

# RH70

CONTROLLER DIAGNOSTIC  
MD-11-DERHA-C

EP DERHA C DL B  
COPYRIGHT 1977  
FICHE 1 OF 2

MAR 1977  
**digital**  
MADE IN USA

This page contains a grid of 140 small diagnostic charts or tables, arranged in 10 rows and 14 columns. Each cell contains technical data, likely related to the MD-11-DERHA-C controller diagnostic. The data is organized into a structured grid, with each cell containing a small table or chart. The content is too small to read in detail but appears to be a comprehensive set of diagnostic information.

This section contains a grid of 12 columns and 12 rows of small, illegible diagnostic data tables. Each cell in the grid appears to contain a small table or list of data points, likely representing test results or system parameters for various components of the MD-11-DERHA-C controller. The text is too small to read, but the layout is consistent across the grid.

A small, illegible table located in the bottom right corner of the page. It appears to be a continuation of the diagnostic data from the main grid, but the text is too small to read.

B01

ECF100GTBCSE2

00010000

770224

PDP10 411

HDR10ERHACSEQ

00010000

770224



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

1. ABSTRACT

THIS PROGRAM TESTS THE RH70 IN THE PDP11/70 SUBSYSTEM. IT USES ANY WORKING RH70 PERIPHERAL CONNECTED TO THE RH70 UNDER TEST. IT CAN TEST UP TO FOUR RH70S CONNECTED ON THE SUBSYSTEM. ALTHOUGH THE PERIPHERAL CONNECTED IS USED, THE PERIPHERAL IS NOT TESTED BY THIS PROGRAM.

2. REQUIREMENTS

2.1 EQUIPMENT

PDP11/70 COMPUTER WITH CONSOLE TELETYPE, AND ANY ONE OF THE RH70 PERIPHERALS, (FOR EXAMPLE RPO4, RSO3, RSO4, TU16). THE PERIPHERAL MUST BE COMPLETE AND WORKING. FOR EXAMPLE THE RPO4, RSO3, RSO4 MUST HAVE A SCRATCH PACK ON IT. THE TU MUST HAVE A SCRATCH TAPE.

2.2 STORAGE

THIS PROGRAM HAS 16K WORDS OF MEMORY.

2.3 PRELIMINARY PROGRAMS

ALL PDP11/70 CPU DIAGNOSTICS MUST BE RUN ERROR FREE BEFORE THIS DIAGNOSTIC IS RUN.

3. LOADING PROCEDURE

USE STANDARD LOADING PROCEDURES FOR .BIN TAPES.

4. STARTING PROCEDURE

4.1 OPERATIONAL SWITCH SETTINGS

SWITCH	USE
15	HALT ON ERROR
14	LOOP ON TEST
13	INHIBIT ERROR TYPEOUTS
11	INHIBIT ITERATIONS
10	BELL ON ERROR
9	LOOP ON ERROR
8	LOOP ON TEST IN SWR<7:0>

4.2 STARTING ADDRESS

START AT ADDRESS 200 --- FOR STANDARD RH70.  
START AT ADDRESS 210 --- FOR NONSTANDARD RH70.  
STARTING ADDRESS 210 SHOULD BE USED TO TEST ANY ONE RH INSTEAD OF ALL THE RH GIVEN BY 200 START

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

## 4.3 PROGRAM AND/OR OPERATOR ACTION

BEFORE STARTING THE PROGRAM, MAKE SURE THAT THE MAGNETIC MEDIUM IS PROPERLY POSITIONED:

- 1.) TU16 - MAGTAPE MUST BE AT LOAD POINT.
- 2.) RP - HEADS MUST BE IN HOME POSITION.

ON STARTING FROM 210 OPERATOR MUST CHOOSE BETWEEN THE RH70 PERIPHERALS AND TYPE THE BASE ADDRESS OF ANY OR ALL OF THE PERIPHERALS CONNECTED TO THE RH70.

## 5. OPERATING PROCEDURE

## 5.1 OPERATIONAL SWITCH SETTINGS

SEE SECTION 4.1

## 5.2 SUB-ROUTINE ABSTRACTS

SEE SECTION 9 "SUBROUTINES"

## 6. ERRORS

## 6.1 ERROR HALTS AND DESCRIPTION

ON DETECTION OF AN ERROR THE PROGRAM WILL PRINT ALL RELEVANT INFORMATION AND THEN PROCEED ACCORDING TO THE SWITCH SETTINGS GIVEN IN 4.1. IF ALL SWITCHES ARE DOWN THE PROGRAM WILL CONTINUE.

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

DOCUMENT  
\*\*\*\*\*  
MAINDEC-11-DERHAC-A  
\*\*\*\*\*

COPYRIGHT 1975, 1977  
DIGITAL EQUIPMENT CORPORATION  
MAYNARD, MASS. 01754

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

2 COPYRIGHT (C) 1975, 1977  
DIGITAL EQUIPMENT CORP.  
MAYNARD, MASS. 01754

THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC  
PACKAGE (MAINDEC-11-DZQAC-A4).

13

\*\*\*\*\*  
OPERATIONAL SWITCH SETTINGS  
\*\*\*\*\*

14

SWITCH	USE
15	HALT ON ERROR
14	LOOP ON TEST
13	INHIBIT ERROR TYPEOUTS
11	INHIBIT ITERATIONS
10	BELL ON ERROR
9	LOOP ON ERROR
8	LOOP ON TEST IN SWR<7:0>
7	STOP FURTHER COMPARES IF SW08 IS LOW
6	TYPE ALL REG. WITH ERROR IF SW6 LOW

27

\*\*\*\*\*  
BASIC DEFINITIONS  
\*\*\*\*\*

- 29 INITIAL ADDRESS OF THE STACK POINTER \*\*\* 1000 \*\*\*
- 40 MISCELLANEOUS DEFINITIONS
- 46 GENERAL PURPOSE REGISTER DEFINITIONS
- 59 PRIORITY LEVEL DEFINITIONS
- 69 "SWITCH REGISTER" SWITCH DEFINITIONS
- 96 DATA BIT DEFINITIONS (BIT00 TO BIT15)
- 124 BASIC "CPL" TRAP VECTOR ADDRESSES



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

```

*****
173 TRAP CATCHER
*****

142 ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS

*****
146 STARTING ADDRESS(ES)
*****

152 STARTING ADDRESS 200 FOR NORMAL STARTS
THIS WILL TEST ALL RPO4'S ON THE SYSTEM A SINGLE DRIVE AT A TIME
STARTING ADDRESS 210 WILL TEST ONLY ONE SPECIFIED DRIVE

*****
160 MEMORY MANAGEMENT DEFINITIONS
*****

162 KT11 VECTOR ADDRESS
166 KT11 STATUS REGISTER ADDRESSES
173 KERNEL "I" PAGE DESCRIPTOR REGISTERS
184 KERNEL "I" PAGE ADDRESS REGISTERS

*****
199 COMMON TAGS
*****

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

```

MAINDEC-11-DERHAC-A

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

```

252 *****
    ERROR POINTER TABLE
    *****

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

260          EM          ;;POINTS TO THE ERROR MESSAGE
             DH          ;;POINTS TO THE DATA HEADER
             DT          ;;POINTS TO THE DATA
             DF          ;;POINTS TO THE DATA FORMAT

269 *****

1587 *****
    REGISTER ADDRESSES
    *****

1834 *****
    REGISTER TEST
    *****

1899 TEST 1 SIZE FOR RH DEVICES
    THIS TEST DETERMINS WHICH RH DEVICE IS ON THE SYSTEM.
    IF THE SYSTEM HAS MORE THAN ONE RH DEVICE THEN ONLY ONE
    THE DEVICES WILL BE USED IN THE FOLLOWING TESTS.
    THE ONE THAT WILL BE USED IS THE FIRST ONE THAT IS PRESE
    IN THE FOLLOWING LIST
    RS
    RP
    TM

    THE WAY THE TEST WORKS IS AS FOLLOWS:-
    A REFERENCE IS MADE TO THE CONTROL AND STATUS REGISTER 1
    FOR THE RS BY A TST INSTRUCTION. IF IT RESPONDS THEN IT
    ASSUMED PRESENT AND THE LOCATIONS FOR THE I/O REGISTERS
    FILLED WITH THE APPROPRIATE ADDRESSES.
    THEN THE DRIVE TYPE REGISTER IS CHECKED TO HAVE GOOD
    VALUES.
    IF THE TST INSTRUCTION TRAPS TO A NON EXISTANT VECTOR
    THEN A SIMILAR ATTEMPT IS MADE TO A RP CONTROL AND STATU
    REGISTER 1.
    THEN A TM IS TRYED.
    THEN A MIXED SYSTEM WITH MORE THAN ONE RH DEVICES IS TRYE

2195 TEST 2 UNIT UNDER TEST
    THIS TYPES THE UNIT TO BE TESTED
    AND IS THE FIRST TEST AFTER THE END OF THE PROGRAM

```

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

MAINDEC-11-DERHAC-A

2302

TEST 3 BIT BANG RHCS1

2304

TEST LOADING AND READING OF ALL POSSIBLE BITS  
IN RHCS1 REGISTER.  
USE A PATTERN OF WALKING 1'S (1,2,4,10 ETC)  
AND WALKING 0'S (-2,-3,-5 ETC)  
IN THIS TEST SC!TRE!MCPE!DVA!BIT12!BIT10!RDY!GO BITS WIL  
AND RDY!DVA BITS WILL ALWAYS BE SET  
AND BIT10 BITS WILL ALWAYS BE CLEARED  
IF SWITCH 14 OR 9 OR 8 ARE SET FOR DEBUGGING  
THEN A RH CLEAR (RHCS2 BIT #5) WILL BE  
GIVEN TO AID SCOPE SYNC'S ON THE CLEAR  
SIGNAL.

2383

TEST 4 BIT BANG RHCS2

2385

TEST LOADING AND READING OF ALL POSSIBLE BITS  
IN RHCS2 REGISTER.  
USE A PATTERN OF WALKING 1'S (1,2,4,10 ETC)  
AND WALKING 0'S (-2,-3,-5 ETC)  
IN THIS TEST 177740 BITS WILL NOT BE WRITTEN INTO  
AND IR BITS WILL ALWAYS BE SET  
IF SWITCH 14 OR 9 OR 8 ARE SET FOR DEBUGGING  
THEN A RH CLEAR (RHCS2 BIT #5) WILL BE  
GIVEN TO AID SCOPE SYNC'S ON THE CLEAR  
SIGNAL.

2460

TEST 5 BIT BANG RHCS3

2462

TEST LOADING AND READING OF ALL POSSIBLE BITS  
IN RHCS3 REGISTER.  
USE A PATTERN OF WALKING 1'S (1,2,4,10 ETC)  
AND WALKING 0'S (-2,-3,-5 ETC)  
IN THIS TEST 177660 BITS WILL NOT BE WRITTEN INTO  
AND 0 BITS WILL ALWAYS BE SET  
AND BIT9!BIT8!BIT7!BIT5!BIT4 BITS WILL ALWAYS BE CLEARED  
IF SWITCH 14 OR 9 OR 8 ARE SET FOR DEBUGGING  
THEN A RH CLEAR (RHCS2 BIT #5) WILL BE  
GIVEN TO AID SCOPE SYNC'S ON THE CLEAR  
SIGNAL.

2541

TEST 6 BIT BANG RHWC

2543

TEST LOADING AND READING OF ALL POSSIBLE BITS  
IN RHWC REGISTER.  
USE A PATTERN OF WALKING 1'S (1,2,4,10 ETC)  
AND WALKING 0'S (-2,-3,-5 ETC)  
IN THIS TEST 0 BITS WILL NOT BE WRITTEN INTO  
AND 0 BITS WILL ALWAYS BE SET  
IF SWITCH 14 OR 9 OR 8 ARE SET FOR DEBUGGING  
THEN A RH CLEAR (RHCS2 BIT #5) WILL BE  
GIVEN TO AID SCOPE SYNC'S ON THE CLEAR  
SIGNAL.

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

MAINDEC-11-DERHAC-A

2518 TEST 7 BIT BANG RHBA  
 2620 TEST LOADING AND READING OF ALL POSSIBLE BITS  
 IN RHBA REGISTER.  
 USE A PATTERN OF WALKING 1'S (1,2,4,10 ETC)  
 AND WALKING 0'S (-2,-3,-5 ETC)  
 IN THIS TEST 0 BITS WILL NOT BE WRITTEN INTO  
 AND 0 BITS WILL ALWAYS BE SET  
 AND BIT00 BITS WILL ALWAYS BE CLEARED  
 IF SWITCH 14 OR 9 OR 8 ARE SET FOR DEBUGGING  
 THEN A RH CLEAR (RHCS2 BIT #5) WILL BE  
 GIVEN TO AID SCOPE SYNCs ON THE CLEAR  
 SIGNAL.

2699 TEST 10 BIT BANG RHBAE  
 2701 TEST LOADING AND READING OF ALL POSSIBLE BITS  
 IN RHBAE REGISTER.  
 USE A PATTERN OF WALKING 1'S (1,2,4,10 ETC)  
 AND WALKING 0'S (-2,-3,-5 ETC)  
 THIS TEST 177700 BITS WILL NOT BE WRITTEN INTO  
 AND 0 BITS WILL ALWAYS BE SET  
 AND 177700 BITS WILL ALWAYS BE CLEARED  
 IF SWITCH 14 OR 9 OR 8 ARE SET FOR DEBUGGING  
 THEN A RH CLEAR (RHCS2 BIT #5) WILL BE  
 GIVEN TO AID SCOPE SYNCs ON THE CLEAR  
 SIGNAL.

2784 TEST 11 SILO TEST 1 (ONE WORD WRITE)  
 AFTER A RH CLEAR IS GIVEN (SET BIT #5 IN RHCS2)  
 RHCS2 IS CHECKED TO HAVE "IR" (BIT #6) HIGH  
 TOGETHER WITH UNIT NUMBER  
 ONE WORD OF ALL ZEROS IS WRITTEN INTO RHDB  
 BY "CLR"  
 "OR" (BIT #7) IS WAITED FOR BY A TRAP INSTRUCTION  
 CALLED "WAT" (NO TIMING IS DONE)  
 HOWEVER IF "OR" DOES NOT SET WITHIN "WAT" COUNT  
 DOWN AN ERROR IS REPORTED  
 RHDB IS READ AND CHECKED TO CONTAIN ZEROS  
 RHCS2 WILL BE CHECKED TO HAVE "IR" AND UNIT NUMBER  
 RHCS1, RHCS3, RHBA, RHBAE, RHWC, WILL BE CHECKED TO HAVE  
 APPROPRIATE VALUES

2973 TEST 12 SILO TEST 2 (ONE WORD WRITE)  
 AFTER A RH CLEAR IS GIVEN (SET BIT #5 IN RHCS2)  
 RHCS2 IS CHECKED TO HAVE "IR" (BIT #6) HIGH  
 TOGETHER WITH UNIT NUMBER  
 ONE WORD OF ALL ONES IS WRITTEN INTO RHDB  
 BY "CLR"  
 "OR" (BIT #7) IS WAITED FOR BY A TRAP INSTRUCTION  
 CALLED "WAT" (NO TIMING IS DONE)  
 HOWEVER IF "OR" DOES NOT SET WITHIN "WAT" COUNT  
 DOWN AN ERROR IS REPORTED  
 RHDB IS READ AND CHECKED TO CONTAIN ALL ONES

MAINDEC-11-DERHAC-A

RHCS2 WILL BE CHECKED TO HAVE "IR" AND UNIT NUMBER  
RHCS1, RHCS3, RHBA, RHBAE, RHWC, WILL BE CHECKED TO HAVE  
APPROPRIATE VALUES

3144 TEST 13 SILO TEST 3 (TWO WORD WRITE)

3146 AFTER A RH CLEAR IS GIVEN (SET BIT #5 IN RHCS2)  
RHCS2 IS CHECKED TO HAVE "IR" (BIT #6) HIGH,  
TOGETHER WITH UNIT NUMBER  
ONE WORD = 52525 IS WRITTEN INTO RHDB  
RHCS2 IS CHECKED TO HAVE "IR" AND UNIT NUMBER  
A SECOND WORD = 12525 IS WRITTEN INTO RHDB  
"OR" (BIT #7 IN RHCS2) IS WAITED FOR BY A TRAP  
INSTRUCTION CALLED "WAT"  
RHDB IS READ AND CHECKED TO CONTAIN 52525  
RHCS2 IS CHECKED TO HAVE "IR", "OR" AND UNIT NUMBER  
RHDB IS READ A SECOND TIME AND CHECKED TO  
CONTAIN 125252  
RHCS2 IS CHECKED TO HAVE "IR" AND UNIT NUMBER  
THEN ALL REGISTERS RHCS1, RHCS3, RHBA, RHBAE, RHWC  
ARE CHECKED TO HAVE APPROPRIATE VALUE

3416 TEST 14 SILO TEST 4 (COUNT PATTERN)

3418 AFTER A RH CLEAR IS GIVEN (SET BIT #5 IN RHCS2)  
RHCS2 IS CHECKED TO HAVE "IR" (BIT #6) HIGH, TOGETHER  
WITH UNIT NUMBER  
EIGHT WORDS, A COUNT PATTERN 0 THRU 7 IS WRITTEN  
INTO RHDB  
"OR" (BIT #7 IN RHCS2) IS WAITED FOR BY A TRAP  
INSTRUCTION CALLED "WAT"  
THEN RHCS2 IS CHECKED TO HAVE "IR" LOW AND  
"OR" HIGH TOGETHER WITH THE UNIT NUMBER  
THEN RHDB IS READ AND COMPARED TO HAVE THE RIGHT VALUE  
EIGHT TIMES  
THEN RHCS2, RHCS1, RHCS3, RHBA, RHBAE, RHWC ARE  
CHECKED TO HAVE THE APPROPRIATE VALUE

3808 TEST 15 SILO TEST 5 (FLOATING ONES)

3810 AFTER ARH CLEAR IS GIVEN (SET BIT #5 IN RHCS2)  
RHCS2 IS CHECKED TO HAVE "IR" (BIT #6) HIGH,  
TOGETHER WITH UNIT NUMBER  
EIGHT WORDS, A PATTERN OF FLOATING ONES (1,2,4,10,20,40,  
IS WRITTEN INTO RHDB  
"OR" (BIT #7 IN RHCS2) IS WAITED FOR BY A TRAP  
INSTRUCTION CALLED "WAT"  
THEN RHCS2 IS CHECKED TO HAVE "IR" LOW AND  
"OR" HIGH TOGETHER WITH THE UNIT NUMBER  
THEN RHDB IS READ AND COMPARED TO HAVE THE  
RIGHT VALUE AFTER EACH OF THE EIGHT READS.  
THEN RHCS2 IS CHECKED TO HAVE "IR"

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

MAINDEC-11-DERHAC-A

AND UNIT NUMBER  
THEN RHCS1, RHCS3, RHBA, RHBAE, RHWC ARE  
CHECKED TO HAVE THE APPROPRIATE VALUE

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

4227 TEST 16 SILO TEST 6 (FLOATING ONES IN UPPER BYTE)

4229 AFTER A RH CLEAR IS GIVEN (SET BIT #5 IN RHCS2)  
RHCS2 IS CHECKED TO HAVE "IR" (BIT #6) HIGH,  
TOGETHER WITH UNIT NUMBER  
EIGHT WORDS, A PATTERN OF FLOATING ONES IN UPPER BYTE  
400, 1000, 2000, 4000, 10000, 20000, 40000, 100000  
IS WRITTEN INTO RHDB  
"OR" (BIT #7 IN RHCS2) IS WAITED FOR BY A TRAP  
INSTRUCTION CALLED "WAT"  
THEN RHCS2 IS CHECKED TO HAVE "IR" LOW AND  
"OR" HIGH TOGETHER WITH THE UNIT NUMBER  
THEN RHDB IS READ AND COMPARED TO HAVE THE  
RIGHT VALUE AFTER EACH OF THE EIGHT READS.  
THEN RHCS2 IS CHECKED TO HAVE "IR"  
AND UNIT NUMBER  
THEN RHCS1, RHCS3, RHBA, RHBAE, RHWC ARE  
CHECKED TO HAVE THE APPROPRIATE VALUE

4649 TEST 17 SILO TEST 7 (FLOATING 0 IN LOWER BYTE)

4651 AFTER A RH CLEAR IS GIVEN (SET BIT #5 IN RHCS2)  
RHCS2 IS CHECKED TO HAVE "IR" (BIT #6) HIGH,  
TOGETHER WITH UNIT NUMBER  
EIGHT WORDS OF FLOATING ZEROS 177776, 177775, 177773,  
177767, 177757, 177737, 177677, 177577  
IS WRITTEN INTO RHDB  
"OR" (BIT #7 IN RHCS2) IS WAITED FOR BY A TRAP  
INSTRUCTION CALLED "WAT"  
THEN RHCS2 IS CHECKED TO HAVE "IR" LOW AND  
"OR" HIGH TOGETHER WITH THE UNIT NUMBER  
THEN RHDB IS READ AND COMPARED TO HAVE THE  
RIGHT VALUE AFTER EACH OF THE EIGHT READS.  
THEN RHCS2 IS CHECKED TO HAVE "IR"  
AND UNIT NUMBER  
THEN RHCS1, RHCS3, RHBA, RHBAE, RHWC ARE  
CHECKED TO HAVE THE APPROPRIATE VALUE

5070 TEST 20 SILO TEST 8 (FLOATING ZEROS IN UPPER BYTE)

5072 AFTER A RH CLEAR IS GIVEN (SET BIT #5 IN RHCS2)  
RHCS2 IS CHECKED TO HAVE "IR" (BIT #6) HIGH,  
TOGETHER WITH UNIT NUMBER  
5074 EIGHT WORDS, A PATTERN OF FLOATING ZEROS IN UPPER BYTE  
177377, 177677, 175777, 173777, 167777, 157777, 137777.  
IS WRITTEN INTO RHDB  
"OR" (BIT #7 IN RHCS2) IS WAITED FOR BY A TRAP  
INSTRUCTION CALLED "WAT"  
THEN RHCS2 IS CHECKED TO HAVE "IR" LOW AND

MAINDEC-11-DERHAC-A

"OR" HIGH TOGETHER WITH THE UNIT NUMBER  
THEN RHDB IS READ AND COMPARED TO HAVE THE  
RIGHT VALUE AFTER EACH OF THE EIGHT READS.  
THEN RHCS2 IS CHECKED TO HAVE "IR"  
AND UNIT NUMBER  
THEN RHCS1, RHCS3, RHBA, RHBAE, RHWC ARE  
CHECKED TO HAVE THE APPROPRIATE VALUE

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

5508 TEST 21 RHCS1 - MCPE BIT #13 (PARITY LINE = 0)

5510 AN RH CLEAR (SET BIT #5 IN RHCS2) IS GIVEN TO  
CLEAR ALL DEVICE REGISTERS  
THEN RHER1 (ERROR REGISTER 1) IS CHECKED TO HAVE  
ZEROS  
SET "PAT" (BIT #4 IN RHCS2) TO INVERT PARITY CHECKING  
READ ANY DEVICE REGISTER - HERE RHER1  
READ AND CHECK RHCS1 TO CONTAIN SC (BIT #15)  
AND MCPE (BIT #13)  
WRITE "1" INTO TRE (BIT #14 IN RHCS1)  
READ RHCS1 SC, MCPE, AND RDY SHOULD BE SET  
GIVE AN RH CLEAR (CLR - BIT #5 IN RHCS2)  
CHECK RHCS1 TO HAVE RDY  
CHECK RHCS2 TO HAVE ONLY IR AND UNIT NUMBER  
CHECK RHCS3, RHBA, RHBAE, RHWC TO HAVE APPROPRIATE  
VALUES.

5785 TEST 22 RHCS1 - MCPE BIT #13 (PARITY LINE = 1)

5787 AN RH CLEAR (SET BIT #5 IN RHCS2) IS GIVEN TO CLEAR  
ALL DEVICE REGISTERS  
WRITE A ONE INTO DISK ADDRESS REGISTER (RHDA)  
(IN TAPE DRIVES CALLED REGISTER)  
SET "PAT" (RHCS2 BIT #4) THIS WILL INVERT THE PARITY CHE  
READ RHDA AND COMPARE IT HAS ONE IN IT  
THIS READING SHOULD SET SC, MCPE IN RHCS1  
CHECK RHCS1 TO HAVE ONLY RDY SET  
CHECK RHCS2, RHCS3, RHBA, RHBAE, RHWC TO HAVE APPROPRIATE  
VALUES

5959 TEST 23 TEST DUPLICATED A16 (RHCS1 BIT #8)

5961 CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)  
MOVE 400 (ONE INTO A16) IN RHCS1  
READ RHCS1 TO CONTAIN 600 (BIT #8 AND RDY)  
READ RHBAE TO CONTAIN "1" (BIT #0 HIGH)

MOVE 0 INTO RHBAE (WRITING 0 INTO RHBAE BIT #0)  
READ RHBAE TO CONTAIN 0  
READ RHCS1 TO CONTAIN ONLY RDY (BIT #8 IN RHCS1 IS ZERO)

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

111

MAINDEC-11-DERHAC-A

6079

TEST 24 TEST DUPLICATED A17 (RHCS1 BIT #9)

6081

CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)  
MOVE 400 (ONE INTO A17) IN RHCS1  
READ RHCS1 TO CONTAIN 1200 (BIT #9 AND RDY)  
READ RHBAE TO CONTAIN "2" (BIT #1 HIGH)

MOVE 0 INTO RHBAE (WRITING 0 INTO RHBAE BIT #1)  
READ RHBAE TO CONTAIN 2  
READ RHCS1 TO CONTAIN ONLY RDY (BIT #9 IN RHCS1 IS ZERO)

6199

TEST 25 TEST DUPLICATED IE (RHCS1 BIT #6)

6201

CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)  
MOVE 100 (ONE INTO IE) IN RHCS1  
READ RHCS1 TO CONTAIN 300 (BIT #6 AND RDY)  
READ RHCS3 TO CONTAIN "100" (BIT #6 HIGH)

MOVE 0 INTO RHCS3 (WRITING 0 INTO RHCS3 BIT #6)

6208

READ RHCS3 TO CONTAIN 100  
READ RHCS1 TO CONTAIN ONLY RDY (BIT #6 IN RHCS1 IS ZERO)

6319

TEST 26 RHCS1 PROGRAM ERROR (PGE BIT #10)

6321

CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)  
SET UP FOR A 10 WORD WRITE  
SET "GO" (RHCS1 BIT #0) TWICE IN TWO SUCCESSIVE INSTRUCTIONS  
THIS SHOULD SET SC AND TRE IN RHCS1  
AND PGE SHOULD BE SET IN RHCS2  
RHCS3, ARE CHECKED  
THE NUMBER OF WORDS ARE LARGE ENOUGH SO THAT AFTER ONE "BIS" INSTRUCTION TO SET "GO" IN RHCS1 THERE IS SUFFICIENT TIME TO GIVE ANOTHER "BIS" INSTRUCTION TO SET "GO" BEFORE THE FIRST "GO" HAS TIME TO COMPLETE

6415

TEST 27 RHCS2 - BUS ADDRESS INHIBIT BIT #3

6417

CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #6)  
SET UP FOR A 2 WORD WRITE FROM A BUFFER TAGGED "WRFROM"  
SET BUS ADDRESS INHIBIT RHCS2 BIT #3 "BAI"  
AT THE END OF WRITE CHECK  
RHCS1 TO HAVE ONLY RDY AND COMMAND  
RHCS2 TO HAVE BAI  
RHCS3 TO HAVE 0



MAINDEC-11-DERHAC-A  
6424RHBA TO HAVE ADDRESS OF "WRFROM"  
RHBAE AND RHWC TO HAVE ZEROSNOW SET UP READ FOR THE SAME DATA WITH  
BAI SET TO READ INTO A BUFFER TAGGED "REINTO"  
AFTER READ CHECK  
RHCS1 TO HAVE ONLY RDY AND COMMAND  
RHCS2 TO HAVE BAI  
RHCS3 TO HAVE 0RHBA TO HAVE ADDRESS OF "REINTO"  
RHBAE AND RHWC TO HAVE ZEROS  
DATA IN REINTO BUFFER IS CHECKED WITH DATA IN  
WRFROM BUFFER

- 6571 TEST 30 RHSC2 MDPE BIT #8 AND RHCS3 IPCK0 BIT #0  
CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)  
SET IPCK0 (RHCS3 BIT #0)  
MOVE ALL ZEROS INTO RHDB ONCE  
THIS SHOULD SET TRE SC IN RHCS1  
READ RHDB ONCE THIS SHOULD SET MDPE (RHCS2 BIT #8)  
CHECK RHCS1 FOR RDY (BIT #7), TRE (BIT #14), SC (BIT #15)  
"CLR" (RHCS2 BIT #5) IS GIVEN  
ALL ERRORS ARE CLEARED  
CHECK RHCS1, RHCS2, RHCS3, RHBA, RHBAE, RHWC
- 6807 TEST 31 RHSC2 MDPE BIT #8 AND RHCS3 IPCK1 BIT #1  
CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)  
SET IPCK1 (RHCS3 BIT #1)  
MOVE ALL ZEROS INTO RHDB ONCE  
THIS SHOULD SET TRE SC IN RHCS1  
READ RHDB ONCE THIS SHOULD SET MDPE (RHCS2 BIT #8)  
CHECK RHCS1 FOR RDY (BIT #7), TRE (BIT #14), SC (BIT #15)  
"CLR" (RHCS2 BIT #5) IS GIVEN  
ALL ERRORS ARE CLEARED  
CHECK RHCS1, RHCS2, RHCS3, RHBA, RHBAE, RHWC
- 7043 TEST 32 RHSC2 MDPE BIT #8 AND RHCS3 IPCK2 BIT #2  
CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)  
SET IPCK2 (RHCS3 BIT #2)  
MOVE ALL ZEROS INTO RHDB TWICE  
THIS SHOULD NOT SET TRE SC IN RHCS1  
READ RHDB ONCE THIS SHOULD SET MDPE (RHCS2 BIT #8)  
CHECK RHCS1 FOR RDY (BIT #7), TRE (BIT #14), SC (BIT #15)  
READ RHDB A SECOND TIME. CHECK RHCS1, RHCS2, RHCS3  
"CLR" (RHCS2 BIT #5) IS GIVEN  
ALL ERRORS ARE CLEARED  
CHECK RHCS1, RHCS2, RHCS3, RHBA, RHBAE, RHWC
- 7345 TEST 33 RHSC2 MDPE BIT #8 AND RHCS3 IPCK3 BIT #3  
CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)  
SET IPCK3 (RHCS3 BIT #3)  
MOVE ALL ZEROS INTO RHDB TWICE  
THIS SHOULD NOT SET TRE SC IN RHCS1  
READ RHDB ONCE THIS SHOULD SET MDPE (RHCS2 BIT #8)

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

MAINDEC-11-DERHAC-A

699  
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

CHECK RHCS1 FOR RDY (BIT #7), TRE (BIT #14), SC (BIT #15)  
 READ RHDB A SECOND TIME. CHECK RHCS1, RHCS2, RHCS3  
 "CLR" (RHCS2 BIT #5) IS GIVEN  
 ALL ERRORS ARE CLEARED  
 CHECK RHCS1, RHCS2, RHCS3, RHBA, RHBAE, RHWC

7648 TEST 34 RHCS2-WCE BIT #14 AND RHCS3-WCE OW BIT #12

7650 CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5).  
 WRITE NINE WORD OF ALL NINES ON TO THE DEVICE  
 SET UP TO DO WRITE CHECK ON THAT WORD FROM AN  
 ODD WORD BOUNDARY  
 DO A ONE WORD WRITE CHECK WITH RHBA FROM AN  
 ODD WORD BOUNDARY  
 THIS SHOULD SET WCE OW (RHCS3 BIT #12)  
 AND WCE (RHCS2 BIT #14)  
 "CLR" (RHCS2 BIT #5) IS GIVEN  
 ALL ERRORS ARE CLEARED  
 CHECK RHCS1, RHCS2, RHCS3, RHBA, RHBAE, RHWC

7881 TEST 35 RHCS2-WCE BIT #14 AND RHCS3-WCE OW BIT #12

7883 CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5).  
 WRITE NINE WORD OF ALL NINES ON TO THE DEVICE  
 SET UP TO DO WRITE CHECK ON THAT WORD FROM AN  
 ODD WORD BOUNDARY  
 DO A ONE WORD WRITE CHECK WITH RHBA FROM AN  
 ODD WORD BOUNDARY  
 THIS SHOULD SET WCE OW (RHCS3 BIT #12)  
 AND WCE (RHCS2 BIT #14)  
 "CLR" (RHCS2 BIT #5) IS GIVEN  
 ALL ERRORS ARE CLEARED  
 CHECK RHCS1, RHCS2, RHCS3, RHBA, RHBAE, RHWC

8114 TEST 36 RHCS2-WCE BIT #14 AND RHCS3-WCE EW BIT #11

8116 CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5).  
 WRITE NINE WORD OF ALL NINES ON TO THE DEVICE  
 SET UP TO DO WRITE CHECK ON THAT WORD FROM AN  
 EVEN WORD BOUNDARY  
 DO A ONE WORD WRITE CHECK WITH RHBA FROM AN  
 EVEN WORD BOUNDARY  
 THIS SHOULD SET WCE EW (RHCS3 BIT #11)  
 AND WCE (RHCS2 BIT #14)  
 "CLR" (RHCS2 BIT #5) IS GIVEN  
 ALL ERRORS ARE CLEARED  
 CHECK RHCS1, RHCS2, RHCS3, RHBA, RHBAE, RHWC

8347 TEST 37 RHCS2-WCE BIT #14 AND RHCS3-WCE EW BIT #11

MAINDEC-11-DERHAC-A  
8349

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

CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)  
WRITE NINE WORD OF ALL NINES ON TO THE DEVICE  
SET UP TO DO WRITE CHECK ON THAT WORD FROM AN  
EVEN WORD BOUNDARY  
DO A ONE WORD WRITE CHECK WITH RHBA FROM AN  
EVEN WORD BOUNDARY  
THIS SHOULD SET WCE EW (RHCS3 BIT #11)  
AND WCE (RHCS2 BIT #14)  
"CLR" (RHCS2 BIT #5) IS GIVEN  
ALL ERRORS ARE CLEARED  
CHECK RHCS1, RHCS2, RHCS3, RHBA, RHBAE, RHWC

8581 TEST 40 TEST DBL (RHCS3 BIT #10) TEST A

8583 CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)

8584 SET UP FOR A NINE WORD WRITE FROM AN EVEN WORD BOUNDARY  
DO A NINE WORD WRITE FROM AN EVEN WORD BOUNDARY  
THIS SHOULD NOT SET RHCS3 BIT #10 DBL  
CHECK RHCS3  
CHECK RHCS1, RHCS2, RHBA, RHBAE, RHWC

9694 TEST 41 TEST DBL (RHCS3 BIT #10) TEST B

8696 CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)  
SET UP FOR A TEN WORD WRITE FROM AN EVEN WORD BOUNDARY  
DO A TEN WORD WRITE FROM AN EVEN WORD BOUNDARY  
THIS SHOULD SET RHCS3 BIT #10 DBL  
CHECK RHCS3  
CHECK RHCS1, RHCS2, RHBA, RHBAE, RHWC

8803 TEST 42 TEST DBL (RHCS3 BIT #10) TEST C

8810 CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)  
SET UP FOR A TEN WORD WRITE FROM AN ODD WORD BOUNDARY  
DO A TEN WORD WRITE FROM AN ODD WORD BOUNDARY  
THIS SHOULD NOT SET RHCS3 BIT #10 DBL  
CHECK RHCS3  
CHECK RHCS1, RHCS2, RHBA, RHBAE, RHWC

8921 TEST 43 TEST DBL (RHCS3 BIT #10) TEST D

8923 CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)  
SET UP FOR A ELEVEN WORD WRITE FROM AN EVEN WORD BOUNDARY  
DO A ELEVEN WORD WRITE FROM AN EVEN WORD BOUNDARY  
THIS SHOULD NOT SET RHCS3 BIT #10 DBL  
CHECK RHCS3  
CHECK RHCS1, RHCS2, RHBA, RHBAE, RHWC

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

MAINDEC-11-DERHAC-A

9034 TEST 44 TEST DBL (RHCS3 BIT #10) TEST E

9036 CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)  
SET UP FOR A TEN WORD WRITE FROM AN EVEN WORD BOUNDARY  
WITH BAI IN RHCS2 BIT #3 SET  
DO A TEN WORD WRITE FROM AN EVEN WORD BOUNDARY  
THIS SHOULD NOT SET RHCS3 BIT #10 DBL  
CHECK RHCS3  
CHECK RHCS1, RHCS2, RHBA, RHBAE, RHWC

9153 TEST 45 TEST DBL (RHCS3 BIT #10) TEST F

9155 CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)  
SET UP FOR A ELEVEN WORD WRITE FROM AN ODD WORD BOUNDARY  
DO A ELEVEN WORD WRITE FROM AN ODD WORD BOUNDARY  
THIS SHOULD SET RHCS3 BIT #10 DBL  
CHECK RHCS3  
CHECK RHCS1, RHCS2, RHBA, RHBAE, RHWC

9268 TEST 46 TEST DBL (RHCS3 BIT #10) TEST G

9270 CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)  
SET UP FOR A NINE WORD READ FROM AN EVEN WORD BOUNDARY  
DO A NINE WORD READ FROM AN EVEN WORD BOUNDARY  
THIS SHOULD NOT SET RHCS3 BIT #10 DBL  
CHECK RHCS3  
CHECK RHCS1, RHCS2, RHBA, RHBAE, RHWC

9381 TEST 47 TEST DBL (RHCS3 BIT #10) TEST H

9383 CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)  
SET UP FOR A TEN WORD READ FROM AN EVEN WORD BOUNDARY  
DO A TEN WORD READ FROM AN EVEN WORD BOUNDARY  
THIS SHOULD SET RHCS3 BIT #10 DBL  
CHECK RHCS3  
CHECK RHCS1, RHCS2, RHBA, RHBAE, RHWC

9495 TEST 50 TEST DBL (RHCS3 BIT #10) TEST I

9497 CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)  
SET UP FOR A TEN WORD READ FROM AN ODD WORD BOUNDARY  
DO A TEN WORD READ FROM AN ODD WORD BOUNDARY  
THIS SHOULD NOT SET RHCS3 BIT #10 DBL  
CHECK RHCS3

9502 CHECK RHCS1, RHCS2, RHBA, RHBAE, RHWC

9508 TEST 51 TEST DBL (RHCS3 BIT #10) TEST J

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

MAINDEC-11-DERHAC-A  
9610

CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)  
SET UP FOR A ELEVEN WORD READ FROM AN EVEN WORD BOUNDARY  
DO A ELEVEN WORD READ FROM AN EVEN WORD BOUNDARY  
THIS SHOULD NOT SET RHCS3 BIT #10 DBL  
CHECK RHCS3  
CHECK RHCS1, RHCS2, RHBA, RHBAE, RHWC

9721 TEST 52 TEST DBL (RHCS3 BIT #10) TEST K

9723 CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)  
SET UP FOR A ELEVEN WORD READ FROM AN ODD WORD BOUNDARY  
DO A ELEVEN WORD READ FROM AN ODD WORD BOUNDARY  
THIS SHOULD SET RHCS3 BIT #10 DBL  
CHECK RHCS3  
CHECK RHCS1, RHCS2, RHBA, RHBAE, RHWC

9835 TEST 53 TEST DBL (RHCS3 BIT #10) TEST L

9837 CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)  
SET UP FOR A TEN WORD READ FROM AN EVEN WORD BOUNDARY  
WITH BAI IN RHCS2 BIT #3 SET  
DO A TEN WORD READ FROM AN EVEN WORD BOUNDARY  
THIS SHOULD NOT SET RHCS3 BIT #10 DBL  
CHECK RHCS3  
CHECK RHCS1, RHCS2, RHBA, RHBAE, RHWC

9955 TEST 54 TEST DBL (RHCS3 BIT #10) TEST M

9957 CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)  
SET UP FOR A ONE WORD WRITE REVERSE FROM AN EVEN WORD BO  
DO A ONE WORD WRITE REVERSE FROM AN EVEN WORD BOUNDARY  
THIS SHOULD NOT SET RHCS3 BIT #10 DBL  
CHECK RHCS3  
CHECK RHCS1, RHCS2, RHBA, RHBAE, RHWC

10068 TEST 55 TEST DBL (RHCS3 BIT #10) TEST N

10070 CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)  
SET UP FOR A TWO WORD WRITE REVERSE FROM AN EVEN WORD BO  
DO A TWO WORD WRITE REVERSE FROM AN EVEN WORD BOUNDARY  
THIS SHOULD NOT SET RHCS3 BIT #10 DBL  
CHECK RHCS3  
CHECK RHCS1, RHCS2, RHBA, RHBAE, RHWC

10181 TEST 56 TEST DBL (RHCS3 BIT #10) TEST O

10183 CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)  
SET UP FOR A TWO WORD WRITE REVERSE FROM AN ODD WORD BOU  
DO A TWO WORD WRITE REVERSE FROM AN ODD WORD BOUNDARY  
THIS SHOULD SET RHCS3 BIT #10 DBL  
CHECK RHCS3  
CHECK RHCS1, RHCS2, RHBA, RHBAE, RHWC

903	MAINDEC-11-DERHAC-A	
904	10295	TEST 57 TEST DBL (RHCS3 BIT #10) TEST P
905		
906	10297	CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5); SET UP FOR A THREE WORD WRITE REVERSE FROM AN EVEN WORD DO A THREE WORD WRITE REVERSE FROM AN EVEN WORD BOUNDARY THIS SHOULD SET RHCS3 BIT #10 DBL CHECK RHCS3 CHECK RHCS1, RHCS2, RHBA, RHBAE, RHWC
907		
908		
909		
910		
911		
912		
913	10409	TEST 60 TEST DBL (RHCS3 BIT #10) TEST Q
914		
915	10411	CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5); SET UP FOR A THREE WORD WRITE REVERSE FROM AN ODD WORD B DO A THREE WORD WRITE REVERSE FROM AN ODD WORD BOUNDARY THIS SHOULD NOT SET RHCS3 BIT #10 DBL CHECK RHCS3 CHECK RHCS1, RHCS2, RHBA, RHBAE, RHWC
916		
917		
918		
919		
920		
921		
922	10522	TEST 61 TEST DBL (RHCS3 BIT #10) TEST R
923		
924	10524	CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5); SET UP FOR A TWO WORD WRITE REVERSE FROM AN EVEN WORD BC WITH BAI IN RHCS2 BIT #3 SET DO A TWO WORD WRITE REVERSE FROM AN EVEN WORD BOUNDARY
925		
926		
927		
928		
929		
930	10528	THIS SHOULD NOT SET RHCS3 BIT #10 DBL CHECK RHCS3 CHECK RHCS1, RHCS2, RHBA, RHBAE, RHWC
931		
932		
933	10642	TEST 62 TEST DBL (RHCS3 BIT #10) TEST S
934		
935	10644	CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5); SET UP FOR A NINE WORD READ REVERSE FROM AN EVEN WORD BC DO A NINE WORD READ REVERSE FROM AN EVEN WORD BOUNDARY THIS SHOULD SET RHCS3 BIT #10 DBL CHECK RHCS3 CHECK RHCS1, RHCS2, RHBA, RHBAE, RHWC IF MAG. TAPE NOT USED. THIS TEST IS NOT DONE
936		
937		
938		
939		
940		
941		
942		
943	10761	TEST 63 TEST DBL (RHCS3 BIT #10) TEST T
944		
945	10763	CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5); SET UP FOR A TEN WORD READ REVERSE FROM AN EVEN WORD BC DO A TEN WORD READ REVERSE FROM AN EVEN WORD BOUNDARY THIS SHOULD NOT SET RHCS3 BIT #10 DBL CHECK RHCS3 CHECK RHCS1, RHCS2, RHBA, RHBAE, RHWC IF MAG. TAPE NOT USED. THIS TEST IS NOT DONE
946		
947		
948		
949		
950		
951		
952		

```

953          MAINDEC-11-DERHAC-A
954          10879 TEST 64 TEST DBL (RHCS3 BIT #10) TEST U
955
956          10881          CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
957          SET UP FOR A TEN WORD READ REVERSE FROM AN ODD WORD BOUN
958          DO A TEN WORD READ REVERSE FROM AN ODD WORD BOUNDARY
959          THIS SHOULD SET RHCS3 BIT #10 DBL
960          CHECK RHCS3
961          CHECK RHCS1, RHCS2, RHBA, RHBAE, RHWC
962          IF MAG. TAPE NOT USED. THIS TEST IS NOT DONE
963
964          10998 TEST 65 TEST DBL (RHCS3 BIT #10) TEST V
965
966          11000          CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
967          SET UP FOR A ELEVEN WORD READ REVERSE FROM AN EVEN WORD
968          DO A ELEVEN WORD READ REVERSE FROM AN EVEN WORD BOUNDARY
969          THIS SHOULD SET RHCS3 BIT #10 DBL
970          CHECK RHCS3
971          CHECK RHCS1, RHCS2, RHBA, RHBAE, RHWC
972          IF MAG. TAPE NOT USED. THIS TEST IS NOT DONE
973
974          11117 TEST 66 TEST DBL (RHCS3 BIT #10) TEST W
975
976          11119          CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
977          SET UP FOR A ELEVEN WORD READ REVERSE FROM AN ODD WORD B
978          DO A ELEVEN WORD READ REVERSE FROM AN ODD WORD BOUNDARY
979
980          11122          THIS SHOULD NOT SET RHCS3 BIT #10 DBL
981          CHECK RHCS3
982          CHECK RHCS1, RHCS2, RHBA, RHBAE, RHWC
983          IF MAG. TAPE NOT USED. THIS TEST IS NOT DONE
984
985          11235 TEST 67 TEST DBL (RHCS3 BIT #10) TEST X
986
987          11237          CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
988          SET UP FOR A TEN WORD READ REVERSE FROM AN ODD WORD BOUN
989          WITH BAI IN RHCS2 BIT #3 SET
990          DO A TEN WORD READ REVERSE FROM AN ODD WORD BOUNDARY
991          THIS SHOULD NOT SET RHCS3 BIT #10 DBL
992          CHECK RHCS3
993          CHECK RHCS1, RHCS2, RHBA, RHBAE, RHWC
994          IF MAG. TAPE NOT USED. THIS TEST IS NOT DONE
995
996          11362 TEST 70 END OF ONE RH
997          THIS IS THE END OF TEST FOR ONE RH
998          IF THERE ARE MORE RH THEN THE PROGRAM
999          JUMPS TO TEST 2 FOR NEXT RH TEST
1000          END PASS IS REACHED ONLY AFTER ALL RH ARE COMPLETE
1001

```

1002  
1003  
1004  
1005  
1006  
1007  
1008  
1009  
1010  
1011  
1012  
1013  
1014  
1015  
1016  
1017  
1018  
1019  
1020  
1021  
1022  
1023  
1024  
1025  
1026  
1027  
1028  
1029  
1030  
1031  
1032  
1033  
1034  
1035  
1036  
1037  
1038  
1039  
1040  
1041

MAINDEC-11-DERHAC-A

```

*****
11400  END OF PASS ROUTINE
*****

11402  INCREMENT THE PASS NUMBER ($PASS)
      TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
      IF THERES A MONITOR GO TO IT
      IF THERE ISN'T JUMP TO TST2

*****
11439  SUBROUTINES
*****

11655  RS DATA TRANSFER SUBROUTINE
*****

11657  THIS SUBROUTINE WRITES OR READS OR DOES A
      WRITE CHECK ON CYLINDER 0 TRACK 0 SECTOR 0
      IT ASSUMES THE RH REGISTERS ARE APPROPRIATELY
      FILLED
      WHEN IT RETURNS FROM THIS SUBROUTINE TO THE
      MAIN PROGRAM IT MEANS THE TRANSFER IS COMPLETE

      THE CALL IS BY A JSR R0, @COMND COMAND

*****
11704  RPO4 DATA TRANSFER SUBROUTINE
*****

11706  THIS SUBROUTINE WRITE/READ/WRITE CHECK ON THE RPO4 CYLIN
      TRACK 0, SECTOR 0.

11708  IT ASSUMES THE RH REGISTERS ARE APPROPRIATELY FILLED.
      WHEN IT RETURNS FROM THIS SUBROUTINE TO THE MAIN
      PROGRAM IT MEANS THE COMMAND IS COMPLETE

      THE CALL IS BY A JSR R0, @COMND COMMAND.

```



1042  
1043  
1044  
1045  
1046  
1047  
1048  
1049  
1050  
1051  
1052  
1053  
1054  
1055  
1056  
1057  
1058  
1059  
1060  
1061  
1062  
1063  
1064  
1065  
1066  
1067  
1068  
1069  
1070  
1071  
1072  
1073  
1074  
1075  
1076  
1077  
1078  
1079  
1080  
1081  
1082  
1083  
1084  
1085  
1086  
1087  
1088  
1089

MAINDEC-11-DERHAC-A

```
*****  
11758 TMO2 DATA TRANSFER SUBROUTINE  
*****  
11760 THIS SUBROUTINE WRITES/READS/WRITE CHECKS ON THE TMO2-TL  
ON THE FIRST BLOCK FROM BEGINNING OF TAPE (BOT)  
11762 IT ASSUMES THE RH REGISTERS ARE APPROPRIATELY FILLED.  
WHEN IT RETURNS FROM THIS SUBROUTINE TO THE  
MAIN PROGRAM IT MEANS THE COMMAND IS COMPLETE.  
THE CALL IS  
JSR RO,2COMND  
11915 THIS ROUTINE WILL ALLOW THE CHANGE OF THE BASE  
ADDRESS FROM 176700 TO ANY TYPED VALUE  
11982 *****  
11995 *****  
SCOPE HANDLER ROUTINE  
*****  
11987 THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT  
AND LOAD THE TEST NUMBER($STNM) INTO THE DISPLAY REG.(DISPLAY<7  
AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>  
THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:  
SW14=1 LOOP ON TEST  
SW11=1 INHIBIT ITERATIONS  
SW09=1 LOOP ON ERROR  
SW08=1 LOOP ON TEST IN SWR<7:0>  
CALL SCOPE ;:SCOPE=IOT  
12051 *****  
CONVERT BINARY TO DECIMAL AND TYPE ROUTINE  
*****  
12053 THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIG  
SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER  
NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE T  
BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS  
REPLACED WITH SPACES.  
CALL:  
MOV NUM,-(SP) ;:PUT THE BINARY NUMBER ON THE S  
TYPDS ;:GO TO THE ROUTINE
```

1090  
1091  
1092  
1093  
1094  
1095  
1096  
1097  
1098  
1099  
1100  
1101  
1102  
1103  
1104  
1105  
1106  
1107  
1108  
1109  
1110  
1111  
1112  
1113  
1114  
1115  
1116  
1117  
1118  
1119  
1120  
1121  
1122  
1123  
1124  
1125  
1126  
1127  
1128  
1129  
1130  
1131  
1132  
1133  
1134  
1135  
1136  
1137  
1138  
1139  
1140  
1141  
1142

MAINDEC-11-DERHAC-A

12119

\*\*\*\*\*  
TYPE ROUTINE  
\*\*\*\*\*

12121 ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 B  
THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE  
NOTE1: \$NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CH  
NOTE2: \$FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED  
NOTE3: \$FILLC CONTAINS THE CHARACTER TO FILL AFTER.

CALL:  
1) USING A TRAP INSTRUCTION  
TYPE ,MESADR ;;MESADR IS FIRST ADDRESS OF AN  
OR  
TYPE  
MESADR

2) USING A JSR INSTRUCTION  
MOV PS,-(SP) ;;PUSH PROCESSOR STATUS WORD ON  
JSR PC,\$TYPE ;;CALL TYPE ROUTINE  
MESADDR ;;FIRST ADDRESS OF MESSAGE

12192

\*\*\*\*\*  
TTY INPUT ROUTINE  
\*\*\*\*\*

12201 TK INITIALIZE ROUTINE  
THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE  
SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT  
CALL:  
JSR PC,\$TKINT  
RETURN

12218 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" (^C) \$TKINT IS CALLED AND  
UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (OPE

12245 THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY  
CALL:  
RDCHR ;;GET A CHARACTER FROM THE QUEUE  
RETURN HERE ;;CHARACTER IS ON THE STACK  
;;WITH PARITY BIT STRIPPED OFF

12266 THIS ROUTINE WILL INPUT A STRING FROM THE TTY  
CALL:  
ROLIN ;;INPUT A STRING FROM THE TTY  
RETURN HERE ;;ADDRESS OF FIRST CHARACTER WIL  
;;TERMINATOR WILL BE A BYTE OF A

1143  
1144  
1145  
1146  
1147  
1148  
1149  
1150  
1151  
1152  
1153  
1154  
1155  
1156  
1157  
1158  
1159  
1160  
1161  
1162  
1163  
1164  
1165  
1166  
1167  
1168  
1169  
1170  
1171  
1172  
1173  
1174  
1175  
1176  
1177  
1178  
1179  
1180  
1181  
1182  
1183  
1184  
1185  
1186  
1187  
1188  
1189  
1190  
1191  
1192  
1193

MAINDEC-11-DERHAC-A

12303 \*\*\*\*\*  
READ AN OCTAL NUMBER FROM THE TTY  
\*\*\*\*\*

12305 THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND  
CHANGE IT TO BINARY.  
THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL  
OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPE  
FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUS  
THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RE  
CALL:  
RDOCT ;:READ AN OCTAL NUMBER  
RETURN HERE ;:LOW ORDER BITS ARE ON TOP OF T  
;:HIGH ORDER BITS ARE IN \$HIOCT

12357 \*\*\*\*\*  
ERROR HANDLER ROUTINE  
\*\*\*\*\*

12359 THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT.  
SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL  
AND GO TO \$ERRTYP ON ERROR  
THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:  
SW15=1 HALT ON ERROR  
HALT CAN OCCUR BEFORE AND AFTER THE ERROR TYPEOU  
SW13=1 INHIBIT ERROR TYPEOUTS  
SW10=1 BELL ON ERROR  
SW09=1 LOOP ON ERROR  
CALL ERROR N ;:ERROR=EMT AND N=ERROR ITEM NUMBER

12401 \*\*\*\*\*  
ERROR MESSAGE TYPEOUT ROUTINE  
\*\*\*\*\*

12402 THIS ROUTINE USES THE "ITEM CONTROL BYTE" (\$ITEMB) TO DETERMINE  
ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE  
AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.  
IT IS A COPY OF THE \$ERRTYP SUBROUTINE FROM SYSMAC.  
WITH ONLY MINOR CHANGES  
FIRST IF SWITCH 6 IS SET AND SWITCH 8 RESET THEN  
ALL REGISTER CONTENTS WILL BE TYPED BEFOR REPORTING THE ERROR

12409 SECOND IF THE CURRENT ERROR HAS THE SAME ITEM NUMBER  
AS THE PREVIOUS ERROR THEN ONLY THE DATA WILL BE TYPED  
AND NOT THE ERROR MESSAGE AND HEADER.

12627 \*\*\*\*\*  
\*\*\*\*\*

1194  
1195  
1196  
1197  
1198  
1199  
1200  
1201  
1202  
1203  
1204  
1205  
1206  
1207  
1208  
1209  
1210  
1211  
1212  
1213  
1214  
1215  
1216  
1217  
1218  
1219  
1220  
1221  
1222  
1223  
1224  
1225  
1226  
1227  
1228  
1229  
1230  
1231  
1232  
1233  
1234  
1235  
1236

MAINDEC-11-DERHAC-A

12631

\*\*\*\*\*  
BINARY TO OCTAL (ASCII) AND TYPE  
\*\*\*\*\*

12633 THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIG  
OCTAL (ASCII) NUMBER AND TYPE IT.  
\$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS  
CALL:

MOV NUM, -(SP) ; NUMBER TO BE TYPED  
TYPOS ; CALL FOR TYPEOUT  
.BYTE N ; N=1 TO 6 FOR NUMBER OF DIGITS  
.BYTE M ; M=1 OR 0  
; ; 1=TYPE LEADING ZEROS  
; ; 0=SUPPRESS LEADING ZER

\$TYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE  
\$TYPOS OR \$TYPOC

CALL:  
MOV NUM, -(SP) ; NUMBER TO BE TYPED  
TYPON ; CALL FOR TYPEOUT

\$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER

CALL:  
MOV NUM, -(SP) ; NUMBER TO BE TYPED  
TYPOC ; CALL FOR TYPEOUT

12709

\*\*\*\*\*  
TRAP DECODER  
\*\*\*\*\*

12711 THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTIO  
AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDR  
OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL  
GO TO THAT ROUTINE.

12724

\*\*\*\*\*  
TRAP TABLE  
\*\*\*\*\*

12726 THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLE  
BY THE "TRAP" INSTRUCTION.

MAINDEC-11-DEHAC-A  
DEHAC.P11

MACY11 27(732) 09-SEP-76 07:55 PAGE 27

1237  
1238  
1239  
1240  
1241  
1242  
1243  
1244  
1245

MAINDEC-11-DEHAC-A

12746

POWER DOWN AND UP ROUTINES

12786

\*\*\*\*\*  
\*\*\*\*\*

\*\*\*\*\*  
\*\*\*\*\*

%

```

1246 .TITLE MAINDEC-11-DERHAC-A
1247 ;*COPYRIGHT (C) 1975-1976
1248 ;*DIGITAL EQUIPMENT CORP.
1249 ;*MAYNARD, MASS. 01754
1250 ;*
1251 ;*
1252 ;*THIS PROGRAM WAS ASSEMBLED SING THE PDP-11 MAINDEC SYSMAC
1253 ;*PACKAGE (MAINDEC-11-DZQAC-C1),MAR 24, 1976.
1254 ;*
1255
1256 .SBTTL OPERATIONAL SWITCH SETTINGS
1257 ;*
1258 ;* SWITCH USE
1259 ;* -----
1260 ;* 15 HALT ON ERROR
1261 ;* 14 LOOP ON TEST
1262 ;* 13 INHIBIT ERROR TYPEOUTS
1263 ;* 11 INHIBIT ITERATIONS
1264 ;* 10 BELL ON ERROR
1265 ;* 9 LOOP ON ERROR
1266 ;* 8 LOOP ON TEST IN SWR<7:0>
1267 ;* 7 STOP FURTHER COMPARES IF SW08 IS LOW
1268 ;* 6 TYPE ALL REG. WITH ERROR IF SW8 LOW
1269 .SBTTL BASIC DEFINITIONS
1270
1271 ;*INITIAL ADDRESS OF THE STACK POINTER *** 1000 ***
1272 001000 STACK= 1000
1273 .EQUIV EMT,ERROR ;;BASIC DEFINITION OF ERROR CALL
1274 .EQUIV IOT,SCOPE ;;BASIC DEFINITION OF SCOPE CALL
1275
1276 ;*MISCELLANEOUS DEFINITIONS
1277 000011 HT= 11 ;;CODE FOR HORIZONTAL TAB
1278 000012 LF= 12 ;;CODE FOR LINE FEED
1279 000015 CR= 15 ;;CODE FOR CARRIAGE RETURN
1280 000200 CRLF= 200 ;;CODE FOR CARRIAGE RETURN-LINE FEED
1281 177776 PS= 177776 ;;PROCESSOR STATUS WORD
1282 .EQUIV PS,PSW
1283 177774 STKLMT= 177774 ;;STACK LIMIT REGISTER
1284 177772 PIRQ= 177772 ;;PROGRAM INTERRUPT REQUEST REGISTER
1285 177570 DSWR= 177570 ;;HARDWARE SWITCH REGISTER
1286 177570 DDISP= 177570 ;;HARDWARE DISPLAY REGISTER
1287
1288 ;*GENERAL PURPOSE REGISTER DEFINITIONS
1289 000000 R0= %0 ;;GENERAL REGISTER
1290 000001 R1= %1 ;;GENERAL REGISTER
1291 000002 R2= %2 ;;GENERAL REGISTER
1292 000003 R3= %3 ;;GENERAL REGISTER
1293 000004 R4= %4 ;;GENERAL REGISTER
1294 000005 R5= %5 ;;GENERAL REGISTER
1295 000006 R6= %6 ;;GENERAL REGISTER
1296 000007 R7= %7 ;;GENERAL REGISTER
1297 .EQUIV R6,SP ;;STACK POINTER
1298 .EQUIV R7,PC ;;PROGRAM COUNTER
1299
1300 ;*PRIORITY LEVEL DEFINITIONS
1301 000000 PRO= 0 ;;PRIORITY LEVEL 0

```

1302	000040	PR1=	40	:: PRIORITY LEVEL 1
1303	000100	PR2=	100	:: PRIORITY LEVEL 2
1304	000140	PR3=	140	:: PRIORITY LEVEL 3
1305	000200	PR4=	200	:: PRIORITY LEVEL 4
1306	000240	PR5=	240	:: PRIORITY LEVEL 5
1307	000300	PR6=	300	:: PRIORITY LEVEL 6
1308	000340	PR7=	340	:: PRIORITY LEVEL 7

.\*"SWITCH REGISTER" SWITCH DEFINITIONS

1310		SW15=	100000
1311	100000	SW14=	40000
1312	040000	SW13=	20000
1313	020000	SW12=	10000
1314	010000	SW11=	4000
1315	004000	SW10=	2000
1316	002000	SW09=	1000
1317	001000	SW08=	400
1318	000400	SW07=	200
1319	000200	SW06=	100
1320	000100	SW05=	40
1321	000040	SW04=	20
1322	000020	SW03=	10
1323	000010	SW02=	4
1324	000004	SW01=	2
1325	000002	SW00=	1
1326	000001	.EQUIV	SW09, SW9
1327		.EQUIV	SW08, SW8
1328		.EQUIV	SW07, SW7
1329		.EQUIV	SW06, SW6
1330		.EQUIV	SW05, SW5
1331		.EQUIV	SW04, SW4
1332		.EQUIV	SW03, SW3
1333		.EQUIV	SW02, SW2
1334		.EQUIV	SW01, SW1
1335		.EQUIV	SW00, SW0

.\*DATA BIT DEFINITIONS (BIT00 TO BIT15)

1338		BIT15=	100000
1339	100000	BIT14=	40000
1340	040000	BIT13=	20000
1341	020000	BIT12=	10000
1342	010000	BIT11=	4000
1343	004000	BIT10=	2000
1344	002000	BIT09=	1000
1345	001000	BIT08=	400
1346	000400	BIT07=	200
1347	000200	BIT06=	100
1348	000100	BIT05=	40
1349	000040	BIT04=	20
1350	000020	BIT03=	10
1351	000010	BIT02=	4
1352	000004	BIT01=	2
1353	000002	BIT00=	1
1354	000001	.EQUIV	BIT09, BIT9
1355		.EQUIV	BIT08, BIT8
1356		.EQUIV	BIT07, BIT7
1357			

```

1358 .EQUIV BIT06,BIT6
1359 .EQUIV BIT05,BIT5
1360 .EQUIV BIT04,BIT4
1361 .EQUIV BIT03,BIT3
1362 .EQUIV BIT02,BIT2
1363 .EQUIV BIT01,BIT1
1364 .EQUIV BIT00,BIT0
1365
1366 ;*BASIC "CPU" TRAP VECTOR ADDRESSES
1367 000004 ERRVEC= 4 ;: TIME OUT AND OTHER ERRORS
1368 000010 RESVEC= 10 ;: RESERVED AND ILLEGAL INSTRUCTIONS
1369 000014 TBITVEC=14 ;: "T" BIT
1370 000014 TRIVEC= 14 ;: TRACE TRAP
1371 000014 BPTVEC= 14 ;: BREAKPOINT TRAP (BPT)
1372 000020 IOTVEC= 20 ;: INPUT/OUTPUT TRAP (IOT) **SCOPE**
1373 000024 PWRVEC= 24 ;: POWER FAIL
1374 000030 EMTVEC= 30 ;: EMULATOR TRAP (EMT) **ERROR**
1375 000034 TRAPVEC=34 ;: "TRAP" TRAP
1376 000060 TKVEC= 60 ;: TTY KEYBOARD VECTOR
1377 000064 TPVEC= 64 ;: TTY PRINTER VECTOR
1378 000240 PIRQVEC=240 ;: PROGRAM INTERRUPT REQUEST VECTOR
1379
1380 .SBTTL TRAP CATCHER
1381
1382 000000 .=0
1383 ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
1384 ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
1385 ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
1386 000174 .=174
1387 000174 000000 DISPREG: .WORD 0 ;: SOFTWARE DISPLAY REGISTER
1388 000176 000000 SWREG: .WORD 0 ;: SOFTWARE SWITCH REGISTER
1389 .SBTTL STARTING ADDRESS(ES)
1390 000200 000137 005522 JMP @#BEGIN ;: JUMP TO STARTING ADDRESS OF PROGRAM
1391 000046 .=46
1392 000046 037554 $ENDAD
1393 000052 .=52
1394 000052 000000 000000
1395 000210 .=210
1396 000210 000137 005532 JMP @#BEGIN2 ;: JUMP SELECT TEST
1397 ;*STARTING ADDRESS 200 FOR NORMAL STARTS
1398 ;*THIS WILL TEST ALL RPO4'S ON THE SYSTEM A SINGLE DRIVE AT A TIME
1399 ;*
1400 ;*STARTING ADDRESS 210 WILL TEST ONLY ONE SPECIFIED DRIVE
1401 ;*
1402 ;*STARTING ADDRESS 220 WILL JUMP OVER THE TESTS REQUIRING AN OPERATOR
1403 ;*AT THE DRIVE
1404 .SBTTL MEMORY MANAGEMENT DEFINITIONS
1405
1406 ;*KT11 VECTOR ADDRESS
1407
1408 000250 MMVEC= 250
1409
1410 ;*KT11 STATUS REGISTER ADDRESSES
1411
1412 177572 SRO= 177572
1413 177574 SRI= 177574

```



```

1414      177576      SR2=   177576
1415      172516      SR3=   172516
1416
1417      ;*KERNEL "I" PAGE DESCRIPTOR REGISTERS
1418
1419      172300      KIPDR0= 172300
1420      172302      KIPDR1= 172302
1421      172304      KIPDR2= 172304
1422      172306      KIPDR3= 172306
1423      172310      KIPDR4= 172310
1424      172312      KIPDR5= 172312
1425      172314      KIPDR6= 172314
1426      172316      KIPDR7= 172316
1427
1428      ;*KERNEL "I" PAGE ADDRESS REGISTERS
1429
1430      172340      KIPAR0= 172340
1431      172342      KIPAR1= 172342
1432      172344      KIPAR2= 172344
1433      172346      KIPAR3= 172346
1434      172350      KIPAR4= 172350
1435      172352      KIPAR5= 172352
1436      172354      KIPAR6= 172354
1437      172356      KIPAR7= 172356
1438
1439      ;*****
1440      001110      .=1110

```

```

1441
1442
1443
1444
1445
1446
1447      001100
1448      001100      000000
1449      001100      000000
1450      001102      000
1451      001103      000
1452      001104      000000
1453      001106      000000
1454      001110      000000
1455      001112      000000
1456      001114      000
1457      001115      001
1458      001116      000000
1459      001120      000000
1460      001122      000000
1461      001124      000000
1462      001126      000000
1463      001130      000000
1464      001132      000000
1465      001134      000
1466      001135      000
1467      001136      000000
1468      001140      177570
1469      001142      177570
1470      001144      177560
1471      001146      177562
1472      001150      177564
1473      001152      177566
1474      001154      000
1475      001155      002
1476      001156      012
1477      001157      000
1478      001160      000000
1479
1480      001162      000000
1481      001164      000000
1482      001166      000000
1483      001170      000000
1484      001172      000000
1485      001174      000000
1486      001176      000000
1487      001200      000000
1488      001202      000000
1489      001204      000000
1490      001206      000000
1491      001210      000000
1492      001212      000000
1493      001214      000000
1494      001216      177607      000377
1495      001222      077
1496      001223      015

```

.SBTTL COMMON TAGS

```

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

```

```

.=1100
$CMTAG:      .WORD      0      ;; START OF COMMON TAGS
$PASS:      .WORD      0      ;; CONTAINS PASS COUNT
$STNM:      .BYTE      0      ;; CONTAINS THE TEST NUMBER
$ERFLG:      .BYTE      0      ;; CONTAINS ERROR FLAG
$ICNT:      .WORD      0      ;; CONTAINS SUBTEST ITERATION COUNT
$LPADR:      .WORD      0      ;; CONTAINS SCOPE LOOP ADDRESS
$LPERR:      .WORD      0      ;; CONTAINS SCOPE RETURN FOR ERRORS
$ERTTL:      .WORD      0      ;; CONTAINS TOTAL ERRORS DETECTED
$ITEMB:      .BYTE      0      ;; CONTAINS ITEM CONTROL BYTE
$ERMAX:      .BYTE      1      ;; CONTAINS MAX. ERRORS PER TEST
$ERRPC:      .WORD      0      ;; CONTAINS PC OF LAST ERROR INSTRUCTION
$GDADR:      .WORD      0      ;; CONTAINS ADDRESS OF 'GOOD' DATA
$BDADR:      .WORD      0      ;; CONTAINS ADDRESS OF 'BAD' DATA
$GDDAT:      .WORD      0      ;; CONTAINS 'GOOD' DATA
$BDDAT:      .WORD      0      ;; CONTAINS 'BAD' DATA
                .WORD      0      ;; RESERVED--NOT TO BE USED
$AUTOB:      .BYTE      0      ;; AUTOMATIC MODE INDICATOR
$INTAG:      .BYTE      0      ;; INTERRUPT MODE INDICATOR
                .WORD      0
$SWR:      .WORD      DSWR      ;; ADDRESS OF SWITCH REGISTER
$DISPLAY:      .WORD      DDISP      ;; ADDRESS OF DISPLAY REGISTER
$TKS:      177560      ;; TTY KBD STATUS
$TKB:      177562      ;; TTY KBD BUFFER
$TPS:      177564      ;; TTY PRINTER STATUS REG. ADDRESS
$TPB:      177566      ;; TTY PRINTER BUFFER REG. ADDRESS
$NULL:      .BYTE      0      ;; CONTAINS NULL CHARACTER FOR FILLS
$FILLS:      .BYTE      2      ;; CONTAINS # OF FILLER CHARACTERS REQUIRED
$FILLC:      .BYTE      12      ;; INSERT FILL CHARS. AFTER A "LINE FEED"
$TPFLG:      .BYTE      0      ;; "TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
$REGAD:      .WORD      0      ;; CONTAINS THE ADDRESS FROM WHICH ($REGO) WAS OBTAINED
$REGO:      .WORD      0      ;; CONTAINS ((REGAD)+0)
$REG1:      .WORD      0      ;; CONTAINS ((REGAD)+2)
$REG2:      .WORD      0      ;; CONTAINS ((REGAD)+4)
$REG3:      .WORD      0      ;; CONTAINS ((REGAD)+6)
$REG4:      .WORD      0      ;; CONTAINS ((REGAD)+8)
$REG5:      .WORD      0      ;; CONTAINS ((REGAD)+10)
$STMP0:      .WORD      0      ;; USER DEFINED
$STMP1:      .WORD      0      ;; USER DEFINED
$STMP2:      .WORD      0      ;; USER DEFINED
$STMP3:      .WORD      0      ;; USER DEFINED
$STMP4:      .WORD      0      ;; USER DEFINED
$STMP5:      .WORD      0      ;; USER DEFINED
$TIMES:      0      ;; MAX. NUMBER OF ITERATIONS
$ESCAPE:      0      ;; ESCAPE ON ERROR ADDRESS
$BELL:      .ASCII <207><377><377>      ;; CODE FOR BELL
$QUES:      .ASCII '?'      ;; QUESTION MARK
$CRLF:      .ASCII <15>      ;; CARRIAGE RETURN

```

H03

MAINDEC-11-DERHAC-A MACY11 27(732) 09-SEP-76 07:55 PAGE 33  
DERHAC.P11 COMMON TAGS

:497 001224 000012  
1498

SLF: .ASCIZ <12> ;;LINE FEED  
:\*\*\*\*\*

1499  
1500  
1501  
1502  
1503  
1504  
1505  
1506  
1507  
1508  
1509  
1510  
1511  
1512  
1513  
1514  
1515  
1516  
1517  
1518  
1519  
1520  
1521  
1522  
1523  
1524  
1525  
1526  
1527  
1528  
1529  
1530  
1531  
1532  
1533  
1534  
1535  
1536  
1537  
1538  
1539  
1540  
1541  
1542  
1543  
1544  
1545  
1546  
1547  
1548  
1549  
1550  
1551  
1552  
1553  
1554

.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

001226

\$ERRTB:

\*\*\*\*\*

:ITEM 1

001226 046712

EM1

:THE RH BASE ADDRESS WAS 772040  
:INDICATING A R504 OR R503  
:BUT THE DRIVE TYPE DID NOT  
:CONTAIN 0,1,2,3 OR 4  
:PC  
:RHDT  
:\$ERRPC,\$BDDAT,0  
:0

001230 065133

DH1

:PC  
:RHDT  
:\$ERRPC,\$BDDAT,0  
:0

001232 066566

DT1

001234 066750

DF1

:ITEM 2

001236 047064

EM2

:THE RH BASE ADDRESS WAS 776700  
:INDICATING AN RPO4, RPO5, OR RPO6  
:BUT THE DRIVE TYPE DID NOT  
:CONTAIN 20020 24020, 20021, 24021,  
:20022, OR 24022

001240 065133

DH1

:PC  
:RHDT  
:\$ERRPC,\$BDDAT,0  
:0

001242 066566

DT1

001244 066750

DF1

:ITEM 3

001246 047266

EM3

:THE RH BASE ADDRESS WAS 772440  
:INDICATING A TM02, BUT THE  
:DRIVE TYPE DID NOT CONTAIN  
:142010 INDICATING NO  
:TM02  
:PC  
:RHDT  
:\$ERRPC,\$BDDAT,0  
:0,0

001250 065133

DH1

001252 066566

DT1

001254 066750

DF1

:ITEM 4

1555	001256	047422	EM4	: THE RH BASE ADDRESS WAS 776300
1556				: INDICATING MORE THAN ONE
1557				: RH DEVICE. BUT THE DRIVE
1558				: TYPE DID NOT CONTAIN
1559				: 0, 1, 2, 3, 4, 20020, 24020, 20021, 24021.
1560				: 20022, 24022, OR 142010 INDICATING NO
1561				: RH DEVICE IS PRESENT
1562	001260	065133	DH1	: PC
1563				: RHDT
1564	001262	066566	DT1	: \$ERRPC, \$BDDAT, 0
1565	001264	066750	DF1	: 0, 0
1566				
1567			: ITEM 5	
1568	001266	047721	EMS	: THE UNIBUS TIMED OUT
1569				: FOR EACH OF THE FOLLOWING
1570				: ADDRESSES, INDICATING
1571				: NO RH ON THE SYSTEM
1572				: SO ABORT PROGRAM
1573				: PC
1574	001270	065142	DH5	: UNIBUS ADDRESSES ON WHICH
1575				: TIME OUT OCCURRED
1576				: \$ERRPC, RSCS1, RPCS1, TMCS1, MIXCS1
1577	001272	066574	DT5	: 0, 0, 0, 0, 0
1578	001274	066752	DF5	
1579				
1580			: ITEM 6	
1581	001276	050074	EM6	: AFTER AN RH CLEAR (BIT #5
1582				: IN RHCS2) RHCS2 DOES NOT
1583				: HAVE ONLY IR AND UNIT
1584				: NUMBER
1585	001300	065216	DH6	: PC
1586				: TEST NO
1587				: RHCS2 GOOD
1588				: RHCS2 BAD
1589	001302	066610	DT6	: \$ERRPC, TSTNM, \$GDDAT, \$BDDAT
1590	001304	066757	DF6	: 0, 0, 0, 0
1591				
1592			: ITEM 7	
1593	001306	050212	EM7	: AFTER CLEARING THE RH AND
1594				: WRITING ONE WORD INTO
1595				: RHDB AND READING IT
1596				: BACK CAUSED RHDB TO
1597				: HAVE WRONG VALUE GIVEN IN BAD RHDB
1598	001310	065261	DH7	: PC
1599				: TEST NO
1600				: RHDB GOOD
1601				: RHDB BAD
1602	001312	066610	DT6	: \$ERRPC, TSTNM, \$GDDAT, \$BDDAT
1603	001314	066757	DF6	: 0, 0, 0, 0
1604				
1605			: ITEM 10	
1606	001316	050404	EM10	: AFTER CLEARING THE RH AND
1607				: WRITING ONE WORD INTO
1608				: RHDB AND READING IT
1609				: BACK, CAUSED RHCS2 TO
1610				: HAVE WRONG DATA



1611					: GIVEN IN BAD RHCS2
1612	001320	065216		DH6	: PC
1613					: TEST NUMBER
1614					: RHCS2 GOOD
1615					: RHCS2 BAD
1616	001322	066610		DT6	: \$ERRPC, TSTNM, \$GDDAT, \$BDDAT
1617	001324	066757		DF6	: 0,0,0,0
1618					
1619					: ITEM 11
1620	001326	050577		EM11	: AFTER CLEARING THE RH AND
1621					: WRITING ONE WORD INTO
1622					: RHDB AND READING IT
1623					: BACK, CAUSED RHCS1 TO
1624					: HAVE WRONG DATA
1625					: GIVEN IN BAD RHCS1
1626	001330	065321		DH11	: PC
1627					: TEST NUMBER
1628					: RHCS1 GOOD
1629					: RHCS1 BAD
1630	001332	066610		DT6	: \$ERRPC, TSTNM, \$GDDAT, \$BDDAT
1631	001334	066757		DF6	: 0,0,0,0
1632					
1633					: ITEM 12
1634	001336	050772		EM12	: AFTER CLEARING THE RH AND
1635					: WRITING ONE WORD INTO
1636					: RHDB AND READING IT
1637					: BACK, CAUSED RHCS3 TO
1638					: HAVE WRONG DATA
1639					: GIVEN IN BAD RHCS3
1640	001340	065363		DH12	: PC
1641					: TEST NUMBER
1642					: RHCS3 GOOD
1643					: RHCS3 BAD
1644	001342	066610		DT6	: \$ERRPC, TSTNM, \$GDDAT, \$BDDAT
1645	001344	066757		DF6	: 0,0,0,0
1646					
1647					: ITEM 13
1648	001346	051165		EM13	: AFTER CLEARING THE RH AND
1649					: WRITING ONE WORD INTO
1650					: RHDB AND READING IT
1651					: BACK, CAUSED RHBA TO
1652					: HAVE WRONG DATA
1653					: GIVEN IN BAD RHBA
1654	001350	065425		DH13	: PC
1655					: TEST NUMBER
1656					: RHBA GOOD
1657					: RHBA BAD
1658	001352	066610		DT6	: \$ERRPC, TSTNM, \$GDDAT, \$BDDAT
1659	001354	066757		DF6	: 0,0,0,0
1660					
1661					: ITEM 14
1662	001356	051357		EM14	: AFTER CLEARING THE RH AND
1663					: WRITING ONE WORD INTO
1664					: RHDB AND READING IT
1665					: BACK, CAUSED RHBAE TO
1666					: HAVE WRONG DATA

1667					: GIVEN IN BAD RHBAE
1668	001360	065465		DH14	: PC
1669					: TEST NUMBER
1670					: RHBAE GOOD
1671					: RHBAE BAD
1672	001362	066610		DT6	: \$ERRPC, TSTNM, \$GDDAT, \$BDDAT
1673	001364	066757		DF6	: 0,0,0,0
1674					
1675					: ITEM 15
1676	001366	051552		EM15	: AFTER CLEARING THE RH AND
1677					: WRITING ONE WORD INTO
1678					: RHDB AND READING IT
1679					: BACK, CAUSED RHWC TO
1680					: HAVE WRONG DATA
1681					: GIVEN IN BAD RHWC
1682	001370	065527		DH15	: PC
1683					: TEST NUMBER
1684					: RHWC GOOD
1685					: RHWC BAD
1686	001372	066610		DT6	: \$ERRPC, TSTNM, \$GDDAT, \$BDDAT
1687	001374	066757		DF6	: 0,0,0,0
1688					
1689					: ITEM 16
1690	001376	051744		EM16	: AFTER CLEARING THE RH AND
1691					: WRITING ONE WORD INTO
1692					: RHDB
1693					: CAUSED RHCS2 TO
1694					: HAVE WRONG DATA
1695					: GIVEN IN BAD RHCS2
1696	001400	065216		DH6	: PC
1697					: TEST NUMBER
1698					: RHCS2 GOOD
1699					: RHCS2 BAD
1700	001402	066610		DT6	: \$ERRPC, TSTNM, \$GDDAT, \$BDDAT
1701	001404	066757		DF6	: 0,0,0,0
1702					
1703					: ITEM 17
1704	001406	052137		EM17	: AFTER CLEARING THE RH AND
1705					: WRITING TWO WORD INTO
1706					: RHDB AND READING IT
1707					: BACK, CAUSED RHDB TO
1708					: HAVE WRONG DATA
1709					: GIVEN IN BAD RHDB
1710	001410	065261		DH7	: PC
1711					: TEST NUMBER
1712					: RHDB GOOD
1713					: RHDB BAD
1714	001412	066610		DT6	: \$ERRPC, TSTNM, \$GDDAT, \$BDDAT
1715	001414	066757		DF6	: 0,0,0,0
1716					
1717					: ITEM 20
1718	001416	052331		EM20	: AFTER CLEARING THE RH AND
1719					: WRITING TWO WORD INTO
1720					: RHDB AND READING ONE
1721					: BACK, CAUSED RHCS2 TO
1722					: HAVE WRONG DATA

1723				: GIVEN IN BAD RHCS2
1724	001420	065216	DH6	: PC
1725				: TEST NUMBER
1726				: RHCS2 GOOD
1727				: RHCS2 BAD
1728	001422	066610	DT6	: \$ERRPC, TSTNM, \$GDDAT, \$BDDAT
1729	001424	066757	DF6	: 0,0,0,0
1730				
1731			: ITEM 21	
1732	001426	052525	EM21	: AFTER CLEARING THE RH AND
1733				: WRITING TWO WORD INTO
1734				: RHDB AND READING IT
1735				: BACK TWICE, CAUSED RHDB TO
1736				: HAVE WRONG DATA
1737				: GIVEN IN BAD RHDB
1738	001430	065261	DH7	: PC
1739				: TEST NUMBER
1740				: RHDB GOOD
1741				: RHDB BAD
1742	001432	066610	DT6	: \$ERRPC, TSTNM, \$GDDAT, \$BDDAT
1743	001434	066757	DF6	: 0,0,0,0
1744				
1745			: ITEM 22	
1746	001436	052725	EM22	: AFTER CLEARING THE RH AND
1747				: WRITING TWO WORD INTO
1748				: RHDB AND READING IT
1749				: BACK TWICE, CAUSED RHCS2 TO
1750				: HAVE WRONG DATA
1751				: GIVEN IN BAD RHCS2
1752	001440	065216	DH6	: PC
1753				: TEST NUMBER
1754				: RHCS2 GOOD
1755				: RHCS2 BAD
1756	001442	066610	DT6	: \$ERRPC, TSTNM, \$GDDAT, \$BDDAT
1757	001444	066757	DF6	: 0,0,0,0
1758				
1759			: ITEM 23	
1760	001446	053126	EM23	: AFTER CLEARING THE RH AND
1761				: WRITING EIGHT WORD INTO
1762				: RHDB
1763				: CAUSED RHCS2 TO
1764				: HAVE WRONG DATA
1765				: GIVEN IN BAD RHCS2
1766	001450	065216	DH6	: PC
1767				: TEST NUMBER
1768				: RHCS2 GOOD
1769				: RHCS2 BAD
1770	001452	066610	DT6	: \$ERRPC, TSTNM, \$GDDAT, \$BDDAT
1771	001454	066757	DF6	: 0,0,0,0
1772				
1773			: ITEM 24	
1774	001456	053323	EM24	: THE RH WAS CLEARED
1775				: AND A PATTERN OF 8 WORDS
1776				: WERE WRITTEN INTO RHDB
1777				: READING RHDB FOR THE
1778				: "N" TH. TIME GAVE WRONG



1779				: VALUE IN RHDB
1780				: N IS GIVEN IN "WORD NO"
1781	001460	065567	DH24	: PC
1782				: TEST NO
1783				: WORD NO
1784				: RHDB GOOD
1785				: RHDB BAD
1786	001462	066622	DT24	: \$ERRPC, TSTNM, \$GDDAT, \$BDDAT
1787	001464	066763	DF24	: 0,0,0,0
1788				
1789				: ITEM 25
1790	001466	053553	EM25	: THE RH WAS CLEARED AND
1791				: A PATTERN OF 8 WORDS WERE
1792				: WRITTEN INTO RHDB
1793				: AFTER READING ALL 8 WORDS
1794				: FOLLOWING REGISTER CONTAINED WRONG
1795				: VALUE
1796	001470	065216	DH6	: PC
1797				: TEST NO
1798				: RHCS2 GOOD
1799				: RHCS2 BAD
1800	001472	066610	DT6	: \$ERRPC, TSTNM, \$GDDAT, \$BDDAT
1801	001474	066757	DF6	: 0,0,0,0
1802				
1803				: ITEM 26
1804	001476	053553	EM25	
1805	001500	065321	DH11	
1806	001502	066610	DT6	
1807	001504	066757	DF6	: 0,0
1808				
1809				: ITEM 27
1810	001506	053553	EM25	
1811	001510	065363	DH12	: PC
1812				: TEST NO
1813				: RHCS3 GOOD
1814				: RHCS3 BAD
1815	001512	066610	DT6	: \$ERRPC, TSTNM, \$GDDAT, \$BDDAT
1816	001514	066757	DF6	: 0,0,0,0
1817				
1818				: ITEM 30
1819	001516	053553	EM25	
1820	001520	065425	DH13	: PC
1821				: TEST NO
1822				: RHBA GOOD
1823				: RHBA BAD
1824	001522	066610	DT6	: \$ERRPC, TSTNM, \$GDDAT, \$BDDAT
1825	001524	066757	DF6	: 0,0,0,0
1826				
1827				: ITEM 31
1828	001526	053553	EM25	
1829	001530	065465	DH14	: PC
1830				: TEST NO
1831				: RHBAE GOOD
1832				: RHBAE BAD
1833	001532	066610	DT6	: \$ERRPC, TSTNM, \$GDDAT, \$BDDAT
1834	001534	066757	DF6	: 0,0,0,0

1835					
1836			: ITEM 32		
1837	001536	053553		EM25	
1838	001540	065527		DH15	: PC
1839					: TEST NO
1840					: RHWC GOOD
1841					: RHWC BAD
1842	001542	066610		DT6	: \$ERRPC, TSTNM, \$GDDAT, \$BDDAT
1843	001544	066757		DF6	: 0,0,0,0
1844					
1845			: ITEM 33		
1846	001546	054002		EM33	: SETTING RH CLEAR BIT #5
1847					: IN RHCS2 CAUSED
1848					: ERROR REGISTER 1 TO HAVE
1849					: WRONG VALUE
1850	001550	065637		DH33	: PC
1851					: TEST NO
1852					: RHER1 GOOD
1853					: RHER1 BAD
1854	001552	066610		DT6	: \$ERRPC, TSTNM, \$GDDAT, \$BDDAT
1855	001554	066757		DF6	
1856					
1857			: ITEM 34		
1858	001556	054116		EM34	: AN RH LCLEAR WAS GIVEN
1859					: RHER1 WAS CHECKED TO HAVE ZERO
1860					: PAT (BIT #4 RHCS2) WAS SET
1861					: TO INVERT PARITY CHECKING
1862					: RHER1 WAS READ BUT DID NOT
1863					: CONTAIN WHAT IS IN RHER1 GOOD
1864	001560	065637		DH33	: PC
1865					: TEST NO
1866					: RHER1 GOOD
1867					: RHER1 BAD
1868	001562	066610		DT6	: \$ERRPC, TSTNM, \$GDDAT, \$BDDAT
1869	001564	066757		DF6	: 0,0,0,0
1870					
1871			: ITEM 35		
1872	001566	054360		EM35	: RH CLEAR WAS GIVEN
1873					: RHER1 WAS CHECKED
1874					: PAT (BIT #4 RHCS2) WAS SET
1875					: AND RHER1 WAS READ
1876					: RHCS1 SHOULD HAVE RDY
1877					: SC AND MCPE SET
1878	001570	065321		DH11	: PC
1879					: TEST NO
1880					: RHCS1 GOOD
1881					: RHCS1 BAD
1882	001572	066610		DT6	: \$ERRPC, TSTNM, \$GDDAT, \$BDDAT
1883	001574	066757		DF6	: 0,0,0,0
1884					
1885			: ITEM 36		
1886	001576	054556		EM36	: RH CLEAR WAS GIVEN
1887					: RHER1 WAS CHECKED
1888					: PAT (RHCS2-BIT #4) WAS SET
1889					: RHER1 WAS READ AND CHECKED
1890					: "1" WAS WRITTEN INTO "TRE"-RHCS1

