ADDRESS = 200
392	COMMON TAGS
469	RH11/RPO4/5/6 ADDRESS & VECTOR LOCATIONS
475	ERROR POINTER TABLE
593	RPO4/5/6 DRIVER COMMANDS
621	PROGRAM START AND INITIALIZATION ROUTINES
629	INITIALIZE THE COMMON TAGS
665	GET VALUE FOR SOFTWARE SWITCH REGISTER
759	SETUP TO CHECK ONLY THE SPECIFIED HEAD
800	MAIN ROUTINE - CHECK ALL HEADS AT ALL ALIGNMENT CYLNDERS
836	ROUTINE TO FIND THE TRACK CENTER
940	ROUTINE TO SELECT HEAD FOR DDU CONTROLLED HEAD ALIGNMENT
969	ROUTINE TO PERFORM 5,000 RANDOM SEEKS
994	COMMON ENTRY TO THE RPO4/5/6 DRIVER
1037	SUBROUTINES
1126	MACRO ROUTINES
1128	ERROR HANDLER ROUTINE
1162	ERROR MESSAGE TYPEOUT ROUTINE
1219	TYPE ROUTINE
1288	BINARY TO OCTAL (ASCII) AND TYPE
1365	CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
1432	TTY INPUT ROUTINE
1690	RANDOM NUMBER GENERATOR ROUTINE
1726	INTEGER DIVIDE ROUTINE
1808	SAVE AND RESTORE R0-R5 ROUTINES
1853	TRAP DECODER
1876	TRAP TABLE
1899	SINGLE/DUAL PORT RH11/RPO4/5/6 DRIVER (REV 1.0)
3296	DATA PARAMETER BLOCK
3355	HEAD CODE TABLE
3402	OFFSET CODE TABLES
3559	MESSAGES
3949	BUSADR - GET BUS ADDRESS AND VECTOR ADDRESS FOR RH11
4014	CK.NUM - CHECK NUMBER (OCTAL)

NO1

MD-11-DZJIC-A. RPO4 5 '6 HEAD ALIGNMENT PROGRAM MACY!! 27(655) 27-APR-76 16:34 PAGE 1
RPO455.010

SEQ 0012

1

TITLE MD-11-DIGIO-A, RPO-5 & HEAD ALIGNMENT PROGRAM
 *COPYRIGHT © 1976
 *DIGITAL EQUIPMENT CORP.
 *MAYNARD, MASS. 01754

*PROGRAM BY C. HESS

*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC 3 SYAC
 *PACKAGE (MAINDEC-11-DIGAC-C11), MAR 24, 1976.

000001

*STN=1
 *SBTTL OPERATIONAL SWITCH SETTINGS

SWITCH	USE
15	HALT ON ERROR
13	INHIBIT ERROR TYPEOUTS
9	LOOP ON ERROR
2	CHECK ONLY THE SPECIFIED HEAD
1	LOOP ON THE CURRENT HEAD
0	TYPE ALL TRACK CENTER VALUES

*SBTTL BASIC DEFINITIONS

001100

*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***

STACK= 1100
 *EQUIV EMT.ERROR ::BASIC DEFINITION OF ERROR CALL
 *EQUIV IOT.SCOPE ::BASIC DEFINITION OF SCOPE CALL

*MISCELLANEOUS DEFINITIONS

000011

HT= 11 ::CODE FOR HORIZONTAL TAB
 LF= 12 ::CODE FOR LINE FEED
 CR= 13 ::CODE FOR CARRIAGE RETURN
 CR LF= 200 ::CODE FOR CARRIAGE RETURN-LINE FEED
 PS= 177775 ::PROCESSOR STATUS WORD
 *EQUIV PS.PSW
 STKLM= 177774 ::STACK LIMIT REGISTER
 PIR= 177772 ::PROGRAM INTERRUPT REQUEST REGISTER
 DSW= 177570 ::HARDWARE SWITCH REGISTER
 DDISP= 177570 ::HARDWARE DISPLAY REGISTER

000000

*GENERAL PURPOSE REGISTER DEFINITIONS

R0= 00 ::GENERAL REGISTER
 R1= 01 ::GENERAL REGISTER
 R2= 02 ::GENERAL REGISTER
 R3= 03 ::GENERAL REGISTER
 R4= 04 ::GENERAL REGISTER
 R5= 05 ::GENERAL REGISTER
 R6= 06 ::GENERAL REGISTER
 R7= 07 ::GENERAL REGISTER
 *EQUIV R6.SP ::STACK POINTER
 *EQUIV R7.PC ::PROGRAM COUNTER

*PRIORITY LEVEL DEFINITIONS

16:34 PAGE 3

[illegible]

9	T	15	=	1000000
8	T	14	=	4000000
7	T	13	=	2000000
6	T	12	=	1000000
5	T	11	=	400000
4	T	10	=	200000
3	T	9	=	100000
2	T	8	=	40000
1	T	7	=	20000
0	T	6	=	10000
9	T	5	=	4000
8	T	4	=	2000
7	T	3	=	1000
6	T	2	=	400
5	T	1	=	200
4	T	0	=	100

000040
 000100
 000200
 000400
 001000
 002000
 004000
 010000
 020000
 040000
 100000

 CLR= 40
 IR= 100
 OR= 200
 MPE= 400
 MXF= 1000
 PGE= 2000
 NEM= 4000
 NED= 10000
 LPE= 20000
 WCE= 40000
 DLT= 100000

 :CLEAR (BIT #5)
 :INPUT READY (BIT #6)
 :OUTPUT READY (BIT #7)
 :MASS BUS PARITY ERROR (BIT #8)
 :MISSED TRANSFER ERROR (BIT #9)
 :PROGRAM ERROR (BIT #10)
 :NON EXISTENT MEMORY (BIT #11)
 :NON EXISTENT DRIVE (BIT #12)
 :UNIBUS PARITY ERROR (BIT #13)
 :WRITE CHECK ERROR (BIT #14)
 :DATA LATE (BIT #15)

 :DATA BUFFER REGISTER (RPOB)
 :EACH BIT IS CALLED BY BIT NUMBER)

:*****

.SSTTL RPO4 S 6 REGISTERS

:*****

:CONTROL AND STATUS 1 REGISTER. (#00)

 000001
 000002
 000004
 000010
 000020
 000040
 004000

 GO= 1
 F1= 2
 F2= 4
 F3= 10
 F4= 20
 F5= 40
 CVA= 4000

 :GO BIT (BIT #0)
 :FUNCTION CODE BIT #1
 :FUNCTION CODE BIT #2
 :FUNCTION CODE BIT #3
 :FUNCTION CODE BIT #4
 :FUNCTION CODE BIT #5
 :DEVICE AVAILABLE (BIT #11)

:DRIVE STATUS REGISTER (RPOS1) (#01)

 000002
 000004
 000010
 000020
 000040
 000100
 000200
 000400
 001000
 002000
 004000
 010000
 020000
 040000
 100000

 :CF5= 1
 :OFF20= 2
 :DIGB= 4
 :GRV= 10
 :DL64= 20
 :DE1= 40
 :V= 100
 :DRY= 200
 :DPR= 400
 :PGM= 1000
 :LST= 2000
 :WRL= 4000
 :MCL= 10000
 :PIP= 20000
 :ERR= 40000
 :ATA= 100000

 :DRIVE FORWARD 5" SEC. (BIT #0)
 :DRIVE FORWARD 20" SEC. (BIT #1)
 :DRIVE TO INNER GUARD BAND (BIT #2)
 :GO REVERSE (BIT #3)
 :DIFFERENCE LESS THAN 64 (BIT #4)
 :DIFFERENCE EQUALS 1 (BIT #5)
 :VOLUME VALID (BIT #6)
 :DRIVE READY (BIT #7)
 :DRIVE PRESENT (BIT #8)
 :PROGRAMABLE (BIT #9)
 :LAST SECTOR TRANSFERRED (BIT #10)
 :WRITE LOCK (BIT #11)
 :MEDIUM ON-LINE (BIT #12)
 :POSITIONING OPERATION IN PROGRESS (BIT #13)
 :COMPOSITE ERROR (BIT #14)
 :ATTENTION ACTIVE (BIT #15)

:ERROR REGISTER #01 (RPER1) (#02)

000001

ILF= 1

:ILLEGAL FUNCTION (BIT #0)

00-11-2210-A, REG 5 6 HEAD ALIGNMENT PROGRAM
 00-11-2210-A, REG 5 6 REGISTERS

MACY 11 27-655, 27-APR-76 16:34 PAGE 6

ST

300002
000004
000010
000020
000040
000100
000200
000400
001000
002000
004000
010000
020000
040000
100000

```
YLR=          2
AMR=          4
PAR=         10
FER=         20
WCF=         40
ECH=        100
HCE=        200
HCRC=        400
AOE=       1000
IAE=       2000
WLE=       4000
UTE=      10000
OPT=      20000
UNS=      40000
DCK=     100000
```

```

:ILLEGAL REGISTER (BIT #1)
:REGISTER MODIFICATION REFUSED (BIT #2)
:PARITY ERROR (BIT #3)
:FORMAT ERROR (BIT #4)
:WRITE CLOCK FAIL (BIT #5)
:ECC HARD ERROR (BIT #6)
:HEADER COMPARE ERROR (BIT #7)
:HEADER CRC ERROR (BIT #8)
:ADDRESS OVERFLOW ERROR (BIT #9)
:INVALID ADDRESS ERROR (BIT #10)
:WRITE LOCK ERROR (BIT #11)
:DRIVE TIMING ERROR (BIT #12)
:OPERATION INCOMPLETE (BIT #13)
:DRIVE UNSAFE (BIT #14)
:DATA CHECK ERROR (BIT #15)

```

:MAINTAINABILITY REGISTER (REMR) (#03)

— 22 —

```

DMO=      1
MCLK=     2
MINX=     4
MSTCK=    10
MRD=     20
MWR=     40
PDSY=    200

```

```
:DIAGNOSTIC MODE BIT #0.  
:MAINTAINABILITY CLOCK (BIT #1)  
:MAINTAINABILITY INDEX (BIT #2)  
:MAINTAINABILITY SECTOR CLOCK (BIT #3)  
:MAINTAINABILITY READ (BIT #4)  
:MAINTAINABILITY WRITE (BIT #5)  
:MAINTAINABILITY SYNC DETECTED (BIT #7)
```

:ATTENTION SUMMARY PSELCO-REGISTER (RPAS) (804)

000001
000002
000004
000010
000020
000040
000100
000200

AT0=	1
AT1=	4
AT2=	10
AT3=	20
AT4=	40
AT5=	100
AT6=	200

```

:DEVICE 0 (BIT #0)
:DEVICE 1 (BIT #1)
:DEVICE 2 (BIT #2)
:DEVICE 3 (BIT #3)
:DEVICE 4 (BIT #4)
:DEVICE 5 (BIT #5)
:DEVICE 6 (BIT #6)
:DEVICE 7 (BIT #7)

```

```

;DESIRED SECTOR, TRACK ADDRESS REGISTER (RPDA) (8CS)
; (EACH BIT IS CALLED BY BIT NUMBER)

```

:DRIVE TYPE REGISTER (RPDT) (#36)

00000:
000002
000004
000010
000020
000040
000100
000200
000400
004000
020000
040000

[illegible]

```

:DRIVE TYPE NUMBER BIT 1
:DRIVE TYPE NUMBER BIT 2
:DRIVE TYPE NUMBER BIT 3
:DRIVE TYPE NUMBER BIT 4
:DRIVE TYPE NUMBER BIT 5
:DRIVE TYPE NUMBER BIT 6
:DRIVE TYPE NUMBER BIT 7
:DRIVE TYPE NUMBER BIT 8
:DRIVE TYPE NUMBER BIT 9
:DRIVE REQUEST REQUIRED (BIT #11)
:MOVING HEAD (BIT #13)
:TAPE DRIVE (BIT #14)

```

100000

NBA= 100000

;NOT BLOCK ADDRESSED (BIT #15)

;LOOK-AHEAD REGISTER (RPLA) (#07)

 000001
 000002
 000004
 000010
 000020
 000040
 000100
 000200
 001000
 002000
 004000
 010000
 020000
 040000
 100000

 EXT1= 1
 EXT2= 2
 EXT4= 4
 EXT10= 10
 EXT20= 20
 EXT40= 40
 SC1= 100
 SC2= 200
 SC4= 400
 SC10= 1000
 SC20= 2000
 TRK1= 4000
 TRK2= 10000
 TRK4= 20000
 TRK10= 40000
 TRK20= 100000

 ;EXTENSION 1 (BIT #0)
 ;EXTENSION 2 (BIT #1)
 ;EXTENSION 3 (BIT #2)
 ;EXTENSION 4 (BIT #3)
 ;EXTENSION 5 (BIT #4)
 ;EXTENSION 6 (BIT #5)
 ;SECTOR COUNT FIELD 0 (BIT #6)
 ;SECTOR COUNT FIELD 1 (BIT #7)
 ;SECTOR COUNT FIELD 2 (BIT #8)
 ;SECTOR COUNT FIELD 3 (BIT #9)
 ;SECTOR COUNT FIELD 4 (BIT #10)
 ;TRACK FIELD 1 (BIT #11)
 ;TRACK FIELD 2 (BIT #12)
 ;TRACK FIELD 3 (BIT #13)
 ;TRACK FIELD 4 (BIT #14)
 ;TRACK FIELD 5 (BIT #15)

;RPO4 ERROR REGISTER #2 (RPER2) (#10)

 000001
 000002
 000004
 000010
 000020
 000040
 000100
 000200
 000400
 001000
 002000
 004000
 010000
 020000
 100000

 WCU= 1
 CSF= 2
 WSU= 4
 CSU= 10
 MSE= 20
 TDF= 40
 TUF= 100
 FEN= 200
 WRU= 400
 MHS= 1000
 NHS= 2000
 IXE= 4000
 VU30= 10000
 PLU= 20000
 ACU= 100000

 ;WRITE CURRENT UNSAFE (BIT #0)
 ;CURRENT SINK FAILURE (BIT #1)
 ;WRITE SELECT UNSAFE (BIT #2)
 ;CURRENT SWITCH UNSAFE (BIT #3)
 ;MOTOR SEQUENCE ERROR (BIT #4)
 ;TRANSITIONS DETECTOR FAILURE (BIT #5)
 ;TRANSITIONS UNSAFE (BIT #6)
 ;FAILSAFE ENABLED (BIT #7)
 ;WRITE READY UNSAFE (BIT #8)
 ;MULTIPLE HEAD SELECT (BIT #9)
 ;NO HEAD SELECTION (BIT #10)
 ;INDEX ERROR (BIT #11)
 ;GOVOLT UNSAFE (BIT #12)
 ;PLO UNSAFE (BIT #13)
 ;AC UNSAFE (BIT #15)

;RPO5,6 ERROR REGISTER #02 (RPER2) (#10)

 000001
 000002
 000004
 000010
 000020
 000040
 000100
 000200
 000400
 001000
 002000
 004000
 020000

 WCU= 1
 CSF= 2
 WSU= 4
 CSU= 10
 RAW= 20
 TDF= 40
 TUF= 100
 ABS= 200
 WRU= 400
 MHS= 1000
 NHS= 2000
 IXE= 4000
 PLJ= 20000

 ;WRITE CURRENT UNSAFE (BIT #0)
 ;CURRENT SINK FAILURE (BIT #1)
 ;WRITE SELECT UNSAFE (BIT #2)
 ;CURRENT SWITCH UNSAFE (BIT #3)
 ;READ AND WRITE (BIT #4)
 ;TRANSITIONS DETECTOR FAILURE (BIT #5)
 ;TRANSITIONS UNSAFE (BIT #6)
 ;ABNORMAL STOP (BIT #7)
 ;WRITE READY UNSAFE (BIT #8)
 ;MULTIPLE HEAD SELECT (BIT #9)
 ;NO HEAD SELECTION (BIT #10)
 ;INDEX ERROR (BIT #11)
 ;PLO UNSAFE (BIT #12)

CPAC.009 RP04 5 6 REGISTERS

:OFFSET REGISTER (RPOF) (#11)

000001
000002
000004
000010
000020
000040
000200
002000
004000
010000OF25= 1
OF50= 2
OF100= 4
OF200= 10
OF400= 20
OF800= 40
OFREV= 200
HCI= 2000
ECI= 4000
FMT22= 10000:OFFSET 25 MICRO INCHES (BIT #0)
:OFFSET 50 MICRO INCHES (BIT #1)
:OFFSET 100 MICRO INCHES (BIT #2)
:OFFSET 200 MICRO INCHES (BIT #3)
:OFFSET 400 MICRO INCHES (BIT #4)
:OFFSET 800 MICRO INCHES (BIT #5)
:OFFSET NEGATIVE (REVERSE) (BIT #5)
:HEADER COMPARE INHIBIT (BIT #10)
:ERROR CORRECTION CODE INHIBIT (BIT #11)
:FORMAT BIT (BIT #12):DESIRED CYLINDER ADDRESS (RPCA) (#12)
:(EACH BIT IS CALLED BY BIT NUMBER):CURRENT CYLINDER ADDRESS (RPCC) (#13)
:(EACH BIT IS CALLED BY BIT NUMBER):SERIAL NUMBER REGISTER (RPSN) (#14)
:(EACH IS CALLED BY BIT NUMBER)

:RP04 ERROR REGISTER #03 (PPER3) (#15)

000001
000002
000010
000040
000100
0-0000
100000FSU= 1
VUF= 2
UWR= 10
ACL= 40
DCL= 100
SKI= 40000
OCYL= 100000:PACK SPEED UNSAFE (BIT #0)
:VELOCITY UNSAFE (BIT #1)
:ANY UNSAFE EXCEPT READ WRITE (BIT #3)
:AC LOW (BIT #5)
:DC LOW (BIT #6)
:SEEK INCOMPLETE (BIT #14)
:OFF CYLINDER (BIT #15)

:RP05/6 ERROR REGISTER #03 (RPER3) (#15)

000001
000002
000040
000100
000000
040000
100000DCU= 1
WAC= 2
ACL= 40
DCL= 100
JPE= 20000
SKI= 40000
OCYL= 100000:DC UNSAFE (BIT #0)
:WRITE AND OFFSET (BIT #1)
:AC LOW (BIT #5)
:DC LOW (BIT #6)
:OPERATOR PLUG ERROR (BIT #13)
:SEEK INCOMPLETE (BIT #14)
:OFF CYLINDER ERROR (BIT #15):ECC POSITION REGISTER (RPEC1) (#16)
:(EACH BIT IS CALLED BY BIT NUMBER):ECC PATTERN REGISTER (RPEC2) (#17)
:(EACH BIT IS CALLED BY BIT NUMBER)

.SBTTL TRAP CATCHER

000000

.=0

:*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A "+2,HALT"
:*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS

		000174	
000174		000900	
000176		000900	
		000200	
00T		0000	000
00W		0001	T-1
00V		0000	000

```
DISPREG: .WORD 0
SWREG: .WORD 0
```

```

::SOFTWARE DISPLAY REGISTER
::SOFTWARE SWITCH REGISTER

```

200

```

JMP      START1
JMP      START

```

```
;START THE PROGRAM, USE THE DEFAULT PH11 ADDRESS
;START THE PROGRAM, CHANGE THE PH11 ADDRESS
```

.SBTTL COMMON TAGS

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

390		001100	SCMTAG: .=1100	START OF COMMON TAGS
391	001100	000000	\$PASS: .WORD 0	CONTAINS PASS COUNT
392	001100	000	\$STNM: .BYTE 00	CONTAINS THE TEST NUMBER
393	001103	000	\$ERFLG: .BYTE 00	CONTAINS ERROR FLAG
394	001104	000000	\$ICNT: .WORD 00	CONTAINS SUBTEST ITERATION COUNT
395	001106	000000	\$LPADR: .WORD 00	CONTAINS SCOPE LOOP ADDRESS
396	001110	000000	\$LPERR: .WORD 00	CONTAINS SCOPE RETURN FOR ERRORS
397	001112	000000	\$ERTTL: .WORD 00	CONTAINS TOTAL ERRORS DETECTED
398	001114	000	\$ITEMB: .BYTE 00	CONTAINS ITEM CONTROL BYTE
399	001115	001	\$ERMAX: .BYTE 1	CONTAINS MAX. ERRORS PER TEST
400	001116	000000	\$ERRPC: .WORD 00	CONTAINS PC OF LAST ERROR INSTRUCTION
401	001120	000000	\$GDADR: .WORD 00	CONTAINS ADDRESS OF 'GOOD' DATA
402	001122	000000	\$BDADR: .WORD 00	CONTAINS ADDRESS OF 'BAD' DATA
403	001124	000000	\$GDDAT: .WORD 00	CONTAINS 'GOOD' DATA
404	001126	000000	\$BDDAT: .WORD 00	CONTAINS 'BAD' DATA
405	001130	000000	.WORD 00	RESERVED--NOT TO BE USED
406	001132	000000	.WORD 00	
407	001134	000	\$AUTOB: .BYTE 00	AUTOMATIC MODE INDICATOR
408	001135	000	\$INTAG: .BYTE 00	INTERRUPT MODE INDICATOR
409	001136	000000	.WORD 0	
410	001140	177570	\$SWR: .WORD DSWR	ADDRESS OF SWITCH REGISTER
411	001142	177570	\$DISPLAY: .WORD DDISP	ADDRESS OF DISPLAY REGISTER
412	001144	177560	\$TKS: 177560	TTY KBD STATUS
413	001146	177562	\$TKB: 177562	TTY KBD BUFFER
414	001150	177564	\$TPS: 177564	TTY PRINTER STATUS REG. ADDRESS
415	001152	177566	\$TPB: 177566	TTY PRINTER BUFFER REG. ADDRESS
416	001154	000	\$NULL: .BYTE 0	CONTAINS NULL CHARACTER FOR FILLS
417	001155	002	\$FILLS: .BYTE 2	CONTAINS # OF FILLER CHARACTERS REQUIRED
418	001156	012	\$FILLC: .BYTE 12	INSERT FILL CHARS. AFTER A "LINE FEED"
419	001157	000	\$TPFLG: .BYTE 0	"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
420	001160	077	\$QUES: .ASCII /?/	QUESTION MARK
421	001161	015	\$CRLF: .ASCII <15>	CARRIAGE RETURN
422	001162	000012	\$LF: .ASCII <12>	LINE FEED
423			*****	
424		000015	CR = 15	
425		000012	LF = 12	
426	001164	000000	\$DRVSEL: .WORD 0	ADDRESS OF SELECTED DRIVE
427	001166	000000	\$DRIVE: .WORD 0	DRIVE NUM STORAGE FOR DRIVER ERR CALLS
428	001170	000000	\$ATTN: .WORD 0	ATTENTION REGISTER STORAGE FOR ERROR MESSAGES
429	001172	000000	\$HEAD: .WORD 0	HEAD ADDRESS STORED FOR ERROR MESSAGES
430	001174	000000	\$TOPLN: .WORD 0	'TYPE THE HEADER' INDICATOR
431	001176	000000	\$OFFDIR: .WORD 0	NEG OFFSET PASS IND
432	001200	000000	\$BITIND: .WORD 0	SIGN CHANGE BIT STORED INDICATOR
433	001202	000000	\$SINCNG: .WORD 0	SIGN CHANGE BIT STORAGE
434	001204	000000	\$PLJS: .WORD 0	STORE THE POS SIGN CHANGE CODE
435	001206	000000	\$INREG: .WORD 0	CONTAINS THE VALUE READ FROM THE REGISTER
436	001210	000000	\$TCLER: .WORD 0	ALIGNMENT TOLERANCE FOR PRESENT CYLINDER

```

444 001212 000000 MAXCYL: .WORD 0 ;CONTAINS THE MAXIMUM CYLINDER ADDRESS
445 ;OF DRIVE UNDER TEST
446 001214 000000 MAXPOS: .WORD 0 ;CONTAINS MAXIMUM POSITIVE OFFSET
447 001216 000000 MAXNEG: .WORD 0 ;CONTAINS MAXIMUM NEGATIVE OFFSET
448 001220 000000 MAXOFF: .WORD 0 ;CONTAINS NUMBER OF OFFSET POSITIONS
449 001222 000000 OUTTOL: .WORD 0 ;CONTAINS TOLERANCE FOR INNER AND OUTER CYLINDERS
450 001224 000000 MIDTOL: .WORD 0 ;CONTAINS TOLERANCE FOR THE ALIGNMENT CYLINDER
451 001226 000000 INCYL: .WORD 0 ;INNER CYLINDER ADDRESS
452 001230 000000 MIDCYL: .WORD 0 ;ALIGNMENT CYLINDER ADDRESS
453 001232 000000 OUTCYL: .WORD 0 ;OUTER CYLINDER ADDRESS
454 001234 000000 TABLE1: .WORD 0 ;CONTAINS HEAD TABLE ADDRESS FOR 'INCYL' & 'OUTCYL'
455 001236 000000 TABLE2: .WORD 0 ;CONTAINS ADDRESS OF OFFSET CODE TABLE
456 001240 000000 SEKCNT: .WORD 0 ;CONTAINS SEEK COUNT
457 001242 000000 CHGADR: .WORD 0 ;CHANGE RH11 BUS ADDRESS FLAG
458 001244 000104 000106 PKV: .WORD 104,106 ;KW11-P VECTOR ADDRESS
459 001250 172540 PKCS: .WORD 172540 ;KW11-P CONTROL AND STATUS REG.
460 001252 172542 PKB: .WORD 172542 ;KW11-P COUNT SET BUFFER
461 001254 172544 PKC: .WORD 172544 ;KW11-P COUNTER
462 001256 000100 000102 LKV: .WORD 100,102 ;KW11-L VECTOR ADDRESS
463 001262 177546 LKS: .WORD 177546 ;KW11-L STATUS REGISTER
464
465 ;:*****
466
467 .SBTTL RH11/RP04/5/6 ADDRESS & VECTOR LOCATIONS
468
469 ;:*****
470
471 001264 176700 $RPADR: .WORD 176700 ;RH11/RP04/5/6 UNIBUS ADDRESS
472 001266 000254 $RPVEC: .WORD 254 ;RH11/RP04/5/6 VECTOR ADDRESS
473

```

.SBTTL ERROR POINTER TABLE

: *THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
 : *THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
 : *LOCATION \$ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
 : *NOTE1: IF \$ITEMB IS 0 THE ONLY PERTINENT DATA IS (\$ERRPC).
 : *NOTE2: EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:

: *	EM	:: POINTS TO THE ERROR MESSAGE
: *	DH	:: POINTS TO THE DATA HEADER
: *	DT	:: POINTS TO THE DATA
: *	DF	:: POINTS TO THE DATA FORMAT

\$ERRTB:

:ERROR 1

001270

	EM1	: RH11 INTERRUPT OCCURRED (RHAS=0)
	DH1	
	DT1	
	DF1	

:ERROR 2

001300	020557
001302	021254
001304	021642
001306	021751

	EM2	: UNEXPECTED ATTENTION OCCURRED
	DH2	
	DT2	
	DF2	

:ERROR 3

001310	020615
001312	021331
001314	021660
001316	021757

	EM3	: MASSBUS PARITY ERROR (MCPE=1)
	DH3	
	DT3	
	DF3	

:ERROR 4

001320	020653
001322	021357
001324	021670
001326	021762

	EM4	: MASSBUS PARITY ERROR (PAR=1)
	DH4	
	DT4	
	DF4	

:ERROR 5

001330	020710
001332	021254
001334	021642
001336	021751

	EM5	: ADDRESS PLUG CHANGE BIT SET
	DH2	
	DT2	
	DF2	

:ERROR 6

DZRJC.009 ERROR POINTER TABLE

528	001340	020744	EM6	:RH11 DIDN'T RESPOND TO ADDRESSING
529	001342	021416	DH6	
530	001344	021702	DT6	
531	001346	021750	DF1	
532				
533			:ERROR 7	
534				
535	001350	000000	0	:UNUSED
536	001352	000000	0	
537	001354	000000	0	
538	001356	000000	0	
539				
540			:ERROR 10	
541				
542	001360	021006	EM10	:DRIVE OR DATA ERROR
543	001362	021431	DH10	
544	001364	021706	DT10	
545	001366	021766	DF10	
546				
547			:ERROR 11	
548				
549	001370	021032	EM11	:DRIVE UNSAFE ERROR
550	001372	021431	DH10	
551	001374	021706	DT10	
552	001376	021766	DF10	
553				
554			:ERROR 12	
555				
556	001400	021055	EM12	:HEAD OUT OF ALIGNMENT
557	001402	021517	DH12	
558	001404	021726	DT12	
559	001406	021775	DF12	
560				
561			:ERROR 13	
562				
563	001410	021103	EM13	:HEAD TOO FAR OUT OF ALIGNMENT
564	001412	021610	DH13	
565	001414	021736	DT13	
566	001416	022000	DF13	
567				
568			:ERROR 14	
569				
570	001420	021141	EM14	:SOFTWARE TIMEOUT
571	001422	021431	DH10	
572	001424	021706	DT10	
573	001426	021766	DF10	
574				
575			:ERROR 15	
576				
577	001430	021162	EM15	:UNCORRECTABLE MASSBUS PARITY ERROR
578	001432	000000	0	
579	001434	000000	0	
580	001436	000000	0	
581				

DZRJC.009 ERROR POINTER TABLE

```

582          ;ERROR 16
583
584          001440 021225          EM16          ;DRIVE IS UNAVAILABLE
585          001442 021627          DH16
586          001444 021744          DT16
587          001446 021750          DF1
588
589          ;;*****
590
591          .SBTTL RPO4/5/6 DRIVER COMMANDS
592
593          ;;*****
594
595          000101          RNOP= 101          ;NO OPERATION
596          000103          UNLOAD= 103          ;UNLOAD
597          000105          SEEK= 105          ;SEEK
598          000107          RECAL= 107          ;RECALIBRATE
599          000111          DRVCLR= 111          ;DRIVE CLEAR
600          000113          REL= 113          ;RELEASE
601          000115          OFFSET= 115          ;OFFSET
602          000117          RTC= 117          ;RETURN TO CENTER LINE
603          000121          PRESET= 121          ;READ IN PRESET
604          000123          ACK= 123          ;PACK ACKNOWLEDGE
605          000131          SEARCH= 131          ;SEARCH
606          000141          GETREG= 141          ;GET REGISTERS
607          000143          SETFMT= 143          ;SET FORMAT (& ECI OR HCI)
608          000145          SELDRV= 145          ;SELECT DRIVE
609          000151          WCHKD= 151          ;WRITE CHECK DATA
610          000153          WCHKHD= 153          ;WRITE CHECK HEADER & DATA
611          000161          WRTDAT= 161          ;WRITE DATA
612          000163          WRTHD= 163          ;WRITE HEADER & DATA
613          000171          RDDAT= 171          ;READ DATA
614          000173          RDHD= 173          ;READ HEADER & DATA
615
616
617          ;;*****
618
619          .SBTTL PROGRAM START AND INITIALIZATION ROUTINES
620
621          ;;*****
622
623          001450 012737 177777 001242 START: MOV #-1,CHGADR          ;SET CHANGE RH11 ADDRESS FLAG
624          001456 000402          BR START2          ;FINISH
625          001460 005037 001242 START1: CLR CHGADR          ;CLEAR RH11 CHANGE ADDRESS FLAG
626          001464 000005          START2: RESET          ;CLEAR THE BUS
627          .SBTTL INITIALIZE THE COMMON TAGS
628          ;;CLEAR THE COMMON TAGS (SCMTAG) AREA
629          001466 012706 001100          MOV #SCMTAG,R6          ;FIRST LOCATION TO BE CLEARED
630          001472 005026          CLR (R6)+          ;CLEAR MEMORY LOCATION
631          001474 022706 001140          CMP #SWR,R6 ;:DONE?
632          001500 001374          BNE -6          ;LOOP BACK IF NO
633          001502 012706 001100          MOV #STACK,SP          ;SETUP THE STACK POINTER
634          .:INITIALIZE A FEW VECTORS
635          001506 012737 004412 000030          MOV #ERROR,EMTVEC ;:EMT VECTOR FOR ERROR ROUTINE

```


002400	001004		BNE	85		:BR IF 'E' NOT ENTERED
002402	000137	003640	JMP	RANDOM		:GO TO EXERCISE ROUTINE
002406	104401	002442	TYPE	.105		:ECHO THE INVALID CHARACTER
002412	104401	001160	85:	TYPE	.50UES	:TYPE A QUESTION MARK
002416	000137	001744	JMP	INIT		:GO ALL THE WAY BACK TO DRIVE ENT
002422	104401	000025	95:	TYPE	.VERIFY	:TYPE 'VERIFY'
002426	002777	000004	95:	9IT	.SW2,3SWR	:SWITCH 2 SET ?
002434	001524	176504	BEG	CYLDER		:BR IF NOT SET - CHECK ALL HEADS
002436	000137	002444	JMP	ONEHD		:CHECK ONLY THE SPECIFIED HEAD
002440	000000		105:	.WORD	0	:OPERATOR ENTRY GOES HERE

.SBTTL SETUP TO CHECK ONLY THE SPECIFIED HEAD

002444	104401	002265	ONEHD:	TYPE	ENTCYL	:ASK FOR CYLINDER ADDRESS
002446	000137	004270	JSR	PC.GETNUM		:GET THE CYLINDER ADDRESS
002450	000137		BR	15		: 'CR' ENTERED
002454	012637	016132	MOV	(SP)+,DPB+CYL		:MOVE IT TO PARAMETER BLOCK
002458	012737	001223	MOV	OUTTOL,TOLER		:SET INITIAL ALIGNMENT TOLERANCE
002462	002777	001232	CMP	OUTCYL,DPB+CYL		:SEE IF OUTER CYLINDER
002466	001432		BEG	45		:BR IF IT IS
002470	002777	001226	CMP	INCYL,DPB+CYL		:SEE IF INNER CYLINDER
002474	001426		BEG	45		:BR IF IT IS
002478	013737	001224	MOV	MIDTOL,TOLER		:CHANGE TOLERANCE
002482	002777	001230	CMP	MIDCYL,DPB+CYL		:SEE IF ALIGNMENT CYLINDER
002486	001414		BEG	35		:BR IF YES
002490	104401	020447	TYPE	BADCYL		:REPORT INVALID CYLINDER
002494	000744		BR	ONEHD		:TRY IT AGAIN
002498	002777	001212	13:	CMP	MAXCYL,411.	:RPO4'S ?
002502	001403		BEG	25		:BR IF YES
002506	104401	020166	TYPE	CYL496		:CYLINDER 496 DEFAULT MESSAGE
002510	000432		BR	35		:CONTINUE
002514	104401	020122	125:	TYPE	CYL245	:CYLINDER 245 DEFAULT MESSAGE
002518	012737	001230	125:	MOV	MIDCYL,DPB+CYL	:CYLINDER ADDRESS
002522	104401	002246	45:	TYPE	ENTHO	:ASK FOR THE HEAD ADDRESS
002526	004737	004270	JSR	PC.GETNUM		:GET THE HEAD ADDRESS
002530	000411		BR	55		: 'CR' ENTERED
002534	012637	001172	MOV	(SP)+,\$HEAD		:SAVE THE HEAD ADDRESS
002538	002777	000022	CMP	19,\$HEAD		:ADDRESS VALID ?
002542	002007		BEG	65		:BR IF YES
002546	104401	020420	TYPE	BADTRK		:INVALID HEAD ADDRESS MESSAGE
002550	000762		BR	45		:TRY AGAIN
002554	104401	020064	55:	TYPE	HOZERO	:HEAD DEFAULT MESSAGE
002558	005037	001172	CLR	\$HEAD		:CLEAR HEAD STORAGE
002562	004537	003736	65:	JSR	R5 DRIVER	:SEEK TO THE SPECIFIED CYLINDER
002566	000137		JSR	SEEK		:SEEK CODE
002570	000137	003032	JSR	PC.CHECK		:CHECK THE HEAD
002574	000137	001744	JMP	INIT		:GET THE NEXT DRIVE HEAD

```
.SBTTL  MAIN ROUTINE - CHECK ALL HEADS AT ALL ALIGNMENT CYLINDERS
```

•••••

Address	Hex	Hex	Hex	Hex	Label	Assembly	Comment
002646	013737	001230	016132		CYLDER:	MOV	MIDCYL,DPB+CYL
002654	013737	001224	001210			MOV	MIDTOL,TOLER
002662	004537	003736				JSR	R5,DRIVER
002666	000105					.WORD	SEEK
002670	012704	016210				MOV	#HEAD1,R4
002674	004737	003014				JSR	PC,15
002700	013737	001226	016132			MOV	INCYL,DPB+CYL
002706	013737	001222	001210			MOV	OUTTOL,TOLER
002714	004537	003736				JSR	R5,DRIVER
002720	000105					.WORD	SEEK
002722	013704	001234				MOV	TABLE1,R4
002726	004737	003014				JSR	PC,15
002732	013737	001232	016132			MOV	OUTCYL,DPB+CYL
002740	004537	003736				JSR	R5,DRIVER
002744	000105					.WORD	SEEK
002746	004737	003014				JSR	PC,15
002752	013737	001230	016132			MOV	MIDCYL,DPB+CYL
002760	013737	001224	001210			MOV	MIDTOL,TOLER
002766	004537	003736				JSR	R5,DRIVER
002772	000105					.WORD	SEEK
002774	012704	016210				MOV	#HEAD1,R4
002800	004737	003014				JSR	PC,15
002804	004401	002050				TYPE	#ENDMSG
002808	000105					.WORD	15
002812	000105					.WORD	15
002816	000105					.WORD	15
002820	000105					.WORD	15
002824	000105					.WORD	15
002828	000105					.WORD	15
002832	000105					.WORD	15
002836	000105					.WORD	15
002840	000105					.WORD	15
002844	000105					.WORD	15
002848	000105					.WORD	15
002852	000105					.WORD	15
002856	000105					.WORD	15
002860	000105					.WORD	15
002864	000105					.WORD	15
002868	000105					.WORD	15
002872	000105					.WORD	15
002876	000105					.WORD	15
002880	000105					.WORD	15
002884	000105					.WORD	15
002888	000105					.WORD	15
002892	000105					.WORD	15
002896	000105					.WORD	15
002900	000105					.WORD	15
002904	000105					.WORD	15
002908	000105					.WORD	15
002912	000105					.WORD	15
002916	000105					.WORD	15
002920	000105					.WORD	15
002924	000105					.WORD	15
002928	000105					.WORD	15
002932	000105					.WORD	15
002936	000105					.WORD	15
002940	000105					.WORD	15
002944	000105					.WORD	15
002948	000105					.WORD	15
002952	000105					.WORD	15
002956	000105					.WORD	15
002960	000105					.WORD	15
002964	000105					.WORD	15
002968	000105					.WORD	15
002972	000105					.WORD	15
002976	000105					.WORD	15
002980</							

.SEVEN ROUTINE TO FIND THE TRACK CENTER

[illegible]

Symbol	Symbol	Symbol
STACK:	OFFDIR	CLEAR THE OFFSET DIRECTION INDICATOR
	BIT: NO	CLEAR THE SIGN BIT INDICATOR
	MAXPOS, R2	MAXIMUM POSITIVE OFFSET VALUE
	MAXOFF, R3	MAXIMUM NUMBER OF OFFSETS
	TABLE, R1	OFFSET CODE TABLE ADDRESS
	HEAD, - SP	HEAD VALUE TO STACK
	SW, 1	SWITCH FOR RPOA LOAD
	R2, R1, R0	LOAD RPOA WITH HEAD ADDRESS
		REGISTER ADDRESS
		UNCORRECTABLE PARITY ERROR RETURN
		OFFSET VALUE TO PARAMETER BLOCK
		DECREMENT OFFSET VALUE
		IF PLUS REVERSE DIRECTION
		IF PLUS OFFSET

```

PR-76 16:34 PAGE 19

: INCREMENT OFFSET VALUE
: OFFSET THE DRIVE
: OFFSET OP CODE
: INITIAL SIGN VALUE STORED ?
: BR IF STORED
: SET THE SIGN BIT INDICATOR
+RPOF : LEAVE THE SIGN BIT
CNG : STORE THE SIGN CHANGE BIT

+RPOF : KEEP ONLY THE SIGN VALUE
CNG : SIGN CHANGED ?
: BRANCH IF IT DID
: DECREMENT THE OFFSET ATTEMPT COUNTER
: BRANCH IF OFFSETS NOT COMPLETED
: CHECK SWRD
: BR IF SET
: REPORT OUT OF LIMIT CONDITION
: SEE IF LOOP ON ERROR
: TYPE THE HEADING LINE ?
: BR IF NOT
: TYPE THE HEADER
: SET THE BYPASS SWITCH

: : SAVE $HEAD FOR TYPEOUT
: : GO TYPE--DECIMAL ASCII WITH SIGN
: : SAVE DPB+CYL FOR TYPEOUT
: : GO TYPE--DECIMAL ASCII WITH SIGN
: : HEAD OUT OF LIMIT
: : CR-LF
: : HALT IF ERROR ?
: : BR IF NOT
: : HEAD JUST TYPED OUT OF TOLERANCE
: : CHECK FOR LOOP
: : SEE WHICH DIRECTION
: : BRANCH IF FORWARD
: : REVERSE DIRECTION
: : FINISHED WITH REVERSE ?
: : BRANCH IF YES
: : SAVE THE SIGN CHANGE VALUE
: : MAXIMUM NEGATIVE OFFSET VALUE
: : NUMBER OF OFFSETS
: : NUMBER OF OFFSETS
: : BEGINNING OF REVERSE DIRECTION VALUES
: : SET REVERSE DIRECTION INDICATOR
: : RESET SIGN BIT INDICATOR
: : START REVERSE OFFSETS
: : ADD THE TWO SIGN CHANGE VALUES
: : 'DIVIDE' BY 2
: : SAVE THE RESULT
: : SEE IF VALUE NEGATIVE
: : BRANCH IF POSITIVE
: : RECOMPLEMENT THE VALUE
: : SEE IF SWRD SET
: : BRANCH IF IT IS

```



```

003630 003636 004037 015012      JSR      RD.WRT.RP      ;LOAD RPO4
003631 003632 000006          RPO4          ;REGISTER ADDRESS
003632 003633 001744          INIT        ;UNCORRECTABLE PARITY ERROR RETURN
003633 003636 000755          BR         3$      ;WAIT FOR NEXT HEAD ENTRY

;*****
.SBTTL ROUTINE TO PERFORM 5,000 RANDOM SEEKS
;*****

003640 104401 020050      RANDOM: TYPE      ,EXER      ;TYPE 'EXERCISE'
003641 012737 011610 001240      MOV      #5000,SEKNT ;NUMBER OF SEEKS
003642 004737 007034      1$:      JSR      PC,SRAND ;CYCLE THE RANDOM NUMBER GENERATOR
003643 013746 007132      MOV      $HNUM,-(SP) ;PUT UPPER RANDOM NUMBER ON THE STACK
003644 005046          CLR      -(SP) ;UPPER DIVIDEND
003645 013746 001212      MOV      MAXCYL,-(SP) ;PUT DIVISOR ON STACK
003646 004737 007136      JSR      PC,SDIV ;GET THE REMAINDER (RANDOM CYLINDER ADDRESS)
003647 021637 016132      CMP      (SP),DPB+CYL ;SEE IF SAME RANDOM CYLINDER AS LAST
003648 001003          BNE      2$ ;BR IF DIFFERENT CYLINDER
003649 006270 000004      ADD      #4,SP ;ADJUST THE STACK POINTER
003650 000761          BR         1$ ;TRY AGAIN
003651 012637 016132      2$:      MOV      (SP)+,DPB+CYL ;PUT CYL ADDR IN THE DPB
003652 005726          TST      (SP)+ ;CORRECT THE STACK PCINTER
003653 004537 003736      JSR      RD.DRIVER ;DO THE SEEK
003654 000105          .WORD      SEEK ;SEEK CODE
003655 005337 001240      DEC      SEKNT ;DECREMENT THE SEEK COUNT
003656 001350          BNE      1$ ;BR IF NOT DONE
003657 000137 001744      JMP      INIT ;RETURN TO COMMAND ROUTINE

;*****
.SBTTL COMMON ENTRY TO THE RPO4 5 6 DRIVER
;*****

003736 012537 016122      DRIVER: MOV      (R5)+,DPB+COMND ;COMMAND CODE
003737 004037 010472      1$:      JSR      RD.RPO4 ;DRIVER ENTRY
003738 016120          DPB          ;DATA PARAMETER BLOCK ADDRESS
003739 000774          BR         1$ ;REQUEST NOT ACCEPTED
003740 005737 016136      2$:      TST      DPB+STATUS ;SEE IF ORDER COMPLETE
003741 001775          BEQ      2$ ;BR IF NOT COMPLETE
003742 100401          BMI      4$ ;BRANCH IF ERROR
003743 000205      3$:      RTS      R5 ;RETURN
003744 032737 000200 016136      4$:      BIT      #BIT7,DPB+STATUS ;DID THE ORDER TERMINATE
003745 001020          BNE      5$ ;BRANCH IF IT DID
003746 032737 060006 016136      BIT      #BIT14!BIT13!BIT02!BIT01,DPB+STATUS ;DRIVE OFFLINE OR
003747          ;NON-EXISTENT
003748          BNE      6$ ;BR IF IT IS
003749 032737 010000 016136      BIT      #BIT12,DPB+STATUS ;DRIVE UNSAFE
003750 001022          BNE      7$ ;BR IF UNSAFE
003751 032737 006000 016136      BIT      #BIT11!BIT10,DPB+STATUS ;UNCORRECTABLE PARITY ERROR
003752 001020          BNE      8$ ;BR IF YES
003753 032737 001400 016136      BIT      #BIT09!BIT08,DPB+STATUS ;TIMEOUT ON THIS DRIVE

```

```

1014 004032 001016      BNE      9$          :BR IF YES
1015 004034 104010      :REPORT DRIVE OR DATA ERROR
1016 004036 032777 001000 175074 5$: ERROR 10 :LOOP ON ERROR ?
1017 004044 001746      BIT      #SW09,3SWP :BR IF NOT
1018 004046 162705 000005      BEQ      3$          :CORRECT R5 FOR LOOP
1019 004048      SLB      #6,R5 :RETURN
1020 004052 000743      BR      3$          :REPORT DRIVE UNAVAILABLE
1021 004054 104016      6$: ERROR 16 :CHECK FOR LOOP
1022 004056 000405      BR      10$         :REPORT PERSISTENT UNSAFE
1023 004060 104011      7$: ERROR 11 :CHECK FOR LOOP
1024 004062 000403      BR      10$         :UNCORRECTABLE PARITY ERROR
1025 004064 104015      9$: ERROR 15 :CHECK FOR LOOP
1026 004066 000401      BR      10$         :SOFTWARE TIMEOUT
1027 004070 104014      9$: ERROR 14 :LOOP ON ERROR ?
1028 004072 032777 001000 175040 10$: BIT #SW09,3SWP:LOOP
1029 004100 001403      BEQ      11$         :BR IF NOT
1030 004102 162705 000005      SUB      #6,R5 :CORRECT R5 FOR LOOP
1031 004106 000725      BR      3$          :RETURN
1032 004110 000137 001744      11$: JMP      INIT :RETURN TO USER
1033
1034 :*****
1035 .SBTTL SUBROUTINES
1036
1037 :*****
1038
1039 :THIS ROUTINE WILL DETERMINE IF THERE IS A CLOCK ON THE SYSTEM
1040 :AND IF THERE IS IT WILL SETUP THE VECTOR AND START THE CLOCK.
1041 :CALL
1042 :      JSR      PC,2$ST.CLK
1043 :      RETURN
1044
1045 ST.CLK: MOV      R1,-(SP) :SAVE R1
1046 004116 012701 000006      MOV      #EARVEC+2,R1 :SAVE AND SETUP TIMEOUT VECTOR
1047 004122 011146      MOV      (R1),-(SP)
1048 004124 005011      CLR      (R1) :LEVEL 0
1049 004126 014146      MOV      -(R1),-(SP)
1050 004130 012711 004146      MOV      #1$,(R1) :GO TO 1$ ON TIMEOUT
1051 004134 005777 175110      TST      2PKCS :IS THERE A KW11-P?
1052 004140 004737 004200      JSR      PC,ST.PCLK :START THE KW11-P
1053 004144 000411      BR      3$          :GO TO EXIT
1054 004146 022626      1$: CMP      (SP)+,(SP)+ :CLEAN UP THE STACK
1055 004150 012711 004166      MOV      #2$,(R1) :IF TIMEOUT GO TO 2$
1056 004154 005777 175102      TST      2LKS :IS THERE A KW11-L?
1057 004160 004737 004232      JSR      PC,ST.LCLK :START THE KW11-L
1058 004164 000401      BR      3$          :EXIT
1059 004166 022626      2$: CMP      (SP)+,(SP)+ :CLEAN UP THE STACK
1060 004170 012621 3$: MOV      (SP)+,(R1)+ :RESTORE THE TIMEOUT VECTOR
1061 004172 012621      MOV      (SP)+,(R1)+
1062 004174 012601      MOV      (SP)+,R1 :RESTORE R1
1063 004176 000207      RTS      PC :RETURN
1064
1065 004200 012777 004256 175036 ST.PCLK: MOV #SRVCLK,2PKV :SETUP THE KW11-P VECTOR
1066 004206 012777 000300 175032      MOV      #300,3PKV+2
1067 004214 012777 000001 175030      MOV      #1,3PFB :COUNT ONE TICK

```

```

1068 004222 012777 000115 175020      MOV      #115,SPKCS      ;"INT.EN." COUNT DOWN", "MODE 1 (REPEAT)".
1069                                ;"LINE FREQ", AND "RUN"
1070 004230 000207                        RTS      PC              ;RETURN
1071
1072 004232 012777 004256 175016 ST.LCLK: MOV      #SRVCLK,ALKV      ;SETUP THE KW11-L VECTOR
1073 004240 012777 000300 175012      MOV      #300,ALKV+2
1074 004246 012777 000100 175006      MOV      #100,ALKS      ;START THE KW11-L
1075 004254 000207                        RTS      PC              ;RETURN
1076
1077 004256 012746 000020      SRVCLK: MOV      #16, -(SP)      ;TIME PER TICK IN MILLISECONDS
1078 004262 004737 014110      JSR      PC,SRPTMR      ;COUNT THE ELAPSED TIME
1079 004266 000002      RTI                          ;RETURN AFTER INTERRUPT
1080
1081                                ;ROUTINE TO GET DECIMAL KEYBOARD ENTRIES
1082                                ;CALL
1083                                ;JSR      PC,GETNUM
1084                                ;RETURN1
1085                                ;RETURN2
1086
1087 004270 011646      GETNUM: MOV      (SP), -(SP)      ;PROVIDE SPACE FOR FIRST CHAR
1088 004272 104411 1$:      ROLIN      ;READ AN ASCII LINE
1089 004274 012600      MOV      (SP)+,R0      ;ADDRESS OF 1ST CHAR
1090 004276 010037 004376      MOV      R0,6$      ;SAVE IN CASE OF BAD INPUT
1091 004302 105710      TSTB      (R0)      ;FIRST CHARACTER A 'CR' ?
1092 004304 001440      BEQ      7$      ;IF IT IS
1093 004306 005046      CLR      -(SP)      ;CLEAR DATA WORD
1094 004310 112001 2$:      MOVB      (R0)+,R1      ;PICKUP THIS CHARACTER
1095 004312 001421      BEQ      4$      ;GET OUT IF ZERO
1096 004314 122701 000060      CMPB      #'0,R1      ;MAKE SURE THAT THIS CHARACTER
1097 004320 003016      BGT      4$      ;IS A DIGIT BETWEEN 0 & 9
1098 004322 122701 000071      CMPB      #'9,R1
1099 004326 002420      BLT      5$
1100 004330 006316      ASL      (SP)      ;*2
1101 004332 011646      MOV      (SP), -(SP)      ;SAVE FOR LATER
1102 004334 006316      ASL      (SP)      ;*4
1103 004336 006316      ASL      (SP)      ;*8
1104 004340 062616      ADD      (SP)+, (SP)      ;*10.
1105 004342 102412      BVS      5$      ;OVERFLOW ISN'T ALLOWED
1106 004344 042701 000060      BIC      #60,R1      ;CLEAR THE UPPER BITS
1107 004350 060116      ADD      R1,(SP)      ;ADD THE NUMBER
1108 004352 102406      BVS      5$      ;OVERFLOW ISN'T ALLOWED
1109 004354 000755      BR      2$      ;LOOP
1110 004356 012666 000002 4$:      MOV      (SP)+,2,(SP)      ;SAVE THE RESULT
1111 004362 062716 000002      ADD      #2,(SP)      ;INCREMENT THE RETURN ADDRESS
1112 004366 000410      BR      8$      ;EXIT
1113 004370 005726 5$:      TST      (SP)+      ;CLEAN PARTIAL NUMBER FROM STACK
1114 004372 105010      CLRB      (R0)      ;SET A TERMINATOR
1115 004374 104401      TYPE      ;TYPE THE INPUT UP TO BAD CHAR
1116 004376 000000 6$:      .WORD      0      ;POINTER GOES HERE
1117 004400 104401 001160      TYPE      $GLES      ;'?' 'CR' 'LF'
1118 004404 000732      BR      1$      ;TRY AGAIN
1119 004406 012616 7$:      MOV      (SP)+, (SP)      ;RESTORE THE PC
1120 004410 000207 8$:      RTS      PC              ;RETURN

```

```

1132      ;;*****
1133      .SBTTL  MACRO ROUTINES
1134
1135      .SBTTL  ERROR HANDLER ROUTINE
1136
1137      ;;*****
1138      ;;THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT.
1139      ;;SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
1140      ;;AND GO TO SERRTYP ON ERROR
1141      ;;THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
1142      ;;SW15=1      HALT ON ERROR
1143      ;;SW13=1      INHIBIT ERROR TYPEOUTS
1144      ;;CALL
1145      ;;      ERPOR  N      ;;ERROR=EMT AND N=ERROR ITEM NUMBER
1146
1138      $ERROR:
1139      CKSWR      ;;TEST FOR CHANGE IN SOFT-SWR
1140      MOV        R1,DRIVE
1141      MOV        R3,ATTN
1142      7$:      INCB  $ERFLG      ;;SET THE ERROR FLAG
1143      BEQ        7$      ;;DON'T LET THE FLAG GO TO ZERO
1144      MOV        $TSTNM,$DISPLAY  ;;DISPLAY TEST NUMBER AND ERROR FLAG
1145      INC        $ERTTL      ;;INC THE ERROR COUNT
1146      MOV        (SP),$ERRPC      ;;GET ADDRESS OF ERROR INSTRUCTION
1147      SUB        #2,$ERRPC
1148      MOVB      2,$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
1149      BIT        #BIT13,$SWR      ;;SKIP TYPEOUT IF SET
1150      BNE        20$      ;;SKIP TYPEOUTS
1151      JSR        PC,$SERRTYP      ;;GO TO USER ERROR ROUTINE
1152      TYPE      $CRLF
1153
1154      20$:      TST  2$SWR      ;;HALT ON ERROR
1155      BPL        3$      ;;SKIP IF CONTINUE
1156      HALT      ;;HALT ON ERROR!
1157      CKSWR      ;;TEST FOR CHANGE IN SOFT-SWR
1158
1159      3$:      RTI      ;;RETURN
1160      .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
1161
1162      ;;*****
1163      ;;THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
1164      ;;ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
1165      ;;AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
1166
1167      $SERRTYP:
1168      TYPE      $CRLF      ;;"CARRIAGE RETURN" & "LINE FEED"
1169      MOV        R0,-(SP)      ;;SAVE R0
1170      CLR        R0      ;;PICKUP THE ITEM INDEX
1171      BISB      2,$ITEMB,R0
1172      BNE        1$      ;;IF ITEM NUMBER IS ZERO, JUST
1173      ;;TYPE THE PC OF THE ERROR
1174      MOV        $ERRPC,-(SP)      ;;SAVE $ERRPC FOR TYPEOUT
1175      ;;ERROR ADDRESS

```

```

1176 004542 104402          TYP0C          ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1177 004544 000445          BR 10$          ;;GET OUT
1178 004546 005300          1$: DEC R0       ;;ADJUST THE INDEX SO THAT IT WILL
1179 004550 006300          ASL R0          ;;      WORK FOR THE ERROR TABLE
1180 004552 006300          ASL R0
1181 004554 006300          ASL R0
1182 004556 062700 001270    ADD #ERRTB,R0  ;;FORM TABLE POINTER
1183 004562 012037 004572    MOV (R0)+,2$  ;;PICKUP "ERROR MESSAGE" POINTER
1184 004566 001404          BEQ 3$          ;;SKIP TYPEOUT IF NO POINTER
1185 004570 104401          TYPE           ;;TYPE THE "ERROR MESSAGE"
1186 004572 000000          2$: .WORD 0     ;;"ERROR MESSAGE" POINTER GOES HERE
1187 004574 104401 001161    TYPE , $CRLF  ;;"CARRIAGE RETURN" & "LINE FEED"
1188 004600 012037 004610    3$: MOV (R0)+,4$ ;;PICKUP "DATA HEADER" POINTER
1189 004604 001404          BEQ 5$          ;;SKIP TYPEOUT IF 0
1190 004606 104401          TYPE           ;;TYPE THE "DATA HEADER"
1191 004610 000000          4$: .WORD 0     ;;"DATA HEADER" POINTER GOES HERE
1192 004612 104401 001161    TYPE , $CRLF  ;;"CARRIAGE RETURN" & "LINE FEED"
1193 004616 010146          5$: MOV R1,-(SP) ;;SAVE R1
1194 004620 012001          MOV (R0)+,R1    ;;PICKUP "DATA TABLE" POINTER
1195 004622 001415          BEQ 9$          ;;BR IF NO DATA TO BE TYPED
1196 004624 012000          MOV (R0)+,R0    ;;PICKUP "DATA FORMAT" POINTER
1197 004626 105720          6$: TSTB (R0)+  ;;"OCTAL" OR "DECIMAL"
1198 004630 001003          BNE 7$          ;;BR IF DECIMAL
1199 004632 013146          MOV 2(R1)+,-(SP) ;;SAVE 2(R1)+ FOR TYPEOUT
1200 004634 104402          TYP0C          ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1201 004636 000402          BR 9$
1202 004640          7$: MOV 2(R1)+,-(SP)    ;;SAVE 2(R1)+ FOR TYPEOUT
1203 004640 013146          MOV TYPDS       ;;GO TYPE--DECIMAL ASCII WITH SIGN
1204 004642 104405          8$: TST (R1)    ;;IS THERE ANOTHER NUMBER?
1205 004644 005711          BEQ 9$          ;;BR IF NO
1206 004646 001403          TYPE 11$       ;;TYPE TWO(2) SPACES
1207 004650 104401 004670    BR 6$        ;;LOOP
1208 004654 000764
1209
1210 004556 012601          9$: MOV (SP)+,R1  ;;RESTORE R1
1211 004660 012600          10$: MOV (SP)+,R0 ;;RESTORE R0
1212 004562 104401 001161    TYPE , $CRLF  ;;"CARRIAGE RETURN" & "LINE FEED"
1213 004566 000207          RTS PC         ;;RETURN
1214 004670 020040 000      11$: .ASCIZ / /  ;;TWO(2) SPACES
1215 004674
1216
1217 .SBTTL TYPE ROUTINE
1218
1219 ;;*****
1220 ;;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
1221 ;;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
1222 ;;*NOTE1: $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
1223 ;;*NOTE2: $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
1224 ;;*NOTE3: $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
1225 ;;*
1226 ;;*CALL:
1227 ;;*1) USING A TRAP INSTRUCTION
1228 ;;* TYPE ,MESADR ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
1229 ;;*
1230 ;;* TYPE