1891				: ON CHECKING RHCS1
1892				: IT DID NOT CONTAIN SC,MCPE
1893				: AND RDY
1894	001600	065321	DH11	: PC
1895				: TEST NUMBER
1896				: RHCS1 GOOD
1897				: RHCS1 BAD
1898	001602	066610	DT6	: \$ERRPC, TSTNM, \$GDDAT, \$BDDAT
1899	001604	066757	DF6	: 0,0,0,0
1900				
1901				: ITEM 37
1902	001606	055033	EM37	: RH WAS CLEARED
1903				: RHER1 WAS CHECKED
1904				: PAT (RHCS2-BIT #4) WAS SET
1905				: RHER1 WAS READ
1906				: "1" WAS WRITTEN IN "TRE" RHCS1
1907				: AN RH CLEAR WAS TO
1908				: GIVE WHAT IS IN "GOOD"
1909				: BUT GAVE WHAT IS IN "BAC"
1910	001610	065321	DH11	: PC
1911				: TEST NO
1912				: RHCS1 GOOD
1913				: RHCS1 BAD
1914	001612	066610	DT6	: \$ERRPC, TSTNM, \$GDDAT, \$BDDAT
1915	001614	066757	DF6	: 0,0,0,0
1916				
1917				: ITEM 40
1918	001616	055033	EM37	
1919	001620	065216	DH6	
1920	001622	066610	DT6	
1921	001624	066757	DF6	
1922				
1923				: ITEM 41
1924	001626	055033	EM37	
1925	001630	065363	DH12	
1926	001632	066610	DT6	
1927	001634	066757	DF6	
1928				
1929				: ITEM 42
1930	001636	055033	EM37	
1931	001640	065425	DH13	
1932	001642	066610	DT6	
1933	001644	066757	DF6	
1934				
1935				: ITEM 43
1936	001646	055033	EM37	
1937	001650	065465	DH14	
1938	001652	066610	DT6	
1939	001654	066757	DF6	
1940				
1941				: ITEM 44
1942	001656	055033	EM37	
1943	001660	065527	DH15	
1944	001662	066610	DT6	
1945	001664	066757	DF6	
1946				

1947			: ITEM 45		
1948	001666	055335	EM45		: AFTER AN RH CLEAR
1949					: "1" WAS WRITTEN IN THE DISK
1950					: ADDRESS REGISTER (IN TAPE
1951					: DRIVES CALLED FRAME COUNT )
1952					: PAT IN RHCSI WAS SET
1953					: ON READING RHDA (IN TAPE
1954					: DRIVES RHFC ) IT DID NOT
1955					: CONTAIN "1"
1956					: RHDA=RHFC
1957	001670	065701	DH45		: PC
1958					: TEST NO
1959					: RHDA GOOD
1960					: RHDA BAD
1961	001672	066610	DT6		: \$RRPC, TSTNM, \$GDDAT, \$BDDAT
1962	001674	066757	DF6		: J, 0, 0, 0
1963					
1964			: ITEM 46		
1965	001676	055573	EM46		: AFTER AN RH CLEAR
1966					: "1" WAS WRITTEN IN THE
1967					: DISK ADDRESS REGISTER (IN
1968					: TAPE
1969					: PAT IN RHCSI WAS SET
1970					: ON READING RHDA (IN TAPE
1971					: ) FOLLOWING
1972					: REGISTER DID NOT CONTAIN
1973					: WHAT IS IN "GOOD"
1974	001700	065321	DH11		: PC
1975					: TEST NO
1976					: RHCSI GOOD
1977					: RHCSI BAD
1978	001702	066610	DT6		: \$ERRPC, TSTNM, \$GDDAT, \$BDDAT
1979	001704	066757	DF6		
1980					
1981			: ITEM 47		
1982	001706	055573	EM46		
1983	001710	065216	DH6		
1984	001712	066610	DT6		
1985	001714	066757	DF6		
1986					
1987			: ITEM 50		
1988	001716	055573	EM46		
1989	001720	065363	DH12		
1990	001722	066610	DT6		
1991	001724	066757	DF6		
1992					
1993			: ITEM 51		
1994	001726	055573	EM46		
1995	001730	065425	DH13		
1996	001732	066610	DT6		
1997	001734	066757	DF6		
1998					
1999			: ITEM 52		
2000	001736	055573	EM46		
2001	001740	065465	DH14		
2002	001742	066610	DT6		

2003	001744	066757	DF6
2004			
2005			: ITEM 53
2006	001746	055573	EM46
2007	001750	065527	DH15
2008	001752	066610	DT6
2009	001754	066757	DF6
2010			
2011			: ITEM 54
2012	001756	056074	EM54
2013			
2014			
2015			
2016			
2017			
2018	001760	065321	DH11
2019	001762	066610	DT6
2020	001764	066757	DF6
2021			
2022			: ITEM 55
2023	001766	056074	EM54
2024	001770	065465	DH14
2025	001772	066610	DT6
2026	001774	066757	DF6
2027			
2028			: ITEM 56
2029	001776	056305	EM56
2030			
2031			
2032			
2033			
2034			
2035	002000	065465	DH14
2036	002002	066610	DT6
2037	002004	066757	DF6
2038			
2039			: ITEM 57
2040	002006	056305	EM56
2041	002010	065321	DH11
2042	002012	066610	DT6
2043	002014	066757	DF6
2044			
2045			: ITEM 60
2046	002016	056543	EM60
2047			
2048			
2049			
2050			
2051			
2052	002020	065321	DH11
2053	002022	066610	DT6
2054	002024	066757	DF6
2055			
2056			: ITEM 61
2057	002026	056543	EM60
2058	002030	065465	DH14

: AN RH CLEAR (RHCS2 BIT #5)  
: WAS GIVEN  
: A16 (RHCS1 BIT #8) WAS SET  
: ON READING THE FOLLOWING  
: REGISTER IT DID NOT  
: CONTAIN WHAT IS IN "GOOD"

: AN RH CLEAR (RHCS2 BIT #5)  
: WAS GIVEN  
: A16 WAS WRITTEN IN RHCS1  
: ALL ZEROS WERE WRITTEN IN RHBAE  
: THE FOLLOWING REGISTER DID  
: NOT CONTAIN WHAT IS IN "GOOD"

: AN RH CLEAR (RHCS2 BIT #5)  
: WAS GIVEN  
: A17 (RHCS1 BIT #9) WAS SET  
: ON READING THE FOLLOWING  
: REGISTER IT DID NOT  
: CONTAIN WHAT IS IN "GOOD"

2059	002032	066610	DT6		
2060	002034	066757	DF6		
2061					
2062				: ITEM 62	
2063	002036	056754	EM62		: AN RH CLEAR (RHCS2 BIT #5)
2064					: WAS GIVEN
2065					: A17 WAS WRITTEN IN RHCS1
2066					: ALL ZEROS WERE WRITTEN IN RHBAE
2067					: THE FOLLOWING REGISTER DID
2068					: NOT CONTAIN WHAT IS IN "GOOD"
2069	002040	065465	DH14		
2070	002042	066610	DT6		
2071	002044	066757	DF6		
2072					
2073				: ITEM 63	
2074	002046	056754	EM62		
2075	002050	065321	DH11		
2076	002052	066610	DT6		
2077	002054	066757	DF6		
2078					
2079				: ITEM 64	
2080	002056	057211	EM64		: AN RH CLEAR (RHCS2 BIT #5)
2081					: WAS GIVEN
2082					: IE (RHCS1 BIT #6) WAS SET
2083					: ON READING THE FOLLOWING
2084					: REGISTER IT DID NOT
2085					: CONTAIN WHAT IS IN "GOOD"
2086	002060	065321	DH11		
2087	002062	066610	DT6		
2088	002064	066757	DF6		
2089					
2090				: ITEM 65	
2091	002066	057211	EM64		
2092	002070	065363	DH12		
2093	002072	066610	DT6		
2094	002074	066757	DF6		
2095					
2096				: ITEM 66	
2097	002076	057211	EM64		: AN RH CLEAR (RHCS2 BIT #5)
2098					: WAS GIVEN
2099					: A16 WAS WRITTEN IN RHCS1
2100					: ALL ZEROS WERE WRITTEN IN RHBAE
2101					: THE FOLLOWING REGISTER DID
2102					: NOT CONTAIN WHAT IS IN "GOOD"
2103	002100	065363	DH12		
2104	002102	066610	DT6		
2105	002104	066757	DF6		
2106					
2107				: ITEM 67	
2108	002106	057211	EM64		
2109	002110	065321	DH11		
2110	002112	066610	DT6		
2111	002114	066757	DF6		
2112					
2113				: ITEM 70	
2114	002116	057660	EM70		: RH CLEAR WAS GIVEN

: TWO SUCCESSIVE "GO" (RHCSI BIT #0)  
: WAS GIVEN WITHOUT GIVING  
: TIME FOR FIRST "GO" TO  
: COMPLETE  
: THE FOLLOWING REGISTER  
: DID NOT CONTAIN WHAT IS  
: IN "GOOD"

2115			
2116			
2117			
2118			
2119			
2120			
2121			
2122	002120	065321	DH11
2123	002122	066610	DT6
2124	002124	066757	DF6
2125			
2126			: ITEM 71
2127	002126	057660	EM70
2128	002130	065216	DH6
2129	002132	066610	DT6
2130	002134	066757	DF6
2131			
2132			: ITEM 72
2133	002136	057660	EM70
2134	002140	065363	DH12
2135	002142	066610	DT6
2136	002144	066757	DF6
2137			
2138			: ITEM 73
2139	002146	057660	EM70
2140	002150	065425	DH13
2141	002152	066610	DT6
2142	002154	066757	DF6
2143			
2144			: ITEM 74
2145	002156	057660	EM70
2146	002160	065465	DH14
2147	002162	066610	DT6
2148	002164	066757	DF6
2149			
2150			: ITEM 75
2151	002166	057660	EM70
2152	002170	065527	DH15
2153	002172	066610	DT6
2154	002174	066757	DF6
2155			
2156			: ITEM 76
2157	002176	060134	EM76
2158			
2159			
2160			
2161			
2162			
2163			
2164			
2165	002200	065321	DH11
2166	002202	066610	DT6
2167	002204	066757	DF6
2168			
2169			: ITEM 77
2170	002206	060134	EM76

: RH CLEAR WAS GIVEN A 2 WORD  
: WRITE WAS DONE FROM A  
: LOCATION TAGED WAFROM  
: AND WITH BAI BIT SET  
: AT THE END OF THE WRITE  
: THE FOLLOWING REGISTER DID  
: NOT CONTAIN WHAT IS  
: IN GOOD

2171	002210	065216	DH6
2172	002212	066610	DT6
2173	002214	066757	DF6
2174			
2175			:ITEM 100
2176	002216	060134	EM76
2177	002220	065363	DH12
2178	002222	066610	DT6
2179	002224	066757	DF6
2180			
2181			:ITEM 101
2182	002226	060134	EM76
2183	002230	065425	DH13
2184	002232	066610	DT6
2185	002234	066757	DF6
2186			
2187			:ITEM 102
2188	002236	060134	EM76
2189	002240	065465	DH14
2190	002242	066610	DT6
2191	002244	066757	DF6
2192			
2193			:ITEM 103
2194	002246	060134	EM76
2195	002250	065527	DH15
2196	002252	066610	DT6
2197	002254	066757	DF6
2198			
2199			:ITEM 104
2200	002256	060427	EM104
2201			
2202			
2203			
2204			
2205			
2206			
2207	002260	065741	DH104
2208	002262	066636	DT104
2209	002264	066770	DF104
2210			
2211			:ITEM 105
2212	002266	060427	EM104
2213	002270	066011	DH105
2214	002272	066652	DT105
2215	002274	066775	DF105
2216			
2217			:ITEM 106
2218	002276	060665	EM106
2219			
2220			
2221			
2222			
2223			
2224			
2225			
2226			

```

:RH CLEAR WAS GIVEN AN
:IPCK BIT SHOWN IN "IPCK" WAS SET
:ZEROS WERE MOVED INTO RHDB
:ON READING RHDB
:THE FOLLOWING REGISTER
:DID NOT CONTAIN WHAT
:IS IN GOOD
:PC TEST NO,IPCK,RHCS3 GOOD,RHCS3 BAD
:$ERRPC,TSTNM,IP,$GDDAT,$BDDAT
:0,0,0,0

```

```

:PC TST NO,IPCK,RHCS2 GOOD,RHCS2 BAD
:$ERRPC,TSTNM,IP,$GDDAT,$BDDAT
:0,0,0,0

```

```

:RH CLEAR WAS GIVEN AN
:IPCK BIT SHOWN IN "IPCK" WAS SET
:ZEROS WERE MOVED INTO
:RHDB FROM AN ODD WORD
:RHDB WAS READ
:AN RH CLEAR WAS GIVEN
:TO CLEAR ALL ERRORS
:THE FOLLOWING REGISTER
:DID NOT CONTAIN WHAT

```



2227				: IS IN GOOD
2228	002300	066061	DH106	: PC TSTNM, IPCK, RHCS1 GOOD, RHCS1 BAD
2229	002302	066666	DT106	: \$ERRPC, TSTNM, IP, \$GDDAT, \$BCDAT
2230	002304	067002	DF106	: 0,0,0,0,0
2231				
2232			: ITEM 107	
2233	002306	060665	EM106	
2234	002310	066011	DH105	: PC, TSTNM, IPCK, RHCS2 GOOD, RHCS2 BAD
2235	002312	066652	DT105	
2236	002314	066775	DF105	
2237				
2238			: ITEM 110	
2239	002316	060665	EM106	
2240	002320	065741	DH104	
2241	002322	066636	DT104	
2242	002324	066770	DF104	
2243				
2244			: ITEM 111	
2245	002326	060665	EM106	
2246	002330	066131	DH111	: PC, TSTNM, IPCK, RHBA GOOD, RHBA BAD
2247	002332	066636	DT104	: \$ERRPC, TSTNM, IP, \$GDDAT, \$BDDAT
2248	002334	066770	DF104	: 0,0,0,0,0
2249				
2250			: ITEM 112	
2251	002336	060665	EM106	
2252	002340	066177	DH112	: PC, TSTNM, IPCK, RHBAE GOOD, RHBAE BAD
2253	002342	066636	DT104	: \$ERRPC, TSTNM, IP, \$GDDAT, \$BDDAT
2254	002344	066770	DF104	: 0,0,0,0,0
2255				
2256			: ITEM 113	
2257	002346	060665	EM106	
2258	002350	066247	DH113	: PC, TSTNM, IPCK, RHWC GOOD, RHWC BAD
2259	002352	066636	DT104	: \$ERRPC, TSTNM, IP, \$GDDAT, \$BDDAT
2260	002354	066770	DF104	: 0,0,0,0,0
2261				
2262			: ITEM 114	
2263	002356	061156	EM114	: RH CLEAR WAS GIVEN AN
2264				: A WRITE CHECK WAS DONE
2265				: THE FOLLOWING REGISTER
2266				: DID NOT CONTAIN WHAT IS
2267				: IN "GOOD"
2268	002360	065216	DH6	
2269	002362	066610	DT6	
2270	002364	066757	DF6	
2271				
2272			: ITEM 115	
2273	002366	061156	EM114	: RH CLEAR WAS GIVEN AN
2274				: A WRITE CHECK WAS DONE
2275				: THE FOLLOWING REGISTER
2276				: DID NOT CONTAIN WHAT IS
2277				: IN "GOOD"
2278	002370	065363	DH12	
2279	002372	066610	DT6	
2280	002374	066757	DF6	
2281				
2282			: ITEM 116	

2283	002376	061325	EM116
2284			
2285			
2286			
2287			
2288			
2289	002400	065321	DH11
2290	002402	066610	DT6
2291	002404	066757	DF6
2292			
2293			: ITEM 117
2294	002406	061325	EM116
2295	002410	065216	DH6
2296	002412	066610	DT6
2297	002414	066757	DF6
2298			
2299			: ITEM 120
2300	002416	061325	EM116
2301	002420	065363	DH12
2302	002422	066610	DT6
2303	002424	066757	DF6
2304			
2305			: ITEM 121
2306	002426	061325	EM116
2307	002430	065425	DH13
2308	002432	066610	DT6
2309	002434	066757	DF6
2310			
2311			: ITEM 122
2312	002436	061325	EM116
2313	002440	065465	DH14
2314	002442	066610	DT6
2315	002444	066757	DF6
2316			
2317			: ITEM 123
2318	002446	061325	EM116
2319	002450	065527	DH15
2320	002452	066610	DT6
2321	002454	066757	DF6
2322			
2323			: ITEM 124
2324	002456	061535	EM124
2325			
2326			
2327			
2328			
2329	002460	065363	DH12
2330	002462	066610	DT6
2331	002464	066757	DF6
2332			
2333			: ITEM 125
2334	002466	061535	EM124
2335	002470	065321	DH11
2336	002472	066610	DT6
2337	002474	066757	DF6
2338			

;RH CLEAR WAS GIVEN AN  
;A WRITE CHECK WAS DONE  
;THEN ANOTHER RH CLEAR WAS  
;GIVEN TO CLEAR ALL ERRORS  
;THE FOLLOWING REGISTER DID  
;NOT CONTAIN WHAT IS IN "GOOD"

:ON A SILO TEST TO  
:TEST DBL RHCS3-BIT #10  
:THE FOLLOWING REGISTER  
:DID NOT CONTAIN WHAT  
:IS IN "GOOD"

2339			:ITEM 125	
2340	002 76	061535	EM124	
2341	002 0	065216	DH6	
2342	002 2	066610	DT6	
2343	0025 4	066757	DF6	
2344				
2345			:ITEM 127	
2346	002506	061535	EM124	
2347	002510	065425	DH13	
2348	002512	066610	DT6	
2349	002514	066757	DF6	
2350				
2351			:ITEM 130	
2352	002516	061535	EM124	
2353	002520	065465	DH14	
2354	002522	066610	DT6	
2355	002524	066757	DF6	
2356				
2357			:ITEM 131	
2358	002526	061535	EM124	
2359	002530	065527	DH15	
2360	002532	066610	DT6	
2361	002534	066757	DF6	
2362			:ITEM132	
2363	002536	061700	EM132	:BEFORE DATA TRANSFER
2364				:COMMAND WAS TO BE GIVEN
2365				:THE RP DRIVE
2366				:STATUS REGISTER DID NOT
2367				:CONTAIN WHAT IS IN GOOD
2368				
2369	002540	066315	DH132	:PC
2370				:TEST NO.
2371				:RHDS1 GOOD
2372				:RHDS1 BAD
2373	002542	066610	DT6	
2374	002544	066757	DF6	
2375			:ITEM133	
2376	002546	062056	EM133	:BEFORE DATA TRANSFER
2377				:COMMAND WAS TO BE GIVEN
2378				:THE RS DRIVE
2379				:STATUS REGISTER DID NOT
2380				:CONTAIN WHAT IS IN GOOD
2381				
2382	002550	066315	DH132	:PC
2383				:TEST NO.
2384				:RHDS1 GOOD
2385				:RHDS1 BAD
2386	002552	066610	DT6	
2387	002554	066757	DF6	
2388			:ITEM 134	
2389	002556	062234	EM134	:BEFORE DATA TRANSFER COMMAND
2390				:WAS TO BE GIVEN THE
2391				:MAG TAPE DRIVE STATUS
2392				:REGISTER DID NOT CONTAIN WHAT
2393				:IS IN GOOD
2394				

2395	002560	066315		DH132	
2396	002562	066610		DT6	
2397	002564	066757		DF6	
2398			: ITEM135		
2399	002566	062424		EM135	: WAS WAITING FOR A BIT TO SET
2400					: BIT IN QUESTION IS IN "BIT WAITED FOR"
2401					: REGISTER IN QUESTION IN "REG ADDR"
2402	002570	066357		DH135	: PC
2403					: TEST NO
2404					: PC OF WAT
2405					: BIT
2406					: REG ADDR
2407	002572	066702		DT135	: \$ERRPC, TSTNM, WATIPC, WAITBT, WAITRE
2408	002574	067007		DF135	: 0,0,0,0,0
2409			: ITEM136		
2410	002576	062624		EM136	: WAS WAITING FOR A BIT TO RESET
2411					: BIT IN QUESTION IS IN "BIT WAITED FOR"
2412					: REGISTER IN QUESTION IN "REG ADDR"
2413	002600	066357		DH135	: PC
2414					: TEST NO
2415					: PC OF WAT
2416					: BIT
2417					: REG ADDR
2418	002602	066702		DT135	: \$ERRPC, TSTNM, WATIPC, WAITBT, WAITRE
2419	002604	067007		DF135	: 0,0,0,0,0
2420			: ITEM 137		
2421	002606	063026		EM137	: ON WRITING AND READING
2422					: THE REGISTER # IN "REG. ADDR"
2423					: IT DID NOT CONTAIN
2424					: EXPECTED VALUE.
2425	002610	066437		DH137	: PC
2426					: TEST NO
2427					: REG ADDR
2428					: GOOD DATA
2429					: BAD DATA
2430	002612	066716		DT137	: \$ERRPC, TSTNM, \$BADDR, \$GDDAT, \$BDDAT
2431	002614	067014		DF137	: 0,0,0,0,0
2432					
2433					
2434			: ITEM 140		
2435	002616	063154		EM140	
2436	002620	066061		DH106	
2437	002622	066666		DT106	
2438	002624	067002		DF106	
2439			: ITEM 141		
2440	002626	063406		EM141	: AFTER CLEARING THE RH AND
2441					: WRITING TWO WORD INTO
2442					: RHDB AND READING IT
2443					: BACK TWICE, CAUSED RHCS1 TO
2444					: HAVE WRONG DATA,
2445					: GIVEN IN BAD RHCS1
2446	002630	065321		DH11	: PC
2447					: TEST NUMBER
2448					: RHCS1 GOOD
2449					: RHCS1 BAD
2450	002632	066610		DT6	: \$ERRPC, TSTNM, \$GDDAT, \$BDDAT

2451	002634	066757	DF6	;0,0,0,0
2452				
2453				
2454			:ITEM 142	
2455	002636	063607	EM142	; AFTER CLEARING THE RH AND
2456				; WRITING TWO WORD INTO
2457				; RHDB AND READING IT
2458				; BACK TWICE, CAUSED RHCS3 TO
2459				; HAVE WRONG DATA
2460				; GIVEN IN BAD RHCS3
2461	002640	065363	DH12	; PC
2462				; TEST NUMBER
2463				; RHCS3 GOOD
2464				; RHCS3 BAD
2465	002642	066610	DT6	; \$ERRPC, TSTNM, \$GDDAT, \$BDDAT
2466	002644	066757	DF6	; 0,0,0,0
2467				
2468				
2469			:ITEM 143	
2470	002646	064010	EM143	; AFTER CLEARING THE RH AND
2471				; WRITING TWO WORD INTO
2472				; RHDB AND READING IT
2473				; BACK TWICE, CAUSED RHBA TO
2474				; HAVE WRONG DATA
2475				; GIVEN IN BAD RHBA
2476	002650	065425	DH13	; PC
2477				; TEST NUMBER
2478				; RHBA GOOD
2479				; RHBA BAD
2480	002652	066610	DT6	; \$ERRPC, TSTNM, \$GDDAT, \$BDDAT
2481	002654	066757	DF6	; 0,0,0,0
2482				
2483				
2484			:ITEM 144	
2485	002656	064210	EM144	; AFTER CLEARING THE RH AND
2486				; WRITING TWO WORD INTO
2487				; RHDB AND READING IT
2488				; BACK TWICE, CAUSED RHBAE TO
2489				; HAVE WRONG DATA
2490				; GIVEN IN BAD RHBAE
2491	002660	065465	DH14	; PC
2492				; TEST NUMBER
2493				; RHBAE GOOD
2494				; RHBAE BAD
2495	002662	066610	DT6	; \$ERRPC, TSTNM, \$GDDAT, \$BDDAT
2496	002664	066757	DF6	; 0,0,0,0
2497				
2498				
2499			:ITEM 145	
2500	002666	064411	EM145	; AFTER CLEARING THE RH AND
2501				; WRITING TWO WORD INTO
2502				; RHDB AND READING IT
2503				; BACK TWICE, CAUSED RHWC TO
2504				; HAVE WRONG DATA
2505				; GIVEN IN BAD RHWC
2506	002670	065527	DH15	; PC

2507					: TEST NUMBER
2508					: RHC GOOD
2509					: RHC BAD
2510	002672	066610			: \$ERRPC, TSTNM, \$GDDAT, \$BDDAT
2511	002674	066757			: 0, 0, 0, 0
2512					
2513	002676	064611			: A DEVICE BASE ADDRESS DID NOT
2514					: TIME OUT BUT CORRESPONDING
2515					: VECTOR ADDRESS DID TIME OUT
2516	002700	066511			: PC
2517					: BASE
2518					: VECTOR
2519	002702	066732			: \$ERRPC, TESTDV, TESTVC, 0
2520	002704	067021			: 0, 0, 0
2521					
2522					
2523	002706	065003			: A DEVICE ADDRESS DID NOT
2524					: TIME OUT BUT NO LNITS HAD
2525					: APPROPRIATE DRIVE TYPE
2526	002710	066540			: PC
2527					: BASE ADDRESS
2528	002712	066742			: \$ERRPC, TESTDV
2529	002714	067024			
2530					

2531  
2532  
2533  
2534  
2535  
2536  
2537  
2538  
2539  
2540  
2541  
2542  
2543  
2544  
2545  
2546  
2547  
2548  
2549  
2550  
2551  
2552  
2553  
2554  
2555  
2556  
2557  
2558  
2559  
2560  
2561  
2562  
2563  
2564  
2565  
2566  
2567  
2568  
2569  
2570  
2571  
2572  
2573  
2574  
2575  
2576  
2577  
2578  
2579  
2580  
2581  
2582  
2583  
2584  
2585  
2586

002716 000254

\*\*\*\*\*  
:RH70 REGISTERS

RPVEC: 254 ;RP VECTOR ADDRESS

\*\*\*\*\*  
:WORD COUNT REGISTER (RHWC)  
:EACH BIT IS CALLED BY BIT NUMBER

:BUS ADDRESS REGISTER (RHBA)  
:EACH BIT IS CALLED BY BIT NUMBER

:CONTROL AND STATUS REGISTER 2 (RHCS2)

US1= 1 ;UNIT SELECT (BIT #0)  
US2= 2 ;UNIT SELECT (BIT #1)  
US4= 4 ;UNIT SELECT (BIT #2)  
BAI= 10 ;BUS ADDRESS INCREMENT INHIBIT (BIT #3)  
PAT= 20 ;INVERT PARITY  
CLR= 40 ;CLEAR (BIT #5)  
IR= 100 ;INPUT READY (BIT #6)  
OR= 200 ;OUTPUT READY (BIT #7)  
MPE= 400 ;MASS BUS PARITY ERROR (BIT #9)  
MDPE= 400 ;MASS BUS PARITY ERROR (BIT #8)  
MXF= 1000 ;MISSED TRANSFER ERROR (BIT #9)  
PGE= 2000 ;PROGRAM ERROR (BIT #10)  
NEM= 4000 ;NON EXISTANT MEMORY (BIT #11)  
NED= 10000 ;NON EXISTANT DRIVE (BIT #12)  
PE= 20000 ;UNIBUS PARITY ERROR (BIT #13)  
WCE= 40000 ;WRITE CHECK ERROR (BIT #14)  
DLT= 100000 ;DATA LATE (BIT #15)

:CONTROL AND STATUS REGISTER 3 (RHCS3)

IPCK0= 1 ;INVERT PARITY CHECK BIT 0  
IPCK1= 2 ;INVERT PARITY CHECK BIT 1  
IPCK2= 4 ;INVERT PARITY CHECK BIT 2  
IPCK3= 10 ;INVERT PARITY CHECK BIT 3  
IE= 100 ;INTERRUPT ENABLE (BIT #6)  
DBL= 2000 ;DOUBLE WORD BOUNDARY (BIT #10)  
WCEEW= 4000 ;WRITE CHECK EVEN WORD (BIT #11)  
WCEOW= 10000 ;WRITE CHECK ODD WORD (BIT #12)  
DPEEW= 20000 ;DATA PARITY ERROR EVEN WORD (BIT #13)  
DPEOW= 40000 ;DATA PARITY ERROR ODD WORD (BIT #14)  
APE= 100000 ;ADDRESS PARITY ERROR (BIT #15)

:DATA BUFFER REGISTER (RHDB)

2587  
2588  
2589  
2590  
2591  
2592  
2593  
2594  
2595  
2596  
2597  
2598  
2599  
2600  
2601  
2602  
2603  
2604  
2605  
2606  
2607  
2608  
2609  
2610  
2611  
2612  
2613  
2614  
2615  
2616  
2617  
2618  
2619  
2620  
2621  
2622  
2623  
2624  
2625  
2626  
2627  
2628  
2629  
2630  
2631  
2632  
2633  
2634  
2635  
2636  
2637  
2638  
2639  
2640  
2641  
2642

000001  
000100  
000200  
000400  
001000  
002000  
004000  
020000  
040000  
100000  
  
000001  
000002  
000002  
000004  
000010  
000020  
000040  
000100  
000200  
000400  
001000  
002000  
004000  
010000  
020000  
040000  
100000  
  
000001  
000002  
000004  
000010  
000020  
000040  
000100  
000200  
000400  
001000  
002000  
004000  
010000

:EACH BIT IS CALLED BY BIT NUMBER

\*\*\*\*\*  
:RPO4 REGISTERS  
\*\*\*\*\*

:CONTROL AND STATUS 1 REGISTER. (#00)

GO= 1 ;GO (BIT #0)  
IE= 100 ;INTERRUPT ENABLE (BIT #6)  
RDY= 200 ;READY (BIT #7)  
A16= 400 ;HIGH ORDER UNIBUS BITS (BIT #8)  
A17= 1000 ;HIGH ORDER UNIBUS BITS (BIT #9)  
PSEL= 2000 ;PORT SELECT (BIT #10)  
DVA= 4000 ;DEVICE AVAILABLE (BIT #11)  
MCPE= 20000 ;MASSBUSS PARITY ERROR (BIT #13)  
TRE= 40000 ;TRANSFER ERROR (BIT #14)  
SC= 100000 ;SPECIAL CONDITION (BIT #15)

:STATUS REGISTER (RHDS1) (#01)

DF5= 1 ;DRIVE FORWARD 5"/SEC. (BIT #0)  
BOT= 2 ;BEGINING OF TAPE (BIT #1)  
DF20= 2 ;DRIVE FORWARD 20"/SEC. (BIT #1)  
DIGB= 4 ;DRIVE TO INNER GAVRD BAND (BIT #2)  
GRV= 10 ;GO REVERSE (BIT #3)  
DL64= 20 ;DIFFERENCE LESS THAN 64 (BIT #4)  
DE1= 40 ;DIFFERENCE EQUALS 1 (BIT #5)  
VV= 100 ;VOLUME VALID (BIT #6)  
DRY= 200 ;DRIVE READY (BIT #7)  
DPR= 400 ;DRIVE PRESENT (BIT #8)  
PROG= 1000 ;PROGRAMABLE (BIT #9)  
LBT= 2000 ;LAST SECTOR TRANSFERRED (BIT #10)  
WRL= 4000 ;WRITE LOCK (BIT #11)  
MOL= 10000 ;MEDIUM ON-LINE (BIT #12)  
PIP= 20000 ;POSITIONING OPERATION IN PROGRESS (BIT #13)  
ERR= 40000 ;COMPOSIT ERROR. (BIT #14)  
ATA= 100000 ;ATTENTION ACTIVE (BIT #15)

:ERROR REGISTER #01 (RHER1) (#02)

ILF= 1 ;ILLEGAL FUNCTION (BIT #0)  
ILR= 2 ;ILLEGAL REGISTER (BIT #1)  
RMR= 4 ;REGISTER MODIFICATION REFUSED (BIT #2)  
PAR= 10 ;PARITY ERROR (BIT #3)  
FER= 20 ;FORMAT ERROR (BIT #4)  
WCF= 40 ;WRITE CLOCK FAIL (BIT #5)  
ECH= 100 ;ECC HARD ERROR (BIT #6)  
HCE= 200 ;HEADER COMPARE ERROR (BIT #7)  
HCRC= 400 ;HEADER CRC ERROR (BIT #8)  
AOE= 1000 ;ADDRESS OVERFLOW ERROR (BIT #9)  
IAE= 2000 ;INVALID ADDRESS ERROR (BIT #10)  
WLE= 4000 ;WRITE LOCK ERROR (BIT #11)  
DTE= 10000 ;DRIVE TIMING ERROR (BIT #12)



2643	020000	OPI= 20000	: OPERATION INCOMPLETE (BIT #13)
2644	040000	UNS= 40000	: DRIVE UNSAFE (BIT #14)
2645	100000	DCK= 100000	: DATA CHECK ERROR (BIT 15)
2646			
2647			
2648			
2649	000001	DMD= 1	: DIAGNOSTIC MODE (BIT #0)
2650	000002	MCLK= 2	: MAINTAINABILITY CLOCK (BIT #1)
2651	000004	MINX= 4	: MAINTAINABILITY INDEX (BIT #2)
2652	000010	MSTCK= 10	: MAINTAINABILITY SECTOR CLOCK (BIT #3)
2653	000020	MRD= 20	: MAINTAINABILITY READ (BIT #4)
2654	000040	MWR= 40	: MAINTAINABILITY WRITE (BIT #5)
2655	001000	DTSY= 1000	: MAINTAINABILITY SYNC DETECTED (BIT #9)
2656			
2657			
2658			
2659	000001	ATO= 1	: DEVICE 0 (BIT #0)
2660	000002	AT1= 2	: DEVICE 1 (BIT #1)
2661	000004	AT2= 4	: DEVICE 2 (BIT #2)
2662	000010	AT3= 10	: DEVICE 3 (BIT #3)
2663	000020	AT4= 20	: DEVICE 4 (BIT #4)
2664	000040	AT5= 40	: DEVICE 5 (BIT #5)
2665	000100	AT6= 100	: DEVICE 6 (BIT #6)
2666	000200	AT7= 200	: DEVICE 7 (BIT #7)
2667			
2668			
2669			
2670			
2671			
2672			
2673			
2674	000001	EXT1= 1	: EXTENSION 1 (BIT #0)
2675	000002	EXT2= 2	: EXTENSION 2 (BIT #1)
2676	000004	EXT4= 4	: EXTENSION 3 (BIT #2)
2677	000010	EXT10= 10	: EXTENSION 4 (BIT #3)
2678	000020	EXT20= 20	: EXTENSION 5 (BIT #4)
2679	000040	EXT40= 40	: EXTENSION 6 (BIT #5)
2680	000100	SC1= 100	: SECTOR COUNT FIELD 0 (BIT #6)
2681	000200	SC2= 200	: SECTOR COUNT FIELD 1 (BIT #7)
2682	000400	SC4= 400	: SECTOR COUNT FIELD 2 (BIT #8)
2683	001000	SC10= 1000	: SECTOR COUNT FIELD 3 (BIT #9)
2684	002000	SC20= 2000	: SECTOR COUNT FIELD 4 (BIT #10)
2685	004000	TRK1= 4000	: TRACK FIELD 1 (BIT #11)
2686	010000	TRK2= 10000	: TRACK FIELD 2 (BIT #12)
2687	020000	TRK4= 20000	: TRACK FIELD 3 (BIT #13)
2688	040000	TRK10= 40000	: TRACK FIELD 4 (BIT #14)
2689	100000	TRK20= 100000	: TRACK FIELD 5 (BIT #15)
2690			
2691			
2692			
2693	000001	WCU= 1	: WRITE CURRENT UNSAFE (BIT #0)
2694	000002	CSF= 2	: CURRENT SINK FAILURE (BIT #1)
2695	000004	WSU= 4	: WRITE SELECT UNSAFE (BIT #2)
2696	000010	CSU= 10	: CURRENT SWITCH UNSAFE (BIT #3)
2697	000020	MSE= 20	: MOTOR SEQUENCE ERROR (BIT #4)
2698	000040	TDF= 40	: TRANSITIONS DETECTOR FAILURE (BIT #5)

2699	000100	TUF=	100	; TRANSITIONS UNSAFE (BIT #6)
2700	000200	FEN=	200	; FAILSAFE ENABLED (BIT #7)
2701	000400	WRU=	400	; WRITE READY UNSAFE (BIT #8)
2702	001000	MHS=	1000	; MULTIPLE HEAD SELECT (BIT #9)
2703	002000	NHS=	2000	; NO HEAD SELECTION (BIT #10)
2704	004000	IXE=	4000	; INDEX ERROR (BIT #11)
2705	010000	VU30=	10000	; 30VOLT UNSAFE (BIT #12)
2706	020000	PLU=	20000	; PLO UNSAFE (BIT #13)
2707	100000	ACU=	100000	; ACUNSAFE (BIT #15)
2708				
2709		; OFFSET REGISTER (RHOF) (#11)		
2710				
2711	000001	OF25=	1	; OFFSET 25 MICRO INCHES (BIT #0)
2712	000002	OF50=	2	; OFFSET 50 MICRO INCHES (BIT #1)
2713	000004	OF100=	4	; OFFSET 100 MICRO INCHES (BIT #2)
2714	000010	OF200=	10	; OFFSET 200 MICRO INCHES (BIT #3)
2715	000020	OF400=	20	; OFFSET 400 MICRO INCHES (BIT #4)
2716	000040	OF800=	40	; OFFSET 800 MICRO INCHES (BIT #5)
2717				
2718	000200	OFREV=	200	; OFFSET NEGATIVE (REVERSE) (BIT #5)
2719	002000	HCI=	2000	; HEADER COMPARE INHIBIT (BIT #10)
2720	004000	ECI=	4000	; ERROR CORRECTION CODE INHIBIT (BIT #11)
2721	010000	FMT22=	10000	; FORMAT BIT (BIT #12)
2722				
2723		; TAPE CONTROL REGISTER (RHTC)		
2724				
2725	000001	S1=	1	; SLAVE NUMBER
2726	000002	S2=	2	; SLAVE NUMBER
2727	000004	S4=	4	; SLAVE NUMBER
2728	000010	EPAR=	10	; EVEN PARITY
2729	000020	FMT1=	20	; FORMAT
2730	000040	FMT2=	40	; FORMAT
2731	000100	FMT4=	100	; FORMAT
2732	000200	FMT8=	200	; FORMAT
2733	000400	DEN1=	400	; DENSITY
2734	001000	DEN2=	1000	; DENSITY
2735	002000	DEN4=	2000	; DENSITY
2736	001400	BPI8=	1400	; 800 BPI
2737	000300	NML=	300	; NORMAL - 2 FRAMES PER WORD
2738				
2739				
2740				
2741				
2742				
2743		; DRIVE TYPE REGISTER		
2744				
2745	002000	SLVPR=	2000	; SLAVE PRESENT (BIT #10)
2746	010000	CH7=	10000	; CHANNEL 7 (BIT #12)
2747				
2748		; DESIRED CYLINDER ADDRESS (RHCA) (#12)		
2749		; EACH BIT IS CALLED BY BIT NUMBER.		
2750				
2751				
2752				
2753				
2754		; CURRENT CYLINDER ADDRESS (RHCC) (#13)		

2755  
2756  
2757  
2758  
2759  
2760  
2761  
2762  
2763  
2764  
2765  
2766  
2767  
2768  
2769  
2770  
2771  
2772  
2773  
2774  
2775  
2776  
2777  
2778  
2779  
2780  
2781  
2782  
2783  
2784  
2785  
2786  
2787  
2788  
2789  
2790  
2791  
2792

000001  
000002  
000010  
000020  
000040  
000100  
040000  
100000

:EACH BIT IS CALLED BY BIT NUMBER

:SERIAL NUMBER REGISTER (RHSN) (#14)  
:EACH IS CALLED BY BIT NUMBER

:ERROR REGISTER #03 (RHER3) (#15)

PSU=	1	:PACK SPEED UNSAFE (BIT #0)
VUF=	2	:VELOCITY UNSAFE (BIT #1)
UWR=	10	:ANY UNSAFE EXCEPT READ/WRITE (BIT #3)
PRE=	20	:DISK PACK ROTATION ERROR (BIT #4)
ACL=	40	:AC LOW (BIT #5)
DCL=	100	:DC LOW (BIT #6)
SKI=	40000	:SEEK INCOMPLETE (BIT #14)
OCYL=	100000	:OFF CYLINDER (BIT #15)

:ECC POSITION REGISTER (RHEC1) (#16)  
:EACH BIT IS CALLED BY BIT NUMBER

:ECC PATTERN REGISTER (RHEC2) (#17)  
:EACH BIT IS CALLED BY BIT NUMBER

::\*\*\*\*\*

2793					
2794					
2795					
2796	000000		CS1=	0	:CONTROL STATUS 1
2797	000002		WC=	2	:WORD COUNT
2798	000004		BA=	4	:BUS ADDRESS
2799	000006		FC=	6	:FRAME COUNT
2800	000006		DA=	6	:DESIRED SECTOR/TRACK
2801	000010		CS2=	10	:CONTROL STATUS 2
2802	000012		DS=	12	:DRIVE STATUS
2803	000014		ER1=	14	:ERROR 1
2804	000016		AS=	16	:ATTENTION SUMMERY
2805	000020		LA=	20	:LOOK AHEAD
2806	000022		DB=	22	:DATA BUFFER
2807	000024		MR=	24	:MAINTENANCE REGISTER
2808	000026		DT=	26	:DRIVE TYPE
2809	000030		SN=	30	:SERIAL NUMBER
2810	000032		OF=	32	:OFFSET
2811	000032		TC=	32	:TAPE CONTROL
2812	000034		DC=	34	:DESIRED CYLINDER
2813	000036		CC=	36	:CURRENT CYLINDER
2814	000040		ER2=	40	:ERROR 2
2815	000042		ER3=	42	:ERROR 3
2816	000044		EC1=	44	:ECC 1
2817	000046		EC2=	46	:ECC 2
2818					
2819					
2820			:BUFFERS		
2821	003000		. =3000		
2822	003000		BUFEW:		
2823	003000	000000	BFEVEN: 0		
2824	003002		BUFOW:		
2825	003002	000422	BFODD: .BLKW 274		:BUFFER
2826			;STARTING ADDRESS OF REGISTERS		
2827					
2828	004046	172040	RSCS1: 172040		:RS ALONG
2829	004050	176700	RPCS1: 176700		:RP ALONE
2830	004052	172440	TMCS1: 172440		:TM ALONE
2831	004054	176300	MIXCS1: 176300		:MIXED SYSTEM
2832	004056	000000	PRESENT: 0		
2833					
2834					
2835					
2836					
			.SETTL REGISTER ADDRESSES		

2837  
2838  
2839  
2840  
2841  
2842  
2843  
2844  
2845  
2846  
2847  
2848  
2849  
2850  
2851  
2852  
2853 004060 000000  
2854 004062 000000  
2855 004064 000000  
2856 004066  
2857 004066  
2858 004066 000000  
2859 004070 000000  
2860 004072 000000  
2861 004074 000000  
2862 004076 000000  
2863 004100  
2864 004100 000000  
2865 004102 000000  
2866 004104 000000  
2867 004106 000000  
2868 004110 000000  
2869 004112  
2870 004112 000000  
2871 004114 000000  
2872 004116 000000  
2873 004120 000000  
2874 004122 000000  
2875 004124 000000  
2876 004126 000000  
2877 004130 000000  
2878 004132 000000  
2879  
2880  
2881  
2882  
2883  
2884  
2885  
2886  
2887 004134  
2888 004134 000000  
2889 004136 000002  
2890 004140 000006  
2891 004142 000010  
2892 004144 000012

: THE FOLLOWING ARE THE I/O REGISTERS FOR THE DEVICE UNDER TEST  
: THEY WILL BE FILLED WITH ADDRESSES DEPENDING ON WHAT DEVICE IS ON  
: THE SYSTEM.  
: IN THE CASE OF A MIXED SYSTEM THAT IS WITH MORE THAN ONE RH DEVICE  
: THE FIRST ONE THAT EXISTS IN THE FOLLOWING LIST WILL BE PICKED  
: AND THE OTHERS WILL NOT BE USED.

- : LIST:-  
: 1) RS03  
: 2) RS04  
: 3) RPO4,5,6  
: 4) TMO2

: THE CONTENTS OF RHCS1 WHICH IS THE BASE ADDRESS IS  
: 172040 FOR RS03/04  
: 176700 FOR RPO4,5,6  
: 172440 FOR TMO2

: THE REST OF THE LOCATIONS WILL BE FILLED BY THE PROGRAM  
RHCS1: 0 ; CONTROL AND STATUS 1  
RHWC: 0 ; WORD COUNT  
RHBA: 0 ; BUS ADDRESS  
RHFC: 0 ; FRAME COUNT  
RHDA: 0 ; DISK ADDRESS  
RHDST: 0 ; DESIRED SECTOR/TRACK ADDRESS  
RHCS2: 0 ; CONTROL AND STATUS 2  
RHDS1: 0 ; DRIVE STATUS  
RHER1: 0 ; ERROR #1  
RHAS: 0 ; ATTENTION SUMMARY  
RHCC: 0 ; CHECK CHARACTER  
RHLA: 0 ; LOOK-AHEAD  
RHDB: 0 ; DATA BUFFER  
RHMR: 0 ; MAINTAINABILITY  
RHDT: 0 ; DRIVE TYPE  
RHSN: 0 ; SERIAL NUMBER  
RHTC: 0 ; TAPE CONTROL  
RHOF: 0 ; OFFSET  
RHCA: 0 ; DESIRED CYLINDER ADDRESS  
RHCCA: 0 ; CURRENT CYLINDER ADDRESS  
RHER2: 0 ; ERROR #2  
RHER3: 0 ; ERROR #3  
RHEC1: 0 ; ECC POSITION  
RHEC2: 0 ; ECC PATTERN  
RHBAE: 0 ; BUS ADDRESS EXTENSION  
RHCS3: 0 ; CONTROL AND STATUS 3

: FUNCTION EQUATES

: TABLE OF FUNCTIONS FOR RHCS1 THEN "GO" BIT HAS TO BE SET  
FUTABL: ;  
NOPERA: 0 ; NO OPERATION  
UNLOAD: 2 ; UNLOAD (STAND BY)  
RECALI: 6 ; RECALIBRATE  
DCLEAR: 10 ; DRIVE CLEAR  
RELEASE: 12 ; RELEASE (DUAL-PORT OPERATION)

2893	004146	000030	SERCH:	30	; SEARCH COMMAND
2894	004150	000050	WRCHK:	50	; WRITE CHECK DATA
2895	004152	000052	WRCHD:	52	; WRITE CHECK HEADER AND DATA
2896	004154	000060	WRIDAT:	60	; WRITE DATA
2897	004156	000062	WRIFOR:	62	; WRITE HEADER AND DATA (FORMAT)
2898	004160	000070	READAT:	70	; READ DATA
2899	004162	000072	REFOR:	72	; READ HEADER AND DATA
2900	004164	000004	SEECOM:	4	; SEEK COMMAND
2901	004166	000014	OFSETC:	14	; OFFSET COMMAND
2902	004170	000016	RETCL:	16	; RETURN TO CENTERLINE
2903	004172	000022	PKACK:	22	; PACK ACKNOWLEDGE
2904	004174	000020	READIN:	20	; READ IN
2905	004176	000006	REWIND:	6	; REWIND
2906	004200	000076	REVRED:	76	; REVERSE READ
2907	004202	000030	SPACFD:	30	; SPACE FORWARD
2908	004204	000066	REVRT:	66	; REVERSE WRITE
2909	004206	000000	ILLEGL:	.WORD 0	; COMPUTED ILLEGAL FUNCTION
2910					
2911					; DATA BUFFER FOR READ WRITE
2912	004210	000106	WRFROM:	.BLKW 70.	; WRITE FROM THIS BUFFER
2913	004424	000106	REINTO:	.BLKW 70.	; READ INTO THIS BUFFER
2914	004640	000000	COMND:	0	; STORE COMMAND FOR COMMAND ROUTINE
2915	004642	000000	COMAND:	0	; STORE COMMAND
2916	004644	000000	NOGO:	0	; IF ZERO GO IS TO BE GIVEN IN COMMAND ROUTINE
2917					; IF ONES GO IS NOT TO BE GIVEN IN COMMAND ROUTINE
2918					; USED IN PGE TEST
2919	004646	000000	WATO:	0	; IF ZERO WAIT FOR BIT TO SET
2920					; IF ONES WAIT FOR BIT TO RESET
2921					; USED IN WAIT TRAP
2922	004650	000000	WRBIT:	0	; BITS NOT WRITTEN INTO
2923	004652	000000	SETBIT:	0	; BITS ALWAYS SET
2924	004654	000000	CLRBIT:	0	; BITS ALWAYS CLEARED
2925					
2926					
2927					
2928					; RESERVED LOCATIONS
2929	004656	000000	TSTNM:	0	; TEST NUMBER
2930	004660	000000	SLAVE:	0	; KEEP SLAVE NO.
2931	004662	000000	IP:	0	; IPCK NUMBER FOR ERROR PRINTOUT
2932	004664	000000	SILONM:	0	; SILO WORD NUMBER FOR ERROR PRINT OUT
2933	004666	000000	WAITPC:	0	; WAIT TRAP PC
2934	004670	000000	WAITRE:	0	; WAITING FOR REGISTER ADDRESS IN WAIT TRAP
2935	004672	000000	WAITBT:	0	; WAITING FOR BIT IN WAIT TRAP
2936					
2937					; TABLE FOR ATTENTION BITS
2938					; ATTENTION TABLE
2939	004674	001	002	004	ATABLE: .BYTE 1,2,4,10,20,40,100,200
2940	004677	010	020	040	
2941	004702	100	200		
2942					
2943					
2944					
2945					; RESERVED LOCATIONS FOR UNIT SELECT
2946	004704	172040	BASEAD:	172040	; RS BASE ADDRESS
2947	004706	176700	BASPAD:	176700	; RP BASE ADDRESS
2948	004710	172440	BASTAD:	172440	; TU BASE ADDRESS

2949	004712	176300	BASEMAD:	176300	; MIXED SYSTEM BASE ADDRESS
2950					
2951					
2952	004714	000000	BASEVC:	0	; RS VECTOR
2953	004716	000254	BHSP:	254	; RP VECTOR
2954	004720	000001	BAST:	1	; TU VECTOR
2955	004722	000002	BASM:	2	; MIXED VECTOR
2956					
2957					; TABLE FOR GIVEN BASE ADDRESSES
2958	004724	000010	BSGIVA:	.BLKW 8.	
2959					
2960					
2961					; TABLE FOR GIVEN VECTOR
2962	004744	000010	BSGIVV:	.BLKW 8.	
2963					
2964					
2965	004764	000004	NMRHS:	4	; NUMBER OF RH
2966	004766	000000	BASINX:	0	; INDEX FOR BASE
2967	004770	000000	WORKBS:	0	; WORKING BASE
2968	004772	000000	WORKNM:	0	; WORKING NUMBER FOR RH
2969	004774	000000	WORKVC:	0	; WORKING VECTOR FOR RH
2970	004776	000000	FORBAE:	0	; TOTAL NO. OF REG. FOR BAE CALCULATION
2971	005000	000000	LOPCT:	0	; LOOP COUNT FOR 200 START
2972	005002	000000	LOPCT1:	0	; LOOP COUNT FOR 210 START
2973	005004	000000	USEVEC:	0	; USE VECTOR
2974					
2975					
2976					
2977	005006	000000	GIVE:	0	; IF ONES OPERATOR WILL GIVE ADDRESSES
2978	005010	000000	UNIT:	0	; UNIT UNDER TEST
2979	005012	000000	ST200:	0	; ALL ONES INDICATE STARTING FROM 200
2980					; TESTING TABLE
2981	005014	000010	TSRHNO:	.BLKW 8.	; RH NO TO BE TESTED IN ORDER OF TEST
2982					
2983	005034	000010	TSDEVICE:	.BLKW 8.	; DEVICE TO BE TESTED IN ORDER OF TEST
2984					; 1=RS03,3/L,3/LA 2=RS04,4/L 4=RPO4,5,6 SINGLE PORT
2985					; 10=RPO4,5,6 DUAL PORT 20=TM02
2986					
2987					
2988	005054	000010	TSUNIT:	.BLKW 8.	; UNIT NO. TO BE TESTED IN ORDER OF TEST
2989					
2990					
2991	005074	000010	TSSLAV:	.BLKW 8.	; SLAVE NO TO BE TESTED IN ORDER OF TEST
2992					
2993					
2994	005114	000000	TSTOTL:	0	; TOTAL NO. OF UNITS TO BE TESTED
2995	005116	000000	TSINDX:	0	; INDEX TO ONE UNDER TEST
2996					
2997					
2998	005120	000010	TSVEC:	.BLKW 8.	; VECTOR ADDRESS IN ORDER OF TEST
2999					
3000					
3001	005140	000010	TSBAE:	.BLKW 8.	; RHBAE ADDRESS IN ORDER OF TEST
3002					
3003					
3004	005160	000010	TSCS3:	.BLKW 8.	; RHCS3 ADDRESS IN ORDER OF TEST

```

3005
3006
3007 005200 000010      TSCOMD: .BLKW  8.          ;COMMAND ADDRESS IN ORDER OF TEST
3008
3009
3010 005220 000010      TSBASE: .BLKW  8.          ;BASE ADDRESS IN ORDER OF TEST
3011
3012
3013
3014
3015
3016
3017 005240 000000      ERFLGS: 0                ;ERROR FLAG
3018 005242 000000      FIRST:  0                ;IF ZERO WILL TYPE HEADER
3019                                     ;IF ONES WILL NOT TYPE HEADER
3020
3021
3022
3023 005244 000000      ATTENT: 0                ;ATTENTION BIT FOR PRESENT UNIT
3024 005246 000000      TOTALAT: 0                ;TATAL ATTENTION BITS
3025
3026 005250 000000      TMP0:  .WORD  0          ;TEMP STORAGE
3027 005252 000000      TMP1:  .WORD  0          ;TEMP STORAGE
3028 005254 000000      TMP4:  .WORD  0          ;TEMP STORAGE
3029                                     ;PERMANENT TABLE
3030 005256 172040      PERRS: 172040            ;RS BASE ADDRESS
3031 005260 176700      PERRP: 176700            ;RP BASE ADDRESS
3032 005262 172440      PERTU: 172440            ;TU BASE ADDRESS
3033 005264 176300      PERMX: 176300            ;MIXED BASE ADDRESS
3034
3035 005266 000204      PERRSV: 204              ;RS VECTOR ADDRESS
3036 005270 000254      PERRPV: 254              ;RP VECTOR ADDRESS
3037 005272 000224      PERTUV: 224              ;TU VECTOR ADDRESS
3038 005274 000000      PERMXV: 0                ;MIXED VECTOR ADDRESS
3039
3040 005276 000004      PERNUM: 4                ;NUMBER OF RH
3041
3042                                     ;WORKING TABLE
3043
3044 005300 000004      DEVPNT: .BLKW  4          ;BASE ADDRESS TO BE TESTED
3045 005310 000004      VECPNT: .BLKW  4          ;VECTOR ADDRESS TO BE TESTED
3046 005320 000000      LTRY:   0                ;NO. OF UNITS TO BE TESTED
3047 005322 000000      TFOUND: 0                ;TWICE NUMBER OF RH FOUND
3048 005324 000000      TESTDV: 0                ;STORES BASE ADDRESS OF TEST DEVICE
3049 005326 000000      TESTVC: 0                ;STORES VECTOR OF TEST DEVICE
3050                                     ;THE ABOVE TWO IS BEFORE ANY
3051                                     ;DEVICE IS FOUND.
3052                                     ;TABLE FOR DRIVE TYPES
3053                                     ;0=RS03, 1=RS03/L, 2=RS04, 3=RS04/L, 4=RS03/LA
3054                                     ;20020=SINGLE PORT RPO4, 24020=DUAL PORT RPO4
3055                                     ;20021=SINGLE PORT RPO5, 24021=DUAL PORT RPO5
3056                                     ;20022=SINGLE PORT RPO6, 24022=DUAL PORT RPO6
3057                                     ;142010=TU16
3058
3059 005330 000000 000001 000002  TYPNT: .WORD  0,1,2,3,4,20020,24020,20021,24021,20022,24022,142010
3060 005336 000003 000004 020020

```



3061	005344	024020	020021	024021		
3062	005352	020022	024022	142010		
3063	005360	000000			POINTT: 0	;POITER TO ABOVE TABLE
3064	005362	000014			NTYPNT: 14	;NUMBER OF ABOVE TYPES
3065						
3066	005364	000000			TMPSLV: 0	;TEMPORARY SLAVE COUNTER
3067						
3068						
3069	005366	000004			FDEVIC: .BLKW 4	;TABLE OF DATA FOR DEVICES FOUND ;DEVICE FOUND, 1=TU, 2=RP DUAL ;3=RP SINGLE, 4=RS04, 5=RS03
3070						
3071	005376	000004			FUNITN: .BLKW 4	;UNIT NUMBER FOUND
3072	005406	000004			FTYPE: .BLKW 4	;DDRIVE TYPE FOUND
3073	005416	000004			FVECTR: .BLKW 4	;VECTOR FOUND
3074	005426	000004			FBASEA: .BLKW 4	;BASE ADDRESS FOUND
3075	005436	000004			FRHNM: .BLKW 4	;RH NUMBER FOUND
3076	005446	000004			FSLAVE: .BLKW 4	;TU16 SLAVE NUMBER FOUND
3077	005456	000004			FDRIVR: .BLKW 4	;DRIVER ADDRESS FOUND
3078	005466	000004			FBAE: .BLKW 4	;BAE ADDRESS FOUND
3079	005476	000004			FCS3: .BLKW 4	;CS3 ADDRESS FOUND
3080						
3081	005506	000000			TSTUNT: 0	;TOTAL NUMBER OF RH FOUND AT ;END OF SIZE
3082						
3083	005510	000000			WORUNT: 0	;SAME AS ABOVE BUT TO BE ;DECREASED AT END PROGRAM
3084						
3085						
3086	005512	000000			VECTOR: 0	;VECTOR UNDER TEST
3087	005514	177740			LERADD: 177740	;LOW ERROR ADDRESS REG.
3088	005516	177742			HERADD: 177742	;HIGH ERROR ADDRESS REG
3089	005520	177744			MEMERR: 177744	;MEMORY SYSTEM ADDRESS REG
3090		000114			PARE70=114	;PARITY ERROR VECTOR FOR 70

```

3091 .SBTTL REGISTER TEST
3092 005522 012737 177777 005012 BEGIN: MOV #-1,ST200
3093 005530 000402 BR START
3094 005532 005037 005012 BEGIN2: CLR ST200
3095
3096 005536 START:
3097 .SBTTL INITIALIZE THE COMMON TAGS
3098 ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
3099 005536 012706 001100 MOV $CMTAG,R6 ;;FIRST LOCATION TO BE CLEARED
3100 005542 005026 CLR (R6)+ ;;CLEAR MEMORY LOCATION
3101 005544 022706 001140 CMP $SWR,R6 ;;DONE?
3102 005550 001374 BNE -6 ;;LOOP BACK IF NO
3103 005552 012706 001000 MOV $STACK,SP ;;SETUP THE STACK POINTER
3104 ;;INITIALIZE A FEW VECTORS
3105 005556 012737 043120 000020 MOV $$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
3106 005564 012737 000340 000022 MOV #340,@#IOTVEC+2 ;;LEVEL 7
3107 005572 012737 044744 000030 MOV $ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
3108 005600 012737 000340 000032 MOV #340,@#EMTVEC+2 ;;LEVEL 7
3109 005606 012737 046450 000034 MOV $TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
3110 005614 012737 000340 000036 MOV #340,@#TRAPVEC+2;LEVEL 7
3111 005622 012737 046530 000024 MOV $PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
3112 005630 012737 000340 000026 MOV #340,@#PWRVEC+2 ;;LEVEL 7
3113 005636 005037 001212 CLR $TIMES ;;INITIALIZE NUMBER OF ITERATIONS
3114 005642 005037 001214 CLR $ESCAPE ;;CLEAR THE ESCAPE ON ERROR ADDRESS
3115 005646 112737 000001 001115 MOVB #1,$ERMAX ;;ALLOW ONE ERROR PER TEST
3116 005654 012737 005654 001106 MOV #,$SLPADR ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
3117 005662 012737 005662 001110 MOV #,$SLPERR ;;SETUP THE ERROR LOOP ADDRESS
3118 ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
3119 ;;EQUAL TO A "-1" SETUP FOR A SOFTWARE SWITCH REGISTER.
3120 005670 013746 000004 MOV @#ERRVEC -(SP) ;;SAVE ERROR VECTOR
3121 005674 012737 005730 000004 MOV #64,$@#ERRVEC ;;SET UP ERROR VECTOR
3122 005702 012737 177570 001140 MOV #DSWR,$SWR ;;SETUP FOR A HARDWARE SWICH REGISTER
3123 005710 012737 177570 001142 MOV #DDISP,$DISPLAY ;;AND A HARDWARE DISPLAY REGISTER
3124 005716 022777 177777 173214 CMP #-1,@$SWR ;;TRY TO REFERENCE HARDWARE SWR
3125 005724 001012 BNE 66$ ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
3126 ;;AND THE HARDWARE SWR IS NOT = -1
3127 005726 000403 BR 65$ ;;BRANCH IF NO TIMEOUT
3128 005730 012716 005736 64$: MOV #65$,(SP) ;;SET UP FOR TRAP RETURN
3129 005734 000002 RTI
3130 005736 012737 000176 001140 65$: MOV $SWPREG,$SWR ;;POINT TO SOFTWARE SWP
3131 005744 012737 000174 001142 MOV #DISPREG,$DISPLAY
3132 005752 012637 000004 66$: MOV (SP)+,@#ERRVEC ;;RESTORE ERROR VECTOR
3133
3134
3135
3136 005756 012737 000000 177776 MOV #0,$PS ;SET PROCESSOR STATUS TO 0
3137 005764 012737 000200 000036 MOV #200,@#TRAPVEC+2 ;TRAP PRIORITY = 4
3138 005772 013700 002716 MOV @#RPVEC,R0 ;GET RP VECTOR ADDRESS
3139 005776 012720 043056 MOV #RPVECT,(R0)+ ;THIS IS FOR UNTIMELY INTERRUPTS
3140 006002 012710 000340 MOV #340,(R0) ;RPO4 INTERRUPT SERVICE ROUTINE
3141 ;PRIORITY = 7
3142 006006 004737 044064 JSR PC,@#$TKINT ;INITILIZE THE TK
3143 006012 005037 045100 CLR PRITEM ;CLEAR FOR ERROR MESSAGE PRINTOUT
3144 006016 005737 005242 TST @#FIRST ;IS THIS FIRST TIME ROUND
3145 006022 001001 BNE 1$ ;BRANCH IF NOT
3146 006024 000402 BR 2$
    
```

```

3147 006026 000137 006124 1$: JMP @#SND1
3148 006032 023737 000042 000046 2$: CMP @#42,@#46 ;ARE WE IN QV OR AUTO ACCEPT MODE?
3149 006040 001431 BEQ SND1 ;IF YES, SKIP HEADER
3150 006042 104401 006050 TYPE ,68$ ;TYPE ASCIZ STRING
3151 006046 000426 BR 67$ ;GET OVER THE ASCIZ
3152 ;:68$: .ASCIZ <15><12>/RH70 FUNCTIONAL CONTROLLER TEST DERHA-C/
3153 67$:
3154 006124 012737 177777 005242 SND1: MOV #-1,@#FIRST ;NEXT TIME DO NOT GIVE HEADER
3155 006132 012737 037746 000114 MOV #PARITY,@#PARE70 ;PARITY VECTOR
3156 006140 012737 000340 000116 MOV #340,@#PARE70+2 ;PRIORITY 7
3157 006146 012737 037610 000004 MOV #TIEOUT,@#ERRVEC ;TIMEOUT VECTOR
3158 006154 012737 000340 000006 MOV #340,@#ERRVEC+2 ;PRIORITY 7
3159
3160
3161
3162
3163
3164
3165
3166
3167
3168
3169
3170
3171
3172
3173
3174
3175
3176
3177
3178
3179
3180
3181
3182
3183
3184
3185
3186
3187
3188
3189
3190
3191
3192
3193
3194
3195
3196
3197
3198
3199
3200
3201
3202

```

```

:*****
:*TEST 1 SIZE FOR RH DEVICES
:* THIS TEST DETERMINS WHICH RH DEVICE IS ON THE SYSTEM.
:* IF THE SYSTEM HAS MORE THAN ONE RH DEVICE THEN ONLY ONE OF
:* THE DEVICES WILL BE USED IN THE FOLLOWING TESTS.
:* THE ONE THAT WILL BE USED IS THE FIRST ONE THAT IS PRESENT
:* IN THE FOLLOWING LIST
:* RS
:* RP
:* TM
:*
:* THE WAY THE TEST WORKS IS AS FOLLOWS:-
:* A REERENCE IS MADE TO THE CONTROL AND STATUS REGISTER 1
:* FOR THE RS BY A TST INSTRUCTION. IF IT RESPONDS THEN IT IS
:* ASSUMED PRESENT AND THE LOCATIONS FOR THE I/O REGISTERS ARE
:* FILLED WITH THE APPROPRIATE ADDRESSES.
:* THEN THE DRIVE TYPE REGISTER IS CHECKED TO HAVE GOOD
:* VALUES.
:* IF THE TST INSTRUCTION TRAPS TO A NON EXISTANT VECTOR
:* THEN A SIMILAR ATTEMPT IS MADE TO A RP CONTROL AND STATUS
:* REGISTER 1.
:* THEN A TM IS TRYED.
:* THEN A MIXED SYSTEM WITH MORE THAN ONE RH DEVICES IS TRYED.
:*****

```

```

3203 006162 000004 TST1: SCOPE
3204 006164 012737 000001 001212 MOV #1,$TIMES ;;DO 1 ITERATION
3205 006172 005737 005012 TST ST200 ;;IS IT A 200 START
3206 006173 001413 BEQ 2$ ;;BRANCH IF NOT 200 START
3207 006200 012700 005256 MOV #PERRS,RO ;;POINTER TO MOVE 9 WMPDS FROM
3208 006204 012701 000011 MOV #9,R1 ;;NUMBER TO MOVE
3209 006210 012702 005300 MOV #DEVPNT,R2 ;;POINTER TO MOVE 9 TORDS TO
3210 006214 012022 1$: MOV (RO)+,(R2)+ ;;FILL WORK TABLE
3211 006216 005301 DEC R1
3212 006220 001375 BNE 1$ ;;BRANCH IF 9 NOT DONE
3213 006222 000137 006522 JMP SETSIZ
3214 006226 2$:
3215 006226 104401 006234 TYPE ,65$ ;;TYPE ASCIZ STRING
3216 006232 000417 BR ,64$ ;;GET OVER THE ASCIZ
3217 ;;65$: .ASCIZ <15><12>/HOW MANY RH TO BE TESTED /
3218 64$:
3219 006272 104401 006300 TYPE ,67$ ;;TYPE ASCIZ STRING
3220 006276 000402 BR ,66$ ;;GET OVER THE ASCIZ
3221 ;;67$: .ASCIZ <15><12>/ /
3222 66$:
3223 006304 RDOCT
3224 006306 104410 MOV (SP),LTRY ;;GET NUMBER OF RH TO BE TESTED
3225 006312 011637 005320 MOV (SP)+,TMP0 ;;WORKING NUMBER OF RH
3226 006316 012637 005250 MOV #1,TMP1 ;;RH NUMBER TO BE TYPED
3227 006324 005000 000001 005252 CLR RO ;;INDEX TO WORKING TABLE
3228
3229 006326 3$:
3230 006326 104401 006334 TYPE ,69$ ;;TYPE ASCIZ STRING
3231 006332 000417 BR ,68$ ;;GET OVER THE ASCIZ
3232 ;;69$: .ASCIZ <15><12>/TYPE BASE ADDRESS OF RH NO/
3233 68$:
3234 006372 MOV TMP1,-(SP)
3235 006372 013746 005252 TYPDS
3236 006376 104405 TYPE ,71$ ;;TYPE ASCIZ STRING
3237 006400 104401 006406 BR ,70$ ;;GET OVER THE ASCIZ
3238 71$: .ASCIZ <15><12>/ /
3239 70$:
3240 006412 RDOCT
3241 006412 104410 MOV (SP)+,DEVPNT(RO) ;;FILL WORKING TABLE WITH BASE ADDRESS
3242 006420 012660 005300 TYPE ,73$ ;;TYPE ASCIZ STRING
3243 006424 000420 006426 BR ,72$ ;;GET OVER THE ASCIZ
3244 ;;73$: .ASCIZ <15><12>/TYPE VECTOR ADDRESS OF RH NO/
3245 72$:
3246 006466 MOV TMP1,-(SP)
3247 006466 013746 005252 TYPDS
3248 006472 104405 TYPE ,CRLF
3249 006500 104410 RDOCT
3250 006502 012660 005310 MOV (SP)+,VECPNT(RO) ;;FILL WORKING TABLE WITH VECTOR ADDRESS
3251 006506 005720 TST (RO)+ ;;ADD 2 TO RO
3252 006510 005237 005252 INC TMP1 ;;GET NEXT RH NUMBER
3253 006514 005337 005250 DEC TMP0 ;;COUNT DOWN RH TO BE TESTED
3254 006520 001302 BNE 3$ ;;BRANCH IF ALL NOT DONE
3255
3256 ;RO WILL HAVE DEVICE POINTER, R1 WILL HAVE VECTOR POINTER
3257 006522 012700 005300 SETSIZ: MOV #DEVPNT,RO ;;DEVICE POINTER
3258 006526 012701 005310 MOV #VECPNT,R1 ;;VECTOR PMINTER

```

```

3259 006532 005037 005322 CLR TFOUND ;WILL BE USED AS POINTER
3260 006536 ;TO FOUND TABLE.
3261 006536 012737 010530 000004 SS2: MOV #SS4,ERRVEC ;TIME OUT VECTOR TO REDUCE LTRY
3262 006544 012037 005324 1$: MOV (R0)+,TESTDV ;ADDRESS OF DEVICE TO BE TESTED
3263 006550 012137 005326 MOV (R1)+,TESTVC ;VECTOR TO BE TESTED
3264 006554 005777 176544 TST @TESTDV ;TRY TIME OUT ON DEVICE BASE
3265 ;IF NO TIME OUT OCCURS THEN DEVICE IN TESTDV IS PRESENT
3266 ;NOW FILL ALL REGISTER ADDRESS TABLE
3267 006560 012702 000026 MOV #22,R2 ;COUNT 22
3268 006564 012703 004060 MOV #RHCS1,R3 ;GET BASE LOCATION
3269 006570 013704 005324 MOV TESTDV,R4 ;GET BASE ADDRESS
3270 006574 010423 2$: MOV R4,(R3)+ ;FILL PROPER ADDRESS
3271 006576 062704 000002 ADD #2,R4 ;INCREMENT ADDRESS BY 2
3272 006602 005302 DEC R2 ;COUNT DOWN
3273 006604 001373 BNE 2$ ;BRANCH IF 22 NOT DONE.
3274 006606 005077 175256 CLR @RHCS2 ;SET UNIT NUMBER
3275 006612 SS6:
3276 006612 012737 005330 005360 3$: MOV #TYPNT,POINTT ;POINTER TO DRIVE TYPE
3277 006620 012737 000014 005362 MOV #14,NTYPNT ;NUMBER OF DRIVE TYPES
3278
3279 006626 SS7:
3280 006626 022737 000001 005362 4$: CMP #1,NTYPNT ;IS IT FOR TU
3281 006634 001424 BEQ 5$ ;IF FOR TU BRANCH
3282 006636 017737 175244 001126 MOV @RHDT,$BDDAT ;READ DRIVE TYPE
3283 006644 012702 000014 MOV #14,R2 ;GET NUMBER FOR DT NOT DONE
3284 006650 163702 005362 SUB NTYPNT,R2 ;GET DT TO DO
3285 006654 006302 ASL R2 ;MULTIPLY R2 BY 2
3286 006656 026237 005330 001126 CMP TYPNT(R2),$BDDAT ;IS DRIVE TYPE FOUND
3287 006664 001402 BEQ 16$ ;DRIVE TYPE FOUND SO BRANCH
3288 006666 000137 010554 JMP SS8
3289 006672 032777 010000 175170 16$: BIT #NED,@RHCS2 ;IS IT NON EXISTANT DRIVE
3290 006700 001437 BEQ 8$ ;NED NOT SET, DRIVE TYPE FOUND SO BRANCH
3291 ;A DEVICE HAS NOT BEEN FOUND
3292 006702 000137 010546 JMP SS5 ;A DEVICE HAS NOT BEEN FOUND SO BRANCH
3293
3294 ;TRYING THE TU16 DRIVE TYPE
3295 006706 005037 005364 5$: CLR TMPSLV ;SLAVE 0
3296 006712 013777 005364 175172 6$: MOV TMPSLV,@RHCTC ;MOVE SLAVE NUMBER IN TAPE CONTROL
3297 006720 017737 175162 001126 MOV @RHDT,$BDDAT ;READ DRIVE TYPE
3298 006726 032737 002000 001126 BIT #SLVPR,$BDDAT ;IS SLAVE PRESENT
3299 006734 001014 BNE 7$
3300 006736 022737 000007 005364 CMP #7,TMPSLV ;IS 7 DONE
3301 006744 001002 BNE 17$ ;BRANCH IF 7 DONE AND NONE FOUND
3302 006746 000137 010554 JMP SS8
3303 006752 005237 005364 17$: INC TMPSLV ;GET NEXT SLAVE
3304 006756 012777 040000 175074 MOV #TRE,@RHCS1 ;CLEAR CONTROLLER ERRORS
3305 006764 000752 BR 6$ ;TRY NEXT SLAVE
3306 ;A DEVICE HAS BEEN FOUND SAVE SLAVE
3307 006766 013702 005322 7$: MOV TFOUND,R2 ;GET NO OF RH FOUND X2 FOR INDEX
3308 006772 013762 005364 005446 MOV TMPSLV,FSLAVE(R2)
3309 ;A DEVICE HAS BEEN FOUND TEST VECTOR
3310 007000 012737 007014 000004 8$: MOV #9$,ERRVEC ;TIME OUT VECTOR TO REPORT ERROR
3311 007006 005777 176314 TST @TESTVC ;TRY TIME OUT ON VECTOR
3312 007012 000403 BR 10$ ;A VECTOR EXISTS SO CONTINUE
3313 007014 104146 9$: ERROR 146 ;A DEVICE BASE ADDRESS DID NOT
3314 ;TIME OUT BLT ITS CORRESPONDING

```

```

3315                                     ;VECTOR ADDRESS TIMED OUT
3316                                     ;HIT CONTINUE TO REPEAT THIS
3317                                     ;TEST
3318 007016 000000                       HALT
3319 007020 000767                       BR      8$ ;REPEAT THIS TEST
3320                                     ;A DEVICE HAS BEEN FOUND AND VECTOR RESPONDS SO STORE RESULTS
3321 007022 013702 005322 10$:          MOV     TFOUND,R2 ;GET NO OF RH FOUND X2 FOR INDEX
3322 007026 013762 005362 005366      MOV     NTYPNT,FDEVIC(R2) ;DEVICE 1=TU; 2,3,4,5,6,7=RP;
3323                                     ;10,11,12,13,14=RS
3324 007034 017762 175030 005376      MOV     @RHCS2,FUNITN(R2) ;GET RHCS2
3325 007042 042762 177770 005376      BIC     #177770,FUNITN(R2) ;KEEP UNIT NUMBER
3326 007050 017762 175032 005406      MOV     @RHDT,FTYPE(R2) ;DRIVE BYTE
3327 007056 013762 005326 005416      MOV     TESTVC,FVECTR(R2) ;VECTOR
3328 007064 013762 005324 005426      MOV     TESTDV,FBASEA(R2) ;BASE ADDRESS
3329 007072 013762 005322 005436      MOV     TFOUND,FRHNM(R2) ;RH NUMBER X2 MINUS ONE
3330 007100 006262 005436               ASR     FRHNM(R2) ;RH NUMBER MINUS ONE
3331 007104 005262 005436               INC     FRHNM(R2) ;RH NUMBER
3332 007110 022762 000001 005366      CMP     #1,FDEVIC(R2) ;IS IT TU
3333 007116 001016                       BNE     11$ ;BRANCH IF NOT TU
3334 007120 012762 041710 005456      MOV     #CMNDTM,FDRIVR(R2) ;DRIVER ADDRESS FOR TU
3335 007126 013746 005324               MOV     TESTDV,-(SP) ;BASE ADDRESS
3336 007132 062716 000034               ADD     #(<2*14.>),(SP) ;15TH ADDRESS IS BAE FOR TU
3337 007136 011662 005466               MOV     (SP),FBAE(R2) ;SAVE BAE FOR TU
3338 007142 062716 000002               ADD     #2,(SP) ;16TH ADDRESS IS CS3 FOR TU
3339 007146 012662 005476               MOV     (SP)+,FCS3(R2) ;SAVE CS3 FOR TU
3340
3341 007152 000511                       BR      SS1
3342 007154 022762 000002 005366 11$:  CMP     #2,FDEVIC(R2) ;IS IT RP DUAL PORT
3343 007162 001424                       BEQ     12$ ;BRANCH IF RP FOUND
3344 007164 022762 000003 005366      CMP     #3,FDEVIC(R2) ;IS IT RP
3345 007172 001420                       BEQ     12$ ;BRANCH IF RP FOUND
3346 007174 022762 000004 005366      CMP     #4,FDEVIC(R2) ;IS IT RP DUAL PORT
3347 007202 001414                       BEQ     12$ ;BRANCH IF RP FOUND
3348 007204 022762 000005 005366      CMP     #5,FDEVIC(R2) ;IS IT RP
3349 007212 001410                       BEQ     12$ ;BRANCH IF RP FOUND
3350 007214 022762 000006 005366      CMP     #6,FDEVIC(R2) ;IS IT RP DUAL PORT
3351 007222 001404                       BEQ     12$ ;BRANCH IF RP FOUND
3352 007224 022762 000007 005366      CMP     #7,FDEVIC(R2) ;IS IT RP
3353 007232 001016                       BNE     13$ ;BRANCH IF NO RP
3354 007234 012762 041572 005456 12$:  MOV     #CMNDRP,FDRIVR(R2) ;DRIVER ADDRESS FOR RP
3355 007242 013746 005324               MOV     TESTDV,-(SP) ;BASE ADDRESS
3356 007246 062716 000050               ADD     #(<2*20.>),(SP) ;21ST ADDRESS IS BAE FOR RP
3357 007252 011662 005466               MOV     (SP),FBAE(R2) ;SAVE BAE FOR RP
3358 007256 062716 000002               ADD     #2,(SP) ;22ND ADDRESS IS CS3 FOR RP
3359 007262 012662 005476               MOV     (SP)+,FCS3(R2) ;SAVE CS3 FOR RP
3360
3361 007266 000443                       BR      SS1
3362 007270 022762 000010 005366 13$:  CMP     #10,FDEVIC(R2) ;IS IT RS03/LA
3363 007276 001420                       BEQ     14$ ;BRANCH IF RS03/LA
3364 007300 022762 000011 005366      CMP     #11,FDEVIC(R2) ;IS IT RS04/L
3365 007306 001414                       BEQ     14$ ;BRANCH IF RS04/L
3366 007310 022762 000012 005366      CMP     #12,FDEVIC(R2) ;IS IT RS04
3367 007316 001410                       BEQ     14$ ;BRANCH IF RS04
3368 007320 022762 000013 005366      CMP     #13,FDEVIC(R2) ;IS IT RS03/L
3369 007326 001404                       BEQ     14$ ;BRANCH IF RS03/L
3370 007330 022762 000014 005366      CMP     #14,FDEVIC(R2) ;IS IT RS03

```

```

3371 007336 001016      BNE      15$      ;BRANCH IF NOT RS03
3372 007340 012762 041502 005456 14$:  MOV      #CMNDRS, FDRIVR(R2) ; DRIVER ADDRESS FOR RS
3373 007346 013746 005324      MOV      TESTDV, -(SP)      ; BASE ADDRESS
3374 007352 062716 000030      ADD      #<2*12>, (SP)      ; 13TH ADDRESS IS BAE FOR RS
3375 007356 011662 005466      MOV      (SP), FBAE(R2)      ; SAVE BAE FOR RS
3376 007362 062716 000002      ADD      #2, (SP)          ; 14TH ADDRESS IS CS3 FOR RS
3377 007366 012662 005476      MOV      (SP)+, FCS3(R2)     ; SAVE CS3 FOR RS
3378 007372 000401      BR       SS1             ; CONTINUE
3379 007374 000000      15$:  HALT                ; PROGRAM ERROR
3380
3381
3382
3383
3384 007376 104401 007404      SS1:  TYPE      65$          ;; TYPE ASCIZ STRING
3385 007402 000406      BR       64$          ;; GET OVER THE ASCIZ
3386
3387 007420      ;;65$: .ASCIZ <15><12>/ON RH NO/
3388 007420 016246 005436      64$:  MOV      FRHNM(R2), -(SP) ; TYPE RH NUMBER
3389 007424 104405      TYPDS
3390 007426 104401 007434      TYPE      67$          ;; TYPE ASCIZ STRING
3391 007432 000411      BR       66$          ;; GET OVER THE ASCIZ
3392
3393      ;;67$: .ASCIZ / BASE /
3394 007456 016246 005426      66$:  MOV      FBASEA(R2), -(SP) ; TYPE BASE ADDRESS
3395 007462 104402      TYPOC
3396 007464 104401 007472      TYPE      69$          ;; TYPE ASCIZ STRING
3397 007470 000404      BR       68$          ;; GET OVER THE ASCIZ
3398
3399      ;;69$: .ASCIZ / FOUND/
3400 007502 022762 000001 005366      68$:  CMP      #1, FDEVIC(R2)    ; IS IT TU
3401 007510 001020      BNE      1$           ; BRANCH IF NOT TU
3402 007512 104401 007520      TYPE      71$          ;; TYPE ASCIZ STRING
3403 007516 000410      BR       70$          ;; GET OVER THE ASCIZ
3404
3405      ;;71$: .ASCIZ / TU16 AT SLAVE/
3406 007540 016246 005446      70$:  MOV      FSLAVE(R2), -(SP) ; TYPE SLAVE NUMBER
3407 007544 104405      TYPDS
3408 007546 000137 010346      JMP      6$
3409
3410
3411
3412
3413
3414
3415 007552 022762 000002 005366 1$:  CMP      #2, FDEVIC(R2)    ; IS IT RPO6 DUAL PORT
3416 007560 001016      BNE      2$           ; BRANCH IF NOT RPO6 DUAL PORT
3417 007562 104401 007570      TYPE      73$          ;; TYPE ASCIZ STRING
3418 007566 000411      BR       72$          ;; GET OVER THE ASCIZ
3419
3420      ;;73$: .ASCIZ / RPO6 DUAL PORT /
3421 007612 000137 010346      72$:  JMP      6$
3422 007616 022762 000003 005366 2$:  CMP      #3, FDEVIC(R2)    ; IS IT RPO6 SINGLE PORT
3423 007624 001016      BNE      3$           ; BRANCH IF NOT RPO6 SINGLE PORT
3424 007626 104401 007634      TYPE      75$          ;; TYPE ASCIZ STRING
3425 007632 000411      BR       74$          ;; GET OVER THE ASCIZ
3426
3427      ;;75$: .ASCIZ / RPO6 SINGLE PORT /

```

```

3427 007656          74$:
3428 007656 000137 010346      JMP      6$
3429 007662 022762 000004 005366 3$:  CMP      #4,FDEVIC(R2)  ;IS IT RPO5 DUAL PORT
3430 007670 001020          BNE      4$              ;BRANCH IF NOT RPO5 DUAL PORT
3431 007672 012762 000002 005366      MOV      #2,FDEVIC(R2)
3432 007700 104401 007706          TYPE     77$           ;;TYPE ASCIZ STRING
3433 007704 000410          BR       76$           ;;GET OVER THE ASCIZ
3434          ;;77$: .ASCIZ / RPO5 DUAL PORT/
3435 007726          76$:
3436 007726 000137 010346      JMP      6$
3437 007732 022762 000005 005366 4$:  CMP      #5,FDEVIC(R2)  ;IS IT RPO5 SINGLE PORT
3438 007740 001020          BNE      8$              ;BRANCH IF NOT RPO5 SINGLE PORT
3439 007742 012762 000003 005366      MOV      #3,FDEVIC(R2)
3440 007750 104401 007756          TYPE     79$           ;;TYPE ASCIZ STRING
3441 007754 000411          BR       78$           ;;GET OVER THE ASCIZ
3442          ;;79$: .ASCIZ / RPO5 SINGLE PORT/
3443          78$:
3444 010000          BR       6$
3445 010002 022762 000006 005366 8$:  CMP      #6,FDEVIC(R2)  ;IS IT RPO4 DUAL PORT
3446 010010 001017          BNE      9$              ;BRANCH IF NOT RPO4 DUAL PORT
3447 010012 012762 000002 005366      MOV      #2,FDEVIC(R2)
3448 010020 104401 010026          TYPE     81$           ;;TYPE ASCIZ STRING
3449 010024 000410          BR       80$           ;;GET OVER THE ASCIZ
3450          ;;81$: .ASCIZ / RPO4 DUAL PORT/
3451 010046          80$:
3452 010046 000537          BR       6$
3453 010050 022762 000007 005366 9$:  CMP      #7,FDEVIC(R2)  ;IS IT RPO4 SINGLE PORT
3454 010056 001020          BNE     10$             ;BRANCH IF NOT RPO4 SINGLE PORT
3455 010060 012762 000003 005366      MOV      #3,FDEVIC(R2)
3456 010066 104401 010074          TYPE     83$           ;;TYPE ASCIZ STRING
3457 010072 000411          BR       82$           ;;GET OVER THE ASCIZ
3458          ;;83$: .ASCIZ / RPO4 SINGLE PORT/
3459          82$:
3460 010116          BR       6$
3461 010120 022762 000010 005366 10$: CMP      #10,FDEVIC(R2) ;IS IT RS03-LA
3462 010126 001014          BNE     11$             ;BRANCH IF NOT RS03-LA
3463 010130 012762 000005 005366      MOV      #5,FDEVIC(R2)
3464 010136 104401 010144          TYPE     85$           ;;TYPE ASCIZ STRING
3465 010142 000405          BR       84$           ;;GET OVER THE ASCIZ
3466          ;;85$: .ASCIZ / RS03-LA/
3467          84$:
3468 010156          BR       6$
3469 010156 000473          BR       6$
3470 010160 022762 000011 005366 11$: CMP      #11,FDEVIC(R2) ;IS IT RS04-L
3471 010166 001013          BNE     12$             ;BRANCH IF NOT RS04-L
3472 010170 012762 000004 005366      MOV      #4,FDEVIC(R2)
3473 010176 104401 010204          TYPE     87$           ;;TYPE ASCIZ STRING
3474 010202 000404          BR       86$           ;;GET OVER THE ASCIZ
3475          ;;87$: .ASCIZ / RS04-L/
3476          86$:
3477 010214          BR       6$
3478 010214 000454          BR       6$
3479 010216 022762 000012 005366 12$: CMP      #12,FDEVIC(R2) ;IS IT RS04
3480 010224 001012          BNE     13$             ;BRANCH IF NOT RS04
3481 010226 012762 000004 005366      MOV      #4,FDEVIC(R2)
3482 010234 104401 010242          TYPE     89$           ;;TYPE ASCIZ STRING
3483 010240 000403          BR       88$           ;;GET OVER THE ASCIZ
3484          ;;89$: .ASCIZ / RS04/

```



```

3483 010250 88$:
3484 010250 000436 BR 6$
3485 010252 022762 000013 005366 13$: CMP #13,FDEVIC(R2) :IS IT RS03-L
3486 010260 001013 BNE 14$ :BRANCH IF NOT RS03-L
3487 010262 012762 000005 005366 MOV #5,FDEVIC(R2)
3488 010270 104401 010276 TYPE 91$ ::TYPE ASCIZ STRING
3489 010274 000404 BR 90$ ::GET OVER THE ASCIZ
3490
3491 010306 ::91$: .ASCIZ / RS03-L/
3492 010306 000417 90$:
3493 010310 022762 000014 005366 14$: BR 6$
3494 010316 001012 BNE 5$ :IS IT RS03
3495 010320 012762 000005 005366 MOV #5,FDEVIC(R2) :BRANCH IF NOT RS03
3496 010326 104401 010334 TYPE 93$ ::TYPE ASCIZ STRING
3497 010332 000403 BR 92$ ::GET OVER THE ASCIZ
3498
3499 010342 ::93$: .ASCIZ / RS03/
3500 010342 000401 92$:
3501 010344 000000 BR 6$
3502 010346 5$: HALT :PROGRAM ERROR
3503 010346 104401 010354 6$:
3504 010352 000411 TYPE 95$ ::TYPE ASCIZ STRING
3505
3506 010376 94$: .ASCIZ <15><12>/AT UNIT NUMBER/
3507 010376 016246 005376 MOV FUNITN(R2),-(SP) ;TYPE UNIT NUMBER
3508 010402 104405 TYPDS
3509 010404 104401 010412 TYPE 97$ ::TYPE ASCIZ STRING
3510 010410 000411 BR 96$ ::GET OVER THE ASCIZ
3511
3512 010434 ::97$: .ASCIZ / VECTOR /
3513 010434 016246 005416 96$:
3514 010440 104402 MOV FVECTR(R2),-(SP) ;TYPE VECTOR
3515 010442 104401 010450 TYPDC
3516 010446 000402 TYPE 99$ ::TYPE ASCIZ STRING
3517
3518 010454 98$: .ASCIZ <15><12>/
3519 010454 104401 010462 BR 101$ ::TYPE ASCIZ STRING
3520 010460 000402 100$:
3521
3522 010466 ::101$: .ASCIZ <15><12>/
3523
3524
3525 010466 062737 000002 005322 :UPDATE TFOUND
3526 ADD #2,TFOUND :GET READY FOR NEXT DEVICE
3527
3528 010474 005337 005320 :ALL UNITS DONE? IF LTRY BECOMES ZERO YES ALL DONE
3529 010500 001402 DEC LTRY :REDUCE DEVICES LEFT TO TRY
3530 010502 000137 006536 BEQ 7$ :BRANCH IF ALL COMPLETE
3531 010506 7$: JMP 55$ :ALL NOT DONE
3532 010506 013746 005322 55$: MOV TFOUND,-(SP) ;GET TWICE NUMBER OF RH FOUR
3533 010512 006216 ASR (SP) ;DIVIDE BY 2
3534 010514 011637 005506 MOV (SP),TSTUNT
3535 010520 012637 005510 MOV (SP)+,WCRUNT
3536 010524 000137 010632 JMP TST2 ;JUMP TO NEXT TEST
3537
3538