```

```

1230      ;*      MESADR
1231      ;*
1232
1233      004674 105737 001157      $TYPE:  TSTB      $TFPLG      ;; IS THERE A TERMINAL?
1234      004700 100002              BPL          1$          ;; BR IF YES
1235      004702 000000              HALT              ;; HALT HERE IF NO TERMINAL
1236      004704 000407              BR          3$          ;; LEAVE
1237      004706 010046      1$:      MOV      RO, -(SP)      ;; SAVE RO
1238      004710 017600 000002      MOV      @2(SP), RO      ;; GET ADDRESS OF ASCII STRING
1239      004714 112046      2$:      MOVB     (RO)+, -(SP)      ;; PUSH CHARACTER TO BE TYPED ONTO STACK
1240      004716 001005              BNE          4$          ;; BR IF IT ISN'T THE TERMINATOR
1241      004720 005726              TST      (SP)+          ;; IF TERMINATOR POP IT OFF THE STACK
1242      004722 012600      60$:      MOV      (SP)+, RO      ;; RESTORE RO
1243      004724 062716 000002      3$:      ADD      #2, (SP)      ;; ADJUST RETURN PC
1244      004730 000002              RTI              ;; RETURN
1245      004732 122716 000011      4$:      CMPB     #HT, (SP)      ;; BRANCH IF <HT>
1246      004736 001430              BEQ          8$          ;;
1247      004740 122716 000200              CMPB     #CRLF, (SP)      ;; BRANCH IF NOT <CRLF>
1248      004744 001005              BNE          5$          ;;
1249      004746 005726              TST      (SP)+          ;; POP <CR><LF> EQUIV
1250      004750 104401              TYPE              ;; TYPE A CR AND LF
1251      004752 001161              $CRLF
1252      004754 105037 005110              CLRB     $CHARCNT      ;; CLEAR CHARACTER COUNT
1253      004760 000755              BR          2$          ;; GET NEXT CHARACTER
1254      004762 004737 005044      5$:      JSR      PC, $TYPEC      ;; GO TYPE THIS CHARACTER
1255      004766 123726 001156      6$:      CMPB     $FILLC, (SP)+      ;; IS IT TIME FOR FILLER CHARS.?
1256      004772 001350              BNE          2$          ;; IF NO GO GET NEXT CHAR.
1257      004774 013746 001154              MOV      $NULL, -(SP)      ;; GET # OF FILLER CHARS. NEEDED
1258                                ;; AND THE NULL CHAR.
1259      005000 105366 000001      7$:      DECB     1(SP)      ;; DOES A NULL NEED TO BE TYPED?
1260      005004 002770              BLT          6$          ;; BR IF NO--GO POP THE NULL OFF OF STACK
1261      005006 004737 005044              JSR      PC, $TYPEC      ;; GO TYPE A NULL
1262      005012 105337 005110              DECB     $CHARCNT      ;; DO NOT COUNT AS A COUNT
1263      005016 000770              BR          7$          ;; LOOP
1264
1265      ;HORIZONTAL TAB PROCESSOR
1266
1267      005020 112716 000040      8$:      MOVB     #', (SP)      ;; REPLACE TAB WITH SPACE
1268      005024 004737 005044      9$:      JSR      PC, $TYPEC      ;; TYPE A SPACE
1269      005030 132737 000007 005110      BITB     #7, $CHARCNT      ;; BRANCH IF NOT AT
1270      005036 001372              BNE          9$          ;; TAB STOP
1271      005040 005726              TST      (SP)+          ;; POP SPACE OFF STACK
1272      005042 000724              BR          2$          ;; GET NEXT CHARACTER
1273      005044 105777 174100      $TYPEC:  TSTB     @2$TPS      ;; WAIT UNTIL PRINTER IS READY
1274      005050 100375              BPL          $TYPEC
1275      005052 116677 000002 174072      MOVB     2(SP), @2$TPB      ;; LOAD CHAR TO BE TYPED INTO DATA REG.
1276      005060 122766 000015 000002      CMPB     #CR, 2(SP)      ;; IS CHARACTER A CARRIAGE RETURN?
1277      005066 001003              BNE          1$          ;; BRANCH IF NO
1278      005070 105037 005110              CLRB     $CHARCNT      ;; YES--CLEAR CHARACTER COUNT
1279      005074 000406              BR          $TYPEX      ;; EXIT
1280      005076 122766 000012 000002      1$:      CMPB     #LF, 2(SP)      ;; IS CHARACTER A LINE FEED?
1281      005104 001402              BEQ          $TYPEX      ;; BRANCH IF YES
1282      005106 105227              INCB     (PC)+          ;; COUNT THE CHARACTER
1283      005110 000000      $CHARCNT: .WORD 0      ;; CHARACTER COUNT STORAGE

```

```

1284 005112 000207      STYPEX: R1S      PC
1285
1286      .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
1287
1288      ;*****
1289      ;THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
1290      ;OCTAL (ASCII) NUMBER AND TYPE IT.
1291      ;$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
1292      ;CALL:
1293      ;      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
1294      ;      TYPOS      ;;CALL FOR TYPEOUT
1295      ;      .BYTE      N      ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
1296      ;      .BYTE      M      ;;M=1 OR 0
1297      ;      ;;1=TYPE LEADING ZEROS
1298      ;      ;;0=SUPPRESS LEADING ZEROS
1299
1300      ;$TYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
1301      ;$TYPOS OR $TYPOC
1302      ;CALL:
1303      ;      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
1304      ;      TYPON      ;;CALL FOR TYPEOUT
1305
1306      ;$TYPCC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
1307      ;CALL:
1308      ;      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
1309      ;      TYPOC      ;;CALL FOR TYPEOUT
1310
1311 005114 017646 000000      $TYPOS: MOV      2(SP),-(SP)      ;;PICKUP THE MODE
1312 005120 116637 000001 005337      MOVVB      1(SP), $OFILL      ;;LOAD ZERO FILL SWITCH
1313 005126 112637 005341      MOVVB      (SP)+, $OMODE+1      ;;NUMBER OF DIGITS TO TYPE
1314 005132 062716 000002      ADD      #2, (SP)      ;;ADJUST RETURN ADDRESS
1315 005136 000406      BR      $TYPON
1316 005140 112737 000001 005337      $TYPOC: MOVVB      #1, $OFILL      ;;SET THE ZERO FILL SWITCH
1317 005146 112737 000006 005341      MOVVB      #6, $OMODE+1      ;;SET FOR SIX(6) DIGITS
1318 005154 112737 000005 005336      $TYPON: MOVVB      #5, $OCNT      ;;SET THE ITERATION COUNT
1319 005162 010346      MOV      R3,-(SP)      ;;SAVE R3
1320 005164 010446      MOV      R4,-(SP)      ;;SAVE R4
1321 005166 010546      MOV      R5,-(SP)      ;;SAVE R5
1322 005170 113704 005341      MOVVB      $OMODE+1, R4      ;;GET THE NUMBER OF DIGITS TO TYPE
1323 005174 005404      NEG      R4
1324 005176 062704 000006      ADD      #5, R4      ;;SUBTRACT IT FOR MAX. ALLOWED
1325 005202 110437 005340      MOVVB      R4, $OMODE      ;;SAVE IT FOR USE
1326 005206 113704 005337      MOVVB      $OFILL, R4      ;;GET THE ZERO FILL SWITCH
1327 005212 016605 000012      MOV      12(SP), R5      ;;PICKUP THE INPUT NUMBER
1328 005216 005003      CLR      R3      ;;CLEAR THE OUTPUT WORD
1329 005220 006105      1$: ROL      R5      ;;ROTATE MSB INTO "C"
1330 005222 000404      BR      2$      ;;GO DO MSB
1331 005224 006105      2$: ROL      R5      ;;FORM THIS DIGIT
1332 005226 006105      ROL      R5
1333 005230 006105      ROL      R5
1334 005232 010503      MOV      R5, R3
1335 005234 006103      3$: ROL      R3      ;;GET LSB OF THIS DIGIT
1336 005236 105337 005340      DEC8      $OMODE      ;;TYPE THIS DIGIT
1337 005242 100016      BPL      7$      ;;BR IF NO

```



```

      BLT      45
      INC      R2
      BR      35
45:      ADD     R1,R5
      TST      R2
      BNE      55
      TSTB     (SP)
      BMI      75
55:      ASLB     (SP)
      BCC      65
      MOVB     1(SP),-1(R3)
65:      BIS      0,R3
      MOVB     R2,(R3)+
      TST      (R3)+
      CMP      R0,#10
      BPL      75
      MOV      R5,R2
      BR      65
85:      TSTB     (SP)+
      BPL      95
95:      MOVB     -1(SP),-2(R3)
      CLRB      (R3)
      MOV      (SP)+,R5
      MOV      (SP)+,R3
      MOV      (SP)+,R2
      MOV      (SP)+,R1
      MOV      (SP)+,R0
      TYPE      SCBLK
      MOV      2(SP),4(SP)
      MOV      (SP)+,(SP)
      RTI
      SCBLK:    10000.
                1000.
                100.
                10.
      SCBLK:    .BLKW 4
      .EBTT_    TTY INPUT ROUTINE

      *****
      .ENABL    LSB
      STKCNT:   .WORD 0
      STKQIN:   .WORD 0
      STKQOUT:  .WORD 0
      STKQSR:   .BLKB 7
      STKQEND=.
      .EVEN

      ::NUMBER OF ITEMS IN QUEUE
      ::INPUT POINTER
      ::OUTPUT POINTER
      ::TTY KEYBOARD QUEUE

      ::TK INITIALIZE ROUTINE
      ::THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
      ::SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
      ::CALL:

```

```

      ::BR IF DONE
      ::INCREASE THE BCD DIGIT 3
      ::ADD BACK THE CONSTANT
      ::CHECK IF BCD DIGIT=0
      ::FALL THROUGH IF 0
      ::STILL DOING LEADING 0'S?
      ::BR IF YES
      ::MSD?
      ::BR IF NO
      ::YES--SET THE SIGN
      ::MAKE THE BCD DIGIT ASCII
      ::MAKE IT A SPACE IF NOT ALREADY A DIGIT
      ::PUT THIS CHARACTER IN THE OUTPUT BUFFER
      ::JUST INCREMENTING
      ::CHECK THE TABLE INDEX
      ::GO DO THE NEXT DIGIT
      ::GO TO EXIT
      ::GET THE LSD
      ::GO CHANGE TO ASCII
      ::WAS THE LSD THE FIRST NON-ZERO?
      ::BR IF NO
      ::YES--SET THE SIGN FOR TYPING
      ::SET THE TERMINATOR
      ::POP STACK INTO R5
      ::POP STACK INTO R3
      ::POP STACK INTO R2
      ::POP STACK INTO R1
      ::POP STACK INTO R0
      ::NOW TYPE THE NUMBER
      ::ADJUST THE STACK
      ::RETURN TO USER

```

```

      *****
      .ENABL    LSB
      STKCNT:   .WORD 0
      STKQIN:   .WORD 0
      STKQOUT:  .WORD 0
      STKQSR:   .BLKB 7
      STKQEND=.
      .EVEN

      ::NUMBER OF ITEMS IN QUEUE
      ::INPUT POINTER
      ::OUTPUT POINTER
      ::TTY KEYBOARD QUEUE

      ::TK INITIALIZE ROUTINE
      ::THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
      ::SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
      ::CALL:

```

```

005504 005037 005556 005570 005570 005570 005570 005570 005570 005570
005510 005037 005574 005570 005570 005570 005570 005570 005570 005570
005516 005037 005574 005570 005570 005570 005570 005570 005570 005570
005524 005037 005574 005570 005570 005570 005570 005570 005570 005570
005532 005037 005574 005570 005570 005570 005570 005570 005570 005570
005540 005037 005574 005570 005570 005570 005570 005570 005570 005570
005548 005037 005574 005570 005570 005570 005570 005570 005570 005570
005552 005037 005574 005570 005570 005570 005570 005570 005570 005570

JSR RETURN PC,STKINT
STKINT: CLR STKCNT ::CLEAR COUNT OF ITEMS IN QUEUE
MOV #STKQST,STKQIN ::MOVE THE STARTING ADDRESS OF THE
MOV STKQIN,STKQOUT ::QUEUE INTO THE INPUT & OUTPUT POINTERS.
MOV #STKSRV,STKVEC ::INITIALIZE THE KEYBOARD VECTOR
MOV #200,STKVEC+2 ::"BR" LEVEL 4
TST STKB ::CLEAR DONE FLAG
MOV #100,STKS ::ENABLE TTY KEYBOARD INTERRUPT
RTS PC ::RETURN TO CALLER

::TK SERVICE ROUTINE
::THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
::BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
::IT IN THE QUEUE.
::IF THE CHARACTER IS A "CONTROL-C" (10) STKINT IS CALLED AND
::UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS .START:

STKSRV: MOV #STKB,SP ::PICKUP THE CHARACTER
BIC #177,SP ::STRIP THE JUNK
CMP (SP),#2 ::IS IT A CONTROL C?
BNE IS ::BRANCH IF NO
TYPE SCNTLC ::TYPE A CONTROL-C (10)
PC,STKINT ::INIT THE KEYBOARD
SP+ ::CLEAN UP STACK
STRT1 ::CONTROL C RESTART
SP, #7 ::IS IT A CONTROL G?
BNE IS ::BRANCH IF NO
SWREG,SWR ::IS SOFT-SWR SELECTED?
BZ SWRCHANGE ::GO TO SWR CHANGE

28: CMP #1,STKCNT ::IS THE QUEUE FULL?
BNE IS ::BRANCH IF NO
BELL ::RING THE TTY BELL
SP+ ::CLEAN CHARACTER OFF OF STACK
EXIT ::EXIT
33: CMP (SP),#23 ::IS IT A CONTROL-S?
BNE IS ::BRANCH IF NO
STKS ::DISABLE TTY KEYBOARD INTERRUPTS
SP+ ::CLEAN CHAR OFF STACK
TSTB STKS ::WAIT FOR A CHAR
BPL 31S ::LOOP UNTIL ITS THERE
MOV #STKB,SP ::GET THE CHARACTER
BIC #177,SP ::MAKE IT 7-BIT ASCII
CMP (SP),#21 ::IS IT A CONTROL-Q?
BNE IS ::BRANCH IF NO
MOV #100,STKS ::REENABLE TTY KEYBOARD INTERRUPTS
RTS ::RETURN
323: STKCNT ::COUNT THIS CHARACTER
SP, #140 ::IS IT UPPER CASE?
BPL 4S ::BRANCH IF YES
SP, #175 ::IS IT A SPECIAL CHAR?

```

```

006034 003002      BGT 4$      ::BRANCH IF YES
006036 042716 000040      BIC 40,(SP)    ::MAKE IT UPPER CASE
006042 112677 175522      MOVB ,SP)+,STKQIN ::AND PUT IT IN QUEUE
006046 005237 005570      INC STKQIN    ::UPDATE THE POINTER
006052 023727 005570 005503      CMP STKQIN,#STKQEND ::GO OFF THE END?
006060 001003      BNE 5$      ::BRANCH IF NO
006066 012737 005574 005570      MOV #STKQSR,STKQIN ::RESET THE POINTER
006070 000002      RTI          ::RETURN

```

```

*****
*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
*CALL WHEN OPERATING IN TTY INTERRUPT MODE.

```

```

006072 022737 000176 001140 $OKSWR: CMP #SWREG,SWR    ::IS THE SOFT-SWR SELECTED
006100 001124      BNE 15$      ::EXIT IF NOT
006102 105777 173036      TSTB 2$TAS    ::IS A CHAR WAITING?
006106 100121      SPL 15$      ::IF NOT, EXIT
006110 117746 173032      MOVB 2$TKB,-(SP) ::YES
006114 042716 177600      BIC #10177,(SP) ::MAKE IT 7-BIT ASCII
006120 021627 000007      CMP (SP),#7 ::IS IT A CONTROL-3?
006124 001300      BNE 2$      ::IF NOT, PUT IT IN THE TTY QUEUE
                                ::AND EXIT

```

```

*****
*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.

```

```

006126 123727 001134 000001 5$: CMPB $AUTOB,#1 ::ARE WE RUNNING IN AUTO-MODE?
006134 001674      BEQ 2$      ::BRANCH IF YES
006136 005726      TST (SP)+    ::CLEAR CONTROL-G OFF STACK
006140 004737 005504      JSR PC,STKINT ::FLUSH THE TTY INPUT QUEUE
006144 005077 172774      CLR 2$TKS  ::DISABLE TTY KEYBOARD INTERRUPTS
006150 112737 000001 001135      MOVB #1,$INTAG ::SET INTERRUPT MODE INDICATOR

```

```

006156 104401 007005      TYPE ,SCNTLG ::ECHO THE CONTROL-G (13)
006162 104401 007012      TYPE ,SMSWR  ::TYPE CURRENT CONTENTS
006166 013746 000176      MOV SWREG,-(SP) ::SAVE SWREG FOR TYPEOUT
006172 104402      TYP0C      ::GO TYPE--OCTAL ASCII ALL DIGITS
006174 104401 007023      TYPE ,SMNEW  ::PROMPT FOR NEW SWR
006200 005046      CLR -(SP)    ::CLEAR COUNTER
006202 005046      CLR -(SP)    ::THE NEW SWR
006204 105777 172734      TSTB 2$TKS  ::CHAR THERE?
006210 100375      BPL 7$      ::IF NOT TRY AGAIN

```

```

006212 117746 172730      MOVB 2$TKB,-(SP) ::PICK UP CHAR
006216 042716 177600      BIC #10177,(SP) ::MAKE IT 7-BIT ASCII

```

```

006222 021627 000003      CMP (SP),#3 ::IS IT A CONTROL-C?
006226 001015      BNE 9$      ::BRANCH IF NOT
006230 104401 006773      TYPE ,SCNTLC ::YES, ECHO CONTROL-C (10)
006234 002706 000006      ADD #6,SP  ::CLEAN UP STACK
006240 123727 001135 000001      CMPB $INTAG,#1 ::REENABLE TTY KEYBOARD INTERRUPTS
006246 001002      BNE 6$      ::BRANCH IF NO

```

Address	Hex	Dec	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	Op41
---------	-----	-----	-------	----	-----	-----	-----	-----	-----	-----	-----	-----	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	------

```

1608 006456 000002 RTI ;;POP NEW PC AND PS
1609 006460 64$: TST $TKCNT ;;WAIT ON A CHARACTER
1610 006460 005737 005566 1$: BEQ 1$
1611 006464 001775 DEC $TKCNT ;;DECREMENT THE COUNTER
1612 006466 005337 005566 MOV $TKQOUT,4(SP) ;;GET ONE CHARACTER
1613 006472 117766 177074 000004 INC $TKQOUT ;;UPDATE THE POINTER
1614 006500 005237 005572 005603 CMP $TKQOUT,$TKQEND ;;DID IT GO OFF OF THE END?
1615 006504 023727 005572 005603 BNE 2$ ;;BRANCH IF NO
1616 006512 001003 MOV $TKQSR, $TKQOUT ;;RESET THE POINTER
1617 006514 012757 005574 005572 2$: RTI ;;RETURN
1618 006522 000002
1619 *****
1620 *THIS ROUTINE WILL INPUT A STRING FROM THE TTY
1621 *CALL:
1622 * ROLIN ;;INPUT A STRING FROM THE TTY
1623 * RETURN HERE ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
1624 * ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
1625
1626 006524 010346 $ROLIN: MOV R3,-(SP) ;;SAVE R3
1627 006526 005046 CLR -(SP) ;;CLEAR THE RUBOUT KEY
1628 006530 012703 006760 1$: MOV $TTYIN,R3 ;;GET ADDRESS
1629 006534 022703 006767 2$: CMP $TTYIN+7,R3 ;;BUFFER FULL?
1630 006540 101456 BLOS 4$ ;;BR IF YES
1631 006542 104410 RDCHR ;;GO READ ONE CHARACTER FROM THE TTY
1632 006544 112613 MOV $SP+,(R3) ;;GET CHARACTER
1633 006546 122713 000177 10$: CMPB #177,(R3) ;;IS IT A RUBOUT
1634 006552 001022 BNE 5$ ;;BR IF NO
1635 006554 005716 TST (SP) ;;IS THIS THE FIRST RUBOUT?
1636 006556 001007 BNE 6$ ;;BR IF NO
1637 006560 112737 000134 006756 MOVB #' \,9$ ;;TYPE A BACK SLASH
1638 006566 104401 006756 TYPE ,9$
1639 006572 012716 177777 MOV #-1,(SP) ;;SET THE RUBOUT KEY
1640 006576 005303 6$: DEC R3 ;;BACKUP BY ONE
1641 006600 020327 006760 CMP R3,$TTYIN ;;STACK EMPTY?
1642 006604 103434 BLOS 4$ ;;BR IF YES
1643 006606 111337 006756 MOVB (R3),9$ ;;SETUP TO TYPEOUT THE DELETED CHAR.
1644 006612 104401 006756 TYPE ,9$
1645 006616 000746 BR 2$ ;;GO TYPE
1646 006620 005716 5$: TST (SP) ;;GO READ ANOTHER CHAR.
1647 006622 001406 BEQ 7$ ;;RUBOUT KEY SET?
1648 006624 112737 000134 006756 MOVB #' \,9$ ;;TYPE A BACK SLASH
1649 006632 104401 006756 TYPE ,9$
1650 006636 005016 CLR (SP) ;;CLEAR THE RUBOUT KEY
1651 006640 122713 000025 7$: CMPB #25,(R3) ;;IS CHARACTER A CTRL "J"?
1652 006644 001003 BNE 8$ ;;BR IF NO
1653 006646 104401 007000 TYPE ,SCNTLU ;;TYPE A CONTROL "U"
1654 006652 000726 BR 1$ ;;GO START OVER
1655 006654 122713 000022 9$: CMPB #22,(R3) ;;IS CHARACTER A "PR"?
1656 006660 001011 BNE 3$ ;;BRANCH IF NO
1657 006662 105013 CLRB (R3) ;;CLEAR THE CHARACTER
1658 006664 104401 001161 TYPE ,SCRLF ;;TYPE A "CR" & "LF"
1659 006670 104401 006760 TYPE ,TTYIN ;;TYPE THE INPUT STRING
1660 006674 000717 BR 2$ ;;GO PICKUP ANOTHER CHARACTER
1661 006676 104401 001160 4$: TYPE ,SQUES ;;TYPE A " "

```

```

1662 006702 000712          BR      1$          ;;CLEAR THE BUFFER AND LOOP
1663 006704 111337 006756 3$:      MOV      (R3),9$      ;;ECHO THE CHARACTER
1664 006710 104401 006756          TYPE      9$
1665 006714 122723 000015          CMPB     #15,(R3)+      ;;CHECK FOR RETURN
1666 006720 001305          BNE      2$          ;;LOOP IF NOT RETURN
1667 006722 105063 177777          CLRB     -1(R3)      ;;CLEAR RETURN (THE 15)
1668 006726 104401 001162          TYPE      $LF      ;;TYPE A LINE FEED
1669 006732 005726          TST      (SP)+      ;;CLEAN RUBOUT KEY FROM THE STACK
1670 006734 012603          MOV      (SP)+,R3      ;;RESTORE R3
1671 006736 011646          MOV      (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
1672 006740 016666 000004 000002  MOV      4(SP),2(SP)      ;;FIRST ASCII CHARACTER ON IT
1673 006746 012766 006760 000004  MOV      #TTYIN,4(SP)
1674 006754 000002          RTI
1675 006756          000          9$:      .BYTE      0          ;;STORAGE FOR ASCII CHAR. TO TYPE
1676 006757          000          .BYTE      0          ;;TERMINATOR
1677 006760 000007          $TTYIN: .BLKB      7          ;;RESERVE 7 BYTES FOR TTY INPUT
1678 006767          207 177777          000 $BELL: .ASCIZ <207><377><377> ;;CODE FOR BELL
1679 006773          136 006503 000012 $CNTLC: .ASCIZ /t<15><12> ;;CONTROL "C"
1680 007000 052536 005015          000 $CNTLU: .ASCIZ /u<15><12> ;;CONTROL "U"
1681 007005          136 006507 000012 $CNTLG: .ASCIZ /g<15><12> ;;CONTROL "G"
1682 007012 005015 053523 020122 $MSWR: .ASCIZ \15><12>/SWR =
1683 007020 020075          000
1684 007023          040 047040 -05350$ $MNEW: .ASCIZ / NEW = /
1685 007030 036440 000040
1686          .SBTTL  RANDOM NUMBER GENERATOR ROUTINE
1687
1688          ;;*****
1689          ;;THIS ROUTINE IS A DOUBLE PRECISION PSEUDO RANDOM NUMBER GENERATOR
1690          ;;WITH A RANGE OF 0 TO 2(+33)-1.
1691          ;;CALL:
1692          ;;      JSR      PC,$RAND          ;;CALL THE ROUTINE
1693          ;;      RETURN          ;;RETURN HERE THE RANDOM
1694          ;;      ;;NUMBER WILL BE IN
1695          ;;      ;;$HINUM,$LONUM
1696
1697          $RAND:
1698          MOV      R0,-(SP)          ;;PUSH R0 ON STACK
1699          MOV      R1,-(SP)          ;;PUSH R1 ON STACK
1700          MOV      R2,-(SP)          ;;PUSH R2 ON STACK
1701 007034 010046          MOV      $LONUM,R0      ;;SET R0 WITH LOW
1702 007036 010146          MOV      $HINUM,R1      ;;SET R1 WITH HIGH
1703 007040 010246          MOV      #-7,R2      ;;SET SHIFT COUNT
1704 007042 013700 007134          ASL      R0          ;;SHIFT R0 LEFT AND
1705 007046 013701 007132          ROL      R1          ;;ROTATE CARRY INTO R1 AND
1706 007052 012702 177771          INC      R2          ;;CHECK FOR DONE
1707 007056 006300          BNE      1$          ;;CONTINUE SHIFT LOOP
1708 007060 006101          ADD      $LONUM,R0      ;;ADD NUMBER TO MAKE X 129
1709 007062 005202          ADC      R1          ;;PROPOGATE CARRY
1710 007064 001374          ADD      $HINUM,R1      ;;ADD NUMBER TO MAKE X 129
1711 007066 063700 007134          ADD      #1057,R0      ;;ADD LOW CONSTANT
1712 007072 005501          ADC      R1          ;;PROPOGATE CARRY
1713 007074 063700 007132          ADD      #47401,R1      ;;ADD HIGH CONSTANT
1714 007100 062700 001057          MOV      R0,$LONUM      ;;SAVE R0
1715 007104 005501          MOV      R1,$HINUM      ;;SAVE R1
1716 007106 062701 047401
1717 007112 010037 007134
1718 007116 010137 007132

```

```

1716 007122 012602
1717 007124 012601
1718 007126 012600
1719 007130 000307
1720 007132 176543
1721 007134 123456

```

```

MOV (SP)+,R2  ; POP STACK INTO R2
MOV (SP)+,R1  ; POP STACK INTO R1
MOV (SP)+,R0  ; POP STACK INTO R0
RTS PC        ; RETURN
$HINUM: .WORD 176543
$LONUM: .WORD 123456
.SBTTL INTEGER DIVIDE ROUTINE

```

```

*****
*THIS ROUTINE WILL DIVIDE A 32-BIT TWO'S COMPLEMENT INTEGER
*DIVIDEND BY A 16-BIT TWO'S COMPLEMENT INTEGER DIVISOR GIVING
*A 16-BIT TWO'S COMPLEMENT INTEGER QUOTIENT AND A 16-BIT REMAINDER.
*DIVISION WILL BE PERFORMED SO THAT THE REMAINDER IS OF THE
*SAVE SIGN AS THE DIVIDEND.
*CALL:

```

```

* MOV LOW DIVIDEND, -(SP) ; THE HIGH DIVIDEND MUST BE < 1/2
* MOV HIGH DIVIDEND, -(SP); AS LARGE AS THE DIVISOR
* MOV DIVISOR, -(SP)
* JSR PC, $DIV
* RETJRN ; QUOTIENT & REMAINDER ARE ON THE STACK
* "V"=0 IMPLIES NO ERROR
* "V"=1 IMPLIES ERROR OCCURRED
* "C"=0 DIVIDE OVERFLOW OCCURRED
* "C"=1 ATTEMPTED TO DIVIDE BY ZERO

```

	STACK	NO ERROR	OVERFLOW	DIVIDE BY ZERO
	TOP	REMAINDER	ALL ZEROS	ALL ONES
	+2	QUOTIENT	ALL ZEROS	ALL ONES

```

1747 007136
1748 007136 104400
1749 007140 042716 000017
1750 007144 010046
1751 007146 010146
1752 007150 010246
1753 007152 010346
1754 007154 005046
1755 007156 012746 000021
1756 007162 016601 000024
1757 007166 016600 000022
1758 007172 100005
1759 007174 105366 000003
1760 007200 005400
1761 007202 005401
1762 007204 005600
1763 007206 016602 000020
1764 007212 002407
1765 007214 003011
1766 007216 052766 000003 000014
1767 007224 012700 177777
1768 007230 000424
1769 007232 005266 000002

```

```

$DIV:
TRAP
BIC #17, (SP)
MOV R0, -(SP)
MOV R1, -(SP)
MOV R2, -(SP)
MOV R3, -(SP)
CLR -(SP)
MOV #17, -(SP)
MOV 24(SP), R1
MOV 22(SP), R0
BPL 1$
DECB 3(SP)
NEG R0
NEG R1
SBC R0
1$: MOV 20(SP), R2
BLT 2$
BGT 3$
BIS #3, 14(SP)
MOV #-1, R0
BR 7$
2$: INC 2(SP)
; PUSH OLD PSW AND PC ON STACK
; STRIP AWAY CONDITION CODES
; PUSH R0 ON STACK
; PUSH R1 ON STACK
; PUSH R2 ON STACK
; PUSH R3 ON STACK
; SAVE A PLACE FOR SIGNS
; SETUP THE ITERATION COUNTER
; PICKUP THE DIVIDEND
; CHECK THE SIGN
; KEEP TRACK OF THE SIGN
; AND NEGATE THE ORIGINAL
; NUMBER
; PICKUP THE DIVISOR
; CHECK THE SIGN
; DIVISOR OF 0 IS A NO-NO
; SET "V" & "C"
; SET REMAINDER TO ALL ONES
; EXIT
; KEEP TRACK OF DIVISOR'S SIGN

```

```

1770 007236 000401          BR      4$
1771 007240 005402          3$: NEG      R2          ;;NEGATE THE ORIGINAL NUMBER
1772 007242 000241          4$: CLC          ;;CLEAR "C"
1773 007244 000405          BR      6$          ;;START FORMING QUOTIENT
1774 007246 006100          5$: ROL      R0          ;;POSITION MSB'S
1775 007250 010003          MOV      R0,R3          ;;COPY
1776 007252 060203          ADD      R2,R3          ;;COMPARE DIVIDEND & DIVISOR
1777 007254 103001          BCC      6$          ;;SR IF DIVIDEND > DIVISOR
1778 007256 010300          MOV      R3,R0          ;;REMAINDER AFTER THIS LOOP
1779 007260 006101          6$: ROL      R1          ;;QUOTIENT BIT ENTERS HERE
1780 007262 005316          DEC      (SP)          ;;DONE?
1781 007264 001370          BNE      5$          ;;BR IF NO
1782 007266 005701          TST      R1          ;;OVERFLOW?
1783 007270 100005          BPL      8$          ;;BR IF NO
1784 007272 052766 000002 000014 BIS      R2,14(SP)          ;;SET "V" IN RETURN STATUS WORD
1785 007300 005000          CLR      R0          ;;SET REMAINDER TO ALL ZEROS
1786 007302 010001          7$: MOV      R0,R1          ;;COPY REMAINDER INTO QUOTIENT
1787 007304 005726          8$: TST      (SP)+          ;;CLEAR COUNTER FROM STACK
1788 007306 005716          TST      (SP)          ;;REMAINDER SIGN CORRECTION NEEDED?
1789 007310 002004          BGE      9$          ;;BR IF NO
1790 007312 005400          NEG      R0          ;;NEGATE REMAINDER
1791 007314 105066 000001          CLRB     1(SP)          ;;CLEAR SIGN
1792 007320 005316          DEC      (SP)          ;;BUT DON'T FORGET QUOTIENT
1793 007322 005726          9$: TST      (SP)+          ;;QUOTIENT SIGN CORRECTION NEEDED?
1794 007324 001401          BEQ      10$          ;;BR IF NO
1795 007326 005401          NEG      R1          ;;NEGATE QUOTIENT
1796 007330 010166 000020 10$: MOV      R1,20(SP)          ;;RETURN QUOTIENT AND
1797 007334 010066 000016          MOV      R0,16(SP)          ;;REMAINDER TO USER
1798 007340 012603          MOV      (SP)+,R3          ;;POP STACK INTO R3
1799 007342 012602          MOV      (SP)+,R2          ;;POP STACK INTO R2
1800 007344 012601          MOV      (SP)+,R1          ;;POP STACK INTO R1
1801 007346 012600          MOV      (SP)+,R0          ;;POP STACK INTO R0
1802 007350 012656 000002          MOV      (SP)+,2(SP)          ;;SETUP TO RETURN CONDITION CODES
1803 007354 000002          RTI          ;;RETURN
1804          .SETTL  SAVE AND RESTORE RO-R5 ROUTINES
1805
1806          ;*****
1807          ;SAVE RO-R5
1808          ;CALL:
1809          ;      SAVREG
1810          ;UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
1811          ;
1812          ;TOP---(+16)
1813          ;+2---(+18)
1814          ;+4---R5
1815          ;+6---R4
1816          ;+8---R3
1817          ;+10---R2
1818          ;+12---R1
1819          ;+14---R0
1820
1821          $SAVREG:
1822          MOV      R0,-(SP)          ;;PUSH R0 ON STACK
1823          MOV      R1,-(SP)          ;;PUSH R1 ON STACK