```

:A RH TIMED OUT SO TRY NEXT UNIT OR END TRYING

```

3539 010530 005337 005320          SS4:  DEC      LTRY          :RH TIMES OUT SO DECREASE RH TYPED
3540 010534 001402                    BEG      1$           :BRANCH IF ALL DONE
3541 010536 000137 006536          SS4:  JMP      SS2           :ALL NOT DONE SO JUMP.
3542 010542 000137 010506          1$:   JMP      SS3           :ALL DONE SO STORE NUMBER OF RH FOUND
3543
3544
3545 010546 012777 040000 173304  SS5:  MOV      #TRE, @RHCS1 :A DRIVE TYPE DID NOT MATCH SO TRY NEXT
3546 010554 005337 005362          SS8:  DEC      NTYPNT      :CLEAR NED ERROR
3547 010560 001402                    BEG      1$           :DECREASE NUMBER OF DRIVE TYPES TRIED
3548 010562 000137 006626          SS7:  JMP      SS7           :BRANCH IF ALL DONE
3549 010566 005277 173276          1$:   INC      @RHCS2      :ALL NOT DONE
3550 010572 012777 040000 173260  SS7:  MOV      #TRE, @RHCS1 :GET NEXT UNIT NO.
3551 010600 017746 173264          SS7:  MOV      @RHCS2, -(SP) :CLEAR NED ERROR
3552 010604 042716 177770          SS7:  BIC      #177770, (SP) :GET RHCS2
3553 010610 022726 000010          SS7:  CMP      #8., (SP)+ :GET UNIT NO
3554 010614 001402                    BEG      2$           :ARE 7 DONE
3555 010616 000137 006612          SS6:  JMP      SS6           :BRANCH IF 7 DONE
3556
3557 010622 104147                    2$:   ERROR  147         :7 NOT DONE SO TRY NEXT UNIT NO
3558 010624 000000                    HALT
3559
3560 010626 000137 010530          SS4:  JMP      SS4           :DEVICE ADDRESS DID NOT TIME
3561
3562
3563
3564
3565
3566
3567
3568
3569
3570
3571
3572
3573 010632 000004                    :*****
3574 010634 012737 000001 001212  :*TEST 2          UNIT UNDER TEST
3575 010642 012737 037610 000004  :* THIS TYPES THE UNIT TO BE TESTED
3576 010650 012737 000340 000006  :* AND IS THE FIRST TEST AFTER THE END OF THE PROGRAM
3577
3578 010656 013746 005510          TST2: SCOPE
3579 010662 013702 005506          MOV      #1 $TIMES ;:DO 1 ITERATION
3580 010666 162602                    MOV      #TIMEOUT, @ERRVEC ;TIMEOUT VECTOR
3581 010670 006302                    MOV      #340, @ERRVEC+2 ;PRIORITY 7
3582 010672 104401 010700          MOV      WORUNT, -(SP) ;GET WORKING RH NUMBER LEFT
3583 010676 000402                    MOV      TSTUNT, R2 ;GET TOTAL NO OF RH FOUND
3584
3585 010704                    SUB      (SP)+, R2 ;NO OF RH TO BE TESTED
3586 010704 104401 010712          ASL      R2 ;INDEX FOR RH TO BE TESTED.
3587 010710 000410                    TYPE     65$ ;:TYPE ASCIZ STRING
3588
3589 010732                    BR       64$ ;:GET OVER THE ASCIZ
3590 010732 016246 005436          ;;65$: .ASCIZ <15><12>/ /
3591 010736 104405                    64$:  TYPE     67$ ;:TYPE ASCIZ STRING
3592
3593 010740 104401 010746          BR       66$ ;:GET OVER THE ASCIZ
3594 010744 000410                    ;;67$: .ASCIZ <15><12>/TESTING RH NO/
3595
3596
3597
3598
3599
3600
3601
3602
3603
3604
3605
3606
3607
3608
3609
3610
3611
3612
3613
3614
3615
3616
3617
3618
3619
3620
3621
3622
3623
3624
3625
3626
3627
3628
3629
3630
3631
3632
3633
3634
3635
3636
3637
3638
3639
3640
3641
3642
3643
3644
3645
3646
3647
3648
3649
3650
3651
3652
3653
3654
3655
3656
3657
3658
3659
3660
3661
3662
3663
3664
3665
3666
3667
3668
3669
3670
3671
3672
3673
3674
3675
3676
3677
3678
3679
3680
3681
3682
3683
3684
3685
3686
3687
3688
3689
3690
3691
3692
3693
3694
3695
3696
3697
3698
3699
3700
3701
3702
3703
3704
3705
3706
3707
3708
3709
3710
3711
3712
3713
3714
3715
3716
3717
3718
3719
3720
3721
3722
3723
3724
3725
3726
3727
3728
3729
3730
3731
3732
3733
3734
3735
3736
3737
3738
3739
3740
3741
3742
3743
3744
3745
3746
3747
3748
3749
3750
3751
3752
3753
3754
3755
3756
3757
3758
3759
3760
3761
3762
3763
3764
3765
3766
3767
3768
3769
3770
3771
3772
3773
3774
3775
3776
3777
3778
3779
3780
3781
3782
3783
3784
3785
3786
3787
3788
3789
3790
3791
3792
3793
3794
3795
3796
3797
3798
3799
3800
3801
3802
3803
3804
3805
3806
3807
3808
3809
3810
3811
3812
3813
3814
3815
3816
3817
3818
3819
3820
3821
3822
3823
3824
3825
3826
3827
3828
3829
3830
3831
3832
3833
3834
3835
3836
3837
3838
3839
3840
3841
3842
3843
3844
3845
3846
3847
3848
3849
3850
3851
3852
3853
3854
3855
3856
3857
3858
3859
3860
3861
3862
3863
3864
3865
3866
3867
3868
3869
3870
3871
3872
3873
3874
3875
3876
3877
3878
3879
3880
3881
3882
3883
3884
3885
3886
3887
3888
3889
3890
3891
3892
3893
3894
3895
3896
3897
3898
3899
3900
3901
3902
3903
3904
3905
3906
3907
3908
3909
3910
3911
3912
3913
3914
3915
3916
3917
3918
3919
3920
3921
3922
3923
3924
3925
3926
3927
3928
3929
3930
3931
3932
3933
3934
3935
3936
3937
3938
3939
3940
3941
3942
3943
3944
3945
3946
3947
3948
3949
3950
3951
3952
3953
3954
3955
3956
3957
3958
3959
3960
3961
3962
3963
3964
3965
3966
3967
3968
3969
3970
3971
3972
3973
3974
3975
3976
3977
3978
3979
3980
3981
3982
3983
3984
3985
3986
3987
3988
3989
3990
3991
3992
3993
3994
3995
3996
3997
3998
3999
4000

```

```

3595      ::69$: .ASCIZ / USING /
3596 010766 68$:
3597 010766 104401 010774      TYPE 71$      ;;TYPE ASCIZ STRING
3598 010772 000404      BR 70$      ;;GET OVER THE ASCIZ
3599      ::71$: .ASCIZ / BASE /
3600 011004 70$:
3601 011004 016246 005426      MOV FBASE(R2),-(SP) ;TYPE BASE ADDRESS
3602 011010 104402      TYPDC
3603 011012 022762 000001 005366      CMP #1,FDEVIC(R2) ;IS IT TU
3604 011020 001017      BNE 1$      ;BRANCH IF NOT TU
3605 011022 104401 011030      TYPE 73$      ;;TYPE ASCIZ STRING
3606 011026 000410      BR 72$      ;;GET OVER THE ASCIZ
3607      ::73$: .ASCIZ / TU16 AT SLAVE/
3608 011050 72$:
3609 011050 016246 005446      MOV FSLAVE(R2),-(SP) ;TYPE SLAVE NUMBER
3610 011054 104405      TYPDS
3611 011056 000467      BR 6$
3612 011060 022762 000002 005366 1$:      CMP #2,FDEVIC(R2) ;IS IT RP DUAL PORT
3613 011066 001014      BNE 2$      ;BRANCH IF NOT RP DUAL PORT
3614 011070 104401 011076      TYPE 75$      ;;TYPE ASCIZ STRING
3615 011074 000410      BR 74$      ;;GET OVER THE ASCIZ
3616      ::75$: .ASCIZ / RP DUAL PORT /
3617 011116 74$:
3618 011116 000447      BR 6$
3619 011120 022762 000003 005366 2$:      CMP #3,FDEVIC(R2) ;IS IT RO SINGLE PORT
3620 011126 001014      BNE 3$      ;BRANCH IF NOT RP
3621 011130 104401 011136      TYPE 77$      ;;TYPE ASCIZ STRING
3622 011134 000410      BR 76$      ;;GET OVER THE ASCIZ
3623      ::77$: .ASCIZ / RP SINGLE PORT/
3624 011156 76$:
3625 011156 000427      BR 6$
3626 011160 022762 000004 005366 3$:      CMP #4,FDEVIC(R2) ;IS IT RS04
3627 011166 001007      BNE 4$      ;BRANCH IF NOT RS04
3628 011170 104401 011176      TYPE 79$      ;;TYPE ASCIZ STRING
3629 011174 000403      BR 78$      ;;GET OVER THE ASCIZ
3630      ::79$: .ASCIZ / RS04/
3631 011204 78$:
3632 011204 000414      BR 6$
3633 011206 022762 000005 005366 4$:      CMP #5,FDEVIC(R2) ;IS IT RS03
3634 011214 001007      BNE 5$      ;BRANCH IF NOT RS03
3635 011216 104401 011224      TYPE 81$      ;;TYPE ASCIZ STRING
3636 011222 000403      BR 80$      ;;GET OVER THE ASCIZ
3637      ::81$: .ASCIZ / RS03/
3638 011232 80$:
3639 011232 000401      BR 6$
3640 011234 000000 5$:      HALT ;PROGRAM ERROR
3641      6$:
3642 011236      TYPE 83$      ;;TYPE ASCIZ STRING
3643 011236 104401 011244      BR 82$      ;;GET OVER THE ASCIZ
3644 011242 000411      ::83$: .ASCIZ <15><12>/AT UNIT NUMBER/
3645      82$:
3646 011266      MOV FUNITN(R2),-(SP) ;TYPE UNIT NUMBER
3647 011266 016246 005376      TYPDS
3648 011272 104405      TYPE 85$      ;;TYPE ASCIZ STRING
3649 011274 104401 011302      BR 84$      ;;GET OVER THE ASCIZ
3650 011300 000411

```

```

3651      ;:85$: .ASCIZ /          VECTOR /
3652      84$:
3653      011324      016246      005416      MOV      FVECTR(R2),-(SP) ;TYPE VECTOR
3654      011330      104402      *YPOC
3655      011332      012703      000024      MOV      #20,R3          ;COUNT 20
3656      011336      012704      004060      MOV      #RHCS1,R4       ;GET BASE LOCATION
3657      011342      016205      005426      MOV      FBASEA(R2),R5   ;GET BASE ADDRESS
3658      011346      010524      7$:      MOV      R5,(R4)+        ;FILL PROPER ADDRESS
3659      011350      062705      000002      ADD      #2,R5          ;INCREMENT BY 2
3660      011354      005303      DEC      R3              ;COUNT DOWN
3661      011356      001373      BNE      7$             ;BRANCH IF 20 NOT DONE
3662
3663      011360      016237      005376      005010      MOV      FUNITN(R2),UNIT ;UNIT NUMBER
3664      011366      016237      005416      005512      MOV      FVECTR(R2),VECTOR ;VECTOR
3665      011374      016237      005446      004660      MOV      FSLAVE(R2),SLAVE ;SLAVE NO
3666      011402      016237      005456      004640      MOV      FDRIVR(R2),COMND ;DRIVER ADDRESS
3667      011410      016237      005466      004130      MOV      FBAE(R2),RHBAE  ;BAE ADDRESS
3668      011416      016237      005476      004132      MOV      FCS3(R2),RHCS3  ;CS3 ADDRESS
3669
3670      011424      005037      045100      CLR      PRITEM
3671
3672
3673
3674
3675
3676
3677
3678
3679
3680      ;:*****
3681      ;*TEST 3          BIT BANG RHCS1
3682
3683      ;*
3684      ;*      TEST LOADING AND READING OF ALL POSSIBLE BITS
3685      ;*      IN RHCS1 REGISTER.
3686      ;*      USE A PATTERN OF WALKING 1'S (1,2,4,10 ETC)
3687      ;*      AND WALKING 0'S (-2,-3,-5 ETC)
3688      ;*      IN THIS TEST SC!TRE!MCPE!DVA!BIT12!BIT10!RDY!GO BITS WILL NOT BE WRITTEN INTO
3689      ;*      AND RDY!DVA BITS WILL ALWAYS BE SET
3690      ;*      AND BIT10 BITS WILL ALWAYS BE CLEARED
3691      ;*      IF SWITCH 14 OR 9 OR 8 ARE SET FOR DEBUGGING
3692      ;*      THEN A RH CLEAR (RHCS2 BIT #5) WILL BE
3693      ;*      GIVEN TO AID SCOPE SYNC'S ON THE CLEAR
3694      ;*      SIGNAL.
3695
3696      ;:*****
3697      ;*ST3:  SCOPE
3698      011430      000004      MOV      #STACK,SP      ;RESET STACK
3699      011432      012706      001000      MOV      #3,@#TSTNM     ;SAVE TEST NUMBER
3700      011436      012737      000003      004656      JSR      PC,@#CLDISK    ;GIVE RH INITIALIZE
3701      ;*
3702      ;*      ;SETUP UNIT NUBER
3703      ;*      ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
3704      011450      012737      176201      004650      MOV      #SC!TRE!MCPE!DVA!BIT12!BIT10!RDY!GO,WRTBIT ;SC!TRE!MCPE!DVA!BIT12!B
3705      011456      012737      004200      004652      MOV      #RDY!DVA,SETBIT ;RDY!DVA ARE BITS ALWAYS SET
3706      011464      012737      002000      004654      MOV      #BIT10,CLRBIT  ;BIT10 ARE BITS ALWAYS CLEARED

```

```

3707
3708
3709 011472 012737 011516 001110      :FLOAT 1'S THRU THE RHCS1 REGISTER
3710 011500 012705 000001              MOV #2$, $LPERR      ;SET LOOP ON ERROR ADDRESS
3711 011504 010537 001124              MOV #1, R5          ;GETTING READY TO FLOAT A ONE
3712 011510 042737 176201 001124 1$: MOV R5, $GDDAT      ;START WITH DATA
3713 011516 032737 041400 001140 2$: BIC #SC!TRE!MCPE!DVA!BIT12!BIT10!RDY!GO, $GDDAT ;CLEAR BITS NOT WRITTEN
3714 011524 001403                      BIT #BIT14!BIT9!BIT8, SWR ;ARE SWITCHES 14 OR 9 OR 8 SET
3715 011526 052777 000040 172334      BEQ 3$              ;BRANCH IF ANY ONE SET
3716                                BIS #CLR, @RHCS2      ;GIVE CLEAR FOR SCOPE
3717 011534 013777 001124 172316 3$: MOV $GDDAT, @RHCS1 ;WRITE RHCS1 REGISTER
3718 011542 017737 172312 001126      MOV @RHCS1, $BDDAT ;READ RHCS1 REGISTER
3719 011550 043737 004654 001124      BIC CLRBIT, $GDDAT ;CLEAR ALWAYS CLEARED BITS
3720 011556 053737 004652 001124      BIS SETBIT, $GDDAT ;SET ALWAYS SET BITS
3721 011564 023737 001124 001126      CMP $GDDAT, $BDDAT ;TEST
3722 011572 001404                      BEQ 4$              ;BRANCH IF GOOD
3723 011574 013737 004060 001122      MOV RHCS1, $BDADR ;GET REGISTER ADDRESS
3724 011602 104137                      ERROR 137           ;ONE WAS BEING FLOATED
3725                                ;THRU RHCS1 REGISTER
3726                                ;ON READING RHCS1 BACK
3727                                ;IT DID NOT CONTAIN WHAT
3728                                ;WAS EXPECTED.
3729 011604 000241 4$: CLC              ;CLEAR CARRY
3730 011606 006305      ASL R5              ;GET 1 ONE LEFT
3731 011610 103335      BCC 1$              ;BRANCH IF 16 NOT DONE
3732
3733                                :FLOAT A ZERO THRU RHCS1 REGISTER.
3734
3735 011612 012737 011636 001110      MOV #6$, $LPERR      ;SET LOOP BACK POINT.
3736 011620 012705 177776              MOV #177776, R5      ;GET READY TO FLOAT A ZERO
3737 011624 010537 001124              MOV R5, $GDDAT       ;GET READY TO WRITE DATA
3738 011630 042737 176201 001124 5$: BIC #SC!TRE!MCPE!DVA!BIT12!BIT10!RDY!GO, $GDDAT ;CLEAR BITS NOT WRITTEN
3739 011636 032737 041400 001140 6$: BIT #BIT14!BIT9!BIT8, SWR ;ARE SWITCHES 14 OR 9 OR 8 SET
3740 011644 001403                      BEQ 7$              ;BRANCH IF ANY OF THE ABOVE SET
3741 011646 012777 000040 172214      MOV #CLR, @RHCS2    ;CLEAR FOR SCOPE AID TO
3742                                ;SYNC IF IN DEBUG
3743 011654 013777 001124 172176 7$: MOV $GDDAT, @RHCS1 ;WRITE INTO RH'XA.
3744 011662 017737 172172 001126      MOV @RHCS1, $BDDAT ;READ RHCS1
3745 011670 053737 004652 001124      BIS SETBIT, $GDDAT ;SET BITS ALWAYS SET
3746 011676 043737 004654 001124      BIC CLRBIT, $GDDAT ;CLEAR BITS ALWAYS 0
3747 011704 023737 001124 001126      CMP $GDDAT, $BDDAT ;TEST
3748 011712 001404                      BEQ 8$              ;BRANCH IF GOOD
3749 011714 013737 004060 001122      MOV RHCS1, $BDADR ;GET REGISTER ADDRESS
3750 011722 104137                      ERROR 137           ;ZERO WAS BEING FLOATED
3751                                ;THRU RHCS1 REGISTER
3752                                ;ON READING IT BACK IT
3753                                ;DID NOT CONTAIN WHAT
3754                                ;WAS EXPECTED
3755
3756 011724 000261 8$: SEC              ;
3757 011726 006105      ROL R5              ;
3758 011730 103735      BCS 5$              ;
3759
3760
3761
3762
::*****
:*TEST 4 BIT BANG RHCS2

```

```

3763
3764
3765
3766
3767
3768
3769
3770
3771
3772
3773
3774
3775
3776 011732 000004
3777 011734 012706 001000
3778 011740 012737 000004 004656
3779
3780 011746 004737 040322
3781
3782
3783
3784 011752 012737 177740 004650
3785 011760 012737 000100 004652
3786
3787
3788 011766 012737 012012 001110
3789 011774 012705 000001
3790 012000 010537 001124 1$:
3791 012004 042737 177740 001124
3792 012012 032737 041400 001140 2$:
3793 012020 001403
3794 012022 052777 000040 172040
3795
3796 012030 013777 001124 172032 3$:
3797 012036 017737 172026 001126
3798 012044 053737 004652 001124
3799 012052 023737 001124 001126
3800 012060 001404
3801 012062 013737 004070 001122
3802 012070 104137
3803
3804
3805
3806
3807 012072 000241 4$:
3808 012074 006305
3809 012076 103340
3810
3811
3812
3813 012100 012737 012124 001110
3814 012106 012705 177776
3815 012112 010537 001124 5$:
3816 012116 042737 177740 001124
3817 012124 032737 041400 001140 6$:
3818 012132 001403

```

```

;* TEST LOADING AND READING OF ALL POSSIBLE BITS
;* IN RHCS2 REGISTER.
;* USE A PATTERN OF WALKING 1'S (1,2,4,10 ETC)
;* AND WALKING 0'S (-2,-3,-5 ETC)
;* IN THIS TEST 177740 BITS WILL NOT BE WRITTEN INTO
;* AND IR BITS WILL ALWAYS BE SET
;* IF SWITCH 14 OR 9 OR 8 ARE SET FOR DEBUGGING
;* THEN A RH CLEAR (RHCS2 BIT #5) WILL BE
;* GIVEN TO AID SCOPE SYNC'S ON THE CLEAR
;* SIGNAL.
;*****
TST4: SCOPE
MOV #STACK,SP ;RESET STACK
MOV #4,@TSTNM ;SAVE TEST NUMBER
JSR PC,@#CLDISK ;GIVE RH INITIALIZE
;SETUP UNIT NUMBER
;CLEAR RHWC AND FUNCTION BITS IN RHCS1
MOV #177740,WRTBIT ;177740 ARE BITS NOT WRITTEN INTO
MOV #IR,SETBIT ;IR ARE BITS ALWAYS SET
;FLOAT 1'S THRU THE RHCS2 REGISTER
MOV #2,$LPERR ;SET LOOP ON ERROR ADDRESS
MOV #1,R5 ;GETTING READY TO FLOAT A ONE
MOV R5,$GDDAT ;START WITH DATA
BIC #177740,$GDDAT ;CLEAR BITS NOT WRITTEN INTO
BIT #BIT14!BIT9!BIT8,SWR ;ARE SWITCHES 14 OR 9 OR 8 SET
BEQ 3$ ;BRANCH IF ANY ONE SET
BIS #CLR,@RHCS2 ;GIVE CLEAR FOR SCOPE
;SYNC IF IN DEBUG:
MOV $GDDAT,@RHCS2 ;WRITE RHCS2 REGISTER
MOV @RHCS2,$BDDAT ;READ RHCS2 REGISTER
BIS SETBIT,$GDDAT ;SET ALWAYS SET BITS
CMP $GDDAT,$BDDAT ;TEST
BEQ 4$ ;BRANCH IF GOOD
MOV RHCS2,$BDDADR ;GET REGISTER ADDRESS
ERROR 137 ;ONE WAS BEING FLOATED
;THRU RHCS2 REGISTER
;ON READING RHCS2 BACK
;IT DID NOT CONTAIN WHAT
;WAS EXPECTED.
CLC ;CLEAR CARRY
ASL R5 ;GET 1 ONE LEFT
BCC 1$ ;BRANCH IF 16 NOT DONE
;FLOAT A ZERO THRU RHCS2 REGISTER.
MOV #6,$LPERR ;SET LOOP BACK POINT.
MOV #177776,R5 ;GET READY TO FLOAT A ZERO
MOV R5,$GDDAT ;GET READY TO WRITE DATA
BIC #177740,$GDDAT ;CLEAR BITS NOT WRITTEN INTO
BIT #BIT14!BIT9!BIT8,SWR ;ARE SWITCHES 14 OR 9 OR 8 SET
BEQ 7$ ;BRANCH IF ANY OF THE ABOVE SET

```



```

3875 012316 013777 001124 171606 3$: MOV $GDDAT, @RHCS3 ;WRITE RHCS3 REGISTER
3876 012324 017737 171602 001126 MOV @RHCS3, $BDDAT ;READ RHCS3 REGISTER
3877 012332 043737 004654 001124 BIC CLRBIT, $GDDAT ;CLEAR ALWAYS CLEARED BITS
3878 012340 053737 004652 001124 BIS SETBIT, $GDDAT ;SET ALWAYS SET BITS
3879 012346 023737 001124 001126 CMP $GDDAT, $BDDAT ;TEST
3880 012354 001404 BEQ 4$ ;BRANCH IF GOOD
3881 012356 013737 004132 001122 MOV RHCS3, $BDADR ;GET REGISTER ADDRESS
3882 012364 104137 ERROR 137 ;ONE WAS BEING FLOATED
3883 ;THRU RHCS3 REGISTER
3884 ;ON READING RHCS3 BACK
3885 ;IT DID NOT CONTAIN WHAT
3886 ;WAS EXPECTED.
3887 012366 000241 4$: CLC ;CLEAR CARRY
3888 012370 006305 ASL R5 ;GET 1 ONE LEFT
3889 012372 103335 BCC 1$ ;BRANCH IF 16 NOT DONE
3890 ;
3891 ;FLOAT A ZERO THRU RHCS3 REGISTER.
3892 ;
3893 012374 012737 012420 001110 MOV #6$, $LPERR ;SET LOOP BACK POINT.
3894 012402 012705 177776 MOV #177776, R5 ;GET READY TO FLOAT A ZERO
3895 012406 010537 001124 5$: MOV R5, $GDDAT ;GET READY TO WRITE DATA
3896 012412 042737 177660 001124 BIC #177660, $GDDAT ;CLEAR BITS NOT WRITTEN INTO
3897 012420 032737 041400 001140 6$: BIT #BIT14!BIT9!BIT8, SWR ;ARE SWITCHES 14 OR 9 OR 8 SET
3898 012426 001403 BEQ 7$ ;BRANCH IF ANY OF THE ABOVE SET
3899 012430 012777 000040 171432 MOV #CLR, @RHCS2 ;CLEAR FOR SCOPE AID TO
3900 ;SYNC IF IN DEBUG
3901 012436 013777 001124 171466 7$: MOV $GDDAT, @RHCS3 ;WRITE INTO RH'XA.
3902 012444 017737 171462 001126 MOV @RHCS3, $BDDAT ;READ RHCS3
3903 012452 053737 004652 001124 BIS SETBIT, $GDDAT ;SET BITS ALWAYS SET
3904 012460 043737 004654 001124 BIC CLRBIT, $GDDAT ;CLEAR BITS ALWAYS 0
3905 012466 023737 001124 001126 CMP $GDDAT, $BDDAT ;TEST
3906 012474 001404 BEQ 8$ ;BRANCH IF GOOD
3907 012476 013737 004132 001122 MOV RHCS3, $BDADR ;GET REGISTER ADDRESS
3908 012504 104137 ERROR 137 ;ZERO WAS BEING FLOATED
3909 ;THRU RHCS3 REGISTER
3910 ;ON READING IT BACK IT
3911 ;DID NOT CONTAIN WHAT
3912 ;WAS EXPECTED
3913 ;
3914 012506 000261 8$: SEC
3915 012510 006105 ROL R5
3916 012512 103735 BCS 5$
3917 ;
3918 ;
3919 ;*****
3920 ;*TEST 6 BIT BANG RHWC
3921 ;
3922 ;* TEST LOADING AND READING OF ALL POSSIBLE BITS
3923 ;* IN RHWC REGISTER.
3924 ;* USE A PATTERN OF WALKING 1'S (1,2,4,10 ETC)
3925 ;* AND WALKING 0'S (-2, -3, -5 ETC)
3926 ;* IN THIS TEST 0 BITS WILL NOT BE WRITTEN INTO
3927 ;* AND 0 BITS WILL ALWAYS BE SET
3928 ;* IF SWITCH 14 OR 9 OR 8 ARE SET FOR DEBUGGING
3929 ;* THE 1 A RH CLEAR (RHCS2 BIT #5) WILL BE
3930 ;* GIVEN TO AID SCOPE SYNC'S ON THE CLEAR

```



```

3931 ;* SIGNAL.
3932
3933 :*****
3934 012514 000004 tST6: SCOPE
3935 012516 012706 001000 MOV #STACK,SP ;RESET STACK
3936 012522 012737 000006 004656 MOV #6,#TSTNM ;SAVE TEST NUMBER
3937
3938 012530 004737 040322 JSR PC,#CLDISK ;GIVE RH INITIALIZE
3939 ;SETUP UNIT NUBER
3940 ;CLEAR RHC AND FUNCTION BITS IN RHCS1
3941
3942 012534 012737 000000 004650 MOV #0,WRTBIT ;0 ARE BITS NOT WRITTEN INTO
3943 012542 012737 000000 004652 MOV #0,SETBIT ;0 ARE BITS ALWAYS SET
3944
3945 ;FLOAT 1'S THRU THE RHC REGISTER
3946 012550 012737 012574 001110 MOV #2$,SLPERR ;SET LOOP ON ERROR ADDRESS
3947 012556 012705 000001 MOV #1,R5 ;GETTING READY TO FLOAT A ONE
3948 012562 010537 001124 1$: MOV R5,$GDDAT ;START WITH DATA
3949 012566 042737 000000 001124 BIC #0,$GDDAT ;CLEAR BITS NOT WRITTEN INTO
3950 012574 032737 041400 001140 2$: BIT #BIT14!BIT9!BIT8,SWR ;ARE SWITCHES 14 OR 9 OR 8 SET
3951 012602 001403 BEQ 3$ ;BRANCH IF ANY ONE SET
3952 012604 052777 000040 171256 BIS #CLR,#RHCS2 ;GIVE CLEAR FOR SCOPE
3953 ;SYNC IF IN DEBUG:
3954 012612 013777 001124 171242 3$: MOV $GDDAT,#RHWC ;WRITE RHC REGISTER
3955 012620 017737 171236 001126 MOV #RHWC,$BDDAT ;READ RHC REGISTER
3956 012626 053737 004652 001124 BIS SETBIT,$GDDAT ;SET ALWAYS SET BITS
3957 012634 023737 001124 001126 CMP $GDDAT,$BDDAT ;TEST
3958 012642 001404 BEQ 4$ ;BRANCH IF GOOD
3959 012644 013737 004062 001122 MOV RHC,$BDADR ;GET REGISTER ADDRESS
3960 012652 104137 ERROR 137 ;ONE WAS BEING FLOATED
3961 ;THRU RHC REGISTER
3962 ;ON READING RHC BAC
3963 ;IT DID NOT CONTAIN ,HAT
3964 ;WAS EXPECTED.
3965 012654 000241 4$: CLC ;CLEAR CARRY
3966 012656 006305 ASL R5 ;GET 1 ONE LEFT
3967 012680 103340 BCC 1$ ;BRANCH IF 16 NOT DONE
3968
3969 ;FLOAT A ZERO THRU RHC REGISTER.
3970
3971 012662 012737 012706 001110 MOV #6$,SLPERR ;SET LOOP BACK POINT.
3972 012670 012705 177776 MOV #177776,R5 ;GET READY TO FLOAT A ZERO
3973 012674 010537 001124 5$: MOV R5,$GDDAT ;GET READY TO WRITE DATA
3974 012700 042737 000000 001124 BIC #0,$GDDAT ;CLEAR BITS NOT WRITTEN INTO
3975 012706 032737 041400 001140 6$: BIT #BIT14!BIT9!BIT8,SWR ;ARE SWITCHES 14 OR 9 OR 8 SET
3976 012714 001403 BEQ 7$ ;BRANCH IF ANY OF THE ABOVE SET
3977 012716 012777 000040 171144 MOV #CLR,#RHCS2 ;CLEAR FOR SCOPE AID TO
3978 ;SYNC IF IN DEBUG
3979 012724 013777 001124 171130 7$: MOV $GDDAT,#RHWC ;WRITE INTO RH'XA.
3980 012732 017737 171124 001126 MOV #RHWC,$BDDAT ;READ RHC
3981 012740 053737 004652 001124 BIS SETBIT,$GDDAT ;SET BITS ALWAYS SET
3982 012746 023737 001124 001126 CMP $GDDAT,$BDDAT ;TEST
3983 012754 001404 BEQ 8$ ;BRANCH IF GOOD
3984 012756 013737 004062 001122 MOV RHC,$BDADR ;GET REGISTER ADDRESS
3985 012764 104137 ERROR 137 ;ZERO WAS BEING FLOATED
3986 ;THRU RHC REGISTER

```

3987  
3988  
3989  
3990  
3991 012766 000261  
3992 012770 006105  
3993 012772 103740  
3994  
3995  
3996  
3997  
3998  
3999  
4000  
4001  
4002  
4003  
4004  
4005  
4006  
4007  
4008  
4009  
4010  
4011  
4012 012774 000004  
4013 012776 012706 001000  
4014 013002 012737 000007 004656  
4015  
4016 013010 004737 040322  
4017  
4018  
4019  
4020 013014 012737 000000 004650  
4021 013022 012737 000000 004652  
4022 013030 012737 000001 004654  
4023  
4024  
4025 013036 012737 013062 001110  
4026 013044 012705 000001  
4027 013050 010537 001124 1\$:  
4028 013054 042737 000000 001124  
4029 013062 032737 041400 001140 2\$:  
4030 013070 001403  
4031 013072 052777 000040 170770  
4032  
4033 013100 013777 001124 170756 3\$:  
4034 013106 017737 170752 001126  
4035 013114 043737 004654 001124  
4036 013122 053737 004652 001124  
4037 013130 023737 001124 001126  
4038 013136 001404  
4039 013140 013737 004064 001122  
4040 013146 104137  
4041  
4042

: ON READING IT BACK IT  
: DID NOT CONTAIN WHAT  
: WAS EXPECTED

8\$: SEC  
ROL R5  
BCS 5\$

::\*\*\*\*\*  
: \*TEST 7 BIT BANG RHBA

: \* TEST LOADING AND READING OF ALL POSSIBLE BITS  
: \* IN RHBA REGISTER.  
: \* USE A PATTERN OF WALKING 1'S (1,2,4,10 ETC)  
: \* AND WALKING 0'S (-2,-3,-5 ETC)  
: \* IN THIS TEST 0 BITS WILL NOT BE WRITTEN INTO  
: \* AND 0 BITS WILL ALWAYS BE SET  
: \* AND BIT00 BITS WILL ALWAYS BE CLEARED  
: \* IF SWITCH 14 OR 9 OR 8 ARE SET FOR DEBUGGING  
: \* THEN A RH CLEAR (RHCS2 BIT #5) WILL BE  
: \* GIVEN TO AID SCOPE SYNC'S ON THE CLEAR  
: \* SIGNAL.

::\*\*\*\*\*

1ST7: SCOPE  
MOV #STACK,SP ;RESET STACK  
MOV #7,#TSTNM ;SAVE TEST NUMBER  
JSR PC,#CLDISK ;GIVE RH INITIALIZE  
;SETUP UNIT NUBER  
;CLEAR RHWC AND FUNCTION BITS IN RHCS1  
MOV #0,WRTBIT ;0 ARE BITS NOT WRITTEN INTO  
MOV #0,SETBIT ;0 ARE BITS ALWAYS SET  
MOV #BIT00,CLRBIT ;BIT00 ARE BITS ALWAYS CLEARED  
;FLOAT 1'S THRU THE RHBA REGISTER  
MOV #2\$,\$LPERR ;SET LOOP ON ERROR ADDRESS  
MOV #1,R5 ;GETTING READY TO FLOAT A ONE  
1\$: MOV R5,\$GDDAT ;START WITH DATA  
BIC #0,\$GDDAT ;CLEAR BITS NOT WRITTEN INTO  
2\$: BIT #BIT14!BIT9!BIT8,SWR ;ARE SWITCHES 14 OR 9 OR 8 SET  
BEQ 3\$ ;BRANCH IF ANY ONE SET  
BIS #CLR,#RHCS2 ;GIVE CLEAR FOR SCOPE  
;SYNC IF IN DEBUG:  
3\$: MOV \$GDDAT,#RHBA ;WRITE RHBA REGISTER  
MOV #RHBA,\$BDDAT ;READ RHBA REGISTER  
BIC CLRBIT,\$GDDAT ;CLEAR ALWAYS CLEARED BITS  
BIS SETBIT,\$GDDAT ;SET ALWAYS SET BITS  
CMP \$GDDAT,\$BDDAT ;TEST  
BEQ 4\$ ;BRANCH IF GOOD  
MOV RHBA,\$BDDADR ;GET REGISTER ADDRESS  
4\$: ERROR 137 ;ONE WAS BEING FLOATED  
;THRU RHBA REGISTER  
: ON READING RHBA BACK

```

4043                                     ;IT DID NOT CONTAIN WHAT
4044                                     ;WAS EXPECTED.
4045 013150 000241 4$: CLC                                     ;CLEAR CARRY
4046 013152 006305 ASL R5                                     ;GET 1 ONE LEFT
4047 013154 103335 BCC 1$                                     ;BRANCH IF 16 NOT DONE
4048
4049                                     ;FLOAT A ZERO THRU RHBA REGISTER.
4050
4051 013156 012737 013202 001110 MOV #6$, $LPERR ;SET LOOP BACK POINT.
4052 013164 012705 177776 MOV #177776, R5 ;GET READY TO FLOAT A ZERO
4053 013170 010537 001124 5$: MOV R5, $GDDAT ;GET READY TO WRITE DATA
4054 013174 042737 000000 001124 BIC #0, $GDDAT ;CLEAR BITS NOT WRITTEN INTO
4055 013202 032737 041400 001140 6$: BIT #BIT14!BIT9!BIT8, SWR ;ARE SWITCHES 14 OR 9 OR 8 SET
4056 013210 001403 BEQ 7$ ;BRANCH IF ANY OF THE ABOVE SET
4057 013212 012777 000040 170650 MOV #CLR, @RHCS2 ;CLEAR FOR SCOPE AID TO
4058                                     ;SYNC IF IN DEBUG
4059 013220 013777 001124 170636 7$: MOV $GDDAT, @RHBA ;WRITE INTO RH'XA.
4060 013226 017737 170632 001126 MOV @RHBA, $BDDAT ;READ RHBA
4061 013234 053737 004652 001124 BIS SETBIT, $GDDAT ;SET BITS ALWAYS SET
4062 013242 043737 004654 001124 BIC CLRBIT, $GDDAT ;CLEAR BITS ALWAYS 0
4063 013250 023737 001124 001126 CMP $GDDAT, $BDDAT ;TEST
4064 013256 001404 BEQ 8$ ;BRANCH IF GOOD
4065 013260 013737 004064 001122 MOV RHBA, $BDADR ;GET REGISTER ADDRESS
4066 013266 104137 ERROR 137 ;ZERO WAS BEING FLOATED
4067                                     ;THRU RHBA REGISTER
4068                                     ;ON READING IT BACK IT
4069                                     ;DID NOT CONTAIN WHAT
4070                                     ;WAS EXPECTED
4071
4072 013270 000261 8$: SEC
4073 013272 006105 ROL R5
4074 013274 103735 BCS 5$
4075
4076
4077 ;:*****
4078 ;*TEST 10 BIT BANG RHBAE
4079
4080 ;* TEST LOADING AND READING OF ALL POSSIBLE BITS
4081 ;* IN RHBAE REGISTER.
4082 ;* USE A PATTERN OF WALKING 1'S (1,2,4,10 ETC)
4083 ;* AND WALKING 0'S (-2,-3,-5 ETC)
4084 ;* IN THIS TEST 17700 BITS WILL NOT BE WRITTEN INTO
4085 ;* AND 0 BITS WILL ALWAYS BE SET
4086 ;* AND 17700 BITS WILL ALWAYS BE CLEARED
4087 ;* IF SWITCH 14 OR 9 OR 8 ARE SET FOR DEBUGGING
4088 ;* THEN A RH CLEAR (RHCS2 BIT #5) WILL BE
4089 ;* GIVEN TO AID SCOPE SYNC'S ON THE CLEAR
4090 ;* SIGNAL.
4091
4092 ;:*****
4093 013276 000004 †ST10: SCOPE
4094 013300 012706 001000 MOV #STACK, SP ;RESET STACK
4095 013304 012737 000010 004656 MOV #10, @†STNM ;SAVE TEST NUMBER
4096
4097 013312 004737 040322 JSR PC, @CLDISK ;GIVE RH INITIALIZE
4098 ;SETUP UNIT NUMBER

```

```

4099                                     :CLEAR RHWC AND FUNCTION BITS IN RHCS1
4100
4101 013316 012737 177700 004650      MOV    #177700,WRTBIT  :177700 ARE BITS NOT WRITTEN INTO
4102 013324 012737 000000 004652      MOV    #0,SETBIT      :0 ARE BITS ALWAYS SET
4103 013332 012737 177700 004654      MOV    #177700,CLRBIT :177700 ARE BITS ALWAYS CLEARED
4104
4105                                     :FLOAT 1'S THRU THE RHBAE REGISTER
4106 013340 012737 013364 001110      MOV    #2$, $LPERR    :SET LOOP ON ERROR ADDRESS
4107 013346 012705 000001              MOV    #1,R5          :GETTING READY TO FLOAT A ONE
4108 013352 010537 001124              MOV    R5,$GDDAT     :START WITH DATA
4109 013356 042737 177700 001124      BIC    #177700,$GDDAT :CLEAR BITS NOT WRITTEN INTO
4110 013364 032737 041400 001140      BIT    #BIT14!BIT9!BIT8,SWR :ARE SWITCHES 14 OR 9 OR 8 SET
4111 013372 001403 001124              BEQ    3$            :BRANCH IF ANY ONE SET
4112 013374 052777 000040 170466      BIS    #CLR,$RHCS2   :GIVE CLEAR FOR SCOPE
4113                                     :SYNC IF IN DEBUG:
4114 013402 013777 001124 170520      MOV    $GDDAT,$RHBAE :WRITE RHBAE REGISTER
4115 013410 017737 170514 001126      MOV    $RHBAE,$BDDAT :READ RHBAE REGISTER
4116 013416 043737 004654 001124      BIC    CLRBIT,$GDDAT  :CLEAR ALWAYS CLEARED BITS
4117 013424 053737 004652 001124      BIS    SETBIT,$GDDAT  :SET ALWAYS SET BITS
4118 013432 023737 001124 001126      CMP    $GDDAT,$BDDAT :TEST
4119 013440 001404 001124              BEQ    4$            :BRANCH IF GOOD
4120 013442 013737 004130 001122      MOV    RHBAE,$BDADR   :GET REGISTER ADDRESS
4121 013450 104137 001122              ERROR  137          :ONE WAS BEING FLOATED
4122                                     :THRU RHBAE REGISTER
4123                                     :ON READING RHBAE BACK
4124                                     :IT DID NOT CONTAIN WHAT
4125                                     :WAS EXPECTED.
4126 013452 000241 001122              CLC                    :CLEAR CARRY
4127 013454 006305 001122              ASL    R5             :GET 1 ONE LEFT
4128 013456 103335 001122              BCC    1$            :BRANCH IF 16 NOT DONE
4129
4130                                     :FLOAT A ZERO THRU RHBAE REGISTER.
4131
4132 013460 012737 013504 001110      MOV    #6$, $LPERR    :SET LOOP BACK POINT.
4133 013466 012705 177776 001110      MOV    #177776,R5     :GET READY TO FLOAT A ZERO
4134 013472 010537 001124              MOV    R5,$GDDAT     :GET READY TO WRITE DATA
4135 013476 042737 177700 001124      BIC    #177700,$GDDAT :CLEAR BITS NOT WRITTEN INTO
4136 013504 032737 041400 001140      BIT    #BIT14!BIT9!BIT8,SWR :ARE SWITCHES 14 OR 9 OR 8 SET
4137 013512 001403 001124              BEQ    7$            :BRANCH IF ANY OF THE ABOVE SET
4138 013514 012777 000040 170346      MOV    #CLR,$RHCS2   :CLEAR FOR SCOPE AID TO
4139                                     :SYNC IF IN DEBUG
4140 013522 013777 001124 170400      MOV    $GDDAT,$RHBAE :WRITE INTO RH'XA.
4141 013530 017737 170374 001126      MOV    $RHBAE,$BDDAT :READ RHBAE
4142 013536 053737 004652 001124      BIS    SETBIT,$GDDAT  :SET BITS ALWAYS SET
4143 013544 043737 004654 001124      BIC    CLRBIT,$GDDAT  :CLEAR BITS ALWAYS 0
4144 013552 023737 001124 001126      CMP    $GDDAT,$BDDAT :TEST
4145 013560 001404 001124              BEQ    8$            :BRANCH IF GOOD
4146 013562 013737 004130 001122      MOV    RHBAE,$BDADR   :GET REGISTER ADDRESS
4147 013570 104137 001122              ERROR  137          :ZERO WAS BEING FLOATED
4148                                     :THRU RHBAE REGISTER
4149                                     :ON READING IT BACK IT
4150                                     :DID NOT CONTAIN WHAT
4151                                     :WAS EXPECTED
4152
4153 013572 000261 001122              SEC                    :SET CARRY
4154 013574 006105 001122              RCL    R5             :GET 1 ONE LEFT

```

4155 013576 103735  
4156  
4157  
4158  
4159  
4160  
4161  
4162  
4163  
4164  
4165  
4166  
4167  
4168  
4169  
4170  
4171  
4172  
4173  
4174  
4175  
4176  
4177  
4178 013600 000004  
4179 013602 012706 001000  
4180 013606 012737 000011 004656  
4181  
4182 013614 004737 040322  
4183  
4184  
4185  
4186  
4187 013620 012737 000100 001124  
4188 013626 053737 005010 001124  
4189  
4190 013634 017737 170230 001126  
4191 013642 023737 001124 001126  
4192  
4193  
4194 013650 001401  
4195 013652 104006  
4196  
4197  
4198  
4199  
4200  
4201  
4202  
4203  
4204 013654  
4205  
4206  
4207 013654 005077 170222  
4208  
4209  
4210 013660 104411

BCS 5\$

```

*****
:TEST 11      SILO TEST 1 (ONE WORD WRITE)
:
:  AFTER A RH CLEAR IS GIVEN (SET BIT #5 IN RHCS2)
:  RHCS2 IS CHECKED TO HAVE "IR" (BIT #6) HIGH
:  TOGETHER WITH UNIT NUMBER
:  ONE WORD OF ALL ZEROS IS WRITTEN INTO RHDB
:  BY "CLR"
:  "OR" (BIT #7) IS WAITED FOR BY A TRAP INSTRUCTION
:  CALLED "WAT" (NO TIMING IS DONE)
:  HOWEVER IF "OR" DOES NOT SET WITHIN "WAT" COUNT
:  DOWN AN ERROR IS REPORTED
:  RHDB IS READ AND CHECKED TO CONTAIN ZEROS
:  RHCS2 WILL BE CHECKED TO HAVE "IR" AND UNIT NUMBER
:  RHCS1, RHCS3, RHBA, RHBAE, RHWC, WILL BE CHECKED TO HAVE
:  APPROPRIATE VALUES
*****

```

```

†ST11:  SCOPE
        MOV      #STACK, SP      ;RESET STACK
        MOV      #11, @#1STNM    ;SAVE TEST NUMBER
        JSR      PC, @#CLDISK    ;GIVE RH INITIALIZE
        ;SETUP UNIT NUBER
        ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
        ;CHECK THAT RHCS2 HAS IR
        MOV      #IR, $GDDAT     ;GET GOOD = 100
        BIS      UNIT, $GDDAT    ;INCLUDE UNIT NUMBER
        MOV      @RHCS2, $BDDAT  ;READ RHCS2 FOR COMPARISON
        CMP      $GDDAT, $BDDAT  ;COMPARE EXPECTED
        ;DATA WITH DATA READ FROM
        ;RHCS2
        BEQ      65$,             ;BRANCH IF GOOD
        ERROR   6
        ;AFTER SETTING CLR BIT #5
        ;IN RHCS2 TO INIT THE RH
        ;AND HAVING DONE NOTHING ELSE
        ;RHCS2 SHOULD HAVE IR
        ;=100
        ;TOGETHER WITH UNIT NUMBER
        ;BUT CONTAINED WHAT IS
        ;GIVEN IN BAD RHCS2
65$:
        ;WRITE ONE WORD OF ALL ZEROS INTO RHDB
        CLR      @RHDB           ;WRITE 0 IN RHDB
        ;WAIT FOR OR BIT IN RHCS2 REGISTER TO SET
        WAT                    ;TRAP TO WAIT.T SUBROUTINE

```



```

4267 013774 104011      ERROR  11
4268
4269
4270
4271
4272
4273
4274
4275
4276 013776      71$:
4277
4278 013776 012737 000000 001124      :CHECK THAT RHCS3 HAS 0
4279
4280 014004 017737 170122 001126      MOV      @RHCS3,$BDDAT      :GET GOOD = 0
4281 014012 023737 001124 001126      CMP      $GDDAT,$BDDAT      :READ RHCS3 FOR COMPARISON
4282
4283
4284 014020 001401      BEQ      73$      :COMPARE EXPECTED
4285 014022 104012      ERROR  12      :DATA WITH DATA READ FROM
4286
4287
4288
4289
4290
4291
4292
4293
4294
4295 014024      73$:
4296
4297 014024 012737 000000 001124      :CHECK THAT RHBA HAS 0
4298 014032 017737 170026 001126      MOV      @RHBA,$BDDAT      :GET GOOD =
4299 014040 023737 001124 001126      CMP      $GDDAT,$BDDAT      :READ RHBA FOR COMPARISON
4300
4301 014046 001401      BEQ      75$      :COMPARE EXPECTED
4302 014050 104013      ERROR  13      :DATA WITH DATA READ FROM
4303
4304
4305
4306
4307
4308
4309
4310
4311 014052      75$:
4312
4313 014052 012737 000000 001124      :CHECK THAT RHBAE HAS 0
4314
4315 014060 017737 170044 001126      MOV      @RHBAE,$BDDAT      :GET GOOD =
4316 014066 023737 001124 001126      CMP      $GDDAT,$BDDAT      :READ RHBAE FOR COMPARISON
4317
4318
4319 014074 001401      BEQ      77$      :COMPARE EXPECTED
4320 014076 104014      ERROR  14      :DATA WITH DATA READ FROM
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

```

```

4323                                     ;INTO RHDB
4324                                     ;THEN READING IT BACK
4325                                     ;RHBAE SHOULD HAVE 0
4326                                     ;=
4327                                     ;BUT CONTAINED WHAT IS
4328                                     ;GIVEN IN BAD RHBAE
4329 014100          79$:
4330                                     ;CHECK THAT RHWC HAS 0
4331 014100 012737 000000 001124      MOV      #0,$GDDAT      ;GET GOOD =
4332
4333 014106 017737 167750 001126      MOV      @RHWC,$BDDAT  ;READ RHWC FOR COMPARISON
4334 014114 023737 001124 001126      CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED
4335                                     ;DATA WITH DATA READ FROM
4336                                     ;RHWC
4337 014122 001401          BEQ      79$      ;BRANCH IF GOOD
4338 014124 104015          ERROR    15
4339                                     ;AFTER CLEARING THE RH
4340                                     ;AND WRITING ALL ZEROS
4341                                     ;INTO RHDB
4342                                     ;THEN READING IT BACK
4343                                     ;RHWC SHOULD HAVE 0
4344                                     ;=
4345                                     ;BUT CONTAINED WHAT IS
4346                                     ;GIVEN IN BAD RHWC
4347 014126          79$:
4348
4349
4350
4351 ::*****
4352 :*TEST 12      SILO TEST 2 (ONE WORD WRITE)
4353 :*
4354 :* AFTER A RH CLEAR IS GIVEN (SET BIT #5 IN RHCS2)
4355 :* RHCS2 IS CHECKED TO HAVE "IR" (BIT #6) HIGH
4356 :* TOGETHER WITH UNIT NUMBER
4357 :* ONE WORD OF ALL ONES IS WRITTEN INTO RHDB
4358 :* BY "CLR"
4359 :* "OR" (BIT #7) IS WAITED FOR BY A TRAP INSTRUCTION
4360 :* CALLED "WAT" (NO TIMING IS DONE)
4361 :* HOWEVER IF "OR" DOES NOT SET WITHIN "WAT" COUNT
4362 :* DOWN AN ERROR IS REPORTED
4363 :* RHDB IS READ AND CHECKED TO CONTAIN ALL ONES
4364 :* RHCS2 WILL BE CHECKED TO HAVE "IR" AND UNIT NUMBER
4365 :* RHCS1,RHCS3,RHBA,RHBAE,RHWC, WILL BE CHECKED TO HAVE
4366 :* APPROPRIATE VALUES
4367 :******
4367 014126 000004      †ST12: SCOPE
4368 014130 012706      MOV      #STACK.SP      ;RESET STACK
4369 014134 012737 000012 004656      MOV      #12,@†STNM      ;SAVE TEST NUMBER
4370
4371 014142 004737 040322      JSR      PC,@CLDISK      ;GIVE RH INITIALIZE
4372                                     ;SETUP UNIT NUBER
4373                                     ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
4374
4375
4376                                     ;CHECK THAT RHCS2 HAS IR
4377 014146 012737 000100 001124      MOV      #IR,$GDDAT      ;GET GOOD = 100
4378 014154 053737 005010 001124      BIS      UNIT,$GDDAT      ;INCLUDE UNIT NUMBER

```





```

4435                                     ;RHCS2
4436 014274 001401                       BEQ      69$
4437 014276 104010                       ERROR   10
4438                                     ;BRANCH IF GOOD
4439                                     ;AFTER CLEARING THE RH
4440                                     ;AND WRITING ALL ONES
4441                                     ;INTO RHDB
4442                                     ;THEN READING IT BACK
4443                                     ;RHCS2 SHOULD HAVE IR
4444                                     ;=100
4445                                     ;TOGETHER WITH UNIT NUMBER
4446                                     ;BUT CONTAINED WHAT IS
4447 014300                               69$:
4448                                     ;CHECK THAT RHCS3 HAS 0
4449 014300 012737 000000 001124         MOV      #0,$GDDAT ;GET GOOD =
4450
4451 014306 017737 167620 001126         MOV      @RHCS3,$BDDAT ;READ RHCS3 FOR COMPARISON
4452 014314 023737 001124 001126         CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED
4453                                     ;DATA WITH DATA READ FROM
4454                                     ;RHCS3
4455 014322 001401                       BEQ      71$
4456 014324 104012                       ERROR   12
4457                                     ;BRANCH IF GOOD
4458                                     ;AFTER CLEARING THE RH
4459                                     ;AND WRITING ALL ONES
4460                                     ;INTO RHDB
4461                                     ;THEN READING IT BACK
4462                                     ;RHCS3 SHOULD HAVE 0
4463                                     ;=
4464                                     ;BUT CONTAINED WHAT IS
4465 014326                               71$:
4466                                     ;CHECK THAT RHBA HAS 0
4467 014326 012737 000000 001124         MOV      #0,$GDDAT ;GET GOOD =
4468
4469 014334 017737 167524 001126         MOV      @RHBA,$BDDAT ;READ RHBA FOR COMPARISON
4470 014342 023737 001124 001126         CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED
4471                                     ;DATA WITH DATA READ FROM
4472                                     ;RHBA
4473 014350 001401                       BEQ      73$
4474 014352 104013                       ERROR   13
4475                                     ;BRANCH IF GOOD
4476                                     ;AFTER CLEARING THE RH
4477                                     ;AND WRITING ALL ONES
4478                                     ;INTO RHDB
4479                                     ;THEN READING IT BACK
4480                                     ;RHBA SHOULD HAVE 0
4481                                     ;=
4482                                     ;BUT CONTAINED WHAT IS
4483 014354                               73$:
4484                                     ;CHECK THAT RHBAE HAS 0
4485 014354 012737 000000 001124         MOV      #0,$GDDAT ;GET GOOD =
4486
4487 014362 017737 167542 001126         MOV      @RHBAE,$BDDAT ;READ RHBAE FOR COMPARISON
4488 014370 023737 001124 001126         CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED
4489                                     ;DATA WITH DATA READ FROM
4490                                     ;RHBAE

```

```

4491 014376 001401          BEQ      75$      ;BRANCH IF GOOD
4492 01440C 104014          ERROR    14
4493                                ;AFTER CLEARING THE RH
4494                                ;AND WRITING ALL ONES
4495                                ;INTO RHOB
4496                                ;THEN READING IT BACK
4497                                ;RHBAE SHOULD HAVE 0
4498                                ;=
4499                                ;BUT CONTAINED WHAT IS
4500                                ;GIVEN IN BAD RHBAE
4501 014402          75$:
4502                                ;CHECK THAT RHWC HAS 0
4503 014402 012737 000000 001124  MOV      #0,$GDDAT ;GET GOOD =
4504
4505 014410 017737 167446 001126  MOV      @RHWC,$BDDAT ;READ RHWC FOR COMPARISON
4506 014416 023737 001124 001126  CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED
4507                                ;DATA WITH DATA READ FROM
4508                                ;RHWC
4509 014424 001401          BEQ      77$      ;BRANCH IF GOOD
4510 014426 104015          ERROR    15
4511                                ;AFTER CLEARING THE RH
4512                                ;AND WRITING ALL ONES
4513                                ;INTO RHOB
4514                                ;THEN READING IT BACK
4515                                ;RHWC SHOULD HAVE 0
4516                                ;=
4517                                ;BUT CONTAINED WHAT IS
4518                                ;GIVEN IN BAD RHWC
4519 014430          77$:

```

```

4520
4521
4522 ;*****
4523 ;*TEST 13          SILO TEST 3 (TWO WORD WRITE)
4524
4525 ;*
4526 ;* AFTER A RH CLEAR IS GIVEN (SET BIT #5 IN RHCS2)
4527 ;* RHCS2 IS CHECKED TO HAVE "IR" (BIT #6) HIGH,
4528 ;* TOGETHER WITH UNIT NUMBER
4529 ;* ONE WORD = 52525 IS WRITTEN INTO RHDB
4530 ;* RHCS2 IS CHECKED TO HAVE "IR" AND UNIT NUMBER
4531 ;* A SECOND WORD = 12525 IS WRITTEN INTO RHDB
4532 ;* "OR" (BIT #7 IN RHCS2) IS WAITED FOR BY A TRAP
4533 ;* INSTRUCTION CALLED "WAT"
4534 ;* RHDB IS READ AND CHECKED TO CONTAIN 52525
4535 ;* RHCS2 IS CHECKED TO HAVE "IR" "OR" AND UNIT NUMBER
4536 ;* RHDB IS READ A SECOND TIME AND CHECKED TO
4537 ;* CONTAIN 125252
4538 ;* RHCS2 IS CHECKED TO HAVE "IR" AND UNIT NUMBER
4539 ;* THEN ALL REGISTERS RHCS1, RHCS3, RHBA, RHBAE, RHWC
4540 ;* ARE CHECKED TO HAVE APPROPRIATE VALUE
4541 ;*
4542 ;*****
4543 014430 000004          †ST13: SCOPE
4544 014432 012706 001000  MOV      #STACK.SP ;RESET STACK
4545 014436 012737 000013 004656  MOV      #13,@†STNM ;SAVE TEST NUMBER
4546

```

MO7

MAINDEC-11-DERHAC-A  
DERHAC.P11 T13

MACY11 27(732) 09-SEP-76 07:55 PAGE 90  
SILO TEST 3 (TWO WORD WRITE)

```

4547 014444 004737 040322 JSR PC, @CLDISK ;GIVE RH INITIALIZE
4548 ;SETUP UNIT NUBER
4549 ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
4550
4551 ;CHECK THAT RHCS2 HAS IR
4552 014450 012737 000100 001124 MOV #IR, $GDDAT ;GET GOOD = 100
4553 014456 053737 005010 001124 BIS UNIT, $GDDAT ;INCLUDE UNIT NUMBER
4554
4555 014464 017737 167400 001126 MOV @RHCS2, $BDDAT ;READ RHCS2 FOR COMPARISON
4556 014472 023737 001124 001126 CMP $GDDAT, $BDDAT ;COMPARE EXPECTED
4557 ;DATA WITH DATA READ FROM
4558 ;RHCS2
4559 014500 001401 BEQ 65$ ;BRANCH IF GOOD
4560 014502 104006 ERROR 6
4561 ;AFTER SETTING "CLR" BIT #5
4562 ;IN RHCS2 TO INIT THE RH
4563 ;AND HAVING DONE NOTHING ELSE
4564
4565 ;RHCS2 SHOULD HAVE IR
4566 ;=100
4567 ;TOGETHER WITH UNIT NUMBER
4568 ;BUT CONTAINED WHAT IS
4569 ;GIVEN IN BAD RHCS2
4570 014504 65$:
4571
4572 ;WRITE ONE WORD = 52525 INTO RHDB
4573 014504 012777 052525 167370 MOV #52525, @RHDB ;WRITE IN RHDB
4574
4575 ;WAIT FOR OR BIT IN RHCS2 REGISTER TO SET
4576 014512 104411 WAT ;TRAP TO WAIT.T SUBROUTINE
4577 014514 004070 RHCS2 ;AND WAIT FOR OR BIT IN
4578 014516 000200 OR ;RHCS2 REGISTER
4579 ;IF ERROR OCCURS HERE
4580 ;IT MEANS "OR" DID NOT
4581 ;SET FOR THE FULL COUNT
4582 ;DOWN OF THE WAIT.T SUBROUTINE
4583 ;THIS TIME IS APPROXIMATELY
4584 ;LARGER THAN 500 MILLISECONDS
4585
4586
4587 ;CHECK THAT RHCS2 HAS IR!OR
4588 014520 012737 000300 001124 MOV #IR!OR, $GDDAT ;GET GOOD = 300
4589 014526 053737 005010 001124 BIS UNIT, $GDDAT ;INCLUDE UNIT NUMBER
4590
4591 014534 017737 167330 001126 MOV @RHCS2, $BDDAT ;READ RHCS2 FOR COMPARISON
4592 014542 023737 001124 001126 CMP $GDDAT, $BDDAT ;COMPARE EXPECTED
4593 ;DATA WITH DATA READ FROM
4594 ;RHCS2
4595 014550 001401 BEQ 67$ ;BRANCH IF GOOD
4596 014552 104016 ERROR 16
4597 ;AFTER SETTING "CLR" BIT #5
4598 ;IN RHCS2 TO INIT THE RH
4599 ;THEN WRITING 52525 INTO
4600 ;RHDB
4601 ;RHCS2 SHOULD HAVE IR!OR
4602 ;=300

```

```

4603                                     ; TOGETHER WITH UNIT NUMBER
4604                                     ; BUT CONTAINED WHAT IS
4605                                     ; GIVEN IN BAD RHCS2
4606 014554                               67$:
4607
4608                                     ; WRITE A SECOND WORD = 125252 INTO RHDB
4609 014554 012777 125252 167320        MOV #125252, @RHDB ; WRITE 125252 INTO RHDB
4610
4611                                     ; WAIT FOR OR BIT IN RHCS2 REGISTER TO SET
4612 014562 104411                        WAT ; TRAP TO WAIT.T SUBROUTINE
4613 014564 004070                        RHCS2 ; AND WAIT FOR OR BIT IN
4614 014566 000200                        OR ; RHCS2 REGISTER
4615                                     ; IF ERROR OCCURS HERE
4616                                     ; IT MEANS "OR" DID NOT
4617                                     ; SET FOR THE FULL COUNT
4618                                     ; DOWN OF THE WAIT.T SUBROUTINE
4619                                     ; THIS TIME IS APPROXIMATELY
4620                                     ; LARGER THAN 500 MILLISECONDS
4621
4622                                     ; CHECK THAT RHDB HAS 52525
4623 014570 012737 052525 001124        MOV #52525, $GDDAT ; GET GOOD = 0
4624
4625 014576 017737 167300 001126        MOV @RHDB, $BDDAT ; READ RHDB FOR COMPARISON
4626 014604 023737 001124 001126        CMP $GDDAT, $BDDAT ; COMPARE EXPECTED
4627                                     ; DATA WITH DATA READ FROM
4628                                     ; RHDB
4629 014612 001401                        BEQ 69$ ; BRANCH IF GOOD
4630 014614 104017                        ERROR 17
4631                                     ; AFTER CLEARING THE RH THEN
4632                                     ; WRITING TWO WORDS INTO
4633                                     ; RHDB 52525 ND 125252
4634                                     ; THEN READING RHDB ONCE
4635                                     ; RHDB SHOULD HAVE 52525
4636                                     ; BUT CONTAINED WHAT IS
4637                                     ; GIVEN IN BAD RHDB
4638 014616                               69$:
4639
4640                                     ; CHECK THAT RHCS2 HAS IR!OR
4641 014616 012737 000300 001124        MOV #IR!OR, $GDDAT ; GET GOOD = 300
4642 014624 053737 005010 001124        BIS UNIT, $GDDAT ; INCLUDE UNIT NUMBER
4643
4644 014632 017737 167232 001126        MOV @RHCS2, $BDDAT ; READ RHCS2 FOR COMPARISON
4645 014640 023737 001124 001126        CMP $GDDAT, $BDDAT ; COMPARE EXPECTED
4646                                     ; DATA WITH DATA READ FROM
4647                                     ; RHCS2
4648 014646 001401                        BEQ 71$ ; BRANCH IF GOOD
4649 014650 104020                        ERROR 20
4650                                     ; AFTER CLEARING THE RH THEN
4651                                     ; WRITING TWO WORDS INTO
4652                                     ; RHDB 52525 ND 125252
4653                                     ; THEN READING RHDB ONCE
4654                                     ; RHCS2 SHOULD HAVE IR!OR
4655                                     ; =300
4656                                     ; TOGETHER WITH UNIT NUMBER
4657                                     ; BUT CONTAINED WHAT IS
4658 014652                               71$:

```

```

4659
4660
4661 ;READ RHDB A SECOND TIME
4662 ;CHECK THAT RHDB HAS 125252
4663 014652 012737 125252 001124 MOV #125252,$GDDAT ;GET GOOD = 0
4664
4665 014660 017737 167216 001126 MOV @RHDB,$BDDAT ;READ RHDB FOR COMPARISON
4666 014666 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
4667 ;DATA WITH DATA READ FROM
4668 ;RHDB
4669 014674 001401 BEQ 73$ ;BRANCH IF GOOD
4670 014676 104021 ERROR 21
4671 ;AFTER CLEARING RH THEN
4672 ;WRITING TWO WORDS INTO
4673 ;RHDB 52525 AND 125252
4674 ;THEN READING RHDB THE
4675 ;SECOND TIME
4676 ;RHDB SHOULD HAVE 125252
4677 ;BUT CONTAINED WHAT IS
4678 ;GIVEN IN BAD RHDB
4679 014700 73$:
4680 ;CHECK THAT RHCS2 HAS IR
4681 014700 012737 000100 001124 MOV #IR,$GDDAT ;GET GOOD = 100
4682 014706 053737 005010 001124 BIS UNIT,$GDDAT ;INCLUDE UNIT NUMBER
4683
4684 014714 017737 167150 001126 MOV @RHCS2,$BDDAT ;READ RHCS2 FOR COMPARISON
4685 014722 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
4686 ;DATA WITH DATA READ FROM
4687 ;RHCS2
4688 014730 001401 BEQ 75$ ;BRANCH IF GOOD
4689 014732 104022 ERROR 22
4690 ;AFTER CLEARING RH THEN
4691 ;WRITING TWO WORDS INTO
4692 ;RHDB 52525 AND 125252
4693 ;THEN READING RHDB THE
4694 ;SECOND TIME
4695 ;RHCS2 SHOULD HAVE IR
4696 ;=100
4697 ;TOGETHER WITH UNIT NUMBER
4698 ;BUT CONTAINED WHAT IS
4699 ;GIVEN IN BAD RHCS2
4700 014734 75$:
4701 ;CHECK THAT RHCS1 HAS DVA!RDY
4702 014734 012737 004200 001124 MOV #DVA!RDY,$GDDAT ;GET GOOD = 4200
4703
4704 014742 017737 167112 001126 MOV @RHCS1,$BDDAT ;READ RHCS1 FOR COMPARISON
4705 014750 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
4706 ;DATA WITH DATA READ FROM
4707 ;RHCS1
4708 014756 001401 BEQ 77$ ;BRANCH IF GOOD
4709 014760 104141 ERROR 141
4710 ;AFTER CLEARING RH THEN
4711 ;WRITING TWO WORDS INTO
4712 ;RHDB 52525 AND 125252
4713 ;THEN READING RHDB THE
4714 ;SECOND TIME

```



```

4771                                     ;BUT CONTAINED WHAT IS
4772                                     ;GIVEN IN BAD RHBAE
4773 015064                               83$:
4774                                     ;CHECK THAT RHWC HAS 0
4775 015064 012737 000000 001124         MOV    #0,$GDDAT    ;GET GOOD = 0
4776
4777 015072 017737 166764 001126         MOV    @RHWC,$BDDAT ;READ RHWC FOR COMPARISON
4778 015100 023737 001124 001126         CMP    $GDDAT,$BDDAT ;COMPARE EXPECTED
4779                                     ;DATA WITH DATA READ FROM
4780                                     ;RHWC
4781 015106 001401                         BEQ    85$          ;BRANCH IF GOOD
4782 015110 104145
4783
4784                                     ;AFTER CLEARING RH THEN
4785                                     ;WRITING TWO WORDS INTO
4786                                     ;RHDB 52525 AND 125252
4787                                     ;THEN READING RHDB THE
4788                                     ;SECOND TIME
4789                                     ;RHWC SHOULD HAVE 0
4790                                     ;BUT CONTAINED WHAT IS
4791 015112                               85$:
4792                                     ;GIVEN IN BAD RHWC
4793
4794                                     ;*****
4795                                     ;*TEST 14      SILO TEST 4 (COUNT PATTERN)
4796
4797                                     ;*
4798                                     ;*   AFTER A RH CLEAR IS GIVEN (SET BIT #5 IN RHCS2)
4799                                     ;*   RHCS2 IS CHECKED TO HAVE "IR" (BIT #6) HIGH, TOGETHER
4800                                     ;*   WITH UNIT NUMBER
4801                                     ;*   EIGHT WORDS, A COUNT PATTERN 0 THRU 7 IS WRITTEN
4802                                     ;*   INTO RHDB
4803                                     ;*   "OR" (BIT #7 IN RHCS2) IS WAITED FOR BY A TRAP
4804                                     ;*   INSTRUCTION CALLED "WAT"
4805                                     ;*   THEN RHCS2 IS CHECKED TO HAVE "IR" LOW AND
4806                                     ;*   "OR" HIGH TOGETHER WITH THE UNIT NUMBER
4807                                     ;*   THEN RHBS READ AND COMPARED TO HAVE THE RIGHT VALUE
4808                                     ;*   EIGHT TIES
4809                                     ;*   THEN RHCS2, RHCS1, RHCS3, RHBA, RHBAE. RHWC ARE
4810                                     ;*   CHECKED TO HAVE THE APPROPRIATE VALUE
4811                                     ;*****
4812 015112 000004       †ST14: SCOPE
4813 015114 012706 001000
4814 015120 012737 000014 004656         MOV    #STACK,SP    ;RESET STACK
4815                                     MOV    #14,@†STNM    ;SAVE TEST NUMBER
4816
4817 015126 004737 040322                 JSR    PC,@#CLDISK  ;GIVE RH INITIALIZE
4818                                     ;SETUP UNIT NUBER
4819                                     ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
4820
4821                                     ;CHECK THAT RHCS2 HAS IR
4822 015132 012737 000100 001124         MOV    #IR,$GDDAT   ;GET GOOD = 100
4823 015140 053737 005010 001124         BIS    UNIT,$GDDAT  ;INCLUDE UNIT NUMBER
4824
4825 015146 017737 166716 001126         MOV    @RHCS2,$BDDAT ;READ RHCS2 FOR COMPARISON
4826 015154 023737 001124 001126         CMP    $GDDAT,$BDDAT ;COMPARE EXPECTED
                                     ;DATA WITH DATA READ FROM
                                     ;RHCS2

```



MAINDEC-11-DERHAC-A  
DERHAC.P11 T14

MACY11 27(732) 09-SEP-76 07:55 PAGE 95  
SILO TEST 4 (COUNT PATTERN)

```

4827 015162 001401          BEQ      65$      ;BRANCH IF GOOD
4828 015164 104006          ERROR    6
4829                                     ;AFTER SETTING "CLR" BIT #5
4830                                     ;IN RHCS2 TO INIT THE RH
4831                                     ;AND HAVING DONE NOTHING ELSE
4832                                     ;RHCS2 SHOULD HAVE IR
4833                                     ;=100
4834                                     ;TOGETHER WITH UNIT NUMBER
4835                                     ;BUT CONTAINED WHAT IS
4836                                     ;GIVEN IN BAD RHCS2
4837 015166                  65$:
4838
4839                                     ;WRITE EIGHT WORDS INTO RHDB TO FILL IT
4840                                     ;THIS IS A COUNT PATTERN 0 THRU 7
4841 015166 012702 000010    MOV      #8.,R2      ;8 NUMBERS TO BE WRITTEN
4842 015172 005001          CLR      R1          ;FIRST WORD TO BE WRITTEN = 0
4843 015174 010177 166702  1$:    MOV      R1,2RHDB    ;WRITE INTO RHDB - A COUNT
4844 015200 005201          INC      R1          ;GET NEXT WORD
4845 015202 005302          DEC      R2          ;COUNT TO 8
4846 015204 001373          BNE     1$          ;BRANCH IF 8 NOT DONE
4847
4848                                     ;WAIT FOR OR BIT IN RHCS2 REGISTER TO SET
4849 015206 104411          WAT      RHCS2      ;TRAP TO WAIT.T SUBROUTINE
4850 015210 004070          OR       OR          ;AND WAIT FOR OR BIT IN
4851 015212 000200          OR       RHCS2      ;RHCS2 REGISTER
4852                                     ;IF ERROR OCCURS HERE
4853                                     ;IT MEANS "OR" DID NOT
4854                                     ;SET FOR THE FULL COUNT
4855                                     ;DOWN OF THE WAIT.T SUBROUTINE
4856                                     ;THIS TIME IS APPROXIMATELY
4857                                     ;LARGER THAN 500 MILLISECONDS
4858
4859                                     ;CHECK THAT RHCS2 HAS OR
4860 015214 012737 000200 001124  MOV      #OR,$GDDAT ;GET GOOD = 200
4861 015222 053737 005010 001124  BIS      UNIT,$GDDAT ;INCLUDE UNIT NUMBER
4862
4863 015230 017737 166634 001126  MOV      2RHCS2,$BDDAT ;READ RHCS2 FOR COMPARISON
4864 015236 023737 001124 001126  CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED
4865                                     ;DATA WITH DATA READ FROM
4866                                     ;RHCS2
4867 015244 001401          BEQ      67$      ;BRANCH IF GOOD
4868 015246 104023          ERROR    23
4869                                     ;AFTER SETTING "CLR" BIT #5
4870                                     ;IN RHCS2 TO INIT THE RH
4871                                     ;AND HAVING WRITTEN A
4872                                     ;COUNT PATTERN 0 THRU 7
4873                                     ;INTO RHDB TO FILL IT
4874                                     ;RHCS2 SHOULD HAVE OR
4875                                     ;=200
4876                                     ;TOGETHER WITH UNIT NUMBER
4877                                     ;BUT CONTAINED WHAT IS
4878                                     ;GIVEN IN BAD RHCS2
4879 015250                  67$:
4880
4881
4882

```

```

4883 015250 012737 000001 004664      MOV      #1,SILONM      ;FIRST WORD TO BE READ
4884      ;CHECK THAT RHDB HAS 0
4885 015256 012737 000000 001124      MOV      #0,$GDDAT      ;GET GOOD = 0
4886
4887 015264 017737 166612 001126      MOV      @RHDB,$BDDAT    ;READ RHDB FOR COMPARISON
4888 015272 023737 001124 001126      CMP      $GDDAT,$BDDAT  ;COMPARE EXPECTED
4889      ;DATA WITH DATA READ FROM
4890      ;RHDB
4891 015300 001401      BEQ      69$            ;BRANCH IF GOOD
4892 015302 104024      ERROR   24
4893      ;AFTER SETTING "CLR" BIT #5
4894      ;IN RHCS2 TO INIT THE RH
4895      ;AND WRITING A
4896      ;COUNT PATTERN 0 THRU 7
4897      ;INTO RHDB TO FILL IT
4898      ;THEN ON READING RHDB THE
4899      ;FIRST TIME
4900      ;RHDB SHOULD HAVE 0
4901      ;BUT CONTAINED WHAT IS
4902      ;GIVEN IN BAD RHDB
4903 015304      69$:
4904
4905 015304 012737 000002 004664      MOV      #2,SILONM      ;SECOND WORD TO BE READ
4906      ;CHECK THAT RHDB HAS 1
4907 015312 012737 000001 001124      MOV      #1,$GDDAT      ;GET GOOD = 0
4908
4909 015320 017737 166556 001126      MOV      @RHDB,$BDDAT    ;READ RHDB FOR COMPARISON
4910 015326 023737 001124 001126      CMP      $GDDAT,$BDDAT  ;COMPARE EXPECTED
4911      ;DATA WITH DATA READ FROM
4912      ;RHDB
4913 015334 001401      BEQ      71$            ;BRANCH IF GOOD
4914 015336 104024      ERROR   24
4915      ;AFTER SETTING "CLR" BIT #5
4916      ;IN RHCS2 TO INIT THE RH
4917      ;AND WRITING A
4918      ;COUNT PATTERN 0 THRU 7
4919      ;INTO RHDB TO FILL IT
4920      ;THEN ON READING RHDB THE
4921      ;SECOND TIME
4922      ;RHDB SHOULD HAVE 1
4923      ;BUT CONTAINED WHAT IS
4924      ;GIVEN IN BAD RHDB
4925 015340      71$:
4926
4927 015340 012737 000003 004664      MOV      #3,SILONM      ;THIRD WORD TO BE READ
4928      ;CHECK THAT RHDB HAS 2
4929 015346 012737 000002 001124      MOV      #2,$GDDAT      ;GET GOOD = 0
4930
4931 015354 017737 166522 001126      MOV      @RHDB,$BDDAT    ;READ RHDB FOR COMPARISON
4932 015362 023737 001124 001126      CMP      $GDDAT,$BDDAT  ;COMPARE EXPECTED
4933      ;DATA WITH DATA READ FROM
4934      ;RHDB
4935 015370 001401      BEQ      73$            ;BRANCH IF GOOD
4936 015372 104024      ERROR   24
4937      ;AFTER SETTING "CLR" BIT #5
4938      ;IN RHCS2 TO INIT THE RH

```





; INTO RHDB TO FILL IT  
; THEN ON READING RHDB THE  
; EIGHTH TIME  
; RHDB SHOULD HAVE 7  
; BUT CONTAINED WHAT IS  
; GIVEN IN BAD RHDB

5051  
5052  
5053  
5054  
5055  
5056  
5057 015610 83\$:  
5058  
5059

5060 ; CHECK THAT RHCS2 HAS IR  
5061 015610 012737 000100 001124 MOV #IR,\$GDDAT  
5062 015616 053737 005010 001124 BIS UNIT,\$GDDAT  
5063  
5064 015624 017737 166240 001126 MOV 2RHCS2,\$BDDAT  
5065 015632 023737 001124 001126 CMP \$GDDAT,\$BDDAT  
5066  
5067  
5068 015640 001401 BEQ 85\$  
5069 015642 104025 ERROR 25

; READ RHCS2 FOR COMPARISON  
; COMPARE EXPECTED  
; DATA WITH DATA READ FROM  
; RHCS2  
; BRANCH IF GOOD

; AFTER SETTING "CLR" BIT #5  
; IN RHCS2 TO INIT THE RH  
; AND WRITING A  
; COUNT PATTERN 0 THRU 7  
; INTO RHDB TO FILL IT  
; THEN ON READING RHDB THE  
; TOTAL 8 TIMES  
; RHCS2 SHOULD HAVE IR  
; =100  
; TOGETHER WITH UNIT NUMBER  
; BUT CONTAINED WHAT IS  
; GIVEN IN BAD RHCS2

5080  
5081  
5082 015644 85\$:  
5083  
5084 015644 012737 004200 001124  
5085  
5086 015652 017737 166202 001126  
5087 015660 023737 001124 001126  
5088  
5089  
5090 015666 001401 BEQ 87\$  
5091 015670 104026 ERROR 26

; CHECK THAT RHCS1 HAS DVA!RDY  
MOV #DVA!RDY,\$GDDAT ; GET GOOD = 4200  
MOV 2RHCS1,\$BDDAT ; READ RHCS1 FOR COMPARISON  
CMP \$GDDAT,\$BDDAT ; COMPARE EXPECTED  
; DATA WITH DATA READ FROM  
; RHCS1  
; BRANCH IF GOOD

; AFTER SETTING "CLR" BIT #5  
; IN RHCS2 TO INIT THE RH  
; AND WRITING A  
; COUNT PATTERN 0 THRU 7  
; INTO RHDB TO FILL IT  
; THEN ON READING RHDB THE  
; TOTAL 8 TIMES  
; RHCS1 SHOULD HAVE DVA!RDY  
; =4200  
; BUT CONTAINED WHAT IS  
; GIVEN IN BAD RHCS1

5100  
5101  
5102  
5103 015672 87\$:  
5104  
5105 015672 012737 000000 001124  
5106

; CHECK THAT RHCS3 HAS 0  
MOV #0,\$GDDAT ; GET GOOD = 0



```

5163 015774          93$:
5164          ;CHECK THAT RHWC HAS 0
5165 015774 012737 000000 001124  MOV    #0,$GDDAT    ;GET GOOD = 0
5166
5167 016002 017737 166054 001126  MOV    @RHWC,$BDDAT ;READ RHWC FOR COMPARISON
5168 016010 023737 001124 001126  CMP    $GDDAT,$BDDAT ;COMPARE EXPECTED
5169          ;DATA WITH DATA READ FROM
5170          ;RHWC
5171 016016 001401          BEQ    95$          ;BRANCH IF GOOD
5172 016020 104032          ERROR  32
5173
5174          ;AFTER SETTING "CLR" BIT #5
5175          ;IN RHCS2 TO INIT THE RH
5176          ;AND WRITING A
5177          ;COUNT PATTERN 0 THRU 7
5178          ;INTO RHDB TO FILL IT
5179          ;THEN ON READING RHDB THE
5180          ;TOTAL 8 TIMES
5181          ;RHWC SHOULD HAVE 0
5182          ;BUT CONTAINED WHAT IS
5183          ;GIVEN IN BAD RHWC
5183 016022          95$:
5184
5185          ;*****
5186          ;*TEST 15      SILO TEST 5 (FLOATING ONES)
5187
5188          ;*
5189          ;* AFTER ARH CLEAR IS GIVEN (SET BIT #5 IN RHCS2)
5190          ;* RHCS2 IS CHECKED TO HAVE "IR" (BIT #6) HIGH.
5191          ;* TOGETHER WITH UNIT NUMBER
5192          ;* EIGHT WORDS, A PATTERN OF FLOATING ONES (1,2,4,10,20,40,100,200)
5193          ;* IS WRITTEN INTO RHDB
5194          ;* "OR" (BIT #7 IN RHCS2) IS WAITED FOR BY A TRAP
5195          ;* INSTRUCTION CALLED "WAT"
5196          ;* THEN RHCS2 IS CHECKED TO HAVE "IR" LOW AND
5197          ;* "OR" HIGH TOGETHER WITH THE UNIT NUMBER
5198          ;* THEN RHDB IS READ AND COMPARED TO HAVE THE
5199          ;* RIGHT VALUE AFTER EACH OF THE EIGHT READS.
5200          ;* THEN RHCS2 IS CHECKED TO HAVE "IR"
5201          ;* AND UNIT NUMBER
5202          ;* THEN RHCS1, RHCS3, RHBA, RHBAE, RHWC ARE
5203          ;* CHECKED TO HAVE THE APPROPRIATE VALUE
5204          ;*****
5205 016022 000004          †ST15: SCOPE
5206 016024 012706 001000  MOV    #STACK,SP    ;RESET STACK
5207 016030 012737 000015 004656  MOV    #15,@†STNM   ;SAVE TEST NUMBER
5208
5209 016036 004737 040322  JSR    PC,@*CLDISK ;GIVE RH INITIALIZE
5210          ;SETUP UNIT NUBER
5211          ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
5212
5213
5214          ;CHECK THAT RHCS2 HAS IR
5215 016042 012737 000100 001124  MOV    #IR,$GDDAT   ;GET GOOD = 100
5216 016050 053737 005010 001124  BIS    UNIT,$GDDAT  ;INCLUDE UNIT NUMBER
5217
5218 016056 017737 166006 001126  MOV    @RHCS2,$BDDAT ;READ RHCS2 FOR COMPARISON

```





```

5275                                     ;GIVEN IN BAD RHCS2
5276 016162                               67$:
5277
5278
5279
5280 016162 012737 000001 004664          MOV    #1,SILONM          ;FIRST WORD TO BE READ
5281                                     ;CHECK THAT RHDB HAS 1
5282 016170 012737 000001 001124          MOV    #1,$GDDAT         ;GET GOOD = 0
5283
5284 016176 017737 165700 001126          MOV    @RHDB,$BDDAT      ;READ RHDB FOR COMPARISON
5285 016204 023737 001124 001126          CMP    $GDDAT,$BDDAT     ;COMPARE EXPECTED
5286                                     ;DATA WITH DATA READ FROM
5287                                     ;RHDB
5288
5289 016212 001401                          BEQ    69$                ;BRANCH IF GOOD
5290 016214 104024                          ERROR  24
5291                                     ;AFTER SETTING "CLR" BIT #5
5292                                     ;IN RHCS2 TO INIT THE RH
5293                                     ;AND WRITING A PATTERN OF
5294                                     ;FLOATING ONES (WHICH IS
5295                                     ;1,2,4,10,20,40,100,200)
5296                                     ;INTO RHDB TO FILL IT
5297                                     ;THEN ON READING RHDB THE
5298                                     ;FIRST TIME
5299                                     ;RHDB SHOULD HAVE 1
5300                                     ;BUT CONTAINED WHAT IS
5301 016216                               69$:
5302
5303
5304 016216 012737 000002 004664          MOV    #2,SILONM          ;SECOND WORD TO BE READ
5305                                     ;CHECK THAT RHDB HAS 2
5306 016224 012737 000002 001124          MOV    #2,$GDDAT         ;GET GOOD = 0
5307
5308 016232 017737 165644 001126          MOV    @RHDB,$BDDAT      ;READ RHDB FOR COMPARISON
5309 016240 023737 001124 001126          CMP    $GDDAT,$BDDAT     ;COMPARE EXPECTED
5310                                     ;DATA WITH DATA READ FROM
5311                                     ;RHDB
5312 016246 001401                          BEQ    71$                ;BRANCH IF GOOD
5313 016250 104024                          ERROR  24
5314                                     ;AFTER SETTING "CLR" BIT #5
5315                                     ;IN RHCS2 TO INIT THE RH
5316                                     ;AND WRITING A PATTERN OF
5317                                     ;FLOATING ONES (WHICH IS
5318                                     ;1,2,4,10,20,40,100,200)
5319                                     ;INTO RHDB TO FILL IT
5320                                     ;THEN ON READING RHDB THE
5321                                     ;SECOND TIME
5322                                     ;RHDB SHOULD HAVE 2
5323                                     ;BUT CONTAINED WHAT IS
5324                                     ;GIVEN IN BAD RHDB
5325 016252                               71$:
5326
5327
5328 016252 012737 000003 004664          MOV    #3,SILONM          ;THIRD WORD TO BE READ
5329                                     ;CHECK THAT RHDB HAS 4
5330 016260 012737 000004 001124          MOV    #4,$GDDAT         ;GET GOOD = 0

```



```

5387
5388
5389
5390
5391
5392
5393
5394
5395
5396
5397 016376          77$:
5398
5399
5400 016376 012737 000006 004664      MOV    #6,SILONM      ;SIXTH WORD TO BE READ
5401      ;CHECK THAT RHDB HAS 40
5402 016404 012737 000040 001124      MOV    #40,$GDDAT    ;GET GOOD = 0
5403
5404 016412 017737 165464 001126      MOV    @RHDB,$BDDAT  ;READ RHDB FOR COMPARISON
5405 016420 023737 001124 001126      CMP    $GDDAT,$BDDAT ;COMPARE EXPECTED
5406      ;DATA WITH DATA READ FROM
5407      ;RHDB
5408 016426 001401      BEQ    79$           ;BRANCH IF GOOD
5409 016430 104024
5410
5411      ;AFTER SETTING "CLR" BIT #5
5412      ;IN RHCS2 TO INIT THE RH
5413      ;AND WRITING A PATTERN OF
5414      ;FLOATING ONES (WHICH IS
5415      ;1 2 4 10 20 40 100 200)
5416      ;INTO RHDB TO FILL IT
5417      ;THEN ON READING RHDB THE
5418      ;SIXTH TIME
5419      ;RHDB SHOULD HAVE 40
5420      ;BUT CONTAINED WHAT IS
5421      ;GIVEN IN BAD RHDB
5421 016432          79$:
5422
5423
5424 016432 012737 000007 004664      MOV    #7,SILONM      ;SEVENTH WORD TO BE READ
5425      ;CHECK THAT RHDB HAS 100
5426 016440 012737 000100 001124      MOV    #100,$GDDAT   ;GET GOOD = 0
5427
5428 016446 017737 165430 001126      MOV    @RHDB,$BDDAT  ;READ RHDB FOR COMPARISON
5429 016454 023737 001124 001126      CMP    $GDDAT,$BDDAT ;COMPARE EXPECTED
5430      ;DATA WITH DATA READ FROM
5431      ;RHDB
5432 016462 001401      BEQ    81$           ;BRANCH IF GOOD
5433 016464 104024
5434
5435      ;AFTER SETTING "CLR" BIT #5
5436      ;IN RHCS2 TO INIT THE RH
5437      ;AND WRITING A PATTERN OF
5438      ;FLOATING ONES (WHICH IS
5439      ;1 2 4 10 20 40 100 200)
5440      ;INTO RHDB TO FILL IT
5441      ;THEN ON READING RHDB THE
5442      ;SEVENTH TIME
5442      ;RHDB SHOULD HAVE 100

```

```

5443                                     ; BUT CONTAINED WHAT IS
5444                                     ; GIVEN IN BAD RHDB
5445 016466                               81$:
5446
5447
5448 016466 012737 000010 004664         MOV    #8, SILONM      ; EIGHTH WORD TO BE READ
5449                                     ; CHECK THAT RHDB HAS 200
5450 016474 012737 000200 001124         MOV    #200, $GDDAT   ; GET GOOD = 0
5451
5452 016502 017737 165374 001126         MOV    @RHDB, $BDDAT  ; READ RHDB FOR COMPARISON
5453 016510 023737 001124 001126         CMP    $GDDAT, $BDDAT ; COMPARE EXPECTED
5454                                     ; DATA WITH DATA READ FROM
5455                                     ; RHDB
5456 016516 001401                         BEQ    83$            ; BRANCH IF GOOD
5457 016520 104024                         ERROR  24
5458                                     ; AFTER SETTING "CLR" BIT #5
5459                                     ; IN RHCS2 TO INIT THE RH
5460                                     ; AND WRITING A PATTERN OF
5461                                     ; FLOATING ONES (WHICH IS
5462                                     ; 1, 2, 4, 10, 20, 40, 100, 200)
5463                                     ; INTO RHDB TO FILL IT
5464                                     ; THEN ON READING RHDB THE
5465                                     ; EIGHTH TIME
5466                                     ; RHDB SHOULD HAVE 200
5467                                     ; BUT CONTAINED WHAT IS
5468                                     ; GIVEN IN BAD RHDB
5469 016522                               83$:
5470
5471
5472                                     ; CHECK THAT RHCS2 HAS IR
5473 016522 012737 000100 001124         MOV    #IR, $GDDAT   ; GET GOOD = 100
5474 016530 053737 005010 001124         BIS    UNIT, $GDDAT  ; INCLUDE UNIT NUMBER
5475
5476 016536 017737 165326 001126         MOV    @RHCS2, $BDDAT ; READ RHCS2 FOR COMPARISON
5477 016544 023737 001124 001126         CMP    $GDDAT, $BDDAT ; COMPARE EXPECTED
5478                                     ; DATA WITH DATA READ FROM
5479                                     ; RHCS2
5480 016552 001401                         BEQ    85$            ; BRANCH IF GOOD
5481 016554 104025                         ERROR  25
5482                                     ; AFTER SETTING "CLR" BIT #5
5483                                     ; IN RHCS2 TO INIT THE RH
5484                                     ; AND WRITING A PATTERN OF
5485                                     ; FLOATING ONES (WHICH IS
5486                                     ; 1, 2, 4, 10, 20, 40, 100, 200)
5487                                     ; INTO RHDB TO FILL IT
5488                                     ; THEN ON READING RHDB THE
5489                                     ; TOTAL 8 TIMES
5490                                     ; RHCS2 SHOULD HAVE IR
5491                                     ; =100
5492                                     ; TOGETHER WITH UNIT NUMBER
5493                                     ; BUT CONTAINED WHAT IS
5494                                     ; GIVEN IN BAD RHCS2
5495 016556                               85$:
5496
5497 016556 012737 004200 001124         MOV    #DVA!RDY, $GDDAT ; GET GOOD = 4200
5498

```



```

5555 ;TOTAL 8 TIMES
5556 ;RHBA SHOULD HAVE 0
5557 ;BUT CONTAINED WHAT IS
5558 ;GIVEN IN BAD RHBA
5559 016660          91$:
5560 ;CHECK THAT RHBAE HAS 0
5561 016660 012737 000000 001124  MOV    #0,$GDDAT ;GET GOOD = 0
5562
5563 016666 017737 165236 001126  MOV    @RHBAE,$BDDAT ;READ RHBAE FOR COMPARISON
5564 016674 023737 001124 001126  CMP    $GDDAT,$BDDAT ;COMPARE EXPECTED
5565 ;DATA WITH DATA READ FROM
5566 ;RHBAE
5567 016702 001401          BEQ    93$ ;BRANCH IF GOOD
5568 016704 104031          ERROR  31
5569
5570 ;AFTER SETTING "CLR" BIT #5
5571 ;IN RHCS2 TO INIT THE RH
5572 ;AND WRITING A PATTERN OF
5573 ;FLOATING ONES (WHICH IS
5574 ;1 2 4 10 20 40 100 200)
5575 ;INTO RHDB TO FILL IT
5576 ;THEN ON READING RHDB THE
5577 ;TOTAL 8 TIMES
5578 ;RHBAE SHOULD HAVE 0
5579 ;BUT CONTAINED WHAT IS
5580 ;GIVEN IN BAD RHBAE
5580 016706          93$:
5581 ;CHECK THAT RHWC HAS 0
5582 016706 012737 000000 001124  MOV    #0,$GDDAT ;GET GOOD = 0
5583
5584 016714 017737 165142 001126  MOV    @RHWC,$BDDAT ;READ RHWC FOR COMPARISON
5585 016722 023737 001124 001126  CMP    $GDDAT,$BDDAT ;COMPARE EXPECTED
5586 ;DATA WITH DATA READ FROM
5587 ;RHWC
5588 016730 001401          BEQ    95$ ;BRANCH IF GOOD
5589 016732 104032          ERROR  32
5590
5591 ;AFTER SETTING "CLR" BIT #5
5592 ;IN RHCS2 TO INIT THE RH
5593 ;AND WRITING A PATTERN OF
5594 ;FLOATING ONES (WHICH IS
5595 ;1 2 4 10 20 40 100 200)
5596 ;INTO RHDB TO FILL IT
5597 ;THEN ON READING RHDB THE
5598 ;TOTAL 8 TIMES
5599 ;RHWC SHOULD HAVE 0
5600 ;BUT CONTAINED WHAT IS
5601 ;GIVEN IN BAD RHWC
5601 016734          95$:
5602
5603
5604
5605 ;*****
5606 ;*TEST 16          SILO TEST 6 (FLOATING ONES IN UPPER BYTE)
5607
5608 ;*          AFTER A RH CLEAR IS GIVEN (SET BIT #5 IN RHCS2)
5609 ;*          RHCS2 IS CHECKED TO HAVE "IR" (BIT #6) HIGH.
5610 ;*          TOGETHER WITH UNIT NUMBER

```

```

5611 : * EIGHT WORDS, A PATTERN OF FLOATING ONES IN UPPER BYTE
5612 : * 400,1000,2000,4000,10000,20000,40000,100000
5613 : * IS WRITTEN INTO RHDB
5614 : * "OR" (BIT #7 IN RHCS2) IS WAITED FOR BY A TRAP
5615 : * INSTRUCTION CALLED "WAT"
5616 : * THEN RHCS2 IS CHECKED TO HAVE "IR" LOW AND
5617 : * "OR" HIGH TOGETHER WITH THE UNIT NUMBER
5618 : * THEN RHDB IS READ AND COMPARED TO HAVE THE
5619 : * RIGHT VALUE AFTER EACH OF THE EIGHT READS.
5620 : * THEN RHCS2 IS CHECKED TO HAVE "IR"
5621 : * AND UNIT NUMBER
5622 : * THEN RHCS1, RHCS3, RHBA, RHBAE, RHWC ARE
5623 : * CHECKED TO HAVE THE APPROPRIATE VALUE
5624 : * *****
5625 016734 000004          †ST16: SCOPE
5626 016736 012796 001000  MOV      #STACK,SP      ;RESET STACK
5627 016742 012737 000016 004656  MOV      #16,2#†STNM    ;SAVE TEST NUMBER
5628
5629 016750 004737 040322      JSR      PC,2#CLDISK    ;GIVE RH INITIALIZE
5630                                ;SETUP UNIT NUMBER
5631                                ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
5632
5633
5634                                ;CHECK THAT RHCS2 HAS IR
5635 016754 012737 000100 001124  MOV      #IR,$GDDAT     ;GET GOOD = 100
5636 016762 053737 005010 001124  BIS      UNIT,$GDDAT    ;INCLUDE UNIT NUMBER
5637
5638 016770 017737 165074 001126  MOV      2#RHCS2,$BDDAT ;READ RHCS2 FOR COMPARISON
5639 016776 023737 001124 001126  CMP      $GDDAT,$BDDAT  ;COMPARE EXPECTED
5640                                ;DATA WITH DATA READ FROM
5641                                ;RHCS2
5642 017004 001401          BEQ      65$            ;BRANCH IF GOOD
5643 017006 104006
5644                                ;AFTER SETTING "CLR" BIT #5
5645                                ;IN RHCS2 TO INIT THE RH
5646                                ;AND HAVING DONE NOTHING
5647                                ;ELSE
5648                                ;RHCS2 SHOULD HAVE IR
5649                                ;=100
5650                                ;TOGETHER WITH UNIT NUMBER
5651                                ;BUT CONTAINED WHAT IS
5652                                ;GIVEN IN BAD RHCS2
5653 017010          65$:
5654
5655                                ;WRITE EIGHT WORDS INTO RHDB TO FILL IT
5656                                ;DATA IS 400,1000,2000,4000,10000,20000,40000
5657                                ;100000
5658 017010 012701 000400      MOV      #400,R1        ;GETTING READY TO FLOAT ONES
5659                                ;IN UPPER BYTE
5660 017014 012702 000010      MOV      #8,R2          ;COUNTER -8 DATA WORDS
5661 017020 010177 165056      1$: MOV      R1,2#RHDB    ;WRITE INTO RHDB
5662 017024 006301          ASL      R1              ;SHIFT 1 ONE POSITION LEFT
5663 017026 005302          DEC      R2              ;COUNT TO 8
5664 017030 001373          BNE     1$              ;BRANCH IF 8 NOT DONE
5665                                ;WAIT FOR OR BIT IN RHCS2 REGISTER TO SET
5666 017032 104411          WAT      ;TRAP TO WAIT.T SUBROUTINE

```





:GIVEN IN BAD RHDB

5723  
5724 017130 69\$:

5725  
5726  
5727 017130 012737 000002 004664 MOV #2,SILONM :SECOND WORD TO BE READ  
5728 :CHECK THAT RHDB HAS 1000  
5729 017136 012737 001000 001124 MOV #1000,\$GDDAT :GET GOOD = 0

5730  
5731 017144 017737 164732 001126 MOV @RHDB,\$BDDAT :READ RHDB FOR COMPARISON  
5732 017152 023737 001124 001126 CMP \$GDDAT,\$BDDAT :COMPARE EXPECTED

5733 :DATA WITH DATA READ FROM  
5734 :RHDB  
5735 017160 001401 BEQ 71\$ :BRANCH IF GOOD  
5736 017162 104024 ERROR 24

:AFTER SETTING "CLR" BIT #5  
:IN RHCS2 TO INIT THE RH  
:AND WRITING A PATTERN OF FLOATING  
:ONES (WHICH IS 400,1000,2000,4000,  
:10000,20000,40000,100000)  
:INTO RHDB TO FILL IT  
:THEN ON READING RHDB THE  
:SECOND TIME  
:RHDB SHOULD HAVE 1000  
:BUT CONTAINED WHAT IS  
:GIVEN IN BAD RHDB

5747  
5748 017164 71\$:

5749  
5750  
5751 017164 012737 000003 004664 MOV #3,SILONM :THIRD WORD TO BE READ  
5752 :CHECK THAT RHDB HAS 2000  
5753 017172 012737 002000 001124 MOV #2000,\$GDDAT :GET GOOD = 0

5754  
5755 017200 017737 164676 001126 MOV @RHDB,\$BDDAT :READ RHDB FOR COMPARISON  
5756 017206 023737 001124 001126 CMP \$GDDAT,\$BDDAT :COMPARE EXPECTED

5757 :DATA WITH DATA READ FROM  
5758 :RHDB  
5759 017214 001401 BEQ 73\$ :BRANCH IF GOOD  
5760 017216 104024 ERROR 24

:AFTER SETTING "CLR" BIT #5  
:IN RHCS2 TO INIT THE RH  
:AND WRITING A PATTERN OF FLOATING  
:ONES (WHICH IS 400,1000,2000,4000,  
:10000,20000,40000,100000)  
:INTO RHDB TO FILL IT  
:THEN ON READING RHDB THE  
:THIRD TIME  
:RHDB SHOULD HAVE 2000  
:BUT CONTAINED WHAT IS  
:GIVEN IN BAD RHDB

5761  
5762  
5763  
5764  
5765  
5766  
5767  
5768  
5769  
5770  
5771  
5772 017220 73\$:

5773  
5774  
5775 017220 012737 000004 004664 MOV #4,SILONM :FOURTH WORD TO BE READ  
5776 :CHECK THAT RHDB HAS 4000  
5777 017226 012737 004000 001124 MOV #4000,\$GDDAT :GET GOOD = 0

5778

MAINDEC-11-DERHAC-A  
DERHAC.P11 T16

MACY11 27(732) 09-SEP-76 07:55 PAGE 112  
SILO TEST 6 (FLOATING ONES IN UPPER BYTE)

```

5779 017234 017737 164642 001126      MOV      @RHDB,$BDDAT      ;READ RHDB FOR COMPARISON
5780 017242 023737 001124 001126      CMP      $GDDAT,$BDDAT    ;COMPARE EXPECTED
5781                                     ;DATA WITH DATA READ FROM
5782                                     ;RHDB
5793 017250 001401      BEQ      75$              ;BRANCH IF GOOD
5794 017252 104024      ERROR    24
5795                                     ;AFTER SETTING "CLR" BIT #5
5796                                     ;IN RHCS2 TO INIT THE RH
5797                                     ;AND WRITING A PATTERN OF FLOATING
5798                                     ;ONES (WHICH IS 400,1000,2000,4000,
5799                                     ;10000,20000,40000,100000)
5800                                     ;INTO RHDB TO FILL IT
5801                                     ;THEN ON READING RHDB THE
5802                                     ;FOURTH TIME
5803                                     ;RHDB SHOULD HAVE 4000
5804                                     ;BUT CONTAINED WHAT IS
5805                                     ;GIVEN IN BAD RHDB
5806
5807 017254          75$:
5808
5809
5810
5811
5812
5813
5814
5815
5816
5817
5818
5819
5820 017254 012737 000005 004664      MOV      #5,SILONM        ;FIFTH WORD TO BE READ
5821                                     ;CHECK THAT RHDB HAS 10000
5822 017262 012737 010000 001124      MOV      #10000,$GDDAT    ;GET GOOD = 0
5823
5824
5825
5826
5827 017270 017737 164606 001126      MOV      @RHDB,$BDDAT    ;READ RHDB FOR COMPARISON
5828 017276 023737 001124 001126      CMP      $GDDAT,$BDDAT    ;COMPARE EXPECTED
5829                                     ;DATA WITH DATA READ FROM
5830                                     ;RHDB
5831 017304 001401      BEQ      77$              ;BRANCH IF GOOD
5832 017306 104024      ERROR    24
5833                                     ;AFTER SETTING "CLR" BIT #5
5834                                     ;IN RHCS2 TO INIT THE RH
5835                                     ;AND WRITING A PATTERN OF FLOATING
5836                                     ;ONES (WHICH IS 400,1000,2000,4000,
5837                                     ;10000,20000,40000,100000)
5838                                     ;INTO RHDB TO FILL IT
5839                                     ;THEN ON READING RHDB THE
5840                                     ;FIFTH TIME
5841                                     ;RHDB SHOULD HAVE 10000
5842                                     ;BUT CONTAINED WHAT IS
5843                                     ;GIVEN IN BAD RHDB
5844
5845
5846
5847
5848
5849
5850
5851
5852
5853
5854
5855
5856
5857
5858
5859
5860
5861
5862
5863
5864
5865
5866
5867
5868
5869
5870
5871
5872
5873
5874
5875
5876
5877
5878
5879
5880
5881
5882
5883
5884
5885
5886
5887
5888
5889
5890
5891
5892
5893
5894
5895
5896
5897
5898
5899
5900
5901
5902
5903
5904
5905
5906
5907
5908
5909
5910
5911
5912
5913
5914
5915
5916
5917
5918
5919
5920
5921
5922
5923
5924
5925
5926
5927
5928
5929
5930
5931
5932
5933
5934
5935
5936
5937
5938
5939
5940
5941
5942
5943
5944
5945
5946
5947
5948
5949
5950
5951
5952
5953
5954
5955
5956
5957
5958
5959
5960
5961
5962
5963
5964
5965
5966
5967
5968
5969
5970
5971
5972
5973
5974
5975
5976
5977
5978
5979
5980
5981
5982
5983
5984
5985
5986
5987
5988
5989
5990
5991
5992
5993
5994
5995
5996
5997
5998
5999

```

```

;AND WRITING A PATTERN OF FLOATING
;ONES (WHICH IS 400,1000,2000,4000,
;10000,20000,40000,100000)
;INTO RHDB TO FILL IT
;THEN ON READING RHDB THE
;SIXTH TIME
;RHDB SHOULD HAVE 20000
;BUT CONTAINED WHAT IS
;GIVEN IN BAD RHDB
    
```

```

5835
5836
5837
5838
5839
5840
5841
5842
5843
5844 017344          79$:
    
```

```

5845
5846 017344 012737 000007 004664      MOV    #7,SILONM      ;SEVENTH WORD TO BE READ
5847      ;CHECK THAT RHDB HAS 40000
5848 017352 012737 040000 001124      MOV    #40000,$GDDAT ;GET GOOD = 0
5849
5850
5851 017360 017737 164516 001126      MOV    @RHDB,$BDDAT  ;READ RHDB FOR COMPARISON
5852 017366 023737 001124 001126      CMP    $GDDAT,$BDDAT ;COMPARE EXPECTED
5853      ;DATA WITH DATA READ FROM
5854      ;RHDB
5855 017374 001401          BEQ    81$           ;BRANCH IF GOOD
5856 017376 104024          ERROR   24
    
```

```

;AFTER SETTING "CLR" BIT #5
;IN RHCS2 TO INIT THE RH
;AND WRITING A PATTERN OF FLOATING
;ONES (WHICH IS 400,1000,2000,4000,
;10000,20000,40000,100000)
;INTO RHDB TO FILL IT
;THEN ON READING RHDB THE
;SEVENTH TIME
;RHDB SHOULD HAVE 40000
;BUT CONTAINED WHAT IS
;GIVEN IN BAD RHDB
    
```

```

5857
5858
5859
5860
5861
5862
5863
5864
5865
5866
5867
5868 017400          81$:
    
```

```

5869
5870
5871 017400 012737 000010 004664      MOV    #8,SILONM      ;EIGHTH WORD TO BE READ
5872      ;CHECK THAT RHDB HAS 100000
5873 017406 012737 100000 001124      MOV    #100000,$GDDAT ;GET GOOD = 0
5874
5875 017414 017737 164462 001126      MOV    @RHDB,$BDDAT  ;READ RHDB FOR COMPARISON
5876 017422 023737 001124 001126      CMP    $GDDAT,$BDDAT ;COMPARE EXPECTED
5877      ;DATA WITH DATA READ FROM
5878      ;RHDB
5879 017430 001401          BEQ    83$           ;BRANCH IF GOOD
5880 017432 104024          ERROR   24
    
```

```

;AFTER SETTING "CLR" BIT #5
;IN RHCS2 TO INIT THE RH
;AND WRITING A PATTERN OF FLOATING
;ONES (WHICH IS 400,1000,2000,4000,
;10000,20000,40000,100000)
;INTO RHDB TO FILL IT
;THEN ON READING RHDB THE
;EIGHTH TIME
;RHDB SHOULD HAVE 100000
;BUT CONTAINED WHAT IS
    
```

```

5881
5882
5883
5884
5885
5886
5887
5888
5889
5890
    
```





```

6003 017620          93$:
6004          :CHECK THAT RHWC HAS 0
6005 017620 012737 000000 001124      MOV    #0,$GDDAT      ;GET .000 = 0
6006
6007 017626 017737 164230 001126      MOV    @RHWC,$BDDAT  ;R' J RHWC FOR COMPARISON
6008 017634 023737 001124 001126      CMP    $GDDAT,$BDDAT ;COMPARE EXPECTED
6009                                     ;DATA WITH DATA READ FROM
6010                                     ;RHWC
6011 017642 001401          BEQ    95$           ;BRANCH IF GOOD
6012 017644 104032          ERROR   32
6013
6014                                     ;AFTER SETTING "CLR" BIT #5
6015                                     ;IN RHCS2 TO INIT THE RH
6016                                     ;AND WRITING A PATTERN OF FLOATING
6017                                     ;ONES (WHICH IS 400,1000,2000,4000,
6018                                     ;10000,20000,40000,100000)
6019                                     ;INTO RHDB TO FILL IT
6020                                     ;THEN ON READING RHDB THE
6021                                     ;TOTAL 8 TIMES
6022                                     ;RHWC SHOULD HAVE 0
6023                                     ;BUT CONTAINED WHAT IS
6024 017646          95$:
6025
6026
6027          ;*****
6028          ;*TEST 17 SILO TEST 7 (FLOATING 0 IN LOWER BYTE)
6029
6030          ;*
6031          ;* AFTER A RH (EAR IS GIVEN (SET BIT #5 IN RHCS2)
6032          ;* RHCS2 IS CHECKED TO HAVE "IR" (BIT #6) HIGH,
6033          ;* TOGETHER WITH UNIT NUMBER
6034          ;* EIGHT WORDS OF FLOATING ZEROS 177776, 177775, 177773,
6035          ;* 177767, 177766, 177737, 177677, 177577
6036          ;* IS WRITTEN INTO RHDB
6037          ;* "OR" (BIT #7 IN RHCS2) IS WAITED FOR BY A TRAP
6038          ;* INSTRUCTION CALLED "WAT"
6039          ;* THEN FCS2 IS CHECKED TO HAVE "IR" LOW AND
6040          ;* "OR" TOGETHER WITH THE UNIT NUMBER
6041          ;* THEN RHDB IS READ AND COMPARED TO HAVE THE
6042          ;* RIGHT VALUE AFTER EACH OF THE EIGHT READS.
6043          ;* THE RHCS2 IS CHECKED TO HAVE "IR"
6044          ;* AT UNIT NUMBER
6045          ;* IN RHCS1, RHCS3, RHBA, RHBAE, RHWC ARE
6046          ;* CHECKED TO HAVE THE APPROPRIATE VALUE
6047          ;*****
6047 017646 000004          *ST17. SCOPE
6048 017650 012706 001000          MOV    #STACK,SP      ;RESET STACK
6049 017654 012737 000017 004656      MOV    #17,@*STNM    ;SAVE TEST NUMBER
6050
6051 017662 004737 040322          JSR    PC,@*CLDISK   ;GIVE RH INITIALIZE
6052                                     ;SETUP UNIT NUMBER
6053                                     ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
6054
6055
6056          ;CHECK THAT RHCS2 HAS IR
6057 017666 012737 000100 001124      MOV    #IR,$GDDAT    ;GET GOOD = 100
6058 017674 053737 005010 001124      BIS    UNIT,$GDDAT   ;INCLUDE UNIT NUMBER

```



6115  
6116  
6117  
6118  
6119  
6120  
6121  
6122  
6123  
6124  
6125  
6126  
6127  
6128  
6129  
6130  
6131  
6132  
6133  
6134  
6135  
6136  
6137  
6138  
6139  
6140  
6141  
6142  
6143  
6144  
6145  
6146  
6147  
6148  
6149  
6150  
6151  
6152  
6153  
6154  
6155  
6156  
6157  
6158  
6159  
6160  
6161  
6162  
6163  
6164  
6165  
6166  
6167  
6168  
6169  
6170

020010

67\$:

020010 012737 000001 004664  
020016 012737 177776 001124  
020024 017737 164052 001126  
020032 023737 001124 001126  
020040 001401  
020042 104024

MOV #1,SILONM  
;CHECK THAT RHDB HAS 177776  
MOV #177776,\$GDDAT  
MOV @RHDB,\$BDDAT  
CMP \$GDDAT,\$BDDAT  
BEQ 69\$  
ERROR 24

;RHCS2 SHOULD HAVE OR  
;=200  
;TOGETHER WITH UNIT NUMBER  
;BUT CONTAINED WHAT IS  
;GIVEN IN BAD RHCS2

;FIRST WORD TO BE READ  
;GET GOOD = 0  
;READ RHDB FOR COMPARISON  
;COMPARE EXPECTED  
;DATA WITH DATA READ FROM  
;RHDB  
;BRANCH IF GOOD

;AFTER SETTING "CLR" BIT #5  
;IN RHCS2 TO INIT THE RH  
;AND WRITING A PATTERN OF FLOATING  
;ZEROS - 177776, 177775, 177773, 177767,  
;177757, 177737, 177677, 177577  
;INTO RHDB TO FILL IT  
;THEN ON READING RHDB THE  
;FIRST TIME  
;RHDB SHOULD HAVE 177776  
;BUT CONTAINED WHAT IS  
;GIVEN IN BAD RHDB

020044

69\$:

020044 012737 000002 004664  
020052 012737 177775 001124  
020060 017737 164016 001126  
020066 023737 001124 001126  
020074 001401  
020076 104024

MOV #2,SILONM  
;CHECK THAT RHDB HAS 177775  
MOV #177775,\$GDDAT  
MOV @RHDB,\$BDDAT  
CMP \$GDDAT,\$BDDAT  
BEQ 71\$  
ERROR 24

;SECOND WORD TO BE READ  
;GET GOOD = 0  
;READ RHDB FOR COMPARISON  
;COMPARE EXPECTED  
;DATA WITH DATA READ FROM  
;RHDB  
;BRANCH IF GOOD

;AFTER SETTING "CLR" BIT #5  
;IN RHCS2 TO INIT THE RH  
;AND WRITING A PATTERN OF FLOATING  
;ZEROS - 177776, 177775, 177773, 177767,  
;177757, 177737, 177677, 177577  
;INTO RHDB TO FILL IT  
;THEN ON READING RHDB THE  
;SECOND TIME  
;RHDB SHOULD HAVE 177775  
;BUT CONTAINED WHAT IS  
;GIVEN IN BAD RHDB

020100

71\$:





```

6227 ;RHDB
6228 020220 001401 BEQ 77$ ;BRANCH IF GOOD
6229 020222 104024 ERROR 24
6230 ;AFTER SETTING "CLR" BIT #5
6231 ;IN RHCS2 TO INIT THE RH
6232 ;AND WRITING A PATTERN OF FLOATING
6233 ;ZEROS - 177776, 177775, 177773, 177767,
6234 ;177757, 177737, 177677, 177577
6235 ;INTO RHDB TO FILL IT
6236 ;THEN ON READING RHDB THE
6237 ;FIFTH TIME
6238 ;RHDB SHOULD HAVE 177757
6239 ;BUT CONTAINED WHAT IS
6240 ;GIVEN IN BAD RHDB
6241 020224 77$:
6242
6243
6244 020224 012737 000006 004664 MOV #6,SILONM ;SIXTH WORD TO BE READ
6245 ;CHECK THAT RHDB HAS 177737
6246 020232 012737 177737 001124 MOV #177737,$GDDAT ;GET GOOD = 0
6247
6248 020240 017737 163636 001126 MOV @RHDB,$BDDAT ;READ RHDB FOR COMPARISON
6249 020246 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
6250 ;DATA WITH DATA READ FROM
6251 ;RHDB
6252 020254 001401 BEQ 79$ ;BRANCH IF GOOD
6253 020256 104024 ERROR 24
6254 ;AFTER SETTING "CLR" BIT #5
6255 ;IN RHCS2 TO INIT THE RH
6256 ;AND WRITING A PATTERN OF FLOATING
6257 ;ZEROS - 177776, 177775, 177773, 177767,
6258 ;177757, 177737, 177677, 177577
6259 ;INTO RHDB TO FILL IT
6260 ;THEN ON READING RHDB THE
6261 ;SIXTH TIME
6262 ;RHDB SHOULD HAVE 177737
6263 ;BUT CONTAINED WHAT IS
6264 ;GIVEN IN BAD RHDB
6265 020260 79$:
6266
6267
6268 020260 012737 000007 004664 MOV #7,SILONM ;SEVENTH WORD TO BE READ
6269 ;CHECK THAT RHDB HAS 177677
6270 020266 012737 177677 001124 MOV #177677,$GDDAT ;GET GOOD = 0
6271
6272 020274 017737 163602 001126 MOV @RHDB,$BDDAT ;READ RHDB FOR COMPARISON
6273 020302 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
6274 ;DATA WITH DATA READ FROM
6275 ;RHDB
6276 020310 001401 BEQ 81$ ;BRANCH IF GOOD
6277 020312 104024 ERROR 24
6278 ;AFTER SETTING "CLR" BIT #5
6279 ;IN RHCS2 TO INIT THE RH
6280 ;AND WRITING A PATTERN OF FLOATING
6281 ;ZEROS - 177776, 177775, 177773, 177767,
6282 ;177757, 177737, 177677, 177577

```

```

6283 ; INTO RHDB TO FILL IT
6284 ; THEN ON READING RHDB THE
6285 ; SEVENTH TIME
6286 ; RHDB SHOULD HAVE 177677
6287 ; BUT CONTAINED WHAT IS
6288 ; GIVEN IN BAD RHDB
6289 020314          91$:
6290
6291
6292 020314 012737 003010 004664      MOV      #8.,SILONM      ; EIGHTH WORD TO BE READ
6293 ; CHECK THAT RHDB HAS 177577
6294 020322 012737 177577 001124      MOV      #177577,$GDDAT ; GET GOOD = 0
6295
6296 020330 017737 163546 001126      MOV      @RHDB,$BDDAT   ; READ RHDB FOR COMPARISON
6297 020336 023737 001124 001126      CMP      $GDDAT,$BDDAT ; COMPARE EXPECTED
6298 ; DATA WITH DATA READ FROM
6299 ; RHDB
6300 020344 001401                      BEQ      83$           ; BRANCH IF GOOD
6301 020346 104024                      ERROR   24
6302
6303 ; AFTER SETTING "CLR" BIT #5
6304 ; IN RHCS2 TO INIT THE RH
6305 ; AND WRITING A PATTERN OF FLOATING
6306 ; ZEROS - 177776, 177775, 177773, 177767,
6307 ; 177757, 177737, 177677, 177577
6308 ; INTO RHDB TO FILL IT
6309 ; THEN ON READING RHDB THE
6310 ; EIGHTH TIME
6311 ; RHDB SHOULD HAVE 177577
6312 ; BUT CONTAINED WHAT IS
6313 ; GIVEN IN BAD RHDB
6314 020350          83$:
6315
6316 ; CHECK THAT RHCS2 HAS IR
6317 020350 012737 000100 001124      MOV      #IR,$GDDAT    ; GET GOOD = 100
6318 020356 053737 005010 001124      BIS      UNIT,$GDDAT   ; INCLUDE UNIT NUMBER
6319
6320 020364 017737 163500 001126      MOV      @RHCS2,$BDDAT ; READ RHCS2 FOR COMPARISON
6321 020372 023737 001124 001126      CMP      $GDDAT,$BDDAT ; COMPARE EXPECTED
6322 ; DATA WITH DATA READ FROM
6323 ; RHCS2
6324 020400 051401                      BEQ      85$           ; BRANCH IF GOOD
6325 020402 104025                      ERROR   25
6326
6327 ; AFTER SETTING "CLR" BIT #5
6328 ; IN RHCS2 TO INIT THE RH
6329 ; AND WRITING A PATTERN OF FLOATING
6330 ; ZEROS - 177776, 177775, 177773, 177767,
6331 ; 177757, 177737, 177677, 177577
6332 ; INTO RHDB TO FILL IT
6333 ; THEN ON READING RHDB THE
6334 ; TOTAL 8 TIMES
6335 ; RHCS2 SHOULD HAVE IR
6336 ; =100
6337 ; TOGETHER WITH UNIT NUMBER
6338 ; BUT CONTAINED WHAT IS
6339 ; GIVEN IN BAD RHCS2

```

```

6339 020404          85$:
6340          ;CHECK THAT RHCS1 HAS DVA!RDY
6341 020404 012737 004200 001124  MOV      #DVA!RDY,$GDDAT ;GET GOOD = 4200
6342          ;
6343 020412 017737 163442 001126  MOV      @RHCS1,$BDDAT ;READ RHCS1 FOR COMPARISON
6344 020420 023737 001124 001126  CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED
6345          ;DATA WITH DATA READ FROM
6346          ;RHCS1
6347 020426 001401          BEQ      87$ ;BRANCH IF GOOD
6348 020430 104026          ERROR    26
6349          ;
6350          ;AFTER SETTING "CLR" BIT #5
6351          ;IN RHCS2 TO INIT THE RH
6352          ;AND WRITING A PATTERN OF FLOATING
6353          ;ZEROS - 177776, 177775, 177773, 177767,
6354          ;177757, 177737, 177677, 177577
6355          ;INTO RHDB TO FILL IT
6356          ;THEN ON READING RHDB THE
6357          ;TOTAL 8 TIMES
6358          ;RHCS1 SHOULD HAVE DVA!RDY
6359          ;=4200
6360          ;BUT CONTAINED WHAT IS
6361 020432          87$: ;GIVEN IN BAD RHCS1
6362          ;CHECK THAT RHCS3 HAS 0
6363 020432 012737 000000 001124  MOV      #0,$GDDAT ;GET GOOD = 0
6364          ;
6365 020440 017737 163466 001126  MOV      @RHCS3,$BDDAT ;READ RHCS3 FOR COMPARISON
6366 020446 023737 001124 001126  CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED
6367          ;DATA WITH DATA READ FROM
6368          ;RHCS3
6369 020454 001401          BEQ      89$ ;BRANCH IF GOOD
6370 020456 104027          ERROR    27
6371          ;
6372          ;AFTER SETTING "CLR" BIT #5
6373          ;IN RHCS2 TO INIT THE RH
6374          ;AND WRITING A PATTERN OF FLOATING
6375          ;ZEROS - 177776, 177775, 177773, 177767,
6376          ;177757, 177737, 177677, 177577
6377          ;INTO RHDB TO FILL IT
6378          ;THEN ON READING RHDB THE
6379          ;TOTAL 8 TIMES
6380          ;RHCS3 SHOULD HAVE 0
6381          ;BUT CONTAINED WHAT IS
6382 020460          89$: ;GIVEN IN BAD RHCS3
6383          ;CHECK THAT RHBA HAS 0
6384 020460 012737 000000 001124  MOV      #0,$GDDAT ;GET GOOD = 0
6385          ;
6386 020466 017737 163372 001126  MOV      @RHBA,$BDDAT ;READ RHBA FOR COMPARISON
6387 020474 023737 001124 001126  CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED
6388          ;DATA WITH DATA READ FROM
6389          ;RHBA
6390 020502 001401          BEQ      91$ ;BRANCH IF GOOD
6391 020504 104030          ERROR    30
6392          ;
6393          ;AFTER SETTING "CLR" BIT #5
6394          ;IN RHCS2 TO INIT THE RH
        
```

6395.  
6396.  
6397  
6398  
6399  
6400  
6401  
6402  
6403  
6404  
6405  
6406  
6407  
6408  
6409  
6410  
6411  
6412  
6413  
6414  
6415  
6416  
6417  
6418  
6419  
6420  
6421  
6422  
6423  
6424  
6425  
6426  
6427  
6428  
6429  
6430  
6431  
6432  
6433  
6434  
6435  
6436  
6437  
6438  
6439  
6440  
6441  
6442  
6443  
6444  
6445  
6446  
6447  
6448  
6449  
6450

020506  
020506  
020514  
020522  
020530  
020532  
020534  
020534  
020542  
020550  
020556  
020560  
020562

012737 000000 001124  
017737 163410 001126  
023737 001124 001126  
001401  
104031  
012737 000000 001124  
017737 163314 001126  
023737 001124 001126  
001401  
104032

91\$:  
93\$:  
95\$:

;CHECK THAT RHBAE HAS 0  
MOV #0,\$GDDAT  
MOV @RHBAE,\$BDDAT  
CMP \$GDDAT,\$BDDAT  
BEQ 93\$  
ERROR 31  
;CHECK THAT RHC HAS 0  
MOV #0,\$GDDAT  
MOV @RHC,\$BDDAT  
CMP \$GDDAT,\$BDDAT  
BEQ 95\$  
ERROR 32

:ZEROS - 177776, 177775, 177773, 177767,  
:177757, 177737, 177677, 177577  
:INTO RHDB TO FILL IT  
:THEN ON READING RHDB THE  
:TOTAL 8 TIMES  
:RHBA SHOULD HAVE 0  
:BUT CONTAINED WHAT IS  
:GIVEN IN BAD RHBA  
;GET GOOD = 0  
;READ RHBAE FOR COMPARISON  
;COMPARE EXPECTED  
;DATA WITH DATA READ FROM  
;RHBAE  
;BRANCH IF GOOD  
;AFTER SETTING "CLR" BIT #5  
;IN RHCS2 TO INIT THE RH  
;AND WRITING A PATTERN OF FLOATING  
:ZEROS - 177776, 177775, 177773, 177767,  
:177757, 177737, 177677, 177577  
:INTO RHDB TO FILL IT  
:THEN ON READING RHDB THE  
:TOTAL 8 TIMES  
:RHBAE SHOULD HAVE 0  
:BUT CONTAINED WHAT IS  
:GIVEN IN BAD RHBAE  
;GET GOOD = 0  
;READ RHC FOR COMPARISON  
;COMPARE EXPECTED  
;DATA WITH DATA READ FROM  
;RHC  
;BRANCH IF GOOD  
;AFTER SETTING "CLR" BIT #5  
;IN RHCS2 TO INIT THE RH  
;AND WRITING A PATTERN OF FLOATING  
:ZEROS - 177776, 177775, 177773, 177767,  
:177757, 177737, 177677, 177577  
:INTO RHDB TO FILL IT  
:THEN ON READING RHDB THE  
:TOTAL 8 TIMES  
:RHC SHOULD HAVE 0  
:BUT CONTAINED WHAT IS  
:GIVEN IN BAD RHC

::\*\*\*\*\*  
;\*TEST 20 SILO TEST 8 (FLOATING ZEROS IN UPPER BYTE)

```

6451 : * AFTER A RH CLEAR IS GIVEN (SET BIT #5 IN RHCS2)
6452 : * RHCS2 IS CHECKED TO HAVE "IR" (BIT #6) HIGH,
6453 : * TOGETHER WITH UNIT NUMBER
6454 : * EIGHT WORDS, A PATTERN OF FLOATING ZEROS IN UPPER BYTE
6455 : * 177377, 177677, 175777, 173777, 167777, 157777, 137777, 77777
6456 : * IS WRITTEN INTO RHDB
6457 : * "OR" (BIT #7 IN RHCS2) IS WAITED FOR BY A TRAP
6458 : * INSTRUCTION CALLED "WAT"
6459 : * THEN RHCS2 IS CHECKED TO HAVE "IR" LOW AND
6460 : * "OR" HIGH TOGETHER WITH THE UNIT NUMBER
6461 : * THEN RHDB IS READ AND COMPARED TO HAVE THE
6462 : * RIGHT VALUE AFTER EACH OF THE EIGHT READS.
6463 : * THEN RHCS2 IS CHECKED TO HAVE "IR"
6464 : * AND UNIT NUMBER
6465 : * THEN RHCS1, RHCS3, RHBA, RHBAE, RHWC ARE
6466 : * CHECKED TO HAVE THE APPROPRIATE VALUE
6467 : * *****
6468 020562 000004 +ST20: SCOPE
6469 020564 012706 001000 MOV #STACK,SP ;RESET STACK
6470 020570 012737 00C020 004656 MOV #20,#STNM ;SAVE TEST NUMBER
6471
6472 020576 004737 040322 JSR PC,#CLDISK ;GIVE RH INITIALIZE
6473 ;SETUP UNIT NUBER
6474 ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
6475
6476
6477 ;CHECK THAT RHCS2 HAS IR
6478 020602 012737 000100 001124 MOV #IR,$GDDAT ;GET GOOD = 100
6479 020610 053737 005010 001124 BIS UNIT,$GDDAT ;INCLUDE UNIT NUMBER
6480
6481 020616 017737 163246 001126 MOV #RHCS2,$BDDAT ;READ RHCS2 FOR COMPARISON
6482 020624 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
6483 ;DATA WITH DATA READ FROM
6484 ;RHCS2
6485 020632 001401 BEQ 65$ ;BRANCH IF GOOD
6486 020634 104006 ERROR 6
6487 ;AFTER SETTING "CLR" BIT #5
6488 ;IN RHCS2 TO INIT THE RH
6489 ;AND HAVING DONE NOTHING
6490 ;ELSE
6491 ;RHCS2 SHOULD HAVE IR
6492 ;=100
6493 ;TOGETHER WITH UNIT NUMBER
6494 ;BUT CONTAINED WHAT IS
6495 ;GIVEN IN BAD RHCS2
6496 020636 65$:
6497
6498 ;WRITE EIGHT WORDS INTO RHDB TO FILL IT
6499 ;A PATTERN OF FLOATING ZEROS IN UPPER BYTE
6500 ;DATA IS - 177377, 176777, 175777, 173777, 167777,
6501 ;157777, 137777, 77777
6502 020636 012701 177377 MOV #177377,R1 ;GETTING READY TO FLOAT ZEROS
6503 ;IN UPPER BYTE
6504 020642 012702 000010 MOV #8,R2 ;COUNTER -8 DATA WORDS
6505 020646 010177 163230 1$: MOV R1,#RHDB ;WRITE INTO RHDB
6506 020652 000261 SEC ;SET CARRY

```



;167777, 157777, 137777, 77777  
; INTO RHDB TO FILL IT  
; THEN ON READING RHDB THE  
; FIRST TIME  
; RHDB SHOULD HAVE 177377  
; BUT CONTAINED WHAT IS  
; GIVEN IN BAD RHDB

;SECOND WORD TO BE READ  
;CHECK THAT RHDB HAS 176777  
;GET GOOD = 0

;READ RHDB FOR COMPARISON  
;COMPARE EXPECTED  
;DATA WITH DATA READ FROM  
;RHDB  
;BRANCH IF GOOD

;AFTER SETTING "CLR" BIT #5  
;IN RHCS2 TO INIT THE RH  
;AND WRITING A PATTERN OF FLOATING  
;ZEROS IN UPPER BYTE  
;177377, 176777, 175777, 173777.  
;167777, 157777, 137777, 77777  
; INTO RHDB TO FILL IT  
; THEN ON READING RHDB THE  
; SECOND TIME  
; RHDB SHOULD HAVE 176777  
; BUT CONTAINED WHAT IS  
; GIVEN IN BAD RHDB

;THIRD WORD TO BE READ  
;CHECK THAT RHDB HAS 175777  
;GET GOOD = 0

;READ RHDB FOR COMPARISON  
;COMPARE EXPECTED  
;DATA WITH DATA READ FROM  
;RHDB  
;BRANCH IF GOOD

;AFTER SETTING "CLR" BIT #5  
;IN RHCS2 TO INIT THE RH  
;AND WRITING A PATTERN OF FLOATING  
;ZEROS IN UPPER BYTE  
;177377, 176777, 175777, 173777.  
;167777, 157777, 137777, 77777  
; INTO RHDB TO FILL IT  
; THEN ON READING RHDB THE  
; THIRD TIME  
; RHDB SHOULD HAVE 175777  
; BUT CONTAINED WHAT IS

6563  
6564  
6565  
6566  
6567  
6568  
6569  
6570 020760 69\$:

6571  
6572  
6573 020760 012737 000002 004664  
6574  
6575 020766 012737 176777 001124  
6576  
6577 020774 017737 163102 001126  
6578 021002 023737 001124 001126  
6579

6580  
6581 021010 001401  
6582 021012 104034

6583  
6584  
6585  
6586  
6587  
6588  
6589  
6590  
6591  
6592  
6593  
6594  
6595 021014 71\$:  
6596

6597  
6598 021014 012737 000003 004664  
6599  
6600 021022 012737 175777 001124  
6601  
6602 021030 017737 163046 001126  
6603 021036 023737 001124 001126  
6604

6605  
6606 021044 001401  
6607 021046 104024

6608  
6609  
6610  
6611  
6612  
6613  
6614  
6615  
6616  
6617  
6618





```

6675 021146 012737 157777 001124      MOV      #157777,$GDDAT ;GET GOOD = 0
6676
6677 021154 017737 162722 001126      MOV      @RHDB,$BDDAT ;READ RHDB FOR COMPARISON
6678 021162 023737 001124 001126      CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED
6679                                     ;DATA WITH DATA READ FROM
6680                                     ;RHDB
6681 021170 001401      BEQ      79$           ;BRANCH IF GOOD
6682 021172 104024      ERROR   24
6683                                     ;AFTER SETTING "CLR" BIT #5
6684                                     ;IN RHCS2 TO INIT THE RH
6685                                     ;AND WRITING A PATTERN OF FLOATING
6686                                     ;ZEROS IN UPPER BYTE
6687                                     ;177377, 176777, 175777, 173777.
6688                                     ;167777, 157777, 137777, 77777.
6689                                     ;INTO RHDB TO FILL IT
6690                                     ;THEN ON READING RHDB THE
6691                                     ;SIXTH TIME
6692                                     ;RHDB SHOULD HAVE 157777
6693                                     ;BUT CONTAINED WHAT IS
6694                                     ;GIVEN IN BAD RHDB
6695 021174      79$:
6696
6697
6698 021174 012737 000007 004664      MOV      #7,SILONM ;SEVENTH WORD TO BE READ
6699                                     ;CHECK THAT RHDB HAS 137777
6700 021202 012737 137777 001124      MOV      #137777,$GDDAT ;GET GOOD = 0
6701
6702 021210 017737 162666 001126      MOV      @RHDB,$BDDAT ;READ RHDB FOR COMPARISON
6703 021216 023737 001124 001126      CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED
6704                                     ;DATA WITH DATA READ FROM
6705                                     ;RHDB
6706 021224 001401      BEQ      81$           ;BRANCH IF GOOD
6707 021226 104024      ERROR   24
6708                                     ;AFTER SETTING "CLR" BIT #5
6709                                     ;IN RHCS2 TO INIT THE RH
6710                                     ;AND WRITING A PATTERN OF FLOATING
6711                                     ;ZEROS IN UPPER BYTE
6712                                     ;177377, 176777, 175777, 173777.
6713                                     ;167777, 157777, 137777, 77777.
6714                                     ;INTO RHDB TO FILL IT
6715                                     ;THEN ON READING RHDB THE
6716                                     ;SEVENTH TIME
6717                                     ;RHDB SHOULD HAVE 137777
6718                                     ;BUT CONTAINED WHAT IS
6719                                     ;GIVEN IN BAD RHDB
6720 021230      81$:
6721
6722
6723 021230 012737 000010 004664      MOV      #8,SILONM ;EIGHTH WORD TO BE READ
6724                                     ;CHECK THAT RHDB HAS 77777
6725 021236 012737 077777 001124      MOV      #77777,$GDDAT ;GET GOOD = 0
6726
6727 021244 017737 162632 001126      MOV      @RHDB,$BDDAT ;READ RHDB FOR COMPARISON
6728 021252 023737 001124 001126      CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED
6729                                     ;DATA WITH DATA READ FROM
6730                                     ;RHDB

```



```

6787
6788
6789
6790
6791
6792
6793
6794
6795 021346      87$:
6796
6797 021346 012737 000000 001124      MOV      #0,$GDDAT      ;CHECK THAT RHCS3 HAS 0
6798
6799 021354 017737 162552 001126      MOV      @RHCS3,$BDDAT  ;READ RHCS3 FOR COMPARISON
6800 021362 023737 001124 001126      CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED
6801
6802
6803 021370 001401      BEQ      89$           ;DATA WITH DATA READ FROM
6804 021372 104027      ERROR   27           ;RHCS3
6805
6806
6807
6808
6809
6810
6811
6812
6813
6814
6815
6816
6817 021374      89$:
6818
6819 021374 012737 000000 001124      MOV      #0,$GDDAT      ;CHECK THAT RHBA HAS 0
6820
6821 021402 017737 162456 001126      MOV      @RHBA,$BDDAT  ;READ RHBA FOR COMPARISON
6822 021410 023737 001124 001126      CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED
6823
6824
6825 021416 001401      BEQ      91$           ;DATA WITH DATA READ FROM
6826 021420 104030      ERROR   30           ;RHBA
6827
6828
6829
6830
6831
6832
6833
6834
6835
6836
6837
6838
6839 021422      91$:
6840
6841 021422 012737 000000 001124      MOV      #0,$GDDAT      ;CHECK THAT RHBAE HAS 0
6842

```

```

;167777, 157777, 137777, 77777
;INTO RHDB TO FILL IT
;THEN ON READING RHDB THE
;TOTAL 8 TIMES
;RHCS1 SHOULD HAVE DVA:RDY
;=4200
;BUT CONTAINED WHAT IS
;GIVEN IN BAD RHCS1

;AFTER SETTING "CLR" BIT #5
;IN RHCS2 TO INIT THE RH
;AND WRITING A PATTERN OF FLOATING
;ZEROS IN UPPER BYTE
;177377, 176777, 175777, 173777,
;167777, 157777, 137777, 77777
;INTO RHDB TO FILL IT
;THEN ON READING RHDB THE
;TOTAL 8 TIMES
;RHCS3 SHOULD HAVE 0
;BUT CONTAINED WHAT IS
;GIVEN IN BAD RHCS3

;AFTER SETTING "CLR" BIT #5
;IN RHCS2 TO INIT THE RH
;AND WRITING A PATTERN OF FLOATING
;ZEROS IN UPPER BYTE
;177377, 176777, 175777, 173777,
;167777, 157777, 137777, 77777
;INTO RHDB TO FILL IT
;THEN ON READING RHDB THE
;TOTAL 8 TIMES
;RHBA SHOULD HAVE 0
;BUT CONTAINED WHAT IS
;GIVEN IN BAD RHBA

```

```

6843 021430 017737 162474 001126      MOV    @RHBAE,$BDDAT ;READ RHBAE FOR COMPARISON
6844 021436 023737 001124 001126      CMP    $GDDAT,$BDDAT ;COMPARE EXPECTED
6845                                     ;DATA WITH DATA READ FROM
6846                                     ;RHBAE
6847 021444 001401      BEQ    93$           ;BRANCH IF GOOD
6848 021446 104031      ERROR  31
6849                                     ;AFTER SETTING "CLR" BIT #5
6850                                     ;IN RHCS2 TO INIT THE RH
6851                                     ;AND WRITING A PATTERN OF FLOATING
6852                                     ;ZEROS IN UPPER BYTE
6853                                     ;177377, 176777, 175777, 173777.
6854                                     ;167777, 157777, 137777, 77777
6855                                     ;INTO RHDB TO FILL IT
6856                                     ;THEN ON READING RHDB THE
6857                                     ;TOTAL 8 TIMES
6858                                     ;RHBAE SHOULD HAVE 0
6859                                     ;BUT CONTAINED WHAT IS
6860                                     ;GIVEN IN BAD RHBAE
6861 021450                                     93$:
6862                                     ;CHECK THAT RHCW HAS 0
6863 021450 012737 000000 001124      MOV    #0,$GDDAT    ;GET GOOD = 0
6864
6865 021456 017737 162400 001126      MOV    @RHCW,$GDDAT ;READ RHCW FOR COMPARISON
6866 021464 023737 001124 001126      CMP    $GDDAT,$BDDAT ;COMPARE EXPECTED
6867                                     ;DATA WITH DATA READ FROM
6868                                     ;RHCW
6869 021472 001401      BEQ    95$           ;BRANCH IF GOOD
6870 021474 104032      ERROR  32
6871                                     ;AFTER SETTING "CLR" BIT #5
6872                                     ;IN RHCS2 TO INIT THE RH
6873                                     ;AND WRITING A PATTERN OF FLOATING
6874                                     ;ZEROS IN UPPER BYTE
6875                                     ;177377, 176777, 175777, 173777,
6876                                     ;167777, 157777, 137777, 77777
6877                                     ;INTO RHDB TO FILL IT
6878                                     ;THEN ON READING RHDB THE
6879                                     ;TOTAL 8 TIMES
6880                                     ;RHCW SHOULD HAVE 0
6881                                     ;BUT CONTAINED WHAT IS
6882                                     ;GIVEN IN BAD RHCW
6883 021476                                     95$:
6884
6885
6886 ;:*****
6887 ;*TEST 21          RHCS1 - MCPE BIT #13 (PARITY LINE = 0)
6888
6889 ;*
6890 ;* AN RH CLEAR (SET BIT #5 IN RHCS2) IS GIVEN TO
6891 ;* CLEAR ALL DEVICE REGISTERS
6892 ;* THEN RHER1 (ERROR REGISTER 1) IS CHECKED TO HAVE
6893 ;* ZEROS
6894 ;* SET "PAT" (BIT #4 IN RHCS2) TO INVERT PARITY CHECKING
6895 ;* READ ANY DEVICE REGISTER - HERE RHER1
6896 ;* READ AND CHECK RHCS1 TO CONTAIN SC (BIT #15)
6897 ;* AND MCPE (BIT #13)
6898 ;* WRITE "1" INTO TRE (BIT #14 IN RHCS1)
6899 ;* READ RHCS1 SC, MCPE, AND RDY SHOULD BE SET

```

```

6899          : *      GIVE AN RH CLEAR (CLR - BIT #5 IN RHCS2)
6900          : *      CHECK RHCS1 TO HAVE RDY
6901          : *      CHECK RHCS2 TO HAVE ONLY IR AND UNIT NUMBER
6902          : *      CHECK RHCS3, RHBA, RHBAE, RHWC TO HAVE APPROPRIATE
6903          : *      VALUES.
6904          : *****
6905 021476 000004          †ST21: SCOPE
6906 021500 012706 001000  MOV      #STACK,SP      ;RESET STACK
6907 021504 012737 000021 004656  MOV      #21,#†STNM      ;SAVE TEST NUMBER
6908
6909 021512 004737 040322          JSR      PC.#CLDISK      ;GIVE RH INITIALIZE
6910                                     ;SETUP UNIT NUBER
6911                                     ;CLEAR RHWC AND FUNCTION BITS IN RHCSI
6912
6913
6914          :CHECK THAT RHER1 HAS 0
6915 021516 012737 000000 001124  MOV      #0,$GDDAT      ;GET GOOD = 0
6916
6917 021524 017737 162344 001126  MOV      @RHER1,$BDDAT  ;READ RHER1 FOR COMPARISON
6918 021532 023737 001124 001126  CMP      $GDDAT,$BDDAT  ;COMPARE EXPECTED
6919                                     ;DATA WITH DATA READ FROM
6920                                     ;RHER1
6921 021540 001401          BEQ      65$            ;BRANCH IF GOOD
6922 021542 104033          ERROR    33
6923                                     ;AFTER SETTING "CLR" BIT #15
6924                                     ;IN RHCS2 TO INIT THE RH
6925                                     ;AND HAVING DONE NOTHING ELSE
6926                                     ;RHER1 SHOULD HAVE 0
6927                                     ;BUT CONTAINED WHAT IS
6928                                     ;GIVEN IN BAD RHER1
6929 021544          65$:
6930
6931          :TO INVERT THE PARITY CHECKING ON THE CONTROL BUS
6932          : "PAT" BIT #4 IN RHCS2 IS SET
6933 021544 052777 000020 162316  BIS      #PAT,@RHCS2    ;SET PAT = 20 IN RHCS2
6934
6935
6936          :CHECK THAT RHER1 HAS 0
6937 021552 012737 000000 001124  MOV      #0,$GDDAT      ;GET GOOD = 0
6938
6939 021560 017737 162310 001126  MOV      @RHER1,$BDDAT  ;READ RHER1 FOR COMPARISON
6940 021566 023737 001124 001126  CMP      $GDDAT,$BDDAT  ;COMPARE EXPECTED
6941                                     ;DATA WITH DATA READ FROM
6942                                     ;RHER1
6943 021574 001401          BEQ      67$            ;BRANCH IF GOOD
6944 021576 104034          ERROR    34
6945                                     ;AFTER CLEARING THE RH AND
6946                                     ;DEVICE BY AN RH CLEAR
6947                                     ;AND READING RHER1 WITH
6948                                     ;WRONG PARITY CHECKING
6949                                     ;RHER1 SHOULD HAVE 0
6950                                     ;BUT CONTAINED WHAT IS
6951                                     ;GIVEN IN BAD RHER1
6952 021600          67$:
6953
6954          :HAVING READ RHER1 WITH WRONG PARITY BEING

```

```

6955 ;CHECKED MCPE - BIT #13 AND SC BIT #15 IN RHCSI SHOULD BE SET
6956
6957 ;CHECK THAT RHCSI HAS 4200
6958 021500 012737 004200 001124 MOV #4200,$GDDAT ;GET GOOD = 124200
6959
6960 021606 017737 162246 001126 MOV @RHCSI,$BDDAT ;READ RHCSI FOR COMPARISON
6961 021614 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
6962 ;DATA WITH DATA READ FROM
6963 ;RHCSI
6964 021622 001401 BEQ 69$ ;BRANCH IF GOOD
6965 021624 104035 ERROR 35
6966 ;AFTER CLEARING THE RH AND
6967 ;DEVICE BY AN RH CLEAR
6968 ;AND READING RHER1 WITH
6969 ;WRONG PARITY CHECKING
6970 ;RHCSI SHOULD HAVE 4200
6971 ;=124200
6972 ;BUT CONTAINED WHAT IS
6973 ;GIVEN IN BAD RHCSI
6974 021626 69$:
6975
6976 ;WRITE A ONE INTO TRE (RHCSI - BIT #14)
6977 ;THIS SHOULD NOT CLEAR MCPE OR SC
6978 021626 012777 040000 162224 MOV #TRE,@RHCSI ;WRITE "1" INTO TRE IN RHCSI
6979
6980
6981
6982 ;CHECK THAT RHCSI HAS 104200
6983 021634 012737 104200 001124 MOV #104200,$GDDAT ;GET GOOD = 124200
6984
6985 021642 017737 162212 001126 MOV @RHCSI,$BDDAT ;READ RHCSI FOR COMPARISON
6986 021650 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
6987 ;DATA WITH DATA READ FROM
6988 ;RHCSI
6989 021656 001401 BEQ 71$ ;BRANCH IF GOOD
6990 021660 104036 ERROR 36
6991 ;AFTER CLEARING THE RH AND
6992 ;DEVICE REGISTERS BY AN
6993 ;RH CLEAR
6994 ;RHER1 WAS CHECKED TO HAVE
6995 ;ZEROS
6996 ;WRONG PARITY CHECKING WAS
6997 ;SET BY "PAT" BIT IN RHCS2
6998 ;RHER1 WAS READ TO SET
6999 ;MCPE AND SC IN RHCSI
7000 ;ON WRITING "1" INTO TRE
7001 ;RHCSI SHOULD HAVE 104200
7002 ;=124200
7003 ;BUT CONTAINED WHAT IS
7004 ;GIVEN IN BAD RHCSI
7005 021662 71$:
7006
7007 ;NOW ERRORS WILL BE CLEARED BY RH CLEAR
7008 ;SET "CLR" BIT #5 IN RHCS2
7009 021662 012777 000040 162200 MOV #CLR,@RHCS2 ;CLEAR BY RH INIT
7010 021670 013777 005010 162172 MOV UNIT,@RHCS2 ;REINSTATE UNIT NUMBER

```

```

7011
7012
7013
7014 021676 012737 004200 001124      ;CHECK THAT RHCS1 HAS DVA!RDY
      MOV      #DVA!RDY,$GDDAT ;GET GOOD = 4200
7015
7016 021704 017737 162150 001126      MOV      @RHCS1,$BDDAT ;READ RHCS1 FOR COMPARISON
7017 021712 023737 001124 001126      CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED
7018                                     ;DATA WITH DATA READ FROM
7019                                     ;RHCS1
7020 021720 001401                                     BEQ      73$ ;BRANCH IF GOOD
7021 021722 104037                                     ERROR    37
7022                                     ;AFTER CLEARING THE RH AND
7023                                     ;DEVICE REGISTERS BY AN
7024                                     ;RH CLEAR
7025                                     ;RHER1 WAS CHECKED TO HAVE
7026                                     ;ZEROS
7027                                     ;WRONG PARITY CHECKING WAS
7028                                     ;SET BY "PAT" BIT IN RHCS2
7029                                     ;RHER1 WAS READ TO SET
7030                                     ;MCPE AND SC IN RHCS1
7031                                     ;ON WRITING "1" INTO TRE
7032                                     ;THEN SETTING "CLR" IN RHCS1
7033                                     ;RHCS1 SHOULD HAVE DVA!RDY
7034                                     ;=4200
7035                                     ;BUT CONTAINED WHAT IS
7036                                     ;GIVEN IN BAD RHCS1
7037 021724                                     73$:
7038                                     ;CHECK THAT RHCS2 HAS IR
7039 021724 012737 000100 001124      MOV      #IR,$GDDAT ;GET GOOD = 100
7040 021732 053737 005010 001124      BIS      UNIT,$GDDAT ;INCLUDE UNIT NUMBER
7041
7042 021740 017737 162124 001126      MOV      @RHCS2,$BDDAT ;READ RHCS2 FOR COMPARISON
7043 021746 023737 001124 001126      CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED
7044                                     ;DATA WITH DATA READ FROM
7045                                     ;RHCS2
7046 021754 001401                                     BEQ      75$ ;BRANCH IF GOOD
7047 021756 104040                                     ERROR    40
7048                                     ;AFTER CLEARING THE RH AND
7049                                     ;DEVICE REGISTERS BY AN
7050                                     ;RH CLEAR
7051                                     ;RHER1 WAS CHECKED TO HAVE
7052                                     ;ZEROS
7053                                     ;WRONG PARITY CHECKING WAS
7054                                     ;SET BY "PAT" BIT IN RHCS2
7055                                     ;RHER1 WAS READ TO SET
7056                                     ;MCPE AND SC IN RHCS1
7057                                     ;ON WRITING "1" INTO TRE
7058                                     ;THEN SETTING "CLR" IN RHCS1
7059                                     ;RHCS2 SHOULD HAVE IR
7060                                     ;=100
7061                                     ;TOGETHER WITH UNIT NUMBER
7062                                     ;BUT CONTAINED WHAT IS
7063                                     ;GIVEN IN BAD RHCS2
7064 021760                                     75$:
7065                                     ;CHECK THAT RHCS3 HAS 0
7066 021760 012737 000000 001124      MOV      #0,$GDDAT ;GET GOOD = 0

```



7067										
7068	021766	017737	162140	001126	MOV	DRHCS3,\$BDDAT				; READ RHCS3 FOR COMPARISON
7069	021774	023737	001124	001126	CMP	\$GDDAT,\$BDDAT				; COMPARE EXPECTED
7070										; DATA WITH DATA READ FROM
7071										; RHCS3
7072	022002	001401			BEQ	77\$				; BRANCH IF GOOD
7073	022004	104041			ERROR	41				
7074										; AFTER CLEARING THE RH AND
7075										; DEVICE REGISTERS BY AN
7076										; RH CLEAR
7077										; RHER1 WAS CHECKED TO HAVE
7078										; ZEROS
7079										; WRONG PARITY CHECKING WAS
7080										; SET BY "PAT" BIT IN RHCS2
7081										; RHER1 WAS READ TO SET
7082										; MCPE AND SC IN RHCS1
7083										; ON WRITING "1" INTO TRE
7084										; THEN SETTING "CLR" IN RHCS1
7085										; RHCS3 SHOULD HAVE 0
7086										; BUT CONTAINED WHAT IS
7087										; GIVEN IN BAD RHCS3
7088	022006									
7089										77\$: ; CHECK THAT RHBA HAS 0
7090	022006	012737	000000	001124	MOV	#0,\$GDDAT				; GET GOOD = 0
7091										
7092	022014	017737	162044	001126	MOV	DRHBA,\$BDDAT				; READ RHBA FOR COMPARISON
7093	022022	023737	001124	001126	CMP	\$GDDAT,\$BDDAT				; COMPARE EXPECTED
7094										; DATA WITH DATA READ FROM
7095										; RHBA
7096	022030	001401			BEQ	79\$				; BRANCH IF GOOD
7097	022032	104042			ERROR	42				
7098										; AFTER CLEARING THE RH AND
7099										; DEVICE REGISTERS BY AN
7100										; RH CLEAR
7101										; RHER1 WAS CHECKED TO HAVE
7102										; ZEROS
7103										; WRONG PARITY CHECKING WAS
7104										; SET BY "PAT" BIT IN RHCS2
7105										; RHER1 WAS READ TO SET
7106										; MCPE AND SC IN RHCS1
7107										; ON WRITING "1" INTO TRE
7108										; THEN SETTING "CLR" IN RHCS1
7109										; RHBA SHOULD HAVE 0
7110										; BUT CONTAINED WHAT IS
7111										; GIVEN IN BAD RHBA
7112	022034									
7113										79\$: ; CHECK THAT RHBAE HAS 0
7114	022034	012737	000000	001124	MOV	#0,\$GDDAT				; GET GOOD = 0
7115										
7116	022042	017737	162062	001126	MOV	DRHBAE,\$BDDAT				; READ RHBAE FOR COMPARISON
7117	022050	023737	001124	001126	CMP	\$GDDAT,\$BDDAT				; COMPARE EXPECTED
7118										; DATA WITH DATA READ FROM
7119										; RHBAE
7120	022056	001401			BEQ	81\$				; BRANCH IF GOOD
7121	022060	104043			ERROR	43				
7122										; AFTER CLEARING THE RH AND

7123  
7124  
7125  
7126  
7127  
7128  
7129  
7130  
7131  
7132  
7133  
7134  
7135  
7136  
7137  
7138  
7139  
7140  
7141  
7142  
7143  
7144  
7145  
7146  
7147  
7148  
7149  
7150  
7151  
7152  
7153  
7154  
7155  
7156  
7157  
7158  
7159  
7160  
7161  
7162  
7163  
7164  
7165  
7166  
7167  
7168  
7169  
7170  
7171  
7172  
7173  
7174  
7175  
7176  
7177  
7178

022062

022062 012737 000000 001124

022070 017737 161766 001126

022076 023737 001124 001126

022104 001401

022106 104044

022110

022110 000004

022112 012706 001000

81\$:

:CHECK THAT RHW C HAS 0  
MOV #0,\$GDDAT

MOV 2RHW C,\$BDDAT  
CMP \$GDDAT,\$BDDAT

BEQ 83\$  
ERROR 44

83\$:

::\*\*\*\*\*  
\*TEST 22 RHCS1 - MCPE BIT #13 (PARITY LINE = 1)

\* AN RH CLEAR (SET BIT #5 IN RHCS2) IS GIVEN TO CLEAR  
\* ALL DEVICE REGISTERS  
\* WRITE A ONE INTO DISK ADDRESS REGISTER (RHDA)  
\* (IN TAPE DRIVES CALLED REGISTER)  
\* SET "PAT" (RHCS2 BIT #4) THIS WILL INVERT THE PARITY CHECKING  
\* READ RHDA AND COMPARE IT HAS ONE IN IT  
\* THIS READING SHOULD SET SC, MCPE IN RHCS1  
\* CHECK RHCS1 TO HAVE ONLY R0Y SET  
\* CHECK RHCS2, RHCS3, RHBA, RHBAE, RHW C TO HAVE APPROPRIATE  
\* VALUES

↑ST22: SCOPE  
MOV #STACK,SP ;RESET STACK

;DEVICE REGISTERS BY AN  
;RH CLEAR  
;RHER1 WAS CHECKED TO HAVE  
;ZEROS  
;WRONG PARITY CHECKING WAS  
;SET BY "PAT" BIT IN RHCS2  
;RHER1 WAS READ TO SET  
;MCPE AND SC IN RHCS1  
;ON WRITING "1" INTO TRE  
;THEN SETTING "CLR" IN RHCS1  
;RHBAE SHOULD HAVE 0  
;BUT CONTAINED WHAT IS  
;GIVEN IN BAD RHBAE

;GET GOOD = 0  
;READ RHW C FOR COMPARISON  
;COMPARE EXPECTED  
;DATA WITH DATA READ FROM  
;RHW C  
;BRANCH IF GOOD

;AFTER CLEARING THE RH AND  
;DEVICE REGISTERS BY AN  
;RH CLEAR  
;RHER1 WAS CHECKED TO HAVE  
;ZEROS  
;WRONG PARITY CHECKING WAS  
;SET BY "PAT" BIT IN RHCS2  
;RHER1 WAS READ TO SET  
;MCPE AND SC IN RHCS1  
;ON WRITING "1" INTO TRE  
;THEN SETTING "CLR" IN RHCS1  
;RHW C SHOULD HAVE 0  
;BUT CONTAINED WHAT IS  
;GIVEN IN BAD RHW C



:GIVEN IN BAD RHCS1

7235  
 7236 022220 67\$:

7237  
 7238 022220 012737 000120 001124  
 7239 022226 053737 005010 001124  
 7240

;CHECK THAT RHCS2 HAS IR!PAT  
 MOV #IR!PAT,\$GDDAT ;GET GOOD = 120  
 BIS UNIT,\$GDDAT ;INCLUDE UNIT NUMBER

7241 022234 017737 161630 001126  
 7242 022242 023737 001124 001126  
 7243

MOV @RHCS2,\$BDDAT ;READ RHCS2 FOR COMPARISON  
 CMP \$GDDAT,\$BDDAT ;COMPARE EXPECTED  
 ;DATA WITH DATA READ FROM

7244  
 7245 022250 001401  
 7246 022252 104047  
 7247

BEG 69\$ ;RHCS2  
 ERROR 47 ;BRANCH IF GOOD

;AFTER AN RH CLEAR "1"  
 ;WAS WRITTEN INTO ADDRESS REGISTER  
 ;(FRAME COUNT IN TAPE)  
 ;PAT IN RHCS2 WAS SET  
 ;THEN THE ADDRESS REGISTER  
 ;WAS READ BACK  
 ;RHCS2 SHOULD HAVE IR!PAT  
 ;=120  
 ;TOGETHER WITH UNIT NUMBER  
 ;BUT CONTAINED WHAT IS  
 ;GIVEN IN BAD RHCS2

7258 022254 69\$:

7259  
 7260 022254 012737 000000 001124  
 7261

;CHECK THAT RHCS3 HAS 0  
 MOV #0,\$GDDAT ;GET GOOD = 0

7262 022262 017737 161644 001126  
 7263 022270 023737 001124 001126  
 7264

MOV @RHCS3,\$BDDAT ;READ RHCS3 FOR COMPARISON  
 CMP \$GDDAT,\$BDDAT ;COMPARE EXPECTED  
 ;DATA WITH DATA READ FROM

7265  
 7266 022276 001401  
 7267 022300 104050  
 7268

BEG 71\$ ;RHCS3  
 ERROR 50 ;BRANCH IF GOOD

;AFTER AN RH CLEAR "1"  
 ;WAS WRITTEN INTO ADDRESS REGISTER  
 ;(FRAME COUNT IN TAPE)  
 ;PAT IN RHCS2 WAS SET  
 ;THEN THE ADDRESS REGISTER  
 ;WAS READ BACK  
 ;RHCS3 SHOULD HAVE 0  
 ;BUT CONTAINED WHAT IS  
 ;GIVEN IN BAD RHCS3

7277 022302 71\$:

7278  
 7279 022302 012737 000000 001124  
 7280

;CHECK THAT RHBA HAS 0  
 MOV #0,\$GDDAT ;GET GOOD = 0

7281 022310 017737 161550 001126  
 7282 022316 023737 001124 001126  
 7283

MOV @RHBA,\$BDDAT ;READ RHBA FOR COMPARISON  
 CMP \$GDDAT,\$BDDAT ;COMPARE EXPECTED  
 ;DATA WITH DATA READ FROM

7284  
 7285 022324 001401  
 7286 022326 104051  
 7287

BEG 73\$ ;RHBA  
 ERROR 51 ;BRANCH IF GOOD

;AFTER AN RH CLEAR "1"  
 ;WAS WRITTEN INTO ADDRESS REGISTER  
 ;(FRAME COUNT IN TAPE)  
 ;PAT IN RHCS2 WAS SET

7288  
 7289  
 7290

; THEN THE ADDRESS REGISTER  
; WAS READ BACK  
; RHBA SHOULD HAVE 0  
; BUT CONTAINED WHAT IS  
; GIVEN IN BAD RHBA

7291  
7292  
7293  
7294  
7295  
7296 022330 73\$:

; CHECK THAT RHBAE HAS 0  
MOV #0, \$GDDAT

; GET GOOD = 0

7297  
7298 022330 012737 000000 001124  
7299  
7300 022336 017737 161566 001126  
7301 022344 023737 001124 001126  
7302  
7303

MOV @RHBAE, \$BDDAT  
CMP \$GDDAT, \$BDDAT

; READ RHBAE FOR COMPARISON  
; COMPARE EXPECTED  
; DATA WITH DATA READ FROM  
; RHBAE  
; BRANCH IF GOOD

7304 022352 001401  
7305 022354 104052

BEQ 75\$  
ERROR 52

; AFTER AN RH CLEAR "1"  
; WAS WRITTEN INTO ADDRESS REGISTER  
; (FRAME COUNT IN TAPE)  
; PAT IN RHCS2 WAS SET  
; THEN THE ADDRESS REGISTER  
; WAS READ BACK  
; RHBAE SHOULD HAVE 0  
; BUT CONTAINED WHAT IS  
; GIVEN IN BAD RHBAE

7306  
7307  
7308  
7309  
7310  
7311  
7312  
7313  
7314

7315 022356 75\$:  
7316  
7317 022356 012737 000000 001124

; CHECK THAT RHCW HAS 0  
MOV #0, \$GDDAT

; GET GOOD = 0

7318  
7319 022364 017737 161472 001126  
7320 022372 023737 001124 001126  
7321  
7322

MOV @RHCW, \$BDDAT  
CMP \$GDDAT, \$BDDAT

; READ RHCW FOR COMPARISON  
; COMPARE EXPECTED  
; DATA WITH DATA READ FROM  
; RHCW  
; BRANCH IF GOOD

7323 022400 001401  
7324 022402 104053

BEQ 77\$  
ERROR 53

; AFTER AN RH CLEAR "1"  
; WAS WRITTEN INTO ADDRESS REGISTER  
; (FRAME COUNT IN TAPE)  
; PAT IN RHCS2 WAS SET  
; THEN THE ADDRESS REGISTER  
; WAS READ BACK  
; RHCW SHOULD HAVE 0  
; BUT CONTAINED WHAT IS  
; GIVEN IN BAD RHCW

7325  
7326  
7327  
7328  
7329  
7330  
7331  
7332  
7333

7334 022404 77\$:  
7335  
7336  
7337

::\*\*\*\*\*  
:\*TEST 23 TEST DUPLICATED A16 (RHCS1 BIT #8)

7338  
7339  
7340  
7341  
7342  
7343  
7344  
7345  
7346

:\* CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5.  
:\* MOVE 400 (ONE INTO A16) IN RHCS1  
:\* READ RHCS1 TO CONTAIN 600 (BIT #8 AND RDY)  
:\* READ RHBAE TO CONTAIN "1" (BIT #0 HIGH)  
:\*  
:\* MOVE 0 INTO RHBAE (WRITING 0 INTO RHBAE BIT #0)



```

7403 022506          67$:
7404
7405
7406 022506 005077 161416      ;NOW MOVE 0 INTO RHBAE (WRITING 0 INTO RHBAE BIT #0)
7407          CLR          @RHBAE          ;WRITE ZERO INTO RHBAE
7408
7409          ;CHECK THAT RHBAE HAS 0
7410 022512 012737 000000 001124      MOV          #0,$GDDAT          ;GET GOOD = 0
7411
7412 022520 017737 161404 001126      MOV          @RHBAE,$BDDAT      ;READ RHBAE FOR COMPARISON
7413 022526 023737 001124 001126      CMP          $GDDAT,$BDDAT      ;COMPARE EXPECTED
7414          ;DATA WITH DATA READ FROM
7415          ;RHBAE
7416 022534 001401          BEQ          69$                ;BRANCH IF GOOD
7417 022536 104056          ERROR          56
7418          ;AFTER SETTING "CLR" BIT #5
7419          ;IN RHCS2 TO INIT THE RH
7420          ;A16 WAS WRITTEN INTO RHCS1
7421          ;RHCS1 WAS READ
7422          ;RHBAE WAS READ
7423          ;THEN ZERO WAS WRITTEN
7424          ;INTO RHBAE
7425          ;ON READING RHBAE
7426          ;RHBAE SHOULD HAVE 0
7427          ;BUT CONTAINED WHAT IS
7428          ;GIVEN IN BAD RHBAE
7429 022540          69$:
7430
7431
7432          ;CHECK THAT RHCS1 HAS DVA!RDY
7433 022540 012737 004200 001124      MOV          #DVA!RDY,$GDDAT    ;GET GOOD = 4200
7434
7435 022546 017737 161306 001126      MOV          @RHCS1,$BDDAT      ;READ RHCS1 FOR COMPARISON
7436 022554 023737 001124 001126      CMP          $GDDAT,$BDDAT      ;COMPARE EXPECTED
7437          ;DATA WITH DATA READ FROM
7438          ;RHCS1
7439 022562 001401          BEQ          71$                ;BRANCH IF GOOD
7440 022564 104057          ERROR          57
7441          ;AFTER SETTING "CLR" BIT #5
7442          ;IN RHCS2 TO INIT THE RH
7443          ;A16 WAS WRITTEN INTO RHCS1
7444          ;RHCS1 WAS READ
7445          ;RHBAE WAS READ
7446          ;THEN ZERO WAS WRITTEN
7447          ;INTO RHBAE
7448          ;RHBAE WAS READ
7449          ;ON READING RHCS1
7450          ;RHCS1 SHOULD HAVE DVA!RDY
7451          ;=4200
7452          ;BUT CONTAINED WHAT IS
7453          ;GIVEN IN BAD RHCS1
7454 022566          71$:
7455
7456
7457          ;*****
7458          ;*TEST 24          TEST DUPLICATED A17 (RHCS1 BIT #9)

```

```

7459
7460
7461
7462
7463
7464
7465
7466
7467
7468
7469
7470
7471
7472 022566 000004
7473 022570 012706 001000
7474 022574 012737 000024 004656
7475
7476 022602 004737 04C322
7477
7478
7479
7480
7481 022606 012777 001000 161244
7482
7483
7484
7485
7486 022614 012737 005200 001124
7487
7488 022622 017737 161232 001126
7489 022630 023737 001124 001126
7490
7491
7492 022636 001401
7493 022640 104060
7494
7495
7496
7497
7498
7499
7500
7501
7502 022642
7503
7504
7505
7506 022642 012737 000002 001124
7507
7508 022650 017737 161254 001126
7509 022656 023737 001124 001126
7510
7511
7512 022664 001401
7513 022666 104061
7514

```

```

;* CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
;* MOVE 400 (ONE INTO A17) IN RHCS1
;* READ RHCS1 TO CONTAIN 1200 (BIT #9 AND RDY)
;* READ RHBAE TO CONTAIN "2" (BIT #1 HIGH)
;*
;* MOVE 0 INTO RHBAE (WRITING 0 INTO RHBAE BIT #1)
;* READ RHBAE TO CONTAIN 2
;* READ RHCS1 TO CONTAIN ONLY RDY (BIT #9 IN RHCS1 IS ZERO)
;*
*****
↑ST24: SCOPE
MOV #STACK, SP ;RESET STACK
MOV #24, @#↑STNM ;SAVE TEST NUMBER
JSR PC, @#CLDISK ;GIVE RH INITIALIZE
;SETUP UNIT NUBER
;CLEAR RHC AND FUNCTION BITS IN RHCS1
;WRITE "1" INTO A17 IN RHCS1
MOV #A17, @RHCS1 ;MOVE 1200 INTO RHCS1
;CHECK THAT RHCS1 HAS A17!DVA!RDY
MOV #A17!DVA!RDY, $GDDAT ;GET GOOD = 4600
MOV @RHCS1, $BDDAT ;READ RHCS1 FOR COMPARISON
CMP $GDDAT, $BDDAT ;COMPARE EXPECTED
;DATA WITH DATA READ FROM
;RHCS1
;BRANCH IF GOOD
BEQ 65$
ERROR 60
;AFTER SETTING "CLR" BIT #5
;IN RHCS2 TO INIT THE RH
;A17 WAS WRITTEN INTO RHCS1
;RHCS1 WAS READ
;RHCS1 SHOULD HAVE A17!DVA!RDY
;=4600
;BUT CONTAINED WHAT IS
;GIVEN IN BAD RHCS1
65$:
;CHECK THAT RHBAE HAS BIT1
MOV #BIT1, $GDDAT ;GET GOOD = 2
MOV @RHBAE, $BDDAT ;READ RHBAE FOR COMPARISON
CMP $GDDAT, $BDDAT ;COMPARE EXPECTED
;DATA WITH DATA READ FROM
;RHBAE
;BRANCH IF GOOD
BEQ 67$
ERROR 61
;AFTER SETTING "CLR" BIT #5

```





```

7571
7572
7573
7574 022750
7575
7576
7577
7578
7579
7580
7581
7582
7583
7584
7585
7586
7587
7588
7589
7590
7591
7592 022750 000004
7593 022752 012706 001000
7594 022756 012737 000025 004656
7595
7596 022764 004737 040322
7597
7598
7599
7600
7601 022770 012777 000100 161062
7602
7603
7604
7605
7606 022776 012737 004300 001124
7607
7608 023004 017737 161050 001126
7609 023012 023737 001124 001126
7610
7611
7612 023020 001401
7613 023022 104064
7614
7615
7616
7617
7618
7619
7620
7621
7622 023024
7623
7624
7625
7626 023024 012737 000100 001124

```

```

;=4200
;BUT CONTAINED WHAT IS
;GIVEN IN BAD RHCS1

71$:
;*****
;*TEST 25 TEST DUPLICATED IE (RHCS1 BIT #6)
;
;* CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
;* MOVE 100 (ONE INTO IE) IN RHCS1
;* READ RHCS1 TO CONTAIN 300 (BIT #6 AND RDY)
;* READ RHCS3 TO CONTAIN "100" (BIT #6 HIGH)
;
;* MOVE 0 INTO RHCS3 (WRITING 0 INTO RHCS3 BIT #6)
;* READ RHCS3 TO CONTAIN 100
;* READ RHCS1 TO CONTAIN ONLY RDY (BIT #6 IN RHCS1 IS ZERO)
;
;*****
†ST25: SCOPE
MOV #STACK,SP ;RESET STACK
MOV #25,#STNM ;SAVE TEST NUMBER
JSR PC,#CLDISK ;GIVE RH INITIALIZE
;SETUP UNIT NUBER
;CLEAR RHWC AND FUNCTION BITS IN RHCS1
;WRITE "1" INTO IE IN RHCS1
MOV #IE,#RHCS1 ;MOVE 100 INTO RHCS1
;CHECK THAT RHCS1 HAS IE!DVA!RDY
MOV #IE!DVA!RDY,$GDDAT ;GET GOOD = 4300
MOV #RHCS1,$BDDAT ;READ RHCS1 FOR COMPARISON
CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
;DATA WITH DATA READ FROM
;RHCS1
BEQ 65$ ;BRANCH IF GOOD
ERROR 64
;AFTER SETTING "CLR" BIT #5
;IN RHCS2 TO INIT THE RH
;IE WAS WRITTEN INTO RHCS1
;RHCS1 WAS READ
;RHCS1 SHOULD HAVE IE!DVA!RDY
;=4300
;BUT CONTAINED WHAT IS
;GIVEN IN BAD RHCS1

65$:
;CHECK THAT RHCS3 HAS IE
MOV #IE,$GDDAT ;GET GOOD = 1

```



7683  
7684  
7685  
7686  
7687  
7688  
7689  
7690  
7691  
7692  
7693  
7694  
7695  
7696  
7697  
7698  
7699  
7700  
7701  
7702  
7703  
7704  
7705  
7706  
7707  
7708  
7709  
7710  
7711  
7712  
7713  
7714  
7715  
7716  
7717  
7718  
7719  
7720  
7721  
7722  
7723  
7724  
7725  
7726  
7727  
7728  
7729  
7730  
7731  
7732  
7733  
7734  
7735  
7736  
7737  
7738

023132

715:

; IE WAS WRITTEN INTO RHCSI  
; RHCSI WAS READ  
; RHCS3 WAS READ  
; THEN ZERO WAS WRITTEN  
; INTO RHCS3  
; RHCS3 WAS READ  
; ON READING RHCSI  
; RHCSI SHOULD HAVE DVA!RDY  
; =4200  
; BUT CONTAINED WHAT IS  
; GIVEN IN BAD RHCSI

\*\*\*\*\*  
; \*TEST 26 RHCSI PROGRAM ERROR (PGE BIT #10)

; \* CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5,  
; \* SET UP FOR A 10 WORD WRITE  
; \* SET "GO" (RHCSI BIT #0) TWICE IN TWO SUCCESSIVE  
; \* INSTRUCTIONS  
; \* THIS SHOULD SET, SC AND TRE IN RHCSI  
; \* AND PGE SHOULD BE SET IN RHCS2  
; \* RHCS3, ARE CHECKED  
; \* THE NUMBER OF WORDS ARE LARGE ENOUGH SO THAT AFTER  
; \* ONE "BIS" INSTRUCTION TO SET "GO" IN RHCSI  
; \* THERE IS SUFFICIENT TIME TO GIVE ANOTHER "BIS" INSTRUCTION  
; \* TO SET "GO" BEFORE THE FIRST "GO" HAS TIME TO COMPLETE

\*\*\*\*\*

†ST26: SCOPE  
MOV #STACK.SP ; RESET STACK  
MOV #26, @†STNM ; SAVE TEST NUMBER  
JSR PC, @CLDISK ; GIVE RH INITIALIZE  
; SETUP UNIT NUBER  
; CLEAR RHWC AND FUNCTION BITS IN RHCS1

; SET UP RH FOR A 10 WORD WRITE  
MOV #-10, @RHWC ; WORD COUNT REGISTER=10  
MOV #WRFROM, @RHBA ; BUS ADDRESS REGISTER=WRTIBUF  
; WRITE 10 WORDS FROM "WRFROM" INTO  
; WHAT EVER RH DEVICE IS AVAILABLE  
MOV #-1, NOGO ; DO NOT GIVE GO IN COMMAND ROUTINE  
JSR RO, @COMND ; GO TO DO COMMAND  
WRIDAT ; WRITE DATA  
BIS #GO, @RHCSI ; SET GO  
BIS #GO, @RHCSI ; SET GO WITHOUT TIME TO COMPLET LAST GO