```

K04

MD-11-CRJC-A, RPO4 5 & HEAD ALIGNMENT PROGRAM MACY11 27.655) 27-APR-76 16:34 PAGE 37
 CRJC.009 SAVE AND RESTORE R3-R5 ROUTINES

351 004E

```

1824 007362 010246      MOV      R2,-(SP)      ;;PUSH R2 ON STACK
1825 007364 010346      MOV      R3,-(SP)      ;;PUSH R3 ON STACK
1826 007366 010446      MOV      R4,-(SP)      ;;PUSH R4 ON STACK
1827 007370 010546      MOV      R5,-(SP)      ;;PUSH R5 ON STACK
1828 007372 016646 000022  MOV      22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
1829 007376 016646 000022  MOV      22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
1830 007402 016646 000022  MOV      22(SP),-(SP)    ;;SAVE PS OF CALL
1831 007406 016646 000022  MOV      22(SP),-(SP)    ;;SAVE PC OF CALL
1832 007412 000002      RTI
1833
1834      ;;*RESTORE R0-R5
1835      ;;*CALL:
1836      ;;*      RESREG
1837      $RESREG:
1838 007414 012666 000022  MOV      (SP)+,22(SP)    ;;RESTORE PC OF CALL
1839 007420 012666 000022  MOV      (SP)+,22(SP)    ;;RESTORE PS OF CALL
1840 007424 012666 000022  MOV      (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
1841 007430 012666 000022  MOV      (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
1842 007434 012605      MOV      (SP)+,R5      ;;POP STACK INTO R5
1843 007436 012604      MOV      (SP)+,R4      ;;POP STACK INTO R4
1844 007440 012603      MOV      (SP)+,R3      ;;POP STACK INTO R3
1845 007442 012602      MOV      (SP)+,R2      ;;POP STACK INTO R2
1846 007444 012601      MOV      (SP)+,R1      ;;POP STACK INTO R1
1847 007446 012600      MOV      (SP)+,R0      ;;POP STACK INTO R0
1848 007450 000002      RTI
1849
1850      .SBTTL  TRAP DECODER
1851
1852      ;;*****
1853      ;;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
1854      ;;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
1855      ;;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
1856      ;;*GO TO THAT ROUTINE.
1857 007452 010046 000002  $TRAP:  MOV      R0,-(SP)      ;;SAVE R0
1858 007454 016600      MOV      2(SP),R0      ;;GET TRAP ADDRESS
1859 007460 005740      TST      -(R0)          ;;BACKUP BY 2
1860 007462 111000      MOVB     (R0),R0          ;;GET RIGHT BYTE OF TRAP
1861 007464 006300      ASL      R0              ;;POSITION FOR INDEXING
1862 007466 016000 007506  MOV      $TRAPD(R0),R0    ;;INDEX TO TABLE
1863 007472 000200      RTS      R0              ;;GO TO ROUTINE
1864
1865
1866      ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
1867 007474 011646 000004 000002  $TRAP2: MOV      (SP),-(SP)    ;;MOVE THE PC DOWN
1868 007476 016666      MOV      4(SP),2(SP)    ;;MOVE THE PSW DOWN
1869 007504 000002      RTI              ;;RESTORE THE PSW
1870
1871      .SBTTL  TRAP TABLE
1872
1873      ;;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
1874      ;;*BY THE "TRAP" INSTRUCTION.
1875
1876      :          ROUTINE
1877

```

DZRJC.009 TRAP TABLE

```

1878
1879 007506 007474
1880 007510 004674
1881 007512 005140
1882 007514 005114
1883 007516 005154
1884 007520 005342
1885
1886 007522 006162
1887
1888 007524 006072
1889 007526 006434
1890 007530 006524
1891 007532 007356
1892 007534 007414
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910 007536 000000 000000 000000
1911 007544 000000
1912
1913
1914
1915
1916
1917
1918 007546 000
1919 007547 000
1920 007550 000
1921 007551 000
1922 007552 000
1923 007553 000
1924 007554 000
1925 007555 000
1926
1927
1928
1929
1930
1931

```

;STRPAD: .WORD STRAP2
 ;TYPE ::CALL=TYPE TRAP+1(104401) TTY TYPEOUT ROUTINE
 ;STYPOC ::CALL=TYPOC TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
 ;STYPOS ::CALL=TYPOS TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS,
 ;STYPON ::CALL=TYPON TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
 ;STYPS ::CALL=TYPS TRAP+5(104405) TYPE DECIMAL NUMBER (WITH SIGN)

 ;GTSWR ::CALL=GTSWR TRAP+6(104406) GET SOFT-SWR SETTING

 ;CKSWR ::CALL=CKSWR TRAP+7(104407) TEST FOR CHANGE IN SOFT-SWR
 ;RDCHR ::CALL=RDCHR TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
 ;RDLIN ::CALL=RDLIN TRAP+11(104411) TTY TYPEIN STRING ROUTINE
 ;SAVREG ::CALL=SAVREG TRAP+12(104412) SAVE R0-R5 ROUTINE
 ;RESREG ::CALL=RESREG TRAP+13(104413) RESTORE R0-R5 ROUTINE

.SBTTL SINGLE/DUAL PORT RH11/RPC4/5/6 DRIVER (REV 1.0)
 ;COPYRIGHT (C) 1976
 ;DIGITAL EQUIPMENT CORP.
 ;MAYNARD, MA 01754
 ;AUTHOR(S): JIM LACEY/CHUCK HESS
 ;*****
 ;STORAGE FOR RPDS1, RPER1, RPER2, AND RPER3 ON AN ERROR "2"
 ;RPERRS = RPDS1
 ;RPERRS+2 = RPER1
 ;RPERRS+4 = RPER2
 ;RPERRS+6 = RPER3
 RPERRS: .WORD 0,0,0,0

;TABLE OF DRIVE ACTIVE INDICATORS (DRVACT=8 BYTES)
 ;DRVACT=0 IF DRIVE IS IDLE
 ;DRVACT>0 IF DRIVE IS ACTIVE WITH A COMMAND
 ;DRVACT<0 IF DRIVE IS ACTIVE WITH AN ERROR RECOVERY OPERATION
 DRVACT: .BYTE 0 ;DRIVE 0
 .BYTE 0 ;DRIVE 1
 .BYTE 0 ;DRIVE 2
 .BYTE 0 ;DRIVE 3
 .BYTE 0 ;DRIVE 4
 .BYTE 0 ;DRIVE 5
 .BYTE 0 ;DRIVE 6
 .BYTE 0 ;DRIVE 7

;TABLE OF DRIVE STATUS INDICATORS (DRVSTA=8 BYTES)
 ;DRVSTA=0 IF DRIVE IS OFFLINE OR NONEXISTENT
 ;DRVSTA>0 IF DRIVE IS ONLINE
 ;DRVSTA<0 IF DRIVE IS UNSAFE

1932 007556 000
 1933 007557 000
 1934 007560 000
 1935 007561 000
 1936 007562 000
 1937 007563 000
 1938 007564 000
 1939 007565 000

DRVSTA: .BYTE 0 ;DRIVE 0
 .BYTE 0 ;DRIVE 1
 .BYTE 0 ;DRIVE 2
 .BYTE 0 ;DRIVE 3
 .BYTE 0 ;DRIVE 4
 .BYTE 0 ;DRIVE 5
 .BYTE 0 ;DRIVE 6
 .BYTE 0 ;DRIVE 7

;TABLE OF DRIVE TYPES (DRVTYP=8 BYTES)
 ;DRVTYP=0 IF DRIVE IS NONEXISTENT (DRVSTA=0, ALSO)
 ;DRVTYP=1 IF DRIVE IS RPO4
 ;DRVTYP=2 IF DRIVE IS RPO5
 ;DRVTYP=4 IF DRIVE IS RPO6
 ;DRVTYP=-1 IF NOT RPO4/5/6

1948 007566 000
 1949 007567 000
 1950 007570 000
 1951 007571 000
 1952 007572 000
 1953 007573 000
 1954 007574 000
 1955 007575 000

DRVTYP: .BYTE 0 ;DRIVE 0
 .BYTE 0 ;DRIVE 1
 .BYTE 0 ;DRIVE 2
 .BYTE 0 ;DRIVE 3
 .BYTE 0 ;DRIVE 4
 .BYTE 0 ;DRIVE 5
 .BYTE 0 ;DRIVE 6
 .BYTE 0 ;DRIVE 7

;TABLE OF DUAL PORT INITIALIZATION INDICATORS
 ;DPINT=0 IF INITIALIZATION IS NOT ACTIVE ON THE DRIVE
 ;DPINT<0 IF INITIALIZATION IS IN PROGRESS

1961 007576 000
 1962 007577 000
 1963 007600 000
 1964 007601 000
 1965 007602 000
 1966 007603 000
 1967 007604 000
 1968 007605 000

DPINT: .BYTE 0 ;DRIVE 0
 .BYTE 0 ;DRIVE 1
 .BYTE 0 ;DRIVE 2
 .BYTE 0 ;DRIVE 3
 .BYTE 0 ;DRIVE 4
 .BYTE 0 ;DRIVE 5
 .BYTE 0 ;DRIVE 6
 .BYTE 0 ;DRIVE 7

;TABLE OF PENDING DUAL PORT REQUESTS
 ;DPRQS=0 IF THAT A DUAL PORT REQUEST IS NOT PENDING FOR THAT DRIVE
 ;DPRQS<0 IF THAT A DUAL PORT REQUEST IS PENDING FOR THAT DRIVE

1974 007606 000
 1975 007607 000
 1976 007610 000
 1977 007611 000
 1978 007612 000
 1979 007613 000
 1980 007614 000
 1981 007615 000

DPRQS: .BYTE 0 ;DRIVE 0
 .BYTE 0 ;DRIVE 1
 .BYTE 0 ;DRIVE 2
 .BYTE 0 ;DRIVE 3
 .BYTE 0 ;DRIVE 4
 .BYTE 0 ;DRIVE 5
 .BYTE 0 ;DRIVE 6
 .BYTE 0 ;DRIVE 7

;TRANSFER WAIT FLAG (TRNSWT=1 WORD)
 ;THIS IS A ONE WORD QUEUE. IT WILL CONTAIN THE ADDRESS OF
 ;"DPB" OF THE I/O OPERATION.

1982
 1983
 1984
 1985

```

1996      007616  000000      TRNSWT: .WORD    0
1997
1998
1999      :SEARCH WAIT KEYS (SRCHWT=1 WORD)
2000      :THIS IS A ONE WORD QUEUE THAT WILL CONTAIN A KEY FOR EACH OF
2001      :THE DRIVES THAT ARE PERFORMING A SEARCH COMMAND FOR THE I/O
2002      :REQUEST THAT IS AT THE TOP OF THEIR REQUEST QUEUE.
2003      :EACH DRIVE IS ASSIGNED ONE BIT, STARTING AT BIT00 FOR DRIVE 0.
2004
2005      007620  000000      SRCHWT: .WORD    0
2006
2007      :RPO4/5/6 DRIVER ACTIVE FLAG (ACTDRV=1 BYTE)
2008      :ACTDRV=0 IF DRIVER IS INACTIVE
2009      :ACTDRV>0 IF DRIVER IS ACTIVE
2010
2011      007622      000      ACTDRV: .BYTE    0
2012
2013      :SOFTWARE TIMER ROUTINE ACTIVE FLAG (ACTSTR=1 BYTE)
2014      :ACTSTR=0 IF SOFTWARE TIMER ROUTINE IS INACTIVE
2015      :ACTSTR>0 IF SOFTWARE TIMER ROUTINE IS ACTIVE
2016
2017      007623      000      ACTSTR: .BYTE    0
2018
2019      :UNLOAD FLAG (ULDFLG=8 BYTES)
2020      :ULDFLG=0 IF NO UNLOAD COMMAND
2021      :ULDFLG>0 IF UNLOAD COMMAND IN PROGRESS
2022      :ULDFLG<0 IF UNLOAD COMMAND IN WAIT QUEUE
2023
2024      007624      000      ULDFLG: .BYTE    0      :DRIVE 0
2025      007625      000      .BYTE    0      :DRIVE 1
2026      007626      000      .BYTE    0      :DRIVE 2
2027      007627      000      .BYTE    0      :DRIVE 3
2028      007630      000      .BYTE    0      :DRIVE 4
2029      007631      000      .BYTE    0      :DRIVE 5
2030      007632      000      .BYTE    0      :DRIVE 6
2031      007633      000      .BYTE    0      :DRIVE 7
2032
2033      :LOOK AHEAD COUNT (LACNT=8 BYTES)
2034      :LACNT WILL INDICATE THE NUMBER OF LOOK AHEADS PERFORMED
2035
2036      007634      000      LACNT: .BYTE    0      :DRIVE 0
2037      007635      000      .BYTE    0      :DRIVE 1
2038      007636      000      .BYTE    0      :DRIVE 2
2039      007637      000      .BYTE    0      :DRIVE 3
2040      007640      000      .BYTE    0      :DRIVE 4
2041      007641      000      .BYTE    0      :DRIVE 5
2042      007642      000      .BYTE    0      :DRIVE 6
2043      007643      000      .BYTE    0      :DRIVE 7
2044
2045      :SAVE REGISTERS FLAG (SAVEFG =1 WORD)
2046      :SAVEFG <0 IF SAVE THE RH11/RPO4/5/6 REGISTERS WHEN THE
2047      :OPERATION IS COMPLETED AS PER (DPB+14).
2048      :SAVEFG=0 IF SAVE THE RH11/RPO4/5/6 REGISTERS. AS PER
2049      : (DPB+14). AFTER AN ERROR.

```

SAVEFG: .WORD 0

```
:SEEK FLAG, SEE FLAG=1 WORD:
:SEEK=0 IF WHEN THE DISK ADDRESS ISN'T IN THE WINDOW
:FOR A DATA TRANSFER START A SEARCH COMMAND
:SEEK=0 IF DATA TRANSFER WILL DO IMPLIED SEEKS.
:DISCARD THE WINDOW
```

SEE.FG: .WORD 0

```

;TIMEOUT TABLE (TIMER=8 WORDS)
;THIS TABLE CONTAINS THE TIME ALLOWED FOR AN OPERATION

```

```

TIMER:      .WORD      -1      :DRIVE 0
            .WORD      -1      :DRIVE 1
            .WORD      -1      :DRIVE 2
            .WORD      -1      :DRIVE 3
            .WORD      -1      :DRIVE 4
            .WORD      -1      :DRIVE 5
            .WORD      -1      :DRIVE 6
            .WORD      -1      :DRIVE 7

```

```

:DATA TRANSFER UNDERWAY INDICATOR (DTUW=1 WORD)
:DTUW<0 IF NO DATA TRANSFER UNDERWAY
:DTUW=+N (WHERE N=0 TO 7) IMPLIES DATA TRANSFER UNDERWAY ON DRIVE N

```

DTUW: .WORD -1

```

:ATTENTION BITS TABLE (ATA217=9 BYTES)
:THIS TABLE CONTAINS THE CORRESPONDING BIT TO EACH DRIVES
:ATTENTION BIT

```

[illegible]

```
:RPO4/5/6 TO RH11 "MASSEBUS CONTROL BUS PARITY ERRORS" (MCPD) ALLOWED BEFORE
:CALLING IT FATAL (MCPD=1 WORD)
```

MCPEMX: .WORD 3

```

: STORAGE FOR RPADR (THE FIRST ADDRESS (776700) OF THE RAIL RPJ4-5-6).
: RPVEC (THE VECTOR ADDRESS (254)), AND RPVEC+2 (THE BR LEVEL (5)).

```

PPADR: .WORD 176700
SPVEC: .WORD 254.5*32.

:MAXIMUM NUMBER OF LOOK AHEADS ALLOWED IS 4 (INLACT=1 WORD)

007712 000004

 MXDLTA: .WORD 4
 :MAXIMUM DELTA DELAY IS 8 SECTORS (MXDLTA=1 WORD)

007714 001000

 MNDLTA: .WORD 8*64
 :MINIMUM DELTA DELAY IS 2 SECTORS (MNDLTA=1 WORD)

007716 002000

 MNWDELTA: .WORD 2*64
 :MAXIMUM SEARCH FOR I/O WINDOW IS 5 SECTORS (MNWDELTA=1 WORD)

007720 000005

MNWDELTA: .WORD 5

:DEFINITIONS OF THE RH11/RP04/5/6 ADDRESS INDEXES

 000000
 000001
 000002
 000003
 000004
 000005
 000006
 000007
 000008
 000009
 000010
 000011
 000012
 000013
 000014
 000015
 000016
 000017
 000018
 000019
 000020
 000021
 000022
 000023
 000024
 000025
 000026
 000027
 000028
 000029
 000030
 000031
 000032
 000033
 000034
 000035
 000036
 000037
 000038
 000039
 000040
 000041
 000042
 000043
 000044
 000045
 000046
 000047
 000048
 000049
 000050
 000051
 000052
 000053
 000054
 000055
 000056
 000057
 000058
 000059
 000060
 000061
 000062
 000063
 000064
 000065
 000066
 000067
 000068
 000069
 000070
 000071
 000072
 000073
 000074
 000075
 000076
 000077
 000078
 000079
 000080
 000081
 000082
 000083
 000084
 000085
 000086
 000087
 000088
 000089
 000090
 000091
 000092
 000093
 000094
 000095
 000096
 000097
 000098
 000099
 000100

 RP051=0
 RP052=1
 RP053=2
 RP054=3
 RP055=4
 RP056=5
 RP057=6
 RP058=7
 RP059=8
 RP060=9
 RP061=10
 RP062=11
 RP063=12
 RP064=13
 RP065=14
 RP066=15
 RP067=16
 RP068=17
 RP069=18
 RP070=19
 RP071=20
 RP072=21
 RP073=22
 RP074=23
 RP075=24
 RP076=25
 RP077=26
 RP078=27
 RP079=28
 RP080=29
 RP081=30
 RP082=31
 RP083=32
 RP084=33
 RP085=34
 RP086=35
 RP087=36
 RP088=37
 RP089=38
 RP090=39
 RP091=40
 RP092=41
 RP093=42
 RP094=43
 RP095=44
 RP096=45
 RP097=46
 RP098=47
 RP099=48
 RP100=49

 :CONTROL AND STATUS REGISTER #1 (DRIVE REG. 00)
 :WORD COUNT REGISTER (NOT A DRIVE REG.)
 :UNIBUS ADDRESS REGISTER (NOT A DRIVE REG.)
 :DESIRED SECTOR/TRACK ADDRESS REGISTER (DRIVE REG. 05)
 :CONTROL AND STATUS REGISTER #2 (NOT A DRIVE REG.)
 :DRIVE STATUS REGISTER (DRIVE REG. 01)
 :ERROR REGISTER #1 (DRIVE REG. 02)
 :ATTENTION SUMMARY PSEUDO REGISTER (DRIVE REG. 04)
 :LOOK AHEAD REGISTER (DRIVE REG. 07)
 :DATA BUFFER REGISTER (NOT A DRIVE REG.)
 :MAINTAINABILITY REGISTER (DRIVE REG. 03)
 :DRIVE TYPE REGISTER (DRIVE REG. 06)
 :SERIAL NUMBER REGISTER (DRIVE REG. 10)
 :OFFSET REGISTER (DRIVE REG. 11)
 :DESIRED CYLINDER ADDRESS REGISTER (DRIVE REG. 12)
 :CURRENT CYLINDER ADDRESS REGISTER (DRIVE REG. 13)
 :ERROR REGISTER #2 (DRIVE REG. 14)
 :ERROR REGISTER #3 (DRIVE REG. 15)
 :ECC POSITION REGISTER (DRIVE REG. 16)
 :ECC PATTERN REGISTER (DRIVE REG. 17)

:RH11/RP04/5/6 DRIVER INITIALIZATION CODE

 :THIS ROUTINE WILL DETERMINE WHICH RP04/5/6 DRIVES ARE
 :AVAILABLE FOR TESTING AND SET THE DRVSTA INDICATOR
 :TO THE PROPER STATE FOR EACH DRIVE.
 :NOTE: THIS ROUTINE CALLS DRVINT

:CALL

 JSR PC,RPINIT
 RETURN

:NOTE: THE 'P' OR 'L' CLOCK MUST BE STARTED

RPINIT: SAVREG

 MOV 2*PS, -1*SP
 MOV 5*32, 3*PS
 JSR PC,CLRQUE
 MOV 5*PERRS, R1
 MOV 5*SEEKFG, R2

 :SAVE R0 - R5
 :SAVE THE PRESENT PROCESSOR STATUS
 :CHANGE THE PRIORITY TO 5
 :CLEAR ALL REQUEST QUEUES
 :FIRST ADDRESS TO BE CLEARED
 :LAST ADDRESS TO BE CLEARED

:77776

27-APR-76 16:34 PAGE 43
 27-APR-76 16:34 PAGE 43
 27-APR-76 16:34 PAGE 43

```

15: CLR (R1)+      :CLEAR
    CMP R1,R2      :ARE WE DONE?
    BLOS 15         :BRANCH IF NO
    MOV #DTUW,R2    :LAST ADDRESS
25: MOV #-1,(R1)+   :INITIALIZE
    CMP R1,R2       :DONE?
    BLOS 25         :LOOP IF NO
    CLR DRVSTA      :SET ALL DRIVES TO OFFLINE
    CLR DRVSTA+2
    CLR DRVSTA+4
    CLR DRVSTA+6
    MOV RPVEC,R3    :SETUP THE RH11/RP04/5/6 VECTOR
    MOV #ISR,(R3)+
    MOV RPVEC+2,(R3)
    MOV RPADR,R4    :FIRST ADDRESS OF RH11/RP04
    MOV #BIT05,RP052,R4 :MASSBUS INIT
    JSR R0,DRVINT  :START WITH DRIVE 0
    BR 45           :INIT THE DRIVE
    BR 55           :DVA NOT SET OR PARITY ERROR
    BR 55           :NORMAL RETURN
45: CLR DRVSTA,R1)  :SET DRIVE STATUS TO OFFLINE
55: INC R1          :GO TO NEXT DRIVE
    BIC #107,R1     :MASK OUT UNUSED BITS
    BNE 55          :BR IF MORE DRIVES TO GO
    MOV #7,R1       :START WITH DRIVE 7
    CLR #PS         :CLEAR THE PROCESSOR STATUS
    JSR R0,DRVINT  :WAITING FOR DRIVE TO SWITCH PORTS
    BR 65           :BR NOT WAITING
    SET IE          :SET INTERRUPT
    JSR R0,DRVINT  :DRIVE SWITCHED PORTS
    BR 65           :BR IF NOT
    INC R1          :GO TO THE NEXT DRIVE
    BNE 65          :CHECK NEXT DRIVE
    MOV (SP)+,3#PS  :RESTORE THE PROCESSOR STATUS
    RESREG R0 - R5 :RESTORE R0 - R5
    RTS PC         :BYE-BYE

```

:DRIVE INITIALIZATION ROUTINE

:THIS ROUTINE DETERMINES IF A DRIVE EXISTS AND IF IT IS
 :AN RP04/5/6. IF IT IS, A "READ-IN PRESET" IS ISSUED AND FMT22
 :IS SET TO A "1". THEN MOL, DPR, DRY, AND VV ARE CHECKED
 :INSURE THEY ARE ALL ON A "1". AND DEPENDING ON THEIR STATE,
 :DRVSTA IS SET TO THE PROPER CONDITION.

:CALL

```

MOV #DRVNUM,R1    :DRIVE NUMBER TO R1
MOV RPADR,R4      :UNIBUS ADDRESS OF RH11/RP04/5/6 (RP05)
JSR R0,DRVINT     :CALLED BY A JSR
RETURN1           :ERROR OCCURRED (PARITY)
RETURN2           :NORMAL RETURN

```

```

DRVINT: MOV R5,-(SP) :SAVE R5
        CLRB DRVSTA,R1 :START DRIVE STATUS AS OFFLINE
        CLRB DRVTYP,R1 :CLEAR THE DRIVE TYPE INDICATOR

```

010146	135061	227624	CLRB	ULDFLG(R1)	:CLEAR THE UNLOAD FLAG
010152	010164	200010	MOV	R1,RPCS2(R4)	:SELECT A DRIVE
010156	112764	000111	MOVB	#111,RPCS1(R4)	:DO A DRIVE CLEAR COMMAND 13 SEIZE CFI.E
010164	032764	010000	BIT	#BIT12,RPCS2(R4)	:NONEXISTENT DRIVE
010172	001403		BEG	15	:NO---BRANCH
010174	004727	015316	JSR	PC,SET.IE	:GO SET "IE" WITHOUT A "TRE"
010200	000520		OR	65	:LEAVE THIS ROUTINE
010202	135061	007556	CLRB	DRVSTA(R1)	:SET DRIVE STATUS TO OFFLINE
010206	032764	004000	BIT	#BIT11,RPCS1(R4)	:SEE IF DRIVE AVAILABLE
010214	001514		BEG	75	:BR IF DRIVE NOT AVAILABLE
010216	004037	014636	JSR	RD,RO.RP	:READ THE DRIVE TYPE REG.
010222	000026		RPDT		
010234	010466		95		:ERROR RETURN ADDRESS
010236	012605		MOV	(SP)+,R5	:PUT DRIVE TYPE IN R5
010237	112761	000001	MOVB	#1,DRVTP(R1)	:SET RPO4 INDICATOR
010238	022705	020020	CMP	#20020,R5	:IS IT A SINGLE PORT RPO4?
010242	001431		BEG	25	:BRANCH IF YES
010244	022705	024020	CMP	#24020,R5	:IS IT A DUAL PORT RPO4?
010250	001426		BEG	25	:BR IF YES
010252	112761	000002	MOVB	#2,DRVTP(R1)	:SET RPO5 INDICATOR
010256	022705	020021	CMP	#20021,R5	:SINGLE PORT RPO5 ?
010264	001420		BEG	25	:BR IF YES
010266	022705	024021	CMP	#24021,R5	:DUAL PORT RPO5 ?
010272	001415		BEG	25	:BR IF YES
010274	112761	000004	MOVB	#4,DRVTP(R1)	:SET RPO6 INDICATOR
010302	022705	020022	CMP	#20022,R5	:SINGLE PORT RPO6 ?
010306	001407		BEG	25	:BR IF YES
010310	022705	024022	CMP	#24022,R5	:DUAL PORT RPO6 ?
010314	001404		BEG	25	:BR IF YES
010316	112761	177777	MOVB	#-1,DRVTP(R1)	:SET INDICATOR TO "OTHER"
010324	000446		BR	65	:EXIT
010326	012746	000121	MOV	#121,-(SP)	:DO A "READ-IN PRESET"
010332	004037	015012	JSR	RD,WRT.RP	
010336	000000		RPCS1		
010340	010466		95		
010342	012746	010000	MOV	#BIT12,-(SP)	:SET FMT22=1
010346	004037	015012	JSR	RD,WRT.RP	
010352	000032		RPOF		
010354	010466		95		
010356	004037	014636	JSR	RD,RO.RP	:READ RPCS1
010362	000012		RPCS1		
010364	010466		95		
010366	012605		MOV	(SP)+,R5	:AND SAVE IT IN R5
010370	100015		BPL	45	:BRANCH IF ATA=0
010372	116164	007672	MOVB	ATABIT(R1),RPAS(R4)	:CLEAR ATTENTION BIT
010400	004037	014636	JSR	RD,RO.RP	:FIND OUT WHY ATA=1
010404	000014		RPER1		
010406	010466		95		
010410	006126		ROL	(SP)+	:IS IT UNSAFE?
010412	100004		BPL	45	:BR IF NOT
010414	112761	177777	MOVB	#-1,DRVSTA(R1)	:SET UNSAFE INDICATOR
010422	000407		BR	65	:EXIT
010424	005105		COM	R5	:CHECK MOL, DPR, DRY, AND VV
010426	042705	167077	BIC	#10 BIT12!BIT08!BIT07!BIT06,R5	

```

2275 010432 001003 BNE 65 :BRANCH IF MOL, DPR, DRY, OR VV IS CLEAR
2276 010434 112761 000001 007556 MOV 65 :SET DRIVE STATUS TO ONLINE
2277 010442 005720 65: TST 65 :STEP OVER THE ERROR RETURN
2278 010444 000410 BR 95 :EXIT
2279 010446 006301 75: ASL R1 :CHANGE INDEX TO ADDRESS WORDS
2280 010450 012761 003720 007650 MOV 65 :START 2 SEC TIMER
2281 010456 006201 ASR R1 :RESTORE R1
2282 010460 112761 177777 007576 MOV 65 :SET PORT INITIALIZE INDICATOR
2283 010466 012605 65: MOV 65 :RESTORE R5
2284 010470 000200 RTS 65 :EXIT

:REQUEST PRE-PROCESSOR-HANDLES SUBSYSTEM REQUEST
:CALL
:
:JSR 65:2#RPO4 :CALL THE RPO4 5/6 DRIVER
:PTRADR :ADDRESS OF POINTER OF DRIVES PARAMETER BLOCK
:RETURN1 :RETURN HERE IF QUEUE IS FULL
:RETURN2 :RETURN HERE IF REQUEST IS IN QUEUE OR THERE
: :IS AN ERROR CONDITION

2285 010472 013746 177775 RPO4: MOV 65:PS - SP, :SAVE THE CALLING STATUS
2286 010476 013737 007710 177776 MOV 65:RVEC+2, 3#PS :DON'T ALLOW ANY RPO4 5/6 INTERRUPTS
2287 010504 112737 000001 007522 MOV 65:1, ACTDRV :SET "ACTIVE DRIVER" FLAG
2288 010512 104412 SAVREG :SAVE R0 - R5
2289 010514 011002 MOV 65:(R0), R2 :PICKUP THE DRIVE PARAMETER BLOCK POINTER
2290 010516 005062 000016 CLR 65:(R2) :CLEAR THE STATUS/ERROR INDICATOR
2291 010522 111201 MOV 65:(R2), R1 :PICKUP THE DRIVE NUMBER
2292 010524 013704 007704 MOV 65:RPAR, R4 :UNIBUS ADDRESS OF RPCS1
2293 010530 105761 007556 TSTB 65:DRVSTA(R1) :CHECK DRIVES STATUS
2294 010534 003014 BGT 65:15 :BRANCH IF ONLINE
2295 010536 105761 007624 TSTB 65:ULDFLG(R1) :UNLOAD COMMAND IN QUEUE?
2296 010542 001036 BNE 65:35 :BRANCH IF YES
2297 010544 105761 007576 TSTB 65:DPINT(R1) :TRYING TO INIT THE DRIVE
2298 010550 001042 BNE 65:55 :BR IF YES
2299 010552 004037 010134 JSR 65:R0, DRVINT :GO INIT. THE DRIVE
2300 010556 000434 BR 65:45 :ERROR RETURN
2301 010560 105761 007556 TSTB 65:DRVSTA(R1) :IS DRIVE STATUS ONLINE?
2302 010564 003445 BLE 65:65 :BR IF NOT
2303 010566 105761 007606 13: TSTB 65:DPROG(R1) :OUTSTANDING PORT REQUEST FOR THE DRIVE ?
2304 010572 001031 BNE 65:55 :BR IF YES
2305 010574 010164 000010 MOV 65:R1, RPCS2(R4) :SELECT THE DRIVE
2306 010600 004037 015760 JSR 65:R0, DRVQUE :PUT THIS REQUEST IN QUEUE
2307 010604 000460 BR 65:95 :QUEUE IS FULL
2308 010606 122762 000103 000002 CMPB 65:#103, 2(R2) :IS THIS REQ. FOR AN UNLOAD?
2309 010614 001003 BNE 65:25 :BR IF NO
2310 010616 112761 177777 007624 MOV 65:#-1, ULDFLG(R1) :SET THE "UNLOAD IN QUEUE" FLAG
2311 010624 105761 007546 25: TSTB 65:DRVACT(R1) :IS THIS DRIVE ACTIVE?
2312 010630 001043 BNE 65:85 :BR IF YES
2313 010632 004737 010764 JSR 65:PC, OPT :CALL THE OPTIMIZER
2314 010636 000440 BR 65:85
2315 010640 012762 120000 000016 35: MOV 65:#BIT15!BIT13, 16 R2) :SET THE "UNLOAD IN QUEUE" ERROR FLAG
2316 010646 000434 BR 65:85 :EXIT
2317 010650 004737 012074 45: JSR 65:PC, C17 :GO HANDLE THE PARITY ERROR