; CHECK THAT RHCSI HAS SC!DVA!TRE!61  
MOV #SC!DVA!TRE!61, \$GDDAT ; GET GOOD = 144000

MOV @RHCSI, \$BDDAT ; READ RHCSI FOR COMPARISON  
CMP \$GDDAT, \$BDDAT ; COMPARE EXPECTED  
; DATA WITH DATA READ FROM

```

7739 ;RHCS1
7740 02324C 001401 BEQ 65$ ;BRANCH IF GOOD
7741 023242 104070 ERROR 70
7742 ;AFTER SETTING "CLR" BIT #5
7743 ;IN RHCS2 TO INIT THE RH
7744 ;AND GIVING TWO SUCCESSIVE
7745 ;"GO" WITHOUT GIVING TIME FOR
7746 ;THE FIRST TO COMPLETE
7747 ;RHCS1 SHOULD HAVE SC!DVA!TRE!S!
7748 ;=144000
7749 ;BUT CONTAINED WHAT IS
7750 ;GIVEN IN BAD RHCS1
7751 023244 65$:
7752 ;CHECK THAT RHCS2 HAS PGE!OR
7753 023244 012737 002200 001124 MOV #PGE!OR,$GDDAT ;GET GOOD = 2000
7754 023252 053737 005010 001124 BIS UNIT,$GDDAT ;INCLUDE UNIT NUMBER
7755
7756 023260 017737 160604 001126 MOV @RHCS2,$BDDAT ;READ RHCS2 FOR COMPARISON
7757 023266 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
7758 ;DATA WITH DATA READ FROM
7759 ;RHCS2
7760 023274 001401 BEQ 67$ ;BRANCH IF GOOD
7761 023276 104071 ERROR 71
7762 ;AFTER SETTING "CLR" BIT #5
7763 ;IN RHCS2 TO INIT THE RH
7764 ;AND GIVING TWO SUCCESSIVE
7765 ;"GO" WITHOUT GIVING TIME FOR
7766 ;THE FIRST TO COMPLETE
7767 ;RHCS2 SHOULD HAVE PGE!OR
7768 ;=2000
7769 ;TOGETHER WITH UNIT NUMBER
7770 ;BUT CONTAINED WHAT IS
7771 ;GIVEN IN BAD RHCS2
7772 023300 67$:
7773 ;CHECK THAT RHCS3 HAS DBL
7774 023300 012737 002000 001124 MOV #DBL,$GDDAT ;GET GOOD = 0
7775
7776 023306 017737 160620 001126 MOV @RHCS3,$BDDAT ;READ RHCS3 FOR COMPARISON
7777 023314 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
7778 ;DATA WITH DATA READ FROM
7779 ;RHCS3
7780 023322 001401 BEQ 69$ ;BRANCH IF GOOD
7781 023324 104072 ERROR 72
7782 ;AFTER SETTING "CLR" BIT #5
7783 ;IN RHCS2 TO INIT THE RH
7784 ;AND GIVING TWO SUCCESSIVE
7785 ;"GO" WITHOUT GIVING TIME FOR
7786 ;THE FIRST TO COMPLETE
7787 ;RHCS3 SHOULD HAVE DBL
7788 ;BUT CONTAINED WHAT IS
7789 ;GIVEN IN BAD RHCS3
7790 023326 69$:
7791
7792
7793 ;*****
7794 ;*TEST 27 RHCS2 - BUS ADDRESS INHIBIT BIT #3

```

```

7795
7796
7797
7798
7799
7800
7801
7802
7803
7804
7805
7806
7807
7808
7809
7810
7811
7812
7813
7814
7815
7816
7817 023326 000004
7818 023330 012706 001000
7819 023334 012737 000027 004656
7820
7821 023342 004737 040322
7822
7823
7824
7825
7826
7827 023346 012777 177766 160506
7828 023354 052777 000010 160506
7829 023362 012777 004210 160474
7830 023370 004077 161244
7831 023374 004154
7832
7833
7834
7835 023376 012737 004260 001124
7836
7837 023404 017737 160450 001126
7838 023412 023737 001124 001126
7839
7840
7841 023420 001401
7842 023422 104076
7843
7844
7845
7846
7847
7848
7849
7850

```

```

: * CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #6)
: * SET UP FOR A 2 WORD WRITE FROM A BUFFER TAGGED "WRFROM"
: * SET BUS ADDRESS INHIBIT RHCS2 BIT #3 "BAI"
: * AT THE END OF WRITE CHECK
: * RHCS1 TO HAVE ONLY RDY AND COMMAND
: * RHCS2 TO HAVE BAI
: * RHCS3 TO HAVE 0
: * RHBA TO HAVE ADDRESS OF "WRFROM"
: * RHBAE AND RHWC TO HAVE ZEROS
: *
: * NOW SET UP READ FOR THE SAME DATA WITH
: * BAI SET TO READ INTO A BUFFER TAGGED "REINTO"
: * AFTER READ CHECK
: * RHCS1 TO HAVE ONLY RDY AND COMMAND
: * RHCS2 TO HAVE BAI
: * RHCS3 TO HAVE 0
: * RHBA TO HAVE ADDRESS OF "REINTO"
: * RHBAE AND RHWC TO HAVE ZEROS
: * DATA IN REINTO BUFFER IS CHECKED WITH DATA IN
: * WRFROM BUFFER
: *
: *****
†ST27: SCOPE
MOV #STACK,SP ;RESET STACK
MOV #27,‡#†STNM ;SAVE TEST NUMBER
JSR PC,‡#CLDISK ;GIVE RH INITIALIZE
;SETUP UNIT NUBER
;CLEAR RHWC AND FUNCTION BITS IN RHCS1
:SET UP 10 WORD WRITE FROM BUFFER TAGGED "WRFROM"
;WITH BAI IN RHCS2
MOV #-10,‡RHWC ;WORD COUNT REGISTER=10
BIS #BAI,‡RHCS2 ;BUS ADDRESS INHIBIT IN RHCS2
MOV #WRFROM,‡RHBA ;BUS ADDRESS REGISTER=WRFROM
JSR RO,‡COMND ;GO TO DO COMMAND
WRIDAT ;WRITE DATA
:CHECK THAT RHCS1 HAS DVA!RDY!60
MOV #DVA!RDY!60,$GDDAT ;GET GOOD = 4252
MOV ‡RHCS1,$BDDAT ;READ RHCS1 FOR COMPARISON
CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
;DATA WITH DATA READ FROM
;RHCS1
;BRANCH IF GOOD
BEQ 65$
ERROR 76
: AFTER SETTING "CLR" BIT #5
: IN RHCS2 TO INIT THE RH
: A 2 WORD WRITE WAS DONE
: WITH BAI BIT IN RHCS2 SET
: AT END OF WRITE
: RHCS1 SHOULD HAVE DVA!RDY!60
: =4252
: BUT CONTAINED WHAT IS

```



```

7907                                     :BUT CONTAINED WHAT IS
7908                                     :GIVEN IN BAD RHBA
7909 023534                               715:
7910                                     :CHECK THAT RHBAE HAS 0
7911 023534 012737 000000 001124         MOV    #0,$GDDAT    :GET GOOD = 0
7912
7913 023542 017737 160362 001126         MOV    @RHBAE,$BDDAT :READ RHBAE FOR COMPARISON
7914 023550 023737 001124 001126         CMP    $GDDAT,$BDDAT :COMPARE EXPECTED
7915                                     :DATA WITH DATA READ FROM
7916                                     :RHBAE
7917 023556 001401                         BEQ    73$          :BRANCH IF GOOD
7918 023560 104102                         ERROR  102
7919
7920                                     :AFTER SETTING "CLR" BIT #5
7921                                     :IN RHCS2 TO INIT THE RH
7922                                     :A 2 WORD WRITE WAS DONE
7923                                     :WITH BAI BIT IN RHCS2 SET
7924                                     :AT END OF WRITE
7925                                     :RHBAE SHOULD HAVE 0
7926                                     :BUT CONTAINED WHAT IS
7927 023562                               73$:
7928                                     :CHECK THAT RHWC HAS 0
7929 023562 012737 000000 001124         MOV    #0,$GDDAT    :GET GOOD = 0
7930
7931 023570 017737 160266 001126         MOV    @RHWC,$BDDAT :READ RHWC FOR COMPARISON
7932 023576 023737 001124 001126         CMP    $GDDAT,$BDDAT :COMPARE EXPECTED
7933                                     :DATA WITH DATA READ FROM
7934                                     :RHWC
7935 023604 001401                         BEQ    75$          :BRANCH IF GOOD
7936 023606 104103                         ERROR  103
7937
7938                                     :AFTER SETTING "CLR" BIT #5
7939                                     :IN RHCS2 TO INIT THE RH
7940                                     :A 2 WORD WRITE WAS DONE
7941                                     :WITH BAI BIT IN RHCS2 SET
7942                                     :AT END OF WRITE
7943                                     :RHWC SHOULD HAVE 0
7944                                     :BUT CONTAINED WHAT IS
7945 023610                               75$:
7946                                     :GIVEN IN BAD RHWC
7947
7948
7949
7950 ::*****
7951 :*TEST 30      RHCS2 MDPE BIT #8 AND RHCS3 IPCKO BIT #0
7952 :*          CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
7953 :*          SET IPCKO (RHCS3 BIT #0)
7954 :*          MOVE ALL ZEROS INTO RHDB ONCE
7955 :*          THIS SHOULD SET TRE SC IN RHCS1
7956 :*          READ RHDB ONCE THIS SHOULD SET MDPE (RHCS2 BIT #9)
7957 :*          CHECK RHCS1 FOR RDY (BIT #7), TRE (BIT #14), SC (BIT #15),
7958 :*          "CLR" (RHCS2 BIT #5) IS GIVEN
7959 :*          ALL ERRORS ARE CLEARED
7960 :*          CHECK RHCS1, RHCS2, RHCS3, RHBA, RHBAE, RHWC
7961 ::*****
7962 023610 000004                         :S*3C. SCOPE

```



MAINDEC-11-DEPHAC-A  
DEPHAC.P11 T30

MACY11 27(732) 09-SEP-76 07:55 PAGE 151  
RHSC2 MDPE BIT #8 AND RHCS3 IPCKO BIT #0

```

7963 023612 012706 001000      MOV      #STACK,SP      ;RESET STACK
7964 023616 012737 000030 004656      MOV      #30,#STNM      ;SAVE TEST NUMBER
7965
7966 023624 004737 040322      JSR      PC,#CLDISK     ;GIVE RH INITIALIZE
7967                                ;SETUP UNIT NUBER
7968                                ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
7969 023630 012737 000000 004662      MOV      #0,IP          ;IPCK BIT NO. FOR PRINTOUT
7970
7971 023636 052777 000001 160266      BIS      #IPCKO,#RHCS3 ;SET IPCKO IN RHCS3-BIT #0
7972 023644 005037 004210      CLR      WRFROM         ;WRITE ZERO INTO LOCATION
7973 023650 013777 004210 160224      MOV      WRFROM,#RHDB   ;WRITE ZEROS INTO RHDB ONCE
7974
7975                                ;CHECK THAT RHCS1 HAS SC!TRE!DVA!RDY
7976 023656 012737 144200 001124      MOV      #SC!TRE!DVA!RDY,$GDDAT ;GET GOOD = 144200
7977
7978 023664 017737 160170 001126      MOV      #RHCS1,$BDDAT  ;READ RHCS1 FOR COMPARISON
7979 023672 023737 001124 001126      CMP      $GDDAT,$BDDAT  ;COMPARE EXPECTED
7980                                ;DATA WITH DATA READ FROM
7981                                ;RHCS1
7982 023700 001401      BEQ      65$           ;BRANCH IF GOOD
7983 023702 104140      ERROR   140
7984                                ;AFTER SETTING "CLR" BIT #5
7985                                ;IN RHCS2 TO INIT THE RH
7986                                ;IPCKO BIT #0 RHCS3 WAS SET
7987                                ;ZEROS WERE MOVED INTO RHDB
7988                                ;RHCS1 SHOULD HAVE SC!TRE!DVA!RDY
7989                                ;=144200
7990                                ;BUT CONTAINED WHAT IS
7991                                ;GIVEN IN BAD RHCS1
7992 023704      65$:
7993 023704 017737 160172 001200      MOV      #RHDB,$TMP1   ;READ RHDB ONCE
7994
7995
7996                                ;CHECK THAT RHCS3 HAS 1
7997 023712 012737 000001 001124      MOV      #1,$GDDAT     ;GET GOOD = 1
7998
7999 023720 017737 160206 001126      MOV      #RHCS3,$BDDAT ;READ RHCS3 FOR COMPARISON
8000 023726 023737 001124 001126      CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED
8001                                ;DATA WITH DATA READ FROM
8002                                ;RHCS3
8003 023734 001401      BEQ      67$           ;BRANCH IF GOOD
8004 023736 104104      ERROR   104
8005                                ;AFTER SETTING "CLR" BIT #5
8006                                ;IN RHCS2 TO INIT THE RH
8007                                ;IPCKO BIT #0 RHCS3 WAS SET
8008                                ;ZEROS WERE MOVED INTO
8009                                ;RHDB
8010                                ;RHDB WAS READ THEN
8011                                ;RHCS3 SHOULD HAVE 1
8012                                ;=1
8013                                ;BUT CONTAINED WHAT IS
8014                                ;GIVEN IN BAD RHCS3
8015 023740      67$:
8016
8017                                ;CHECK THAT RHCS2 HAS MDPE!IR
8018 023740 012737 000500 001124      MOV      #MDPE!IR,$GDDAT ;GET GOOD = 500

```

```

8019 023746 053737 005010 001124      BIS      UNIT,$GDDAT      ;INCLUDE UNIT NUMBER
8020
8021 023754 017737 160110 001126      MOV      @RHCS2,$BDDAT   ;READ RHCS2 FOR COMPARISON
8022 023762 023737 001124 001126      CMP      $GDDAT,$BDDAT   ;COMPARE EXPECTED
8023                                     ;DATA WITH DATA READ FROM
8024                                     ;RHCS2
8025 023770 001401      BEQ      69$             ;BRANCH IF GOOD
8026 023772 104105      ERROR   105
8027                                     ;AFTER SETTING "CLR" BIT #5
8028                                     ;IN RHCS2 TO INIT THE RH
8029                                     ;IPCKO BIT #0 RHCS3 WAS SET
8030                                     ;ZEROS WERE MOVED INTO
8031                                     ;RHDB
8032                                     ;RHDB WAS READ THEN
8033                                     ;RHCS2 SHOULD HAVE MDPE!IR
8034                                     ;=500
8035                                     ;TOGETHER WITH UNIT NUMBER
8036                                     ;BUT CONTAINED WHAT IS
8037                                     ;GIVEN IN BAD RHCS2
8038 023774          69$:
8039
8040
8041 023774 004737 040322      JSR      PC,@#CLDISK    ;GIVE RH INITIALIZE
8042                                     ;SETUP UNIT NUBER
8043                                     ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
8044
8045
8046                                     ;CHECK THAT RHCS1 HAS RDY!DVA
8047 024000 012737 004200 001124      MOV      #RDY!DVA,$GDDAT ;GET GOOD = 4200
8048
8049 024006 017737 160046 001126      MOV      @RHCS1,$BDDAT   ;READ RHCS1 FOR COMPARISON
8050 024014 023737 001124 001126      CMP      $GDDAT,$BDDAT   ;COMPARE EXPECTED
8051                                     ;DATA WITH DATA READ FRJM
8052                                     ;RHCS1
8053 024022 001401      BEQ      71$             ;BRANCH IF GOOD
8054 024024 104106      ERROR   106
8055                                     ;AFTER SETTING "CLR" BIT #5
8056                                     ;IN RHCS2 TO INIT THE RH
8057                                     ;IPCKO BIT #0 RHCS3 WAS SET
8058                                     ;ZEROS WERE MOVED INTO
8059                                     ;RHDB
8060                                     ;RHDB WAS READ THEN
8061                                     ;AN RH CLEAR (RHCS2 BIT #5)
8062                                     ;WAS GIVEN THIS SHOULD
8063                                     ;CLEAR ALL ERRORS
8064                                     ;RHCS1 SHOULD HAVE RDY!DVA
8065                                     ;=4200
8066                                     ;BUT CONTAINED WHAT IS
8067                                     ;GIVEN IN BAD RHCS1
8068 024026          71$:
8069                                     ;CHECK THAT RHCS2 HAS IR
8070 024026 012737 000100 001124      MOV      #IR,$GDDAT     ;GET GOOD = 100
8071 024034 053737 005010 001124      BIS      UNIT,$GDDAT     ;INCLUDE UNIT NUMBER
8072
8073 024042 017737 160022 001126      MOV      @RHCS2,$BDDAT   ;READ RHCS2 FOR COMPARISON
8074 024050 023737 001124 001126      CMP      $GDDAT,$BDDAT   ;COMPARE EXPECTED

```



```

8131                                     ; AN RH CLEAR (RHCS2 BIT #5)
8132                                     ; WAS GIVEN THIS SHOULD
8133                                     ; CLEAR ALL ERRORS
8134                                     ; RHBA SHOULD HAVE 0
8135                                     ; BUT CONTAINED WHAT IS
8136                                     ; GIVEN IN BAD RHBA
8137 024136                               77$:
8138                                     ; CHECK THAT RHBAE HAS 0
8139 024136 012737 000000 001124          MOV    #0,$GDDAT    ; GET GOOD = 0
8140
8141 024144 017737 157760 001126          MOV    @RHBAE,$BDDAT ; READ RHBAE FOR COMPARISON
8142 024152 023737 001124 001126          CMP    $GDDAT,$BDDAT ; COMPARE EXPECTED
8143                                     ; DATA WITH DATA READ FROM
8144                                     ; RHBAE
8145 024160 001401                          BEQ    79$          ; BRANCH IF GOOD
8146 024162 104112
8147
8148                                     ; AFTER SETTING "CLR" BIT #5
8149                                     ; IN RHCS2 TO INIT THE RH
8150                                     ; IPCKO BIT #0 RHCS3 WAS SET
8151                                     ; ZEROS WERE MOVED INTO
8152                                     ; RHDB
8153                                     ; RHDB WAS READ THEN
8154                                     ; AN RH CLEAR (RHCS2 BIT #5)
8155                                     ; WAS GIVEN THIS SHOULD
8156                                     ; CLEAR ALL ERRORS
8157                                     ; RHBAE SHOULD HAVE 0
8158                                     ; BUT CONTAINED WHAT IS
8159                                     ; GIVEN IN BAD RHBAE
8159 024164                               79$:
8160                                     ; CHECK THAT RHWC HAS 0
8161 024164 012737 000000 001124          MOV    #0,$GDDAT    ; GET GOOD = 0
8162
8163 024172 017737 157664 001126          MOV    @RHWC,$BDDAT ; READ RHWC FOR COMPARISON
8164 024200 023737 001124 001126          CMP    $GDDAT,$BDDAT ; COMPARE EXPECTED
8165                                     ; DATA WITH DATA READ FROM
8166                                     ; RHWC
8167 024206 001401                          BEQ    81$          ; BRANCH IF GOOD
8168 024210 104113
8169
8170                                     ; AFTER SETTING "CLR" BIT #5
8171                                     ; IN RHCS2 TO INIT THE RH
8172                                     ; IPCKO BIT #0 RHCS3 WAS SET
8173                                     ; ZEROS WERE MOVED INTO
8174                                     ; RHDB
8175                                     ; RHDB WAS READ THEN
8176                                     ; AN RH CLEAR (RHCS2 BIT #5)
8177                                     ; WAS GIVEN THIS SHOULD
8178                                     ; CLEAR ALL ERRORS
8179                                     ; RHWC SHOULD HAVE 0
8180                                     ; BUT CONTAINED WHAT IS
8181                                     ; GIVEN IN BAD RHWC
8181 024212                               81$:
8182
8183
8184
8185
8186

```

\*\*\*\*\*  
; \*TEST 31 RHSC2 MDPE BIT #8 AND RHCS3 IPCK1 BIT #1

```

8187      :*      CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
8188      :*      SET IPCK1 (RHCS3 BIT #1)
8189      :*      MOVE ALL ZEROS INTO RHDB ONCE
8190      :*      THIS SHOULD SET TRE SC IN RHCS1
8191      :*      READ RHDB ONCE THIS SHOULD SET MDPE (RHCS2 BIT #8)
8192      :*      CHECK RHCS1 FOR RDY (BIT #7), TRE (BIT #14), SC (BIT #15)
8193      :*      "CLR" (RHCS2 BIT #5) IS GIVEN
8194      :*      ALL ERRORS ARE CLEARED
8195      :*      CHECK RHCS1, RHCS2, RHCS3, RHBA, RHBAE, RHW
8196
8197      :*****
8198      †ST31: SCOPE
8199      024212 000004      MOV      #STACK,SP      ;RESET STACK
8200      024214 012706 001000 004656      MOV      #31,‡#†STNM      ;SAVE TEST NUMBER
8201
8202      024226 004737 040322      JSR      PC,‡#CLDISK      ;GIVE RH INITIALIZE
8203      ;SETUP UNIT NUBER
8204      ;CLEAR RHW AND FUNCTION BITS IN RHCS1
8205      024232 012737 000001 004662      MOV      #1,IP      ;IPCK BIT NO. FOR PRINTOUT
8206
8207      024240 052777 000002 157664      BIS      #IPCK1,‡RHCS3 ;SET IPCK1 IN RHCS3-BIT #1
8208      024246 005037 004210      CLR      WFROM      ;WRITE ZERO INTO LOCATION
8209      024252 013777 004210 157622      MOV      WFROM,‡RHDB      ;WRITE ZEROS INTO RHDB ONCE
8210
8211      ;CHECK THAT RHCS1 HAS SC!TRE!DVA!RDY
8212      024260 012737 144200 001124      MOV      #SC!TRE!DVA!RDY,$GDDAT ;GET GOOD = 144200
8213
8214      024266 017737 157566 001126      MOV      ‡RHCS1,$BDDAT ;READ RHCS1 FOR COMPARISON
8215      024274 023737 001124 001126      CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED
8216      ;DATA WITH DATA READ FROM
8217      ;RHCS1
8218      024302 001401      BEQ      65$ ;BRANCH IF GOOD
8219      024304 104140      ERROR    140
8220
8221      ;AFTER SETTING "CLR" BIT #5
8222      ;IN RHCS2 TO INIT THE RH
8223      ;IPCK1 BIT #1 RHCS3 WAS SET
8224      ;ZEROS WERE MOVED INTO RHDB
8225      ;RHCS1 SHOULD HAVE SC!TRE!DVA!RDY
8226      ;=144200
8227      ;BUT CONTAINED WHAT IS
8228      ;GIVEN IN BAD RHCS1
8229      024306 017737 157570 001200      65$: MOV      ‡RHDB,$TMP1 ;READ RHDB ONCE
8230
8231
8232      ;CHECK THAT RHCS3 HAS 2
8233      024314 012737 000002 001124      MOV      #2,$GDDAT ;GET GOOD = 2
8234
8235      024322 017737 157604 001126      MOV      ‡RHCS3,$BDDAT ;READ RHCS3 FOR COMPARISON
8236      024330 023737 001124 001126      CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED
8237      ;DATA WITH DATA READ FROM
8238      ;RHCS3
8239      024336 001401      BEQ      67$ ;BRANCH IF GOOD
8240      024340 104104      ERROR    104
8241
8242      ;AFTER SETTING "CLR" BIT #5
      ;IN RHCS2 TO INIT THE RH

```

```

8243 ;IPCK1 BIT #1 RHCS3 WAS SET
8244 ;ZEROS WERE MOVED INTO
8245 ;RHDB
8246 ;RHDB WAS READ THEN
8247 ;RHCS3 SHOULD HAVE 2
8248 ;=2
8249 ;BUT CONTAINED WHAT IS
8250 ;GIVEN IN BAD RHCS3
8251 024342 67$:
8252
8253 ;CHECK THAT RHCS2 HAS MDPE!IR
8254 024342 012737 000500 001124 MOV #MDPE!IR,$GDDAT ;GET GOOD = 500
8255 024350 053737 005010 001124 BIS UNIT,$GDDAT ;INCLUDE UNIT NUMBER
8256
8257 024356 017737 157506 001126 MOV @RHCS2,$BDDAT ;READ RHCS2 FOR COMPARISON
8258 024364 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
8259 ;DATA WITH DATA READ FROM
8260 ;RHCS2
8261 024372 001401 BEQ 69$ ;BRANCH IF GOOD
8262 024374 104105 ERROR 105
8263 ;AFTER SETTING "CLR" BIT #5
8264 ;IN RHCS2 TO INIT THE RH
8265 ;IPCK1 BIT #1 RHCS3 WAS SET
8266 ;ZEROS WERE MOVED INTO
8267 ;RHDB
8268 ;RHDB WAS READ THEN
8269 ;RHCS2 SHOULD HAVE MDPE!IR
8270 ;=500
8271 ;TOGETHER WITH UNIT NUMBER
8272 ;BUT CONTAINED WHAT IS
8273 ;GIVEN IN BAD RHCS2
8274 024376 69$:
8275
8276
8277 024376 004737 040322 JSR PC,@#CLDTRK ;GIVE RH INITIALIZE
8278 ;SETUP UNIT NUBER
8279 ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
8280
8281
8282 ;CHECK THAT RHCS1 HAS RDY!DVA
8283 024402 012737 004200 001124 MOV #RDY!DVA,$GDDAT ;GET GOOD = 4200
8284
8285 024410 017737 157444 001126 MOV @RHCS1,$BDDAT ;READ RHCS1 FOR COMPARISON
8286 024416 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
8287 ;DATA WITH DATA READ FROM
8288 ;RHCS1
8289 024424 001401 BEQ 71$ ;BRANCH IF GOOD
8290 024426 104106 ERROR 106
8291 ;AFTER SETTING "CLR" BIT #5
8292 ;IN RHCS2 TO INIT THE RH
8293 ;IPCK1 BIT #1 RHCS3 WAS SET
8294 ;ZEROS WERE MOVED INTO
8295 ;RHDB
8296 ;RHDB WAS READ THEN
8297 ;AN RH CLEAR (RHCS2 BIT #5)
8298 ;WAS GIVEN THIS SHOULD

```

```

8299                                     ;CLEAR ALL ERRORS
8300                                     ;RHCS1 SHOULD HAVE RDY!DVA
8301                                     ;=4200
8302                                     ;BUT CONTAINED WHAT IS
2303                                     ;GIVEN IN BAD RHCS1
8304 024430                               71$:
8305                                     ;CHECK THAT RHCS2 HAS IR
8306 024430 012737 000100 001124        MOV    #IR,$GDDAT    ;GET GOOD = 100
8307 024436 053737 005010 001124        BIS    UNIT,$GDDAT  ;INCLUDE UNIT NUMBER
8308
8309 024444 017737 157420 001126        MOV    @RHCS2,$BDDAT ;READ RHCS2 FOR COMPARISON
8310 024452 023737 001124 001126        CMP    $GDDAT,$BDDAT ;COMPARE EXPECTED
8311                                     ;DATA WITH DATA READ FROM
8312                                     ;RHCS2
8313 024460 001401                          BEQ    73$           ;BRANCH IF GOOD
8314 024462 104107                          ERROR  107
8315
8316                                     ;AFTER SETTING "CLR" BIT #5
8317                                     ;IN RHCS2 TO INIT THE RH
8318                                     ;IPCK1 BIT #1 RHCS3 WAS SET
8319                                     ;ZEROS WERE MOVED INTO
8320                                     ;RHDB
8321                                     ;RHDB WAS READ THEN
8322                                     ;AN RH CLEAR (RHCS2 BIT #5)
8323                                     ;WAS GIVEN THIS SHOULD
8324                                     ;CLEAR ALL ERRORS
8325                                     ;RHCS2 SHOULD HAVE IR
8326                                     ;=100
8327                                     ;TOGETHER WITH UNIT NUMBER
8328                                     ;BUT CONTAINED WHAT IS
8329 024464                               73$:
8330                                     ;CHECK THAT RHCS3 HAS 0
8331 024464 012737 000000 001124        MOV    #0,$GDDAT   ;GET GOOD = 0
8332
8333 024472 017737 157434 001126        MOV    @RHCS3,$BDDAT ;READ RHCS3 FOR COMPARISON
8334 024500 023737 001124 001126        CMP    $GDDAT,$BDDAT ;COMPARE EXPECTED
8335                                     ;DATA WITH DATA READ FROM
8336                                     ;RHCS3
8337 024506 001401                          BEQ    75$           ;BRANCH IF GOOD
8338 024510 104110                          ERROR  110
8339
8340                                     ;AFTER SETTING "CLR" BIT #5
8341                                     ;IN RHCS2 TO INIT THE RH
8342                                     ;IPCK1 BIT #1 RHCS3 WAS SET
8343                                     ;ZEROS WERE MOVED INTO
8344                                     ;RHDB
8345                                     ;RHDB WAS READ THEN
8346                                     ;AN RH CLEAR (RHCS2 BIT #5)
8347                                     ;WAS GIVEN THIS SHOULD
8348                                     ;CLEAR ALL ERRORS
8349                                     ;RHCS3 SHOULD HAVE 0
8350                                     ;BUT CONTAINED WHAT IS
8351 024512                               75$:
8352                                     ;CHECK THAT RHBA HAS 0
8353 024512 012737 000000 001124        MOV    #0,$GDDAT   ;GET GOOD = 0
8354

```





;AN RH CLEAR (RHCS2 BIT #5)  
;WAS GIVEN THIS SHOULD  
;CLEAR ALL ERRORS  
;RHWC SHOULD HAVE 0  
;BUT CONTAINED WHAT IS  
;GIVEN IN BAD RHWC

8411  
8412  
8413  
8414  
8415  
8416  
8417  
8418  
8419  
8420  
8421  
8422  
8423  
8424  
8425  
8426  
8427  
8428  
8429  
8430  
8431  
8432  
8433  
8434  
8435  
8436  
8437  
8438  
8439  
8440  
8441  
8442  
8443  
8444  
8445  
8446  
8447  
8448  
8449  
8450  
8451  
8452  
8453  
8454  
8455  
8456  
8457  
8458  
8459  
8460  
8461  
8462  
8463  
8464  
8465  
8466

024614

BIS:

```
*****
*TEST 32      RHSC2 MDPE BIT #8 AND RHCS3 IPCK2 BIT #2
*      CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
*      SET IPCK2 (RHCS3 BIT #2)
*      MOVE ALL ZEROS INTO RHDB TWICE
*      THIS SHOULD NOT SET TRE SC IN RHCS1
*      READ RHDB ONCE THIS SHOULD SET MDPE (RHCS2 BIT #8)
*      CHECK RHCS1 FOR RDY (BIT #7), TRE (BIT #14), SC (BIT #15)
*      READ RHDB A SECOND TIME. CHECK RHCS1, RHCS2, RHCS3
*      "CLR" (RHCS2 BIT #5) IS GIVEN
*      ALL ERRORS ARE CLEARED
*      CHECK RHCS1, RHCS2, RHCS3, RHBA, RHBAE, RHWC
*****
```

024614 000004  
024616 012706 001000  
024622 012737 000032 004656  
024630 004737 040322  
024634 012737 000002 004662  
024642 052777 000004 157262  
024650 005037 004210  
024654 013777 004210 157220  
024662 013777 004210 157212  
024670 012737 004200 001124  
024676 017737 157156 001126  
024704 023737 001124 001126  
024712 001401  
024714 104140

```
*****
*ST32: SCOPE
*      MOV      #STACK,SP      ;RESET STACK
*      MOV      #32,@#STNM     ;SAVE TEST NUMBER
*      JSR      PC,@#CLDISK    ;GIVE RH INITIALIZE
*                               ;SETUP UNIT NUBER
*                               ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
*      MOV      #2,IP          ;IPCK BIT NO. FOR PRINTOUT
*      BIS      #IPCK2,@RHCS3 ;SET IPCK2 IN RHCS3-BIT #2
*      CLR      WRFROM         ;WRITE ZERO INTO LOCATION
*      MOV      WRFROM,@RHDB   ;WRITE ZEROS INTO RHDB ONCE
*      MOV      WRFROM,@RHDB   ;WRITE ZERO SECOND TIME
*      ;CHECK THAT RHCS1 HAS DVA!RDY
*      MOV      #DVA!RDY,$GDDAT ;GET GOOD = 4200
*      MOV      @RHCS1,$BDDAT  ;READ RHCS1 FOR COMPARISON
*      CMP      $GDDAT,$BDDAT  ;COMPARE EXPECTED
*                               ;DATA WITH DATA READ FROM
*                               ;RHCS1
*      BEQ      65$           ;BRANCH IF GOOD
*      ERROR   14C
*****
```

;AFTER SETTING "CLR" BIT #5  
;IN RHCS2 TO INIT THE RH  
;IPCK2 BIT #2 RHCS3 WAS SET  
;ZEROS WERE MOVED INTO RHDB  
;TWICE  
;RHCS1 SHOULD HAVE DVA!RDY  
;=4200  
;BUT CONTAINED WHAT IS  
;GIVEN IN BAD RHCS1

MAINDEC-11-DERHAC-A  
DERHAC.P11 T32MACY11 27(732) 09-SEP-76 07:55 PAGE 160  
RHSC2 MDPE BIT #8 AND RHCS3 IPCK2 BIT #2

```

8467 024716          65$:
8468 024716 017737 157160 001200      MOV      @RHDB,$TMP1      ;READ RHDB ONCE
8469
8470
8471          ;CHECK THAT RHCS3 HAS 4
8472 024724 012737 000004 001124      MOV      #4,$GDDAT      ;GET GOOD = 4
8473
8474 024732 017737 157174 001126      MOV      @RHCS3,$BDDAT  ;READ RHCS3 FOR COMPARISON
8475 024740 023737 001124 001126      CMP      $GDDAT,$BDDAT  ;COMPARE EXPECTED
8476          ;DATA WITH DATA READ FROM
8477          ;RHCS3
8478 024746 001401          BEQ      673              ;BRANCH IF GOOD
8479 024750 104104          ERROR    104
8480
8481          ;AFTER SETTING "CLR" BIT #5
8482          ;IN RHCS2 TO INIT THE RH
8483          ;IPCK2 BIT #2 RHCS3 WAS SET
8484          ;ZEROS WERE MOVED INTO
8485          ;RHDB
8486          ;RHDB WAS READ THEN
8487          ;RHCS3 SHOULD HAVE 4
8488          ;=4
8489          ;BUT CONTAINED WHAT IS
8490 024752          67$:
8491
8492          ;CHECK THAT RHCS2 HAS MDPE!OR!IR
8493 024752 012737 000700 001124      MOV      #MDPE!OR!IR,$GDDAT ;GET GOOD = 700
8494 024760 053737 005010 001124      BIS      UNIT,$GDDAT      ;INCLUDE UNIT NUMBER
8495
8496 024766 017737 157076 001126      MOV      @RHCS2,$BDDAT  ;READ RHCS2 FOR COMPARISON
8497 024774 023737 001124 001126      CMP      $GDDAT,$BDDAT  ;COMPARE EXPECTED
8498          ;DATA WITH DATA READ FROM
8499          ;RHCS2
8500 025002 001401          BEQ      69$              ;BRANCH IF GOOD
8501 025004 104105          ERROR    105
8502
8503          ;AFTER SETTING "CLR" BIT #5
8504          ;IN RHCS2 TO INIT THE RH
8505          ;IPCK2 BIT #2 RHCS3 WAS SET
8506          ;ZEROS WERE MOVED INTO
8507          ;RHDB
8508          ;RHDB WAS READ THEN
8509          ;RHCS2 SHOULD HAVE MDPE!OR!IR
8510          ;=700
8511          ;TOGETHER WITH UNIT NUMBER
8512          ;BUT CONTAINED WHAT IS
8513 025006          69$:
8514
8515 025006 017737 157070 001202      MOV      @RHDB,$TMP2      ;READ RHDB SECOND TIME
8516          ;CHECK THAT RHCS1 HAS SC!TRE!DVA!RDY
8517 025014 012737 144200 001124      MOV      #SC!TRE!DVA!RDY,$GDDAT ;GET GOOD = 144200
8518
8519 025022 017737 157032 001126      MOV      @RHCS1,$BDDAT  ;READ RHCS1 FOR COMPARISON
8520 025030 023737 001124 001126      CMP      $GDDAT,$BDDAT  ;COMPARE EXPECTED
8521          ;DATA WITH DATA READ FROM
8522          ;RHCS1

```

MAINDEC-11-DE RHAC-A  
DERHAC.P11 T32

MACY11 27(732) 09-SEP-76 07:55 PAGE 161  
RHSC2 MDPE BIT #8 AND RHCS3 IPCK2 BIT #2

```

8523 025036 001401      BEQ      71$      ;BRANCH IF GOOD
8524 025040 104140      ERROR    140
8525                                     ;AFTER SETTING "CLR" BIT #5
8526                                     ;IN RHCS2 TO INIT THE RH
8527                                     ;IPCK2 BIT #2 RHCS3 WAS SET
8528                                     ;ZEROS WERE MOVED INTO
8529                                     ;RHDB
8530                                     ;RHDB WAS READ THEN
8531                                     ;RHCS1 SHOULD HAVE SC!TRE!DVA!RDY
8532                                     ;=144200
8533                                     ;BUT CONTAINED WHAT IS
8534                                     ;GIVEN IN BAD RHCS1
8535 025042      71$:
8536                                     ;CHECK THAT RHCS2 HAS MDPE!IR
8537 025042 012737 000500 001124      MOV      #MDPE!IR,$GDDAT ;GET GOOD = 500
8538 025050 053737 005010 001124      BIS      UNIT,$GDDAT    ;INCLUDE UNIT NUMBER
8539
8540 025056 017737 157006 001126      MOV      @RHCS2,$BDDAT  ;READ RHCS2 FOR COMPARISON
8541 025064 023737 001124 001126      CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED
8542                                     ;DATA WITH DATA READ FROM
8543                                     ;RHCS2
8544 025072 001401      BEQ      73$      ;BRANCH IF GOOD
8545 025074 104105      ERROR    105
8546                                     ;AFTER SETTING "CLR" BIT #5
8547                                     ;IN RHCS2 TO INIT THE RH
8548                                     ;IPCK2 BIT #2 RHCS3 WAS SET
8549                                     ;ZEROS WERE MOVED INTO
8550                                     ;RHDB
8551                                     ;RHDB WAS READ THEN
8552                                     ;RHCS2 SHOULD HAVE MDPE!IR
8553                                     ;=500
8554                                     ;TOGETHER WITH UNIT NUMBER
8555                                     ;BUT CONTAINED WHAT IS
8556                                     ;GIVEN IN BAD RHCS2
8557 025076      73$:
8558                                     ;CHECK THAT RHCS3 HAS 4
8559 025076 012737 000004 001124      MOV      #4,$GDDAT    ;GET GOOD = 4
8560
8561 025104 017737 157022 001126      MOV      @RHCS3,$BDDAT ;READ RHCS3 FOR COMPARISON
8562 025112 023737 001124 001126      CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED
8563                                     ;DATA WITH DATA READ FROM
8564                                     ;RHCS3
8565 025120 001401      BEQ      75$      ;BRANCH IF GOOD
8566 025122 104104      ERROR    104
8567                                     ;AFTER SETTING "CLR" BIT #5
8568                                     ;IN RHCS2 TO INIT THE RH
8569                                     ;IPCK2 BIT #2 RHCS3 WAS SET
8570                                     ;ZEROS WERE MOVED INTO
8571                                     ;RHDB
8572                                     ;RHDB WAS READ THEN
8573                                     ;RHCS3 SHOULD HAVE 4
8574                                     ;=4
8575                                     ;BUT CONTAINED WHAT IS
8576                                     ;GIVEN IN BAD RHCS3
8577 025124      75$:
8578

```

MAINDEC-11-DERHAC-A  
 DERHAC.P11 T32

MACY11 27(732) 09-SEP-76 07:55 PAGE 162  
 RHSC2 MDPE BIT #8 AND RHCS3 IPCK2 BIT #2

```

9579 025124 004737 040322 JSR PC, @CLDISK ;GIVE RH INITIALIZE
9580 ;SETUP UNIT NUMBER
9581 ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
9582
9583
9584 ;CHECK THAT RHCS1 HAS RDY!DVA
9585 025130 012737 004200 001124 MOV #RDY!DVA, $GDDAT ;GET GOOD = 4200
9586
9587 025136 017737 156716 001126 MOV @RHCS1, $BDDAT ;READ RHCS1 FOR COMPARISON
9588 025144 023737 001124 001126 CMP $GDDAT, $BDDAT ;COMPARE EXPECTED
9589 ;DATA WITH DATA READ FROM
9590 ;RHCS1
9591 025152 001401 BEQ 77$ ;BRANCH IF GOOD
9592 025154 104106 ERROR 106
9593 ;AFTER SETTING "CLR" BIT #5
9594 ;IN RHCS2 TO INIT THE RH
9595 ;IPCK2 BIT #2 RHCS3 WAS SET
9596 ;ZEROS WERE MOVED INTO
9597 ;RHDB
9598 ;RHDB WAS READ THEN
9599 ;AN RH CLEAR (RHCS2 BIT #5)
9600 ;WAS GIVEN THIS SHOULD
9601 ;CLEAR ALL ERRORS
9602 ;RHCS1 SHOULD HAVE RDY!DVA
9603 ;=4200
9604 ;BUT CONTAINED WHAT IS
9605 ;GIVEN IN BAD RHCS1
9606 025156 77$:
9607 ;CHECK THAT RHCS2 HAS IR
9608 025156 012737 000100 001124 MOV #IR, $GDDAT ;GET GOOD = 100
9609 025164 053737 005010 001124 BIS JNIT, $GDDAT ;INCLUDE UNIT NUMBER
9610
9611 025172 017737 156672 001126 MOV @RHCS2, $BDDAT ;READ RHCS2 FOR COMPARISON
9612 025200 023737 001124 001126 CMP $GDDAT, $BDDAT ;COMPARE EXPECTED
9613 ;DATA WITH DATA READ FROM
9614 ;RHCS2
9615 025206 001401 BEQ 79$ ;BRANCH IF GOOD
9616 025210 104107 ERROR 107
9617 ;AFTER SETTING "CLR" BIT #5
9618 ;IN RHCS2 TO INIT THE RH
9619 ;IPCK2 BIT #2 RHCS3 WAS SET
9620 ;ZEROS WERE MOVED INTO
9621 ;RHDB
9622 ;RHDB WAS READ THEN
9623 ;AN RH CLEAR (RHCS2 BIT #5)
9624 ;WAS GIVEN THIS SHOULD
9625 ;CLEAR ALL ERRORS
9626 ;RHCS2 SHOULD HAVE IR
9627 ;=100
9628 ;TOGETHER WITH UNIT NUMBER
9629 ;BUT CONTAINED WHAT IS
9630 ;GIVEN IN BAD RHCS2
9631 025212 79$:
9632 ;CHECK THAT RHCS3 HAS 0
9633 025212 012737 000000 001124 MOV #0, $GDDAT ;GET GOOD = 0
9634
  
```



```

8691                                     ;AN RH CLEAR (RHCS2 BIT #5)
8692                                     ;WAS GIVEN THIS SHOULD
8693                                     ;CLEAR ALL ERRORS
8694                                     ;RHBAE SHOULD HAVE 0
8695                                     ;BUT CONTAINED WHAT IS
8696                                     ;GIVEN IN BAD RHBAE
8697 025314                               85$:
8698                                     ;CHECK THAT RHW C HAS 0
8699 025314 012737 000000 001124      MOV    #0,$GDDAT      ;GET GOOD = 0
8700
8701 025322 017737 156534 001126      MOV    @RHW C,$BDDAT  ;READ RHW C FOR COMPARISON
8702 025330 023737 001124 001126      CMP    $GDDAT,$BDDAT ;COMPARE EXPECTED
8703                                     ;DATA WITH DATA READ FROM
8704                                     ;RHW C
8705 025336 001401                       BEQ    87$           ;BRANCH IF GOOD
8706 025340 104113                       ERROR  113
8707
8708                                     ;AFTER SETTING "CLR" BIT #5
8709                                     ;IN RHCS2 TO INIT THE RH
8710                                     ;IPCK2 BIT #2 RHCS3 WAS SET
8711                                     ;ZEROS WERE MOVED INTO
8712                                     ;RHDB
8713                                     ;RHDB WAS READ THEN
8714                                     ;AN RH CLEAR (RHCS2 BIT #5)
8715                                     ;WAS GIVEN THIS SHOULD
8716                                     ;CLEAR ALL ERRORS
8717                                     ;RHW C SHOULD HAVE 0
8718                                     ;BUT CONTAINED WHAT IS
8719 025342                               87$:
8720
8721
8722
8723
8724
8725
8726
8727
8728
8729
8730
8731
8732
8733
8734
8735
8736
8737 025342 000004                       ;*****
8738 025344 012706 001000                       ;TEST 33      RHSC2 MDPE BIT #8 AND RHCS3 IPCK3 BIT #3
8739 025350 012737 000033 004656           ;*          CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
8740
8741 025356 004737 040322                       ;*          SET IPCK3 (RHCS3 BIT #3)
8742
8743
8744 025362 012737 000003 004662           ;*          MOVE ALL ZEROS INTO RHDB TWICE
8745
8746 025370 052777 000010 156534           ;*          THIS SHOULD NOT SET TRE SC IN RHCS1
;*****
;ST33: SCOPE
MOV    #STACK,SP      ;RESET STACK
MOV    #33,@#1STNM    ;SAVE TEST NUMBER
JSR    PC,@#CLDISK    ;GIVE RH INITIALIZE
;SETUP UNIT NUBER
MOV    #3,IP          ;CLEAR RHW C AND FUNCTION BITS IN RHCS1
;IPCK BIT NO. FOR PRINTOUT
BIS    #IPCK3,@RHCS3 ;SET IPCK3 IN RHCS3-BIT #3

```

```

8747 025376 005037 004210          CLR      WRFROM          ;WRITE ZERO INTO LOCATION
8748 025402 013777 004210 156472    MOV      WRFROM, @RHDB  ;WRITE ZEROS INTO RHDB ONCE
8749 025410 013777 004210 156464    MOV      WRFROM, @RHDB  ;WRITE ZERO SECOND TIME
8750
8751                                     ;CHECK THAT RHCS1 HAS DVA!RDY
8752 025416 012737 004200 001124    MOV      #DVA!RDY, $GDDAT ;GET GOOD = 4200
8753
8754 025424 017737 156430 001126    MOV      @RHCS1, $BDDAT  ;READ RHCS1 FOR COMPARISON
8755 025432 023737 001124 001126    CMP      $GDDAT, $BDDAT  ;COMPARE EXPECTED
8756                                     ;DATA WITH DATA READ FROM
8757                                     ;RHCS1
8758 025440 001401          BEQ      65$             ;BRANCH IF GOOD
8759 025442 104140          ERROR    140
8760                                     ;AFTER SETTING "CLR" BIT #5
8761                                     ;IN RHCS2 TO INIT THE RH
8762                                     ;IPCK3 BIT #3 RHCS3 WAS SET
8763                                     ;ZEROS WERE MOVED INTO RHDB
8764                                     ;TWICE
8765                                     ;RHCS1 SHOULD HAVE DVA!RDY
8766                                     ;=4200
8767                                     ;BUT CONTAINED WHAT IS
8768                                     ;GIVEN IN BAD RHCS1
8769 025444          65$:
8770 025444 017737 156432 001200    MOV      @RHDB, $TMP1   ;READ RHDB ONCE
8771
8772
8773                                     ;CHECK THAT RHCS3 HAS 10
8774 025452 012737 000010 001124    MOV      #10, $GDDAT    ;GET GOOD = 10
8775
8776 025460 017737 156446 001126    MOV      @RHCS3, $BDDAT ;READ RHCS3 FOR COMPARISON
8777 025466 023737 001124 001126    CMP      $GDDAT, $BDDAT ;COMPARE EXPECTED
8778                                     ;DATA WITH DATA READ FROM
8779                                     ;RHCS3
8780 025474 001401          BEQ      67$             ;BRANCH IF GOOD
8781 025476 104104          ERROR    104
8782                                     ;AFTER SETTING "CLR" BIT #5
8783                                     ;IN RHCS2 TO INIT THE RH
8784                                     ;IPCK3 BIT #3 RHCS3 WAS SET
8785                                     ;ZEROS WERE MOVED INTO
8786                                     ;RHDB
8787                                     ;RHDB WAS READ THEN
8788                                     ;RHCS3 SHOULD HAVE 10
8789                                     ;=10
8790                                     ;BUT CONTAINED WHAT IS
8791                                     ;GIVEN IN BAD RHCS3
8792 025500          67$:
8793
8794                                     ;CHECK THAT RHCS2 HAS MDPE!OR!IR
8795 025500 012737 000700 001124    MOV      #MDPE!OR!IR, $GDDAT ;GET GOOD = 700
8796 025506 053737 005010 001124    BIS      UNIT, $GDDAT    ;INCLUDE UNIT NUMBER
8797
8798 025514 017737 156350 001126    MOV      @RHCS2, $BDDAT ;READ RHCS2 FOR COMPARISON
8799 025522 023737 001124 001126    CMP      $GDDAT, $BDDAT ;COMPARE EXPECTED
8900                                     ;DATA WITH DATA READ FROM
8901                                     ;RHCS2
8902 025530 001401          BEQ      69$             ;BRANCH IF GOOD

```

```

8803 025532 104105          ERROR 105
8804
8805
8806
8807
8808
8809
8810
8811
8812
8813
8814
8815 025534          69$:
8816
8817 025534 017737 156342 001202      MOV    @RHDB,$TMP2      ;READ RHDB SECOND TIME
8818
8819 025542 012737 144200 001124      ;CHECK THAT RHCS1 HAS SC!TRE!DVA!RDY
8820
8821 025550 017737 156304 001126      MOV    @RHCS1,$BDDAT   ;READ RHCS1 FOR COMPARISON
8822 025556 023737 001124 001126      CMP    $GDDAT,$BDDAT   ;COMPARE EXPECTED
8823
8824
8825 025564 001401          BEQ    71$             ;BRANCH IF GOOD
8826 025566 104140          ERROR 140
8827
8828
8829
8830
8831
8832
8833
8834
8835
8836
8837 025570          71$:
8838
8839 025570 012737 000500 001124      ;CHECK THAT RHCS2 HAS MDPE!IR
8840 025576 053737 005010 001124      MOV    #MDPE!IR,$GDDAT ;GET GOOD = 500
8841
8842 025604 017737 156260 001126      BIS    UNIT,$GDDAT     ;INCLUDE UNIT NUMBER
8843 025612 023737 001124 001126      MOV    @RHCS2,$BDDAT   ;READ RHCS2 FOR COMPARISON
8844
8845
8846 025620 001401          BEQ    73$             ;BRANCH IF GOOD
8847 025622 104105          ERROR 105
8848
8849
8850
8851
8852
8853
8854
8855
8856
8857
8858
8859

```

```

;AFTER SETTING "CLR" BIT #5
;IN RHCS2 TO INIT THE RH
;IPCK3 BIT #3 RHCS3 WAS SET
;ZEROS WERE MOVED INTO
;RHDB
;RHDB WAS READ THEN
;RHCS2 SHOULD HAVE MDPE!OR!IR
;=700
;TOGETHER WITH UNIT NUMBER
;BUT CONTAINED WHAT IS
;GIVEN IN BAD RHCS2

;AFTER SETTING "CLR" BIT #5
;IN RHCS2 TO INIT THE RH
;IPCK3 BIT #3 RHCS3 WAS SET
;ZEROS WERE MOVED INTO
;RHDB
;RHDB WAS READ THEN
;RHCS1 SHOULD HAVE SC!TRE!DVA!RDY
;=144200
;BUT CONTAINED WHAT IS
;GIVEN IN BAD RHCS1

;AFTER SETTING "CLR" BIT #5
;IN RHCS2 TO INIT THE RH
;IPCK3 BIT #3 RHCS3 WAS SET
;ZEROS WERE MOVED INTO
;RHDB
;RHDB WAS READ THEN
;RHCS2 SHOULD HAVE MDPE!IR
;=500
;TOGETHER WITH UNIT NUMBER
;BUT CONTAINED WHAT IS
;GIVEN IN BAD RHCS2

```



MAINDEC-11-DEHAC-A  
DEHAC.P11 T33

MACY11 27(732) 09-SEP-76 07:55 PAGE 167  
RHSC2 MOPE BIT #8 AND RHCS3 IPCK3 BIT #3

```

8859 025624          73$:
8860                                     ;CHECK THAT RHCS3 HAS 10
8861 025624 012737 000010 001124      MOV      #10,$GDDAT      ;GET GOOD = 10
8862
8863 025632 017737 156274 001126      MOV      @RHCS3,$BDDAT  ;READ RHCS3 FOR COMPARISON
8864 025640 023737 001124 001126      CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED
8865                                     ;DATA WITH DATA READ FROM
8866                                     ;RHCS3
8867 025646 001401          BEQ      75$           ;BRANCH IF GOOD
8868 025650 104104          ERROR    104
8869
8870                                     ;AFTER SETTING "CLR" BIT #5
8871                                     ;IN RHCS2 TO INIT THE RH
8872                                     ;IPCK3 BIT #3 RHCS3 WAS SET
8873                                     ;ZEROS WERE MOVED INTO
8874                                     ;RHDB
8875                                     ;RHDB WAS READ THEN
8876                                     ;RHCS3 SHOULD HAVE 10
8877                                     ;=10
8878                                     ;BUT CONTAINED WHAT IS
8879                                     ;GIVEN IN BAD RHCS3
8879 025652          75$:
8880
8881 025652 004737 040322      JSR      PC,@#CLDISK   ;GIVE RH INITIALIZE
8882                                     ;SETUP UNIT NUBER
8883                                     ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
8884
8885
8886                                     ;CHECK THAT RHCS1 HAS RDY!DVA
8887 025656 012737 004200 001124      MOV      #RDY!DVA,$GDDAT ;GET GOOD = 4200
8888
8889 025664 017737 156170 001126      MOV      @RHCS1,$BDDAT ;READ RHCS1 FOR COMPARISON
8890 025672 023737 001124 001126      CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED
8891                                     ;DATA WITH DATA READ FROM
8892                                     ;RHCS1
8893 025700 001401          BEQ      77$           ;BRANCH IF GOOD
8894 025702 104106          ERROR    106
8895
8896                                     ;AFTER SETTING "CLR" BIT #5
8897                                     ;IN RHCS2 TO INIT THE RH
8898                                     ;IPCK3 BIT #3 RHCS3 WAS SET
8899                                     ;ZEROS WERE MOVED INTO
8900                                     ;RHDB
8901                                     ;RHDB WAS READ THEN
8902                                     ;AN RH CLEAR (RHCS2 BIT #5)
8903                                     ;WAS GIVEN THIS SHOULD
8904                                     ;CLEAR ALL ERRORS
8905                                     ;RHCS1 SHOULD HAVE RDY!DVA
8906                                     ;=4200
8907                                     ;BUT CONTAINED WHAT IS
8908                                     ;GIVEN IN BAD RHCS1
8908 025704          77$:
8909                                     ;CHECK THAT RHCS2 HAS IR
8910 025704 012737 000100 001124      MOV      #IR,$GDDAT    ;GET GOOD = 100
8911 025712 053737 005010 001124      BIS      UNIT,$GDDAT   ;INCLUDE UNIT NUMBER
8912
8913 025720 017737 156144 001126      MOV      @RHCS2,$BDDAT ;READ RHCS2 FOR COMPARISON
8914 025726 023737 001124 001126      CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED

```



```

8971                                     ;AN RH CLEAR (RHCS2 BIT #5)
8972                                     ;WAS GIVEN THIS SHOULD
8973                                     ;CLEAR ALL ERRORS
8974                                     ;RHBA SHOULD HAVE 0
8975                                     ;BUT CONTAINED WHAT IS
8976                                     ;GIVEN IN BAD RHBA
8977 026014                               83$:
8978                                     ;CHECK THAT RHBAE HAS 0
8979 026014 012737 000000 001124      MOV      #0,$GDDAT      ;GET GOOD = 0
8980
8981 026022 017737 156102 001126      MOV      @RHBAE,$BDDAT  ;READ RHBAE FOR COMPARISON
8982 026030 023737 001124 001126      CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED
8983                                     ;DATA WITH DATA READ FROM
8984                                     ;RHBAE
8985 026036 001401                          BEQ      85$           ;BRANCH IF GOOD
8986 026040 104112                          ERROR    112
8987
8988                                     ;AFTER SETTING "CLR" BIT #5
8989                                     ;IN RHCS2 TO INIT THE RH
8990                                     ;IPCK3 BIT #3 RHCS3 WAS SET
8991                                     ;ZEROS WERE MOVED INTO
8992                                     ;RHDB
8993                                     ;RHDB WAS READ THEN
8994                                     ;AN RH CLEAR (RHCS2 BIT #5)
8995                                     ;WAS GIVEN THIS SHOULD
8996                                     ;CLEAR ALL ERRORS
8997                                     ;RHBAE SHOULD HAVE 0
8998                                     ;BUT CONTAINED WHAT IS
8999                                     ;GIVEN IN BAD RHBAE
9000
9001 026042                               85$:
9002                                     ;CHECK THAT RHWC HAS 0
9003 026042 012737 000000 001124      MOV      #0,$GDDAT      ;GET GOOD = 0
9004 026050 017737 156006 001126      MOV      @RHWC,$BDDAT  ;READ RHWC FOR COMPARISON
9005 026056 023737 001124 001126      CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED
9006                                     ;DATA WITH DATA READ FROM
9007                                     ;RHWC
9008 026064 001401                          BEQ      87$           ;BRANCH IF GOOD
9009 026066 104113                          ERROR    113
9010
9011                                     ;AFTER SETTING "CLR" BIT #5
9012                                     ;IN RHCS2 TO INIT THE RH
9013                                     ;IPCK3 BIT #3 RHCS3 WAS SET
9014                                     ;ZEROS WERE MOVED INTO
9015                                     ;RHDB
9016                                     ;RHDB WAS READ THEN
9017                                     ;AN RH CLEAR (RHCS2 BIT #5)
9018                                     ;WAS GIVEN THIS SHOULD
9019                                     ;CLEAR ALL ERRORS
9020                                     ;RHWC SHOULD HAVE 0
9021                                     ;BUT CONTAINED WHAT IS
9022                                     ;GIVEN IN BAD RHWC
9023
9024 026070                               87$:
9025
9026

```

\*\*\*\*\*

```

9027          ;*TEST 34          RHCS2-WCE BIT #14 AND RHCS3-WCE OW BIT #12
9028
9029          ;*          CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5,
9030          ;*          WRITE NINE WORD OF ALL NINES ON TO THE DEVICE
9031          ;*          SET UP TO DO WRITE CHECK ON THAT WORD FROM AN
9032          ;*          ODD WORD BOUNDARY
9033          ;*          DO A ONE WORD WRITE CHECK WITH RHBA FROM AN
9034          ;*          ODD WORD BOUNDARY
9035          ;*          THIS SHOULD SET WCE OW (RHCS3 BIT #12)
9036          ;*          AND WCE (RHCS2 BIT #14)
9037          ;*          "CLR" (RHCS2 BIT #5) IS GIVEN
9038          ;*          ALL ERRORS ARE CLEARED
9039          ;*          CHECK RHCS1, RHCS2, RHCS3, RHBA, RHBAE, RHWC
9040
9041          ;*****
9042 026070 000004          †ST34: SCOPE
9043 026072 012706 001000          MOV      #STACK, SP          ;RESET STACK
9044 026076 012737 000034 004656          MOV      #34, @#TSTNM          ;SAVE TEST NUMBER
9045
9046 026104 004737 040322          JSR      PC, @#CLDISK          ;GIVE RH INITIALIZE
9047                                     ;SETUP UNIT NUBER
9048                                     ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
9049
9050          ;WRITE ONE WORD OF ALL ONES ON THE DEVICE
9051 026110 012737 177777 004210          MOV      #-1, WRFROM          ;ALL ONES INTO WRITE FROM
9052 026116 012777 177767 155736          MOV      #-9, @RHWC          ;NINE WORD FOR WORD COUNT REGISTER
9053 026124 012777 004210 155732          MOV      #WRFROM, @RHBA          ;WRITE FROM BUFFER INTO BUS ADDRESS
9054
9055 026132 004077 156502          JSR      RD, @COMND          ;GO TO DO COMMAND
9056 026136 004154          WRIDAT          ;WRITE DATA
9057
9058          ;SET UP FOR WRITE CHECK
9059 026140 005037 003002          CLR      BUFOW          ;WRITE DATA FOR WRITE CHECK ERROR
9060 026144 012777 177767 155710          MOV      #-9, @RHWC          ;NINE WORD FOR WORD COUNT REGISTER
9061 026152 012777 003002 155704          MOV      #BUFOW, @RHBA          ;WRITE FROM BUFFER INTO BUS ADDRESS
9062                                     ;FROM ODD WORD BOUNDARY
9063          ;WRITE CHECK ON ODD WORD BOUNDARY
9064 026160 004077 156454          JSR      RD, @COMND          ;GO TO DC COMMAND
9065 026164 004150          WRCHEK          ;WRITE CHECK DATA
9066
9067
9068
9069
9070          ;CHECK THAT RHCS2 HAS WCE!OR!IR
9071 026166 012737 040300 001124          MOV      #WCE!OR!IR, $GDDAT          ;GET GOOD = 40300
9072 026174 053737 005010 001124          BIS      UNIT, $GDDAT          ;INCLUDE UNIT NUMBER
9073
9074 026202 017737 155662 001126          MOV      @RHCS2, $BDDAT          ;READ RHCS2 FOR COMPARISON
9075 026210 023737 001124 001126          CMP      $GDDAT, $BDDAT          ;COMPARE EXPECTED
9076                                     ;DATA WITH DATA READ FROM
9077                                     ;RHCS2
9078 026216 001401          BEQ      65$          ;BRANCH IF GOOD
9079 026220 104114          ERROR   114
9080
9081          ;AFTER SETTING "CLR" BIT #5
9082          ;IN RHCS2 TO INIT THE RH
          ;ONE WORD OF ALL ONES IS

```

```

9083 ;WRITTEN ON THE DEVICE
9084 ;A WRITE CHECK WAS DONE
9085 ;ON AN ODD WORD BOUNDARY THEN
9086 ;RHCS2 SHOULD HAVE WCE!OR!IR
9087 ;=40300
9088 ;TOGETHER WITH UNIT NUMBER
9089 ;BUT CONTAINED WHAT IS
9090 ;GIVEN IN BAD RHCS2
9091 026222 65$:
9092
9093 ;CHECK THAT RHCS3 HAS DBL!WCEOW
9094 026222 012737 012000 001124 MOV #DBL!WCEOW,$GDDAT ;GET GOOD = 12000
9095
9096 026230 017737 155676 001126 MOV @RHCS3,$BDDAT ;READ RHCS3 FOR COMPARISON
9097 026236 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
9098 ;DATA WITH DATA READ FROM
9099 ;RHCS3
9100 026244 001401 BEQ 67$ ;BRANCH IF GOOD
9101 026246 104115 ERROR 115
9102 ;AFTER SETTING "CLR" BIT #5
9103 ;IN RHCS2 TO INIT THE RH
9104 ;ONE WORD OF ALL ONES IS
9105 ;WRITTEN ON THE DEVICE
9106 ;A WRITE CHECK WAS DONE
9107 ;ON AN ODD WORD BOUNDARY THEN
9108 ;RHCS3 SHOULD HAVE DBL!WCEOW
9109 ;=12000
9110 ;BUT CONTAINED WHAT IS
9111 ;GIVEN IN BAD RHCS3
9112 026250 67$:
9113
9114
9115 026250 004737 040322 JSR PC,@#CLDISK ;GIVE RH INITIALIZE
9116 ;SETUP UNIT NUBER
9117 ;CLEAR RHWG AND FUNCTION BITS IN RHCS1
9118
9119
9120 ;CHECK THAT RHCS1 HAS RDY!DVA
9121 026254 012737 004200 001124 MOV #RDY!DVA,$GDDAT ;GET GOOD = 4200
9122
9123 026262 017737 155572 001126 MOV @RHCS1,$BDDAT ;READ RHCS1 FOR COMPARISON
9124 026270 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
9125 ;DATA WITH DATA READ FROM
9126 ;RHCS1
9127 026276 001401 BEQ 69$ ;BRANCH IF GOOD
9128 026300 104116 ERROR 116
9129 ;AFTER SETTING "CLR" BIT #5
9130 ;IN RHCS2 TO INIT THE RH
9131 ;ONE WORD OF ALL ONES IS
9132 ;WRITTEN ON THE DEVICE
9133 ;A WRITE CHECK WAS DONE
9134 ;ON AN ODD WORD BOUNDARY THEN
9135 ;AN RH CLEAR (RHCS2 BIT #5)
9136 ;WAS GIVEN THIS SHOULD
9137 ;CLEAR ALL ERRORS
9138 ;RHCS1 SHOULD HAVE RDY!DVA

```

```

9139                                     :=4200
9140                                     ;BUT CONTAINED WHAT IS
9141                                     ;GIVEN IN BAD RHCS1
9142 026302                               69$:
9143                                     ;CHECK THAT RHCS2 HAS IR
9144 026302 012737 000100 001124          MOV   #IR,$GDDAT   ;GET GOOD = 100
9145 026310 053737 005010 001124          BIS   UNIT,$GDDAT  ;INCLUDE UNIT NUMBER
9146
9147 026316 017737 155546 001126          MOV   @RHCS2,$BDDAT ;READ RHCS2 FOR COMPARISON
9148 026324 023737 001124 001126          CMP   $GDDAT,$BDDAT ;COMPARE EXPECTED
9149                                     ;DATA WITH DATA READ FROM
9150                                     ;RHCS2
9151 026332 001401                          BEQ   71$           ;BRANCH IF GOOD
9152 026334 104117
9153                                     ;AFTER SETTING "CLR" BIT #5
9154                                     ;IN RHCS2 TO INIT THE RH
9155                                     ;ONE WORD OF ALL ONES IS
9156                                     ;WRITTEN ON THE DEVICE
9157                                     ;A WRITE CHECK WAS DONE
9158                                     ;ON AN ODD WORD BOUNDARY THEN
9159                                     ;AN RH CLEAR (RHCS2 BIT #5)
9160                                     ;WAS GIVEN THIS SHOULD
9161                                     ;CLEAR ALL ERRORS
9162                                     ;RHCS2 SHOULD HAVE IR
9163                                     ;=100
9164                                     ;TOGETHER WITH UNIT NUMBER
9165                                     ;BUT CONTAINED WHAT IS
9166                                     ;GIVEN IN BAD RHCS2
9167 026336                               71$:
9168                                     ;CHECK THAT RHCS3 HAS 0
9169 026336 012737 000000 001124          MOV   #0,$GDDAT   ;GET GOOD = 0
9170
9171 026344 017737 155562 001126          MOV   @RHCS3,$BDDAT ;READ RHCS3 FOR COMPARISON
9172 026352 023737 001124 001126          CMP   $GDDAT,$BDDAT ;COMPARE EXPECTED
9173                                     ;DATA WITH DATA READ FROM
9174                                     ;RHCS3
9175 026360 001401                          BEQ   73$           ;BRANCH IF GOOD
9176 026362 104120
9177                                     ;AFTER SETTING "CLR" BIT #5
9178                                     ;IN RHCS2 TO INIT THE RH
9179                                     ;ONE WORD OF ALL ONES IS
9180                                     ;WRITTEN ON THE DEVICE
9181                                     ;A WRITE CHECK WAS DONE
9182                                     ;ON AN ODD WORD BOUNDARY THEN
9183                                     ;AN RH CLEAR (RHCS2 BIT #5)
9184                                     ;WAS GIVEN THIS SHOULD
9185                                     ;CLEAR ALL ERRORS
9186                                     ;RHCS3 SHOULD HAVE 0
9187                                     ;BUT CONTAINED WHAT IS
9188                                     ;GIVEN IN BAD RHCS3
9189 026364                               73$:
9190                                     ;CHECK THAT RHBA HAS 0
9191 026364 012737 000000 001124          MOV   #0,$GDDAT   ;GET GOOD = 0
9192
9193 026372 017737 155466 001126          MOV   @RHBA,$BDDAT ;READ RHBA FOR COMPARISON
9194 026400 023737 001124 001126          CMP   $GDDAT,$BDDAT ;COMPARE EXPECTED

```



MAINDEF-11-DERHAC-A  
DERHAC.P11 T34

MACY11 27(732) 09-SEP-76 07:55 PAGE 174  
RHCS2-WCE BIT #14 AND RHCS3-WCE OW BIT #12

```

9251 ;CLEAR ALL ERRORS
9252 ;RHWC SHOULD HAVE 0
9253 ;BUT CONTAINED WHAT IS
9254 ;GIVEN IN BAD RHWC
9255 026466 79$:
9256
9257
9258
9259 ::*****
9260 :*TEST 35 RHCS2-WCE BIT #14 AND RHCS3-WCE OW BIT #12
9261
9262 :* CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
9263 :* WRITE NINE WORD OF ALL NINES ON TO THE DEVICE
9264 :* SET UP TO DO WRITE CHECK ON THAT WORD FROM AN
9265 :* ODD WORD BOUNDARY
9266 :* DO A ONE WORD WRITE CHECK WITH RHBA FROM AN
9267 :* ODD WORD BOUNDARY
9268 :* THIS SHOULD SET WCE OW (RHCS3 BIT #12)
9269 :* AND WCE (RHCS2 BIT #14)
9270 :* "CLR" (RHCS2 BIT #5) IS GIVEN
9271 :* ALL ERRORS ARE CLEARED
9272 :* CHECK RHCS1, RHCS2, RHCS3, RHBA, RHBAE, RHWC
9273
9274 ::*****
9275 026466 000004 †ST35: SCOPE
9276 026470 012706 001000 MOV #STACK,SP ;RESET STACK
9277 026474 012737 000035 004656 MOV #35,‡†STNM ;SAVE TEST NUMBER
9278
9279 026502 004737 040322 JSR PC,‡‡CLDISK ;GIVE RH INITIALIZE
9280 ;SETUP UNIT NUBER
9281 ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
9282
9283 ;WRITE ONE WORD OF ALL ONES ON THE DEVICE
9284 026506 012737 177777 004210 MOV #-1,‡‡RFROM ;ALL ONES INTO WRITE FROM
9285 026514 012777 177767 155340 MOV #-9,‡‡RHWC ;NINE WORD FOR WORD CCUNT REGISTER
9286 026522 012777 004210 155334 MOV #‡‡RFROM,‡‡RHBA ;WRITE FROM BUFFER INTO BUS ADDRESS
9287
9288 026530 004077 156104 JSR RD,‡‡COMND ;GO TO DO COMMAND
9289 026534 004154 WRIDAT ;WRITE DATA
9290
9291 ;SET UP FOR WRITE CHECK
9292 026536 005037 003002 CLR BUFOW ;WRITE DATA FOR WRITE CHECK ERROR
9293 026542 012777 177767 155312 MOV #-9,‡‡RHWC ;NINE WORD FOR WORD COUNT REGISTER
9294 026550 012777 003002 155306 MOV #BUFOW,‡‡RHBA ;WRITE FROM BUFFER INTO BUS ADDRESS
9295 ;FROM ODD WORD BOUNDARY
9296 ;WRITE CHECK ON ODD WORD BOUNDARY
9297 026556 004077 156056 JSR RD,‡‡COMND ;GO TO DO COMMAND
9298 026562 004150 WRCHK ;WRITE CHECK DATA
9299
9300
9301
9302
9303 ;CHECK THAT RHCS2 HAS WCE!OR!IR
9304 026564 012737 040300 001124 MOV #WCE!OR!IR,$GDDAT ;GET GOOD = 40300
9305 026572 053737 005010 001124 BIS UNIT,$GDDAT ;INCLUDE UNIT NUMBER
9306

```





9363  
9364  
9365  
9366  
9367  
9368  
9369  
9370  
9371  
9372  
9373  
9374  
9375  
9376  
9377  
9378  
9379  
9380  
9381  
9382  
9383  
9384  
9385  
9386  
9387  
9388  
9389  
9390  
9391  
9392  
9393  
9394  
9395  
9396  
9397  
9398  
9399  
9400  
9401  
9402  
9403  
9404  
9405  
9406  
9407  
9408  
9409  
9410  
9411  
9412  
9413  
9414  
9415  
9416  
9417  
9418

026700

69\$:

026700 012737 000100 001124  
026706 053737 005010 001124  
026714 017737 155150 001126  
026722 023737 001124 001126  
026730 001401  
026732 104117

;CHECK THAT RHCS2 HAS IR  
MOV #IR,\$GDDAT  
BIS UNIT,\$GDDAT  
MOV @RHCS2,\$BDDAT  
CMP \$GDDAT,\$BDDAT  
BEQ 71\$  
ERROR 117

; IN RHCS2 TO INIT THE RH  
; ONE WORD OF ALL ONES IS  
; WRITTEN ON THE DEVICE  
; A WRITE CHECK WAS DONE  
; ON AN ODD WORD BOUNDARY THEN  
; AN RH CLEAR (RHCS2 BIT #5)  
; WAS GIVEN THIS SHOULD  
; CLEAR ALL ERRORS  
; RHCS1 SHOULD HAVE RDY!DVA  
; =4200  
; BUT CONTAINED WHAT IS  
; GIVEN IN BAD RHCS1

026734

71\$:

026734 012737 000000 001124  
026742 017737 155164 001126  
026750 023737 001124 001126  
026756 001401  
026760 104120

;CHECK THAT RHCS3 HAS 0  
MOV #0,\$GDDAT  
MOV @RHCS3,\$BDDAT  
CMP \$GDDAT,\$BDDAT  
BEQ 73\$  
ERROR 120

; AFTER SETTING "CLR" BIT #5  
; IN RHCS2 TO INIT THE RH  
; ONE WORD OF ALL ONES IS  
; WRITTEN ON THE DEVICE  
; A WRITE CHECK WAS DONE  
; ON AN ODD WORD BOUNDARY THEN  
; AN RH CLEAR (RHCS2 BIT #5)  
; WAS GIVEN THIS SHOULD  
; CLEAR ALL ERRORS  
; RHCS2 SHOULD HAVE IR  
; =100  
; TOGETHER WITH UNIT NUMBER  
; BUT CONTAINED WHAT IS  
; GIVEN IN BAD RHCS2

; AFTER SETTING "CLR" BIT #5  
; IN RHCS2 TO INIT THE RH  
; ONE WORD OF ALL ONES IS  
; WRITTEN ON THE DEVICE  
; A WRITE CHECK WAS DONE  
; ON AN ODD WORD BOUNDARY THEN  
; AN RH CLEAR (RHCS2 BIT #5)  
; WAS GIVEN THIS SHOULD  
; CLEAR ALL ERRORS



9475 027062 104123  
9476  
9477  
9478  
9479  
9480  
9481  
9482  
9483  
9484  
9485  
9486  
9487  
9488 027064  
9489  
9490  
9491  
9492  
9493  
9494  
9495  
9496  
9497  
9498  
9499  
9500  
9501  
9502  
9503  
9504  
9505  
9506  
9507  
9508 027064 000004  
9509 027066 012706 001000  
9510 027072 012737 000036 004656  
9511  
9512 027100 004737 040322  
9513  
9514  
9515  
9516  
9517 027104 012737 177777 004210  
9518 027112 012777 177767 154742  
9519 027120 012777 004210 154736  
9520  
9521 027126 004077 155506  
9522 027132 004154  
9523  
9524  
9525 027134 005037 003000  
9526 027140 012777 177767 154714  
9527 027146 012777 003000 154710  
9528  
9529  
9530 027154 004077 155460

ERROR 123

: AFTER SETTING "CLR" BIT #5  
: IN RHCS2 TO INIT THE RH  
: ONE WORD OF ALL ONES IS  
: WRITTEN ON THE DEVICE  
: A WRITE CHECK WAS DONE  
: ON AN ODD WORD BOUNDARY THEN  
: AN RH CLEAR (RHCS2 BIT #5)  
: WAS GIVEN THIS SHOULD  
: CLEAR ALL ERRORS  
: RHWC SHOULD HAVE 0  
: BUT CONTAINED WHAT IS  
: GIVEN IN BAD RHWC

79\$:

::\*\*\*\*\*  
;\*TEST 36 RHCS2-WCE BIT #14 AND RHCS3-WCE EW BIT #11

\* CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)  
\* WRITE NINE WORD OF ALL NINES ON TO THE DEVICE  
\* SET UP TO DO WRITE CHECK ON THAT WORD FROM AN  
\* EVEN WORD BOUNDARY  
\* DO A ONE WORD WRITE CHECK WITH RHBA FROM AN  
\* EVEN WORD BOUNDARY  
\* THIS SHOULD SET WCE EW (RHCS3 BIT #11)  
\* AND WCE (RHCS2 BIT #14)  
\* "CLR" (RHCS2 BIT #5) IS GIVEN  
\* ALL ERRORS ARE CLEARED  
\* CHECK RHCS1, RHCS2, RHCS3, RHBA, RHBAE, RHWC

::\*\*\*\*\*  
†ST36:

SCOPE  
MOV #STACK, SP ;RESET STACK  
MOV #36, #†STNM ;SAVE TEST NUMBER  
JSR PC, #CLDISK ;GIVE RH INITIALIZE  
;SETUP UNIT NUBER  
;CLEAR RHWC AND FUNCTION BITS IN RHCS1  
;WRITE ONE WORD OF ALL ONES ON THE DEVICE  
MOV #-1, WRFROM ;ALL ONES INTO WRITE FROM  
MOV #-9, #RHWC ;NINE WORD FOR WORD COUNT REGISTER  
MOV #WRFROM, #RHBA ;WRITE FROM BUFFER INTO BUS ADDRESS  
JSR RO, #COMND ;GO TO DO COMMAND  
WRIDAT ;WRITE DATA  
;SET UP FOR WRITE CHECK  
CLR BUFEW ;WRITE DATA FOR WRITE CHECK ERROR  
MOV #-9, #RHWC ;NINE WORD FOR WORD COUNT REGISTER  
MOV #BUFEW, #RHBA ;WRITE FROM BUFFER INTO BUS ADDRESS  
;FROM EVEN WORD BOUNDARY  
;WRITE CHECK ON EVEN WORD BOUNDARY  
JSR RO, #COMND ;GO TO DO COMMAND

```

9531 027160 004150          WRCHK          ;WRITE CHECK DATA
9532
9533
9534
9535
9536          ;CHECK THAT RHCS2 HAS WCE!OR!IR
9537 027162 012737 040300 001124      MOV      #WCE!OR!IR,$GDDAT      ;GET GOOD = 40300
9538 027170 053737 005010 001124      BIS      UNIT,$GDDAT          ;INCLUDE UNIT NUMBER
9539
9540 027176 017737 154666 001126      MCV      @RHCS2,$BDDAT        ;READ RHCS2 FOR COMPARISON
9541 027204 023737 001124 001126      CMP      $GDDAT,$BDDAT        ;COMPARE EXPECTED
9542          ;DATA WITH DATA READ FROM
9543          ;RHCS2
9544 027212 001401          BEQ      65$                  ;BRANCH IF GOOD
9545 027214 104114          ERROR    114
9546          ;AFTER SETTING "CLR" BIT #5
9547          ;IN RHCS2 TO INIT THE RH
9548          ;ONE WORD OF ALL ONES IS
9549          ;WRITTEN ON THE DEVICE
9550          ;A WRITE CHECK WAS DONE
9551          ;ON AN EVEN WORD BOUNDARY THEN
9552          ;RHCS2 SHOULD HAVE WCE!OR!IR
9553          ;=40300
9554          ;TOGETHER WITH UNIT NUMBER
9555          ;BUT CONTAINED WHAT IS
9556          ;GIVEN IN BAD RHCS2
9557 027216          65$:
9558
9559          ;CHECK THAT RHCS3 HAS DBL!WCEEW
9560 027216 012737 006000 001124      MOV      #DBL!WCEEW,$GDDAT    ;GET GOOD = 6000
9561
9562 027224 017737 154702 001126      MOV      @RHCS3,$BDDAT        ;READ RHCS3 FOR COMPARISON
9563 027232 023737 001124 001126      CMP      $GDDAT,$BDDAT        ;COMPARE EXPECTED
9564          ;DATA WITH DATA READ FROM
9565          ;RHCS3
9566 027240 001401          BEQ      67$                  ;BRANCH IF GOOD
9567 027242 104115          ERROR    115
9568          ;AFTER SETTING "CLR" BIT #5
9569          ;IN RHCS2 TO INIT THE RH
9570          ;ONE WORD OF ALL ONES IS
9571          ;WRITTEN ON THE DEVICE
9572          ;A WRITE CHECK WAS DONE
9573          ;ON AN EVEN WORD BOUNDARY THEN
9574          ;RHCS3 SHOULD HAVE DBL!WCEEW
9575          ;=6000
9576          ;BUT CONTAINED WHAT IS
9577          ;GIVEN IN BAD RHCS3
9578 027244          67$:
9579
9580
9581 027244 004737 040322          JSR      PC,@CLDISK          ;GIVE RH INITIALIZE
9582          ;SETUP UNIT NUBER
9583          ;CLEAR RHWC AND FUNCTION BITS IN RHCS:
9584
9585
9586          ;CHECK THAT RHCS1 HAS RDY!DVA

```

```

9587 027250 012737 004200 001124      MOV      #RDY!DVA,$GDDAT ;GET GOOD = 4200
9588
9589 027256 017737 154576 001126      MOV      @RHCS1,$BDDAT ;READ RHCS1 FOR COMPARISON
9590 027264 023737 001124 001126      CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED
9591                                     ;DATA WITH DATA READ FROM
9592                                     ;RHCS1
9593 027272 001401      BEQ      69$ ;BRANCH IF GOOD
9594 027274 104116      ERROR   116
9595
9596                                     ;AFTER SETTING "CLR" BIT #5
9597                                     ;IN RHCS2 TO INIT THE RH
9598                                     ;ONE WORD OF ALL ONES IS
9599                                     ;WRITTEN ON THE DEVICE
9600                                     ;A WRITE CHECK WAS DONE
9601                                     ;ON AN EVEN WORD BOUNDARY THEN
9602                                     ;AN RH CLEAR (RHCS2 BIT #5)
9603                                     ;WAS GIVEN THIS SHOULD
9604                                     ;CLEAR ALL ERRORS
9605                                     ;RHCS1 SHOULD HAVE RDY!DVA
9606                                     ;=4200
9607                                     ;BUT CONTAINED WHAT IS
9608                                     ;GIVEN IN BAD RHCS1
9608 027276                                     69$:
9609                                     ;CHECK THAT RHCS2 HAS IR
9610 027276 012737 000100 001124      MOV      #IR,$GDDAT ;GET GOOD = 100
9611 027304 053737 005010 001124      BIS      UNIT,$GDDAT ;INCLUDE UNIT NUMBER
9612
9613 027312 017737 154552 001126      MOV      @RHCS2,$BDDAT ;READ RHCS2 FOR COMPARISON
9614 027320 023737 001124 001126      CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED
9615                                     ;DATA WITH DATA READ FROM
9616                                     ;RHCS2
9617 027326 001401      BEQ      71$ ;BRANCH IF GOOD
9618 027330 104117      ERROR   117
9619
9620                                     ;AFTER SETTING "CLR" BIT #5
9621                                     ;IN RHCS2 TO INIT THE RH
9622                                     ;ONE WORD OF ALL ONES IS
9623                                     ;WRITTEN ON THE DEVICE
9624                                     ;A WRITE CHECK WAS DONE
9625                                     ;ON AN EVEN WORD BOUNDARY THEN
9626                                     ;AN RH CLEAR (RHCS2 BIT #5)
9627                                     ;WAS GIVEN THIS SHOULD
9628                                     ;CLEAR ALL ERRORS
9629                                     ;RHCS2 SHOULD HAVE IR
9630                                     ;=100
9631                                     ;TOGETHER WITH UNIT NUMBER
9632                                     ;BUT CONTAINED WHAT IS
9633                                     ;GIVEN IN BAD RHCS2
9633 027332                                     71$:
9634                                     ;CHECK THAT RHCS3 HAS 0
9635 027332 012737 000000 001124      MOV      #0,$GDDAT ;GET GOOD = 0
9636
9637 027340 017737 154566 001126      MOV      @RHCS3,$BDDAT ;READ RHCS3 FOR COMPARISON
9638 027346 023737 001124 001126      CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED
9639                                     ;DATA WITH DATA READ FROM
9640                                     ;RHCS3
9641 027354 001401      BEQ      73$ ;BRANCH IF GOOD
9642 027356 104120      ERROR   120

```

; AFTER SETTING "CLR" BIT #5  
; IN RHCS2 TO INIT THE RH  
; ONE WORD OF ALL ONES IS  
; WRITTEN ON THE DEVICE  
; A WRITE CHECK WAS DONE  
; ON AN EVEN WORD BOUNDARY THEN  
; AN RH CLEAR (RHCS2 BIT #5)  
; WAS GIVEN THIS SHOULD  
; CLEAR ALL ERRORS  
; RHCS3 SHOULD HAVE 0  
; BUT CONTAINED WHAT IS  
; GIVEN IN BAD RHCS3

; CHECK THAT RHBA HAS 0  
; GET GOOD = 0

; READ RHBA FOR COMPARISON  
; COMPARE EXPECTED  
; DATA WITH DATA READ FROM  
; RHBA  
; BRANCH IF GOOD

; AFTER SETTING "CLR" BIT #5  
; IN RHCS2 TO INIT THE RH  
; ONE WORD OF ALL ONES IS  
; WRITTEN ON THE DEVICE  
; A WRITE CHECK WAS DONE  
; ON AN EVEN WORD BOUNDARY THEN  
; AN RH CLEAR (RHCS2 BIT #5)  
; WAS GIVEN THIS SHOULD  
; CLEAR ALL ERRORS  
; RHBA SHOULD HAVE 0  
; BUT CONTAINED WHAT IS  
; GIVEN IN BAD RHBA

; CHECK THAT RHBAE HAS 0  
; GET GOOD = 0

; READ RHBAE FOR COMPARISON  
; COMPARE EXPECTED  
; DATA WITH DATA READ FROM  
; RHBAE  
; BRANCH IF GOOD

; AFTER SETTING "CLR" BIT #5  
; IN RHCS2 TO INIT THE RH  
; ONE WORD OF ALL ONES IS  
; WRITTEN ON THE DEVICE  
; A WRITE CHECK WAS DONE  
; ON AN EVEN WORD BOUNDARY THEN  
; AN RH CLEAR (RHCS2 BIT #5)  
; WAS GIVEN THIS SHOULD  
; CLEAR ALL ERRORS  
; RHBAE SHOULD HAVE 0  
; BUT CONTAINED WHAT IS  
; GIVEN IN BAD RHBAE

```
9643
9644
9645
9646
9647
9648
9649
9650
9651
9652
9653
9654
9655 027360          73$:
9656                ; CHECK THAT RHBA HAS 0
9657 027360 012737 000000 001124  MOV      #C,$GDDAT
9658
9659 027366 017737 154472 001126  MOV      @RHBA,$BDDAT
9660 027374 023737 001124 001126  CMP      $GDDAT,$BDDAT
9661
9662
9663 027402 001401      BEQ      75$
9664 027404 104121      ERROR    121
9665
9666
9667
9668
9669
9670
9671
9672
9673
9674
9675
9676
9677 027406          75$:
9678                ; CHECK THAT RHBAE HAS 0
9679 027406 012737 000000 001124  MOV      #0,$GDDAT
9680
9681 027414 017737 154510 001126  MOV      @RHBAE,$BDDAT
9682 027422 023737 001124 001126  CMP      $GDDAT,$BDDAT
9683
9684
9685 027430 001401      BEQ      77$
9686 027432 104122      ERROR    122
9687
9688
9689
9690
9691
9692
9693
9694
9695
9696
9697
9698
9699
```

```

9699 027434          77$:
9700          ;CHECK THAT RHWC HAS 0
9701 027434 012737 000000 001124      MOV      #0,$GDDAT      ;GET GOOD = 0
9702
9703 027442 017737 154414 001126      MOV      @RHWC,$BDDAT   ;READ RHWC FOR COMPARISON
9704 027450 023737 001124 001126      CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED
9705          ;DATA WITH DATA READ FROM
9706          ;RHWC
9707 027456 001401          BEQ      79$           ;BRANCH IF GOOD
9708 027460 104123          ERROR    123
9709
9710          ;AFTER SETTING "CLR" BIT #5
9711          ;IN RHCS2 TO INIT THE RH
9712          ;ONE WORD OF ALL ONES IS
9713          ;WRITTEN ON THE DEVICE
9714          ;A WRITE CHECK WAS DONE
9715          ;ON AN EVEN WORD BOUNDARY THEN
9716          ;AN RH CLEAR (RHCS2 BIT #5)
9717          ;WAS GIVEN THIS SHOULD
9718          ;CLEAR ALL ERRORS
9719          ;RHWC SHOULD HAVE 0
9720          ;BUT CONTAINED WHAT IS
9721          ;GIVEN IN BAD RHWC
9721 027462          79$:
9722
9723
9724
9725          ;*****
9726          ;*TEST 37      RHCS2-WCE BIT #14 AND RHCS3-WCE EW BIT #11
9727
9728          ;*
9729          ;*      CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
9730          ;*      WRITE NINE WORD OF ALL NINES ON TO THE DEVICE
9731          ;*      SET UP TO DO WRITE CHECK ON THAT WORD FROM AN
9732          ;*      EVEN WORD BOUNDARY
9733          ;*      DO A ONE WORD WRITE CHECK WITH RHBA FROM AN
9734          ;*      EVEN WORD BOUNDARY
9735          ;*      THIS SHOULD SET WCE EW (RHCS3 BIT #11)
9736          ;*      AND WCE (RHCS2 BIT #14)
9737          ;*      "CLR" (RHCS2 BIT #5) IS GIVEN
9738          ;*      ALL ERRORS ARE CLEARED
9739          ;*      CHECK RHCS1, RHCS2, RHCS3, RHBA, RHBAE, RHWC
9740          ;*****
9741 027462 000004          TST37: SCOPE
9742 027464 012706 001000          MOV      #STACK SP     ;RESET STACK
9743 027470 012737 000037 004656      MOV      #37,@#TSTNM  ;SAVE TEST NUMBER
9744
9745 027476 004737 040322          JSR      PC,@#CLDISK  ;GIVE RH INITIALIZE
9746          ;SETUP UNIT NUBER
9747          ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
9748
9749          ;WRITE ONE WORD OF ALL ONES ON THE DEVICE
9750 027502 012737 177777 004210      MOV      #-1,WRFROM   ;ALL ONES INTO WRITE FROM
9751 027510 012777 177767 154344      MOV      #-9,@RHWC   ;NINE WORD FOR WORD COUNT REGISTER
9752 027516 012777 004210 154340      MOV      #WRFROM,@RHBA ;WRITE FROM BUFFER INTO BLS ADDRESS
9753
9754 027524 004077 155110          JSR      RD,@COMND    ;GO TO DO COMMAND