```

```

2210 010654 000431 BR 9$
2211 010656 004037 015760 5$: JSR R0,DRVQUE ;PUT REQUEST IN QUEUE
2212 010662 000431 BR 9$ ;QUEUE IS FULL
2213 010664 032714 000100 BIT #BIT06,(R4) ;IS 'IE' SET ALREADY ?
2214 010670 001023 BNE 9$ ;BR IF IT IS
2215 010572 004737 015316 JSR PC,SET.IE ;SET INTERRUPT
2216 010676 000420 BR 9$ ;RETURN, REQUEST IN QUEUE
2217 010700 105761 007556 6$: TSTB DRVSTA(R1) ;SEE IF DRIVE OFFLINE OR UNSAFE
2218 010704 002412 BLT 7$ ;BR IF UNSAFE
2219 010706 012762 140000 000016 MOV #BIT15:BIT14.16,R2) ;SET OFFLINE ERROR INDICATOR
2220 010714 105761 007556 TSTB DRVSTP(R1) ;SEE IF OFFLINE OR NONEXISTENT
2221 010720 001007 BNE 9$ ;BR IF OFFLINE
2222 010722 012762 100002 000016 MOV #BIT15:BIT01.16,R2) ;REPORT DRIVE NONEXISTENT
2223 010730 000403 BR 8$ ;GO TO EXIT
2224 010732 012762 110000 000016 7$: MOV #BIT15:BIT12.16,R2) ;DRIVE IS UNSAFE
2225 010740 104413 9$: RESREG ;RESTORE R0 - R5
2226 010742 005720 TST (R0)+ ;SETUP FOR NORMAL RETURN
2227 010744 000401 BR 10$ ;FINISH UP, THEN EXIT
2228 010746 104413 9$: RESREG ;RESTORE R0 - R5
2229 010750 005720 10$: TST (R0)+ ;CORRECT THE RETURN ADDRESS
2230 010752 105037 007622 CLRB ACTDRV ;CLEAR "ACTIVE DRIVER" FLAG
2231 010756 012637 177776 MOV (SP)+,3*PS ;RETURN "PS" TO USER LEVEL
2232 010762 000200 RTS R0 ;RETURN TO CALLER

;OPTIMIZER-CALLED FOR A PARTICULAR DRIVE
;CALL
;MOV #DRVNUM,R1 ;DRIVE NUMBER TO R1
;JSR PC,OPT ;SETUP A COMMAND
;OPT: SAVREG ;SAVE R0 - R5
;MOV 3*PS, -(SP) ;SAVE PROC. STATUS
;BICB ATABIT(R1),SRCHW ;CLEAR "SEARCH WAIT" KEY
;JSR PC,GETREQ ;GET "DPB" POINTER OF REQUEST
;TST R2 ;IS THERE A REQUEST IN QUEUE?
;BEQ 7$ ;NO--BRANCH TO EXIT
;BIT #BIT12,RPDS1(R4) ;IS MOL STILL SET ?
;BNE 9$ ;BR IF NOT
;BIT #BIT6,RPDS1(R4) ;IS VV SET ?
;BNE 10$ ;BR IF IT IS
;JSR R0,DRVINT ;SEE IF DRIVE STILL ONLINE ?
;BR 6$ ;PARITY OR 'DVA' NOT SE
;TSTB DRVSTA(R1) ;IS DRIVE ONLINE?
;BGT 1$ ;YES--BRANCH
;JSR PC,POPQUE ;NO--REMOVE REQUEST FROM QUEUE
;MOV #BIT15:BIT14.16,R2) ;SET OFFLINE STATUS ERROR INDICATOR
;TSTB DRVSTA(R1) ;IS DRIVE UNSAFE ?
;BPL 8$ ;BR TO EXIT IF NOT
;MOV #BIT15:BIT12.16,R2) ;SET UNSAFE STATUS ERROR INDICATOR
;BR 8$ ;BRANCH TO EXIT
;MOV #111, -(SP) ;LOAD COMMAND ONTO THE STACK
;JSR R0,WRT.RP ;LOAD THE REGISTER
;RPCS1 ;REGISTER INCREMENT
;6$ ;ERROR RETURN ADDRESS

```

H05

MD-11-CZRIC-A, RPO4 5 6 HEAD ALIGNMENT PROGRAM MACY!! 27(655) 27-APR-76 16:34 PAGE 47
 CZRIC.009 SINGLE DUAL PORT RH11 RPO4 5 6 DRIVER (REV 1.0)

SEG CC58

```

2364 011110 032714 004000      BIT      #BIT11,(R4)      :DRIVE AVAILABLE ?
2365 011114 001427      SEQ      5$              :BR IF NOT
2366 011116 122762 000150 000002 CMPB     #150,2(R2)      :IS THE REQUEST FOR I/O?
2367 011124 002403      BLT      2$              :YES--BRANCH
2368 011126 004737 011460      JSR      PC,C14          :CALL THE COMMAND INITIATOR
2369 011132 000440      BR       8$              :BRANCH TO EXIT
2370 011134 005737 007670 2$:   TST      DTUW          :DATA TRANSFER UNDERWAY?
2371 011140 002012      BGE      4$              :YES--GO START A SEARCH
2372 011142 005737 007646      TST      SEEKFG         :DO IMPLIED SEEKS?
2373 011146 100404      BMI      3$              :YES---BRANCH
2374 011150 004037 012430      JSR      RD,LA          :NO--DO LOOK AHEAD
2375 011154 000427      BR       8$              :RETURN HERE ON A PARITY ERROR
2376 011156 000403      BR       4$              :GO START A SEARCH
2377 011160 004737 011244 3$:   JSR      PC,C11          :START A DATA TRANSFER
2378 011164 000423      BR       8$              :
2379 011166 004737 011352 4$:   JSR      PC,C13          :START A SEARCH
2380 011172 000420      BR       8$              :GO TO THE EXIT
2381 011174 112761 177777 007606 5$:  MOVB     #-1,DPRQS(R1) :SET PORT REQUEST INDICATOR
2382 011202 010103      MOV      R1,R3          :SET UP TO ADDRESS WORDS
2383 011204 006303      ASL      R3              :CONVERT TO WORD INDEX
2384 011206 012763 023420 007650      MOV      #10000.,TIMER(R3) :START 10 SEC TIMER
2385 011214 000402      BR       7$              :EXIT
2386 011216 004737 012074 6$:   JSR      PC,C17          :PROCESS THE PARITY ERROR
2387 011222 032714 000100 7$:   BIT      #BIT06,(R4)      :SEE IF 'IE' ALREADY SET
2388 011226 001002      BNE      8$              :BR IF SET
2389 011230 004737 015316      JSR      PC,SET,IE      :SET "IE" WITHOUT A "TRE"
2390 011234 012637 177776 8$:   MOV      (SP)+,3*PS      :RESTORE PROC. STATUS
2391 011240 104413      RESREG          :RESTORE R0 - R5
2392 011242 000207      RTS      PC
2393
2394 :COMMAND INITIATOR
2395 :
2396 :CALL
2397 :
2398 :MOV      #DRVNUM,R1      :DRIVE NUMBER
2399 :MOV      #DPB,R2         :ADDRESS OF DPB
2400 :JSR      PC,C1?          :C1?= C11,C13, OR C14
2401 :WHERE:
2402 :C11=DATA TRANSFER
2403 :C12=SEARCH REQUESTED BY DATA XFER
2404 :C14=NOT DATA TRANSFER
2405
2406 011244 004737 016056 C11:   JSR      PC,POPQJE      :REMOVE REQUEST FROM "DRIVES WAIT" QUEUE
2407 011250 010237 007516      MOV      R2,TRANSWT      :PUT REQ. IN TRANSFER WAIT QUEUE
2408 011254 010203      MOV      R2,R3          :DPB ADDRESS TO R3
2409 011256 013704 007704      MOV      RPAOR,R4        :RPCS1 ADDRESS
2410 011262 010164 000010      MOV      R1,RPCS2(R4)      :SELECT DRIVE
2411 011266 062703 000004      ADD      #4,R3          :DESIRED WORD COUNT
2412 011272 062704 000002      ADD      #2,R4          :RPWC ADDRESS
2413 011276 012324      MOV      (R3)+,(R4)+      :LOAD WORD COUNT
2414 011300 012324      MOV      (R3)+,(R4)+      :LOAD BUFFER ADDRESS
2415 011304 004037 015012      MOV      (R3)+,-(SP)      :LOAD SECTOR AND TRACK
2416 011310 000006      JSR      RD,WRT,RP      :CALL THE LOAD(WRITE) ROUTINE
2417 011312 012074      RPD      C17          :INDEX OF REGISTER TO LOAD
2418 :ERROR RETURN ADDRESS

```

418	011314	012346		MOV	(R3)+, -(SP)	;LOAD CYLINDER ADDRESS
419	011316	004037	015012	JSR	RO, WRT.RP	
420	011322	000034		RPCA		
421	011324	012074		CI7		
422	011326	016246	000002	MOV	2(R2), -(SP)	;LOAD "COMMAND+GO", "A173A15", AND "PSEL"
423	011332	004037	015012	JSR	RO, WRT.RP	
424	011336	000000		RPCS1		
425	011340	012074		CI7		
426	011342	010137	007670	MOV	R1, DTUW	;SET "DATA TRANSFER UNDERWAY"
427	011346	000137	012036	JMP	CI5	
428	011352	013704	007704	MOV	RPAOR, R4	;RPCS1 ADDRESS
429	011356	010164	000010	MOV	R1, RPCS2(R4)	;SELECT DRIVE
430	011362	016246	000012	MOV	12(R2), -(SP)	;DESIRED CYLINDER ADDRESS
431	011366	004037	015012	JSR	RO, WRT.RP	
432	011372	000034		RPCA		
433	011374	012074		CI7		
434	011376	116203	000010	MOVB	10(R2), R3	;PICKUP SECTOR ADDRESS
435	011402	163703	007720	SUB	MXWINDW, R3	;BACKUP BY MAX. SEARCH FOR I/O WINDOW
436	011406	002002		BGE	15	
437	011410	062703	000026	ADD	#22, R3	
438	011414	010346		MOV	R3, -(SP)	;COMBINE THE ADJUSTED SECTOR WITH
439	011416	116266	000011	MOVB	11(R2), 1(SP)	;THE DESIRED TRACK
440	011424	004037	015012	JSR	RO, WRT.RP	;LOAD DESIRED TRACK & SECTOR
441	011430	000006		RPDA		
442	011432	012074		CI7		
443	011434	012746	000131	MOV	#131, -(SP)	;START A SEARCH
444	011440	004037	015012	JSR	RO, WRT.RP	
445	011444	000000		RPCS1		
446	011446	012074		CI7		
447	011450	156137	007672	BISB	ATABIT(R1), SRCHWT	;SET "SEARCH WAIT" KEY
448	011456	000567	007620	BR	CI5	
449	011460	013704	007704	MOV	RPAOR, R4	;RPCS1 ADDRESS
450	011464	010164	000010	MOV	R1, RPCS2(R4)	;SELECT DRIVE
451	011470	116203	000002	MOVB	2(R2), R3	;PICKUP THE REQUESTED COMMAND
452	011474	122703	000131	CMFB	#131, R3	;IS IT A SEARCH COMMAND?
453	011500	001007		BNE	15	;BRANCH IF NO
454	011502	016246	000010	MOV	10(R2), -(SP)	;LOAD DESIRED TRACK & SECTOR
455	011506	004037	015012	JSR	RO, WRT.RP	
456	011512	000006		RPDA		
457	011514	012074		CI7		
458	011516	000403		BR	25	;GO LOAD CYLINDER
459	011520	122703	000135	CMFB	#105, R3	;IS IT A SEEK COMMAND
460	011524	001007		BNE	35	;BRANCH IF NO
461	011526	016246	000012	MOV	12(R2), -(SP)	;LOAD DESIRED CYLINDER
462	011532	004037	015012	JSR	RO, WRT.RP	
463	011536	000034		RPCA		
464	011540	012074		CI7		
465	011542	000546		BR	CI6	
466	011544	122703	000115	CMFB	#115, R3	;IS IT AN "OFFSET" COMMAND?
467	011550	001013		BNE	45	;BR IF NO
468	011552	004037	014636	JSR	RO, RO.RP	;MERGE THE OFFSET VALUE INTO RPOF
469	011556	000032		RPOF		;BUT DON'T CHANGE THE UPPER
470	011560	012074		CI7		
471	011562	116216	000001	MOVB	1(R2), (SP)	;BYTE WHEN LOADING THE

2472	011566	004037	015012		JSR	RD,WRT.RP	;REGISTER (RPOF)
2473	011572	000032			RPOF		
2474	011574	012074			CI7		
2475	011576	000530			BR	CI6	;GO START THE COMMAND
2476	011600	122703	000107	4\$:	CMPB	#107,R3	;IS IT A "RECALIBRATE" COMMAND?
2477	011604	001525			BEQ	CI6	;BRANCH IF YES
2478	011606	122703	000117		CMPB	#117,R3	;IS IT A RETURN TO CENTER?
2479	011612	001522			BEQ	CI6	;BRANCH IF YES
2480	011614	122703	000103		CMPB	#103,R3	;IS IT AN "UNLOAD" COMMAND?
2481	011620	001016			BNE	5\$	;BRANCH IF NO
2482	011622	112761	000001	007546	MOVB	#1,DRVACT(R1)	;SET THE DRIVE ACTIVE INDICATOR
2483	011630	105061	007556		CLRB	DRVSTA(R1)	;PUT DRIVE STATUS TO OFFLINE
2484	011634	112761	000001	007624	MOVB	#1,ULDFLG(R1)	;SET "UNLOAD IN PROGRESS" FLAG
2485	011642	010346			MOV	R3,-(SP)	;START THE "UNLOAD" COMMAND
2486	011644	004037	015012		JSR	RD,WRT.RP	
2487	011650	000000			RPCS1		
2488	011652	012074			CI7		
2489	011654	000207			RTS	PC	;RETURN TO USER
2490	011656	122703	000143	5\$:	CMPB	#143,R3	;IS IT A "SET FORMAT" COMMAND?
2491	011662	001014			BNE	6\$	;BRANCH IF NO
2492	011664	004037	014636		JSR	RD,RD.RP	;READ THE OFFSET REGISTER
2493	011670	000032			RPOF		
2494	011672	012074			CI7		
2495	011674	116266	000001	000001	MOVB	1(R2),1(SP)	;COMBINE "FMT22" "ECI" AND "HCI"
2496	011702	004037	015012		JSR	RD,WRT.RP	;LOAD "FMT22", "ECI", AND/OR "HCI".
2497	011706	000032			RPOF		
2498	011710	012074			CI7		
2499	011712	000436			BR	12\$	
2500	011714	122703	000141	6\$:	CMPB	#141,R3	;IS IT A "GET REGISTER" COMMAND?
2501	011720	001023			BNE	10\$	;BRANCH IF NO
2502	011722	016203	000006	7\$:	MOV	6(R2),R3	;POINTS TO 1ST ADDRESS OF WHERE
2503							;TO PUT THE REGISTER(S)
2504	011726	116237	000010	011744	MOVB	10(R2),9\$	;INIT. THE INDEX FOR THE FIRST REG.
2505	011734	116205	000011		MOVB	11(R2),R5	;INDEX OF LAST REG. TO MOVE
2506	011740	004037	014636	8\$:	JSR	RD,RD.RP	;READ RPO4/5/6 REGISTER
2507	011744	000000		9\$:	RPCS1		;INDEX OF REG. TO READ
2508	011746	012074			CI7		
2509	011750	012623			MOV	(SP)+,(R3)+	;GET THE CONTENTS OF RH11/RPO4 5/6 REG.
2510	011752	023705	011744		CMP	9\$,R5	;LAST REG. BEEN READ?
2511	011756	001414			BEQ	12\$	;GET OUT IF YES
2512	011760	062737	000032	011744	ADD	#2,9\$	;INCREASE THE INDEX BY 2
2513	011766	000764			BR	8\$	;LOOP--MORE TO READ
2514	011770	122703	000145	10\$:	CMPB	#145,R3	;IS IT A "SELECT DRIVE" COMMAND?
2515	011774	001405			BEQ	12\$	;BRANCH IF YES
2516	011776	010346		11\$:	MOV	R3,-(SP)	;LOAD THE COMMAND
2517	012000	004037	015012		JSR	RD,WRT.RP	
2518	012004	000000			RPCS1		
2519	012006	012074			CI7		
2520	012010	004737	016056	12\$:	JSR	PC,POPQUE	;REMOVE REG. FROM QUEUE
2521	012014	052762	000200	000016	BIS	#BIT07,16(R2)	;SET THE "DONE" BIT
2522	012022	005737	007644		TST	SAVEFG	;SAVE THE RH11/RPO4/5/6 REGISTERS
2523	012026	100002			BPL	13\$	;BRANCH IF NO
2524	012030	004737	015200		JSR	PC,SVRH11	;YES--GO SAVE THE REGISTERS
2525	012034	000207		13\$:	RTS	PC	;RETURN TO USER

```

2526 012036 006301          CIS:  ASL      R1
2527 012040 012761 001750 007650  MOV     #1000.,TIMER(R1)      ;SET A ONE SECOND TIMER
2528 012046 006201          ASR      R1
2529 012050 112761 000001 007546  MOV     #1,DRVACT(R1)      ;SET THE DRIVE ACTIVE
2530 012056 000207          RTS      PC      ;RETURN TO THE USER
2531 012060 010346          C16:  MOV     R3,-(SP)      ;LOAD THE COMMAND
2532 012062 004037 015012      JSR     RD,WRT.RP
2533 012066 000000          RPCS1
2534 012070 012074          C17
2535 012072 000761          BR
2536 012074 032764 010000 000010 C17:  BIT     #BIT12,RPCS2(R4)      ;DRIVE NON-EXISTENT ?
2537 012102 001034          BNE      C18      ;BR IF YES
2538 012104 005702          1$:  TST     R2      ;ANYTHING IN QUEUE ?
2539 012106 001405          BEQ     C17B      ;BR IF NOT
2540 012110 012762 104000 000016  MOV     #BIT15:BIT11,16(R2)      ;SET "PARITY" ERROR INDICATOR
2541 012116 004737 015200      JSR     PC,SVRH11      ;GO SAVE THE RH11/RPO4/5/6 REGISTERS
2542 012122 012746 000111      C17B: MOV     #11,-(SP)      ;DO A "DRIVE CLEAR"
2543 012126 004037 015012      JSR     RD,WRT.RP
2544 012132 000000          RPCS1
2545 012134 012174          C18
2546 012136 004737 015740      JSR     PC,EMPTYQ      ;EMPTY THE QUEUE
2547 012142 105061 007624      CLRB    ULDFLG(R1)      ;CLEAR THE UNLOAD IN QUEUE FLAG
2548 012146 105061 007546      CLRB    DRVACT(R1)      ;DRIVE IS IDLE
2549 012152 020137 007670      CMP     R1,DTUW      ;IF THIS DRIVE HAD AN I/O REQUEST
2550 012156 001005          BNE      1$      ;IN PROGRESS CLEAR ALL OF THE FLAGS
2551 012160 005037 007616      CLR     TRANSW
2552 012164 012737 177777 007670  MOV     #-1,DTUW
2553 012172 000207          1$:  RTS      PC
2554 012174 104412          C18:  SAVREG      ;SAVE R0 - R5
2555 012176 032764 010000 000010  BIT     #BIT12,RPCS2(R4)      ;IS 'NED' SET ?
2556 012204 001002          BNE      1$      ;BR IF YES
2557 012206 005001          CLR     R1
2558 012210 005003          CLR     R3
2559 012212 105761 007546          1$:  TSTB    DRVACT(R1)      ;DRIVE ACTIVE?
2560 012216 001443          BEQ     5$      ;BRANCH IF NO
2561 012220 013702 007616      MOV     TRANSW,R2      ;GET THE "TRANSFER WAIT" QUEUE
2562 012224 020137 007670      CMP     R1,DTUW      ;DID THIS DRIVE HAVE AN I/O IN PROGRESS?
2563 012230 001402          BEQ     2$      ;BRANCH IF YES
2564 012232 004737 016034      JSR     PC,GETREQ      ;GET THE DPB POINTER
2565 012236 005702          2$:  TST     R2      ;QUEUE ENTRY FOR DRIVE ?
2566 012240 001415          BEQ     4$      ;BR IF NOT
2567 012242 032764 010000 000010  BIT     #BIT12,RPCS2(R4)      ;'NED' SET ?
2568 012250 001404          BEQ     3$      ;BR IF NOT
2569 012252 012762 100002 000016  MOV     #BIT15:BIT10,16(R2)      ;SET 'DRIVE NON-EXISTENT' INDICATOR
2570 012260 000405          BR      4$      ;CONTINUE
2571 012262 012762 102000 000016  3$:  MOV     #BIT15:BIT10,16(R2)      ;SET "NON-CLEARABLE PARITY" ERROR INDICATOR
2572 012270 004737 015200      JSR     PC,SVRH11      ;SAVE RH11/RPO4/5/6 REGISTERS
2573 012274 012763 177777 007650  4$:  MOV     #-1,TIMER(R3)      ;STOP THE TIMER
2574 012302 105061 007546      CLRB    DRVACT(R1)      ;SET "DRIVE ACTIVE" TO IDLE
2575 012306 020137 007670      CMP     R1,DTUW      ;IS THIS DRIVE SETUP FOR A TRANSFER
2576 012312 001005          BNE      5$      ;BR IF NOT
2577 012314 012737 177777 007670  MOV     #-1,DTUW      ;RESET THE INDICATOR
2578 012322 005037 007616      CLR     TRANSW      ;CLEAR THE TRANSFER QUEUE
2579 012326 105061 007524          5$:  CLRB    ULDFLG(P1)      ;CLEAR UNLOAD FLAG

```

```

2580 012332 032764 010000 000010      BIT      #BIT12,RPCS2(R4)      ;'NEC' SET ?
2591 012340 001021      BNE      6$              ;BR IF YES
2582 012342 005201      INC      R1              ;MOVE TO THE NEXT DRIVE
2583 012344 062703 003002      ADD      #2,R3
2584 012350 042701 177770      BIC      #1C7,R1
2585 012354 001316      BNE      1$              ;BRANCH IF MORE DRIVES
2586 012356 012737 177777 007670      MOV      #-1,DTUW      ;NO DATA TRANSFERS UNDERWAY
2587 012364 005037 007616      CLR      TRNSWT      ;CLEAR THE 'TRANSFER WAIT' QUEUE
2588 012370 004737 015662      JSR      PC,CLRQUE      ;CLEAR ALL OF THE REQUEST QUEUES
2589 012374 012764 000040 00C010      MOV      #BIT05,RPCS2(R4) ;DO A MASSBUS INIT.
2590 012402 000406      BR      7$              ;CONTINUE
2591 012404 004737 015740      JSR      PC,EMPTYQ      ;CLEAR THE DRIVE'S QUEUE
2592 012410 105061 007556      CLRB     DRVSTA(R1)      ;SET DRIVE TO OFFLINE
2593 012414 105061 007566      CLRB     DRVTP(R1)      ;CLEAR THE DRIVE TYPE INDICATOR
2594 012420 004737 015316      JSR      PC,SET.IE      ;SET "IE" WITHOUT "TRE"
2595 012424 104413      RESREG      ;RESTORE R0 - R5
2596 012426 00C207      RTS      PC              ;RETURN
2597
2598      ;LOOK AHEAD ROUTINE
2599
2600      ;CALL
2601      ;
2602      ;      MOV      #DRVNUM,R1      ;DRIVE NUMBER
2603      ;      MOV      #DPB,R2        ;POINT TO DPB
2604      ;      JSR      R0,LA          ;GO CHECK THE WINDOW
2605      ;      RETURN1      ;ERROR RETURN
2606      ;      RETURN2      ;START A SEARCH
2607      ;      RETURN3      ;START A DATA TRANSFER
2608 012430 013704 007704      LA:      MOV      RPADR,R4      ;GET RPCS1'S ADDRESS
2609 012434 01C164 000010      MOV      R1,RPCS2(R4)      ;SELECT DRIVE
2610 012440 004037 014636      JSR      R0,RD.RP      ;READ CURRENT CYLINDER
2611 012444 000036      RPCC
2612 012446 012560      4$
2613 012450 022662 000012      CMP      (SP)+,12(R2)      ;ERROR RETURN ADDRESS
2614      ;IS CURRENT CYLINDER=DESIRED
2615      ;CYLINDER?
2616 012454 001037      BNE      3$              ;EXIT IF NO
2617 012456 105261 007634 007712      INCB     LACNT(R1)      ;INCREMENT THE LOOK AHEAD COUNT
2618 012462 126137 007634      CMPB     LACNT(R1),MXLACT      ;EXCEED MAX?
2619 012470 003026      BGT      2$              ;BRANCH IF YES
2620 012472 116203 000010      MOVB     10(R2),R3      ;GET DESIRED SECTOR ADDRESS AND
2621 012476 000303      SWAB     R3              ;MULT. BY 64--ALIGN WITH
2622 012500 006203      ASR      R3              ;LOOK AHEAD REGISTER
2623 012502 006203      ASR      R3
2624 012504 012737 000340 177776      MOV      #340,0#PS      ;PRIORITY LEVEL "7"
2625 012512 004037 014636      JSR      R0,RD.RP      ;READ LOOK AHEAD REGISTER
2626 012516 000020      RPLA
2627 012520 012560      4$
2628 012522 162603      SUB      (SP)+,R3      ;CALCULATE THE DELTA
2629 012524 002002      BGE      1$              ;MAKE THE DELTA POSITIVE
2630 012526 062703 002600      ADD      #22,*64,>,R3      ;CHECK THE DELTA TO SEE
2631 012532 023703 007714      CMP      MXDLTA,R3      ;IF IT IS WITHIN THE
2632 012536 002406      BLT      3$              ;WINDOW---IF YES, ZERO
2633 012540 023703 007716      CMP      MNDLTA,R3      ;THE LOOK AHEAD COUNT
2634 012544 002003      BGE      3$

```

```

2634 012546 105061 007634 2$: CLR B LACNT(R1) ;AND TAKE THE I/O EXIT
2635 012552 005720 TST (R0)+
2636 012554 005720 3$: TST (R0)+ ;ADJUST THE RETURN ADDRESS
2637 012556 000402 BR 5$ ;EXIT
2638 012560 004737 012074 4$: JSR PC,C17 ;PROCESS THE ERROR
2639 012564 000200 5$: RTS R0 ;RETURN
2640
2641 ;INTERRUPT SERVICE ROUTINE
2642
2643 012566 112737 000001 007622 ISR: MOV B #1,ACTDRV ;SET "ACTIVE DRIVER" FLAG
2644 012574 104412 SAVREG ;SAVE R0 - R5
2645 012576 013704 007704 MOV RPADR,R4 ;ADDRESS OF RHSCSI
2646 012602 013701 007670 MOV DTUW,R1 ;GET "DATA TRANSFER UNDERWAY" INDICATOR
2647 012606 002403 BLT 1$ ;BRANCH IF NO DATA TRANSFER UNDERWAY
2648 012610 004737 012632 JSR PC,TD ;CALL TRANSFER DONE
2649 012614 000402 BR 2$ ;EXIT
2650 012616 004737 012772 1$: JSR PC,SC ;CALL SPECIAL CONDITIONS
2651 012622 104413 2$: RESREG ;RESTORE R0 - R5
2652 012624 105037 007622 CLR B ACTDRV ;CLEAR "ACTIVE DRIVER" FLAG
2653 012630 000002 RTI ;RETURN
2654
2655 ;TRANSFER DONE ROUTINE
2656
2657 012632 105061 007546 TD: CLR B DRVACT(R1) ;SET DRIVE ACTIVE INDICATOR TO IDLE
2658 012636 012737 177777 007670 MOV #-1,DTUW ;NO DATA TRANSFERS UNDERWAY
2659 012644 006301 ASL R1
2660 012646 012761 177777 007650 MOV #-1,TIMER(R1) ;CANCEL TIMEOUT
2661 012654 006201 ASR R1
2662 012656 013702 007616 MOV TRNSWT,R2 ;GET "DPB" ADDRESS FROM THE
2663 012662 005037 007616 CLR TRNSWT ;TRANSFER WAIT QUEUE--CLEAR QUEUE
2664 012666 052762 000200 000016 BIS #BIT07,16(R2) ;SET DONE
2665 012674 010164 000010 MOV R1,RPCS2(R4) ;SELECT THE DRIVE
2666 012700 004037 014636 JSR R0,RD.RP ;TRANSFER ERROR(TRE=1)?
2667 012704 000000 RPCS1
2668 012706 012074 C17
2669 012710 006126 ROL (SP)+
2670 012712 100413 BMI 3$ ;BR IF YES
2671 012714 005737 007644 TST SAVEFG ;SAVE THE RH11/RPO4/5/6 REGISTERS
2672 012720 100002 BPL 1$ ;BRANCH IF NO
2673 012722 004737 015200 JSR PC,SVRH11 ;YES--SAVE THE REGISTERS
2674 012726 004737 010764 1$: JSR PC,OPT ;CALL OPTIMIZER
2675 012732 000417 BR SC ;CHECK OTHER DRIVES
2676 012734 012714 000113 2$: MOV #113,(R4) ;RELEASE THE DRIVE
2677 012740 000414 BR SC ;CHECK FOR OTHER DRIVES
2678 012742 052762 100100 000016 3$: BIS #BIT15:BIT06,16(R2) ;SET DATA ERROR FLAG
2679 012750 004737 015740 JSR PC,EMPTYQ ;EMPTY THE "DRIVE'S WAIT" QUEUE
2680 012754 004737 015200 JSR PC,SVRH11 ;SAVE THE RH11/RPO4/5/6 REGISTERS
2681 012760 012714 040111 MOV #40111,(R4) ;ISSUE A "DRIVE CLEAR"
2682 012764 012714 000113 MOV #113,(R4) ;ISSUE A RELEASE TO THE DRIVE
2683 012770 000400 BR SC ;CHECK FOR OTHER DRIVES
2684
2685 ;SPECIAL CONDITION ROUTINE
2686
2687 012772 116403 000016 SC: MOV B RPAS(R4),R3 ;READ "RPAS"

```

```

2688 012776 001012      BNE      2$          ;BRANCH IF ANY 'ATA' BITS SET
2689 013000 004037      JSR      RD, RD.RP      ;READ CONTROL AND STATUS REGISTER
2690 013004 000000      RPCS1
2691 013006 012174      CIB
2692 013010 106126      ROLB      (SP)+      ;IS "IE"=1?
2693 013012 100403      BMI      1$          ;YES, NO DRIVES TO CHECK
2694 013014 104001      ERROR      1          ;REPORT AN ILLEGAL INTERRUPT
2695 013016 004737      JSR      PC, SET.IE      ;SET INTERRUPT ENABLE
2696 013022 000207      RTS      PC          ;RETURN
2697 013024 005046      CLR      -(SP)      ;PROCESS ALL DRIVES THAT HAVE
2698 013026 110316      MOV      R3, (SP)      ;AN "ATA"=1
2699 013030 012703      MOV      #1, R3
2700 013034 005001      CLR      R1
2701 013036 030316      SC3:    BIT      R3, (SP)      ;ATA=1?
2702 013040 001005      BNE      SC5          ;YES--BRANCH
2703 013042 005201      SC4:    INC      R1          ;MOVE TO THE NEXT DRIVE
2704 013044 106303      ASLB      R3
2705 013046 001373      BNE      SC3          ;BRANCH IF MORE TO CHECK?
2706 013050 005726      TST      (SP)+      ;CLEAN OFF THE STACK
2707 013052 000207      RTS      PC          ;RETURN TO USER
2708 013054 105761      SC5:    TSTB     DPINT(R1)      ;INITIALIZING THE DRIVE ?
2709 013060 001402      BEQ      1$          ;BR IF NOT
2710 013062 000137      JMP      SC13         ;PROCESS THE DRIVE
2711 013066 105761      1$:    TSTB     DPRQS(R1)      ;PORT REQUEST OUTSTANDING ?
2712 013072 001402      BEQ      2$          ;BR IF NOT
2713 013074 000137      JMP      SC13         ;START THE OUTSTANDING COMMAND
2714 013100 105761      2$:    TSTB     DRVSTA(R1)      ;CHECK THE DRIVE STATUS
2715 013104 003025      BIT      5$          ;BRANCH IF ONLINE
2716 013106 105761      TSTB     ULDFLG(R1)      ;UNLOAD IN PROGRESS?
2717 013112 003422      BLE      5$          ;BRANCH IF NOT
2718 013114 004737      JSR      PC, GETREQ      ;GET DPB POINTER
2719 013120 004737      JSR      PC, SVRH11      ;SAVE THE RH11, RPO4, 5/6 REGISTERS
2720 013124 004737      JSR      PC, SC12      ;SAVE RPO51, RPER1, RPER2, AND RPER3
2721 013130 105761      TSTB     DRVSTA, R1      ;ALSO DO A DRIVE INIT, DRVINT,
2722 013134 003416      BLE      6$          ;DID DRIVE COME ONLINE?
2723 013136 032737      BIT      #BIT14, RPERRS      ;NO---BRANCH
2724 013144 001002      BNE      3$          ;WAS THERE AN ERROR?
2725 013146 000137      JMP      SC11         ;BR IF ERROR
2726 013152 013705      MOV      RPERRS+2, R5      ;NO ERROR
2727 013156 000475      BR      SC6A          ;YES -- PICKUP RPER1 AND
2728 013160 105761      3$:    TSTB     DRVACT, R1      ;GO PROCESS THE ERROR
2729 013164 001027      BNE      SC6          ;DRIVE ACTIVE WITH COMMAND OR ERROR RECOVERY ?
2730 013166 004737      JSR      PC, SC12      ;BR IF EITHER
2731 013172 105761      6$:    TSTB     DPINT(R1)      ;SAVE RPO51, RPER1, RPER2, AND RPER3
2732 013176 001321      BNE      SC4          ;ALSO DO A DRVINT
2733 013200 105761      TSTB     DRVSTA, R1      ;TRYING TO INIT THE DRIVE ?
2734 013204 100412      BMI      7$          ;BR IF YES, CHECK ON MORE DRIVES
2735 013206 032737      BIT      #BIT13, RPERRS+6      ;CHECK ON DRIVE'S STATUS
2736 013214 001011      BNE      8$          ;BR IF UNSAFE
2737 013216 012746      MOV      #113, - SP      ;ADDRESS PLUG CHANGED ?
2738 013222 004037      JSR      PC, SET.RP      ;BR IF YES
2739 013226 000000      RPCS1      ;RELEASE COMMAND
2740 013226 000000      RPCS1      ;WRITE THE COMMAND INTO RPCS1
2741 013226 000000      RPCS1      ;REGISTER INDEX

```

[illegible]

[illegible]

```

:      MOV      #DRVNUM,R1      :DRIVE NUMBER
:      JSR      PC,STO          :CALL
:      RETURN
:
:      STO:      MOV      R1,-(SP)      :SAVE R1
:               MOV      R3,-(SP)      :SAVE R3
:               MOV      RPHDR,R4      :SET ADDRESS OF "RPOS1"
:               MOV      R1,RPCS2(R4)  :SELECT THE DRIVE
:               JSR      RD,RC,RP      :READ "DRIVE STATUS REG"
:               RPOS1
:               STJ5
:               TSTB      (SP)+        :IS "DRY"=1?
:               BMT       ST02         :BR IF YES
:               ST01:      TSTB      OPINT(R1) :TRYING TO INITIALIZE THE DRIVE ?
:               BNE      ST02         :BR IF YES
:               TSTB      CFRQS,R1     :OUTSTANDING PORT REQUEST FOR THE DRIVE ?
:               BNE      ST02         :BR IF YES
:               MOV      TRANSW,R2     :PICKUP TRANSFER WAIT QUEUE
:               CMP      R1,DTUW       :TRANSFER UNDERWAY ON THIS DRIVE?
:               BEQ      1$           :BRANCH IF YES
:               JSR      PC,GETREQ      :GET DPB ADDRESS
:               BIS      #BIT15!BIT09,16(R2) :SET THE ERROR FLAGS
:               JSR      PC,SVRH11     :SAVE RH11 RPO4 5 6 REGISTERS
:               MOV      #BIT05,RPCS2(R4) :INIT THE MASS BUS
:               CLRB      DRVACT(R1)   :DRIVE IS IDLE
:               CLRB      ULDFLG(R1)   :CLEAR THE UNLOAD FLAG
:               CLRA      R1           :START WITH DRIVE 0
:               CLRA      R3
:               JSR      RD,DRVINT      :INIT. THIS DRIVE
:               BR       ST05          :PARITY ERROR RETURN
:               TSTB      DRVACT,R1    :DRIVE IDLE BEFORE THE INIT.
:               BEQ      4$           :YES--BRANCH
:               MOV      TRANSW,R2     :GET TRANSFER WAIT QUEUE
:               CMP      DTUW,R1       :WAS THERE I/O ON THIS DRIVE?
:               BEQ      3$           :YES--BRANCH
:               JSR      PC,GETREQ      :GET THE DPB POINTER FROM QUEUE
:               BIS      #BIT15!BIT08,16(R2) :INFORM USER OF INIT.
:               CLRB      DRVACT(R1)   :SET DRIVE ACTIVE TO IDLE
:               CLRB      ULDFLG(R1)   :NO UNLOAD
:               MOV      #-1,TIMER(R3) :STOP THE TIMER
:               TST      (R3)+        :UPDATE THE INDEX
:               INC      R1           :INCREMENT THE DRIVE NUMBER
:               CMP      #8.,R1       :LAST DRIVE BEEN CHECKED?
:               BGT      2$           :NO--LOOP
:               MOV      #-1,DTUW     :NO DATA TRANSFERS UNDERWAY
:               CLRA      TRANSW       :CLEAR TRANSFER WAIT QUEUE
:               JSR      PC,CLEARQLE    :CLEAR ALL REQUEST QUEUES
:               BR       ST09          :EXIT
:               ST02:      MOV      RPHAS(R4),R5 :READ ATTENTION REG
:               BITB      ATABIT(R1),R5 :IS ATTENTION FOR THIS DRIVE UP ?
:               BNE      ST03         :YES--BRANCH
:               TSTB      OPINT(R1)   :TRYING TO INITIALIZE THE DRIVE ?
:               BNE      ST06         :BR IF YES - DRIVE NOT ONLINE
:               TSTB      CFRQS,R1    :OUTSTANDING PORT REQUEST FOR THE DRIVE

```



```

00010 014700 032716 000010 ADD #RPCS2,(SP) ;FORM THE ADDRESS OF RPCS2
00011 014704 032736 010000 BIT #BIT12,(SP)+ ;CHECK THE 'NED' BIT
00012 014710 001035 BNE RD.RP3 ;BR IF DRIVE NON-EXISTENT
00013 014712 017746 173756 MOV JRPADR,-(SP) ;READ RPCS1
00014 014716 032716 030000 BIT #BIT13,(SP) ;DID MCPE SET?
00015 014722 001000 BNE IS ;BRANCH IF YES
00016 014724 022620 CMP (SP)+,(RC)+ ;ADJUST FOR RETURN
00017 014726 000430 BR RD.RP4 ;EXIT
00018 014730 :3:
00019 014732 104003 ERROR 3 ;REPORT "MCPE" ERROR
00020 014732 005737 007670 TST DTUM ;DATA TRANSFER UNDERWAY?
00021 014736 100405 BMI 2$ ;NO--BRANCH
00022 014740 032716 040000 BIT #BIT14,(SP) ;NO--"TRE"=1?
00023 014744 001402 BEQ 2$ ;NO--BRANCH
00024 014746 005726 JST (SP)+ ;YES--CLEAN OFF THE STACK AND
00025 014750 000415 BR RD.RP3 ;TAKE THE FATAL ERROR EXIT
00026 014752 052716 040000 2$: BIS #BIT14,(SP) ;CLEAR "MCPE" BY SENDING A "1" TO "TRE"
00027 014756 000316 SWAB (SP) ;POSITION BEFORE WRITING
00028 014760 013737 007704 014774 MOV RPADR,3$ ;FORM ADDRESS OF HIGH BYTE
00029 014766 005237 014774 INC 3$
00030 014772 112637 MOVB (SP)+,3(PC)+ ;WRITE THE HIGH BYTE OF RPCS1
00031 014774 000000 3$: .WORD 0 ;ADDRESS STORAGE
00032 014776 005327 DEC (PC)+ ;EXCEEDED MAX. RETRYS
00033 015000 000003 RD.RP2: .WORD 3
00034 015002 002326 BGE RD.RP1 ;BRANCH IF NO
00035 015004 011000 RD.RP3: MOV (RC),RD ;FATAL ERROR EXIT
00036 015006 012615 MOV (SP)+,(SP)
00037 015008 000200 RD.RP4: RTS RD
00038 015010 :ROUTINE TO WRITE A REGISTER
00039 015012 :CALL
00040 015014 :
00041 015016 : MOV DATA,-(SP) ;DATA TO BE LOADED ON THE STACK
00042 015018 : JSR RD.WRT.RP ;CALL THE ROUTINE TO LOAD(WRITE) THE REG.
00043 015020 : INDEX ;INDEX OF THE REGISTER TO BE LOADED
00044 015022 : ERRACR ;ADDRESS TO RETURN TO ON AN ERROR
00045 015024 : RETURN ;ERROR FREE RETURN
00046 015026 :
00047 015028 :
00048 015030 :
00049 015032 :
00050 015034 :
00051 015036 :
00052 015038 :
00053 015040 :
00054 015042 :
00055 015044 :
00056 015046 :
00057 015048 :
00058 015050 :
00059 015052 :
00060 015054 :
00061 015056 :
00062 015058 :
00063 015060 :
00064 015062 :
00065 015064 :
00066 015066 :
00067 015068 :
00068 015070 :
00069 015072 :
00070 015074 :
00071 015076 :
00072 015078 :
00073 015080 :
00074 015082 :
00075 015084 :
00076 015086 :
00077 015088 :
00078 015090 :
00079 015092 :
00080 015094 :
00081 015096 :
00082 015098 :
00083 015100 :
00084 015102 :
00085 015104 :
00086 015106 :
00087 015108 :
00088 015110 :
00089 015112 :
00090 015114 :
00091 015116 :
00092 015118 :
00093 015120 :
00094 015122 :
00095 015124 :
00096 015126 :
00097 015128 :
00098 015130 :
00099 015132 :
00100 015134 :
00101 015136 :
00102 015138 :
00103 015140 :
00104 015142 :
00105 015144 :
00106 015146 :
00107 015148 :
00108 015150 :
00109 015152 :
00110 015154 :
00111 015156 :
00112 015158 :
00113 015160 :
00114 015162 :
00115 015164 :
00116 015166 :
00117 015168 :
00118 015170 :
00119 015172 :
00120 015174 :
00121 015176 :
00122 015178 :
00123 015180 :
00124 015182 :
00125 015184 :
00126 015186 :
00127 015188 :
00128 015190 :
00129 015192 :
00130 015194 :
00131 015196 :
00132 015198 :
00133 015200 :
00134 015202 :
00135 015204 :
00136 015206 :
00137 015208 :
00138 015210 :
00139 015212 :
00140 015214 :
00141 015216 :
00142 015218 :
00143 015220 :
00144 015222 :
00145 015224 :
00146 015226 :
00147 015228 :
00148 015230 :
00149 015232 :
00150 015234 :
00151 015236 :
00152 015238 :
00153 015240 :
00154 015242 :
00155 015244 :
00156 015246 :
00157 015248 :
00158 015250 :
00159 015252 :
00160 015254 :
00161 015256 :
00162 015258 :
00163 015260 :
00164 015262 :
00165 015264 :
00166 015266 :
00167 015268 :
00168 015270 :
00169 015272 :
00170 015274 :
00171 015276 :
00172 015278 :
00173 015280 :
00174 015282 :
00175 015284 :
00176 015286 :
00177 015288 :
00178 015290 :
00179 015292 :
00180 015294 :
00181 015296 :
00182 015298 :
00183 015300 :
00184 015302 :
00185 015304 :
00186 015306 :
00187 015308 :
00188 015310 :
00189 015312 :
00190 015314 :
00191 015316 :
00192 015318 :
00193 015320 :
00194 015322 :
00195 015324 :
00196 015326 :
00197 015328 :
00198 015330 :
00199 015332 :
00200 015334 :
00201 015336 :
00202 015338 :
00203 015340 :
00204 015342 :
00205 015344 :
00206 015346 :
00207 015348 :
00208 015350 :
00209 015352 :
00210 015354 :
00211 015356 :
00212 015358 :
00213 015360 :
00214 015362 :
00215 015364 :
00216 015366 :
00217 015368 :
00218 015370 :
00219 015372 :
00220 015374 :
00221 015376 :
00222 015378 :
00223 015380 :
00224 015382 :
00225 015384 :
00226 015386 :
00227 015388 :
00228 015390 :
00229 015392 :
00230 015394 :
00231 015396 :
00232 015398 :
00233 015400 :
00234 015402 :
00235 015404 :
00236 015406 :
00237 015408 :
00238 015410 :
00239 015412 :
00240 015414 :
00241 015416 :
00242 015418 :
00243 015420 :
00244 015422 :
00245 015424 :
00246 015426 :
00247 015428 :
00248 015430 :
00249 015432 :
00250 015434 :
00251 015436 :
00252 015438 :
00253 015440 :
00254 015442 :
00255 015444 :
00256 015446 :
00257 015448 :
00258 015450 :
00259 015452 :
00260 015454 :
00261 015456 :
00262 015458 :
00263 015460 :
00264 015462 :
00265 015464 :
00266 015466 :
00267 015468 :
00268 015470 :
00269 015472 :
00270 015474 :
00271 015476 :
00272 015478 :
00273 015480 :
00274 015482 :
00275 015484 :
00276 015486 :
00277 015488 :
00278 015490 :
00279 015492 :
00280 015494 :
00281 015496 :
00282 015498 :
00283 015500 :
00284 015502 :
00285 015504 :
00286 015506 :
00287 015508 :
00288 015510 :
00289 015512 :
00290 015514 :
00291 015516 :
00292 015518 :
00293 015520 :
00294 015522 :
00295 015524 :
00296 015526 :
00297 015528 :
00298 015530 :
00299 015532 :
00300 015534 :
00301 015536 :
00302 015538 :
00303 015540 :
00304 015542 :
00305 015544 :
00306 015546 :
00307 015548 :
00308 015550 :
00309 015552 :
00310 015554 :
00311 015556 :
00312 015558 :
00313 015560 :
00314 015562 :
00315 015564 :
00316 015566 :
00317 015568 :
00318 015570 :
00319 015572 :
00320 015574 :
00321 015576 :
00322 015578 :
00323 015580 :
00324 015582 :
00325 015584 :
00326 015586 :
00327 015588 :
00328 015590 :
00329 015592 :
00330 015594 :
00331 015596 :
00332 015598 :
00333 015600 :
00334 015602 :
00335 015604 :
00336 015606 :
00337 015608 :
00338 015610 :
00339 015612 :
00340 015614 :
00341 015616 :
00342 015618 :
00343 015620 :
00344 015622 :
00345 015624 :
00346 015626 :
00347 015628 :
00348 015630 :
00349 015632 :
00350 015634 :
00351 015636 :
00352 015638 :
00353 015640 :
00354 015642 :
00355 015644 :
00356 015646 :
00357 015648 :
00358 015650 :
00359 015652 :
00360 015654 :
00361 015656 :
00362 015658 :
00363 015660 :
00364 015662 :
00365 015664 :
00366 015666 :
00367 015668 :
00368 015670 :
00369 015672 :
00370 015674 :
00371 015676 :
00372 015678 :
00373 015680 :
00374 015682 :
00375 015684 :
00376 015686 :
00377 015688 :
00378 015690 :
00379 015692 :
00380 015694 :
00381 015696 :
00382 015698 :
00383 015700 :
00384 015702 :
00385 015704 :
00386 015706 :
00387 015708 :
00388 015710 :
00389 015712 :
00390 015714 :
00391 015716 :
00392 015718 :
00393 015720 :
00394 015722 :
00395 015724 :
00396 015726 :
00397 015728 :
00398 015730 :
00399 015732 :
00400 015734 :
00401 015736 :
00402 015738 :
00403 015740 :
00404 015742 :
00405 015744 :
00406 015746 :
00407 015748 :
00408 015750 :
00409 015752 :
00410 015754 :
00411 015756 :
00412 015758 :
00413 015760 :
00414 015762 :
00415 015764 :
00416 015766 :
00417 015768 :
00418 015770 :
00419 015772 :
00420 015774 :
00421 015776 :
00422 015778 :
00423 015780 :
00424 015782 :
00425 015784 :
00426 015786 :
00427 015788 :
00428 015790 :
00429 015792 :
00430 015794 :
00431 015796 :
00432 015798 :
00433 015800 :
00434 015802 :
00435 015804 :
00436 015806 :
00437 015808 :
00438 015810 :
00439 015812 :
00440 015814 :
00441 015816 :
00442 015818 :
00443 015820 :
00444 015822 :
00445 015824 :
00446 015826 :
00447 015828 :
00448 015830 :
00449 015832 :
00450 015834 :
00451 015836 :
00452 015838 :
00453 015840 :
00454 015842 :
00455 015844 :
00456 015846 :
00457 015848 :
00458 015850 :
00459 015852 :
00460 015854 :
00461 015856 :
00462 015858 :
00463 015860 :
00464 015862 :
00465 015864 :
00466 015866 :
00467 015868 :
00468 015870 :
00469 015872 :
00470 015874 :
00471 015876 :
00472 015878 :
00473 015880 :
00474 015882 :
00475 015884 :
00476 015886 :
00477 015888 :
00478 015890 :
00479 015892 :
00480 015894 :
00481 015896 :
00482 015898 :
00483 015900 :
00484 015902 :
00485 015904 :
00486 015906 :
00487 015908 :
00488 015910 :
00489 015912 :
00490 015914 :
00491 015916 :
00492 015918 :
00493 015920 :
00494 015922 :
00495 015924 :
00496 015926 :
00497 015928 :
00498 015930 :
00499 015932 :
00500 015934 :
00501 015936 :
00502 015938 :
00503 015940 :
00504 015942 :
00505 015944 :
00506 015946 :
00507 015948 :
00508 015950 :
00509 015952 :
00510 015954 :
00511 015956 :
00512 015958 :
00513 015960 :
00514 015962 :
00515 015964 :
00516 015966 :
00517 015968 :
00518 015970 :
00519 015972 :
00520 015974 :
00521 015976 :
00522 015978 :
00523 015980 :
00524 015982 :
00525 015984 :
00526 015986 :
00527 015988 :
00528 015990 :
00529 015992 :
00530 015994 :
00531 015996 :
00532 015998 :
00533 016000 :
00534 016002 :
00535 016004 :
00536 016006 :
00537 016008 :
00538 016010 :
00539 016012 :
00540 016014 :
00541 016016 :
00542 016018 :
00543 016020 :
00544 016022 :
00545 016024 :
00546 016026 :
00547 016028 :
00548 016030 :
00549 016032 :
00550 016034 :
00551 016036 :
00552 016038 :
00553 016040 :
00554 016042 :
00555 016044 :
00556 016046 :
00557 016048 :
00558 016050 :
00559 016052 :
00560 016054 :
00561 016056 :
00562 016058 :
00563 016060 :
00564 016062 :
00565 016064 :
00566 016066 :
00567 016068 :
00568 016070 :
00569 016072 :
00570 016074 :
00571 016076 :
00572 016078 :
00573 016080 :
00574 016082 :
00575 016084 :
00576 016086 :
00577 016088 :
00578 016090 :
00579 016092 :
00580 016094 :
00581 016096 :
00582 016098 :
00583 016100 :
00584 016102 :
00585 016104 :
00586 016106 :
00587 016108 :
00588 016110 :
00589 016112 :
00590 016114 :
00591 016116 :
00592 016118 :
00593 016120 :
00594 016122 :
00595 016124 :
00596 016126 :
00597 016128 :
00598 016130 :
00599 016132 :
00600 016134 :
00601 016136 :
00602 016138 :
00603 016140 :
00604 016142 :
00605 016144 :
00606 016146 :
00607 016148 :
00608 016150 :
00609 016152 :
00610 016154 :
00611 016156 :
00612 016158 :
00613 016160 :
00614 016162 :
00615 016164 :
00616 016166 :
00617 016168 :
00618 016170 :
00619 016172 :
00620 016174 :
00621 016176 :
00622 016178 :
00623 016180 :
00624 016182 :
00625 016184 :
00626 016186 :
00627 016188 :
00628 016190 :
00629 016192 :
00630 016194 :
00631 016196 :
00632 016198 :
00633 016200 :
00634 016202 :
00635 016204 :
00636 016206 :
00637 016208 :
00638 016210 :
00639 016212 :
00640 016214 :
00641 016216 :
00642 016218 :
00643 016220 :
00644 016222 :
00645 016224 :
00646 016226 :
00647 016228 :
00648 016230 :
00649 016232 :
00650 016234 :
00651 016236 :
00652 016238 :
00653 016240 :
00654 016242 :
00655 016244 :
00656 016246 :
00657 016248 :
00658 016250 :
00659 016252 :
00660 016254 :
00661 016256 :
00662 016258 :
00663 016260 :
00664 016262 :
00665 016264 :
00666 016266 :
00667 016268 :
00668 016270 :
00669 016272 :
00670 016274 :
00671 016276 :
00672 016278 :
00673 016280 :
00674 016282 :
00675 016284 :
00676 016286 :
00677 016288 :
00678 016290 :
00679 016292 :
00680 016294 :
00681 016296 :
00682 016298 :
00683 016300 :
00684 016302 :
00685 016304 :
00686 016306 :
00687 016308 :
00688 016310 :
00689 016312 :
00690 016314 :
00691 016316 :
00692 016318 :
00693 016320 :
00694 016322 :
00695 016324 :
00696 016326 :
00697 016328 :
00698 016330 :
00699 016332 :
00700 016334 :
00701 016336 :
00702 016338 :
00703 016340 :
00704 016342 :
00705 016344 :
00706 016346 :
00707 016348 :
00708 016350 :
00709 016352 :
00710 016354 :
00711 016356 :
00712 016358 :
00713 016360 :
00714 016362 :
00715 016364 :
00716 016366 :
00717 016368 :
00718 016370 :
00719 016372 :
00720 016374 :
00721 016376 :
00722 016378 :
00723 016380 :
00724 016382 :
00725 016384 :
00726 016386 :
00727 016388 :
00728 016390 :
00729 016392 :
00730 016394 :
00731 016396 :
00732 016398 :
00733 016400 :
00734 016402 :
00735 016404 :
00736 016406 :
00737 016408 :
00738 016410 :
00739 016412 :
00740 016414 :
00741 016416 :
00742 016418 :
00743 016420 :
00744 016422 :
00745 016424 :
00746 016426 :
00747 016428 :
00748 016430 :
00749 016432 :
00750 016434 :
00751 016436 :
00752 016438 :
00753 016440 :
00754 016442 :
00755 016444 :
00756 016446 :
00757 016448 :
00758 016450 :
00759 016452 :
00760 016454 :
00761 016456 :
00762 016458 :
00763 016460 :
00764 016462 :
00765 016464 :
00766 016466 :
00767 016468 :
00768 016470 :
00769 016472 :
00770 016474 :
00771 016476 :
00772 016478 :
00773 016480 :
00774 016482 :
00775 016484 :
00776 016486 :
00777 016488 :
00778 016490 :
00779 016492 :
00780 016494 :
00781 016496 :
00782 016498 :
00783 016500 :
00784 016502 :
00785 016504 :
00786 016506 :
00787 016508 :
00788 016510 :
00789 016512 :
00790 016514 :
00791 016516 :
00792 016518 :
00793 016520 :
00794 016522 :
00795 016524 :
00796 016526 :
00797 016528 :
00798 016530 :
00799 016532 :
00800 016534 :
00801 016536 :
00802 016538 :
00803 016540 :
00804 016542 :
00805 016544 :
00806 016546 :
00807 016548 :
00808 016550 :
00809 016552 :
00810 016554 :
00811 016556 :
00812 016558 :
00813 016560 :
00814 016562 :
00815 016564 :
00816 016566 :
00817 016568 :
00818 016570 :
00819 016572 :
00820 016574 :
00821 016576 :
00822 016578 :
00823 016580 :
00824 016582 :
00825 016584 :
00826 016586 :
00827 016588 :
00828 016590 :
00829 016592 :
00830 016594 :
00831 016596 :
00832 016598 :
00833 016600 :
00
```

```

3056 015102 000000      WRT.RD: .WORD 0      :ADDRESS IS FORMED HERE
3057 015104 013746 007704      MOV RPADR, -(SP) :PUT THE ADDRESS ON THE STACK
3058 015110 062716 000010      ADD #RPCS2, (SP) :FORM THE ADDRESS OF RPCS2
3059 015114 032736 010000      BIT #BIT12, 2(SP)+ :CHECK THE 'NED' BIT
3070 015120 001023      BNE WRT.R3 :BR IF DRIVE NON-EXISTENT
3071 015122 004037 014636      JSR RD, RD.RP :CHECK FOR PARITY ERROR ON WRITE
3072 015126 000014      RPER1
3073 015130 015170      WRT.R3
3074 015132 032726 000010      BIT #BIT03, (SP)+ :BRANCH IF "PAR=0"
3075 015136 001416      BEQ WRT.R4 :PICKUP THE INDEX
3076 015140 016037 177776 015152      MOV -2(RD), 1$ :READ THE REG.
3077 015146 004037 014636      JSR RD, RD.RP :REG. INDEX
3078 015152 000000      1$: .WORD 0 :RETURN TO THIS ADDRESS ON ERROR
3079 015154 015170      WRT.R3 :REPORT THE PARITY ON WRITE ERROR
3080 015156 104004      ERROR 4 :DECREMENT THE ERROR COUNT
3081 015160 005327      DEC (R0)+ :RETRY COUNTER
3082 015162 000003      WRT.R2: .WORD 3 :TRY AGAIN IF NOT FINISHED
3083 015164 002344      BGE WRT.R1 :CLEAN OFF THE STACK
3084 015166 005726      TST (SP)+ :TAKE THE "PARITY ON WRITE" ERROR EXIT
3085 015170 011000      WRT.R3: MOV (R0), R0 :EXIT
3086 015172 000401      BR WRT.R5 :ADJUST FOR ERROR FREE EXIT
3087 015174 005720      WRT.R4: TST (R0)+
3088 015176 000200      WRT.R5: RTS R0
3089
3090 :ROUTINE TO SAVE THE RH11/RPO4 5 6 REGISTERS AS PER DPB+14
3091 :CALL
3092 :
3093 :      MOV #DPBNJM, R2 :DPB POINTER TO R2
3094 :      JSR PC, SVRH1! :SAVE THE DRIVES REG'S
3095 :
3096 015200 104412      SVRH1: SAVREG :SAVE R0 - R5
3097 015202 005702      TST R2 :QUEUE ENTRY FOR THE DRIVE
3098 015204 001430      BEQ 4$ :BR IF NONE
3099 015206 013704 007704      MOV RPADR, R4
3100 015212 111264 000010      MOV (R2), RPCS2(R4) :SELECT DRIVE
3101 015216 016203 000014      MOV 14(R2), R3 :SET THE ERROR TABLE POINTER
3102 015222 001433      BEQ 6$ :EXIT IF NO ADDRESS
3103 015224 005037 015260      CLR 3$ :COUNTER & POINTER
3104 015230 023727 015250 000022 1$: CMP 3$, #RPOB :REACHED THE BUFFER REGISTER
3105 015236 001006      BNE 2$ :BR IF NOT
3106 015240 032764 000200 000010      BIT #BIT07, RPCS2(R4) :'CR' SET?
3107 015246 001002      BNE 2$ :BR IF SET
3108 015250 005023      CLR (R3)+ :STORE RPOB AS ZEROES
3109 015252 000405      BR 4$ :CONTINUE
3110 015254 004037 014636      2$: JSR RD, RD.RP :READ THE SELECTED REGISTER
3111 015260 000000      3$: .WORD 0 :REGISTER INDEX
3112 015262 015306      5$: :ERROR RETURN ADDRESS
3113 015264 012623      MOV (SP)+, (R3)+ :STORE THE REGISTER CONTENTS
3114 015266 023727 015260 000046 4$: CMP 3$, #RPEC2 :REACHED THE END?
3115 015274 001406      BEQ 6$ :BR IF YES
3116 015276 062737 000002 015260      ADD #2, 3$ :INCREMENT THE REGISTER INDEX
3117 015304 000751      BR 1$ :CONTINUE READING THE REGISTERS
3118 015306 004737 012074      5$: JSR PC, 01$ :PROCESS THE UNCORRECTABLE PARITY ERROR
3119 015312 104412      6$: RESREG :RESTORE R0 - R5