```



```

9755 027530 004154 WRIDAT ;WRITE DATA
9756
9757 ;SET UP FOR WRITE CHECK
9758 027532 005037 003000 CLR BUFEW ;WRITE DATA FOR WRITE CHECK ERROR
9759 027536 012777 177767 154316 MOV #-9,DRHWC ;NINE WORD FOR WORD COUNT REGISTER
9760 027544 012777 003000 154312 MOV #BUFEW,DRHBA ;WRITE FROM BUFFER INTO BUS ADDRESS
9761 ;FROM EVEN WORD BOUNDARY
9762 ;WRITE CHECK ON EVEN WORD BOUNDARY
9763 027552 004077 155062 JSR RD,DCOMND ;GO TO DO COMMAND
9764 027556 004150 WRCHEK ;WRITE CHECK DATA
9765
9766
9767
9768
9769 ;CHECK THAT RHCS2 HAS WCE!OR!IR
9770 027567 012737 040300 001124 MOV #WCE!OR!IR,$GDDAT ;GET GOOD = 40300
9771 027569 053737 005010 001124 BIS UNIT,$GDDAT ;INCLUDE UNIT NUMBER
9772
9773 027574 017737 154270 001126 MOV DRHCS2,$BDDAT ;READ RHCS2 FOR COMPARISON
9774 027602 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
9775 ;DATA WITH DATA READ FROM
9776 ;RHCS2
9777 027610 001401 BEQ 65$ ;BRANCH IF GOOD
9778 027612 104114 ERROR 114
9779 ;AFTER SETTING "CLR" BIT #5
9780 ;IN RHCS2 TO INIT THE RH
9781 ;ONE WORD OF ALL ONES IS
9782 ;WRITTEN ON THE DEVICE
9783 ;A WRITE CHECK WAS DONE
9784 ;ON AN EVEN WORD BOUNDARY THEN
9785 ;RHCS2 SHOULD HAVE WCE!OR!IR
9786 ;=40300
9787 ;TOGETHER WITH UNIT NUMBER
9788 ;BUT CONTAINED WHAT IS
9789 ;GIVEN IN BAD RHCS2
9790 027614 65$:
9791
9792 ;CHECK THAT RHCS3 HAS DBL!WCEEW
9793 027614 012737 006000 001124 MOV #DBL!WCEEW,$GDDAT ;GET GOOD = 6000
9794
9795 027622 017737 154304 001126 MOV DRHCS3,$BDDAT ;READ RHCS3 FOR COMPARISON
9796 027630 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
9797 ;DATA WITH DATA READ FROM
9798 ;RHCS3
9799 027636 001401 BEQ 67$ ;BRANCH IF GOOD
9800 027640 104115 ERROR 115
9801 ;AFTER SETTING "CLR" BIT #5
9802 ;IN RHCS2 TO INIT THE RH
9803 ;ONE WORD OF ALL ONES IS
9804 ;WRITTEN ON THE DEVICE
9805 ;A WRITE CHECK WAS DONE
9806 ;ON AN EVEN WORD BOUNDARY THEN
9807 ;RHCS3 SHOULD HAVE DBL!WCEEW
9808 ;=6000
9809 ;BUT CONTAINED WHAT IS
9810 ;GIVEN IN BAD RHCS3

```

```

9811 027642          67$:
9812
9813
9814 027642 004737 040322      JSR      PC, @#CLDISK      ;GIVE RH INITIALIZE
9815                                ;SETUP UNIT NUMBER
9816                                ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
9817
9818
9819                                ;CHECK THAT RHCS1 HAS RDY!DVA
9820 027646 012737 004200 001124  MOV      #RDY!DVA, $GDDAT ;GET GOOD = 4200
9821
9822 027654 017737 154200 001126  MOV      @RHCS1, $BDDAT    ;READ RHCS1 FOR COMPARISON
9823 027662 023737 001124 001126  CMP      $GDDAT, $BDDAT    ;COMPARE EXPECTED
9824                                ;DATA WITH DATA READ FROM
9825                                ;RHCS1
9826 027670 001401          BEQ      69$              ;BRANCH IF GOOD
9827 027672 104116          ERROR    116
9828
9829                                ;AFTER SETTING "CLR" BIT #5
9830                                ;IN RHCS2 TO INIT THE RH
9831                                ;ONE WORD OF ALL ONES IS
9832                                ;WRITTEN ON THE DEVICE
9833                                ;A WRITE CHECK WAS DONE
9834                                ;ON AN EVEN WORD BOUNDARY THEN
9835                                ;AN RH CLEAR (RHCS2 BIT #5)
9836                                ;WAS GIVEN THIS SHOULD
9837                                ;CLEAR ALL ERRORS
9838                                ;RHCS1 SHOULD HAVE RDY!DVA
9839                                ;=4200
9840                                ;BUT CONTAINED WHAT IS
9841                                ;GIVEN IN BAD RHCS1
9841 027674          69$:
9842                                ;CHECK THAT RHCS2 HAS IR
9843 027674 012737 000100 001124  MOV      #IR, $GDDAT      ;GET GOOD = 100
9844 027702 053737 005010 001124  BIS      UNIT, $GDDAT     ;INCLUDE UNIT NUMBER
9845
9846 027710 017737 154154 001126  MOV      @RHCS2, $BDDAT   ;READ RHCS2 FOR COMPARISON
9847 027716 023737 001124 001126  CMP      $GDDAT, $BDDAT   ;COMPARE EXPECTED
9848                                ;DATA WITH DATA READ FROM
9849                                ;RHCS2
9850 027724 001401          BEQ      71$              ;BRANCH IF GOOD
9851 027726 104117          ERROR    117
9852
9853                                ;AFTER SETTING "CLR" BIT #5
9854                                ;IN RHCS2 TO INIT THE RH
9855                                ;ONE WORD OF ALL ONES IS
9856                                ;WRITTEN ON THE DEVICE
9857                                ;A WRITE CHECK WAS DONE
9858                                ;ON AN EVEN WORD BOUNDARY THEN
9859                                ;AN RH CLEAR (RHCS2 BIT #5)
9860                                ;WAS GIVEN THIS SHOULD
9861                                ;CLEAR ALL ERRORS
9862                                ;RHCS2 SHOULD HAVE IR
9863                                ;=100
9864                                ;TOGETHER WITH UNIT NUMBER
9865                                ;BUT CONTAINED WHAT IS
9866 027730          71$:
9866                                ;GIVEN IN BAD RHCS2

```



```

9923 ;WRITTEN ON THE DEVICE
9924 ;A WRITE CHECK WAS DONE
9925 ;ON AN EVEN WORD BOUNDARY THEN
9926 ;AN RH CLEAR (RHCS2 BIT #5)
9927 ;WAS GIVEN THIS SHOULD
9928 ;CLEAR ALL ERRORS
9929 ;RHBAE SHOULD HAVE 0
9930 ;BUT CONTAINED WHAT IS
9931 ;GIVEN IN BAD RHBAE
9932 030032 79$:
9933 ;CHECK THAT RHCW HAS 0
9934 030032 012737 000000 001124 MOV #0,$GDDAT ;GET GOOD = 0
9935
9936 030040 017737 154016 001126 MOV @RHCW,$BDDAT ;READ RHCW FOR COMPARISON
9937 030046 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
9938 ;DATA WITH DATA READ FROM
9939 ;RHCW
9940 030054 001401 BEQ 79$ ;BRANCH IF GOOD
9941 030056 104123 ERROR 123
9942 ;AFTER SETTING "CLR" BIT #5
9943 ;IN RHCS2 TO INIT THE RH
9944 ;ONE WORD OF ALL ONES IS
9945 ;WRITTEN ON THE DEVICE
9946 ;A WRITE CHECK WAS DONE
9947 ;ON AN EVEN WORD BOUNDARY THEN
9948 ;AN RH CLEAR (RHCS2 BIT #5)
9949 ;WAS GIVEN THIS SHOULD
9950 ;CLEAR ALL ERRORS
9951 ;RHCW SHOULD HAVE 0
9952 ;BUT CONTAINED WHAT IS
9953 ;GIVEN IN BAD RHCW
9954 030060 79$:
9955
9956
9957
9958
9959 ;*****
9960 ;*TEST 40 TEST DBL (RHCS3 BIT #10) TEST A
9961
9962 ;* CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
9963 ;* SET UP FOR A NINE WORD WRITE FROM AN EVEN WORD BOUNDARY
9964 ;* DO A NINE WORD WRITE FROM AN EVEN WORD BOUNDARY
9965 ;* THIS SHOULD NOT SET RHCS3 BIT #10 DBL
9966 ;* CHECK RHCS3
9967 ;* CHECK RHCS1,RHCS2,RHBA,RHBAE,RHCW
9968
9969 ;*****
9970 030060 000004 †ST40: SCOPE
9971 030062 012706 001000 MOV #STACK,SP ;RESET STACK
9972 030066 012737 000040 004656 MOV #40,@†STNM ;SAVE TEST NUMBER
9973
9974 030074 004737 040322 JSR PC,@#CLDISK ;GIVE RH INITIALIZE
9975 ;SETUP UNIT NUMBER
9976 ;CLEAR RHCW AND FUNCTION BITS IN RHCS1
9977
9978 ;SET UP FOR A NINE WORD WRITE FROM AN EVEN WORD BOUNDARY

```

```

9979 030100 012701 00C003      MOV      #3,R1      ;COUNT OF THREE
9980 030104 012702 003000      MOV      #BFEVEN,R2 ;START THREE WORD BUFFER
9981                                     ;FROM AN EVEN WORD BOUNDARY
9982 030110 012722 052525      1$:  MOV      #52525,(R2)+ ;MOVE THREE 52525 INTO BUFFER
9983 030114 005301             DEC      R1          ;COUNT
9984 030116 001374             BNE     1$          ;BRANCH IF THREE NOT DONE
9985 030120 012777 003000 153736 MOV      #BFEVEN,DRHBA ;SET BUS ADDRESS TO START
9986                                     ;FROM AN EVEN WORD BOUNDARY
9987 030126 012777 177767 153726 MOV #-9,DRHWC ;WORD COUNT NINE
9988 030134 004077 154500      JSR     RD,DCOMND   ;GO TO DO COMMAND
9989 030140 004154             WRIDAT             ;WRITE DATA
9990
9991
9992                                     ;CHECK THAT RHCS3 HAS 0
9993 030142 012737 000000 001124 MOV      #0,$GDDAT  ;GET GOOD = 0
9994
9995 030150 017737 153756 001126 MOV      DRHCS3,$BDDAT ;READ RHCS3 FOR COMPARISON
9996 030156 023737 001124 001126 CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED
9997                                     ;DATA WITH DATA READ FROM
9998                                     ;RHCS3
9999 030164 001401             BEQ     65$        ;BRANCH IF GOOD
10000 030166 104124             ERROR   124
10001                                     ;AFTER SETTING "CLR" BIT #5
10002                                     ;IN RHCS2 TO INIT THE RH
10003                                     ;A NINE WORD WRITE FROM AN
10004                                     ;EVEN WORD BOUNDARY WAS DONE
10005                                     ;THEN
10006                                     ;RHCS3 SHOULD HAVE 0
10007                                     ;BUT CONTAINED WHAT IS
10008                                     ;GIVEN IN BAD RHCS3
10009 030170             65$:
10010 030170 042777 000076 153662 BIC     #76,DRHCS1 ;CLEAR FUNCTION BITS
10011                                     ;CHECK THAT RHCS1 HAS RDY!DVA
10012 030176 012737 004200 001124 MOV      #RDY!DVA,$GDDAT ;GET GOOD = 4200
10013
10014 030204 017737 153650 001126 MOV      DRHCS1,$BDDAT ;READ RHCS1 FOR COMPARISON
10015 030212 023737 001124 001126 CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED
10016                                     ;DATA WITH DATA READ FROM
10017                                     ;RHCS1
10018 030220 001401             BEQ     67$        ;BRANCH IF GOOD
10019 030222 104125             ERROR   125
10020                                     ;AFTER SETTING "CLR" BIT #5
10021                                     ;IN RHCS2 TO INIT THE RH
10022                                     ;A NINE WORD WRITE FROM AN
10023                                     ;EVEN WORD BOUNDARY WAS DONE
10024                                     ;THEN
10025                                     ;RHCS1 SHOULD HAVE RDY!DVA
10026                                     ;=4200
10027                                     ;BUT CONTAINED WHAT IS
10028                                     ;GIVEN IN BAD RHCS1
10029 030224             67$:
10030                                     ;CHECK THAT RHCS2 HAS IR
10031 030224 012737 000100 001124 MOV      #IR,$GDDAT  ;GET GOOD = 100
10032 030232 053737 005010 001124 BIS     UNIT,$GDDAT  ;INCLUDE UNIT NUMBER
10033
10034 030240 017737 153624 001126 MOV      DRHCS2,$BDDAT ;READ RHCS2 FOR COMPARISON

```

```

10035 030246 023737 001124 001126      CMP      $GDDAT,$BDDAT      ;COMPARE EXPECTED
10036                                     ;DATA WITH DATA READ FROM
10037                                     ;RHCS2
10038 030254 001401      BEQ      69$                ;BRANCH IF GOOD
10039 030256 104126      ERROR    126
10040                                     ;AFTER SETTING "CLR" BIT #5
10041                                     ;IN RHCS2 TO INIT THE RH
10042                                     ;A NINE WORD WRITE FROM AN
10043                                     ;EVEN WORD BOUNDARY WAS DONE
10044                                     ;THEN
10045                                     ;RHCS2 SHOULD HAVE IR
10046                                     ;=100
10047                                     ;TOGETHER WITH UNIT NUMBER
10048                                     ;BUT CONTAINED WHAT IS
10049                                     ;GIVEN IN BAD RHCS2
10050 030260                                     69$:
10051                                     ;CHECK THAT RHWC HAS 0
10052 030260 012737 000000 001124      MOV      #0,$GDDAT          ;GET GOOD = 0
10053
10054 030266 017737 153570 001126      MOV      @RHWC,$BDDAT      ;READ RHWC FOR COMPARISON
10055 030274 023737 001124 001126      CMP      $GDDAT,$BDDAT      ;COMPARE EXPECTED
10056                                     ;DATA WITH DATA READ FROM
10057                                     ;RHWC
10058 030302 001401      BEQ      71$                ;BRANCH IF GOOD
10059 030304 104131      ERROR    131
10060                                     ;AFTER SETTING "CLR" BIT #5
10061                                     ;IN RHCS2 TO INIT THE RH
10062                                     ;A NINE WORD WRITE FROM AN
10063                                     ;EVEN WORD BOUNDARY WAS DONE
10064                                     ;THEN
10065                                     ;RHWC SHOULD HAVE 0
10066                                     ;BUT CONTAINED WHAT IS
10067                                     ;GIVEN IN BAD RHWC
10068 030306                                     71$:
10069
10070
10071
10072                                     ;*****
10073                                     ;*TEST 41      TEST DBL (RHCS3 BIT #10) TEST B
10074
10075                                     ;*      CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
10076                                     ;*      SET UP FOR A TEN WORD WRITE FROM AN EVEN WORD BOUNDARY
10077                                     ;*      DO A TEN WORD WRITE FROM AN EVEN WORD BOUNDARY
10078                                     ;*      THIS SHOULD SET RHCS3 BIT #10 DBL
10079                                     ;*      CHECK RHCS3
10080                                     ;*      CHECK RHCS1,RHCS2,RHBA,RHBAE,RHWC
10081
10082                                     ;*****
10083 030306 000004      †ST41: SCOPE
10084 030310 012706 001000      MOV      #STACK,SP          ;RESET STACK
10085 030314 012737 000041 004656      MOV      #41,@†STNM         ;SAVE TEST NUMBER
10086
10087 030322 004737 040322      JSR      PC,@CLDISK         ;GIVE RH INITIALIZE
10088                                     ;SETUP UNIT NUMBER
10089                                     ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
10090
10091
10092
10093
10094

```

```

10091 ;SET UP FOR A TEN WORD WRITE FROM AN EVEN WORD BOUNDARY
10092 030326 012701 000003 MOV #3,R1 ;COUNT OF THREE
10093 030332 012702 003000 MOV #BFEVEN,R2 ;START THREE WORD BUFFER
10094 ;FROM AN EVEN WORD BOUNDARY
10095 030336 012722 052525 1$: MOV #52525,(R2)+ ;MOVE THREE 52525 INTO BUFFER
10096 030342 005301 DEC R1 ;COUNT
10097 030344 001374 BNE 1$ ;BRANCH IF THREE NOT DONE
10098 030346 012777 003000 153510 MOV #BFEVEN,ARHBA ;SET BUS ADDRESS TO START
10099 ;FROM AN EVEN WORD BOUNDARY
10100 030354 012777 177766 153500 MOV #-10,ARHWC ;WORD COUNT TEN
10101 030362 004077 154252 JSR R0,ACOMND ;GO TO DO COMMAND
10102 030366 004154 WRIDAT ;WRITE DATA
10103
10104
10105 ;CHECK THAT RHCS3 HAS DBL
10106 030370 012737 002000 001124 MOV #DBL,$GDDAT ;GET GOOD = 2000
10107
10108 030376 017737 153530 001126 MOV ARHCS3,$BDDAT ;READ RHCS3 FOR COMPARISON
10109 030404 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
10110 ;DATA WITH DATA READ FROM
10111 ;RHCS3
10112 030412 001401 BEQ 65$ ;BRANCH IF GOOD
10113 030414 104124 ERROR 124
10114
10115 ;AFTER SETTING "CLR" BIT #5
10116 ;IN RHCS2 TO INIT THE RH
10117 ;A TEN WORD WRITE FROM AN
10118 ;EVEN WORD BOUNDARY WAS DONE
10119 ;THEN
10120 ;RHCS3 SHOULD HAVE DBL
10121 ;=2000
10122 ;BUT CONTAINED WHAT IS
10123 ;GIVEN IN BAD RHCS3
10123 030416 042777 000076 153434 65$: BIC #76,ARHCS1 ;CLEAR FUNCTION BITS
10124 030416 ;CHECK THAT RHCS1 HAS RDY!DVA
10125 MOV #RDY!DVA,$GDDAT ;GET GOOD = 4200
10126 030424 012737 004200 001124
10127
10128 030432 017737 153422 001126 MOV ARHCS1,$BDDAT ;READ RHCS1 FOR COMPARISON
10129 030440 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
10130 ;DATA WITH DATA READ FROM
10131 ;RHCS1
10132 030446 001401 BEQ 67$ ;BRANCH IF GOOD
10133 030450 104125 ERROR 125
10134
10135 ;AFTER SETTING "CLR" BIT #5
10136 ;IN RHCS2 TO INIT THE RH
10137 ;A TEN WORD WRITE FROM AN
10138 ;EVEN WORD BOUNDARY WAS DONE
10139 ;THEN
10140 ;RHCS1 SHOULD HAVE RDY!DVA
10141 ;=4200
10142 ;BUT CONTAINED WHAT IS
10143 ;GIVEN IN BAD RHCS1
10143 030452 67$:
10144 ;CHECK THAT RHCS2 HAS IR
10145 030452 012737 000100 001124 MOV #IR,$GDDAT ;GET GOOD = 100
10146 030460 053737 005010 001124 BIS UNIT,$GDDAT ;INCLUDE UNIT NUMBER

```





:CLEAR RHWC AND FUNCTION BITS IN RHCS1

```

10203
10204
10205 ;SET UP FOR A TEN WORD WRITE FROM AN ODD WORD BOUNDARY
10206 030554 012701 000003 MOV #3,R1 ;COUNT OF THREE
10207 030560 012702 003002 MOV #BF0DD,R2 ;START THREE WORD BUFFER
10208 ;FROM AN ODD WORD BOUNDARY
10209 030564 012722 052525 1$: MOV #52525,(R2)+ ;MOVE THREE 52525 INTO BUFFER
10210 030570 005301 DEC R1 ;COUNT
10211 030572 001374 BNE 1$ ;BRANCH IF THREE NOT DONE
10212 030574 012777 003002 153262 MOV #BF0DD,DRHBA ;SET BUS ADDRESS TO START
10213 ;FROM AN ODD WORD BOUNDARY
10214 030602 012777 177766 153252 MOV #-10,DRHWC ;WORD COUNT TEN
10215 030610 004077 154024 JSR RD,DCOMND ;GO TO DC COMMAND
10216 030614 004154 WRIDAT ;WRITE DATA
10217
10218
10219 ;CHECK THAT RHCS3 HAS 0
10220 030616 012737 000000 001124 MOV #0,$GDDAT ;GET GOOD = 0
10221
10222 030624 017737 153302 001126 MOV DRHCS3,$BDDAT ;READ RHCS3 FOR COMPARISON
10223 030632 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
10224 ;DATA WITH DATA READ FROM
10225 ;RHCS3
10226 030640 001401 BEG 65$ ;BRANCH IF GOOD
10227 030642 104124 ERROR 124
10228
10229 ;AFTER SETTING "CLR" BIT #5
10230 ;IN RHCS2 TO INIT THE RH
10231 ;A TEN WORD WRITE FROM AN
10232 ;ODD WORD BOUNDARY WAS DONE
10233 ;THEN
10234 ;RHCS3 SHOULD HAVE 0
10235 ;BUT CONTAINED WHAT IS
10236 ;GIVEN IN BAD RHCS3
10236 030644 042777 000076 153206 65$: BIC #76,DRHCS1 ;CLEAR FUNCTION BITS
10237 030644 ;CHECK THAT RHCS1 HAS RDY!DVA
10238 MOV #RDY!DVA,$GDDAT ;GET GOOD = 4200
10239 030652 012737 004200 001124
10240
10241 030660 017737 153174 001126 MOV DRHCS1,$BDDAT ;READ RHCS1 FOR COMPARISON
10242 030666 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
10243 ;DATA WITH DATA READ FROM
10244 ;RHCS1
10245 030674 001401 BEG 67$ ;BRANCH IF GOOD
10246 030676 104125 ERROR 125
10247
10248 ;AFTER SETTING "CLR" BIT #5
10249 ;IN RHCS2 TO INIT THE RH
10250 ;A TEN WORD WRITE FROM AN
10251 ;ODD WORD BOUNDARY WAS DONE
10252 ;THEN
10253 ;RHCS1 SHOULD HAVE RDY!DVA
10254 ;=4200
10255 ;BUT CONTAINED WHAT IS
10256 ;GIVEN IN BAD RHCS1
10256 030700 012737 000100 001124 67$: ;CHECK THAT RHCS2 HAS IR
10257 MOV #IR,$GDDAT ;GET GOOD = 100
10258 030700

```

```

10259 030706 053737 005010 001124      BIS      UNIT,$GDDAT      ;INCLUDE UNIT NUMBER
10260
10261 030714 017737 153150 001126      MOV      2RHCS2,$BDDAT   ;READ RHCS2 FOR COMPARISON
10262 030722 023737 001124 001126      CMP      $GDDAT,$BDDAT  ;COMPARE EXPECTED
10263                                     ;DATA WITH DATA READ FROM
10264                                     ;RHCS2
10265 030730 001401      BEQ      69$             ;BRANCH IF GOOD
10266 030732 104126      ERROR   126
10267                                     ;AFTER SETTING "CLR" BIT #5
10268                                     ;IN RHCS2 TO INIT THE RH
10269                                     ;A TEN WORD WRITE FROM AN
10270                                     ;ODD WORD BOUNDARY WAS DONE
10271                                     ;THEN
10272                                     ;RHCS2 SHOULD HAVE IR
10273                                     ;=100
10274                                     ;TOGETHER WITH UNIT NUMBER
10275                                     ;BUT CONTAINED WHAT IS
10276                                     ;GIVEN IN BAD RHCS2
10277 030734      69$:
10278                                     ;CHECK THAT RHWC HAS 0
10279 030734 012737 000000 001124      MOV      #0,$GDDAT      ;GET GOOD = 0
10280
10281 030742 017737 153114 001126      MOV      2RHWC,$BDDAT   ;READ RHWC FOR COMPARISON
10282 030750 023737 001124 001126      CMP      $GDDAT,$BDDAT  ;COMPARE EXPECTED
10283                                     ;DATA WITH DATA READ FROM
10284                                     ;RHWC
10285 030756 001401      BEQ      71$             ;BRANCH IF GOOD
10286 030760 104131      ERROR   131
10287                                     ;AFTER SETTING "CLR" BIT #5
10288                                     ;IN RHCS2 TO INIT THE RH
10289                                     ;A TEN WORD WRITE FROM AN
10290                                     ;ODD WORD BOUNDARY WAS DONE
10291                                     ;THEN
10292                                     ;RHWC SHOULD HAVE 0
10293                                     ;BUT CONTAINED WHAT IS
10294                                     ;GIVEN IN BAD RHWC
10295 030762      71$:
10296
10297
10298
10299
10300      ;*****
10301      ;*TEST 43      TEST DBL (RHCS3 BIT #10) TEST D
10302
10303      ;*      CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
10304      ;*      SET UP FOR A ELEVEN WORD WRITE FROM AN EVEN WORD BOUNDARY
10305      ;*      DO A ELEVEN WORD WRITE FROM AN EVEN WORD BOUNDARY
10306      ;*      THIS SHOULD NOT SET RHCS3 BIT #10 DBL
10307      ;*      CHECK RHCS3
10308      ;*      CHECK RHCS1,RHCS2,RHBA,RHBAE,RHWC
10309
10310      ;*****
10310 030762 000004      TEST43: SCOPE
10311 030764 012706 001000      MOV      #STACK,$P      ;RESET STACK
10312 030770 012737 000043 004656      MOV      #43,2#TSTNM    ;SAVE TEST NUMBER
10313
10314 030776 004737 040322      JSR      PC,2#CLDISK    ;GIVE RH INITIALIZE

```

```

10315                                     ;SETUP UNIT NUBER
10316                                     ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
10317
10318                                     ;SET UP FOR A ELEVEN WORD WRITE FROM AN EVEN WORD BOUNDARY
10319 031002 012701 000003 MOV #3,R1 ;COUNT OF THREE
10320 031006 012702 003000 MOV #BFEVEN,R2 ;START THREE WORD BUFFER
10321                                     ;FROM AN EVEN WORD BOUNDARY
10322 031012 012722 052525 1$: MOV #52525,(R2)+ ;MOVE THREE 52525 INTO BUFFER
10323 031016 005301 DEC R1 ;COUNT
10324 031020 001374 BNE 1$ ;BRANCH IF THREE NOT DONE
10325 031022 012777 003000 153034 MOV #BFEVEN,ARHBA ;SET BUS ADDRESS TO START
10326                                     ;FROM AN EVEN WORD BOUNDARY
10327 031030 012777 177765 153024 MOV #-11,ARHWC ;WORD COUNT ELEVEN
10328 031036 004077 153576 JSR R0,ACOMND ;GO TO DO COMMAND
10329 031042 004154 WRIDAT ;WRITE DATA
10330
10331
10332                                     ;CHECK THAT RHCS3 HAS 0
10333 031044 012737 000000 001124 MOV #0,$GDDAT ;GET GOOD = 0
10334
10335 031052 017737 153054 001126 MOV ARHCS3,$BDDAT ;READ RHCS3 FOR COMPARISON
10336 031060 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
10337                                     ;DATA WITH DATA READ FROM
10338                                     ;RHCS3
10339 031066 001401 BEQ 65$ ;BRANCH IF GOOD
10340 031070 104124 ERROR 124
10341
10342                                     ;AFTER SETTING "CLR" BIT #5
10343                                     ;IN RHCS2 TO INIT THE RH
10344                                     ;A ELEVEN WORD WRITE FROM AN
10345                                     ;EVEN WORD BOUNDARY WAS DONE
10346                                     ;THEN
10347                                     ;RHCS3 SHOULD HAVE 0
10348                                     ;BUT CONTAINED WHAT IS
10349                                     ;GIVEN IN BAD RHCS3
10349 031072 65$: BIC #76,ARHCS1 ;CLEAR FUNCTION BITS
10350 031072 042777 000076 152760 ;CHECK THAT RHCS1 HAS RDY!DVA
10351
10352 031100 012737 004200 001124 MOV #RDY!DVA,$GDDAT ;GET GOOD = 4200
10353
10354 031106 017737 152746 001126 MOV ARHCS1,$BDDAT ;READ RHCS1 FOR COMPARISON
10355 031114 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
10356                                     ;DATA WITH DATA READ FROM
10357                                     ;RHCS1
10358 031122 001401 BEQ 67$ ;BRANCH IF GOOD
10359 031124 104125 ERROR 125
10360
10361                                     ;AFTER SETTING "CLR" BIT #5
10362                                     ;IN RHCS2 TO INIT THE RH
10363                                     ;A ELEVEN WORD WRITE FROM AN
10364                                     ;EVEN WORD BOUNDARY WAS DONE
10365                                     ;THEN
10366                                     ;RHCS1 SHOULD HAVE RDY!DVA
10367                                     ;=4200
10368                                     ;BUT CONTAINED WHAT IS
10369                                     ;GIVEN IN BAD RHCS1
10369 031126 67$: ;CHECK THAT RHCS2 HAS IR
10370

```

```

10371 031126 012737 000100 001124      MOV      #IR,$GDDAT      ;GET GOOD = 100
10372 031134 053737 005010 001124      BIS      UNIT,$GDDAT    ;INCLUDE UNIT NUMBER
10373
10374 031142 017737 152722 001126      MOV      @RHCS2,$BDDAT  ;READ RHCS2 FOR COMPARISON
10375 031150 023737 001124 001126      CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED
10376                                     ;DATA WITH DATA READ FROM
10377                                     ;RHCS2
10378 031156 001401      BEQ      69$           ;BRANCH IF GOOD
10379 031160 104125      ERROR   126
10380                                     ;AFTER SETTING "CLR" BIT #5
10381                                     ;IN RHCS2 TO INIT THE RH
10382                                     ;A ELEVEN WORD WRITE FROM AN
10383                                     ;EVEN WORD BOUNDARY WAS DONE
10384                                     ;THEN
10385                                     ;RHCS2 SHOULD HAVE IR
10386                                     ;=100
10387                                     ;TOGETHER WITH UNIT NUMBER
10388                                     ;BUT CONTAINED WHAT IS
10389                                     ;GIVEN IN BAD RHCS2
10390 031162                                     69$:
10391                                     ;CHECK THAT RHWC HAS 0
10392 031162 012737 000000 001124      MOV      #0,$GDDAT     ;GET GOOD = 0
10393
10394 031170 017737 152666 001126      MOV      @RHWC,$BDDAT  ;READ RHWC FOR COMPARISON
10395 031176 023737 001124 001126      CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED
10396                                     ;DATA WITH DATA READ FROM
10397                                     ;RHWC
10398 031204 001401      BEQ      71$           ;BRANCH IF GOOD
10399 031206 104131      ERROR   131
10400                                     ;AFTER SETTING "CLR" BIT #5
10401                                     ;IN RHCS2 TO INIT THE RH
10402                                     ;A ELEVEN WORD WRITE FROM AN
10403                                     ;EVEN WORD BOUNDARY WAS DONE
10404                                     ;THEN
10405                                     ;RHWC SHOULD HAVE 0
10406                                     ;BUT CONTAINED WHAT IS
10407                                     ;GIVEN IN BAD RHWC
10408 031210                                     71$:
10409
10410
10411
10412                                     ;*****
10413                                     ;*TEST 44      TEST DBL (RHCS3 BIT #10) TEST E
10414
10415                                     ;*
10416                                     ;*      CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
10417                                     ;*      SET UP FOR A TEN WORD WRITE FROM AN EVEN WORD BOUNDARY
10418                                     ;*      WITH BAI IN RHCS2 BIT #3 SET
10419                                     ;*      DO A TEN WORD WRITE FROM AN EVEN WORD BOUNDARY
10420                                     ;*      THIS SHOULD NOT SET RHCS3 BIT #10 DBL
10421                                     ;*      CHECK RHCS3
10422                                     ;*      CHECK RHCS1,RHCS2,RHBA,RHBAE,RHWC
10423                                     ;*****
10424 031210 000004      †ST44: SCOPE
10425 031212 012706 001000      MOV      #STACK,SP     ;RESET STACK
10426 031216 012737 000044 004656      MOV      #44,@†STNM    ;SAVE TEST NUMBER

```

```

10427
10428 031224 004737 040322 JSR PC, @CLDISK ;GIVE RH INITIALIZE
10429 ;SETUP UNIT NUBER
10430 ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
10431
10432 ;SET UP FOR A TEN WORD WRITE FROM AN EVEN WORD BOUNDARY
10433 031230 012701 000003 MOV #3, R1 ;COUNT OF THREE
10434 031234 012702 003000 MOV #BF EVEN, R2 ;START THREE WORD BUFFER
10435 ;FROM AN EVEN WORD BOUNDARY
10436 031240 012722 052525 1$: MOV #52525, (R2)+ ;MOVE THREE 52525 INTO BUFFER
10437 031244 005301 DEC R1 ;COUNT
10438 031246 001374 BNE 1$ ;BRANCH IF THREE NOT DONE
10439 031250 012777 003000 152606 MOV #BF EVEN, @RHBA ;SET BUS ADDRESS TO START
10440 ;FROM AN EVEN WORD BOUNDARY
10441 031256 012777 177766 152576 MOV #-10, @RHWC ;WORD COUNT TEN
10442 031264 052777 000010 152576 BIS #BAI, @RHCS2 ;SET BAI IN RHCS2
10443 031272 004077 .53342 JSR @COMND ;GO TO DO COMMAND
10444 031276 004154 WRIDAT ;WRITE DATA
10445
10446
10447 ;CHECK THAT RHCS3 HAS 0
10448 031300 012737 000000 001124 MOV #0, $GDDAT ;GET GOOD = 0
10449
10450 031306 017737 152620 001126 MOV @RHCS3, $BDDAT ;READ RHCS3 FOR COMPARISON
10451 031314 023737 001124 001126 CMP $GDDAT, $BDDAT ;COMPARE EXPECTED
10452 ;DATA WITH DATA READ FROM
10453 ;RHCS3
10454 031322 001401 BEQ 65$ ;BRANCH IF GOOD
10455 031324 104124 ERROR 124
10456 ;AFTER SETTING "CLR" BIT #5
10457 ;IN RHCS2 TO INIT THE RH
10458 ;A TEN WORD WRITE FROM AN
10459 ;EVEN WORD BOUNDARY WAS DONE
10460 ;WITH BAI IN RHCS2 SET
10461 ;THEN
10462 ;RHCS3 SHOULD HAVE 0
10463 ;BUT CONTAINED WHAT IS
10464 ;GIVEN IN BAD RHCS3
10465 031326 65$:
10466 031326 042777 000076 152524 BIC #76, @RHCS1 ;CLEAR FUNCTION BITS
10467 ;CHECK THAT RHCS1 HAS RDY!DVA
10468 031334 012737 004200 001124 MOV #RDY!DVA, $GDDAT ;GET GOOD = 4200
10469
10470 031342 017737 152512 001126 MOV @RHCS1, $BDDAT ;READ RHCS1 FOR COMPARISON
10471 031350 023737 001124 001126 CMP $GDDAT, $BDDAT ;COMPARE EXPECTED
10472 ;DATA WITH DATA READ FROM
10473 ;RHCS1
10474 031356 001401 BEQ 67$ ;BRANCH IF GOOD
10475 031360 104125 ERROR 125
10476 ;AFTER SETTING "CLR" BIT #5
10477 ;IN RHCS2 TO INIT THE RH
10478 ;A TEN WORD WRITE FROM AN
10479 ;EVEN WORD BOUNDARY WAS DONE
10480 ;WITH BAI IN RHCS2 SET
10481 ;THEN
10482 ;RHCS1 SHOULD HAVE RDY!DVA

```

```

10483                                     ;=4200
10484                                     ;BUT CONTAINED WHAT IS
10485                                     ;GIVEN IN BAD RHCS1
10486 031362                               67$:
10487                                     ;CHECK THAT RHCS2 HAS IR!BAI
10488 031362 012737 000110 001124        MOV   #IR!BAI,$GDDAT ;GET GOOD = 110
10489 031370 053737 005010 001124        BIS   UNIT,$GDDAT   ;INCLUDE UNIT NUMBER
10490
10491 031376 017737 152466 001126        MOV   @RHCS2,$BDDAT ;READ RHCS2 FOR COMPARISON
10492 031404 023737 001124 001126        CMP   $GDDAT,$BDDAT ;COMPARE EXPECTED
10493                                     ;DATA WITH DATA READ FROM
10494                                     ;RHCS2
10495 031412 001401                          BEQ   69$           ;BRANCH IF GOOD
10496 031414 104126                          ERROR 126
10497
10498                                     ;AFTER SETTING "CLR" BIT #5
10499                                     ;IN RHCS2 TO INIT THE RH
10500                                     ;A TEN WORD WRITE FROM AN
10501                                     ;EVEN WORD BOUNDARY WAS DONE
10502                                     ;WITH BAI IN RHCS2 SET
10503                                     ;THEN
10504                                     ;RHCS2 SHOULD HAVE IR!BAI
10505                                     ;=110
10506                                     ;TOGETHER WITH UNIT NUMBER
10507                                     ;BUT CONTAINED WHAT IS
10508                                     ;GIVEN IN BAD RHCS2
10508 031416                               69$:
10509                                     ;CHECK THAT RHWC HAS 0
10510 031416 012737 000000 001124        MOV   #0,$GDDAT   ;GET GOOD = 0
10511
10512 031424 017737 152432 001126        MOV   @RHWC,$BDDAT ;READ RHWC FOR COMPARISON
10513 031432 023737 001124 001126        CMP   $GDDAT,$BDDAT ;COMPARE EXPECTED
10514                                     ;DATA WITH DATA READ FROM
10515                                     ;RHWC
10516 031440 001401                          BEQ   71$           ;BRANCH IF GOOD
10517 031442 104131                          ERROR 131
10518
10519                                     ;AFTER SETTING "CLR" BIT #5
10520                                     ;IN RHCS2 TO INIT THE RH
10521                                     ;A TEN WORD WRITE FROM AN
10522                                     ;EVEN WORD BOUNDARY WAS DONE
10523                                     ;WITH BAI IN RHCS2 SET
10524                                     ;THEN
10525                                     ;RHWC SHOULD HAVE 0
10526                                     ;BUT CONTAINED WHAT IS
10527 031444                               71$:
10528
10529
10530
10531                                     ;*****
10532                                     ;*TEST 45      TEST DBL (RHCS3 BIT #10) TEST F
10533
10534                                     ;*
10535                                     ;* CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
10536                                     ;* SET UP FOR A ELEVEN WORD WRITE FROM AN ODD WORD BOUNDARY
10537                                     ;* DO A ELEVEN WORD WRITE FROM AN ODD WORD BOUNDARY
10538                                     ;* THIS SHOULD SET RHCS3 BIT #10 DBL
10539                                     ;* CHECK RHCS3

```



```

10595 ;A ELEVEN WORD WRITE FROM AN
10596 ;ODD WORD BOUNDARY WAS DONE
10597 ;THEN
10598 ;RHCS1 SHOULD HAVE RDY!DVA
10599 ;=4200
10600 ;BUT CONTAINED WHAT IS
10601 ;GIVEN IN BAD RHCS1
10602 031610 67$:
10603 ;CHECK THAT RHCS2 HAS IR
10604 031610 012737 000100 001124 MOV #IR,$GDDAT ;GET GOOD = 100
10605 031616 053737 005010 001124 BIS UNIT,$GDDAT ;INCLUDE UNIT NUMBER
10606
10607 031624 017737 152240 001126 MOV @RHCS2,$BDDAT ;READ RHCS2 FOR COMPARISON
10608 031632 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
10609 ;DATA WITH DATA READ FROM
10610 ;RHCS2
10611 031640 001401 BEQ 69$ ;BRANCH IF GOOD
10612 031642 104126 ERROR 126
10613 ;AFTER SETTING "CLR" BIT #5
10614 ;IN RHCS2 TO INIT THE RH
10615 ;A ELEVEN WORD WRITE FROM AN
10616 ;ODD WORD BOUNDARY WAS DONE
10617 ;THEN
10618 ;RHCS2 SHOULD HAVE IR
10619 ;=100
10620 ;TOGETHER WITH UNIT NUMBER
10621 ;BUT CONTAINED WHAT IS
10622 ;GIVEN IN BAD RHCS2
10623 031644 69$:
10624 ;CHECK THAT RHWC HAS 0
10625 031644 012737 000000 001124 MOV #0,$GDDAT ;GET GOOD = 0
10626
10627 031652 017737 152204 001126 MOV @RHWC,$BDDAT ;READ RHWC FOR COMPARISON
10628 031660 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
10629 ;DATA WITH DATA READ FROM
10630 ;RHWC
10631 031666 001401 BEQ 71$ ;BRANCH IF GOOD
10632 031670 104131 ERROR 131
10633 ;AFTER SETTING "CLR" BIT #5
10634 ;IN RHCS2 TO INIT THE RH
10635 ;A ELEVEN WORD WRITE FROM AN
10636 ;ODD WORD BOUNDARY WAS DONE
10637 ;THEN
10638 ;RHWC SHOULD HAVE 0
10639 ;BUT CONTAINED WHAT IS
10640 ;GIVEN IN BAD RHWC
10641 031672 71$:
10642
10643
10644
10645
10646 ;*****
10647 ;*TEST 46 TEST DBL (RHCS3 BIT #10) TEST G
10648
10649 ;* CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
10650 ;* SET UP FOR A NINE WORD READ FROM AN EVEN WORD BOUNDARY

```



```

10651      : *      DO A NINE WORD READ FROM AN EVEN WORD BOUNDARY
10652      : *      THIS SHOULD NOT SET RHCS3 BIT #10 DBL
10653      : *      CHECK RHCS3
10654      : *      CHECK RHCS1, RHCS2, RHBA, RHBAE, RHWC
10655
10656      : *****
10657 031672 000004      †ST46: SCOPE
10658 031674 012706      MOV      #STACK, SP      ;RESET STACK
10659 031700 012737 000046 004656      MOV      #46, @†STNM      ;SAVE TEST NUMBER
10660
10661 031706 004737 040322      JSR      PC, @#CLDISK      ;GIVE RH INITIALIZE
10662                                     ;SETUP UNIT NUBER
10663                                     ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
10664
10665      ;SET UP FOR A NINE WORD READ FROM AN EVEN WORD BOUNDARY
10666 031712 012701 000003      MOV      #3, R1      ;COUNT OF THREE
10667 031716 012702 003000      MOV      #BF EVEN, R2      ;START THREE WORD BUFFER
10668                                     ;FROM AN EVEN WORD BOUNDARY
10669 031722 012722 052525      1$: MOV      #52525, (R2)+      ;MOVE THREE 52525 INTO BUFFER
10670 031726 005301      DEC      R1      ;COUNT
10671 031730 001374      BNE      1$      ;BRANCH IF THREE NOT DONE
10672 031732 012777 003000 152124      MOV      #BF EVEN, @RHBA      ;SET BUS ADDRESS TO START
10673                                     ;FROM AN EVEN WORD BOUNDARY
10674 031740 012777 177767 152114      MOV      #-9, @RHWC ;WORD COUNT NINE
10675 031746 004077 152666      JSR      RO, @COMND      ;GO TO DO COMMAND
10676 031752 004160      READAT      ;READ DATA
10677
10678
10679      ;CHECK THAT RHCS3 HAS 0
10680 031754 012737 000000 001124      MOV      #0, $GDDAT      ;GET GOOD = 0
10681
10682 031762 017737 152144 001126      MOV      @RHCS3, $BDDAT      ;READ RHCS3 FOR COMPARISON
10683 031770 023737 001124 001126      CMP      $GDDAT, $BDDAT      ;COMPARE EXPECTED
10684                                     ;DATA WITH DATA READ FROM
10685                                     ;RHCS3
10686 031776 001401      BEQ      65$      ;BRANCH IF GOOD
10687 032000 104124      ERROR      124
10688                                     ;AFTER SETTING "CLR" BIT #5
10689                                     ;IN RHCS2 TO INIT THE RH
10690                                     ;A NINE WORD READ FROM AN
10691                                     ;EVEN WORD BOUNDARY WAS DONE
10692                                     ;THEN
10693                                     ;RHCS3 SHOULD HAVE 0
10694                                     ;BUT CONTAINED WHAT IS
10695                                     ;GIVEN IN BAD RHCS3
10696 032002      65$: BIC      #76, @RHCS1      ;CLEAR FUNCTION BITS
10697 032002 042777 000076 152050      ;CHECK THAT RHCS1 HAS RDY!DVA
10698      MOV      #RDY!DVA, $GDDAT ;GET GOOD = 4200
10699 032010 012737 004200 001124
10700
10701 032016 017737 152036 001126      MOV      @RHCS1, $BDDAT      ;READ RHCS1 FOR COMPARISON
10702 032024 023737 001124 001126      CMP      $GDDAT, $BDDAT      ;COMPARE EXPECTED
10703                                     ;DATA WITH DATA READ FROM
10704                                     ;RHCS1
10705 032032 001401      BEQ      67$      ;BRANCH IF GOOD
10706 032034 104125      ERROR      125

```

```

10707                                     ;AFTER SETTING "CLR" BIT #5
10708                                     ;IN RHCS2 TO INIT THE RH
10709                                     ;A NINE WORD READ FROM AN
10710                                     ;EVEN WORD BOUNDARY WAS DONE
10711                                     ;THEN
10712                                     ;RHCS1 SHOULD HAVE RDY!DVA
10713                                     ;=4200
10714                                     ;BUT CONTAINED WHAT IS
10715                                     ;GIVEN IN BAD RHCS1
10716 032036                               67$:
10717                                     ;CHECK THAT RHCS2 HAS IR
10718 032036 012737 000100 001124          MOV   #IR,$GDDAT   ;GET GOOD = 100
10719 032044 053737 005010 001124          BIS   UNIT,$GDDAT ;INCLUDE UNIT NUMBER
10720
10721 032052 017737 152012 001126          MOV   @RHCS2,$BDDAT ;READ RHCS2 FOR COMPARISON
10722 032060 023737 001124 001126          CMP   $GDDAT,$BDDAT ;COMPARE EXPECTED
10723                                     ;DATA WITH DATA READ FROM
10724                                     ;RHCS2
10725 032066 001401                          BEQ   69$           ;BRANCH IF GOOD
10726 032070 104126
10727
10728                                     ;AFTER SETTING "CLR" BIT #5
10729                                     ;IN RHCS2 TO INIT THE RH
10730                                     ;A NINE WORD READ FROM AN
10731                                     ;EVEN WORD BOUNDARY WAS DONE
10732                                     ;THEN
10733                                     ;RHCS2 SHOULD HAVE IR
10734                                     ;=100
10735                                     ;TOGETHER WITH UNIT NUMBER
10736                                     ;BUT CONTAINED WHAT IS
10737                                     ;GIVEN IN BAD RHCS2
10737 032072                               69$:
10738                                     ;CHECK THAT RHWC HAS 0
10739 032072 012737 000000 001124          MOV   #0,$GDDAT   ;GET GOOD = 0
10740
10741 032100 017737 151756 001126          MOV   @RHWC,$BDDAT ;READ RHWC FOR COMPARISON
10742 032106 023737 001124 001126          CMP   $GDDAT,$BDDAT ;COMPARE EXPECTED
10743                                     ;DATA WITH DATA READ FROM
10744                                     ;RHWC
10745 032114 001401                          BEQ   71$           ;BRANCH IF GOOD
10746 032116 104131
10747
10748                                     ;AFTER SETTING "CLR" BIT #5
10749                                     ;IN RHCS2 TO INIT THE RH
10750                                     ;A NINE WORD READ FROM AN
10751                                     ;EVEN WORD BOUNDARY WAS DONE
10752                                     ;THEN
10753                                     ;RHWC SHOULD HAVE 0
10754                                     ;BUT CONTAINED WHAT IS
10755                                     ;GIVEN IN BAD RHWC
10755 032120                               71$:
10756
10757
10758
10759 ;:*****
10760 ;*TEST 47          TEST DBL (RHCS3 BIT #10) TEST H
10761
10762 ;*          CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)

```

```

10763      : *   SET UP FOR TEN WORD READ FROM AN EVEN WORD BOUNDARY
10764      : *   DO A TEN WORD READ FROM AN EVEN WORD BOUNDARY
10765      : *   THIS SHOULD SET RHCS3 BIT #10 DBL
10766      : *   CHECK RHCS3
10767      : *   CHECK RHCS1, RHCS2, RHBA, RHBAE, RHC
10768
10769      : *****
10770      032120 000004      ST47: SCOPE
10771      032122 012706 001000      MOV #STACK, SP ; RESET STACK
10772      032126 012737 000047 004656      MOV #47, @STNM ; SAVE TEST NUMBER
10773
10774      032134 004737 040322      JSR PC, @CLDISK ; GIVE RH INITIALIZE
10775                                     ; SETUP UNIT NUMBER
10776                                     ; CLEAR RHC AND FUNCTION BITS IN RHCS1
10777
10778      : SET UP FOR A TEN WORD READ FROM AN EVEN WORD BOUNDARY
10779      032140 012701 000003      MOV #3, R1 ; COUNT OF THREE
10780      032144 012702 003000      MOV #BFEVEN, R2 ; START THREE WORD BUFFER
10781                                     ; FROM AN EVEN WORD BOUNDARY
10782      032150 012722 052525      1$: MOV #52525, (R2)+ ; MOVE THREE 52525 INTO BUFFER
10783      032154 005301      DEC R1 ; COUNT
10784      032156 001374      BNE 1$ ; BRANCH IF THREE NOT DONE
10785      032160 012777 003000 151676      MOV #BFEVEN, @RHBA ; SET BUS ADDRESS TO START
10786                                     ; FROM AN EVEN WORD BOUNDARY
10787      032166 012777 177766 151666      MOV #-10, @RHC ; WORD COUNT TEN
10788      032174 004077 152440      JSR R0, @COMND ; GO TO DO COMMAND
10789      032200 004160      READAT ; READ DATA
10790
10791
10792      : CHECK THAT RHCS3 HAS DBL
10793      032202 012737 002000 001124      MOV #DBL, $GDDAT ; GET GOOD = 2000
10794
10795      032210 017737 151716 001126      MOV @RHCS3, $BDDAT ; READ RHCS3 FOR COMPARISON
10796      032216 023737 001124 001126      CMP $GDDAT, $BDDAT ; COMPARE EXPECTED
10797                                     ; DATA WITH DATA READ FROM
10798                                     ; RHCS3
10799      032224 001401      BEQ 65$ ; BRANCH IF GOOD
10800      032226 104124      ERROR 124
10801
10802      : AFTER SETTING "CLR" BIT #5
10803      : IN RHCS2 TO INIT THE RH
10804      : A TEN WORD READ FROM AN
10805      : EVEN WORD BOUNDARY WAS DONE
10806      : THEN
10807      : RHCS3 SHOULD HAVE DBL
10808      : =2000
10809      : BUT CONTAINED WHAT IS
10810      : GIVEN IN BAD RHCS3
10810      032230      65$:
10811      032230 042777 000076 151622      BIC #76, @RHCS1 ; CLEAR FUNCTION BITS
10812      : CHECK THAT RHCS1 HAS RDY!DVA
10813      032236 012737 004200 001124      MOV #RDY!DVA, $GDDAT ; GET GOOD = 4200
10814
10815      032244 017737 151610 001126      MOV @RHCS1, $BDDAT ; READ RHCS1 FOR COMPARISON
10816      032252 023737 001124 001126      CMP $GDDAT, $BDDAT ; COMPARE EXPECTED
10817                                     ; DATA WITH DATA READ FROM
10818                                     ; RHCS1

```

```

10819 032260 001401          BEQ      67$          :BRANCH IF GOOD
10820 032262 104125          ERROR   125
10821                                     :AFTER SETTING "CLR" BIT #5
10822                                     :IN RHCS2 TO INIT THE RH
10823                                     :A TEN WORD READ FROM AN
10824                                     :EVEN WORD BOUNDARY WAS DONE
10825                                     :THEN
10826                                     :RHCS1 SHOULD HAVE RDY!DVA
10827                                     :=4200
10828                                     :BUT CONTAINED WHAT IS
10829                                     :GIVEN IN BAD RHCS1
10830 032264          67$:
10831                                     :CHECK THAT RHCS2 HAS IR
10832 032264 012737 000100 001124  MOV     #IR,$GDDAT  :GET GOOD = 100
10833 032272 053737 005010 001124  BIS     UNIT,$GDDAT :INCLUDE UNIT NUMBER
10834
10835 032300 017737 151564 001126  MOV     @RHCS2,$BDDAT :READ RHCS2 FOR COMPARISON
10836 032306 023737 001124 001126  CMP     $GDDAT,$BDDAT :COMPARE EXPECTED
10837                                     :DATA WITH DATA READ FROM
10838                                     :RHCS2
10839 032314 001401          BEQ      69$          :BRANCH IF GOOD
10840 032316 104126          ERROR   126
10841                                     :AFTER SETTING "CLR" BIT #5
10842                                     :IN RHCS2 TO INIT THE RH
10843                                     :A TEN WORD READ FROM AN
10844                                     :EVEN WORD BOUNDARY WAS DONE
10845                                     :THEN
10846                                     :RHCS2 SHOULD HAVE IR
10847                                     :=100
10848                                     :TOGETHER WITH UNIT NUMBER
10849                                     :BUT CONTAINED WHAT IS
10850                                     :GIVEN IN BAD RHCS2
10851 032320          69$:
10852                                     :CHECK THAT RHWC HAS 0
10853 032320 012737 000000 001124  MOV     #0,$GDDAT   :GET GOOD = 0
10854
10855 032326 017737 151530 001126  MOV     @RHWC,$BDDAT :READ RHWC FOR COMPARISON
10856 032334 023737 001124 001126  CMP     $GDDAT,$BDDAT :COMPARE EXPECTED
10857                                     :DATA WITH DATA READ FROM
10858                                     :RHWC
10859 032342 001401          BEQ      71$          :BRANCH IF GOOD
10860 032344 104131          ERROR   131
10861                                     :AFTER SETTING "CLR" BIT #5
10862                                     :IN RHCS2 TO INIT THE RH
10863                                     :A TEN WORD READ FROM AN
10864                                     :EVEN WORD BOUNDARY WAS DONE
10865                                     :THEN
10866                                     :RHWC SHOULD HAVE 0
10867                                     :BUT CONTAINED WHAT IS
10868                                     :GIVEN IN BAD RHWC
10869 032346          71$:
10870
10871
10872
10873
10874

```

```

:*****
: *TEST 50      TEST DBL (RHCS3 BIT #10) TEST I

```



```

10931                                     :RHCS1
10932 032506 001401                       BEG      67$    ;BRANCH IF GOOD
10933 032510 104125                       ERROR    125
10934                                     ;AFTER SETTING "CLR" BIT #5
10935                                     ;IN RHCS2 TO INIT THE RH
10936                                     ;A TEN WORD READ FROM AN
10937                                     ;ODD WORD BOUNDARY WAS DONE
10938                                     ;THEN
10939                                     ;RHCS1 SHOULD HAVE RDY!DVA
10940                                     ;=4200
10941                                     ;BUT CONTAINED WHAT IS
10942                                     ;GIVEN IN BAD RHCS1
10943 032512                               67$:
10944                                     ;CHECK THAT RHCS2 HAS IR
10945 032512 012737 000100 001124          MOV      #IR,$GDDAT ;GET GOOD = 100
10946 032520 053737 005010 001124          BIS      UNIT,$GDDAT ;INCLUDE UNIT NUMBER
10947
10948 032526 017737 151336 001126          MOV      @RHCS2,$BDDAT ;READ RHCS2 FOR COMPARISON
10949 032534 023737 001124 001126          CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED
10950                                     ;DATA WITH DATA READ FROM
10951                                     ;RHCS2
10952 032542 001401                       BEG      69$    ;BRANCH IF GOOD
10953 032544 104126                       ERROR    126
10954                                     ;AFTER SETTING "CLR" BIT #5
10955                                     ;IN RHCS2 TO INIT THE RH
10956                                     ;A TEN WORD READ FROM AN
10957                                     ;ODD WORD BOUNDARY WAS DONE
10958                                     ;THEN
10959                                     ;RHCS2 SHOULD HAVE IR
10960                                     ;=100
10961                                     ;TOGETHER WITH UNIT NUMBER
10962                                     ;BUT CONTAINED WHAT IS
10963                                     ;GIVEN IN BAD RHCS2
10964 032546                               69$:
10965                                     ;CHECK THAT RHWC HAS 0
10966 032546 012737 000000 001124          MOV      #0,$GDDAT ;GET GOOD = 0
10967
10968 032554 017737 151302 001126          MOV      @RHWC,$BDDAT ;READ RHWC FOR COMPARISON
10969 032552 023737 001124 001126          CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED
10970                                     ;DATA WITH DATA READ FROM
10971                                     ;RHWC
10972 032570 001401                       BEG      71$    ;BRANCH IF GOOD
10973 032572 104131                       ERROR    131
10974                                     ;AFTER SETTING "CLR" BIT #5
10975                                     ;IN RHCS2 TO INIT THE RH
10976                                     ;A TEN WORD READ FROM AN
10977                                     ;ODD WORD BOUNDARY WAS DONE
10978                                     ;THEN
10979                                     ;RHWC SHOULD HAVE 0
10980                                     ;BUT CONTAINED WHAT IS
10981                                     ;GIVEN IN BAD RHWC
10982 032574                               71$:
10983
10984
10985
10986

```

::\*\*\*\*\*

```

10987          : *TEST S1          TEST DBL (RHCS3 BIT #10) TEST J
10988
10989          : *          CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5,
10990          : *          SET UP FOR A ELEVEN WORD READ FROM AN EVEN WORD BOUNDARY
10991          : *          DO A ELEVEN WORD READ FROM AN EVEN WORD BOUNDARY
10992          : *          THIS SHOULD NOT SET RHCS3 BIT #10 DBL
10993          : *          CHECK RHCS3
10994          : *          CHECK RHCS1, RHCS2, RHBA, RHBAE, RHCW
10995
10996          : *****
10997          TS^S1: SCOPE
10998          032574 000004          MOV      #STACK, SP          ;RESET STACK
10999          032576 012706 001000          MOV      #S1, @^TSTNM        ;SAVE TEST NUMBER
11000          032602 012737 000051 004656          JSR      PC, @^CLDISK        ;GIVE RH INITIALIZE
11001          032610 004737 04^322          ;SETUP UNIT NUBER
11002          ;CLEAR RHCW AND FUNCTION BITS IN RHCS1
11003
11004
11005          ;SET UP FOR A ELEVEN WORD READ FROM AN EVEN WORD BOUNDARY
11006          032614 012701 000003          MOV      #3, R1              ;COUNT OF THREE
11007          032620 012702 003000          MOV      #BF EVEN, R2        ;START THREE WORD BUFFER
11008          ;FROM AN EVEN WORD BOUNDARY
11009          032624 012722 052525          1$: MOV      #52525, (R2)+    ;MOVE THREE 52525 INTO BUFFER
11010          032630 005301          DEC      R1                  ;COUNT
11011          032632 001374          BNE     1$                   ;BRANCH IF THREE NOT DONE
11012          032634 012777 003000 151222          MOV      #BF EVEN, @RHBA     ;SET BUS ADDRESS TO START
11013          ;FROM AN EVEN WORD BOUNDARY
11014          032642 012777 177765 151212          MOV      #-11, @RHCW        ;WORD COUNT ELEVEN
11015          032650 004077 151754          JSR      @^COMND            ;GO TO DO COMMAND
11016          032654 004160          READAT          ;READ DATA
11017
11018
11019          ;CHECK THAT RHCS3 HAS 0
11020          032656 012737 000000 001124          MOV      #0, $GDDAT         ;GET GOOD = 0
11021
11022          032664 017737 151242 001126          MOV      @RHCS3, $BDDAT     ;READ RHCS3 FOR COMPARISON
11023          032672 023737 001124 001126          CMP      $GDDAT, $BDDAT     ;COMPARE EXPECTED
11024          ;DATA WITH DATA READ FROM
11025          ;RHCS3
11026          032700 001401          BEQ     65$                  ;BRANCH IF GOOD
11027          032702 104124          ERROR   124
11028          ;AFTER SETTING "CLR" BIT #5
11029          ;IN RHCS2 TO INIT THE RH
11030          ;A ELEVEN WORD READ FROM AN
11031          ;EVEN WORD BOUNDARY WAS DONE
11032          ;THEN
11033          ;RHCS3 SHOULD HAVE 0
11034          ;BUT CONTAINED WHAT IS
11035          ;GIVEN IN BAD RHCS3
11036          032704          65$: BIC      #76, @RHCS1        ;CLEAR FUNCTION BITS
11037          032704 042777 000076 151146          ;CHECK THAT RHCS1 HAS RDY!DVA
11038          032712 012737 004200 001124          MOV      #RDY!DVA, $GDDAT   ;GET GOOD = 4200
11039
11040
11041          032720 017737 151134 001126          MOV      @RHCS1, $BDDAT     ;READ RHCS1 FOR COMPARISON
11042          032726 023737 001124 001126          CMP      $GDDAT, $BDDAT     ;COMPARE EXPECTED

```





M16

MAINDEC-11-DEHAC-A  
DERHAC.P11 T52

MACY11 27(732) 09-SEP-76 07:55 PAGE 207  
TEST DBL (RHCS3 BIT #10) TEST K

```

11099          ;*****
11100          ;*TEST 52          TEST DBL (RHCS3 BIT #10) TEST K
11101
11102          ;*          CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
11103          ;*          SET UP FOR A ELEVEN WORD READ FROM AN ODD WORD BOUNDARY
11104          ;*          DO A ELEVEN WORD READ FROM AN ODD WORD BOUNDARY
11105          ;*          THIS SHOULD SET RHCS3 BIT #10 DBL
11106          ;*          CHECK RHCS3
11107          ;*          CHECK RHCS1, RHCS2, RHBA, RHBAE, RHWC
11108
11109          ;*****
11110 033000 000004          †ST52: SCOPE
11111 033024 012706 001000          MOV      #STACK, SP          ;RESET STACK
11112 033030 012737 000052 004656          MOV      #52, 2#†STNM          ;SAVE TEST NUMBER
11113
11114 033036 004737 040322          JSR      PC, 2#CLDISK          ;GIVE RH INITIALIZE
11115          ;SETUP UNIT NUBER
11116          ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
11117
11118          ;SET UP FOR A ELEVEN WORD READ FROM AN ODD WORD BOUNDARY
11119 033042 012701 000003          MOV      #3, R1          ;COUNT OF THREE
11120 033046 012702 003002          MOV      #BFODD, R2          ;START THREE WORD BUFFER
11121          ;FROM AN ODD WORD BOUNDARY
11122 033052 012722 052525          1$: MOV      #52525, (R2)+          ;MOVE THREE 52525 INTO BUJFFER
11123 033056 005301          DEC      R1          ;COUNT
11124 033060 001374          BNE     1$          ;BRANCH IF THREE NOT DONE
11125 033062 012777 003002 150774          MOV      #BFODD, 2#RHBA          ;SET BUS ADDRESS TO START
11126          ;FROM AN ODD WORD BOUNDARY
11127 033070 012777 177765 150764          MOV      #-11, 2#RHWC          ;WORD COUNT ELEVEN
11128 033076 004077 151536          JSR      RD, 2#COMND          ;GO TO DO COMMAND
11129 033102 004160          READAT          ;READ DATA
11130
11131
11132          ;CHECK THAT RHCS3 HAS DBL
11133 033104 012737 002000 001124          MOV      #DBL, $GDDAT          ;GET GOOD = 2000
11134
11135 033112 017737 151014 001126          MOV      2#RHCS3, $BDDAT          ;READ RHCS3 FOR COMPARISON
11136 033120 023737 001124 001126          CMP      $GDDAT, $BDDAT          ;COMPARE EXPECTED
11137          ;DATA WITH DATA READ FROM
11138          ;RHCS3
11139 033126 001401          BEQ     65$          ;BRANCH IF GOOD
11140 033130 104124          ERROR   124
11141          ;AFTER SETTING "CLR" BIT #5
11142          ;IN RHCS2 TO INIT THE RH
11143          ;A ELEVEN WORD READ FROM AN
11144          ;ODD WORD BOUNDARY WAS DONE
11145          ;THEN
11146          ;RHCS3 SHOULD HAVE DBL
11147          ;=2000
11148          ;BUT CONTAINED WHAT IS
11149          ;GIVEN IN BAD RHCS3
11150 033132          65$:
11151 033132 042777 000076 150720          BIC      #76, 2#RHCS1          ;CLEAR FUNCTION BITS
11152          ;CHECK THAT RHCS1 HAS RDY!DVA
11153 033140 012737 004200 001124          MOV      #RDY!DVA, $GDDAT          ;GET GOOD = 4200
11154

```

11155	033146	017737	150706	001126	MOV	DRHCS1,\$BDDAT	; READ RHCS1 FOR COMPARISON
11156	033154	023737	001124	001126	CMP	\$GDDAT,\$BDDAT	; COMPARE EXPECTED
11157							; DATA WITH DATA READ FROM
11158							; RHCS1
11159	033162	001401			BEQ	67\$	; BRANCH IF GOOD
11160	033164	104125			ERROR	125	
11161							; AFTER SETTING "CLR" BIT #5
11162							; IN RHCS2 TO INIT THE RH
11163							; A ELEVEN WORD READ FROM AN
11164							; ODD WORD BOUNDARY WAS DONE
11165							; THEN
11166							; RHCS1 SHOULD HAVE RDY!DVA
11167							; =4200
11168							; BUT CONTAINED WHAT IS
11169							; GIVEN IN BAD RHCS1
11170	033166					67\$:	
11171							; CHECK THAT RHCS2 HAS IR
11172	033166	012737	000100	001124	MOV	#IR,\$GDDAT	; GET GOOD = 100
11173	033174	053737	005010	001124	BIS	UNIT,\$GDDAT	; INCLUDE UNIT NUMBER
11174							
11175	033202	017737	150662	001126	MOV	DRHCS2,\$BDDAT	; READ RHCS2 FOR COMPARISON
11176	033210	023737	001124	001126	CMP	\$GDDAT,\$BDDAT	; COMPARE EXPECTED
11177							; DATA WITH DATA READ FROM
11178							; RHCS2
11179	033216	001401			BEQ	69\$	; BRANCH IF GOOD
11180	033220	104126			ERROR	126	
11181							; AFTER SETTING "CLR" BIT #5
11182							; IN RHCS2 TO INIT THE RH
11183							; A ELEVEN WORD READ FROM AN
11184							; ODD WORD BOUNDARY WAS DONE
11185							; THEN
11186							; RHCS2 SHOULD HAVE IR
11187							; =100
11188							; TOGETHER WITH UNIT NUMBER
11189							; BUT CONTAINED WHAT IS
11190							; GIVEN IN BAD RHCS2
11191	033222					69\$:	
11192							; CHECK THAT RHWC HAS 0
11193	033222	012737	000000	001124	MOV	#0,\$GDDAT	; GET GOOD = 0
11194							
11195	033230	017737	150626	001126	MOV	DRHWC,\$BDDAT	; READ RHWC FOR COMPARISON
11196	033236	023737	001124	001126	CMP	\$GDDAT,\$BDDAT	; COMPARE EXPECTED
11197							; DATA WITH DATA READ FROM
11198							; RHWC
11199	033244	001401			BEQ	71\$	; BRANCH IF GOOD
11200	033246	104131			ERROR	131	
11201							; AFTER SETTING "CLR" BIT #5
11202							; IN RHCS2 TO INIT THE RH
11203							; A ELEVEN WORD READ FROM AN
11204							; ODD WORD BOUNDARY WAS DONE
11205							; THEN
11206							; RHWC SHOULD HAVE 0
11207							; BUT CONTAINED WHAT IS
11208							; GIVEN IN BAD RHWC
11209	033250					71\$:	
11210							

```

11211
11212
11213
11214
11215
11216
11217
11218
11219
11220
11221
11222
11223
11224
11225 033250 000004
11226 033252 012706 001000
11227 033256 012737 000053 004656
11228
11229 033264 004737 040322
11230
11231
11232
11233
11234 033270 012701 000003
11235 033274 012702 003000
11236
11237 033300 012722 052525
11238 033304 005301
11239 033306 001374
11240 033310 012777 003000 150546
11241
11242 033316 012777 177766 150536
11243 033324 052777 000010 150536
11244 033332 004077 151302
11245 033336 004160
11246
11247
11248
11249 033340 012737 000000 001124
11250
11251 033346 017737 150560 001126
11252 033354 023737 001124 001126
11253
11254
11255 033362 001401
11256 033364 104124
11257
11258
11259
11260
11261
11262
11263
11264
11265
11266 033366

```

```

;*****
;TEST 53 TEST DBL (RHCS3 BIT #10) TEST L
;
; CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
; SET UP FOR A TEN WORD READ FROM AN EVEN WORD BOUNDARY
; WITH BAI IN RHCS2 BIT #3 SET
; DO A TEN WORD READ FROM AN EVEN WORD BOUNDARY
; THIS SHOULD NOT SET RHCS3 BIT #10 DBL
; CHECK RHCS3
; CHECK RHCS1, RHCS2, RHBA, RHBAE, RHWC
;*****
TST53: SCOPE
MOV #STACK, SP ;RESET STACK
MOV #53, @#TSTNM ;SAVE TEST NUMBER
JSR PC, @#CLDISK ;GIVE RH INITIALIZE
;SETUP UNIT NUBER
;CLEAR RHWC AND FUNCTION BITS IN RHCS1
;SET UP FOR A TEN WORD READ FROM AN EVEN WORD BOUNDARY
MOV #3, R1 ;COUNT OF THREE
MOV #BFEVEN, R2 ;START THREE WORD BUFFER
;FROM AN EVEN WORD BOUNDARY
1$: MOV #52525, (R2)+ ;MOVE THREE 52525 INTO BUFFER
DEC R1 ;COUNT
BNE 1$ ;BRANCH IF THREE NOT DONE
MOV #BFEVEN, @RHBA ;SET BUS ADDRESS TO START
;FROM AN EVEN WORD BOUNDARY
MOV #-10, @RHWC ;WORD COUNT TEN
BIS #BAI, @RHCS2 ;SET BAI IN RHCS2
JSR @R0, @COMND ;GO TO DO COMMAND
READAT ;READ DATA
;CHECK THAT RHCS3 HAS 0
MOV #0, $GDDAT ;GET GOOD = 0
MOV @RHCS3, $BDDAT ;READ RHCS3 FOR COMPARISON
CMP $GDDAT, $BDDAT ;COMPARE EXPECTED
;DATA WITH DATA READ FROM
;RHCS3
BEQ 65$ ;BRANCH IF GOOD
ERROR 124
;AFTER SETTING "CLR" BIT #5
;IN RHCS2 TO INIT THE RH
;A TEN WORD READ FROM AN
;EVEN WORD BOUNDARY WAS DONE
;WITH BAI IN RHCS2 SET
;THEN
;RHCS3 SHOULD HAVE 0
;BUT CONTAINED WHAT IS
;GIVEN IN BAD RHCS3
65$:

```



11323  
11324  
11325  
11326  
11327  
11328  
11329  
11330  
11331  
11332  
11333  
11334  
11335  
11336  
11337  
11338  
11339  
11340  
11341  
11342  
11343  
11344  
11345  
11346  
11347  
11348  
11349  
11350  
11351  
11352  
11353  
11354  
11355  
11356  
11357  
11358  
11359  
11360  
11361  
11362  
11363  
11364  
11365  
11366  
11367  
11368  
11369  
11370  
11371  
11372  
11373  
11374  
11375  
11376  
11377  
11378

033504

715:

:WITH BAI IN RHCS2 SET  
:THEN  
:RHC SHOULD HAVE 0  
:BUT CONTAINED WHAT IS  
:GIVEN IN BAD RHC

::\*\*\*\*\*  
;\*TEST 54 TEST DBL (RHCS3 BIT #10) TEST M

:\* CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)  
:\* SET UP FOR A ONE WORD WRITE REVERSE FROM AN EVEN WORD BOUNDARY  
:\* DO A ONE WORD WRITE REVERSE FROM AN EVEN WORD BOUNDARY  
:\* THIS SHOULD NOT SET RHCS3 BIT #10 DBL  
:\* CHECK RHCS3  
:\* CHECK RHCS1, RHCS2, RHBA, RHBAE, RHC

::\*\*\*\*\*

†T54:

SCOPE  
MOV #STACK, SP ;RESET STACK  
MOV #54, @†TSTNM ;SAVE TEST NUMBER  
JSR PC, @#CLDISK ;GIVE RH INITIALIZE  
;SETUP UNIT NUBER  
;CLEAR RHC AND FUNCTION BITS IN RHCS1  
;SET UP FOR A ONE WORD WRITE REVERSE FROM AN EVEN WORD BOUNDARY  
MOV #3, R1 ;COUNT OF THREE  
MOV #BFEVEN, R2 ;START THREE WORD BUFFER  
;FROM AN EVEN WORD BOUNDARY  
1\$: MOV #52525, (R2)+ ;MOVE THREE 52525 INTO BUFFER  
DEC R1 ;COUNT  
BNE 1\$ ;BRANCH IF THREE NOT DONE  
MOV #BFEVEN, @RHBA ;SET BUS ADDRESS TO START  
;FROM AN EVEN WORD BOUNDARY  
MOV #-1, @RHC ;WORD COUNT ONE  
JSR @R0, @COMND ;GO TO DO COMMAND  
REVWRT ;REVERSEWRITE DATA

000004  
012706 001000  
012737 000054 004656  
004737 040322  
033524 012701 000003  
033530 012702 003000  
033534 012722 052525  
033540 005301  
033542 001374  
033544 012777 003000 150312  
033552 012777 177777 150302  
033560 004077 151054  
033564 004204

;CHECK THAT RHCS3 HAS 0  
MOV #0, \$GDDAT ;GET GOOD = 0

MOV @RHCS3, \$BDDAT ;READ RHCS3 FOR COMPARISON  
CMP \$GDDAT, \$BDDAT ;COMPARE EXPECTED  
;DATA WITH DATA READ FROM  
;RHCS3  
BEQ 65\$ ;BRANCH IF GOOD  
ERROR 124

:AFTER SETTING "CLR" BIT #5  
:IN RHCS2 TO INIT THE RH  
:A ONE WORD WRITE REVERSE FROM AN  
:EVEN WORD BOUNDARY WAS DONE



```

11435 ; IN RHCS2 TO INIT THE RH
11436 ; A ONE WORD WRITE REVERSE FROM AN
11437 ; EVEN WORD BOUNDARY WAS DONE
11438 ; THEN
11439 ; RHCW SHOULD HAVE 0
11440 ; BUT CONTAINED WHAT IS
11441 ; GIVEN IN BAD RHCW
11442 033732 71$:
11443
11444
11445
11446 ::*****
11447 :*TEST 55 TEST DBL (RHCS3 BIT #10) TEST N
11448
11449 ;* CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
11450 ;* SET UP FOR A TWO WORD WRITE REVERSE FROM AN EVEN WORD BOUNDARY
11451 ;* DO A TWO WORD WRITE REVERSE FROM AN EVEN WORD BOUNDARY
11452 ;* THIS SHOULD NOT SET RHCS3 BIT #10 DBL
11453 ;* CHECK RHCS3
11454 ;* CHECK RHCS1, RHCS2, RHBA, RHBAE, RHCW
11455
11456 ::*****
11457 033732 000004 †ST55: SCOPE
11458 033734 012706 001000 MOV #STACK, SP ;RESET STACK
11459 033740 012737 000055 0J4656 MOV #55, †STNM ;SAVE TEST NUMBER
11460
11461 033746 004737 040322 JSR PC, †CLDISK ;GIVE RH INITIALIZE
11462 ;SETUP UNIT NUBER
11463 ;CLEAR RHCW AND FUNCTION BITS IN RHCS1
11464
11465 ;SET UP FOR A TWO WORD WRITE REVERSE FROM AN EVEN WORD BOUNDARY
11466 033752 012701 000003 MOV #3, R1 ;COUNT OF THREE
11467 033756 012702 003000 MOV #8FEVEN, R2 ;START THREE WORD BUFFER
11468 ;FROM AN EVEN WORD BOUNDARY
11469 033762 012722 052525 1$: MOV #52525, (R2)+ ;MOVE THREE 52525 INTO BUFFER
11470 033766 005301 DEC R1 ;COUNT
11471 033770 001374 BNE 1$ ;BRANCH IF THREE NOT DONE
11472 033772 012777 003000 150064 MOV #8FEVEN, †RHBA ;SET BUS ADDRESS TO START
11473 ;FROM AN EVEN WORD BOUNDARY
11474 034000 012777 177776 150054 MOV #-2, †RHCW ;WORD COUNT TWO
11475 034006 004077 150626 JSR †RO, †COMND ;GO TO DO COMMAND
11476 034012 004204 REVWRT ;REVERSEWRITE DATA
11477
11478
11479 ;CHECK THAT RHCS3 HAS 0
11480 034014 012737 000000 001124 MOV #0, $GDDAT ;GET GOOD = 0
11481
11482 034022 017737 150104 001126 MOV †RHCS3, $BDDAT ;READ RHCS3 FOR COMPARISON
11483 034030 023737 001124 001126 CMP $GDDAT, $BDDAT ;COMPARE EXPECTED
11484 ;DATA WITH DATA READ FROM
11485 ;RHCS3
11486 034036 001401 BEQ 65$ ;BRANCH IF GOOD
11487 034040 104124 ERROR 124
11488 ;AFTER SETTING "CLR" BIT #5
11489 ;IN RHCS2 TO INIT THE RH
11490 ;A TWO WORD WRITE REVERSE FROM AN

```

```

11491                                     ;EVEN WORD BOUNDARY WAS DONE
11492                                     ;THEN
11493                                     ;RHCS3 SHOULD HAVE 0
11494                                     ;BUT CONTAINED WHAT IS
11495                                     ;GIVEN IN BAD RHCS3
11496 034042                                65$:
11497 034042 042777 000076 150010        BIC #76, @RHCS1 ;CLEAR FUNCTION BITS
11498                                     ;CHECK THAT RHCS1 HAS SC!RDY!DVA
11499 034050 012737 104200 001124        MOV #SC!RDY!DVA, $GDDAT ;GET GOOD = 104200
11500
11501 034056 017737 147776 001126        MOV @RHCS1, $BDDAT ;READ RHCS1 FOR COMPARISON
11502 034064 023737 001124 001126        CMP $GDDAT, $BDDAT ;COMPARE EXPECTED
11503                                     ;DATA WITH DATA READ FROM
11504                                     ;RHCS1
11505 034072 001401                          BEQ 67$ ;BRANCH IF GOOD
11506 034074 104125                          ERROR 125
11507
11508                                     ;AFTER SETTING "CLR" BIT #5
11509                                     ;IN RHCS2 TO INIT THE RH
11510                                     ;A TWO WORD WRITE REVERSE FROM AN
11511                                     ;EVEN WORD BOUNDARY WAS DONE
11512                                     ;THEN
11513                                     ;RHCS1 SHOULD HAVE SC!RDY!DVA
11514                                     ;=104200
11515                                     ;BUT CONTAINED WHAT IS
11516                                     ;GIVEN IN BAD RHCS1
11516 034076                                67$:
11517                                     ;CHECK THAT RHCS2 HAS IR!OR
11518 034076 012737 000300 001124        MOV #IR!OR, $GDDAT ;GET GOOD = 300
11519 034104 053737 005010 001124        BIS UNIT, $GDDAT ;INCLUDE UNIT NUMBER
11520
11521 034112 017737 147752 001126        MOV @RHCS2, $BDDAT ;READ RHCS2 FOR COMPARISON
11522 034120 023737 001124 001126        CMP $GDDAT, $BDDAT ;COMPARE EXPECTED
11523                                     ;DATA WITH DATA READ FROM
11524                                     ;RHCS2
11525 034126 001401                          BEQ 69$ ;BRANCH IF GOOD
11526 034130 104126                          ERROR 126
11527
11528                                     ;AFTER SETTING "CLR" BIT #5
11529                                     ;IN RHCS2 TO INIT THE RH
11530                                     ;A TWO WORD WRITE REVERSE FROM AN
11531                                     ;EVEN WORD BOUNDARY WAS DONE
11532                                     ;THEN
11533                                     ;RHCS2 SHOULD HAVE IR!OR
11534                                     ;=300
11535                                     ;TOGETHER WITH UNIT NUMBER
11536                                     ;BUT CONTAINED WHAT IS
11537                                     ;GIVEN IN BAD RHCS2
11537 034132                                69$:
11538                                     ;CHECK THAT RHWC HAS 0
11539 034132 012737 000000 001124        MOV #0, $GDDAT ;GET GOOD = 0
11540
11541 034140 017737 147716 001126        MOV @RHWC, $BDDAT ;READ RHWC FOR COMPARISON
11542 034146 023737 001124 001126        CMP $GDDAT, $BDDAT ;COMPARE EXPECTED
11543                                     ;DATA WITH DATA READ FROM
11544                                     ;RHWC
11545 034154 001401                          BEQ 71$ ;BRANCH IF GOOD
11546 034156 104131                          ERROR 131

```



; AFTER SETTING "CLR" BIT #5  
; IN RHCS2 TO INIT THE RH  
; A TWO WORD WRITE REVERSE FROM AN  
; EVEN WORD BOUNDARY WAS DONE  
; THEN  
; RHWC SHOULD HAVE 0  
; BUT CONTAINED WHAT IS  
; GIVEN IN BAD RHWC

11547  
11548  
11549  
11550  
11551  
11552  
11553  
11554  
11555  
11556  
11557  
11558  
11559  
11560  
11561  
11562  
11563  
11564  
11565  
11566  
11567  
11568  
11569  
11570  
11571  
11572  
11573  
11574  
11575  
11576  
11577  
11578  
11579  
11580  
11581  
11582  
11583  
11584  
11585  
11586  
11587  
11588  
11589  
11590  
11591  
11592  
11593  
11594  
11595  
11596  
11597  
11598  
11599  
11600  
11601  
11602

034160

715:

\*\*\*\*\*  
\*TEST 56 TEST DBL (RHCS3 BIT #10) TEST 0

\* CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)  
\* SET UP FOR A TWO WORD WRITE REVERSE FROM AN ODD WORD BOUNDARY  
\* DO A TWO WORD WRITE REVERSE FROM AN ODD WORD BOUNDARY  
\* THIS SHOULD SET RHCS3 BIT #10 DBL  
\* CHECK RHCS3  
\* CHECK RHCS1, RHCS2, RHBA, RHBAE, RHWC

\*\*\*\*\*  
TST56: SCOPE

034160 000004  
034162 012706 001000  
034166 012777 000056 004656

MOV #STACK, SP ; RESET STACK  
MOV #56, @#1STNM ; SAVE TEST NUMBER

034174 004737 040322

JSR PC, @#CLDISK ; GIVE RH INITIALIZE  
; SETUP UNIT NUBER  
; CLEAR RHWC AND FUNCTION BITS IN RHCS1

034200 012701 000003  
034204 012702 003002

; SET UP FOR A TWO WORD WRITE REVERSE FROM AN ODD WORD BOUNDARY  
MOV #3, R1 ; COUNT OF THREE  
MOV #BF0DD, R2 ; START THREE WORD BUFFER  
; FROM AN ODD WORD BOUNDARY

034210 012722 052525  
034214 005301  
034216 001374  
034220 012777 003002 147636

15: MOV #52525, (R2)+ ; MOVE THREE 52525 INTO BUFFER  
DEC R1 ; COUNT  
BNE 15 ; BRANCH IF THREE NOT DONE  
MOV #BF0DD, @RHBA ; SET BUS ADDRESS TO START  
; FROM AN ODD WORD BOUNDARY

034226 012777 177776 147626  
034234 004077 150400  
034240 004204

MOV #-2, @RHWC ; WORD COUNT TWO  
JSR RO, @COMND ; GO TO DO COMMAND  
REVWRT ; REVERSEWRITE DATA

034242 012737 002000 001124

; CHECK THAT RHCS3 HAS DBL  
MOV #DBL, \$GDDAT ; GET GOOD = 2000

034250 017737 147656 001126  
034256 023737 001124 001126

MOV @RHCS3, \$BDDAT ; READ RHCS3 FOR COMPARISON  
CMP \$GDDAT, \$BDDAT ; COMPARE EXPECTED  
; DATA WITH DATA READ FROM  
; RHCS3  
BEQ 65\$ ; BRANCH IF GOOD

034264 001401  
034266 104124

ERROR 124

; AFTER SETTING "CLR" BIT #5  
; IN RHCS2 TO INIT THE RH

```

11603 ;A TWO WORD WRITE REVERSE FROM AN
11604 ;ODD WORD BOUNDARY WAS DONE
11605 ;THEN
11606 ;RHCS3 SHOULD HAVE DBL
11607 ;=2000
11608 ;BUT CONTAINED WHAT IS
11609 ;GIVEN IN BAD RHCS3
11610 034270 65$:
11611 034270 042777 000076 147562 BIC #76,DRHCS1 ;CLEAR FUNCTION BITS
11612 ;CHECK THAT RHCS1 HAS SC!RDY!DVA
11613 034276 012737 104200 001124 MOV #SC!RDY!DVA,$GDDAT ;GET GOOD = 104200
11614
11615 034304 017737 147550 001126 MOV DRHCS1,$BDDAT ;READ RHCS1 FOR COMPARISON
11616 034312 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
11617 ;DATA WITH DATA READ FROM
11618 ;RHCS1
11619 034320 001401 BEQ 67$ ;BRANCH IF GOOD
11620 034322 104125 ERROR 125
11621 ;AFTER SETTING "CLR" BIT #5
11622 ;IN RHCS2 TO INIT THE RH
11623 ;A TWO WORD WRITE REVERSE FROM AN
11624 ;ODD WORD BOUNDARY WAS DONE
11625 ;THEN
11626 ;RHCS1 SHOULD HAVE SC!RDY!DVA
11627 ;=104200
11628 ;BUT CONTAINED WHAT IS
11629 ;GIVEN IN BAD RHCS1
11630 034324 67$:
11631 ;CHECK THAT RHCS2 HAS IR!OR
11632 034324 012737 000300 001124 MOV #IR!OR,$GDDAT ;GET GOOD = 300
11633 034332 053737 005010 001124 BIS UNIT,$GDDAT ;INCLUDE UNIT NUMBER
11634
11635 034340 017737 147524 001126 MOV DRHCS2,$BDDAT ;READ RHCS2 FOR COMPARISON
11636 034346 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
11637 ;DATA WITH DATA READ FROM
11638 ;RHCS2
11639 034354 001401 BEQ 69$ ;BRANCH IF GOOD
11640 034356 104126 ERROR 126
11641 ;AFTER SETTING "CLR" BIT #5
11642 ;IN RHCS2 TO INIT THE RH
11643 ;A TWO WORD WRITE REVERSE FROM AN
11644 ;ODD WORD BOUNDARY WAS DONE
11645 ;THEN
11646 ;RHCS2 SHOULD HAVE IR!OR
11647 ;=300
11648 ;TOGETHER WITH UNIT NUMBER
11649 ;BUT CONTAINED WHAT IS
11650 ;GIVEN IN BAD RHCS2
11651 034360 69$:
11652 ;CHECK THAT RHWC HAS 0
11653 034360 012737 000000 001124 MOV #0,$GDDAT ;GET GOOD = 0
11654
11655 034366 017737 147470 001126 MOV DRHWC,$BDDAT ;READ RHWC FOR COMPARISON
11656 034374 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
11657 ;DATA WITH DATA READ FROM
11658 ;RHWC