```

```

313 015314 000207      RIS      PC      :RETURN
314      :ROUTINE TO SET THE INTERRUPT WITHOUT GETTING A "TRE"
315      :CALL
316      :      MOV      #DRVNUM,R1      :DRIVE NUMBER TO R1
317      :      JSR      PC.SET.IE      :SET "IE"
318      :      RETURN
319
320      SET.IE: MOV      R4,-(SP)      :SAVE R4
321      :      MOV      RPAOR,R4      :PICKUP ADDRESS OF RPO4
322      :      MOV      R1,RPCS2(R4)   :SELECT DRIVE
323      :      MOV      (R4),-(SP)     :READ RPO4
324      :      SIS      #BIT14,(SP)   :SET THE "TRE" BIT OF THE WORD READ
325      :      SWAB      (SP)         :ADJUST FOR DATA
326      :      MOV      #BIT05,(R4)   :SET "IE"
327      :      BIT      #BIT12,RPCS2(R4) :IS "NED"=1?
328      :      BNE      1$            :YES--CLEAR "TRE"
329      :      TST      (SP)+         :CLEAN OFF THE STACK
330      :      BR       2$
331      :      1$: MOV      (SP)+,R4    :CLEAR "TRE"
332      :      2$: MOV      (SP)+,R4    :RESTORE R4
333      :      RTS      PC            :RETURN TO CALLER
334
335      :QUEUE COUNT
336      JCNT: .BYTE      0            :DRIVE 0
337      :      .BYTE      0            :DRIVE 1
338      :      .BYTE      0            :DRIVE 2
339      :      .BYTE      0            :DRIVE 3
340      :      .BYTE      0            :DRIVE 4
341      :      .BYTE      0            :DRIVE 5
342      :      .BYTE      0            :DRIVE 6
343      :      .BYTE      0            :DRIVE 7
344
345      :QUEUE INPUT POINTERS
346      GINPT: .WORD      QDRV0        :DRIVE 0
347      :      .WORD      QDRV1        :DRIVE 1
348      :      .WORD      QDRV2        :DRIVE 2
349      :      .WORD      QDRV3        :DRIVE 3
350      :      .WORD      QDRV4        :DRIVE 4
351      :      .WORD      QDRV5        :DRIVE 5
352      :      .WORD      QDRV6        :DRIVE 6
353      :      .WORD      QDRV7        :DRIVE 7
354
355      :QUEUE OUTPUT POINTERS
356      QOUTPT: .WORD      QDRV0       :DRIVE 0
357      :      .WORD      QDRV1       :DRIVE 1
358      :      .WORD      QDRV2       :DRIVE 2
359      :      .WORD      QDRV3       :DRIVE 3
360      :      .WORD      QDRV4       :DRIVE 4
361      :      .WORD      QDRV5       :DRIVE 5
362      :      .WORD      QDRV6       :DRIVE 6
363      :      .WORD      QDRV7       :DRIVE 7

```

```

3174
3175 015440 015462          QSTART: .WORD QDRV0          ;DRIVE 0 START ADDRESS
3176 015442 015502          QSTOP:  .WORD QDRV1          ;DRIVE 0 STOP ADDRESS & DRIVE 1 START ADDRESS
3177 015444 015522          .WORD QDRV2          ;STOP DRIVE 1--START DRIVE 2
3178 015446 015542          .WORD QDRV3          ;STOP DRIVE 2--START DRIVE 3
3179 015450 015562          .WORD QDRV4          ;STOP DRIVE 3--START DRIVE 4
3180 015452 015602          .WORD QDRV5          ;STOP DRIVE 4--START DRIVE 5
3181 015454 015622          .WORD QDRV6          ;STOP DRIVE 5--START DRIVE 6
3182 015456 015642          .WORD QDRV7          ;STOP DRIVE 6--START DRIVE 7
3183 015460 015662          .WORD QTERM          ;STOP DRIVE 7
3184
3185          ;DRIVE REQUEST QUEUES
3186
3187 015462 000010          QDRV0:  .BLKW 10
3188 015502 000010          QDRV1:  .BLKW 10
3189 015522 000010          QDRV2:  .BLKW 10
3190 015542 000010          QDRV3:  .BLKW 10
3191 015562 000010          QDRV4:  .BLKW 10
3192 015602 000010          QDRV5:  .BLKW 10
3193 015622 000010          QDRV6:  .BLKW 10
3194 015642 000010          QDRV7:  .BLKW 10
3195          QTERM=.
3196
3197          ;ROUTINE TO CLEAR ALL OF THE REQUEST QUEUES
3198
3199          ;CALL
3200          ;      JSR      PC,CLRQLE
3201
3202 015662 104412          CLRQLE: SAVREG          ;SAVE R0 - R5
3203 015664 012702 015370          MOV      #QCNT,R2          ;ZERO THE QUEUE COUNTS
3204 015670 005022          CLR      (R2)+          ;DRIVES 0 & 1
3205 015672 005022          CLR      (R2)+          ;DRIVES 2 & 3
3206 015674 005022          CLR      (R2)+          ;DRIVES 4 & 5
3207 015676 005022          CLR      (R2)+          ;DRIVES 6 & 7
3208 015700 012703 000010          MOV      #8,R3          ;MOVE THE STARTING
3209 015704 012701 015440          MOV      #QSTART,R1          ;ADDRESS OF THE QUEUE INTO
3210 015710 012122          1$: MOV      (R1)+,(R2)+          ;THE QUEUE INPUT POINTER
3211 015712 005303          DEC      R3
3212 015714 001375          BNE      1$
3213 015716 012703 000010          MOV      #8,R3          ;MOVE THE STARTING ADDRESS
3214 015722 012701 015440          MOV      #QSTART,R1          ;OF THE QUEUE INTO THE
3215 015726 012122          2$: MOV      (R1)+,(R2)+          ;QUEUE OUTPUT POINTER
3216 015730 005303          DEC      R3
3217 015732 001375          BNE      2$
3218 015734 104413          RESREG
3219 015736 009207          RTS      PC          ;RESTORE R0 - R5
3220
3221          ;EMPTY THE QUEUE SPECIFIED BY R1
3222
3223          ;CALL
3224          ;      MOV      DRVNUM,R1          ;DRIVE NUMBER TO R1
3225          ;      JSR      PC,EMPTYQ
3226
3227 015740 105061 015370          EMPTYQ: CLRB      QCNT(R1)          ;CLEAR NUMBER OF ITEMS IN QUEUE

```

```

3228 015744 006301      ASL      R1
3229 015746 016161 015400 015420  MOV    QINPT(R1),QOUTPT(R1) ;SET OUTPUT QUEUE POINTER=INPUT POINTER
3230 015754 006201      ASR      R1
3231 015756 000207      RTS      PC
3232
3233      ;ROUTINE TO PUT A REQUEST IN QUEUE
3234      ;CALL
3235      :      MOV      #DRVNUM,R1      ;DRIVE NUMBER
3236      :      MOV      #DPB,R2      ;ADDRESS OF PARAMETER BLOCK
3237      :      JSR      RO,DRVQUE      ;GO PUT REQUEST IN QUEUE
3238      :      RETURN1      ;RETURN HERE IF QUEUE IS FULL
3239      :      RETURN2      ;RETURN HERE IF REQUEST IS IN QUEUE
3240
3241
3242 015760 122761 000010 015370 DRVQUE: CMPB    #1C,QCNT(R1)      ;IS QUEUE FULL?
3243 015766 001421      BEQ      2$      ;BR IF YES-TAKE RETURN1
3244 015770 105261 015370      INCB    QCNT(R1)      ;INCREMENT QUEUE COUNT
3245 015774 006301      ASL      R1
3246 015776 010271 015400      MOV      R2,QINPT(R1)      ;PUT THIS REQUEST IN QUEUE
3247 016002 062761 000002 015400      ADD      #2,QINPT(R1)      ;UPDATE THE QUEUE POINTER
3248 016010 026161 015400 015442      CMP      QINPT(R1),QSTOP(R1) ;TIME TO RESET THE POINTER
3249 016016 001003      BNE      1$      ;BRANCH IF NO
3250 016020 016161 015440 015400      MOV      QSTART(R1),QINPT(R1) ;YES--RESET POINTER
3251 016026 006201      1$:      ASR      R1
3252 016030 005720      TST      (R0)+      ;TAKE RETURN 2
3253 016032 000200      2$:      RTS      R0      ;RETURN TO USER
3254
3255      ;ROUTINE TO GET THE "DPB" ADDRESS OF NEXT REQUEST IN QUEUE
3256      ;CALL
3257      :      MOV      #DRVNUM,R1      ;DRIVE NUMBER TO R1
3258      :      JSR      PC,GETREQ      ;GO GET THE REQUEST
3259      :      RETURN      ;R2="DPB" ADDRESS OF THE REQUEST
3260      :      ;R2=0 IF NO REQUEST IN QUEUE
3261
3262
3263 016034 005002      GETREQ: CLR      R2
3264 016036 105761 015370      TSTB    QCNT(R1)      ;IS THERE ANY REQUEST IN QUEUE?
3265 016042 001404      BEQ      2$      ;NO---BRANCH
3266 016044 006301      1$:      ASL      R1
3267 016046 017102 015420      MOV      QOUTPT(R1),R2      ;PICKUP "DPB" POINTER FOR THIS DRIVE
3268 016052 006201      ASR      R1
3269 016054 000207      RTS      PC      ;RETURN TO USER
3270
3271      ;ROUTINE TO "POP" THE REQUEST FROM QUEUE
3272      ;CALL
3273      :      MOV      #DRVNUM,R1      ;DRIVE NUMBER TO R1
3274      :      JSR      PC,POPQUE      ;CALL TO REMOVE REQUEST
3275      :      RETURN      ;R2=ADDRESS OF DPB REMOVED
3276
3277
3278 016056 105361 015370      POPQUE: DECB    QCNT(R1)      ;DECREMENT QUEUE COUNT
3279 016062 006301      ASL      R1
3280 016064 017102 015420      MOV      QOUTPT(R1),R2      ;GET THE "DPB" POINTER
3281 016070 062761 000002 015420      ADD      #2,QOUTPT(R1)      ;UPDATE THE QUEUE POINTER

```

```

3292 016076 026161 015420 015442      CMP      QOUTPT(R1),QSTOP(R1) ;TIME TO RESET THE POINTER?
3293 016104 001003      BNE      15 ;NO--BRANCH TO EXIT
3294 016106 016161 015440 015420      MOV      QSTART(R1),QOUTPT(R1) ;YES--RESET THE POINTER
3295 016114 006201      ASR      R1
3296 016116 000207      RTS      PC ;RETURN TO USER
3297
3298
3299
3300
3301
3302
3303
3304
3305
3306
3307
3308
3309
3310
3311
3312
3313
3314
3315
3316
3317
3318
3319
3320
3321
3322
3323
3324
3325
3326
3327
3328
3329
3330
3331
3332
3333
3334
3335

```

15:

;*****

.SBTTL DATA PARAMETER BLOCK

;*****

DPB:

.BYTE 0 ;DRIVE NUMBER
 .BYTE 0 ;OFFSET VALUE OR FMT22, ECI, AND HCI
 .BYTE 0 ;COMMAND
 .BYTE 0 ;PSEL AND A17 AND A16
 .WORD 0 ;WORD COUNT (MUST BE NEG)
 .WORD INREG ;BUFFER ADDRESS OR
 ;REGISTER TABLE POINTER
 .BYTE 0 ;SECTOR ADDRESS OR
 .BYTE 0 ;FIRST REGISTER INDEX
 .WORD 0 ;TRACK ADDRESS OR
 .WORD REG ;LAST REGISTER INDEX
 ;CYLINDER ADDRESS
 ;ERROR TABLE POINTER
 ;POINTS TO THE FIRST OF TWENTY
 ;LOCATIONS WHERE THE DRIVER IS
 ;TO STORE THE RH11/RPO4 REGISTERS
 ;ON AN ERROR. IF ZERO, REGISTERS
 ;ARE NOT SAVED.
 .WORD 0 ;STATUS/ERROR INDICATOR
 ;BIT15 = 1 => ERROR OCCURRED
 ;BIT07 = 1 => DONE
 ;BIT 14 - BIT10 AND BIT06 - BIT03
 ;INDICATE TYPE OF ERROR

REG:

.WORD 0 ;STORE RPO4 REGISTERS HERE
 .WORD 0 ;RPWC
 .WORD 0 ;RPBA
 .WORD 0 ;RPDA
 .WORD 0 ;RPCS2
 .WORD 0 ;RPDS1
 .WORD 0 ;RPER1
 .WORD 0 ;RPAS
 .WORD 0 ;RPLA
 .WORD 0 ;RPDB
 .WORD 0 ;RPMR
 .WORD 0 ;RPDT
 .WORD 0 ;RPSN
 .WORD 0 ;RPOF
 .WORD 0 ;RPCA
 .WORD 0 ;RPCC

3336	016200	000000	.WORD	0	;RPER2
3337	016202	000000	.WORD	0	;RPER3
3338	016204	000000	.WORD	0	;RPEC1
3339	016206	000000	.WORD	0	;RPEC2
3340					
3341		000001	CODE=1		;DPB INDEX EQUATES
3342		000002	COMND=2		
3343		000006	BUF=6		
3344		000010	SEC=10		
3345		000011	TRK=11		
3346		000012	CYL=12		
3347		000016	STATUS=16		
3348					
3349					

;;*****

.SBTTL HEAD CODE TABLE

;;*****

3355	016210	000000	HEAD1: .WORD	0	;HEAD ADDRESSES FOR CYLINDERS 245 & 495
3356	016212	000001	.WORD	1	
3357	016214	000002	.WORD	2	
3358	016216	000003	.WORD	3	
3359	016220	000004	.WORD	4	
3360	016222	000005	.WORD	5	
3361	016224	000006	.WORD	6	
3362	016226	000007	.WORD	7	
3363	016230	000010	.WORD	10	
3364	016232	000011	.WORD	11	
3365	016234	000012	.WORD	12	
3366	016236	000013	.WORD	13	
3367	016240	000014	.WORD	14	
3368	016242	000015	.WORD	15	
3369	016244	000016	.WORD	16	
3370	016246	000017	.WORD	17	
3371	016250	000020	.WORD	20	
3372	016252	000021	.WORD	21	
3373	016254	000022	.WORD	22	
3374	016256	100000	.WORD	100000	;TABLE TERMINATOR
3375					
3376	016260	000000	HEAD6: .WORD	0	;HEAD ADDRESSES FOR CYLS 800 & 009
3377	016262	000001	.WORD	1	
3378	016264	000012	.WORD	10.	
3379	016266	000021	.WORD	17.	
3380	016270	000022	.WORD	18.	
3381	016272	100000	.WORD	100000	
3382	016274	000000	.WORD	0	
3383	016276	000001	.WORD	1	
3384	016300	000012	.WORD	10.	
3385	016302	000021	.WORD	17.	
3386	016304	000022	.WORD	18.	
3387	016306	100000	.WORD	100000	;TERMINATOR
3388					
3389	016310	000000	HEAD45: .WORD	0	;HEAD ADDRESSES FOR CYLINDERS 400 & 4

16:34 PAGE 66

```

; TERMINATOR

```

.....

.SBTTL OFFSET CODE TABLES

.....

OFTBL6: :OFFSET TABLE FOR CYLINDER 496

CFTBL6:

.BYTE	30
.BYTE	27
.BYTE	26
.BYTE	25
.BYTE	24
.BYTE	23
.BYTE	22
.BYTE	21
.BYTE	20
.BYTE	17
.BYTE	16
.BYTE	15
.BYTE	14
.BYTE	13
.BYTE	12
.BYTE	11
.BYTE	10
.BYTE	7
.BYTE	6
.BYTE	5
.BYTE	4
.BYTE	3
.BYTE	2
.BYTE	1
.BYTE	0
.BYTE	201
.BYTE	202
.BYTE	203
.BYTE	204
.BYTE	205
.BYTE	206
.BYTE	207
.BYTE	210
.BYTE	211
.BYTE	212
.BYTE	213
.BYTE	214
.BYTE	215
.BYTE	216
.BYTE	217
.BYTE	222

REC 4 5 6 HEAD ALIGNMENT PROGRAM MAY 11 27.655
OFFSET CODE TABLES

[illegible][illegible][illegible]

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.

1. (א) (ב) (ג) (ד) (ה) (ו) (ז) (ח) (ט) (י) (יא) (יב) (יג) (יד) (טו) (טז) (יז) (יח) (יט) (כ) (כא) (כב) (כג) (כד) (כה) (כו) (כז) (כח) (כט) (ל) (לא) (לב) (לג) (לד) (לה) (לו) (לז) (לח) (לט) (מ) (מא) (מב) (מג) (מד) (מה) (מו) (מז) (מח) (מט) (נ) (נא) (נב) (נג) (נד) (נה) (נו) (נז) (נח) (נט) (ס) (סא) (סב) (סג) (סד) (סה) (סו) (סז) (סח) (סט) (ע) (עא) (עב) (עג) (עד) (עה) (עו) (עז) (עח) (עט) (פ) (פא) (פב) (פג) (פד) (פה) (פו) (פז) (פח) (פט) (צ) (ца) (цб) (цг) (цд) (це) (цу) (цз) (цח) (цט) (ק) (קא) (קב) (קג) (קד) (קה) (קו) (קז) (קח) (קט) (ר) (רא) (רב) (רג) (רד) (רה) (רו) (רז) (רח) (רט) (ס) (סא) (סב) (סג) (סד) (סה) (סו) (סז) (סח) (סט) (ת) (תא) (תב) (תג) (תד) (תה) (תו) (תז) (תח) (תט) (י) (יא) (יב) (יג) (יד) (יט) (כ) (כא) (כב) (כג) (כד) (כה) (כו) (כז) (כח) (כט) (ל) (לא) (לב) (לג) (לד) (לה) (לו) (לז) (לח) (לט) (מ) (מא) (מב) (מג) (מד) (מה) (מו) (מז) (מח) (מט) (נ) (נא) (נב) (נג) (נד) (נה) (נו) (נז) (נח) (נט) (ס) (סא) (סב) (סג) (סד) (סה) (סו) (סז) (סח) (סט) (ע) (עא) (עב) (עג) (עד) (עה) (עו) (עז) (עח) (עט) (פ) (פא) (פב) (פג) (פד) (פה) (פו) (פז) (פח) (פט) (צ) (ца) (цб) (цг) (цд) (це) (цу) (цз) (цח) (цט) (ק) (קא) (קב) (קג) (קד) (קה) (קו) (קז) (קח) (קט) (ר) (רא) (רב) (רג) (רד) (רה) (רו) (רז) (רח) (רט) (ס) (סא) (סב) (סג) (סד) (סה) (סו) (סז) (סח) (סט) (ת) (תא) (תב) (תג) (תד) (תה) (תו) (תז) (תח) (תט)

.SBTTL MESSAGES

TITLE: .ASCII (CR)(LF) MAINDEC-11-DZRJC-A (CR)(LF)

.ASCII BRP04 5 6 HEAD ALIGNMENT VERIFICATION PROGRAM (CR)(LF)

DOU: .ASCII DO NOT SELECT DRIVE UNTIL HEAD ALIGNMENT TEST BOX HAS BEEN CONNECTED

.ASCII NOTE: WITH THE ALIGNMENT PACK MOUNTED, CONSTANT INDEX ERRORS MAY OCCUR

.ASCII THESE INDEX ERRORS ARE CAUSED BY THE ALIGNMENT PACK AND MAY (CR)(LF)

504 5 6 HEAD ALIGNMENT PROGRAM
 MESSAGES

 27-APR-76 16:34 PAGE 70
 27-APR-76 16:34 PAGE 70
 27-APR-76 16:34 PAGE 70

.ASCIZ NOT OCCUR WITH SOME ALIGNMENT PACKS. TO INHIBIT THE ALIGNMENT PACKING

.ASCIZ RELATED INDEX ERROR. GROUND THE FOLLOWING PINS: (CR) (LF)

.ASCIZ RP04: PIN A1A0503 (FUNCTION '+INDEXER' (CR) (LF))

.ASCIZ 2 RP05 6: PIN D04R19 (FUNCTION '-INDEX' (CR) (LF) (LF))

NO DRV: .ASCIZ (CR) (LF) DRIVE NOT AVAILABLE (CR) (LF) (LF)

ENTERD: .ASCIZ (CR) (LF) ENTER DRIVE NUMBER:

WLOCK: .ASCIZ (CR) (LF) DRIVE NOT WRITE PROTECTED (CR) (LF) (LF)

52-11-25800-A, REC'D S 6 HEAD ALIGNMENT PROGRAM MACY!! 27(655) 27-APR-76 16:34 PAGE 72
25800.009 MESSAGES

NO-11-DEPTO-A. REC4 5 6
DEPTO.009 MESSAGES

| | | | | |
|--------|--------|--------|--------|--|
| 020174 | 051105 | 032040 | 033071 | |
| 020202 | 051440 | 046105 | 041505 | |
| 020210 | 042524 | 020104 | 054502 | |
| 020216 | 042040 | 043105 | 052501 | |
| 020224 | 052114 | 005015 | 000012 | |
| 020232 | 044504 | 045523 | 044440 | ALN245: .ASCIZ DISK IS POSITIONED AT CYLINDER 245 (CR) (LF) (LF) |
| 020240 | 020123 | 047520 | 044523 | |
| 020246 | 044524 | 047117 | 042105 | |
| 020254 | 040440 | 020124 | 054503 | |
| 020262 | 044514 | 042116 | 051105 | |
| 020270 | 031040 | 032464 | 005015 | |
| 020276 | 000012 | | | |
| 020300 | 044504 | 045523 | 044440 | ALN496: .ASCIZ DISK IS POSITIONED AT CYLINDER 496 (CR) (LF) (LF) |
| 020306 | 020123 | 047520 | 044523 | |
| 020314 | 044524 | 047117 | 042105 | |
| 020322 | 040440 | 020124 | 054503 | |
| 020330 | 044514 | 042116 | 051105 | |
| 020336 | 032040 | 033071 | 005015 | |
| 020344 | 000012 | | | |
| 020346 | 005015 | 047105 | 042524 | ENTHD: .ASCIZ (CR) (LF) ENTER HEAD: / |
| 020354 | 020122 | 042510 | 042101 | |
| 020362 | 020072 | 000 | | |
| 020365 | 015 | 042412 | 052116 | ENTCYL: .ASCIZ (CR) (LF) ENTER CYLINDER ADDRESS: / |
| 020372 | 051105 | 041440 | 046131 | |
| 020400 | 047111 | 042504 | 020122 | |
| 020406 | 042101 | 051104 | 051505 | |
| 020414 | 035123 | 000040 | | |
| 020420 | 047111 | 040526 | 044514 | BADTRK: .ASCIZ /INVALID HEAD ADDRESS (CR) (LF) |
| 020426 | 020104 | 042510 | 042101 | |
| 020434 | 040440 | 042104 | 042522 | |
| 020442 | 051523 | 005015 | 000 | |
| 020447 | 111 | 053116 | 046101 | BADCYL: .ASCIZ /INVALID CYLINDER ADDRESS (CR) (LF) |
| 020454 | 042111 | 041440 | 046131 | |
| 020462 | 047111 | 042504 | 020122 | |
| 020470 | 042101 | 051104 | 051505 | |
| 020476 | 006523 | 000012 | | |
| 020502 | 005015 | 042012 | 047117 | ENDMSG: .ASCIZ (CR) (LF) (LF) /DONE (CR) (LF) (LF) |
| 020510 | 006505 | 005012 | 000 | |
| | 020516 | | | .EVEN |
| 020516 | 044122 | 030461 | 044440 | EMI: .ASCIZ /RHI1 INTERRUPT OCCURRED (RPAS=0) / |
| 020524 | 052116 | 051105 | 052522 | |
| 020532 | 052120 | 047440 | 041503 | |
| 020540 | 051123 | 042522 | 020104 | |
| 020546 | 051050 | 040520 | 036523 | |

H07

40-11-22RJC-A. RPO4 5 6 HEAD ALIGNMENT PROGRAM MACY!! 27(655) 27-APR-76 16:34 PAGE 73
 22RJC.009 MESSAGES

SEC 0084

| | | | | | |
|------|--------|--------|--------|--------|--|
| 3768 | 020554 | 024460 | 000 | | |
| 3769 | | | | | |
| 3770 | 020557 | 125 | 042516 | 050130 | EM2: .ASCIZ UNEXPECTED ATTENTION OCCURRED/ |
| 3771 | 020554 | 041505 | 042524 | 020104 | |
| 3772 | 020572 | 052101 | 042524 | 052116 | |
| 3773 | 020600 | 047511 | 020116 | 041517 | |
| 3774 | 020606 | 052503 | 051122 | 042105 | |
| 3775 | 020614 | 000 | | | |
| 3776 | | | | | |
| 3777 | 020615 | 115 | 051501 | 041123 | EM3: .ASCIZ MASSBUS PARITY ERROR (MCPE=1)/ |
| 3778 | 020622 | 051525 | 050040 | 051101 | |
| 3779 | 020630 | 052111 | 020131 | 051105 | |
| 3780 | 020636 | 047522 | 020122 | 046450 | |
| 3781 | 020644 | 050103 | 036505 | 024451 | |
| 3782 | 020652 | 000 | | | |
| 3783 | | | | | |
| 3784 | 020653 | 115 | 051501 | 041123 | EM4: .ASCIZ MASSBUS PARITY ERROR (PAR=1)/ |
| 3785 | 020660 | 051525 | 050040 | 051101 | |
| 3786 | 020666 | 052111 | 020131 | 051105 | |
| 3787 | 020674 | 047522 | 020122 | 050050 | |
| 3788 | 020702 | 051101 | 030475 | 000051 | |
| 3789 | | | | | |
| 3790 | 020710 | 042101 | 051104 | 051505 | EM5: .ASCIZ ADDRESS PLUG CHANGE BIT SET/ |
| 3791 | 020716 | 020123 | 046120 | 043525 | |
| 3792 | 020724 | 041440 | 040510 | 043516 | |
| 3793 | 020732 | 020105 | 044502 | 020124 | |
| 3794 | 020740 | 042523 | 000124 | | |
| 3795 | | | | | |
| 3796 | 020744 | 044122 | 030461 | 042040 | EM6: .ASCIZ RM11 DIDN'T RESPOND TO ADDRESSING/ |
| 3797 | 020752 | 042111 | 023516 | 020124 | |
| 3798 | 020760 | 042522 | 050123 | 047117 | |
| 3799 | 020766 | 020104 | 047524 | 040440 | |
| 3800 | 020774 | 042104 | 042522 | 051523 | |
| 3801 | 021002 | 047111 | 000107 | | |
| 3802 | | | | | |
| 3803 | 021006 | 051104 | 053111 | 020105 | EM10: .ASCIZ DRIVE OR DATA ERROR |
| 3804 | 021014 | 051117 | 042040 | 052101 | |
| 3805 | 021022 | 020101 | 051105 | 047522 | |
| 3806 | 021030 | 000122 | | | |
| 3807 | | | | | |
| 3808 | 021032 | 051104 | 053111 | 020105 | EM11: .ASCIZ DRIVE UNSAFE ERROR/ |
| 3809 | 021040 | 047125 | 040523 | 042506 | |
| 3810 | 021048 | 042440 | 051122 | 051117 | |
| 3811 | 021054 | 000 | | | |
| 3812 | | | | | |
| 3813 | 021055 | 110 | 040505 | 020104 | EM12: .ASCIZ HEAD OUT OF ALIGNMENT |
| 3814 | 021062 | 052517 | 020124 | 043117 | |
| 3815 | 021070 | 040440 | 044514 | 047107 | |
| 3816 | 021076 | 042515 | 052116 | 000 | |
| 3817 | | | | | |
| 3818 | 021103 | 110 | 040505 | 020104 | EM13: .ASCIZ HEAD TOO FAR OUT OF ALIGNMENT/ |
| 3819 | 021110 | 047524 | 020117 | 040506 | |
| 3820 | 021116 | 020122 | 052517 | 020124 | |
| 3821 | 021124 | 043117 | 040440 | 044514 | |

| | | | | | | | | | |
|------|--------|--------|--------|--------|-------|--------|---|--|--|
| 3822 | 021132 | 047107 | 042515 | 052116 | | | | | |
| 3823 | 021140 | 000 | | | | | | | |
| 3824 | | | | | | | | | |
| 3825 | 021141 | 123 | 043117 | 053524 | EM14: | .ASCIZ | SOFTWARE TIMEOUT | | |
| 3826 | 021146 | 051101 | 020105 | 044524 | | | | | |
| 3827 | 021154 | 042515 | 052517 | 000124 | | | | | |
| 3828 | | | | | | | | | |
| 3829 | 021162 | 047125 | 047503 | 051122 | EM15: | .ASCIZ | UNCORRECTABLE MASSBUS PARITY ERROR | | |
| 3830 | 021170 | 041505 | 040524 | 046102 | | | | | |
| 3831 | 021176 | 020105 | 040515 | 051523 | | | | | |
| 3832 | 021204 | 052502 | 020123 | 040520 | | | | | |
| 3833 | 021212 | 044522 | 054524 | 042440 | | | | | |
| 3834 | 021220 | 051122 | 051117 | 000 | | | | | |
| 3835 | | | | | | | | | |
| 3836 | 021225 | 104 | 044522 | 042526 | EM16: | .ASCIZ | DRIVE UNAVAILABLE | | |
| 3837 | 021232 | 052440 | 040516 | 040526 | | | | | |
| 3838 | 021240 | 046111 | 041101 | 042514 | | | | | |
| 3839 | 021246 | 000 | | | | | | | |
| 3840 | | | | | | | | | |
| 3841 | | | | | | | | | |
| 3842 | 021247 | 122 | 040520 | 000123 | DH1: | .ASCIZ | RPAS | | |
| 3843 | | | | | | | | | |
| 3844 | 021254 | 051104 | 053111 | 020105 | DH2: | .ASCIZ | DRIVE RPOS1 RPER1 RPER2 RPER3 RPAS | | |
| 3845 | 021262 | 020040 | 050122 | 051504 | | | | | |
| 3846 | 021270 | 020061 | 020040 | 050122 | | | | | |
| 3847 | 021276 | 051105 | 020061 | 020040 | | | | | |
| 3848 | 021304 | 050122 | 051105 | 020062 | | | | | |
| 3849 | 021312 | 020040 | 050122 | 051105 | | | | | |
| 3850 | 021320 | 020063 | 020040 | 050122 | | | | | |
| 3851 | 021326 | 051501 | 000 | | | | | | |
| 3852 | | | | | | | | | |
| 3853 | 021331 | 104 | 044522 | 042526 | DH3: | .ASCIZ | DRIVE REG ADR DATA | | |
| 3854 | 021336 | 020040 | 051040 | 043505 | | | | | |
| 3855 | 021344 | 040440 | 051104 | 020040 | | | | | |
| 3856 | 021352 | 040504 | 040524 | 000 | | | | | |
| 3857 | | | | | | | | | |
| 3858 | 021357 | 104 | 044522 | 042526 | DH4: | .ASCIZ | DRIVE REG ADR GOOD BAD | | |
| 3859 | 021364 | 020040 | 051040 | 043505 | | | | | |
| 3860 | 021372 | 040440 | 051104 | 020040 | | | | | |
| 3861 | 021400 | 020040 | 047507 | 042117 | | | | | |
| 3862 | 021406 | 020040 | 020040 | 040502 | | | | | |
| 3863 | 021414 | 000104 | | | | | | | |
| 3864 | | | | | | | | | |
| 3865 | 021416 | 044122 | 040440 | 042104 | DH5: | .ASCIZ | RM ADDRESS | | |
| 3866 | 021424 | 042522 | 051523 | 000 | | | | | |
| 3867 | | | | | | | | | |
| 3868 | 021431 | 104 | 044522 | 042526 | DH10: | .ASCIZ | DRIVE RPOS1 RPOS2 RPOS3 RPER1 RPER2 RPER3 | | |
| 3869 | 021436 | 020040 | 051040 | 041520 | | | | | |
| 3870 | 021444 | 030523 | 020040 | 051040 | | | | | |
| 3871 | 021452 | 041520 | 031123 | 020040 | | | | | |
| 3872 | 021460 | 051040 | 042120 | 030523 | | | | | |
| 3873 | 021466 | 020040 | 051040 | 042520 | | | | | |
| 3874 | 021474 | 030522 | 020040 | 051040 | | | | | |
| 3875 | 021502 | 042520 | 031122 | 020040 | | | | | |