```

```

11659 034402 001401          BEQ      715          ;BRANCH IF GOOD
11660 034404 104131          ERROR    131
11661                                     ;AFTER SETTING "CLR" BIT #5
11662                                     ;IN RHCS2 TO INIT THE RH
11663                                     ;A TWO WORD WRITE REVERSE FROM AN
11664                                     ;ODD WORD BOUNDARY WAS DONE
11665                                     ;THEN
11666                                     ;RHWC SHOULD HAVE 0
11667                                     ;BUT CONTAINED WHAT IS
11668                                     ;GIVEN IN BAD RHWC
11669 034406          715:
11670
11671
11672
11673                                     ;*****
11674                                     ;*TEST 57          TEST DBL (RHCS3 BIT #10) TEST P
11675
11676                                     ;*
11677                                     ;*   CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
11678                                     ;*   SET UP FOR A THREE WORD WRITE REVERSE FROM AN EVEN WORD BOUNDARY
11679                                     ;*   DO A THREE WORD WRITE REVERSE FROM AN EVEN WORD BOUNDARY
11680                                     ;*   THIS SHOULD SET RHCS3 BIT #10 DBL
11681                                     ;*   CHECK RHCS3
11682                                     ;*   CHECK RHCS1 , RHCS2 , RHBA , RHBAE , RHWC
11683                                     ;*****
11684 034406 000004          †ST57: SCOPE
11685 034410 012706 001000          MOV      #STACK, SP          ;RESET STACK
11686 034414 012737 000057 304656  MOV      #57, #†STNM          ;SAVE TEST NUMBER
11687
11688 034422 004737 040322          JSR      PC, #CLDISK          ;GIVE RH INITIALIZE
11689                                     ;SETUP UNIT NUBER
11690                                     ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
11691
11692                                     ;SET UP FOR A THREE WORD WRITE REVERSE FROM AN EVEN WORD BOUNDARY
11693 034426 012701 000003          MOV      #3, R1              ;COUNT OF THREE
11694 034432 012702 003000          MOV      #8FEVEN, R2          ;START THREE WORD BUFFER
11695                                     ;FROM AN EVEN WORD BOUNDARY
11696 034436 012722 052525          15:   MOV      #52525, (R2)+      ;MOVE THREE 52525 INTO BUFFER
11697 034442 005301          DEC      R1                  ;COUNT
11698 034444 001374          BNE     15                   ;BRANCH IF THREE NOT DONE
11699 034446 012777 003000 147410  MOV      #8FEVEN, #RHBA          ;SET BUS ADDRESS TO START
11700                                     ;FROM AN EVEN WORD BOUNDARY
11701 034454 012777 177775 147400  MOV      #-3, #RHWC ;WORD COUNT THREE
11702 034462 004077 150152          JSR      R0, #COMMC          ;GO TO DO COMMAND
11703 034466 004204          REVWRT          ;REVERSEWRITE DATA
11704
11705
11706                                     ;CHECK THAT RHCS3 HAS DBL
11707 034470 012737 002000 001124  MOV      #. #L, #GDDAT          ;GET GOOD = 2000
11708
11709 034476 017737 147430 001126  MOV      #RHCS3, #BDDAT          ;READ RHCS3 FOR COMPARISON
11710 034504 023737 001124 001126  CMP      #GDDAT, #BDDAT          ;COMPARE EXPECTED
11711                                     ;DATA WITH DATA READ FROM
11712                                     ;RHCS3
11713 034512 001401          BEQ      655          ;BRANCH IF GOOD
11714 034514 104124          ERROR    124

```

```

11715 ;AFTER SETTING "CLR" BIT #5
11716 ;IN RHCS2 TO INIT THE RH
11717 ;A THREE WORD WRITE REVERSE FROM AN
11718 ;EVEN WORD BOUNDARY WAS DONE
11719 ;THEN
11720 ;RHCS3 SHOULD HAVE DBL
11721 ;=2000
11722 ;BUT CONTAINED WHAT IS
11723 ;GIVEN IN BAD RHCS3
11724 034516 042777 000076 147334 65$: BIC #76,2RHCS1 ;CLEAR FUNCTION BITS
11725 034516 042777 000076 147334 ;CHECK THAT RHCS1 HAS SC!RDY!DVA
11726 ;MOV #SC!RDY!DVA,$GDDAT ;GET GOOD = 104200
11727 034524 012737 104200 001124
11728
11729 034532 017737 147322 001126 MOV 2RHCS1,$BDDAT ;READ RHCS1 FOR COMPARISON
11730 034540 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
11731 ;DATA WITH DATA READ FROM
11732 ;RHCS1
11733 034546 001401 BEQ 67$ ;BRANCH IF GOOD
11734 034550 104125 ERROR 125
11735 ;AFTER SETTING "CLR" BIT #5
11736 ;IN RHCS2 TO INIT THE RH
11737 ;A THREE WORD WRITE REVERSE FROM AN
11738 ;EVEN WORD BOUNDARY WAS DONE
11739 ;THEN
11740 ;RHCS1 SHOULD HAVE SC!RDY!DVA
11741 ;=104200
11742 ;BUT CONTAINED WHAT IS
11743 ;GIVEN IN BAD RHCS1
11744 034552 67$: ;CHECK THAT RHCS2 HAS IR!OR
11745 ;MOV #IR!OR,$GDDAT ;GET GOOD = 300
11746 034552 012737 000300 001124 BIS UNIT,$GDDAT ;INCLUDE UNIT NUMBER
11747 034560 053737 005010 001124
11748
11749 034566 017737 147276 001126 MOV 2RHCS2,$BDDAT ;READ RHCS2 FOR COMPARISON
11750 034574 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
11751 ;DATA WITH DATA READ FROM
11752 ;RHCS2
11753 034602 001401 BEQ 69$ ;BRANCH IF GOOD
11754 034604 104126 ERROR 126
11755 ;AFTER SETTING "CLR" BIT #5
11756 ;IN RHCS2 TO INIT THE RH
11757 ;A THREE WORD WRITE REVERSE FROM AN
11758 ;EVEN WORD BOUNDARY WAS DONE
11759 ;THEN
11760 ;RHCS2 SHOULD HAVE IR!OR
11761 ;=300
11762 ;TOGETHER WITH UNIT NUMBER
11763 ;BUT CONTAINED WHAT IS
11764 ;GIVEN IN BAD RHCS2
11765 034606 69$: ;CHECK THAT RHWC HAS 0
11766 ;MOV #0,$GDDAT ;GET GOOD = 0
11767 034606 012737 000000 001124
11768
11769 034614 017737 147242 001126 MOV 2RHWC,$BDDAT ;READ RHWC FOR COMPARISON
11770 034622 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED

```

```

11771                                     ;DATA WITH DATA READ FROM
11772                                     ;RHWC
11773 034630 001401                       BEQ      71$
11774 034632 104131                       ERROR    131
11775                                     ;AFTER SETTING "CLR" BIT #5
11776                                     ;IN RHCS2 TO INIT THE RH
11777                                     ;A THREE WORD WRITE REVERSE FROM AN
11778                                     ;EVEN WORD BOUNDARY WAS DONE
11779                                     ;THEN
11780                                     ;RHWC SHOULD HAVE 0
11781                                     ;BUT CONTAINED WHAT IS
11782                                     ;GIVEN IN BAD RHWC
11783 034634                               71$:
11784
11785
11786
11787                                     ;*****
11788                                     ;TEST 60      TEST DBL (RHCS3 BIT #10) TEST Q
11789
11790                                     ;*
11791                                     ;*      CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
11792                                     ;*      SET UP FOR A THREE WORD WRITE REVERSE FROM AN ODD WORD BOUNDARY
11793                                     ;*      DO A THREE WORD WRITE REVERSE FROM AN ODD WORD BOUNDARY
11794                                     ;*      THIS SHOULD NOT SET RHCS3 BIT #10 DBL
11795                                     ;*      CHECK RHCS3
11796                                     ;*      CHECK RHCS1, RHCS2, RHBA, RHBAE, RHWC
11797                                     ;*****
11798 034634 000004                          †ST60: SCOPE
11799 034636 012706 001000                   MOV      #STACK, SP      ;RESET STACK
11800 034642 012737 000060 004656          MOV      #60, †STNM     ;SAVE TEST NUMBER
11801
11802 034650 004737 040322                   JSR      PC, †CLDISK    ;GIVE RH INITIALIZE
11803                                     ;SETUP UNIT NUBER
11804                                     ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
11805
11806                                     ;SET UP FOR A THREE WORD WRITE REVERSE FROM AN ODD WORD BOUNDARY
11807 034654 012701 000003                   MOV      #3, R1          ;COUNT OF THREE
11808 034660 012702 003002                   MOV      #BF00D, R2     ;START THREE WORD BUFFER
11809                                     ;FROM AN ODD WORD BOUNDARY
11810 034664 012722 052525                   1$:   MOV      #52525, (R2)+ ;MOVE THREE 52525 INTO BUFFER
11811 034670 005301                           DEC      R1              ;COUNT
11812 034672 001374                           BNE     1$              ;BRANCH IF THREE NOT DONE
11813 034674 012777 003002 147162          MOV      #BF00D, †RHBA  ;SET BUS ADDRESS TO START
11814                                     ;FROM AN ODD WORD BOUNDARY
11815 034702 012777 177775 147152          MOV      #-3, †RHWC     ;WORD COUNT THREE
11816 034710 004077 147724                   JSR      RO, †COMND     ;GO TO DO COMMAND
11817 034714 0042C4                           REVWRT                   ;REVERSEWRITE DATA
11818
11819
11820                                     ;CHECK THAT RHCS3 HAS 0
11821 034716 012737 000000 001124          MOV      #0, $GDDAT    ;GET GOOD = 0
11822
11823 034724 017737 147202 001126          MOV      †RHCS3, $BDDAT ;READ RHCS3 FOR COMPARISON
11824 034732 023737 001124 001126          CMP      $GDDAT, $BDDAT ;COMPARE EXPECTED
11825                                     ;DATA WITH DATA READ FROM
11826                                     ;RHCS3

```

```

11827 034740 001401          BEQ      65$      :BRANCH IF GOOD
11828 034742 104124          ERROR    124
11829                                     :AFTER SETTING "CLR" BIT #5
11830                                     :IN RHCS2 TO INIT THE RH
11831                                     :A THREE WORD WRITE REVERSE FROM AN
11832                                     :ODD WORD BOUNDARY WAS DONE
11833                                     :THEN
11834                                     :RHCS3 SHOULD HAVE 0
11835                                     :BUT CONTAINED WHAT IS
11836                                     :GIVEN IN BAD RHCS3
11837 034744          65$:
11838 034744 042777 000076 147106 BIC      #76, @RHCS1 :CLEAR FUNCTION BITS
11839                                     :CHECK THAT RHCS1 HAS SC!RDY!DVA
11840 034752 012737 104200 001124 MOV      #SC!RDY!DVA, $GDDAT ;GET GOOD = 104200
11841
11842 034760 017737 147074 001126 MOV      @RHCS1, $BDDAT ;READ RHCS1 FOR COMPARISON
11843 034766 023737 001124 001126 CMP      $GDDAT, $BDDAT ;COMPARE EXPECTED
11844                                     :DATA WITH DATA READ FROM
11845                                     :RHCS1
11846 034774 001401          BEQ      67$      :BRANCH IF GOOD
11847 034776 104125          ERROR    125
11848                                     :AFTER SETTING "CLR" BIT #5
11849                                     :IN RHCS2 TO INIT THE RH
11850                                     :A THREE WORD WRITE REVERSE FROM AN
11851                                     :ODD WORD BOUNDARY WAS DONE
11852                                     :THEN
11853                                     :RHCS1 SHOULD HAVE SC!RDY!DVA
11854                                     :=104200
11855                                     :BUT CONTAINED WHAT IS
11856                                     :GIVEN IN BAD RHCS1
11857 035000          67$:
11858                                     :CHECK THAT RHCS2 HAS IR!OR
11859 035000 012737 000300 001124 MOV      #IR!OR, $GDDAT ;GET GOOD = 300
11860 035006 053737 005010 001124 BIS      UNIT, $GDDAT ;INCLUDE UNIT NUMBER
11861
11862 035014 017737 147050 001126 MOV      @RHCS2, $BDDAT ;READ RHCS2 FOR COMPARISON
11863 035022 023737 001124 001126 CMP      $GDDAT, $BDDAT ;COMPARE EXPECTED
11864                                     :DATA WITH DATA READ FROM
11865                                     :RHCS2
11866 035030 001401          BEQ      69$      :BRANCH IF GOOD
11867 035032 104126          ERROR    126
11868                                     :AFTER SETTING "CLR" BIT #5
11869                                     :IN RHCS2 TO INIT THE RH
11870                                     :A THREE WORD WRITE REVERSE FROM AN
11871                                     :ODD WORD BOUNDARY WAS DONE
11872                                     :THEN
11873                                     :RHCS2 SHOULD HAVE IR!OR
11874                                     :=300
11875                                     :TOGETHER WITH UNIT NUMBER
11876                                     :BUT CONTAINED WHAT IS
11877                                     :GIVEN IN BAD RHCS2
11878 035034          69$:
11879                                     :CHECK THAT RHWC HAS 0
11880 035034 012737 000000 001124 MOV      #0, $GDDAT ;GET GOOD = 0
11881
11882 035042 017737 147014 001126 MOV      @RHWC, $BDDAT ;READ RHWC FOR COMPARISON

```

```

11883 035050 023737 001124 001126      CMP      $GDDAT,$BDDAT      ;COMPARE EXPECTED
11884                                     ;DATA WITH DATA READ FROM
11885                                     ;RHC
11886 035056 001401      BEQ      71$                ;BRANCH IF GOOD
11887 035060 104131      ERROR    131
11888                                     ;AFTER SETTING "CLR" BIT #5
11889                                     ;IN RHCS2 TO INIT THE RH
11890                                     ;A THREE WORD WRITE REVERSE FROM AN
11891                                     ;ODD WORD BOUNDARY WAS DONE
11892                                     ;THEN
11893                                     ;RHC SHOULD HAVE 0
11894                                     ;BUT CONTAINED WHAT IS
11895                                     ;GIVEN IN BAD RHC
11896 035062      71$:
11897
11898
11899
11900                                     ;*****
11901                                     ;*TEST 61      TEST DBL (RHCS3 BIT #10) TEST R
11902
11903                                     ;*
11904                                     ;*      CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
11905                                     ;*      SET UP FOR A TWO WORD WRITE REVERSE FROM AN EVEN WORD BOUNDARY
11906                                     ;*      WITH BAI IN RHCS2 BIT #3 SET
11907                                     ;*      DO A TWO WORD WRITE REVERSE FROM AN EVEN WORD BOUNDARY
11908                                     ;*      THIS SHOULD NOT SET RHCS3 BIT #10 DBL
11909                                     ;*      CHECK RHCS3
11910                                     ;*      CHECK RHCS1, RHCS2, RHBA, RHBAE, RHC
11911                                     ;*****
11912 035062 000004      †ST61: SCOPE
11913 035064 012706 001000      MOV      #STACK,SP      ;RESET STACK
11914 035070 012737 000061 004656      MOV      #61,‡#†STNM    ;SAVE TEST NUMBER
11915
11916 035076 004737 040322      JSR      PC,‡#CLDISK    ;GIVE RH INITIALIZE
11917                                     ;SETUP UNIT NUBER
11918                                     ;CLEAR RHC AND FUNCTION BITS IN RHCS1
11919
11920                                     ;SET UP FOR A TWO WORD WRITE REVERSE FROM AN EVEN WORD BOUNDARY
11921 035102 012701 000003      MOV      #3,R1          ;COUNT OF THREE
11922 035106 012702 003000      MOV      #BFEVEN,R2    ;START THREE WORD BUFFER
11923                                     ;FROM AN EVEN WORD BOUNDARY
11924 035112 012722 052525      1$: MOV      #52525,(R2)+ ;MOVE THREE 52525 INTO BUFFER
11925 035116 005301      DEC      R1            ;COUNT
11926 035120 001374      BNE     1$            ;BRANCH IF THREE NOT DONE
11927 035122 012777 003000 146734      MOV      #BFEVEN,‡RHBA ;SET BUS ADDRESS TO START
11928                                     ;FROM AN EVEN WORD BOUNDARY
11929 035130 012777 177776 146724      MOV      #-2,‡RHC      ;WORD COUNT TWO
11930 035136 052777 000010 146724      BIS     #BAI,‡RHCS2    ;SET BAI IN RHCS2
11931 035144 004077 147470      JSR      ‡R0,‡COMND    ;GO TO DO COMMAND
11932 035150 004204      REVWRT                ;REVERSEWRITE DATA
11933
11934
11935                                     ;CHECK THAT RHCS3 HAS 0
11936 035152 012737 000000 001124      MOV      #0,$GDDAT     ;GET GOOD = 0
11937
11938 035160 017737 146746 001126      MOV      ‡RHCS3,$BDDAT ;READ RHCS3 FOR COMPARISON

```





```

11995                                     ;GIVEN IN BAD RHCS2
11996 035270                               69$:
11997                                     ;CHECK THAT RHWC HAS 0
11998 035270 012737 000000 001124         MOV     #0,$GDDAT      ;GET GOOD = 0
11999
12000 035276 017737 146560 001126         MOV     @RHWC,$BDDAT   ;READ RHWC FOR COMPARISON
12001 035304 023737 001124 001126         CMP     $GDDAT,$BDDAT ;COMPARE EXPECTED
12002                                     ;DATA WITH DATA READ FROM
12003                                     ;RHWC
12004 035312 001401                         BEQ     71$           ;BRANCH IF GOOD
12005 035314 104131                         ERROR   131
12006
12007                                     ;AFTER SETTING "CLR" BIT #5
12008                                     ;IN RHCS2 TO INIT THE RH
12009                                     ;A TWO WORD WRITE REVERSE FROM AN
12010                                     ;EVEN WORD BOUNDARY WAS DONE
12011                                     ;WITH BAI IN RHCS2 SET
12012                                     ;THEN
12013                                     ;RHWC SHOULD HAVE 0
12014                                     ;BUT CONTAINED WHAT IS
12015 035316                               71$:
12016
12017
12018
12019
12020                                     ;*****
12021                                     ;*TEST 62      TEST DBL (RHCS3 BIT #10) TEST S
12022
12023                                     ;*
12024                                     ;* CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
12025                                     ;* SET UP FOR A NINE WORD READ REVERSE FROM AN EVEN WORD BOUNDARY
12026                                     ;* DO A NINE WORD READ REVERSE FROM AN EVEN WORD BOUNDARY
12027                                     ;* THIS SHOULD SET RHCS3 BIT #10 DBL
12028                                     ;* CHECK RHCS3
12029                                     ;* CHECK RHCS1,RHCS2,RHBA,RHBAE,RHWC
12030                                     ;* IF MAG. TAPE NOT USED. THIS TEST IS NOT DONE
12031                                     ;*****
12032 035316 000004                               TST62: SCOPE
12033 035320 012706 001000                       MOV     #STACK,SP     ;RESET STACK
12034 035324 012737 000062 004656             MOV     #62,@TSTNM    ;SAVE TEST NUMBER
12035
12036 035332 004737 040322                       JSR     PC,@#CLDISK   ;GIVE RH INITIALIZE
12037                                     ;SETUP UNIT NUBER
12038                                     ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
12039 035336 022737 041710 004640             CMP     #CMNDTM,COMND ;IS TU THERE
12040
12041 035344 001103                               BNE     TST63      ;BRANCH IF TU NOT THERE
12042
12043
12044                                     ;SET UP FOR A NINE WORD READ REVERSE FROM AN EVEN WORD BOUNDARY
12045 035346 012701 000003                       MOV     #3,R1         ;COUNT OF THREE
12046 035352 012702 003000                       MOV     #BF-EVEN,R2   ;START THREE WORD BUFFER
12047                                     ;FROM AN EVEN WORD BOUNDARY
12048 035356 012722 052525                               1$: MOV     #52525,(R2)+ ;MOVE THREE 52525 INTO BUFFER
12049 035362 005301                               DEC     R1            ;COUNT
12050 035364 001374                               BNE     1$           ;BRANCH IF THREE NOT DONE

```

MAINDEC-11-DERHAC-A  
DERHAC.P11 T62MACY11 27(732) 09-SEP-76 07:55 PAGE 224  
TEST DBL (RHCS3 BIT #10) TEST 5

```

12051 035366 012777 003000 146470      MOV      #BFEVEN,@RHBA      ;SET BUS ADDRESS TO START
12052                                     ;FROM AN EVEN WORD BOUNDARY
12053 035374 012777 177767 146460      MOV      #-9.,@RHWC ;WORD COUNT NINE
12054
12055 035402 004077 147232               JSR      RD,@COMND          ;GO TO DO COMMAND
12056 035406 004200                       REVRED                    ;REVERSE READ
12057
12058                                     ;CHECK THAT RHCS3 HAS DBL
12059 035410 012737 002000 001124      MOV      #DBL,$GDDAT      ;GET GOOD = 2000
12060
12061 035416 017737 146510 001126      MOV      @RHCS3,$BDDAT    ;READ RHCS3 FOR COMPARISON
12062 035424 023737 001124 001126      CMP      $GDDAT,$BDDAT    ;COMPARE EXPECTED
12063                                     ;DATA WITH DATA READ FROM
12064                                     ;RHCS3
12065 035432 001401                       BEQ      65$              ;BRANCH IF GOOD
12066 035434 104124                       ERROR    124
12067                                     ;AFTER SETTING "CLR" BIT #5
12068                                     ;IN RHCS2 TO INIT THE RH
12069                                     ;A NINE WORD READ REVERSE FROM AN
12070                                     ;EVEN WORD BOUNDARY WAS DONE
12071                                     ;THEN
12072                                     ;RHCS3 SHOULD HAVE DBL
12073                                     ;=2000
12074                                     ;BUT CONTAINED WHAT IS
12075                                     ;GIVEN IN BAD RHCS3
12076 035436                                65$:
12077 035436 042777 000076 146414      BIC      #76,@RHCS1      ;CLEAR FUNCTION BITS
12078                                     ;CHECK THAT RHCS1 HAS RDY!DVA
12079 035444 012737 004200 001124      MOV      #RDY!DVA,$GDDAT ;GET GOOD = 4200
12080
12081 035452 017737 146402 001126      MOV      @RHCS1,$BDDAT    ;READ RHCS1 FOR COMPARISON
12082 035460 023737 001124 001126      CMP      $GDDAT,$BDDAT    ;COMPARE EXPECTED
12083                                     ;DATA WITH DATA READ FROM
12084                                     ;RHCS1
12085 035466 001401                       BEQ      67$              ;BRANCH IF GOOD
12086 035470 104125                       ERROR    125
12087                                     ;AFTER SETTING "CLR" BIT #5
12088                                     ;IN RHCS2 TO INIT THE RH
12089                                     ;A NINE WORD READ REVERSE FROM AN
12090                                     ;EVEN WORD BOUNDARY WAS DONE
12091                                     ;THEN
12092                                     ;RHCS1 SHOULD HAVE RDY!DVA
12093                                     ;=4200
12094                                     ;BUT CONTAINED WHAT IS
12095                                     ;GIVEN IN BAD RHCS1
12096 035472                                67$:
12097                                     ;CHECK THAT RHCS2 HAS IR
12098 035472 012737 000100 001124      MOV      #IR,$GDDAT      ;GET GOOD = 100
12099 035500 053737 005010 001124      BIS      UNIT,$GDDAT      ;INCLUDE UNIT NUMBER
12100
12101 035506 017737 146356 001126      MOV      @RHCS2,$BDDAT    ;READ RHCS2 FOR COMPARISON
12102 035514 023737 001124 001126      CMP      $GDDAT,$BDDAT    ;COMPARE EXPECTED
12103                                     ;DATA WITH DATA READ FROM
12104                                     ;RHCS2
12105 035522 001401                       BEQ      69$              ;BRANCH IF GOOD
12106 035524 104126                       ERROR    126

```

; AFTER SETTING "CLR" BIT #5  
; IN RHCS2 TO INIT THE RH  
; A NINE WORD READ REVERSE FROM AN  
; EVEN WORD BOUNDARY WAS DONE  
; THEN  
; RHCS2 SHOULD HAVE IR  
; =100  
; TOGETHER WITH UNIT NUMBER  
; BUT CONTAINED WHAT IS  
; GIVEN IN BAD RHCS2

12107  
12108  
12109  
12110  
12111  
12112  
12113  
12114  
12115  
12116  
12117 035526 69\$:  
12118  
12119 035526 012737 000000 001124  
12120  
12121 035534 017737 146322 001126  
12122 035542 023737 001124 001126  
12123  
12124  
12125 035550 001401  
12126 035552 104131

; CHECK THAT RHWC HAS 0  
MOV #0, \$GDDAT  
MOV @RHWC, \$BDDAT  
CMP \$GDDAT, \$BDDAT  
BEQ 71\$  
ERROR 131

; GET GOOD = 0  
; READ RHWC FOR COMPARISON  
; COMPARE EXPECTED  
; DATA WITH DATA READ FROM  
; RHWC  
; BRANCH IF GOOD

; AFTER SETTING "CLR" BIT #5  
; IN RHCS2 TO INIT THE RH  
; A NINE WORD READ REVERSE FROM AN  
; EVEN WORD BOUNDARY WAS DONE  
; THEN  
; RHWC SHOULD HAVE 0  
; BUT CONTAINED WHAT IS  
; GIVEN IN BAD RHWC

12127  
12128  
12129  
12130  
12131  
12132  
12133  
12134  
12135 035554 71\$:  
12136  
12137  
12138  
12139  
12140  
12141

\*\*\*\*\*  
; \*TEST 63 TEST DBL (RHCS3 BIT #10) TEST T

12142  
12143  
12144  
12145  
12146  
12147  
12148  
12149

; \* CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)  
; \* SET UP FOR A TEN WORD READ REVERSE FROM AN EVEN WORD BOUNDARY  
; \* DO A TEN WORD READ REVERSE FROM AN EVEN WORD BOUNDARY  
; \* THIS SHOULD NOT SET RHCS3 BIT #10 DBL  
; \* CHECK RHCS3  
; \* CHECK RHCS1, RHCS2, RHBA, RHBAE, RHWC  
; \* IF MAG. TAPE NOT USED. THIS TEST IS NOT DONE

12150  
12151 035554 000004  
12152 035556 012706 001000  
12153 035562 012737 000063 004656  
12154  
12155 035570 004737 040322  
12156  
12157  
12158 035574 022737 041710 004640  
12159  
12160 035602 001103  
12161  
12162

\*\*\*\*\*  
; \*TEST 63: SCOPE  
MOV #STACK, SP ; RESET STACK  
MOV #63, @TSTNM ; SAVE TEST NUMBER  
JSR PC, @CLDISK ; GIVE RH INITIALIZE  
; SETUP UNIT NUMBER  
; CLEAR RHWC AND FUNCTION BITS IN RHCS:  
CMP #CMNDTM, COMND ; IS TU THERE  
BNE TST64 ; BRANCH IF TU NOT THERE

```

12163 ;SET UP FOR A TEN WORD READ REVERSE FROM AN EVEN WORD BOUNDARY
12164 035604 012701 000003 MOV #3,R1 ;COUNT OF THREE
12165 035610 012702 003000 MOV #8FEVEN,R2 ;START THREE WORD BUFFER
12166 ;FROM AN EVEN WORD BOUNDARY
12167 035614 012722 052525 1$: MOV #52525,(R2)+ ;MOVE THREE 52525 INTO BUFFER
12168 035620 005301 DEC R1 ;COUNT
12169 035622 001374 BNE 1$ ;BRANCH IF THREE NOT DONE
12170 035624 012777 003000 146232 MOV #8FEVEN,@RHBA ;SET BUS ADDRESS TO START
12171 ;FROM AN EVEN WORD BOUNDARY
12172 035632 012777 177766 146222 MOV #-10.,@RHWC ;WORD COUNT TEN
12173
12174 035640 004077 146774 JSR RO,@COMND ;GO TO DO COMMAND
12175 035644 004200 REVRED ;REVERSE READ
12176
12177 ;CHECK THAT RHCS3 HAS 0
12178 035646 012737 000000 001124 MOV #0,$GDDAT ;GET GOOD = 0
12179
12180 035654 017737 146252 001126 MOV @RHCS3,$BDDAT ;READ RHCS3 FOR COMPARISON
12181 035662 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
12182 ;DATA WITH DATA READ FROM
12183 ;RHCS3
12184 035670 001401 BEQ 65$ ;BRANCH IF GOOD
12185 035672 104124 ERROR 124
12186 ;AFTER SETTING "CLR" BIT #5
12187 ;IN RHCS2 TO INIT THE RH
12188 ;A TEN WORD READ REVERSE FROM AN
12189 ;EVEN WORD BOUNDARY WAS DONE
12190 ;THEN
12191 ;RHCS3 SHOULD HAVE 0
12192 ;BUT CONTAINED WHAT IS
12193 ;GIVEN IN BAD RHCS3
12194 035674 65$:
12195 035674 042777 000076 146156 BIC #76,@RHCS1 ;CLEAR FUNCTION BITS
12196 ;CHECK THAT RHCS1 HAS RDY!DVA
12197 035702 012737 004200 001124 MOV #RDY!DVA,$GDDAT ;GET GOOD = 4200
12198
12199 035710 017737 146144 001126 MOV @RHCS1,$BDDAT ;READ RHCS1 FOR COMPARISON
12200 035716 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
12201 ;DATA WITH DATA READ FROM
12202 ;RHCS1
12203 035724 001401 BEQ 67$ ;BRANCH IF GOOD
12204 035726 104125 ERROR 125
12205 ;AFTER SETTING "CLR" BIT #5
12206 ;IN RHCS2 TO INIT THE RH
12207 ;A TEN WORD READ REVERSE FROM AN
12208 ;EVEN WORD BOUNDARY WAS DONE
12209 ;THEN
12210 ;RHCS1 SHOULD HAVE RDY!DVA
12211 ;=4200
12212 ;BUT CONTAINED WHAT IS
12213 ;GIVEN IN BAD RHCS1
12214 035730 67$:
12215 ;CHECK THAT RHCS2 HAS IR
12216 035730 012737 000100 001124 MOV #IR,$GDDAT ;GET GOOD = 100
12217 035736 053737 005010 001124 BIS UNIT,$GDDAT ;INCLUDE UNIT NUMBER
12218

```



```

12275                                     ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
12276 036032 022737 041710 004640      CMP    #CMNDTM,COMND ;IS TU THERE
12277                                     ;
12278 036040 001103                       SNE    TST65 ;BRANCH IF TU NOT THERE
12279                                     ;
12280                                     ;
12281                                     ;SET UP FOR A TEN WORD READ REVERSE FROM AN ODD WORD BOUNDARY
12282 036042 012701 000003                MOV    #3,R1 ;COUNT OF THREE
12283 036046 012702 003002                MOV    #BFODD,R2 ;START THREE WORD BUFFER
12284                                     ;FROM AN ODD WORD BOUNDARY
12285 036052 012722 052525                1$: MOV    #52525,(R2)+ ;MOVE THREE 52525 INTO BUFFER
12286 036056 005301                       DEC    R1 ;COUNT
12287 036060 001374                       BNE    1$ ;BRANCH IF THREE NOT DONE
12288 036062 012777 003002 145774        MOV    #BFODD,@RHBA ;SET BUS ADDRESS TO START
12289                                     ;FROM AN ODD WORD BOUNDARY
12290 036070 012777 177766 145764        MOV    #-10.,@RHWC ;WORD COUNT TEN
12291                                     ;
12292 036076 004077 146536                 JSR    RD,@COMND ;GO TO DO COMMAND
12293 036102 004200                       REVRED ;REVERSE READ
12294                                     ;
12295                                     ;CHECK THAT RHCS3 HAS DBL
12296 036104 012737 002000 001124        MOV    #DBL,$GDDAT ;GET GOOD = 2000
12297                                     ;
12298 036112 017737 146014 001126        MOV    @RHCS3,$BDDAT ;READ RHCS3 FOR COMPARISON
12299 036120 023737 001124 001126        CMP    $GDDAT,$BDDAT ;COMPARE EXPECTED
12300                                     ;DATA WITH DATA READ FROM
12301                                     ;RHCS3
12302 036126 001401                       BEQ    65$ ;BRANCH IF GOOD
12303 036130 104124                       ERROR  124
12304                                     ;
12305                                     ;AFTER SETTING "CLR" BIT #5
12306                                     ;IN RHCS2 TO INIT THE RH
12307                                     ;A TEN WORD READ REVERSE FROM AN
12308                                     ;ODD WORD BOUNDARY WAS DONE
12309                                     ;THEN
12310                                     ;RHCS3 SHOULD HAVE DBL
12311                                     ;=2000
12312                                     ;BUT CONTAINED WHAT IS
12313                                     ;GIVEN IN BAD RHCS3
12313 036132 042777 000076 145720        65$: BIC    #76,@RHCS1 ;CLEAR FUNCTION BITS
12314 036132 042777 000076 145720        ;CHECK THAT RHCS1 HAS RDY!DVA
12315                                     ;
12316 036140 012737 004200 001124        MOV    #RDY!DVA,$GDDAT ;GET GOOD = 4200
12317                                     ;
12318 036146 017737 145706 001126        MOV    @RHCS1,$BDDAT ;READ RHCS1 FOR COMPARISON
12319 036154 023737 001124 001126        CMP    $GDDAT,$BDDAT ;COMPARE EXPECTED
12320                                     ;DATA WITH DATA READ FROM
12321                                     ;RHCS1
12322 036162 001401                       BEQ    67$ ;BRANCH IF GOOD
12323 036164 104125                       ERROR  125
12324                                     ;
12325                                     ;AFTER SETTING "CLR" BIT #5
12326                                     ;IN RHCS2 TO INIT THE RH
12327                                     ;A TEN WORD READ REVERSE FROM AN
12328                                     ;ODD WORD BOUNDARY WAS DONE
12329                                     ;THEN
12330                                     ;RHCS1 SHOULD HAVE RDY!DVA
12330                                     ;=4200

```

```

12331                                     ;BUT CONTAINED WHAT IS
12332                                     ;GIVEN IN BAD RHCS1
12333 036166                               67$:
12334                                     ;CHECK THAT RHCS2 HAS IR
12335 036166 012737 000100 001124        MOV  #IR,$GDDAT ;GET GOOD = 100
12336 036174 053737 005010 001124        BIS  UNIT,$GDDAT ;INCLUDE UNIT NUMBER
12337
12338 036202 017737 145662 001126        MOV  @RHCS2,$BDDAT ;READ RHCS2 FOR COMPARISON
12339 036210 023737 001124 001126        CMP  $GDDAT,$BDDAT ;COMPARE EXPECTED
12340                                     ;DATA WITH DATA READ FROM
12341                                     ;RHCS2
12342 036216 001401                          BEQ  69$          ;BRANCH IF GOOD
12343 036220 104126                          ERROR 126
12344                                     ;AFTER SETTING "CLR" BIT #5
12345                                     ;IN RHCS2 TO INIT THE RH
12346                                     ;A TEN WORD READ REVERSE FROM AN
12347                                     ;ODD WORD BOUNDARY WAS DONE
12348                                     ;THEN
12349                                     ;RHCS2 SHOULD HAVE IR
12350                                     ;=100
12351                                     ;TOGETHER WITH UNIT NUMBER
12352                                     ;BUT CONTAINED WHAT IS
12353                                     ;GIVEN IN BAD RHCS2
12354 036222                               69$:
12355                                     ;CHECK THAT RHWC HAS 0
12356 036222 012737 000000 001124        MOV  #0,$GDDAT ;GET GOOD = 0
12357
12358 036230 017737 145626 001126        MOV  @RHWC,$BDDAT ;READ RHWC FOR COMPARISON
12359 036236 023737 001124 001126        CMP  $GDDAT,$BDDAT ;COMPARE EXPECTED
12360                                     ;DATA WITH DATA READ FROM
12361                                     ;RHWC
12362 036244 001401                          BEQ  71$          ;BRANCH IF GOOD
12363 036246 104131                          ERROR 131
12364                                     ;AFTER SETTING "CLR" BIT #5
12365                                     ;IN RHCS2 TO INIT THE RH
12366                                     ;A TEN WORD READ REVERSE FROM AN
12367                                     ;ODD WORD BOUNDARY WAS DONE
12368                                     ;THEN
12369                                     ;RHWC SHOULD HAVE 0
12370                                     ;BUT CONTAINED WHAT IS
12371                                     ;GIVEN IN BAD RHWC
12372 036250                               71$:
12373
12374
12375
12376                                     ;*****
12377                                     ;*TEST 65      TEST DBL (RHCS3 BIT #10) TEST V
12378
12379                                     ;*
12380                                     ;*      CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
12381                                     ;*      SET UP FOR A ELEVEN WORD READ REVERSE FROM AN EVEN WORD BOUNDARY
12382                                     ;*      DO A ELEVEN WORD READ REVERSE FROM AN EVEN WORD BOUNDARY
12383                                     ;*      THIS SHOULD SET RHCS3 BIT #10 DBL
12384                                     ;*      CHECK RHCS3
12385                                     ;*      CHECK RHCS1,RHCS2,RHBA,RHBAE,RHWC
12386                                     ;*      IF MAG. TAPE NOT USED THIS TEST IS NOT DONE

```

```

12387
12388 036250 000004
12389 036252 012706 001000
12390 036256 012737 000065 004656
12391
12392 036264 004737 040322
12393
12394
12395 036270 022737 041710 004640
12396
12397 036276 001103
12398
12399
12400
12401 036300 012701 000003
12402 036304 012702 003000
12403
12404 036310 012722 052525
12405 036314 005301
12406 036316 001374
12407 036320 012777 003000 145536
12408
12409 036326 012777 177765 145526
12410
12411 036334 004077 146300
12412 036340 004200
12413
12414
12415 036342 012737 002000 001124
12416
12417 036350 017737 145556 001126
12418 036356 023737 001124 001126
12419
12420
12421 036364 001401
12422 036366 104124
12423
12424
12425
12426
12427
12428
12429
12430
12431
12432 036370
12433 036370 042777 000076 145462
12434
12435 036376 012737 004200 001124
12436
12437 036404 017737 145450 001126
12438 036412 023737 001124 001126
12439
12440
12441 036420 001401
12442 036422 104125

```

```

:*****
†T65: SCOPE
MOV #STACK,SP ;RESET STACK
MOV #65,‡†STNM ;SAVE TEST NUMBER
JSR PC,‡‡CLDISK ;GIVE RH INITIALIZE
;SETUP UNIT NUBER
;CLEAR RHWC AND FUNCTION BITS IN RHCS1
CMP #CMNDTM,COMND ;IS TU THERE
BNE TST66 ;BRANCH IF TU NOT THERE

;SET UP FOR A ELEVEN WORD READ REVERSE FROM AN EVEN WORD BOUNDARY
MOV #3,R1 ;COUNT OF THREE
MOV #BFEVEN,R2 ;START THREE WORD BUFFER
;FROM AN EVEN WORD BOUNDARY
1$: MOV #52525,(R2)+ ;MOVE THREE 52525 INTO BUFFER
DEC R1 ;COUNT
BNE 1$ ;BRANCH IF THREE NOT DONE
MOV #BFEVEN,‡RHBA ;SET BUS ADDRESS TO START
;FROM AN EVEN WORD BOUNDARY
MOV #-11.,‡RHWC ;WORD COUNT ELEVEN
JSR RO,‡COMND ;GO TO DO COMMAND
REVRED ;REVERSE READ

;CHECK THAT RHCS3 HAS DBL
MOV #DBL,$GDDAT ;GET GOOD = 2000
MOV ‡RHCS3,$BDDAT ;READ RHCS3 FOR COMPARISON
CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
;DATA WITH DATA READ FROM
;RHCS3
BEQ 65$ ;BRANCH IF GOOD
ERROR 124

;AFTER SETTING "CLR" BIT #5
;IN RHCS2 TO INIT THE RH
;A ELEVEN WORD READ REVERSE FROM AN
;EVEN WORD BOUNDARY WAS DONE
;THEN
;RHCS3 SHOULD HAVE DBL
;=2000
;BUT CONTAINED WHAT IS
;GIVEN IN BAD RHCS3
65$: BIC #76,‡RHCS1 ;CLEAR FUNCTION BITS
;CHECK THAT RHCS1 HAS RDY!DVA
MOV #RDY!DVA,$GDDAT ;GET GOOD = 4200
MOV ‡RHCS1,$BDDAT ;READ RHCS1 FOR COMPARISON
CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
;DATA WITH DATA READ FROM
;RHCS1
BEQ 67$ ;BRANCH IF GOOD
ERROR 125

```



```

12443 ; AFTER SETTING "CLR" BIT #5
12444 ; IN RHCS2 TO INIT THE RH
12445 ; A ELEVEN WORD READ REVERSE FROM AN
12446 ; EVEN WORD BOUNDARY WAS DONE
12447 ; THEN
12448 ; RHCS1 SHOULD HAVE RDY!DVA
12449 ; =4200
12450 ; BUT CONTAINED WHAT IS
12451 ; GIVEN IN BAD RHCS1
12452 036424 67$:
12453 ; CHECK THAT RHCS2 HAS IR
12454 036424 012737 000100 001124 MOV #IR,$GDDAT ; GET GOOD = 100
12455 036432 053737 005010 001124 BIS UNIT,$GDDAT ; INCLUDE UNIT NUMBER
12456
12457 036440 017737 145424 001126 MOV @RHCS2,$BDDAT ; READ RHCS2 FOR COMPARISON
12458 036446 023737 001124 001126 CMP $GDDAT,$BDDAT ; COMPARE EXPECTED
12459 ; DATA WITH DATA READ FROM
12460 ; RHCS2
12461 036454 001401 BEQ 69$ ; BRANCH IF GOOD
12462 036456 104126 ERROR 126
12463 ; AFTER SETTING "CLR" BIT #5
12464 ; IN RHCS2 TO INIT THE RH
12465 ; A ELEVEN WORD READ REVERSE FROM AN
12466 ; EVEN WORD BOUNDARY WAS DONE
12467 ; THEN
12468 ; RHCS2 SHOULD HAVE IR
12469 ; =100
12470 ; TOGETHER WITH UNIT NUMBER
12471 ; BUT CONTAINED WHAT IS
12472 ; GIVEN IN BAD RHCS2
12473 036460 69$:
12474 ; CHECK THAT RHWC HAS 0
12475 036460 012737 000000 001124 MOV #0,$GDDAT ; GET GOOD = 0
12476
12477 036466 017737 145370 001126 MOV @RHWC,$BDDAT ; READ RHWC FOR COMPARISON
12478 036474 023737 001124 001126 CMP $GDDAT,$BDDAT ; COMPARE EXPECTED
12479 ; DATA WITH DATA READ FROM
12480 ; RHWC
12481 036502 001401 BEQ 71$ ; BRANCH IF GOOD
12482 036504 104131 ERROR 131
12483 ; AFTER SETTING "CLR" BIT #5
12484 ; IN RHCS2 TO INIT THE RH
12485 ; A ELEVEN WORD READ REVERSE FROM AN
12486 ; EVEN WORD BOUNDARY WAS DONE
12487 ; THEN
12488 ; RHWC SHOULD HAVE 0
12489 ; BUT CONTAINED WHAT IS
12490 ; GIVEN IN BAD RHWC
12491 036506 71$:
12492
12493
12494
12495 ; *****
12496 ; *TEST 66 TEST DBL (RHCS3 BIT #10) TEST W
12497
12498 ; * CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)

```

```

12499      ;*      SET UP FOR A ELEVEN WORD READ REVERSE FROM AN ODD WORD BOUNDARY
12500      ;*      DO A ELEVEN WORD READ REVERSE FROM AN ODD WORD BOUNDARY
12501      ;*      THIS SHOULD NOT SET RHCS3 BIT #10 DBL
12502      ;*      CHECK RHCS3
12503      ;*      CHECK RHCS1, RHCS2, RHBA, RHBAE, RHC
12504      ;*      IF MAG. TAPE NOT USED. THIS TEST IS NOT DONE
12505
12506      ;*****
12507      036506 000004      †T66: SCOPE
12508      036510 012706 001000      MOV      #STACK, SP      ;RESET STACK
12509      036514 012737 000066 004656      MOV      #66, †STNM      ;SAVE TEST NUMBER
12510
12511      036522 004737 040322      JSR      PC, †CLDISK      ;GIVE RH INITIALIZE
12512      ;SETUP UNIT NUBER
12513      ;CLEAR RHC AND FUNCTION BITS IN RHCS1
12514      036526 022737 041710 004640      CMP      #CMNDM, COMND      ;IS TU THERE
12515
12516      036534 001103      BNE      TST67      ;BRANCH IF TU NOT THERE
12517
12518
12519      ;SET UP FOR A ELEVEN WORD READ REVERSE FROM AN ODD WORD BOUNDARY
12520      036536 012701 000003      MOV      #3, R1      ;COUNT OF THREE
12521      036542 012702 003002      MOV      #BF0DD, R2      ;START THREE WORD BUFFER
12522      ;FROM AN ODD WORD BOUNDARY
12523      036546 012722 052525      1$: MOV      #52525, (R2)+      ;MOVE THREE 52525 INTO BUFFER
12524      036552 005301      DEC      R1      ;COUNT
12525      036554 001374      BNE      1$      ;BRANCH IF THREE NOT DONE
12526      036556 012777 003002 145300      MOV      #BF0DD, †RHBA      ;SET BUS ADDRESS TO START
12527      ;FROM AN ODD WORD BOUNDARY
12528      036564 012777 177765 145270      MOV      #-11., †RHC      ;WORD COUNT ELEVEN
12529
12530      036572 004077 146042      JSR      RO, †COMND      ;GO TO DO COMMAND
12531      036576 004200      REVRED      ;REVERSE READ
12532
12533      ;CHECK THAT RHCS3 HAS 0
12534      036600 012737 000000 001124      MOV      #0, $GDDAT      ;GET GOOD = 0
12535
12536      036606 017737 145320 001126      MOV      †RHCS3, $BDDAT      ;READ RHCS3 FOR COMPARISON
12537      036614 023737 001124 001126      CMP      $GDDAT, $BDDAT      ;COMPARE EXPECTED
12538      ;DATA WITH DATA READ FROM
12539      ;RHCS3
12540      036622 001401      BEQ      65$      ;BRANCH IF GOOD
12541      036624 104124      ERROR      124
12542
12543      ;AFTER SETTING "CLR" BIT #5
12544      ;IN RHCS2 TO INIT THE RH
12545      ;A ELEVEN WORD READ REVERSE FROM AN
12546      ;ODD WORD BOUNDARY WAS DONE
12547      ;THEN
12548      ;RHCS3 SHOULD HAVE 0
12549      ;BUT CONTAINED WHAT IS
12550      ;GIVEN IN BAD RHCS3
12551      036626 042777 000076 145224      65$: BIC      #76, †RHCS1      ;CLEAR FUNCTION BITS
12552      ;CHECK THAT RHCS1 HAS RDY!DVA
12553      036634 012737 004200 001124      MOV      #RDY!DVA, $GDDAT ;GET GOOD = 4200
12554

```

12555	036642	017737	145212	001126	MOV	2RHCS1,\$BDDAT	; READ RHCS1 FOR COMPARISON
12556	036650	023737	001124	001126	CMP	\$GDDAT,\$BDDAT	; COMPARE EXPECTED
12557							; DATA WITH DATA READ FROM
12558							; RHCS1
12559	036656	001401			BEG	67\$	; BRANCH IF GOOD
12560	036660	104125			ERROR	125	
12561							; AFTER SETTING "CLR" BIT #5
12562							; IN RHCS2 TO INIT THE RH
12563							; A ELEVEN WORD READ REVERSE FROM AN
12564							; ODD WORD BOUNDARY WAS DONE
12565							; THEN
12566							; RHCS1 SHOULD HAVE RDY!DVA
12567							; =4200
12568							; BUT CONTAINED WHAT IS
12569							; GIVEN IN BAD RHCS1
12570	036662					67\$:	
12571							; CHECK THAT RHCS2 HAS IR
12572	036662	012737	000100	001124	MOV	#IR,\$GDDAT	; GET GOOD = 100
12573	036670	053737	005010	001124	BIS	UNIT,\$GDDAT	; INCLUDE UNIT NUMBER
12574							
12575	036676	017737	145166	001126	MOV	2RHCS2,\$BDDAT	; READ RHCS2 FOR COMPARISON
12576	036704	023737	001124	001126	CMP	\$GDDAT,\$BDDAT	; COMPARE EXPECTED
12577							; DATA WITH DATA READ FROM
12578							; RHCS2
12579	036712	001401			BEG	69\$	; BRANCH IF GOOD
12580	036714	104126			ERROR	126	
12581							; AFTER SETTING "CLR" BIT #5
12582							; IN RHCS2 TO INIT THE RH
12583							; A ELEVEN WORD READ REVERSE FROM AN
12584							; ODD WORD BOUNDARY WAS DONE
12585							; THEN
12586							; RHCS2 SHOULD HAVE IR
12587							; =100
12588							; TOGETHER WITH UNIT NUMBER
12589							; BUT CONTAINED WHAT IS
12590							; GIVEN IN BAD RHCS2
12591	036716					69\$:	
12592							; CHECK THAT RHWC HAS 0
12593	036716	012737	000000	001124	MOV	#0,\$GDDAT	; GET GOOD = 0
12594							
12595	036724	017737	145132	001126	MOV	2RHWC,\$BDDAT	; READ RHWC FOR COMPARISON
12596	036732	023737	001124	001126	CMP	\$GDDAT,\$BDDAT	; COMPARE EXPECTED
12597							; DATA WITH DATA READ FROM
12598							; RHWC
12599	036740	001401			BEG	71\$	; BRANCH IF GOOD
12600	036742	104131			ERROR	131	
12601							; AFTER SETTING "CLR" BIT #5
12602							; IN RHCS2 TO INIT THE RH
12603							; A ELEVEN WORD READ REVERSE FROM AN
12604							; ODD WORD BOUNDARY WAS DONE
12605							; THEN
12606							; RHWC SHOULD HAVE 0
12607							; BUT CONTAINED WHAT IS
12608							; GIVEN IN BAD RHWC
12609	036744					71\$:	
12610							

12611  
12612  
12613  
12614  
12615  
12616  
12617  
12618  
12619  
12620  
12621  
12622  
12623  
12624  
12625  
12626  
12627  
12628  
12629  
12630  
12631  
12632  
12633  
12634  
12635  
12636  
12637  
12638  
12639  
12640  
12641  
12642  
12643  
12644  
12645  
12646  
12647  
12648  
12649  
12650  
12651  
12652  
12653  
12654  
12655  
12656  
12657  
12658  
12659  
12660  
12661  
12662  
12663  
12664  
12665  
12666

\*\*\*\*\*  
; \*TEST 67 TEST DBL (RHCS3 BIT #10) TEST X

; \* CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)  
; \* SET UP FOR A TEN WORD READ REVERSE FROM AN ODD WORD BOUNDARY  
; \* WITH BAI IN RHCS2 BIT #3 SET  
; \* DO A TEN WORD READ REVERSE FROM AN ODD WORD BOUNDARY  
; \* THIS SHOULD NOT SET RHCS3 BIT #10 DBL  
; \* CHECK RHCS3  
; \* CHECK RHCS1, RHCS2, RHBA, RHBAE, RHCW  
; \* IF MAG. TAPE NOT USED. THIS TEST IS NOT DONE

\*\*\*\*\*  
†ST67: SCOPE

MOV #STACK, SP ; RESET STACK  
MOV #67, †STNM ; SAVE TEST NUMBER  
JSR PC, †CLDISK ; GIVE RH INITIALIZE  
; SETUP UNIT NUMBER  
; CLEAR RHCW AND FUNCTION BITS IN RHCS1  
CMP #CMNDTM, COMND ; IS TU THERE  
BNE TST70 ; BRANCH IF TU NOT THERE

; SET UP FOR A TEN WORD READ REVERSE FROM AN ODD WORD BOUNDARY  
MOV #3, R1 ; COUNT OF THREE  
MOV #BF0DD, R2 ; START THREE WORD BUFFER  
; FROM AN ODD WORD BOUNDARY  
1\$: MOV #52525, (R2)+ ; MOVE THREE 52525 INTO BUFFER  
DEC R1 ; COUNT  
BNE 1\$ ; BRANCH IF THREE NOT DONE  
MOV #BF0DD, †RHBA ; SET BUS ADDRESS TO START  
; FROM AN ODD WORD BOUNDARY

MOV #-10, †RHCW ; WORD COUNT TEN  
BIS #BAI, †RHCS2 ; SET BAI IN RHCS2

JSR RO, †COMND ; GO TO DO COMMAND  
REVRED ; REVERSE READ

; CHECK THAT RHCS3 HAS 0  
MOV #0, \$GDDAT ; GET GOOD = 0

MOV †RHCS3, \$BDDAT ; READ RHCS3 FOR COMPARISON  
CMP \$GDDAT, \$BDDAT ; COMPARE EXPECTED  
; DATA WITH DATA READ FROM  
; RHCS3

BEQ 65\$ ; BRANCH IF GOOD  
ERROR 124

; AFTER SETTING "CLR" BIT #5  
; IN RHCS2 TO INIT THE RH  
; A TEN WORD READ REVERSE FROM AN  
; ODD WORD BOUNDARY WAS DONE  
; WITH BAI IN RHCS2 SET

```

12667                                     ; THEN
12668                                     ; RHCS3 SHOULD HAVE 0
12669                                     ; BUT CONTAINED WHAT IS
12670                                     ; GIVEN IN BAD RHCS3
12671 037072                                     65$:
12672 037072 042777 000076 144760          BIC    #76, @RHCS1      ; CLEAR FUNCTION BITS
12673                                     ; CHECK THAT RHCS1 HAS RDY!DVA
12674 037100 012737 004200 001124          MOV    #RDY!DVA, $GDDAT ; GET GOOD = 4200
12675
12676 037106 017737 144746 001126          MOV    @RHCS1, $BDDAT   ; READ RHCS1 FOR COMPARISON
12677 037114 023737 001124 001126          CMP    $GDDAT, $BDDAT  ; COMPARE EXPECTED
12678                                     ; DATA WITH DATA READ FROM
12679                                     ; RHCS1
12680 037122 001401                               BEQ    67$              ; BRANCH IF GOOD
12681 037124 104125                               ERROR  125
12682
12683                                     ; AFTER SETTING "CLR" BIT #5
12684                                     ; IN RHCS2 TO INIT THE RH
12685                                     ; A TEN WORD READ REVERSE FROM AN
12686                                     ; ODD WORD BOUNDARY WAS DONE
12687                                     ; WITH BAI IN RHCS2 SET
12688                                     ; THEN
12689                                     ; RHCS1 SHOULD HAVE RDY!DVA
12690                                     ; =4200
12691                                     ; BUT CONTAINED WHAT IS
12692 037126                                     67$:
12693                                     ; CHECK THAT RHCS2 HAS IR!BAI
12694 037126 012737 000110 001124          MOV    #IR!BAI, $GDDAT ; GET GOOD = 110
12695 037134 053737 005010 001124          BIS    UNIT, $GDDAT    ; INCLUDE UNIT NUMBER
12696
12697 037142 017737 144722 001126          MOV    @RHCS2, $BDDAT  ; READ RHCS2 FOR COMPARISON
12698 037150 023737 001124 001126          CMP    $GDDAT, $BDDAT ; COMPARE EXPECTED
12699                                     ; DATA WITH DATA READ FROM
12700                                     ; RHCS2
12701 037156 001401                               BEQ    69$              ; BRANCH IF GOOD
12702 037160 104126                               ERROR  126
12703
12704                                     ; AFTER SETTING "CLR" BIT #5
12705                                     ; IN RHCS2 TO INIT THE RH
12706                                     ; A TEN WORD READ REVERSE FROM AN
12707                                     ; ODD WORD BOUNDARY WAS DONE
12708                                     ; WITH BAI IN RHCS2 SET
12709                                     ; THEN
12710                                     ; RHCS2 SHOULD HAVE IR!BAI
12711                                     ; =110
12712                                     ; TOGETHER WITH UNIT NUMBER
12713                                     ; BUT CONTAINED WHAT IS
12714 037162                                     69$:
12715                                     ; CHECK THAT RHWC HAS 0
12716 037162 012737 000000 001124          MOV    #0, $GDDAT      ; GET GOOD = 0
12717
12718 037170 017737 144666 001126          MOV    @RHWC, $BDDAT   ; READ RHWC FOR COMPARISON
12719 037176 023737 001124 001126          CMP    $GDDAT, $BDDAT ; COMPARE EXPECTED
12720                                     ; DATA WITH DATA READ FROM
12721                                     ; RHWC
12722 037204 001401                               BEQ    71$              ; BRANCH IF GOOD

```

12723 037206 104131  
12724  
12725  
12726  
12727  
12728  
12729  
12730  
12731  
12732  
12733 037210  
12734  
12735  
12736  
12737  
12738  
12739  
12740  
12741  
12742  
12743  
12744  
12745  
12746  
12747 037210 000004  
12748 037212 012737 000001 001212  
12749 037220 005037 177776  
12750 037224 104401 037232  
12751 037230 000414  
12752  
12753 037262  
12754 037262 013746 005510  
12755 037266 013702 005506  
12756 037272 162602  
12757 037274 006302  
12758 037276 016246 005436  
12759 037302 104405  
12760 037304 104401 037312  
12761 037310 000410  
12762  
12763 037332  
12764 037332 013746 004060  
12765 037336 104402  
12766 037340 104401 037346  
12767 037344 000421  
12768  
12769 037410  
12770 037410 013746 001112  
12771 037414 104405  
12772 037416 104401 037424  
12773 037422 000402  
12774  
12775 037430  
12776 037430 104401 037436  
12777 037434 000402  
12778

ERROR 131

; AFTER SETTING "CLR" BIT #5  
; IN RHCS2 TO INIT THE RH  
; A TEN WORD READ REVERSE FROM AN  
; ODD WORD BOUNDARY WAS DONE  
; WITH BAI IN RHCS2 SET  
; THEN  
; RHWC SHOULD HAVE 0  
; BUT CONTAINED WHAT IS  
; GIVEN IN BAD RHWC

71\$:

```
*****
*TEST 70      END OF ONE RH
*             THIS IS THE END OF TEST FOR ONE RH
*             IF THERE ARE MORE RH THEN THE PROGRAM
*             JUMPS TO TEST 2 FOR NEXT RH TEST
*             END PASS IS REACHED ONLY AFTER ALL RH ARE COMPLETE
*****
TST70: SCOPE
MOV      #1, $TIMES      ;; DO 1 ITERATION
CLR      PS              ;; REINSTATE PS TO 0
TYPE     ,65$            ;; TYPE ASCIZ STRING
BR       64$            ;; GET OVER THE ASCIZ
;;65$: .ASCIZ <15><12>/END OF TEST FOR RH NO/
64$:    MOV      WORUNT, -(SP)  ;; GET WORKING RH NUMBER LEFT
MOV      TSTUNT, R2        ;; GET TOTAL NO OF RH FOUND
SUB      (SP)+, R2        ;; NO OF RH TESTED
ASL     R2                ;; INDEX FOR RH TESTED
MOV      FRHNM(R2), -(SP)  ;; TYPE OF RH NO.
TYPDS
TYPE     ,67$            ;; TYPE ASCIZ STRING
BR       66$            ;; GET OVER THE ASCIZ
;;67$: .ASCIZ /          BASE /
66$:    MOV      RHCS1, -(SP)  ;; TYPE BASE ADDRESS
TYPDC
TYPE     ,69$            ;; TYPE ASCIZ STRING
BR       68$            ;; GET OVER THE ASCIZ
;;69$: .ASCIZ <15><12>/TOTAL NO OF ERRORS ON THIS PASS/
68$:    MOV      $ERTTL, -(SP)  ;; TYPE TOTAL NO OF ERRORS
TYPDS
TYPE     ,71$            ;; TYPE ASCIZ STRING
BR       70$            ;; GET OVER THE ASCIZ
;;71$: .ASCIZ <15><12>/ /
70$:    TYPE     ,73$            ;; TYPE ASCIZ STRING
BR       72$            ;; GET OVER THE ASCIZ
;;73$: .ASCIZ <15><12>/ /
```



```

12786
12787
12788
12789
12790
12791
12792
12793
12794 037466
12795 037466 000004
12796 037470 005037 001102
12797 037474 005037 001212
12798 037500 005237 001100
12799 037504 042737 100000 001100
12800 037512 005327
12801 037514 000001
12802 037516 003022
12803 037520 012737
12804 037522 000001
12805 037524 037514
12806 037526 104401 037573
12807 037532 013746 001100
12808 037536 104405
12809 037540 104401 037570
12810 037544 013700 000042
12811 037550 001405
12812 037552 000005
12813 037554 004710
12814 037556 000240
12815 037560 000240
12816 037562 000240
12817 037564
12818 037564 000137
12819 037566 010632
12820 037570 377 377 000
12821 037573 015 042412 042116
12822 037600 050040 051501 020123
12823 037606 000043
12824
12825
12826
12827
12828
12829
12830
12831
12832
12833
12834
12835
12836
12837
12838 037610
12839 037610 104401 037616
12840 037614 000433
12841

```

.SBTTL END OF PASS ROUTINE

```

;*****
;*INCREMENT THE PASS NUMBER ($PASS)
;*TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
;*IF THERES A MONITOR GO TO IT
;*IF THERE ISN'T JUMP TO TST2

```

\$EOP:

```

SCOPE
CLR $TSTNM ;:ZERO THE TEST NUMBER
CLR $TIMES ;:ZERO THE NUMBER OF ITERATIONS
INC $PASS ;:INCREMENT THE PASS NUMBER
BIC #100000,$PASS ;:DON'T ALLOW A NEG. NUMBER
DEC (PC)+ ;:LOOP?
$EOPCT: .WORD 1
BGT $DOAGN ;:YES
MOV (PC)+,2(PC)+ ;:RESTORE COUNTER
$ENDCT: .WORD 1
TYPE $ENDMG ;:TYPE "END PASS #"
MOV $PASS,-(SP) ;:SAVE $PASS FOR TYPEOUT
TYPDS ;:GO TYPE--DECIMAL ASCII WITH SIGN
TYPE $ENULL ;:TYPE A NULL CHARACTER
$GET42: MOV 2#42,R0 ;:GET MONITOR ADDRESS
BEQ $DOAGN ;:BRANCH IF NO MONITOR
RESET ;:CLEAR THE WORLD
$ENDAD: JSR PC,(R0) ;:GO TO MONITOR
NOP ;:SAVE ROOM
NOP ;:FOR
NOP ;:ACT11
$DOAGN: JMP 2(PC)+ ;:RETURN
$RTNAD: .WORD TST2
$ENULL: .BYTE -1,-1,0 ;:NULL CHARACTER STRING
$ENDMG: .ASCIZ <15><12>/END PASS #

```

.SBTTL SUBROUTINES

```

;*****
;* THIS ROUTINE HANDLES TIME OUT TRAPS THROUGH LOC. 4
;*****

```

TIEOUT:

```

TYPE 65$ ;:TYPE ASCIZ STRING
BR 64$ ;:GET OVER THE ASCIZ
;:65$: .ASCIZ <15><12>/TIMEOUT OCCURED THROUGH LOCATION 4 FROM PROGRAM PC

```



```

12842 037704
12843 037704 162716 000002
12844 037710 104402
12845 037712 104401 037720
12846 037716 000410
12847
12848 037740
12849 037740 104402
12850 037742 000137 043120
12851
12852
12853
12854
12855
12856
12857
12858
12859
12860
12861
12862 037746
12863 037746 104401 037754
12864 037752 000417
12865
12866 040012
12867 040012 104401 040020
12868 040016 000413
12869
12870 040046
12871 040046 162716 000002
12872 040052 104402
12873 040054 104401 040062
12874 040060 000407
12875
12876 040100
12877 040100 104402
12878 040102 104401 040110
12879 040106 000421
12880
12881 040152
12882 040152 017746 145340
12883 040156 104402
12884 040160 104401 040166
12885 040164 000421
12886
12887 040230
12888 040230 017746 145260
12889 040234 104402
12890 040236 104401 040244
12891 040242 000422
12892
12893 040310
12894 040310 017746 145204
12895 040314 104402
12896 040316 000137 043120
12897

```

```

64$: SUB #2.(SP) ;TYPE PC
      TYPOC
      TYPE ,67$ ;;TYPE ASCIZ STRING
      BR ,66$ ;;GET OVER THE ASCIZ
;;67$: .ASCIZ <15><12>/PSW WAS /
66$: TYPOC
      JMP $SCOPE ;RETURN TO TEST

*****
* THIS ROUTINE HANDLES PARITY ERRORS
*****
PARITY:
        TYPE ,65$ ;;TYPE ASCIZ STRING
        BR ,64$ ;;GET OVER THE ASCIZ
;;65$: .ASCIZ <15><12>/PARITY TRAP THRU VECTOR 114/
64$: TYPE ,67$ ;;TYPE ASCIZ STRING
      BR ,66$ ;;GET OVER THE ASCIZ
;;67$: .ASCIZ <15><12>/FROM PROGRAM PC /
66$: SUB #2.(SP) ;TYPE PC
      TYPOC
      TYPE ,69$ ;;TYPE ASCIZ STRING
      BR ,68$ ;;GET OVER THE ASCIZ
;;69$: .ASCIZ <15><12>/PSW WAS /
68$: TYPOC
      TYPE ,71$ ;;TYPE ASCIZ STRING
      BR ,70$ ;;GET OVER THE ASCIZ
;;71$: .ASCIZ <15><1>/HIGH ERROR ADDRESS REGISTER /
70$: MOV @HERADD,-(SP) ;TYPE HIGH ERROR
      TYPOC
      TYPE ,73$ ;;TYPE ASCIZ STRING
      BR ,72$ ;;GET OVER THE ASCIZ
;;73$: .ASCIZ <15><12>/LOW ERROR ADDRESS REGISTER
72$: MOV @LERADD,-(SP) ;TYPE LOW ERROR
      TYPOC
      TYPE ,75$ ;;TYPE ASCIZ STRING
      BR ,74$ ;;GET OVER THE ASCIZ
;;75$: .ASCIZ <15><12>/MEMORY SYSTEM ERROR REGISTER
74$: MOV @MEMERR,-(SP)
      TYPOC
      JMP $SCOPE

```

12898  
12899  
12900  
12901  
12902  
12903  
12904  
12905  
12906  
12907  
12908

040322 012777 000040 143540  
040330 013777 005010 143532  
040336 042777 000077 143514  
040344 005077 143512  
040350 J0C207

CLDISK: MOV  
MOV  
BIC  
CLR  
RTS

\*CLR,@RHCS2  
@#UNIT,@RHCS2  
\*77,@RHCS1  
@RHWC  
PC

:CLEAR ALL REG.  
:REINSTATE UNIT NO.  
:CLEAR FUNCTION BITS  
:CLEAR RHWC

```

12909
12910
12911
12912
12913      ;THIS CHECKS DEVICE AVAILABLE (DVA) AND READY (RDY) IN RHCS1
12914      ;AND CHECKS MEDIUM ON LINE (MOL), DEVICE PRESENT (DPR), DEVICE READY (DRY) IN RHDS1
12915 040352 000003 PCJSR: 0 ;PC OF JSR
12916
12917
12918
12919
12920 040354 011637 040352 CHECK: MOV (SP), @#PCJSR ;SAVE PC OF JSR+4
12921 040360 162737 000004 040352 SUB #4, @#PCJSR ;GET PC OF JSR
12922 040366 011346 MOV @R3, -(SP) ;GET RHDS1
12923 040370 052716 000100 BIS #VV, (SP) ;DONT CHECK VV BIT
12924 040374 000406 BR CHECKC ;GOTO COMMON CHECK ROUTINE
12925 040376 011637 040352 CHECKT: MOV (SP), @#PCJSR ;SAVE PC OF JSR+4
12926 040402 162737 000004 040352 SUB #4, @#PCJSR ;GET PC OF JSR
12927 040410 011346 MOV @R3, -(SP) ;GET RHDS1 & DO VV CHECK AT 3$
12928 040412 011146 CHECKC: MOV @R1, -(SP) ;GET CSI
12929 040414 042716 173577 BIC #173577, (SP) ;CLEAR UNWANTED BITS
12930 040420 022726 004200 CMP #DVA!RDY, (SP)+ ;RHCS1 SHOULD HAVE DEVICE AVAILABLE
12931
12932 040424 001403 BEQ 3$ ;AND BE READY
12933 040426 011137 001122 MOV @R1, @#$BDADR ;BRANCH IF GOOD
12934 040432 104062 ERROR 62 ;BAD DATA REGISTER (RHCS1)
12935 ;RHCS1 DID NOT HAVE DEVICE
12936 ;AVAILABLE RIGHT AT THE START
12937 ;ALL OTHER BITS SHOULD BE 0
12938 040434 042716 102000 3$: BIC #ATA!LBT, (SP) ;CLEAR UNWANTED BITS
12939 040440 022726 010700 CMP #MOL!DPR!DRY!VV, (SP)+ ;RHDS1 SHOULD HAVE THESE SET
12940 040444 001403 BEQ 7$ ;BRANCH IF GOOD
12941 040446 011337 001122 MOV @R3, @#$BDADR ;BAD DATA IN REGISTER (RHDS1)
12942 040452 104061 ERROR 61 ;RHDS1 HAS SOME BITS OTHER
12943 ;THAN MOL, DRY, DPR, VV SET
12944 040454 000207 7$: RTS PC ;ALL OTHER BITS SHOULD BE 0
12945 ;RETURN TO TEST NO.

```

```

12946
12947
12948
12949
12950
12951
12952
12953
12954
12955
12956
12957
12958
12959
12960
12961
12962
12963 040456 177777
12964
12965 040460
12966 040460 010046
12967 040462 010346
12968 040464 016600 000004
12969 040470 010037 004666
12970 040474 162737 000002 004666
12971 040502 013037 004670
12972 040506 012037 004672
12973 040512 010066 000004
12974
12975
12976
12977 040516 005737 004646
12978 040522 001025
12979 040524 013703 040456
12980 040530 033777 004672 144132 1$:
12981 040536 001042
12982 040540 005303
12983 040542 001372
12984 040544 013703 040456
12985 040550 033777 004672 144112 2$:
12986 040556 001032
12987 040560 005303
12988 040562 001372
12989 040564 017737 144100 001126
12990 040572 104135
12991
12992
12993 040574 000423
12994
12995
12996
12997
12998
12999 040576 013703 040456 4$:
13000 040602 012700 000020
13001 040606 033777 004672 144054 5$:

```

```

:THIS IS A WAIT LOOP WHEN NO P-CLOCK IS AVAILABLE
:NO TIMING IS DONE
:CALL IS
:
:       WAT
:       A           ;POINTER TO REGISTER ADDRESS
:       B           ;BIT WAITED FOR
:R3-IS A TEMPORARY COUNTER
TIMCNT: 177777           ;COUNT FOR WAIT LOOP

WAIT.T:
MOV     R0,-(SP)       ;; PUSH R0 ON STACK
MOV     R3,-(SP)       ;; PUSH R3 ON STACK
MOV     4(SP),R0       ;R0 HAS ADDRESS OF NEXT LOCATION
MOV     R0,@#WAITPC    ;WAT PC +2 IS IN WAITPC
SUB     #2,@#WAITPC    ;WAT PC IS IN WAITPC
MOV     @ (R0)+,@#WAITRE ;WAIT ON REGISTER ADDRESS
MOV     (R0)+,@#WAITBT  ;WAIT ON BIT
MOV     R0,4(SP)      ;RESTORE RETURN ON STACK

;THIS HAS THE TWO COUNT DOWNS FROM 177777
;FOR WAITING FOR A BIT TO SET
TST     WATO           ;WAITING FOR 1?
BNE     4$             ;BRANCH IF WAITING FOR 0
MOV     @#TIMCNT,R3    ;R3 HAS TEMPORARY COUNT
BIT     @#WAITBT,@#WAITRE ;IS REQUIRED BIT THERE
BNE     3$             ;BRANCH IF YES
DEC     R3             ;COUNT IF REQUIRED BIT NOT THERE
BNE     1$             ;
MOV     @#TIMCNT,R3    ;SECOND COUNT DOWN FROM 177777
BIT     @#WAITBT,@#WAITRE ;IS REQUIRED BIT THERE
BNE     3$             ;BRANCH IF YES
DEC     R3             ;COUNT IF REQUIRED BIT NOT THERE
BNE     2$             ;
MOV     @#WAITRE,@#SBDDAT ;REGISTER CONTENTS FOR TYPEOUT
ERROR   135           ;WAS WAITING FOR A BIT TO SET
;BIT IN QUESTION IS IN "BIT WAITED"
;REGISTER IN QUESTION IS IN "REG ADDR"
BR      3$            ;BRANCH OUT

;THIS HAS THE TWO COUNT DOWNS FROM 177777
;FOR WAITING FOR A BIT TO RESET
MOV     @#TIMCNT,R3    ;R3 HAS TEMPORARY COUNT
MOV     #20,R0         ;R0 HAS TEMPORARY COUNT
BIT     @#WAITBT,@#WAITRE ;IS REQUIRED BIT RESET?

```

K03

```
13002 040614 001413      BEQ      3$      ;BRANCH IF YES
13003 040616 005303      DEC      R3      ;COUNT IF REQUIRED BIT HASN'T RESET YET
13004 040620 001372      BNE      5$      ;TRY AGAIN
13005 040622 005300      DEC      R0      ;DECREMENT R0 COUNT
13006 040624 001370      BNE      5$      ;START OVER IF COUNT ISN'T ZERO YET
13007 040626 017737 144036 001126  MOV      @WAITRE,@#$BDDAT ;REGISTER CONTENTS FOR TYPEOUT
13008 040634 104136      ERROR    136     ;WAS WAITING FOR A BIT TO RESET
13009                                ;BIT IN QUESTION IS IN "BIT WAITED"
13010                                ;REGISTER IN QUESTION IS IN "REG ADDR"
13011 040636 017737 144026 001126  MOV      @WAITRE,@#$BDDAT ;REGISTER CONTENTS FOR TYPEOUT
13012 040644                                3$:
13013 040644 012603      MOV      (SP)+,R3 ;;POP STACK INTO R3
13014 040646 012600      MOV      (SP)+,R0 ;;POP STACK INTO R0
13015 040650 000002      RTI                                ;RETURN TO MAIN TEST
```

```

13016
13017
13018
13019
13020
13021
13022
13023
13024
13025
13026
13027
13028
13029
13030
13031
13032
13033
13034
13035
13036
13037
13038 040652 000000
13039 040654
13040 040654 005037 177776
13041 040660 012737 177777 045100
13042 040666 104401 040674
13043 040672 000421
13044
13045 040736
13046 040736 013746 004656
13047 040742 104402
13048 040744 104401 040752
13049 040750 000414
13050
13051 041002
13052 041002 013746 001110
13053 041006 104402
13054 041010 104401 001223
13055 041014 104401 041022
13056 041020 000430
13057
13058 041102
13059 041102 104401 041110
13060 041106 000430
13061
13062 041170
13063 041170 104401 041176
13064 041174 000422
13065
13066 041242
13067 041242 104410
13068 041244 062716 000002
13069 041250 012637 001106
13070 041254 104401 041262
13071 041260 000417

```

```

;HERE IS A DETAILED EXPLANATION OF HOW THE LOOP ON ERROR WORKS.
;ON HITTING AN ERROR IF THE LOOP ON ERROR SWITCH IS SET, THE
;PROGRAM GOES BACK - USUALLY BACK TO THE BEGINNING OF THE TEST.

;WHEN THIS OPERATOR SELECTABLE SCOPE LOOP IS USED THEN THE POINT
;THE PROGRAM GOES BACK TO CAN BE CHANGED.
;THE RESTRICTIONS TO THE POINT WHERE THE PROGRAM CAN GO ARE: -
;1. IT MUST BE WITHIN THE TEST UNDER CONSIDERATION
;2. LOOP ON ERROR SWITCH MUST BE SET
;3. THE ERROR MUST OCCUR WITHIN THE TEST UNDER CONSIDERATION
;IF THE ERROR DOES NOT OCCUR WITHIN THE TEST UNDER CONSIDERATION
;THE PROGRAM WILL REVERT TO NORMAL OPERATION. HOWEVER, IF LOOP ON
;TEST SWITCH IS SET AND THIS OPERATOR SELECTABLE SCOPE LOOP IS USED
;THEN THE PROGRAM WILL LOOP BACK TO THE SELECTED POINT WHEN IT
;COMES TO THE END OF THE TEST UNDER CONSIDERATION.

;AFTER LOOPING FOR SOME TIME IF THE LOOP SWITCH IS PUT DOWN THEN
;NORMAL OPERATION WILL CONTINUE.

```

```

TESTAD: 0 ;FIRST ADDRESS OF TEST
OPERSEL:
CLR PS ;MAKE PROCESSOR STATUS ZERO
MOV #-1, @#PRITEM ;CLEAR PREVIOUS ITEM NUMBER
TYPE ,65$ ;;TYPE ASCIZ STRING
BR ,64$ ;;GET OVER THE ASCIZ
;;65$: .ASCIZ <15><12>/THE PROGRAM WAS IN TEST NUMBER /
64$:
MOV @#TSTNM, -(SP) ;GET READY TO TYPE TEST
TYPOC ;NUMBER
TYPE ,67$ ;;TYPE ASCIZ STRING
BR ,66$ ;;GET OVER THE ASCIZ
;;67$: .ASCIZ <15><12>/THE LOOP BACK PC WAS /
66$:
MOV @#$LPERR, -(SP) ;GET READY TO TYPE LOOP BACK PC
TYPOC
TYPE ,69$ ;;TYPE ASCIZ STRING
BR ,68$ ;;GET OVER THE ASCIZ
;;69$: .ASCIZ <15><12>/SET SWITCH FOR LOOP ON ERROR OR LOOP ON TEST/
68$:
TYPE ,71$ ;;TYPE ASCIZ STRING
BR ,70$ ;;GET OVER THE ASCIZ
;;71$: .ASCIZ <15><12>/TYPE THE FIRST PC OF THE TEST TO BE LOOPED ON/
70$:
TYPE ,73$ ;;TYPE ASCIZ STRING
BR ,72$ ;;GET OVER THE ASCIZ
;;73$: .ASCIZ <15><12>/ FOLLOWED BY A CARRIAGE RETURN /<15><12>
72$:
RDOCT
ADD #2, (SP) ;GET LPADR
MOV (SP)+, @#$LPADR
TYPE ,75$ ;;TYPE ASCIZ STRING
BR ,74$ ;;GET OVER THE ASCIZ

```

```

13072
13073 041320
13074 041320 104401 041326
13075 041324 000440
13076
13077 041426
13078 041426 104410
13079 041430 012637 001110
13080 041434 013746 0C1106
13081 041440 000002
13082
13083
13084
13085
13086
13087
13088
13089
13090
13091
13092
13093
13094
13095 041442
13096 041442 010046
13097 041444 010146
13098 041446 010246
13099 041450 012700 004062
13100 041454 012701 000002
13101 041460 012702 000010
13102 041464 013021
13103 041466 005302
13104 041470 001375
13105 041472 012602
13106 041474 012601
13107 041476 012600
13108 041500 000207
  
```

```

::75$: .ASCIZ <15><12>/TYPE THE PC WHERE YOU WANT/
74$:
TYPE ,77$ ;;TYPE ASCIZ STRING
BR 76$ ;;GET OVER THE ASCIZ
::77$: .ASCIZ <15><12>/ THE PROGRAM TO LOOP BACK TO FOLLOWED BY A CARRIAGE RETURN /<15
76$:
RDOCT
MOV (SP)+, @#SLPERR ;GET LPERR
MOV @#SLPADR, -(SP)
RTI
  
```

```

;THIS SAVES THE CONTENTS OF ALL HARDWARE REGISTERS
;IN MEMORY LOCATIONS TAGED FROM "WC" TO "EC2"
;THIS IS DONE SO THAT COMPARES ARE DONE WITH SAVED LOCATIONS
;AND NOT THE REGISTERS THEMSELVES. THIS WILL MAKE
;ERROR PRINTOUTS FOR GOOD AND BAD DATA ALWAYS DIFFRENT
  
```

```

PUTREG:
MOV R0, -(SP) ;;PUSH R0 ON STACK
MOV R1, -(SP) ;;PUSH R1 ON STACK
MOV R2, -(SP) ;;PUSH R2 ON STACK
MOV @RHWC, R0 ;;STARTING ADDRESS OF REG
MOV @WC, R1 ;;STARTING ADDRESS OF WERE SAVED
MOV @RHCC-RHWC+2/2, R2 ;NUMBER OF REG. INTO R2
10$: MOV @ (R0)+, (R1)+ ;SAVE HARDWARE REG.
DEC R2
BNE 10$
MOV (SP)+, R2 ;;POP STACK INTO R2
MOV (SP)+, R1 ;;POP STACK INTO R1
MOV (SP)+, R0 ;;POP STACK INTO R0
RTS PC
  
```

```

13109
13110
13111
13112
13113
13114
13115
13116
13117
13118
13119
13120
13121 041502 012037 004642
13122
13123
13124 041506 012737 010600 001124
13125
13126 041514 017737 142352 001126
13127 041522 023737 001124 001126
13128
13129
13130 041530 001401
13131 041532 104133
13132
13133
13134
13135
13136
13137 041534
13138
13139 041534 005077 142326
13140
13141
13142 041540 017777 143076 142312
13143 041546 005737 004644
13144 041552 001006
13145 041554 052777 000001 142276
13146
13147
13148 041562 104411
13149 041564 004060
13150 041566 000200
13151
13152
13153
13154
13155
13156
13157 041570 000200
13158
13159
13160
13161
13162
13163
13164

.SBTTL RS DATA TRANSFER SUBROUTINE
;*
;* THIS SUBROUTINE WRITES OR READS OR DOES A
;* WRITE CHECK ON CYLINDER 0 TRACK 0 SECTOR 0
;* IT ASSUMES THE RH REGISTERS ARE APPROPRIATELY
;* FILLED
;* WHEN IT RETURNS FROM THIS SUBROUTINE TO THE
;* MAIN PROGRAM IT MEANS THE TRANSFER IS COMPLETE
;*
;* THE CALL IS BY A JSR R0, @COMND COMAND
CMNDRS: MOV (R0)+, COMAND ;GET COMMAND
;CHECK THAT RHDS1 HAS DRY!DPR!MOL
MOV #DRY!DPR!MOL, $GDDAT ;GET GOOD = 10300
MOV @RHDS1, $BDDAT ;READ RHDS1 FOR COMPARISON
CMP $GDDAT, $BDDAT ;COMPARE EXPECTED
;DATA WITH DATA READ FROM
;RHDS1
;BRANCH IF GOOD
BEQ 65$
ERROR 133
;BEFORE ANY COMMAND IS GIVEN
;RHDS1 SHOULD HAVE DRY!DPR!MOL
;=10300
;BUT CONTAINED WHAT IS
;GIVEN IN BAD RHDS1
65$:
CLR @RHDA ;TRANSFER ON CYLINDER 0
;TRANSFER ON SECTOR 0
;TRACK 0
MOV @COMAND, @RHCS1 ;SET UP COMMAND
TST NOGO ;IS GO TO BE GIVEN
BNE 1$ ;BRANCH IF NO
BIS #GO, @RHCS1 ;SET GO
;WAIT FOR RDY BIT IN RHCS1 REGISTER TO SET
WAT ;TRAP TO WAIT.T SUBROUTINE
RHCS1 ;AND WAIT FOR RDY BIT IN
RDY ;RHCS1 REGISTER
;IF ERROR OCCURS HERE
;IT MEANS "RDY" DID NOT
;SET FOR THE FULL COUNT
;DOWN OF THE WAIT.T SUBROUTINE
;THIS TIME IS APPROXIMATELY
;LARGER THAN 500 MILLISECONDS
1$: RTS R0

.SBTTL RPO4 DATA TRANSFER SUBROUTINE
;*
;* THIS SUBROUTINE WRITE/READ/WRITE CHECK ON THE RPO4 CYLINDER 0.
;* TRACK 0, SECTOR 0.
;* IT ASSUMES THE RH REGISTERS ARE APPROPRIATELY FILLED.
;* WHEN IT RETURNS FROM THIS SUBROUTINE TO THE MAIN

```



```

13165          ;*      PROGRAM IT MEANS THE COMMAND IS COMPLETE
13166          ;*
13167          ;*      THE CALL IS BY A JSR RD, @COMND COMMAND.
13168
13169 041572 012037 004642      CMNDRP: MOV      (RD)+, COMAND      ;GET COMMAND
13170
13171 041576 013746 004172      MOV      PKACK, -(SP)      ;GET READY TO SET VV
13172 041602 052716 000001      BIS      #GO, (SP)      ;GET PACK ACKNOWLEDGE COMMAND
13173 041606 012677 142246      MOV      (SP)+, @RHCSI      ;SET VV
13174
13175          ;CHECK THAT RHDS1 HAS VV! DRY! DPR! MOL
13176 041612 012737 010700 001124      MOV      #VV! DRY! DPR! MOL, $GDDAT ;GET GOOD = 10700
13177
13178 041620 017737 142246 001126      MOV      @RHDS1, $BDDAT      ;READ RHDS1 FOR COMPARISON
13179 041626 023737 001124 001126      CMP      $GDDAT, $BDDAT      ;COMPARE EXPECTED
13180          ;DATA WITH DATA READ FROM
13181          ;RHDS1
13182 041634 001401              BEQ      65$                ;BRANCH IF GOOD
13183 041636 104132              ERROR   132
13184          ;BEFORE ANY COMMAND IS GIVEN
13185          ;RHDS1 SHOULD HAVE VV! DRY! DPR! MOL
13186          ;=10700
13187          ;BUT CONTAINED WHAT IS
13188          ;GIVEN IN BAD RHDS1
13189 041640                      65$:
13190
13191 041640 005077 142250      CLR      @RHCA              ;TRANSFER ON CYLINDER 0
13192 041644 005077 142216      CLR      @RHDS1            ;TRANSFER ON SECTOR 0
13193          ;TRACK 0
13194 041650 012777 014000 142234      MOV      #ECI! FMT22, @RHOF ;INHIBIT ECC
13195          ;FORMAT 16 BIT PER WORD
13196 041656 017777 142760 142174      MOV      @COMAND, @RHCSI    ;SET UP COMMAND
13197 041664 005737 004644      TST      NOGO              ;IS GO TO BE GIVEN
13198 041670 001006              BNE      1$                 ;BRANCH IF NO
13199 041672 052777 000001 142160      BIS      #GO, @RHCSI        ;SET GO
13200
13201          ;WAIT FOR RDY BIT IN RHCSI REGISTER TO SET
13202 041700 104411              WAT
13203 041702 004060              RHCSI
13204 041704 000200              RDY
13205          ;TRAP TO WAIT.T SUBROUTINE
13206          ;AND WAIT FOR RDY BIT IN
13207          ;RHCSI REGISTER
13208          ;IF ERROR OCCURS HERE
13209          ;IT MEANS "RDY" DID NOT
13210          ;SET FOR THE FULL COUNT
13211          ;DOWN OF THE WAIT.T SUBROUTINE
13212          ;THIS TIME IS APPROXIMATELY
13213          ;LARGER THAN 500 MILLISECONDS
13214
13215 041706 000200              1$:      RTS      RD
13216
13217          .SBTTL  TMO2 DATA TRANSFER SUBROUTINE
13218
13219          ;*      THIS SUBROUTINE WRITES/READS/WRITE CHECKS ON THE TMO2-TU16
13220          ;*      ON THE FIRST BLOCK FROM BEGINNING OF TAPE (BOT)
13221          ;*      IT ASSUMES THE RH REGISTERS ARE APPROPRIATELY FILLED.
13222          ;*      WHEN IT RETURNS FROM THIS SUBROUTINE TO THE
13223          ;*      MAIN PROGRAM IT MEANS THE COMMAND IS COMPLETE.
13224          ;*
  
```

```

13221 ;* THE CALL IS
13222 ;* JSR RO,COMND
13223
13224 041710 012037 004642 CMNDTM: MOV (RO)+,COMAND
13225 041714 013777 004660 142170 MOV SLAVE,ARHTC ;GET SLAVE NUMBER
13226 041722 012737 010600 001124 MOV #MOL!OPR!DRY,$GDDAT ;GET GOOD DATA
13227 041730 017737 142136 001126 MOV ARHDSI,$BDDAT ;GET RHDSI
13228 041736 042737 000062 001126 BIC #BOT!BIT05!BIT04,$BDDAT ;DISREGARD BOT
13229 041744 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED DATA WITH
13230 ;DATA READ FROM RHDSI
13231 041752 001401 BEQ 1$ ;BRANCH IF GOOD
13232 041754 104134 ERROR 134 ;BEFORE ANY COMMAND
13233 ;IS GIVEN DRIVE STATUS REGISTER
13234 ;DID NOT CONTAIN WHAT IS IN
13235 ;GOOD
13236 041756 032777 000002 142106 1$: BIT #BOT,ARHDSI ;IS IT ALREADY AT BOT
13237 041764 001035 BNE 2$ ;BRANCH IF YES
13238 ;IF NOT AT BOT GIVE A REWIND COMMAND
13239 041766 013777 004176 142064 MOV REWIND,ARHCSI ;LOAD RHCSI WITH REWIND COMMAND
13240 041774 052777 000001 142056 BIS #GO,ARHCSI
13241 ;WAIT FOR PIP BIT IN RHDSI REGISTER TO SET
13242 042002 104411 WAT ;TRAP TO WAIT.T SUBROUTINE
13243 042004 004072 RHDSI ;AND WAIT FOR PIP BIT IN
13244 042006 020000 PIP ;RHDSI REGISTER
13245 ;IF ERROR OCCURS HERE
13246 ;IT MEANS "PIP" DID NOT
13247 ;SET FOR THE FULL COUNT
13248 ;DOWN OF THE WAIT.T SUBROUTINE
13249 ;THIS TIME IS APPROXIMATELY
13250 ;LARGER THAN 500 MILLISECONDS
13251 ;WAIT FOR PIP BIT IN RHDSI REGISTER TO SET
13252 042010 012737 177777 004646 MOV #-1,WATO ;WAIT FOR PIP TO GO FROM 1 TO 0
13253 042016 104411 WAT ;TRAP TO WAIT.T SUBROUTINE
13254 042020 004072 RHDSI ;AND WAIT FOR PIP BIT IN
13255 042022 020000 PIP ;RHDSI REGISTER
13256 ;IF ERROR OCCURS HERE
13257 ;IT MEANS "PIP" DID NOT
13258 ;SET FOR THE FULL COUNT
13259 ;DOWN OF THE WAIT.T SUBROUTINE
13260 ;THIS TIME IS APPROXIMATELY
13261 ;LARGER THAN 500 MILLISECONDS
13262 042024 005037 004646 CLR WATO
13263 ;WAIT FOR BOT BIT IN RHDSI REGISTER TO SET
13264 042030 104411 WAT ;TRAP TO WAIT.T SUBROUTINE
13265 042032 004072 RHDSI ;AND WAIT FOR BOT BIT IN
13266 042034 000002 BOT ;RHDSI REGISTER
13267 ;IF ERROR OCCURS HERE
13268 ;IT MEANS "BOT" DID NOT
13269 ;SET FOR THE FULL COUNT
13270 ;DOWN OF THE WAIT.T SUBROUTINE
13271 ;THIS TIME IS APPROXIMATELY
13272 ;LARGER THAN 500 MILLISECONDS
13273
13274 ;CLEAR ATTENTIONS
13275 042036 013746 004142 MOV DCLEAR,-(SP)
13276 042042 052716 000001 BIS #GO,(SP)