| | | | | | | |
|------|--------|--------|--------|--------|-------|--|
| 3976 | 021510 | 051040 | 042520 | 031522 | | |
| 3977 | 021516 | 000 | | | | |
| 3978 | | | | | | |
| 3979 | 021517 | 040 | 020040 | 020040 | DH12: | .ASCII / TRK CENTER<<CR><LF> |
| 3980 | 021524 | 020040 | 020040 | 020040 | | |
| 3981 | 021532 | 020040 | 020040 | 052040 | | |
| 3982 | 021540 | 045522 | 041440 | 047105 | | |
| 3983 | 021546 | 042524 | 006522 | 012 | | |
| 3984 | 021553 | 040 | 020040 | 042510 | | .ASCIZ / HEAD CYL (IN U INCHES)/ |
| 3985 | 021560 | 042101 | 020040 | 020040 | | |
| 3986 | 021566 | 054503 | 020114 | 044450 | | |
| 3987 | 021574 | 020116 | 020125 | 047111 | | |
| 3988 | 021602 | 044103 | 051505 | 000051 | | |
| 3989 | | | | | | |
| 3990 | 021610 | 020040 | 044040 | 040505 | DH13: | .ASCIZ / HEAD CYL/ |
| 3991 | 021616 | 020104 | 020040 | 041440 | | |
| 3992 | 021624 | 046131 | 000 | | | |
| 3993 | | | | | | |
| 3994 | 021627 | 104 | 044522 | 042526 | DH16: | .ASCIZ /DRIVE/ |
| 3995 | 021634 | 000 | | | | |
| 3996 | | | | | | |
| 3997 | | 021636 | | | | .EVEN |
| 3998 | | | | | | |
| 3999 | 021636 | 001170 | 000000 | | DT1: | .WORD ATTN,0 |
| 3900 | | | | | | |
| 3901 | 021642 | 001166 | 007536 | 007540 | DT2: | .WORD DRIVE,RPERRS,RPERRS+2,RPERRS+4,RPERRS+6,ATTN,0 |
| 3902 | 021650 | 007542 | 007544 | 001170 | | |
| 3903 | 021656 | 000000 | | | | |
| 3904 | | | | | | |
| 3905 | 021660 | 001166 | 014662 | 014664 | DT3: | .WORD DRIVE,RD.ADR,RD.WRD,0 |
| 3906 | 021666 | 000000 | | | | |
| 3907 | | | | | | |
| 3908 | 021670 | 001166 | 015102 | 015100 | DT4: | .WORD DRIVE,WRT.AD,WRT.WD,RD.WRD,0 |
| 3909 | 021676 | 014664 | 000000 | | | |
| 3910 | | | | | | |
| 3911 | 021702 | 001264 | 000000 | | DT5: | .WORD SRPADR,0 |
| 3912 | | | | | | |
| 3913 | 021706 | 001164 | 016140 | 016150 | DT10: | .WORD DRVSEL,REG,REG+RPCS2,REG+RPCS1,REG+RPER1,REG+RPER2,REG+RPER3,0 |
| 3914 | 021714 | 016152 | 016154 | 016200 | | |
| 3915 | 021722 | 016202 | 000000 | | | |
| 3916 | | | | | | |
| 3917 | 021726 | 001172 | 016132 | 001204 | DT12: | .WORD \$HEAD,DPB+CYL,PLUS,0 |
| 3918 | 021734 | 000000 | | | | |
| 3919 | | | | | | |
| 3920 | 021736 | 001172 | 016132 | 000000 | DT13: | .WORD \$HEAD,DPB+CYL,0 |
| 3921 | | | | | | |
| 3922 | 021744 | 001164 | 000000 | | DT16: | .WORD DRVSEL,0 |
| 3923 | | | | | | |
| 3924 | | | | | | |
| 3925 | | | | | | |
| 3926 | 021750 | 000 | | | DF1: | .BYTE 0 |
| 3927 | | | | | | |
| 3928 | 021751 | 000 | 000 | 000 | DF2: | .BYTE 0,0,0,0,0,0 |
| 3929 | 021754 | 000 | 000 | 000 | | |

```

3930
3931 021757 000 000 000 DF3: .BYTE 0,0,0
3932
3933 021762 000 000 000 DF4: .BYTE 0,0,0,0
3934 021765 000
3935
3936 021766 000 000 000 DF10: .BYTE 0,0,0,0,0,0,0
3937 021771 000 000 000
3938 021774 000
3939
3940 021775 001 001 001 DF12: .BYTE 1,1,1
3941
3942 022000 001 001 DF13: .BYTE 1,1
3943
3944
3945
3946
3947
3948
3949
3950
3951
3952
3953
3954
3955
3956 022002 005737 001242
3957 022006 001450
3958 022010 005037 001242
3959 022014 012700 001264
3960 022020 104401 022202
3961 022024 012046
3962 022026 104402
3963 022030 104401 022224
3964 022034 104411
3965 022036 012601
3966 022040 004537 022230
3967 022044 022064
3968 022046 022130
3969 022050 022014
3970 022052 022060
3971 022054 022014
3972 022056 022124
3973 022060 010260 177776
3974 022064 104401 022213
3975 022070 012046
3976 022072 104402
3977 022074 104401 022224
3978 022100 104411
3979 022102 012601
3980 022104 004537 022230
3981 022110 022130
3982 022112 022130
3983 022114 022064

```

```

.SBTTL BUSADR - GET BUS ADDRESS AND VECTOR ADDRESS FOR RH11
;THIS ROUTINE IS USED TO INSURE THE BUS ADDRESS
;OF THE RH11 IS SETUP FOR THE PROPER ADDRESS.
;IT WILL ALSO READ THE ADDRESS FROM THE TTY IF
;REQUIRED.
;NOTE: THIS ROUTINE DESTROYS R0-R4
;CALL

```

```

;
;      JSR      PC,BUSADR
;      RETURN
;

```

```

BUSADR: TST      CHGADR      ;INPUT FROM TTY REQUESTED?
        BEQ      7$         ;NO--BRANCH
        CLR      CHGADR      ;YES--CLEAR THE REQUEST FLAG
1$:      MOV      #SRPADR,R0  ;FIRST ADDRESS
        TYPE     ,MRPCS1     ;"RPCS1="
        MOV      (R0)+,-(SP) ;PRESENT RPCS1 ADDRESS
        TYP0C    ,LINSIP     ;TYPE IT
        TYPE     ,LINSIP     ;2 SPACES
        RDLIN    (SP)+,R1     ;GET THE ENTRY
        JSR      R5,CK.NUM    ;ADDRESS OF ASCII TEXT
        3$       ;CHECK THE NUMBER
        7$       ;CARRIAGE RETURN ONLY ENTERED
        1$       ;PERIOD ONLY ENTERED
        2$       ;ILLEGAL INPUT
        4$       ;TERMINATED WITH A CARRIAGE RETURN
        5$       ;TERMINATED WITH A "."
        6$       ;TERMINATED WITH A " "
2$:      MOV      R2,-2(R0)   ;SAVE NEW RPCS1
3$:      TYPE     ,MRHVEC     ;"RHVEC="
        MOV      (R0)+,-(SP) ;PRESENT RH11 VECTOR ADDRESS ON THE STACK
        TYP0C    ,LINSIP     ;TYPE IT
        TYPE     ,LINSIP     ;2 SPACES
        RDLIN    (SP)+,R1     ;READ THE ENTRY
        JSR      R5,CK.NUM    ;ASCII TEXT ADDRESS
        7$       ;CHECK THE NUMBER
        7$       ;CARRIAGE RETURN ONLY ENTERED
        7$       ;PERIOD ONLY ENTERED
        3$       ;ILLEGAL INPUT

```

```

3994 022116 022124      4$      : TERMINATED WITH A CARRIAGE RETURN
3985 022120 022064      3$      : TERMINATED WITH A "."
3986 022122 022124      4$      : TERMINATED WITH A "."
3997 022124 010260 177776 4$: MOV R2,-2(R0) :SAVE INPUT
3998 022130 013701 000004 7$: MOV ERARVEC,R1 :SAVE THE ERROR VECTOR
3989 022134 012737 022170 000004 MOV #8$,ERARVEC :SETUP FOR TRAP
3990 022142 005777 157116 TST @SRPADR :CHECK FOR RH11
3991 022146 010137 000004 MOV R1,ERARVEC :RESTORE ERROR VECTOR
3992 022152 012700 001264 MOV #SRPADR,R0 :FIRST ADDRESS OF NEW PARAMETERS
3993 022156 012701 007704 MOV #RPADR,R1 :FIRST ADDRESS OF WHERE TO PUT THEM
3994 022162 012021 MOV (R0)+,(R1)+ :BUS ADDRESS
3995 022164 012021 MOV (R0)+,(R1)+ :VECTOR ADDRESS
3996 022166 000207 RTS PC :RETURN
3997 022170 010137 000004 S$: MOV R1,ERARVEC :RESTORE ERROR VECTOR
3998 022174 022626 CMP (SP)+,(SP)+ :CLEAN OFF THE STACK
3999 022176 104006 ERROR 6 :REPORT THE ERROR
4000 022200 000705 BR 1$ :ASK FOR BUS ADDRESS
4001
4002 022202 050122 051503 020061 MRPCS1: .ASCIZ @RPCS1 = @
4003 022210 020075 000
4004 022213 122 053110 041505 MRHVEC: .ASCIZ @RHVEC = @
4005 022220 036440 000040
4006 022224 020040 000
4007
4008 022230 .EVEN
4009
4010 .SBTTL CK.NUM - CHECK NUMBER (OCTAL)
4011 ;THIS ROUTINE CHECKS AN ASCIZ STRING FOR LEGAL CHARACTERS
4012 ;AND FORMS AN OCTAL NUMBER IN R2
4013 ;CALL:
4014 : MOV #ADR,R1 :ADDRESS OF ASCIZ STRING
4015 : MOV #NUM,R2 :MAX SIZE OF INPUT NUMBER
4016 : JSR R5,CK.NUM :GO FORM THE NUMBER
4017 : RETURN ADR1 : "CR" ONLY ENTERED -- R2 = 0
4018 : RETURN ADR2 : "PERIOD" ONLY ENTERED -- R2 = 0
4019 : RETURN ADR3 : ILLEGAL CHARACTER IN THE INPUT STRING
4020 : RETURN ADR4 : "CR" ENTERED -- R2 = NUMBER
4021 : RETURN ADR5 : "COMMA" -- R2 = NUMBER
4022 : RETURN ADR6 : "PERIOD" -- R2 = NUMBER
4023
4024 022230 010446 CK.NUM: MOV R4,-(SP) :SAVE R4
4025 022232 010346 MOV R3,-(SP) :SAVE R3
4026 022234 010246 MOV R2,-(SP) :SAVE R2
4027 022236 005004 CLR R4 :RETURN POINTER
4028 022240 005003 CLR R3 :START NUMBER AT ZERO
4029 022242 005002 CLR R2 :STORE RESULT
4030 022244 004537 022444 JSR R5,CK.CHR :CHECK ONE CHARACTER
4031 022250 022346 6$ : ILLEGAL CHARACTER
4032 022252 022352 8$ : CARRIAGE RETURN
4033 022254 022346 6$ : "."
4034 022256 022350 7$ : "."
4035 022260 022264 1$ : DIGIT 0-7
4036 022262 022346 6$ : DIGIT 8-9
4037 022264 062705 000004 1$: ADD #4,R5 :INCREMENT RETURN PAST "CR" AND "PERIOD" ONLY RETURNS

```

```

4038 022270 006303      2$: ASL      R3      ;FOR THE OCTAL NUMBER IN R3
4039 022272 103425      BCS      6$      ;DON'T LET IT GET TO BIG
4040 022274 006303      ASL      R3
4041 022276 103423      BCS      6$
4042 022300 006303      ASL      R3
4043 022302 103421      BCS      6$
4044 022304 060203      ADD      R2,R3
4045 022306 004537      JSR      R5,CK.CHR      ;CHECK ONE CHARACTER
4046 022312 022352      6$      ;ILLEGAL CHARACTER
4047 022314 022336      5$      ;CARRIAGE RETURN
4048 022316 022334      4$      ;" "
4049 022320 022326      3$      ;"."
4050 022322 022270      2$      ;DIGIT 0-7
4051 022324 022352      8$      ;DIGIT 8-9
4052 022326 105711      3$: TSTB      (R1)      ;DOES A "CR" FOLLOW THE "PERIOD"
4053 022330 001010      BNE      8$      ;BR IF NOT
4054 022332 005724      TST      (R4)+      ;INCREMENT THE RETURN
4055 022334 005724      4$: TST      (R4)+      ;INCREMENT THE RETURN INDEX
4056 022336 005724      5$: TST      (R4)+      ;INCREMENT THE RETURN INDEX
4057 022340 020316      CMP      R3,(SP)      ;INPUT VALUE TOO LARGE ?
4058 022342 101003      BHI      8$      ;BR IF IT IS
4059 022344 000401      BR      7$      ;BR IF NOT
4060 022346 005725      6$: TST      (R5)+      ;INCREMENT THE RETURN ADDRESS
4061 022350 005725      7$: TST      (R5)+      ;INCREMENT THE RETURN ADDRESS
4062 022352 060405      8$: ADD      R4,R5      ;SETUP FOR PROPER RETURN
4063 022354 010302      MOV      R3,R2      ;LOAD ENTERED VALUE
4064 022356 005726      TST      (SP)+      ;CLEAN OFF THE STACK
4065 022360 012603      MOV      (SP)+,R3      ;RESTORE R3
4066 022362 012604      MOV      (SP)+,R4      ;RESTORE R4
4067 022364 011505      MOV      (R5),R5      ;GET RETURN ADDRESS
4068 022366 000205      RTS      R5      ;RETURN
4069
4070
4071
4072      ;THIS ROUTINE IS USED TO CHECK IF AN
4073      ;ASCII CHARACTER IS A DIGIT BETWEEN 0 AND 7.
4074      ;CALL
4075      ;      MOV      #ADR,R1      ;ADDRESS OF ASCII CHARACTER
4076      ;      JSR      R5,CK.OCT      ;CHECK THE CHARACTER
4077      ;      RETURN1      ;CHARACTER IS NOT BETWEEN 0-7
4078      ;      RETURN2      ;CHARACTER IS IN R2 AS A
4079      ;      ;      ;OCTAL DIGIT
4080
4081 022370 121127 000060      CK.OCT: CMPB      (R1),#'0      ;LESS THAN ZERO?
4082 022374 103407      BLO      1$      ;YES -- BRANCH
4083 022376 121127 000067      CMPB      (R1),#'7      ;GREATER THAN SEVEN?
4084 022402 101004      BHI      1$      ;YES -- BRANCH
4085 022404 111102      MOVB      (R1),R2      ;GET THE CHARACTER
4086 022406 042702 177770      BIC      #1C7,R2      ;STRIP AWAY THE ASCII
4087 022412 005725      TST      (R5)+      ;ADJUST FOR RETURN
4088 022414 000205      1$: RTS      R5      ;RETURN
4089
4090      ;THIS ROUTINE IS USED TO CHECK AN ASCII CHARACTER
4091      ;AND DETERMINE IF IT IS A DIGIT BETWEEN 0 AND 9.

```

```

4092      :CALL
4093      :      MOV      #ADR,R1      ;ADDRESS OF ASCII CHARACTER
4094      :      JSR      R5,CK.DEC    ;CHECK THE CHARACTER
4095      :      RETURN1   ;NOT BETWEEN 0 AND 9
4096      :      RETJRN2  ;BETWEEN 0 AND 9
4097      :      :          ;R2 = DIGIT
4098
4099      022416 121127 000060      CK.DEC: CMPB      (R1),#'0      ;LESS THAN ZERO?
4100      022422 103407              BLO          1$          ;YES -- BRANCH
4101      022424 121127 000071      CMPB      (R1),#'9      ;GREATER THAN NINE?
4102      022430 101004              BHI          1$          ;YES -- BRANCH
4103      022432 111102              MOVB      (R1),R2      ;GET THE CHARACTER
4104      022434 042702 000060      BIC          #'0,R2      ;STRIP AWAY THE ASCII
4105      022440 005725              TST       (R5)+        ;ADJUST FOR RETURN
4106      022442 000205      1$:      RTS          R5          ;RETURN
4107
4108      :THIS ROUTINE WILL CHECK AN ASCII CHARACTER TO
4109      :DETERMINE WHAT IT IS.
4110      :CALL
4111      :      MOV      #ADR,R1      ;ADDRESS OF ASCII CHARACTER
4112      :      JSR      R5,CK.CHR    ;CHECK CHARACTER
4113      :      RETURN   ADR1        ;UNKNOWN CHARACTER
4114      :      RETURN   ADR2        ;CARRIAGE RETURN * (R1)=ADR+1
4115      :      RETURN   ADR3        ;COMMA * (R1)=ADR+1
4116      :      RETURN   ADR4        ;PERIOD * (R1)=ADR+1
4117      :      RETURN   ADR5        ;DIGIT BETWEEN 0 AND 7.
4118      :      RETURN   ADR6        ;DIGIT BETWEEN 8 AND 9.
4119      :      :          ;R2 = DIGIT * (R1)=ADR+1
4120
4121      022444 105711      CK.CHR: TSTB      (R1)          ;"CARRIAGE RETURN"?
4122      022446 001417              BEQ          3$          ;YES -- BRANCH
4123      022450 121127 000054      CMPB      (R1),#'.      ;"COMMA"?
4124      022454 001413              BEQ          2$          ;YES -- BRANCH
4125      022456 121127 000056      CMPB      (R1),#'.      ;"PERIOD"?
4126      022462 001407              BEQ          1$          ;YES -- BRANCH
4127      022464 004537 022416      JSR      R5,CK.DEC    ;"DIGIT"?
4128      022470 000410              BR          4$          ;NO -- BRANCH
4129      022472 004537 022370      JSR      R5,CK.OCT    ;OCTAL ?
4130      022476 005725              TST       (R5)+        ;DIGIT BETWEEN 8-9
4131      022500 005725              TST       (R5)+        ;DIGIT BETWEEN 0-7
4132      022502 005725      1$:      TST       (R5)+        ;PERIOD
4133      022504 005725      2$:      TST       (R5)+        ;COMMA
4134      022506 005725      3$:      TST       (R5)+        ;CARRIAGE RETURN
4135      022510 005201              INC          R1          ;MOVE POINTER TO NEXT CHARACTER
4136      022512 011505      4$:      MOV       (R5),R2      ;UNKNOWN CHARACTER
4137      022514 000205      RTS          R5          ;RETURN
4138
4139      :
4140      :
4141      :
4142      :
4143      :
4144      :
4145      :
4146      :
4147      :
4148      :
4149      :
4150      :
4151      :
4152      :
4153      :
4154      :
4155      :
4156      :
4157      :
4158      :
4159      :
4160      :
4161      :
4162      :
4163      :
4164      :
4165      :
4166      :
4167      :
4168      :
4169      :
4170      :
4171      :
4172      :
4173      :
4174      :
4175      :
4176      :
4177      :
4178      :
4179      :
4180      :
4181      :
4182      :
4183      :
4184      :
4185      :
4186      :
4187      :
4188      :
4189      :
4190      :
4191      :
4192      :
4193      :
4194      :
4195      :
4196      :
4197      :
4198      :
4199      :
4200      :
4201      :
4202      :
4203      :
4204      :
4205      :
4206      :
4207      :
4208      :
4209      :
4210      :
4211      :
4212      :
4213      :
4214      :
4215      :
4216      :
4217      :
4218      :
4219      :
4220      :
4221      :
4222      :
4223      :
4224      :
4225      :
4226      :
4227      :
4228      :
4229      :
4230      :
4231      :
4232      :
4233      :
4234      :
4235      :
4236      :
4237      :
4238      :
4239      :
4240      :
4241      :
4242      :
4243      :
4244      :
4245      :
4246      :
4247      :
4248      :
4249      :
4250      :
4251      :
4252      :
4253      :
4254      :
4255      :
4256      :
4257      :
4258      :
4259      :
4260      :
4261      :
4262      :
4263      :
4264      :
4265      :
4266      :
4267      :
4268      :
4269      :
4270      :
4271      :
4272      :
4273      :
4274      :
4275      :
4276      :
4277      :
4278      :
4279      :
4280      :
4281      :
4282      :
4283      :
4284      :
4285      :
4286      :
4287      :
4288      :
4289      :
4290      :
4291      :
4292      :
4293      :
4294      :
4295      :
4296      :
4297      :
4298      :
4299      :
4300      :
4301      :
4302      :
4303      :
4304      :
4305      :
4306      :
4307      :
4308      :
4309      :
4310      :
4311      :
4312      :
4313      :
4314      :
4315      :
4316      :
4317      :
4318      :
4319      :
4320      :
4321      :
4322      :
4323      :
4324      :
4325      :
4326      :
4327      :
4328      :
4329      :
4330      :
4331      :
4332      :
4333      :
4334      :
4335      :
4336      :
4337      :
4338      :
4339      :
4340      :
4341      :
4342      :
4343      :
4344      :
4345      :
4346      :
4347      :
4348      :
4349      :
4350      :
4351      :
4352      :
4353      :
4354      :
4355      :
4356      :
4357      :
4358      :
4359      :
4360      :
4361      :
4362      :
4363      :
4364      :
4365      :
4366      :
4367      :
4368      :
4369      :
4370      :
4371      :
4372      :
4373      :
4374      :
4375      :
4376      :
4377      :
4378      :
4379      :
4380      :
4381      :
4382      :
4383      :
4384      :
4385      :
4386      :
4387      :
4388      :
4389      :
4390      :
4391      :
4392      :
4393      :
4394      :
4395      :
4396      :
4397      :
4398      :
4399      :
4400      :
4401      :
4402      :
4403      :
4404      :
4405      :
4406      :
4407      :
4408      :
4409      :
4410      :
4411      :
4412      :
4413      :
4414      :
4415      :
4416      :
4417      :
4418      :
4419      :
4420      :
4421      :
4422      :
4423      :
4424      :
4425      :
4426      :
4427      :
4428      :
4429      :
4430      :
4431      :
4432      :
4433      :
4434      :
4435      :
4436      :
4437      :
4438      :
4439      :
4440      :
4441      :
4442      :
4443      :
4444      :
4445      :
4446      :
4447      :
4448      :
4449      :
4450      :
4451      :
4452      :
4453      :
4454      :
4455      :
4456      :
4457      :
4458      :
4459      :
4460      :
4461      :
4462      :
4463      :
4464      :
4465      :
4466      :
4467      :
4468      :
4469      :
4470      :
4471      :
4472      :
4473      :
4474      :
4475      :
4476      :
4477      :
4478      :
4479      :
4480      :
4481      :
4482      :
4483      :
4484      :
4485      :
4486      :
4487      :
4488      :
4489      :
4490      :
4491      :
4492      :
4493      :
4494      :
4495      :
4496      :
4497      :
4498      :
4499      :
4500      :
4501      :
4502      :
4503      :
4504      :
4505      :
4506      :
4507      :
4508      :
4509      :
4510      :
4511      :
4512      :
4513      :
4514      :
4515      :
4516      :
4517      :
4518      :
4519      :
4520      :
4521      :
4522      :
4523      :
4524      :
4525      :
4526      :
4527      :
4528      :
4529      :
4530      :
4531      :
4532      :
4533      :
4534      :
4535      :
4536      :
4537      :
4538      :
4539      :
4540      :
4541      :
4542      :
4543      :
4544      :
4545      :
4546      :
4547      :
4548      :
4549      :
4550      :
4551      :
4552      :
4553      :
4554      :
4555      :
4556      :
4557      :
4558      :
4559      :
4560      :
4561      :
4562      :
4563      :
4564      :
4565      :
4566      :
4567      :
4568      :
4569      :
4570      :
4571      :
4572      :
4573      :
4574      :
4575      :
4576      :
4577      :
4578      :
4579      :
4580      :
4581      :
4582      :
4583      :
4584      :
4585      :
4586      :
4587      :
4588      :
4589      :
4590      :
4591      :
4592      :
4593      :
4594      :
4595      :
4596      :
4597      :
4598      :
4599      :
4600      :
4601      :
4602      :
4603      :
4604      :
4605      :
4606      :
4607      :
4608      :
4609      :
4610      :
4611      :
4612      :
4613      :
4614      :
4615      :
4616      :
4617      :
4618      :
4619      :
4620      :
4621      :
4622      :
4623      :
4624      :
4625      :
4626      :
4627      :
4628      :
4629      :
4630      :
4631      :
4632      :
4633      :
4634      :
4635      :
4636      :
4637      :
4638      :
4639      :
4640      :
4641      :
4642      :
4643      :
4644      :
4645      :
4646      :
4647      :
4648      :
4649      :
4650      :
4651      :
4652      :
4653      :
4654      :
4655      :
4656      :
4657      :
4658      :
4659      :
4660      :
4661      :
4662      :
4663      :
4664      :
4665      :
4666      :
4667      :
4668      :
4669      :
4670      :
4671      :
4672      :
4673      :
4674      :
4675      :
4676      :
4677      :
4678      :
4679      :
4680      :
4681      :
4682      :
4683      :
4684      :
4685      :
4686      :
4687      :
4688      :
4689      :
4690      :
4691      :
4692      :
4693      :
4694      :
4695      :
4696      :
4697      :
4698      :
4699      :
4700      :
4701      :
4702      :
4703      :
4704      :
4705      :
4706      :
4707      :
4708      :
4709      :
4710      :
4711      :
4712      :
4713      :
4714      :
4715      :
4716      :
4717      :
4718      :
4719      :
4720      :
4721      :
4722      :
4723      :
4724      :
4725      :
4726      :
4727      :
4728      :
4729      :
4730      :
4731      :
4732      :
4733      :
4734      :
4735      :
4736      :
4737      :
4738      :
4739      :
4740      :
4741      :
4742      :
4743      :
4744      :
4745      :
4746      :
4747      :
4748      :
4749      :
4750      :
4751      :
4752      :
4753      :
4754      :
4755      :
4756      :
4757      :
4758      :
4759      :
4760      :
4761      :
4762      :
4763      :
4764      :
4765      :
4766      :
4767      :
4768      :
4769      :
4770      :
4771      :
4772      :
4773      :
4774      :
4775      :
4776      :
4777      :
4778      :
4779      :
4780      :
4781      :
4782      :
4783      :
4784      :
4785      :
4786      :
4787      :
4788      :
4789      :
4790      :
4791      :
4792      :
4793      :
4794      :
4795      :
4796      :
4797      :
4798      :
4799      :
4800      :
4801      :
4802      :
4803      :
4804      :
4805      :
4806      :
4807      :
4808      :
4809      :
4810      :
4811      :
4812      :
4813      :
4814      :
4815      :
4816      :
4817      :
4818      :
4819      :
4820      :
4821      :
4822      :
4823      :
4824      :
4825      :
4826      :
4827      :
4828      :
4829      :
4830      :
4831      :
4832      :
4833      :
4834      :
4835      :
4836      :
4837      :
4838      :
4839      :
4840      :
4841      :
4842      :
4843      :
4844      :
4845      :
4846      :
4847      :
4848      :
4849      :
4850      :
4851      :
4852      :
4853      :
4854      :
4855      :
4856      :
4857      :
4858      :
4859      :
4860      :
4861      :
4862      :
4863      :
4864      :
4865      :
4866      :
4867      :
4868      :
4869      :
4870      :
4871      :
4872      :
4873      :
4874      :
4875      :
4876      :
4877      :
4878      :
4879      :
4880      :
4881      :
4882      :
4883      :
4884      :
4885      :
4886      :
4887      :
4888      :
4889      :
4890      :
4891      :
4892      :
4893      :
4894      :
4895      :
4896      :
4897      :
4898      :
4899      :
4900      :
4901      :
4902      :
4903      :
4904      :
4905      :
4906      :
4907      :
4908      :
4909      :
4910      :
4911      :
4912      :
4913      :
4914      :
4915      :
4916      :
4917      :
4918      :
4919      :
4920      :
4921      :
4922      :
4923      :
4924      :
4925      :
4926      :
4927      :
4928      :
4929      :
4930      :
4931      :
4932      :
4933      :
4934      :
4935      :
4936      :
4937      :
4938      :
4939      :
4940      :
4941      :
4942      :
4943      :
4944      :
4945      :
4946      :
4947      :
4948      :
4949      :
4950      :
4951      :
4952      :
4953      :
4954      :
4955      :
4956      :
4957      :
4958      :
4959      :
4960      :
4961      :
4962      :
4963      :
4964      :
4965      :
4966      :
4967      :
4968      :
4969      :
4970      :
4971      :
4972      :
4973      :
4974      :
4975      :
4976      :
4977      :
4978      :
4979      :
4980      :
4981      :
4982      :
4983      :
4984      :
4985      :
4986      :
4987      :
4988      :
4989      :
4990      :
4991      :
4992      :
4993      :
4994      :
4995      :
4996      :
4997      :
4998      :
4999      :
5000      :

```


Spooler runtime 13 Seconds, 83 KCS, 3% disk reads, 0 disk writes, 93 pages printed.

[illegible]