```

```

13277 042046 012677 142006      MOV      (SP)+, @RHCS1
13278
13279      ;WAIT FOR DRY BIT IN RHDS1 REGISTER TO SET
13280 042052 104411      WAT      ;TRAP TO WAIT.T SUBROUTINE
13281 042054 004072      RHDS1    ;AND WAIT FOR DRY BIT IN
13282 042056 000200      DRY      ;RHDS1 REGISTER
13283      ;IF ERROR OCCURS HERE
13284      ;IT MEANS "DRY" DID NOT
13285      ;SET FOR THE FULL COUNT
13286      ;DOWN OF THE WAIT.T SUBROUTINE
13287      ;THIS TIME IS APPROXIMATELY
13288      ;LARGER THAN 500 MILLISECONDS
13289 042060      2$:      ;CHECK IF COMMAND IS REVERSE READ
13290      ;IF IT IS REVERSE READ, SPACE FORWARD ONE BLOCK
13291 042060 023777 004200 142554    CMP      REVRED, @COMAND ;IS IT REVERSE READ
13292 042066 001035      BNE      3$ ;BRANCH IF NOT
13293 042070 052777 001700 142014    BIS      #BPIB!NML, @RHTC ;800 BPI, NORMAL
13294 042076 042777 000010 142006    BIC      #EPAR, @RHTC ;ODD PARITY
13295
13296 042104 012777 177777 141754    MOV      #-1, @RHFC ;FRAME COUNT 1
13297      ;FOR SPACE FORWARD ONE BLOCK
13298 042112 013777 004202 141740    MOV      SPACFD, @RHCS1 ;SPACE FORWARD
13299 042120 052777 000001 141732    BIS      #GO, @RHCS1 ;GO
13300      ;WAIT FOR PIP BIT IN RHDS1 REGISTER TO SET
13301 042126 104411      WAT      ;TRAP TO WAIT.T SUBROUTINE
13302 042130 004072      RHDS1    ;AND WAIT FOR PIP BIT IN
13303 042132 020000      PIP      ;RHDS1 REGISTER
13304      ;IF ERROR OCCURS HERE
13305      ;IT MEANS "PIP" DID NOT
13306      ;SET FOR THE FULL COUNT
13307      ;DOWN OF THE WAIT.T SUBROUTINE
13308      ;THIS TIME IS APPROXIMATELY
13309      ;LARGER THAN 500 MILLISECONDS
13310      ;WAIT FOR PIP BIT IN RHDS1 REGISTER TO SET
13311 042134 012737 177777 004646    MOV      #-1, WATO ;WAIT FOR PIP TO GO FROM 1 TO 0
13312 042142 104411      WAT      ;TRAP TO WAIT.T SUBROUTINE
13313 042144 004072      RHDS1    ;AND WAIT FOR PIP BIT IN
13314 042146 020000      PIP      ;RHDS1 REGISTER
13315      ;IF ERROR OCCURS HERE
13316      ;IT MEANS "PIP" DID NOT
13317      ;SET FOR THE FULL COUNT
13318      ;DOWN OF THE WAIT.T SUBROUTINE
13319      ;THIS TIME IS APPROXIMATELY
13320      ;LARGER THAN 500 MILLISECONDS
13321 042150 005037 004646      CLR      WATO
13322      ;WAIT FOR DRY BIT IN RHDS1 REGISTER TO SET
13323 042154 104411      WAT      ;TRAP TO WAIT.T SUBROUTINE
13324 042156 004072      RHDS1    ;AND WAIT FOR DRY BIT IN
13325 042160 000200      DRY      ;RHDS1 REGISTER
13326      ;IF ERROR OCCURS HERE
13327      ;IT MEANS "DRY" DID NOT
13328      ;SET FOR THE FULL COUNT
13329      ;DOWN OF THE WAIT.T SUBROUTINE
13330      ;THIS TIME IS APPROXIMATELY
13331      ;LARGER THAN 500 MILLISECONDS
13332 042162      3$:      ;NOW GIVE COMMAND

```

13333	042162	052777	001700	141722	BIS	#BPIB!NML,@RHTC	;800 BPI, NORMAL
13334	042170	042777	000010	141714	BIC	#EPAR,@RHTC	;ODD PARITY
13335	042176	017746	141660		MOV	@RHWC,-(SP)	;GET WORD COUNT
13336	042202	005416			NEG	(SP)	;GET POSITIVE NUMBER
13337	042204	061616			ADD	(SP),(SP)	;DOUBLE WORD COUNT
13338	042206	005416			NEG	(SP)	;GET NEGATIVE NUMBER FOR
13339							;FRAME COUNT REGISTER
13340	042210	012677	141652		MOV	(SP)+,@RHFC	;FILL FRAME COUNT
13341	042214	017777	142422	141636	MOV	@COMAND,@RHCS1	;GET COMMAND
13342							
13343	042222	005737	004644		TST	NOGO	;IS GO TO BE GIVEN
13344	042226	001006			BNE	4\$	;BRANCH IF NO
13345	042230	052777	000001	141622	BIS	#GO,@RHCS1	;GO
13346							
13347							;WAIT FOR DRY BIT IN RHDS1 REGISTER TO SET
13348	042236	104411			WAT		;TRAP TO WAIT.T SUBROUTINE
13349	042240	004072			RHDS1		;AND WAIT FOR DRY BIT IN
13350	042242	000200			DRY		;RHDS1 REGISTER
13351							;IF ERROR OCCURS HERE
13352							;IT MEANS "DRY" DID NOT
13353							;SET FOR THE FULL COUNT
13354							;DOWN OF THE WAIT.T SUBROUTINE
13355							;THIS TIME IS APPROXIMATELY
13356							;LARGER THAN 500 MILLISECONDS
13357	042244				4\$:		
13358							;WAIT FOR RDY BIT IN RHCS1 REGISTER TO SET
13359	042244	104411			WAT		;TRAP TO WAIT.T SUBROUTINE
13360	042246	004060			RHCS1		;AND WAIT FOR RDY BIT IN
13361	042250	000200			RDY		;RHCS1 REGISTER
13362							;IF ERROR OCCURS HERE
13363							;IT MEANS "RDY" DID NOT
13364							;SET FOR THE FULL COUNT
13365							;DOWN OF THE WAIT.T SUBROUTINE
13366							;THIS TIME IS APPROXIMATELY
13367							;LARGER THAN 500 MILLISECONDS
13368	042252	000200			RTS	RO	

```

13369
13370
13371
13372
13373 042254
13374 042254 104401 042262
13375 042260 000424
13376
13377 042332
13378 042332 013746 004060
13379 042336 104402
13380 042340 104401 042346
13381 042344 000425
13382
13383 042420
13384 042420 104410
13385 042422 012700 004102
13386 042426 012701 000024
13387 042432 042710 177700
13388 042436 051620
13389 042440 005301
13390 042442 001373
13391 042444 104401 042452
13392 042450 000417
13393
13394 042510
13395 042510 013746 002716
13396 042514 104402
13397 042516 104401 042524
13398 042522 000437
13399
13400 042622
13401 042622 104410
13402 042624 012637 002716
13403 042630 104401 042636
13404 042634 000421
13405
13406 042700
13407 042700 104401 042706
13408 042704 000414
13409
13410 042736
13411 042736 013746 004060
13412 042742 104402
13413 042744 104401 042752
13414 042750 000415
13415
13416 043004
13417 043004 013746 002716
13418 043010 104402
13419 043012 104401 043020
13420 043016 000416
13421
13422 043054
13423 043054 000000
13424

; * THIS ROUTINE WILL ALLOW THE CHANGE OF THE BASE
; * ADDRESS FROM 176700 TO ANY TYPED VALUE

BASECH:
TYPE ,65$ ;:TYPE ASCIZ STRING
BR 64$ ;:GET OVER THE ASCIZ
;:65$: .ASCIZ <15><12>/PRESENT BASE ADDRESS OF REGISTERS IS /
64$:
MOV @#RHCS1,-(SP) ;GET READY TO TYPE OLD BASE
TYPOC
TYPE ,67$ ;:TYPE ASCIZ STRING
BR 66$ ;:GET OVER THE ASCIZ
;:67$: .ASCIZ <15><12>/TYPE NEW BASE ADDRESS FOLLOWED BY 'CR' /
66$:
RDOCT
MOV #RHDB,R0 ;GET STARTING ADDRESS OF REGISTERS
MOV #20,R1 ;NUMBER OF REGISTERS
;:1$: BIC #1C7,(R0) ;CLEAR OLD BASE
BIS (SP),(R0)+ ;SET NEW BASE
DEC R1 ;COUNT
BNE 1$ ;BRANCH IF 20 NOT DONE
TYPE ,69$ ;:TYPE ASCIZ STRING
BR 68$ ;:GET OVER THE ASCIZ
;:69$: .ASCIZ <15><12>/PRESENT VECTOR ADDRESS IS /
68$:
MOV @#RPVEC,-(SP) ;GET READY TO TYPE OLD VECTOR ADDRESS
TYPOC
TYPE ,71$ ;:TYPE ASCIZ STRING
BR 70$ ;:GET OVER THE ASCIZ
;:71$: .ASCIZ <15><12>/TYPE NEW VECTOR ADDRESS OR RETYPE OLD ONE FOLLOWED BY "CR" /
70$:
RDOCT
MOV (SP)+,@#RPVEC ;SETUP VECTOR ADDRESS
TYPE ,73$ ;:TYPE ASCIZ STRING
BR 72$ ;:GET OVER THE ASCIZ
;:73$: .ASCIZ <15><12>/RESTART PROGRAM FROM 200 OR 210 /
72$:
TYPE ,75$ ;:TYPE ASCIZ STRING
BR 74$ ;:GET OVER THE ASCIZ
;:75$: .ASCIZ <15><12>/NEW BASE WILL REMAIN /
74$:
MOV @#RHCS1,-(SP)
TYPOC
TYPE ,77$ ;:TYPE ASCIZ STRING
BR 76$ ;:GET OVER THE ASCIZ
;:77$: .ASCIZ <15><12>/NEW VECTOR WILL REMAIN /
76$:
MOV @#RPVEC,-(SP)
TYPOC
TYPE ,79$ ;:TYPE ASCIZ STRING
BR 78$ ;:GET OVER THE ASCIZ
;:79$: .ASCIZ <15><12>/UNTIL PROGRAM IS RELOADED /
78$:
HALT

```

G04

```

13425 043056          RPVECT:
13426 043056 104401 043064      TYPE      ,65$          ;;TYPE ASCIZ STRING
13427 043062 000411          BR      ,64$          ;;GET OVER THE ASCIZ
13428          ;;65$: .ASCIZ /TRAPED FROM PC = /
13429 043106          64$:
13430 043106 104402          TYPOC          ;TYPE FROM PC
13431 043110 012777 043056 137600  MOV      #RPVECT,RPVEC      ;RESTORE TRAP RP04 VECTOR
13432 043116 000000          HALT          ;CHANGE TO CONTINUE
13433
13434
13435
13436

```

;\*\*\*\*\*

13437  
13438  
13439  
13440  
13441  
13442  
13443  
13444  
13445  
13446  
13447  
13448  
13449  
13450  
13451  
13452  
13453  
13454  
13455  
13456  
13457  
13458  
13459  
13460  
13461  
13462  
13463  
13464  
13465  
13466  
13467  
13468  
13469  
13470  
13471  
13472  
13473  
13474  
13475  
13476  
13477  
13478  
13479  
13480  
13481  
13482  
13483  
13484  
13485  
13486  
13487  
13488  
13489  
13490  
13491  
13492

.SBTTL SCOPE HANDLER ROUTINE

;;\*\*\*\*\*  
; \*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT  
; \*AND LOAD THE TEST NUMBER(\$TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)  
; \*AND LOAD THE ERROR FLAG (\$ERFLG) INTO DISPLAY<15:08>  
; \*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:  
; \*SW14=1 LOOP ON TEST  
; \*SW11=1 INHIBIT ITERATIONS  
; \*SW09=1 LOOP ON ERROR  
; \*SW08=1 LOOP ON TEST IN SWR<7:0>  
; \*CALL  
; \* SCOPE ;:SCOPE=IOT

\$SCOPE:  
CLR WATO ;WAIT FOR BIT TO SET  
;USED IN "WAT" WAIT ROUTINE  
CLR NOGO ;DO GIVE GO IN COMMAND ROUTINE  
1\$: BIT #BIT14, \$SWR ;LOOP ON PRESENT TEST?  
BNE \$OVER ;YES IF SW14=1  
;\*\*\*\*\*START OF CODE FOR THE XOR TESTER\*\*\*\*\*  
\$XTSTR: BR 6\$ ;IF RUNNING ON THE "XOR" TESTER CHANGE  
;THIS INSTRUCTION TO A "NOP" (NOP=240)  
MOV @#ERRVEC, -(SP) ;SAVE THE CONTENTS OF THE ERROR VECTOR  
MOV #5, @#ERRVEC ;SET FOR TIMEOUT  
TST @#177060 ;TIME OUT ON XOR?  
MOV (SP)+, @#ERRVEC ;RESTORE THE ERROR VECTOR  
BR \$SVLAD ;GO TO THE NEXT TEST  
5\$: CMP (SP)+, (SP)+ ;CLEAR THE STACK AFTER A TIME OUT  
MOV (SP)+, @#ERRVEC ;RESTORE THE ERROR VECTOR  
BR 7\$ ;LOOP ON THE PRESENT TEST  
6\$; ;\*\*\*\*\*END OF CODE FOR THE XOR TESTER\*\*\*\*\*  
BIT #BIT08, \$SWR ;LOOP ON SPEC. TEST?  
BEQ 2\$ ;BR IF NO  
CMPB \$SWR, \$TSTNM ;ON THE RIGHT TEST? SWR<7:0>  
BEQ \$OVER ;BR IF YES  
2\$: TSTB \$ERFLG ;HAS AN ERROR OCCURRED?  
BEQ 3\$ ;BR IF NO  
CMPB \$ERMAX, \$ERFLG ;MAX. ERRORS FOR THIS TEST OCCURRED?  
BHI 3\$ ;BR IF NO  
BIT #BIT09, \$SWR ;LOOP ON ERROR?  
BEQ 4\$ ;BR IF NO  
7\$: MOV \$LPERR, \$LPADR ;SET LOOP ADDRESS TO LAST SCOPE  
BR \$OVER  
4\$: CLRB \$ERFLG ;ZERO THE ERROR FLAG  
CLR \$TIMES ;CLEAR THE NUMBER OF ITERATIONS TO MAKE  
BR 1\$ ;ESCAPE TO THE NEXT TEST  
3\$: BIT #BIT11, \$SWR ;INHIBIT ITERATIONS?  
BNE 1\$ ;BR IF YES  
TST \$PASS ;IF FIRST PASS OF PROGRAM  
BEQ 1\$ ;INHIBIT ITERATIONS  
INC \$ICNT ;INCREMENT ITERATION COUNT  
CMP \$TIMES, \$ICNT ;CHECK THE NUMBER OF ITERATIONS MADE  
BGE \$OVER ;BR IF MORE ITERATION REQUIRED  
1\$: MOV #1, \$ICNT ;REINITIALIZE THE ITERATION COUNTER

13493	043326	013737	043376	001212		MOV	\$MXCNT,\$TIMES	;; SET NUMBER OF ITERATIONS TO C0
13494	043334	105237	001102		\$SVLAD:	INCB	\$STNM	;; COUNT TEST NUMBERS
13495	043340	011637	001106			MOV	(SP), \$LPADR	;; SAVE SCOPE LOOP ADDRESS
13496	043344	011637	001110			MOV	(SP), \$LPERR	;; SAVE ERROR LOOP ADDRESS
13497	043350	005037	001214			CLR	\$ESCAPE	;; CLEAR THE ESCAPE FROM ERROR ADDRESS
13498	043354	112737	000001	001115		MOVB	#1, \$ERMAX	;; ONLY ALLCW ONE(1) ERROR ON NEXT TEST
13499	043362	013777	001102	135552	\$OVER:	MOV	\$STNM, @DISPLAY	;; DISPLAY TEST NUMBER
13500	043370	013716	001106			MOV	\$LPADR, (SP)	;; FUDGE RETURN ADDRESS
13501	043374	000002				RTI		;; FIXES PS
13502	043376	000004			\$MXCNT:	4		;; MAX. NUMBER OF ITERATIONS



```

13503 .SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
13504
13505 ;*****
13506 ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
13507 ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
13508 ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
13509 ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
13510 ;*REPLACED WITH SPACES.
13511 ;*CALL:
13512 ;*      MOV      NUM,-(SP)      ;;PUT THE BINARY NUMBER ON THE STACK
13513 ;*      TYPDS                    ;;GO TO THE ROUTINE
13514
13515 $TYPDS:
13516 043400 010046      MOV      R0,-(SP)      ;;PUSH R0 ON STACK
13517 043402 010146      MOV      R1,-(SP)      ;;PUSH R1 ON STACK
13518 043404 010246      MOV      R2,-(SP)      ;;PUSH R2 ON STACK
13519 043406 010346      MOV      R3,-(SP)      ;;PUSH R3 ON STACK
13520 043410 010546      MOV      R5,-(SP)      ;;PUSH R5 ON STACK
13521 043412 012746 020200      MOV      #20200,-(SP)  ;;SET BLANK SWITCH AND SIGN
13522 043416 016605 000020      MOV      20(SP),R5    ;;GET THE INPUT NUMBER
13523 043422 100004      BPL      1$           ;;BR IF INPUT IS POS.
13524 043424 005405      NEG      R5           ;;MAKE THE BINARY NUMBER POS.
13525 043426 112766 000055 000001      MOVVB   #'-,1(SP)    ;;MAKE THE ASCII NUMBER NEG.
13526 043434 005000      1$:      CLR      R0           ;;ZERO THE CONSTANTS INDEX
13527 043436 012703 043614      MOV      #SDBLK,R3   ;;SETUP THE OUTPUT POINTER
13528 043442 112723 000040      MOVVB   #' ,(R3)+    ;;SET THE FIRST CHARACTER TO A BLANK
13529 043446 005002      2$:      CLR      R2           ;;CLEAR THE BCD NUMBER
13530 043450 016001 043604      MOV      $DTBL(R0),R1 ;;GET THE CONSTANT
13531 043454 160105      3$:      SUB      R1,R5       ;;FORM THIS BCD DIGIT
13532 043456 002402      BLT     4$           ;;BR IF DONE
13533 043460 005202      INC     R2           ;;INCREASE THE BCD DIGIT BY 1
13534 043462 000774      BR     3$
13535 043464 060105      4$:      ADD     R1,R5       ;;ADD BACK THE CONSTANT
13536 043466 005702      TST     R2           ;;CHECK IF BCD DIGIT=0
13537 043470 001002      BNE     5$           ;;FALL THROUGH IF 0
13538 043472 105716      TSTB   (SP)         ;;STILL DOING LEADING 0'S?
13539 043474 100407      BMI     7$           ;;BR IF YES
13540 043476 106316      5$:      ASLB   (SP)         ;;MSD?
13541 043500 103003      BCC     6$           ;;BR IF NO
13542 043502 116663 000001 177777      MOVVB   1(SP),-1(R3)  ;;YES--SET THE SIGN
13543 043510 052702 000060      6$:      BIS     #'0,R2       ;;MAKE THE BCD DIGIT ASCII
13544 043514 052702 000040      7$:      BIS     #' ,R2       ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
13545 043520 110223      MOVVB   R2,(R3)+    ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
13546 043522 005720      TST     (R0)+       ;;JUST INCREMENTING
13547 043524 020027 000010      CMP     R0,#10      ;;CHECK THE TABLE INDEX
13548 043530 002746      BLT     2$           ;;GO DO THE NEXT DIGIT
13549 043532 003002      BGT     8$           ;;GO TO EXIT
13550 043534 010502      MOV     R5,R2       ;;GET THE LSD
13551 043536 000764      BR     6$           ;;GO CHANGE TO ASCII
13552 043540 105726      8$:      TSTB   (SP)+       ;;WAS THE LSD THE FIRST NON-ZERO?
13553 043542 100003      BPL     9$           ;;BR IF NO
13554 043544 116663 177777 177776      MOVVB   -1(SP),-2(R3) ;;YES--SET THE SIGN FOR TYPING
13555 043552 105013      9$:      CLRB   (R3)        ;;SET THE TERMINATOR
13556 043554 012605      MOV     (SP)+,R5    ;;POP STACK INTO R5
13557 043556 012603      MOV     (SP)+,R3    ;;POP STACK INTO R3
13558 043560 012602      MOV     (SP)+,R2    ;;POP STACK INTO R2

```

```

13559 043562 012601      MOV      (SP)+,R1      ;; POP STACK INTO R1
13560 043564 012600      MOV      (SP)+,RO      ;; POP STACK INTO RO
13561 043566 104401 043614  TYPE      $DBLK      ;; NOW TYPE THE NUMBER
13562 043572 016666 000002 000004  MOV      2(SP),4(SP)  ;; ADJUST THE STACK
13563 043600 012616      MOV      (SP)+,(SP)
13564 043602 000002      RTI                      ;; RETURN TO USER
13565 043604 023420      $DTBL: 10000.
13566 043606 001750      1000.
13567 043610 000144      100.
13568 043612 000012      10.
13569 043614 000004      $DBLK: .BLKW 4
13570      .SBTTL TYPE ROUTINE
13571
13572      ;;*****
13573      ;;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
13574      ;;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
13575      ;;*NOTE1: $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
13576      ;;*NOTE2: $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
13577      ;;*NOTE3: $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
13578      ;;*
13579      ;;*CALL:
13580      ;;*1) USING A TRAP INSTRUCTION
13581      ;;* TYPE ,MESADR      ;; MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
13582      ;;*OR
13583      ;;* TYPE
13584      ;;* MESADR
13585      ;;*
13586
13587 043624 105737 001157  $TYPE: TSTB $TFPLG      ;; IS THERE A TERMINAL?
13588 043630 100002      BPL 1$      ;; BR IF YES
13589 043632 000000      HALT      ;; HALT HERE IF NO TERMINAL
13590 043634 000407      BR 3$      ;; LEAVE
13591 043636 010046 1$: MOV RO,-(SP)      ;; SAVE RO
13592 043640 017600 000002  MOV 2(SP),RO      ;; GET ADDRESS OF ASCIZ STRING
13593 043644 112046 2$: MOVB (RO)+,-(SP)  ;; PUSH CHARACTER TO BE TYPED ONTO STACK
13594 043646 001005      BNE 4$      ;; BR IF IT ISN'T THE TERMINATOR
13595 043650 005726      TST (SP)+      ;; IF TERMINATOR POP IT OFF THE STACK
13596 043652 012600 6$: MOV (SP)+,RO      ;; RESTORE RO
13597 043654 062716 000002 3$: ADD #2,(SP)      ;; ADJUST RETURN PC
13598 043660 000002      RTI                      ;; RETURN
13599 043662 122716 000011 4$: CMPB #HT,(SP)      ;; BRANCH IF <HT>
13600 043666 001430      BEQ 8$
13601 043670 122716 000200      CMPB #CRLF,(SP)      ;; BRANCH IF NOT <CRLF>
13602 043674 001006      BNE 5$
13603 043676 005726      TST (SP)+      ;; POP <CR><LF> EQUIV
13604 043700 104401      TYPE      ;; TYPE A CR AND LF
13605 043702 001223      $CRLF
13606 043704 105037 044040      CLRB $CHARCNT      ;; CLEAR CHARACTER COUNT
13607 043710 000755      BR 2$      ;; GET NEXT CHARACTER
13608 043712 004737 043774 5$: JSR PC,$TYPEC      ;; GO TYPE THIS CHARACTER
13609 043716 123726 001156 6$: CMPB $FILLC,(SP)+  ;; IS IT TIME FOR FILLER CHARS.?
13610 043722 001350      BNE 2$      ;; IF NO GO GET NEXT CHAR.
13611 043724 013746 001154      MOV $NULL,-(SP)      ;; GET # OF FILLER CHARS. NEEDED
13612      AND THE NULL CHAR.
13613 043730 105366 000001 7$: DECB 1(SP)      ;; DOES A NULL NEED TO BE TYPED?
13614 043734 002770      BLT 6$      ;; BR IF NO--GO POP THE NULL OFF OF STACK

```

```

13615 043736 004737 043774      JSR    PC,$TYPEC      ;;GO TYPE A NULL
13616 043742 105337 044040      DECB  $CHARCNT       ;;DO NOT COUNT AS A COUNT
13617 043746 000770              BR     7$            ;;LOOP
13618
13619                               ;HORIZONTAL TAB PROCESSOR
13620
13621 043750 112716 000040      8$:   MOVB  #' (SP)      ;;REPLACE TAB WITH SPACE
13622 043754 004737 043774      9$:   JSR    PC,$TYPEC      ;;TYPE A SPACE
13623 043760 132737 000007 044040  BITB  #7,$CHARCNT     ;;BRANCH IF NOT AT
13624 043766 001372              BNE   9$            ;;TAB STOP
13625 043770 005726              TST   (SP)+         ;;POP SPACE OFF STACK
13626 043772 000724              BR     2$            ;;GET NEXT CHARACTER
13627 043774 105777 135150  $TYPEC: TSTB  @2$TPS     ;;WAIT UNTIL PRINTER IS READY
13628 044000 100375              BPL  $TYPEC
13629 044002 116677 000002 135142  MOVB  2(SP),@2$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
13630 044010 122766 000015 000002  CMPB  #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
13631 044016 001003              BNE  1$            ;;BRANCH IF NO
13632 044020 105037 044040      CLRB  $CHARCNT       ;;YES--CLEAR CHARACTER COUNT
13633 044024 000406              BR     $TYPEX       ;;EXIT
13634 044026 122766 000012 000002  1$:   CMPB  #LF,2(SP)     ;;IS CHARACTER A LINE FEED?
13635 044034 001402              BEQ  $TYPEX       ;;BRANCH IF YES
13636 044036 105227              INCB  (PC)+         ;;COUNT THE CHARACTER
13637 044040 000000  $CHARCNT: .WORD 0      ;;CHARACTER COUNT STORAGE
13638 044042 000207  $TYPEX: RTS    PC
13639
13640                               .SBTTL TTY INPUT ROUTINE
13641
13642                               ;*****
13643                               .ENABL  LSB
13644 044044 000000  $TKCNT: .WORD 0      ;;NUMBER OF ITEMS IN QUEUE
13645 044046 000000  $TKQIN: .WORD 0     ;;INPUT POINTER
13646 044050 000000  $TKQOUT: .WORD 0    ;;OUTPUT POINTER
13647 044052 000011  $TKQSRT: .BLKB 9.  ;;TTY KEYBOARD QUEUE
13648 044063  $TKQEND=.
13649 044064  .EVEN
13650
13651                               ;*TK INITIALIZE ROUTINE
13652                               ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
13653                               ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
13654
13655                               ;*CALL:
13656                               ;*      JSR    PC,$TKINT
13657                               ;*      RETURN
13658
13659 044064 005037 044044  $TKINT: CLR  $TKCNT      ;;CLEAR COUNT OF ITEMS IN QUEUE
13660 044070 012737 044052 044046  MOV   #2$TKQSRT,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
13661 044076 013737 044046 044050  MOV   $TKQIN,$TKQOUT  ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
13662 044104 012737 044134 000060  MOV   #2$TKSRV,@2$TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
13663 044112 012737 000200 000062  MOV   #200,@2$TKVEC+2 ;;"BR" LEVEL 4
13664 044120 005777 135022  TST   @2$TKB         ;;CLEAR DJNE FLAG
13665 044124 012777 000100 135012  MOV   #100,@2$TKS    ;;ENABLE TTY KEYBOARD INTERRUPT
13666 044132 000207              RTS    PC           ;;RETURN TO CALLER
13667
13668                               ;*TK SERVICE ROUTINE
13669                               ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
13670                               ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING

```

```

13671 ;*IT IN THE QUEUE.
13672 ;*IF THE CHARACTER IS A "CONTROL-C" (↑C) $TKINT IS CALLED AND
13673 ;*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (OPERSEL)
13674
13675 044134 117746 135006 $TKSRV: MOVB @STKB,-(SP) ;; PICKUP THE CHARACTER
13676 044140 042716 177600 BIC #↑C177,(SP) ;; STRIP THE JUNK
13677 044144 021627 000003 CMP (SP),#3 ;; IS IT A CONTROL C?
13678 044150 001007 BNE 1$ ;; BRANCH IF NO
13679 044152 104401 044543 TYPE ,SCNTLC ;; TYPE A CONTROL-C (↑C)
13680 044156 004737 044064 JSR PC,$TKINT ;; INIT THE KEYBOARD
13681 044162 005726 TST (SP)+ ;; CLEAN UP STACK
13682 044164 000137 040654 JMP OPERSEL ;; CONTROL C RESTART
13683
13684 044170 1$:
13685 044170 022737 000011 044044 CMP #9,,$TKCNT ;; IS THE QUEUE FULL?
13686 044176 001004 BNE 3$ ;; BRANCH IF NO
13687 044200 104401 001216 TYPE ,SBELL ;; RING THE TTY BELL
13688 044204 005726 TST (SP)+ ;; CLEAN CHARACTER OFF OF STACK
13689 044206 000451 BR 5$ ;; EXIT
13690 044210 021627 000023 3$: CMP (SP),#23 ;; IS IT A CONTROL-S?
13691 044214 001021 BNE 32$ ;; BRANCH IF NO
13692 044216 005077 134722 CLR @STKS ;; DISABLE TTY KEYBOARD INTERRUPTS
13693 044222 005726 TST (SP)+ ;; CLEAN CHAR OFF STACK
13694 044224 105777 134714 31$: TSTB @STKS ;; WAIT FOR A CHAR
13695 044230 100375 BPL 31$ ;; LOOP UNTIL ITS THERE
13696 044232 117746 134710 MOVB @STKB,-(SP) ;; GET THE CHARACTER
13697 044236 042716 177600 BIC #↑C177,(SP) ;; MAKE IT 7-BIT ASCII
13698 044242 022627 000021 CMP (SP)+,#21 ;; IS IT A CONTROL-Q?
13699 044246 001366 BNE 31$ ;; BRANCH IF NO
13700 044250 012777 000100 134666 MOV #100,@STKS ;; REENABLE TTY KEYBOARD INTERRUPTS
13701 044256 000002 RTI ;; RETURN
13702 044260 005237 044044 32$: INC $TKCNT ;; COUNT THIS CHARACTER
13703 044264 021627 000140 CMP (SP),#140 ;; IS IT UPPER CASE?
13704 044270 002405 BLT 4$ ;; BRANCH IF YES
13705 044272 021627 000175 CMP (SP),#175 ;; IS IT A SPECIAL CHAR?
13706 044276 003002 BGT 4$ ;; BRANCH IF YES
13707 044300 042716 000040 BIC #40,(SP) ;; MAKE IT UPPER CASE
13708 044304 112677 177536 4$: MOVB (SP)+,@$TKQIN ;; AND PUT IT IN QUEUE
13709 044310 005237 044046 INC $TKQIN ;; UPDATE THE POINTER
13710 044314 023727 044046 044063 CMP $TKQIN,$$TKQEND ;; GO OFF THE END?
13711 044322 001003 BNE 5$ ;; BRANCH IF NO
13712 044324 012737 044052 044046 MOV $$TKQSRST,$TKQIN ;; RESET THE POINTER
13713 044332 000002 5$: RTI ;; RETURN
13714
13715 .DSABL LSB
13716
13717
13718 ;*****
13719 ;THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
13720 ;*CALL:
13721 ;* ROCHR ;; GET A CHARACTER FROM THE QUEUE
13722 ;* RETURN HERE ;; CHARACTER IS ON THE STACK
13723 ;* ;; WITH PARITY BIT STRIPPED OFF
13724 ;*
13725 ;
13726 044334 011646 $ROCHR: MOV (SP),-(SP) ;; PUSH DOWN THE PC AND

```



```

13783
13784
13785
13786
13787
13788
13789
13790
13791
13792
13793
13794
13795
13796
13797
13798 044604 011646
13799 044606 016666 000004 000002
13800 044614 010046
13801 044616 010146
13802 044620 010246
13803 044622 104407
13804 044624 012600
13805 044626 010037 044732
13806 044632 005001
13807 044634 005002
13808 044636 112046
13809 044640 001420
13810 044642 122716 000060
13811 044646 003026
13812 044650 122716 000067
13813 044654 002423
13814 044656 006301
13815 044660 006102
13816 044662 006301
13817 044664 006102
13818 044666 006301
13819 044670 006102
13820 044672 042716 177770
13821 044676 062601
13822 044700 000756
13823 044702 005726
13824 044704 010166 000012
13825 044710 010237 044742
13826 044714 012602
13827 044716 012601
13828 044720 012600
13829 044722 000002
13830 044724 005726
13831 044726 105010
13832 044730 104401
13833 044732 000000
13834 044734 104401 001222
13835 044740 000730
13836 044742 000000

```

.SBTTL READ AN OCTAL NUMBER FROM THE TTY

```

*****
;THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
;CHANGE IT TO BINARY.
;THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
;OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED
;FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
;THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
;CALL:
;*      RDOCT                ;;READ AN OCTAL NUMBER
;*      RETURN HERE         ;;LOW ORDER BITS ARE ON TOP OF THE STACK
;*                          ;;HIGH ORDER BITS ARE IN $HIOCT

```

```

$RDOCT: MOV      (SP), -(SP)      ;; PROVIDE SPACE FOR THE
        MOV      4(SP), 2(SP)    ;; INPUT NUMBER
        MOV      RO, -(SP)       ;; PUSH RO ON STACK
        MOV      R1, -(SP)       ;; PUSH R1 ON STACK
        MOV      R2, -(SP)       ;; PUSH R2 ON STACK
1$:     RDLIN                    ;; READ AN ASCII LINE
        MOV      (SP)+, RO        ;; GET ADDRESS OF 1ST CHARACTER
        MOV      RO, 5$          ;; AND SAVE IT
        CLR      R1              ;; CLEAR DATA WORD
        CLR      R2
2$:     MOVB      (RO)+, -(SP)    ;; PICKUP THIS CHARACTER
        BEQ      3$              ;; IF ZERO GET OUT
        CMPB     #'0, (SP)       ;; MAKE SURE THIS CHARACTER
        BGT      4$              ;; IS AN OCTAL DIGIT
        CMPB     #'7, (SP)
        BLT      4$
        ASL      R1              ;; *2
        ROL      R2
        ASL      R1              ;; *4
        ROL      R2
        ASL      R1              ;; *8
        ROL      R2
        BIC      #'C7, (SP)     ;; STRIP THE ASCII JUNK
        ADD      (SP)+, R1       ;; ADD IN THIS DIGIT
        BR       2$             ;; LOOP
3$:     TST      (SP)+          ;; CLEAN TERMINATOR FROM STACK
        MOV      R1, 12(SP)      ;; SAVE THE RESULT
        MOV      R2, $HIOCT
        MOV      (SP)+, R2
        MOV      (SP)+, R1
        MOV      (SP)+, RO
        RTI
4$:     TST      (SP)+          ;; CLEAN PARTIAL FROM STACK
        CLPB     (RO)           ;; SET A TERMINATOR
        TYPE     TYPE           ;; TYPE UP THRU THE BAD CHAR.
5$:     .WORD    0
        TYPE     $QUES          ;; "?" "CR" & "LF"
        BR      1$             ;; TRY AGAIN
$HIOCT: .WORD    0             ;; HIGH ORDER BITS GO HERE

```

```

13837
13838
13839
13840
13841
13842
13843
13844
13845
13846
13847
13848
13849
13850
13851 044744
13852 044744 105237 001103
13853 044750 001775
13854 044752 013777 001102 134162
13855 044760 032777 002000 134152
13856 044766 001402
13857 044770 104401 001216
13858 044774 005237 001112
13859 045000 011637 001116
13860 045004 162737 000002 001116
13861 045012 117737 134100 001114
13862 045020 032777 020000 134112
13863 045026 001004
13864 045030 004737 045102
13865 045034 104401 001223
13866 045040
13867 045040 005777 134074
13868 045044 100001
13869 045046 000000
13870 045050 032777 001000 134062
13871 045056 001402
13872 045060 013716 001110
13873 045064 035737 001214
13874 045070 001402
13875 045072 013716 001214
13876 045076
13877 045076 000002
13878
13879
13880
13881
13882
13883
13884
13885
13886
13887
13888
13889
13890 045100 000000
13891 045102 013746 001140
13892 045106 042716 177277

```

```

.SBTTL ERROR HANDLER ROUTINE
;*****
;THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
;SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
;AND GO TO $ERRTYP ON ERROR
;THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
;*SW15=1 HALT ON ERROR
;*SW13=1 INHIBIT ERROR TYPEOUTS
;*SW10=1 BELL ON ERROR
;*SW09=1 LOOP ON ERROR
;*CALL
;* ERROR N ;;ERROR=EMT AND N=ERROR ITEM NUMBER

$ERROR:
7$: INCB $ERFLG ;; SET THE ERROR FLAG
BEQ 7$ ;; DON'T LET THE FLAG GO TO ZERO
MOV $STNM, $DISPLAY ;; DISPLAY TEST NUMBER AND ERROR FLAG
BIT #BIT10, $SWR ;; BELL ON ERROR?
BEQ 1$ ;; NO - SKIP
TYPE $BELL ;; RING BELL
1$: INC $ERTTL ;; COUNT THE NUMBER OF ERRORS
MOV (SP), $ERRPC ;; GET ADDRESS OF ERROR INSTRUCTION
SUB #2, $ERRPC
MOVB $ERRPC, $ITEMB ;; STRIP AND SAVE THE ERROR ITEM CODE
BIT #BIT13, $SWR ;; SKIP TYPEOUT IF SET
BNE 20$ ;; SKIP TYPEOUTS
JSR PC, $ERRTYP ;; GO TO USER ERROR ROUTINE
TYPE , $CRLF

20$:
2$: TST $SWR ;; HALT ON ERROR
BPL 3$ ;; SKIP IF CONTINUE
HALT ;; HALT ON ERROR!
3$: BIT #BIT09, $SWR ;; LOOP ON ERROR SWITCH SET?
BEQ 4$ ;; BR IF NO
MOV $LPERR, (SP) ;; FUDGE RETURN FOR LOOPING
4$: TST $ESCAPE ;; CHECK FOR AN ESCAPE ADDRESS
BEQ 5$ ;; BR IF NONE
MOV $ESCAPE, (SP) ;; FUDGE RETURN ADDRESS FOR ESCAPE

5$: RTI ;; RETURN
;*****
.SBTTL ERROR MESSAGE TYPEOUT ROUTINE
;THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
;ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
;AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
;IT IS A COPY OF THE $ERRTYP SUBROUTINE FROM SYSMAC.
;WITH ONLY MINOR CHANGES
;FIRST IF SWITCH 6 IS SET AND SWITCH 8 RESET THEN
;ALL REGISTER CONTENTS WILL BE TYPED BEFORE REPORTING THE ERROR
;SECOND IF THE CURRENT ERROR HAS THE SAME ITEM NUMBER
;AS THE PREVIOUS ERROR THEN ONLY THE DATA WILL BE TYPED
;AND NOT THE ERROR MESSAGE AND HEADER.
PRIEM: 0 ;; PREVIOUS ITEM NO. LOCATION
$ERRTYP: MOV $SWR, -(SP) ;; GET SWITCH SETTING
BIC #1C500, (SP) ;; KEEP ONLY SWITCH 8 AND 6

```

```

13893 045112 022726 000100      CMP      #SW06,(SP)+      ;IS 6 SET AND 8 RESET
13894 045116 001001      BNE      1$              ;IF NOT BRANCH
13895 045120 000402      BR       2$              ;BRANCH IF SW 6 IS SET AND 8 RESET
13896 045122 000137 046022      1$:      JMP      @*TYPERR      ;JUMP IF SW 8 IS SET
13897                                ;OR IF SW 8 IS RESET AND SW 6 IS RESET
13898 045126 104401 000200      2$:      TYPE      ,CRLF
13899
13900 045132 104401 045140      TYPE      ,65$          ;;TYPE ASCIZ STRING
13901 045136 000405      BR       64$            ;;GET OVER THE ASCIZ
13902                                ;;65$: .ASCIZ / RHWC= /
13903 045152                                64$:
13904 045152 017746 136704      MOV      @RHWC,-(SP)    ;GET READY TO TYPE RHWC CONTENTS
13905 045156 104402      TYPOC
13906
13907
13908 045160 104401 045166      TYPE      ,67$          ;;TYPE ASCIZ STRING
13909 045164 000405      BR       66$            ;;GET OVER THE ASCIZ
13910                                ;;67$: .ASCIZ / RHBA= /
13911 045200                                66$:
13912 045200 017746 136660      MOV      @RHBA,-(SP)    ;GET READY TO TYPE RHBA CONTENTS
13913 045204 104402      TYPOC
13914
13915
13916 045206 104401 045214      TYPE      ,69$          ;;TYPE ASCIZ STRING
13917 045212 000406      BR       68$            ;;GET OVER THE ASCIZ
13918                                ;;69$: .ASCIZ / RHCS2= /
13919 045230                                68$:
13920 045230 017746 136634      MOV      @RHCS2,-(SP)  ;GET READY TO TYPE RHCS2 CONTENTS
13921 045234 104402      TYPOC
13922
13923
13924 045236 104401 045244      TYPE      ,71$          ;;TYPE ASCIZ STRING
13925 045242 000406      BR       70$            ;;GET OVER THE ASCIZ
13926                                ;;71$: .ASCIZ / RHCS1= /
13927 045260                                70$:
13928 045260 017746 136574      MOV      @RHCS1,-(SP)  ;GET READY TO TYPE RHCS1 CONTENTS
13929 045264 104402      TYPOC
13930
13931
13932 045266 104401 045274      TYPE      ,73$          ;;TYPE ASCIZ STRING
13933 045272 000406      BR       72$            ;;GET OVER THE ASCIZ
13934                                ;;73$: .ASCIZ / RHDS1= /
13935 045310                                72$:
13936 045310 017746 136556      MOV      @RHDS1,-(SP)  ;GET READY TO TYPE RHDS1 CONTENTS
13937 045314 104402      TYPOC
13938
13939
13940 045316 104401 045324      TYPE      ,75$          ;;TYPE ASCIZ STRING
13941 045322 000406      BR       74$            ;;GET OVER THE ASCIZ
13942                                ;;75$: .ASCIZ / RHER1= /
13943 045340                                74$:
13944 045340 017746 136530      MOV      @RHER1,-(SP)  ;GET READY TO TYPE RHER1 CONTENTS
13945 045344 104402      TYPOC
13946
13947
13948 045346 104401 045354      TYPE      ,77$          ;;TYPE ASCIZ STRING

```



13949	045352	000406			BR	76\$		::GET OVER THE ASCIZ
13950				::77\$:	.ASCIZ	/	RHER2= /	
13951	045370			76\$:				
13952	045370	017746	136524		MOV	QRHER2,-(SP)		;GET READY TO TYPE RHER2 CONTENTS
13953	045374	104402			TYPOC			
13954								
13955								
13956	045376	104401	045404		TYPE	,79\$		::TYPE ASCIZ STRING
13957	045402	000406			BR	78\$		::GET OVER THE ASCIZ
13958				::79\$:	.ASCIZ	/	RHER3= /	
13959	045420			78\$:				
13960	045420	017746	136476		MOV	QRHER3,-(SP)		;GET READY TO TYPE RHER3 CONTENTS
13961	045424	104402			TYPOC			
13962								
13963								
13964	045426	104401	045434		TYPE	,81\$		::TYPE ASCIZ STRING
13965	045432	000406			BR	80\$		::GET OVER THE ASCIZ
13966				::81\$:	.ASCIZ	/	RHDST= /	
13967	045450			80\$:				
13968	045450	017746	136412		MOV	QRHDST,-(SP)		;GET READY TO TYPE RHDST CONTENTS
13969	045454	104402			TYPOC			
13970								
13971								
13972	045456	104401	045464		TYPE	,83\$		::TYPE ASCIZ STRING
13973	045462	000405			BR	82\$		::GET OVER THE ASCIZ
13974				::83\$:	.ASCIZ	/	RHCA= /	
13975	045476			82\$:				
13976	045476	017746	136412		MOV	QRHCA,-(SP)		;GET READY TO TYPE RHCA CONTENTS
13977	045502	104402			TYPOC			
13978								
13979								
13980	045504	104401	045512		TYPE	,85\$		::TYPE ASCIZ STRING
13981	045510	000405			BR	84\$		::GET OVER THE ASCIZ
13982				::85\$:	.ASCIZ	/	RHAS= /	
13983	045524			84\$:				
13984	045524	017746	136346		MOV	QRHAS,-(SP)		;GET READY TO TYPE RHAS CONTENTS
13985	045530	104402			TYPOC			
13986								
13987								
13988	045532	104401	045540		TYPE	,87\$		::TYPE ASCIZ STRING
13989	045536	000405			BR	86\$		::GET OVER THE ASCIZ
13990				::87\$:	.ASCIZ	/	RHOF= /	
13991	045552			86\$:				
13992	045552	017746	136334		MOV	QRHOF,-(SP)		;GET READY TO TYPE RHOF CONTENTS
13993	045556	104402			TYPOC			
13994								
13995								
13996	045560	104401	045566		TYPE	,89\$		::TYPE ASCIZ STRING
13997	045564	000405			BR	88\$		::GET OVER THE ASCIZ
13998				::89\$:	.ASCIZ	/	RHMR= /	
13999	045600			88\$:				
14000	045600	017746	136300		MOV	QRHMR,-(SP)		;GET READY TO TYPE RHMR CONTENTS
14001	045604	104402			TYPOC			
14002								
14003								
14004	045606	104401	045614		TYPE	,91\$		::TYPE ASCIZ STRING



```

14061 046044 104402          TYP0C          ;GO TYPE--OCTAL ASCII(ALL DIGITS)
14062 046046 000454          BR          10$      ;GET OUT
14063 046050 005300          1$: DEC        RO      ;ADJUST THE INDEX SO THAT IT WILL
14064 046052 006300          ASL        RO      ;      WORK FOR THE ERROR TABLE
14065 046054 006300          ASL        RO
14066 046056 006300          ASL        RO
14067 046060 062700 001226   ADD        #SERRTB,RO ;FORM TABLE POINTER
14068 046064 020037 045100   CMP        RO,@#PRITEM ;WAS PREVIOUS ERROR SAME
14069 046070 001002          BNE        13$      ;BRANCH IF NOT
14070 046072 022020          CMP        (RO)+,(RO)+ ;POP RO OVER EM AND DH
14071 046074 000420          BR          5$
14072 046076 010037 045100   13$: MOV        RO,@#PRITEM ;SAVE NEW ERROR ITEM
14073 046102 012037 046112   MOV        (RO)+,2$   ;PICKUP "ERROR MESSAGE" POINTER
14074 046106 001404          BEQ        3$        ;SKIP TYPEOUT IF NO POINTER
14075 046110 104401          TYPE      ;TYPE THE "ERROR MESSAGE"
14076 046112 000000          .WORD     0          ;"ERROR MESSAGE" POINTER GOES HERE
14077 046114 104401 001223   TYPE      .$CRLF     ;"CARRIAGE RETURN" & "LINE FEED"
14078 046120 012037 046130   3$: MOV        (RO)+,4$   ;PICKUP "DATA HEADER" POINTER
14079 046124 001404          BEQ        5$        ;SKIP TYPEOUT IF 0
14080 046126 104401          TYPE      ;TYPE THE "DATA HEADER"
14081 046130 000000          .WORD     0          ;"DATA HEADER" POINTER GOES HERE
14082 046132 104401 001223   TYPE      .$CRLF     ;"CARRIAGE RETURN" & "LINE FEED"
14083 046136 010146          5$: MOV        R1,-(SP) ;SAVE R1
14084 046140 012001          MOV        (RO)+,R1  ;PICKUP "DATA TABLE" POINTER
14085 046142 001415          BEQ        9$        ;BR IF NO DATA TO BE TYPED
14086 046144 012000          MOV        (RO)+,RO  ;PICKUP "DATA FORMAT" POINTER
14087 046146 105720          6$: TSTB       (RO)+   ;"OCTAL" OR "DECIMAL"
14088 046150 001003          BNE        7$        ;BR IF DECIMAL
14089 046152 013146          MOV        @ (R1)+,-(SP) ;SAVE @ (R1)+ FOR TYPEOUT
14090 046154 104402          TYP0C      ;GO TYPE--OCTAL ASCII(ALL DIGITS)
14091 046156 000402          BR          8$
14092 046160          7$:
14093 046160 013146          MOV        @ (R1)+,-(SP) ;SAVE @ (R1)+ FOR TYPEOUT
14094 046162 104405          TYPDS      ;GO TYPE--DECIMAL ASCII WITH SIGN
14095 046164 005711          8$: TST        (R1)   ;IS THERE ANOTHER NUMBER?
14096 046166 001403          BEQ        9$        ;BR IF NO
14097 046170 104401 046216   TYPE      .11$      ;TYPE TWO(2) SPACES
14098 046174 000764          BR          6$
14099
14100 046176 012601          9$: MOV        (SP)+,R1  ;RESTORE R1
14101 046200 012600          10$: MOV       (SP)+,RO ;"CARRIAGE RETURN" & "LINE FEED"
14102 046202 023737 000042 000046  CMP        @#42,@#46 ;ARE WE IN QV OR AUTO ACCEPT MODE?
14103 046210 001001          BNE        .+4      ;BRANCH OVER HALT IF NOT
14104 046212 000000          HALT      ;HALT AFTER ERROR MESSAGE
14105 046214 000207          RTS       PC        ;RETURN
14106 046216 020040 000          11$: .ASCIZ  / /      ;TWO(2) SPACES
14107          .EVEN
14108          ;*****
14109          ;*****
14110          .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
14111
14112          ;*****
14113          ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
14114          ;*OCTAL (ASCII) NUMBER AND TYPE IT.
14115          ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
14116          ;*CALL:

```

```

14117      ;*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
14118      ;*      TYPOS      ;;CALL FOR TYPEOUT
14119      ;*      .BYTE      N      ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
14120      ;*      .BYTE      M      ;;M=1 OR 0
14121      ;*      ;;1=TYPE LEADING ZEROS
14122      ;*      ;;0=SUPPRESS LEADING ZEROS
14123      ;*
14124      ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
14125      ;*$TYPOS OR $TYPOC
14126      ;*CALL:
14127      ;*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
14128      ;*      TYPON      ;;CALL FOR TYPEOUT
14129      ;*
14130      ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
14131      ;*CALL:
14132      ;*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
14133      ;*      TYPOC      ;;CALL FOR TYPEOUT
14134      ;*
14135      046222 017646 000000      $TYPOS: MOV      2(SP),-(SP)      ;;PICKUP THE MODE
14136      046226 116637 000001 046445      MOVVB     1(SP),$OFILL      ;;LOAD ZERO FILL SWITCH
14137      046234 112637 046447      MOVVB     (SP)+,$SOMODE+1      ;;NUMBER OF DIGITS TO TYPE
14138      046240 062716 000002      ADD      #2,(SP)      ;;ADJUST RETURN ADDRESS
14139      046244 000406      BR      $TYPON
14140      046246 112737 000001 046445      $TYPOC: MOVVB     #1,$OFILL      ;;SET THE ZERO FILL SWITCH
14141      046254 112737 000006 046447      MOVVB     #6,$SOMODE+1      ;;SET FOR SIX(6) DIGITS
14142      046262 112737 000005 046444      $TYPON: MOVVB     #5,$OCNT      ;;SET THE ITERATION COUNT
14143      046270 010346      MOV      R3,-(SP)      ;;SAVE R3
14144      046272 010446      MOV      R4,-(SP)      ;;SAVE R4
14145      046274 010546      MOV      R5,-(SP)      ;;SAVE R5
14146      046276 113704 046447      MOVVB     $SOMODE+1,R4      ;;GET THE NUMBER OF DIGITS TO TYPE
14147      046302 005404      NEG      R4
14148      046304 062704 000006      ADD      #6,R4      ;;SUBTRACT IT FOR MAX. ALLOWED
14149      046310 110437 046446      MOVVB     R4,$SOMODE      ;;SAVE IT FOR USE
14150      046314 113704 046445      MOVVB     $OFILL,R4      ;;GET THE ZERO FILL SWITCH
14151      046320 016605 000012      MOV      12(SP),R5      ;;PICKUP THE INPUT NUMBER
14152      046324 005003      CLR      R3      ;;CLEAR THE OUTPUT WORD
14153      046326 006105      1$:      ROL      R5      ;;ROTATE MSB INTO "C"
14154      046330 000404      BR      3$      ;;GO DO MSB
14155      046332 006105      2$:      ROL      R5      ;;FORM THIS DIGIT
14156      046334 006105      ROL      R5
14157      046336 006105      ROL      R5
14158      046340 010503      MOV      R5,R3
14159      046342 006103      3$:      ROL      R3      ;;GET LSB OF THIS DIGIT
14160      046344 105337 046446      DECB     $SOMODE      ;;TYPE THIS DIGIT?
14161      046350 100016      BPL      7$      ;;BR IF NO
14162      046352 042703 177770      BIC      #177770,R3      ;;GET RID OF JUNK
14163      046356 001002      BNE      4$      ;;TEST FOR 0
14164      046360 005704      TST      R4      ;;SUPPRESS THIS 0?
14165      046362 001403      BEQ      5$      ;;BR IF YES
14166      046364 005204      4$:      INC      R4      ;;DON'T SUPPRESS ANYMORE 0'S
14167      046366 052703 000060      BIS      #'0,R3      ;;MAKE THIS DIGIT ASCII
14168      046372 052703 000040      5$:      BIS      #' ,R3      ;;MAKE ASCII IF NOT ALREADY
14169      046376 110337 046442      MOVVB     R3,8$      ;;SAVE FOR TYPING
14170      046402 104401 046442      TYPE     8$      ;;GO TYPE THIS DIGIT
14171      046406 105337 046444      7$:      DECB     $OCNT      ;;COUNT BY 1
14172      046412 003347      BGT      2$      ;;BR IF MORE TO DO

```

14173	046414	002402		BLT	6\$	;;BR IF DONE
14174	046416	005204		INC	R4	;;INSURE LAST DIGIT ISN'T A BLANK
14175	046420	000744		BR	2\$	;;GO DO THE LAST DIGIT
14176	046422	012605	6\$:	MOV	(SP)+,R5	;;RESTORE R5
14177	046424	012604		MOV	(SP)+,R4	;;RESTORE R4
14178	046426	012603		MOV	(SP)+,R3	;;RESTORE R3
14179	046430	016666	000002 000004	MOV	2(SP),4(SP)	;;SET THE STACK FOR RETURNING
14180	046436	012616		MOV	(SP)+,(SP)	
14181	046440	000002		RTI		;;RETURN
14182	046442	000	8\$:	.BYTE	0	;;STORAGE FOR ASCII DIGIT
14183	046443	000		.BYTE	0	;;TERMINATOR FOR TYPE ROUTINE
14184	046444	000	\$OCNT:	.BYTE	0	;;OCTAL DIGIT COUNTER
14185	046445	000	\$OFILL:	.BYTE	0	;;ZERO FILL SWITCH
14186	046446	000000	\$OMODE:	.WORD	0	;;NUMBER OF DIGITS TO TYPE

14197  
14188  
14189  
14190  
14191  
14192  
14193  
14194  
14195 046450 010046  
14196 046452 016600 000002  
14197 046456 005740  
14198 046460 111000  
14199 046462 006300  
14200 046464 016000 046504  
14201 046470 000200  
14202  
14203  
14204  
14205  
14206 046472 011646  
14207 046474 016666 000004 000002  
14208 046502 000002  
14209  
14210  
14211  
14212  
14213  
14214  
14215  
14216  
14217 046504 046472  
14218 046506 043624  
14219 046510 046246  
14220 046512 046222  
14221 046514 046262  
14222 046516 043400  
14223  
14224  
14225 046520 044334  
14226 046522 044424  
14227 046524 044604  
14228 046526 040460  
14229  
14230  
14231

.SBTTL TRAP DECODER

;;\*\*\*\*\*  
;\*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION  
;\*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS  
;\*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL  
;\*GO TO THAT ROUTINE.

```
$TRAP:  MOV    RO, -(SP)      ;; SAVE RO
        MOV    2(SP), RO    ;; GET TRAP ADDRESS
        TST   -(RO)        ;; BACKUP BY 2
        MOVB  (RO), RO      ;; GET RIGHT BYTE OF TRAP
        ASL   RO           ;; POSITION FOR INDEXING
        MOV   $TRPAD(RO), RO ;; INDEX TO TABLE
        RTS   RO           ;; GO TO ROUTINE
```

;;THIS IS USE TO HANDLE THE "GETPRI" MACRO

```
$TRAP2: MOV   (SP), -(SP)    ;; MOVE THE PC DOWN
        MOV   4(SP), 2(SP)  ;; MOVE THE PSW DOWN
        RTI                          ;; RESTORE THE PSW
```

.SBTTL TRAP TABLE

;;\*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED  
;;\*BY THE "TRAP" INSTRUCTION.

	ROUTINE		
\$TRPAD:	WORD	\$TRAP2	
	\$TYPE	;;CALL=TYPE	TRAP+1(104401) TTY TYPEOUT ROUTINE
	\$TYPOC	;;CALL=TYPOC	TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
	\$TYPOS	;;CALL=TYPOS	TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
	\$TYPON	;;CALL=TYPON	TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
	\$TYPDS	;;CALL=TYPDS	TRAP+5(104405) TYPE DECIMAL NUMBER (WITH SIGN)
	\$RDCHR	;;CALL=RDCHR	TRAP+6(104406) TTY TYPEIN CHARACTER ROUTINE
	\$RDLIN	;;CALL=RDLIN	TRAP+7(104407) TTY TYPEIN STRING ROUTINE
	\$RDOCT	;;CALL=RDOCT	TRAP+10(104410) READ AN OCTAL NUMBER FROM TTY
	WAIT.T	;;CALL=WAT	TRAP+11(104411) DONT ADD ABOVE THIS TRAP

```

14232          .SBTTL  POWER DOWN AND UP ROUTINES
14233
14234          ;;*****
14235          :POWER DOWN ROUTINE
14236 046530 012737 046674 000024 $PWRDN: MOV  $SILLUP, @#PWRVEC  ;; SET FOR FAST UP
14237 046536 012737 000340 000026      MOV  #340, @#PWRVEC+2  ;; PRIO:7
14238 046544 010046          MOV  R0, -(SP)        ;; PUSH R0 ON STACK
14239 046546 010146          MOV  R1, -(SP)        ;; PUSH R1 ON STACK
14240 046550 010246          MOV  R2, -(SP)        ;; PUSH R2 ON STACK
14241 046552 010346          MOV  R3, -(SP)        ;; PUSH R3 ON STACK
14242 046554 010446          MOV  R4, -(SP)        ;; PUSH R4 ON STACK
14243 046556 010546          MOV  R5, -(SP)        ;; PUSH R5 ON STACK
14244 046560 017746 132354          MOV  @SWR, -(SP)      ;; PUSH @SWR ON STACK
14245 046564 010637 046700          MOV  SP, $SAVR6      ;; SAVE SP
14246 046570 012737 046602 000024      MOV  $PWRUP, @#PWRVEC ;; SET UP VECTOR
14247 046576 000000          HALT
14248 046600 000776          BR   .-2            ;; HANG UP
14249
14250          ;;*****
14251          :POWER UP ROUTINE
14252 046602 012737 046674 000024 $PWRUP: MOV  $SILLUP, @#PWRVEC  ;; SET FOR FAST DOWN
14253 046610 013706 046700          MOV  $SAVR6, SP      ;; GET SP
14254 046614 005037 046700          CLR  $SAVR6         ;; WAIT LOOP FOR THE TTY
14255 046620 005237 046700 1$: INC  $SAVR6         ;; WAIT FOR THE INC
14256 046624 001375          BNE  1$             ;; OF WORD
14257 046626 012677 132306          MOV  (SP)+, @SWR    ;; POP STACK INTO @SWR
14258 046632 012605          MOV  (SP)+, R5      ;; POP STACK INTO R5
14259 046634 012604          MOV  (SP)+, R4      ;; POP STACK INTO R4
14260 046636 012603          MOV  (SP)+, R3      ;; POP STACK INTO R3
14261 046640 012602          MOV  (SP)+, R2      ;; POP STACK INTO R2
14262 046642 012601          MOV  (SP)+, R1      ;; POP STACK INTO R1
14263 046644 012600          MOV  (SP)+, R0      ;; POP STACK INTO R0
14264 046646 012737 046530 000024      MOV  $PWRDN, @#PWRVEC ;; SET UP THE POWER DOWN VECTOR
14265 046654 012737 000340 000026      MOV  #340, @#PWRVEC+2 ;; PRIO:7
14266 046662 104401          - TYPE             ;; REPORT THE POWER FAILURE
14267 046664 046702 $PWRMG: .WORD $POWER  ;; POWER FAIL MESSAGE POINTER
14268 046666 012716          MOV  (PC)+, (SP)   ;; RESTART AT BEGIN
14269 046670 005522 $PWRAD: .WORD BEGIN  ;; RESTART ADDRESS
14270 046672 000002          RTI
14271 046674 000000 $SILLUP: HALT        ;; THE POWER UP SEQUENCE WAS STARTED
14272 046676 000776          BR   .-2            ;; BEFORE THE POWER DOWN WAS COMPLETE
14273 046700 000000 $SAVR6: 0            ;; PUT THE SP HERE
14274 046702 005015 047520 042527 $POWER: .ASCIZ <15><12>"POWER"
14275 046710 000122
14276          .EVEN

```

```

14277
14278
14279
14280
14281
14282
14283 046712 044124 020105 044122
14284 046720 041040 051501 020105
14285 046726 042101 051104 051505
14286 046734 020123 040527 020123
14287 046742 033461 030062 030064
14288 046750 044440 042116 041511
14289 046756 052101 047111 020107
14290 046764 047101 051040 020123
14291 046772 052502 020124 044124
14292 047000 020105 051104 053111
14293 047006 020105 054524 042520
14294 047014 005015
14295 047016 044504 020104 047516
14296 047024 020124 047503 052116
14297 047032 044501 020116 020060
14298 047040 051117 030440 047440
14299 047046 020122 020062 051117
14300 047054 031440 047440 020122
14301 047062 000064
14302
14303 047064 044124 020105 044122 EM2: .ASCII /THE RH BASE ADDRESS WAS 176700 INDICATING AN RPD4,5,6 BUT THE DRIVE TYP
14304 047072 041040 051501 020105
14305 047100 042101 051104 051505
14306 047106 020123 040527 020123
14307 047114 033461 033466 030060
14308 047122 044440 042116 041511
14309 047130 052101 047111 020107
14310 047136 047101 051040 030120
14311 047144 026064 026065 020066
14312 047152 052502 020124 044124
14313 047160 020105 051104 053111
14314 047166 020105 054524 042520
14315 047174 005015
14316 047176 044504 020104 047516
14317 047204 020124 047503 052116
14318 047212 044501 020116 031062
14319 047220 031060 026060 032062
14320 047226 031060 026060 030062
14321 047234 031060 026061 032062
14322 047242 031060 026061 030062
14323 047250 031060 026062 047440
14324 047256 020122 032062 031060
14325 047264 000062
14326
14327 047266 044124 020105 044122 EM3: .ASCII /THE RH BASE ADDRESS WAS 172440 INDICATING A TMO2 BUT THE DRIVE TYPE/<15
14328 047274 041040 051501 020105
14329 047302 042101 051104 051505
14330 047310 020123 040527 020123
14331 047316 033461 032062 030064
14332 047324 044440 042116 041511

```



14333	047332	052101	047111	020107	
14334	047340	020101	046524	031060	
14335	047346	041040	052125	052040	
14336	047354	042510	042040	044522	
14337	047362	042526	052040	050131	
14338	047370	006505	012		
14339	047373	104	042111	047040	.ASCIZ /DID NOT CONTAIN 142010/
14340	047400	052117	041440	047117	
14341	047406	040524	047111	030440	
14342	047414	031064	030460	000060	
14343					
14344	047422	044124	020105	044122	EM4: .ASCII /THE RH BASE ADDRESS WAS 176300 INDICATING MORE THAN ONE RH DEVICE BUT/
14345	047430	041040	051501	020105	
14346	047436	042101	051104	051505	
14347	047444	020123	040527	020123	
14348	047452	033461	031466	030060	
14349	047460	044440	042116	041511	
14350	047466	052101	047111	020107	
14351	047474	047515	042522	052040	
14352	047502	040510	020116	047117	
14353	047510	020105	044122	042040	
14354	047516	053105	041511	020105	
14355	047524	052502	006524	012	
14356	047531	124	042510	042040	.ASCII /THE DRIVE TYPE DID NOT CONTAIN 0,1,2,3,4,20020,24020, /<15><12>
14357	047536	044522	042526	052040	
14358	047544	050131	020105	044504	
14359	047552	020104	047516	020124	
14360	047560	047503	052116	044501	
14361	047566	020116	026060	026061	
14362	047574	026062	026063	026064	
14363	047602	030062	031060	026060	
14364	047610	032062	031060	026060	
14365	047616	005015			
14366	047620	030062	031060	026061	.ASCIZ /20021,24021,20022,24022,142010/<15><12>
14367	047626	032062	031060	026061	
14368	047634	030062	031060	026062	
14369	047642	032062	031060	026062	
14370	047650	032061	030062	030061	
14371	047656	005015	000		
14372	047661	111	042116	041511	.ASCIZ /INDICATING NO RH DEVICE PRESENT/
14373	047666	052101	047111	020107	
14374	047674	047516	051040	020110	
14375	047702	042504	044526	042503	
14376	047710	050040	042522	042523	
14377	047716	052116	000		
14378					
14379	047721	124	042510	052440	EM5: .ASCII /THE UNIBUS TIMED OUT FOR EACH OF THE FOLLOWING ADDRESSES. INDICATING/
14380	047726	044516	052502	020123	
14381	047734	044524	042515	020104	
14382	047742	052517	020124	047506	
14383	047750	020122	040505	044103	
14384	047756	047440	020106	044124	
14385	047764	020105	047506	046114	
14386	047772	053517	047111	020107	
14387	050000	042101	051104	051505	
14388	050006	042523	026123	044440	

14389	050014	042116	041511	052101	
14390	050022	047111	006507	012	
14391	050027	116	020117	044122	.ASCIZ /NO RH ON THE SYSTEM SO ABORT PROGRAM/
14392	050034	047440	020116	044124	
14393	050042	020105	054523	052123	
14394	050050	046505	051440	020117	
14395	050056	041101	051117	020124	
14396	050064	051120	043517	040522	
14397	050072	000115			
14398					
14399	050074	043101	042524	020122	EM6: .ASCII /AFTER AN RH CLEAR (RHCS2-BIT #5) RHCS2 DOES NOT HAVE ONLY IR/<15><12>
14400	050102	047101	051040	020110	
14401	050110	046103	040505	020122	
14402	050116	051050	041510	031123	
14403	050124	041055	052111	021440	
14404	050132	024465	051040	041510	
14405	050140	031123	042040	042517	
14406	050146	020123	047516	020124	
14407	050154	040510	042526	047440	
14408	050162	046116	020131	051111	
14409	050170	005015			
14410	050172	047101	020104	047125	.ASCIZ /AND UNIT NUMBER/
14411	050200	052111	047040	046525	
14412	050206	042502	000122		
14413					
14414	050212	043101	042524	020122	EM7: .ASCII /AFTER CLEARING THE RH AND WRITING ONE WORD INTO RHDB AND/<15><12>
14415	050220	046103	040505	044522	
14416	050226	043516	052040	042510	
14417	050234	051040	020110	047101	
14418	050242	020104	051127	052111	
14419	050250	047111	020107	047117	
14420	050256	020105	047527	042122	
14421	050264	044440	052116	020117	
14422	050272	044122	041104	040440	
14423	050300	042116	005015		
14424	050304	042522	042101	047111	.ASCIZ /READING IT BACK, CAUSED RHDB TO HAVE WRONG VALUE GIVEN IN "BAD"/
14425	050312	020107	052111	041040	
14426	050320	041501	026113	041440	
14427	050326	052501	042523	020104	
14428	050334	044122	041104	052040	
14429	050342	020117	040510	042526	
14430	050350	053440	047522	043516	
14431	050356	053040	046101	042525	
14432	050364	043440	053111	047105	
14433	050372	044440	020116	041042	
14434	050400	042101	000042		
14435					
14436	050404	043101	042524	020122	EM10: .ASCII /AFTER CLEARING THE RH AND WRITING ONE WORD INTO RHDB AND/<15><12>
14437	050412	046103	040505	044522	
14438	050420	043516	052040	042510	
14439	050426	051040	020110	047101	
14440	050434	020104	051127	052111	
14441	050442	047111	020107	047117	
14442	050450	020105	047527	042122	
14443	050456	044440	052116	020117	
14444	050464	044122	041104	040440	

14445	050472	042116	005015		
14446	050476	042522	042101	047111	.ASCIZ /READING IT BACK, CAUSED RHCS2 TO HAVE WRONG VALUE GIVEN IN "BAD"/
14447	050504	020107	052111	041040	
14448	050512	041501	026113	041440	
14449	050520	052501	042523	020104	
14450	050526	044122	051503	020062	
14451	050534	047524	044040	053101	
14452	050542	020105	051127	047117	
14453	050550	020107	040526	052514	
14454	050556	020105	044507	042526	
14455	050564	020116	047111	021040	
14456	050572	040502	021104	000	
14457					
14458	050577	101	052106	051105	EM11: .ASCII /AFTER CLEARING THE RH AND WRITING ONE WORD INTO RHDB AND/<15><12>
14459	050604	041440	042514	051101	
14460	050612	047111	020107	044124	
14461	050620	020105	044122	040440	
14462	050626	042116	053440	044522	
14463	050634	044524	043516	047440	
14464	050642	042516	053440	051117	
14465	050650	020104	047111	047524	
14466	050656	051040	042110	020102	
14467	050664	047101	006504	012	
14468	050671	122	040505	044504	.ASCIZ /READING IT BACK, CAUSED RHCS1 TO HAVE WRONG VALUE GIVEN IN "BAD"/
14469	050676	043516	044440	020124	
14470	050704	040502	045503	020054	
14471	050712	040503	051525	042105	
14472	050720	051040	041510	030523	
14473	050726	052040	020117	040510	
14474	050734	042526	053440	047522	
14475	050742	043516	053040	046101	
14476	050750	042525	043440	053111	
14477	050756	047105	044440	020116	
14478	050764	041042	042101	000042	
14479					
14480	050772	043101	042524	020122	EM12: .ASCII /AFTER CLEARING THE RH AND WRITING ONE WORD INTO RHDB AND/<15><12>
14481	051000	046103	040505	044522	
14482	051006	043516	052040	042510	
14483	051014	051040	020110	047101	
14484	051022	020104	051127	052111	
14485	051030	047111	020107	047117	
14486	051036	020105	047527	042122	
14487	051044	044440	052116	020117	
14488	051052	044122	041104	040440	
14489	051060	042116	005015		
14490	051064	042522	042101	047111	.ASCIZ /READING IT BACK, CAUSED RHCS3 TO HAVE WRONG VALUE GIVEN IN "BAD"/
14491	051072	020107	052111	041040	
14492	051100	041501	026113	041440	
14493	051106	052501	042523	020104	
14494	051114	044122	051503	020063	
14495	051122	047524	044040	053101	
14496	051130	020105	051127	047117	
14497	051136	020107	040526	052514	
14498	051144	020105	044507	042526	
14499	051152	020116	047111	021040	
14500	051160	040502	021104	000	

14501					
14502	051165	101	052106	051105	EM13: .ASCII /AFTER CLEARING THE RH AND WRITING ONE WORD INTO RHDB AND/<15><12>
14503	051172	041440	042514	051101	
14504	051200	047111	020107	044124	
14505	051206	020105	044122	040440	
14506	051214	042116	053440	044522	
14507	051222	044524	043516	047440	
14508	051230	042516	053440	051117	
14509	051236	020104	047111	047524	
14510	051244	051040	042110	020102	
14511	051252	047101	006504	012	
14512	051257	122	040505	044504	.ASCIZ /READING IT BACK, CAUSED RHBA TO HAVE WRONG VALUE GIVEN IN "BAD"/
14513	051264	043516	044440	020124	
14514	051272	040502	045503	020054	
14515	051300	040503	051525	042105	
14516	051306	051040	041110	020101	
14517	051314	047524	044040	053101	
14518	051322	020105	051127	047117	
14519	051330	020107	040526	052514	
14520	051336	020105	044507	042526	
14521	051344	020116	047111	021040	
14522	051352	040502	021104	000	
14523					
14524	051357	101	052106	051105	EM14: .ASCII /AFTER CLEARING THE RH AND WRITING ONE WORD INTO RHDB AND/<15><12>
14525	051364	041440	042514	051101	
14526	051372	047111	020107	044124	
14527	051400	020105	044122	040440	
14528	051406	042116	053440	044522	
14529	051414	044524	043516	047440	
14530	051422	042516	053440	051117	
14531	051430	020104	047111	047524	
14532	051436	051040	042110	020102	
14533	051444	047101	006504	012	
14534	051451	122	040505	044504	.ASCIZ /READING IT BACK, CAUSED RHBAE TO HAVE WRONG VALUE GIVEN IN "BAD"/
14535	051456	043516	044440	020124	
14536	051464	040502	045503	020054	
14537	051472	040503	051525	042105	
14538	051500	051040	041110	042501	
14539	051506	052040	020117	040510	
14540	051514	042526	053440	047522	
14541	051522	043516	053040	046101	
14542	051530	042525	043440	053111	
14543	051536	047105	044440	020116	
14544	051544	041042	042101	000042	
14545					
14546	051552	043101	042524	020122	EM15: .ASCII /AFTER CLEARING THE RH AND WRITING ONE WORD INTO RHDB AND/<15><12>
14547	051560	046103	040505	044522	
14548	051566	043516	052040	042510	
14549	051574	051040	020110	047101	
14550	051602	020104	051127	052111	
14551	051610	047111	020107	047117	
14552	051616	020105	047527	042122	
14553	051624	044440	052116	020117	
14554	051632	044122	041104	040440	
14555	051640	042116	005015		
14556	051644	042522	042101	047111	.ASCIZ /READING IT BACK, CAUSED RHWC TO HAVE WRONG VALUE GIVEN IN "BAD"/

14557	051652	020107	052111	041040	
14558	051660	041501	026113	041440	
14559	051666	052501	042523	020104	
14560	051674	044122	041527	052040	
14561	051702	020117	040510	042526	
14562	051710	053440	047522	043516	
14563	051716	053040	046101	042525	
14564	051724	043440	053111	047105	
14565	051732	044440	020116	041042	
14566	051740	042101	000042		
14567					
14568	051744	043101	042524	020122	EM16: .ASCII /AFTER CLEARING THE RH AND WRITING ONE WORD INTO RHDB AND/<15><12>
14569	051752	046103	040505	044522	
14570	051760	043516	052040	042510	
14571	051766	051040	020110	047101	
14572	051774	020104	051127	052111	
14573	052002	047111	020107	047117	
14574	052010	020105	047527	042122	
14575	052016	044440	052116	020117	
14576	052024	044122	041104	040440	
14577	052032	042116	005015		
14578	052036	042522	042101	047111	.ASCIZ /READING IT BACK, CAUSED RHCS2 TO HAVE WRONG VALUE GIVEN IN "BAD"/
14579	052044	020107	052111	041040	
14580	052052	041501	026113	041440	
14581	052060	052501	042523	020104	
14582	052066	044122	051503	020062	
14583	052074	047524	044040	053101	
14584	052102	020105	051127	047117	
14585	052110	020107	040526	052514	
14586	052116	020105	044507	042526	
14587	052124	020116	047111	021040	
14588	052132	040502	021104	000	
14589					
14590	052137	101	052106	051105	EM17: .ASCII /AFTER CLEARING THE RH AND WRITING TWO WORD INTO RHDB AND/<15><12>
14591	052144	041440	042514	051101	
14592	052152	047111	020107	044124	
14593	052160	020105	044122	040440	
14594	052166	042116	053440	044522	
14595	052174	044524	043516	052040	
14596	052202	047527	053440	051117	
14597	052210	020104	047111	047524	
14598	052216	051040	042110	020102	
14599	052224	047101	006504	012	
14600	052231	122	040505	044504	.ASCIZ /READING IT BACK, CAUSED RHDB TO HAVE WRONG VALUE GIVEN IN "BAD"/
14601	052236	043516	044440	020124	
14602	052244	040502	045503	020054	
14603	052252	040503	051525	042105	
14604	052260	051040	042110	020102	
14605	052266	047524	044040	053101	
14606	052274	020105	051127	047117	
14607	052302	020107	040526	052514	
14608	052310	020105	044507	042526	
14609	052316	020116	047111	021040	
14610	052324	040502	021104	000	
14611					
14612	052331	101	052106	051105	EM20: .ASCII /AFTER CLEARING THE RH AND WRITING TWO WORD INTO RHDB AND/<15><12>

14613	052336	041440	042514	051101
14614	052344	047111	020107	044124
14615	052352	020105	044122	040440
14616	052360	042116	053440	044522
14617	052366	044524	043516	052040
14618	052374	047527	053440	051117
14619	052402	020104	047111	047524
14620	052410	051040	042110	020102
14621	052416	047101	006504	012
14622	052423	122	040505	044504
14623	052430	043516	047440	042516
14624	052436	041040	041501	026113
14625	052444	041440	052501	042523
14626	052452	020104	044122	051503
14627	052460	020062	047524	044040
14628	052466	053101	020105	051127
14629	052474	047117	020107	040526
14630	052502	052514	020105	044507
14631	052510	042526	020116	047111
14632	052516	021040	040502	021104
14633	052524	000		
14634				
14635	052525	101	052106	051105
14636	052532	041440	042514	051101
14637	052540	047111	020107	044124
14638	052546	020105	044122	040440
14639	052554	042116	053440	044522
14640	052562	044524	043516	052040
14641	052570	047527	053440	051117
14642	052576	020104	047111	047524
14643	052604	051040	042110	020102
14644	052612	047101	006504	012
14645	052617	122	040505	044504
14646	052624	043516	044440	020124
14647	052632	040502	045503	052040
14648	052640	044527	042503	020054
14649	052646	040503	051525	042105
14650	052654	051040	042110	020102
14651	052662	047524	044040	053101
14652	052670	020105	051127	047117
14653	052676	020107	040526	052514
14654	052704	020105	044507	042526
14655	052712	020116	047111	021040
14656	052720	040502	021104	000
14657				
14658	052725	101	052106	051105
14659	052732	041440	042514	051101
14660	052740	047111	020107	044124
14661	052746	020105	044122	040440
14662	052754	042116	053440	044522
14663	052762	044524	043516	052040
14664	052770	047527	053440	051117
14665	052776	020104	047111	047524
14666	053004	051040	042110	020102
14667	053012	047101	006504	012
14668	053017	122	040505	044504

.ASCIZ /READING ONE BACK, CAUSED RHCS2 TO HAVE WRONG VALUE GIVEN IN "BAD"/

EM21: .ASCII /AFTER CLEARING THE RH AND WRITING TWO WORD INTO RHDB AND/<15><12>

.ASCIZ /READING IT BACK TWICE, CAUSED RHDB TO HAVE WRONG VALUE GIVEN IN "BAD"/

EM22: .ASCII /AFTER CLEARING THE RH AND WRITING TWO WORD INTO RHDB AND/<15><12>

.ASCIZ /READING IT BACK TWICE, CAUSED RHCS2 TO HAVE WRONG VALUE GIVEN IN "BAD"/

14669	053024	043516	044440	020124
14670	053032	040502	045503	052040
14671	053040	044527	042503	020054
14672	053046	040503	051525	042105
14673	053054	051040	041510	031123
14674	053062	052040	020117	040510
14675	053070	042526	053440	047522
14676	053076	043516	053040	046101
14677	053104	042525	043440	053111
14678	053112	047105	044440	020116
14679	053120	041042	042101	000042
14680				
14681	053126	043101	042524	020122
14682	053134	046103	040505	044522
14683	053142	043516	052040	042510
14684	053150	051040	020110	047101
14685	053156	020104	051127	052111
14686	053164	047111	020107	044505
14687	053172	044107	020124	047527
14688	053200	042122	044440	052116
14689	053206	020117	044122	041104
14690	053214	040440	042116	005015
14691	053222	042522	042101	047111
14692	053230	020107	052111	041040
14693	053236	041501	026113	041440
14694	053244	052501	042523	020104
14695	053252	044122	051503	020062
14696	053260	047524	044040	053101
14697	053266	020105	051127	047117
14698	053274	020107	040526	052514
14699	053302	020105	044507	042526
14700	053310	020116	047111	021040
14701	053316	040502	021104	000
14702				
14703	053323	124	042510	051040
14704	053330	020110	040527	020123
14705	053336	046103	040505	042522
14706	053344	020104	047101	020104
14707	053352	020101	040520	052124
14708	053360	051105	020116	043117
14709	053366	034040	053440	051117
14710	053374	051504	053440	051105
14711	053402	020105	051127	052111
14712	053410	042524	020116	047111
14713	053416	047524	051040	042110
14714	053424	006502	012	
14715	053427	122	040505	044504
14716	053434	043516	051040	042110
14717	053442	020102	047506	020122
14718	053450	044124	020105	047042
14719	053456	020042	044124	020056
14720	053464	044524	042515	043440
14721	053472	053101	020105	051127
14722	053500	047117	020107	040526
14723	053506	052514	020105	047111
14724	053514	051040	042110	006502

EM23: .ASCII /AFTER CLEARING THE RH AND WRITING EIGHT WORD INTO RHDB AND/<15><12>

.ASCIZ /READING IT BACK, CAUSED RHCS2 TO HAVE WRONG VALUE GIVEN IN "BAD"/

EM24: .ASCII /THE RH WAS CLEARED AND A PATTERN OF 8 WORDS WERE WRITTEN INTO RHDB/<15>

.ASCII /READING RHDB FOR THE "N" TH. TIME GAVE WRONG VALUE IN RHDB/<15><12>

14725	053522	012			
14726	053523	042	021116	044440	.ASCIZ /"N" IS GIVEN IN WORD NO/
14727	053530	020123	044507	042526	
14728	053536	020116	047111	053440	
14729	053544	051117	020104	047516	
14730	053552	000			
14731					
14732	053553	124	042510	051040	EM25: .ASCII /THE RH WAS CLEARED AND A PATTERN OF 8 WORDS WERE WRITTEN INTO RHDB/(:5)
14733	053560	020110	040527	020123	
14734	053566	046103	040505	042522	
14735	053574	020104	047101	020104	
14736	053602	020101	040520	052124	
14737	053610	051105	020116	043117	
14738	053616	034040	053440	051117	
14739	053624	051504	053440	051105	
14740	053632	020105	051127	052111	
14741	053640	042524	020116	047111	
14742	053646	047524	051040	042110	
14743	053654	006502	012		
14744	053657	101	052106	051105	.ASCIZ /AFTER READING ALL 8 WORDS, FOLLOWING REGISTER CONTAINED WRONG VALUE GIV
14745	053664	051040	040505	044504	
14746	053672	043516	040440	046114	
14747	053700	034040	053440	051117	
14748	053706	051504	020054	047506	
14749	053714	046114	053517	047111	
14750	053722	020107	042522	044507	
14751	053730	052123	051105	041440	
14752	053736	047117	040524	047111	
14753	053744	042105	053440	047522	
14754	053752	043516	053040	046101	
14755	053760	042525	043440	053111	
14756	053766	047105	044440	020116	
14757	053774	041042	042101	000042	
14758					
14759	054002	042523	052124	047111	EM33: .ASCIZ /SETTING RH CLEAR (RHCS2-BIT #5) CAUSED ERROR REGISTER 1 TO HAVE WRONG V
14760	054010	020107	044122	041440	
14761	054016	042514	051101	024040	
14762	054024	044122	051503	026462	
14763	054032	044502	020124	032443	
14764	054040	020051	040503	051525	
14765	054046	042105	042440	051122	
14766	054054	051117	051040	043505	
14767	054062	051511	042524	020122	
14768	054070	020061	047524	044040	
14769	054076	053101	020105	051127	
14770	054104	047117	020107	040526	
14771	054112	052514	000105		
14772					
14773	054116	047101	051040	020110	EM34: .ASCII /AN RH CLEAR WAS GIVEN RHER1 WAS CHECKED TO HAVE ZERO. PAT (RHCS2-BIT #4
14774	054124	046103	040505	020122	
14775	054132	040527	020123	044507	
14776	054140	042526	020116	044122	
14777	054146	051105	020061	040527	
14778	054154	020123	044103	041505	
14779	054162	042513	020104	047524	
14780	054170	044040	053101	020105	



14781	054176	042532	047522	020056
14782	054204	040520	020121	051050
14783	054212	041510	031123	041055
14784	054220	052111	021440	024464
14785	054226	053440	051501	051440
14786	054234	052105	005015	
14787	054240	047524	044440	053116
14788	054246	051105	020124	040520
14789	054254	044522	054524	041440
14790	054262	042510	045503	047111
14791	054270	027107	051040	042510
14792	054276	030522	053440	051501
14793	054304	051040	040505	020104
14794	054312	052502	020124	044504
14795	054320	020104	047516	020124
14796	054326	047503	052116	044501
14797	054334	020116	044127	052101
14798	054342	044440	020123	047111
14799	054350	021040	047507	042117
14800	054356	000042		
14801				
14802	054360	044122	041440	042514
14803	054366	051101	053440	051501
14804	054374	043440	053111	047105
14805	054402	020056	044122	051105
14806	054410	020061	040527	020123
14807	054416	044103	041505	042513
14808	054424	027104	050040	052101
14809	054432	024040	044122	051503
14810	054440	026462	044502	020124
14811	054446	032043	020051	040527
14812	054454	020123	042523	006524
14813	054462	012		
14814	054463	101	042116	051040
14815	054470	042510	030522	053440
14816	054476	051501	051040	040505
14817	054504	027104	051040	041510
14818	054512	030523	051440	047510
14819	054520	046125	020104	040510
14820	054526	042526	051040	054504
14821	054534	020054	041523	040440
14822	054542	042116	046440	050103
14823	054550	020105	042523	000124
14824				
14825	054556	044122	041440	042514
14826	054564	051101	053440	051501
14827	054572	043440	053111	047105
14828	054600	020056	044122	051105
14829	054606	020061	040527	020123
14830	054614	044103	041505	042513
14831	054622	027104	050040	052101
14832	054630	024040	044122	051503
14833	054636	026462	044502	020124
14834	054644	032043	020051	040527
14835	054652	020123	042523	006524
14836	054660	012		

.ASCIZ /TO INVERT PARITY CHECKING. RHER1 WAS READ BUT DID NOT CONTAIN WHAT IS :

EM35: .ASCII /RH CLEAR WAS GIVEN. RHER1 WAS CHECKED. PAT (RHCS2-BIT #4) WAS SET/<15><

.ASCIZ /AND RHER1 WAS READ. RHCS1 SHOULD HAVE RDY, SC AND MCPE SET/

EM36: .ASCII /RH CLEAR WAS GIVEN. RHER1 WAS CHECKED. PAT (RHCS2-BIT #4) WAS SET/<15><

14837	054661	122	042510	030522	.ASCII /RHER1 WAS READ. "1" WAS WRITTEN INTO "TRE" IN RHCS1/<15><12>
14838	054666	053440	051501	051040	
14839	054674	040505	027104	021040	
14840	054702	021061	053440	051501	
14841	054710	053440	044522	052124	
14842	054716	047105	044440	052116	
14843	054724	020117	052042	042522	
14844	054732	020042	047111	051040	
14845	054740	041510	030523	005015	
14846	054746	047117	041440	042510	.ASCIZ /ON CHECKING RHCS1 IT DID NOT CONTAIN WHAT IS IN GOOD/
14847	054754	045503	047111	020107	
14848	054762	044122	051503	020061	
14849	054770	052111	042040	042111	
14850	054776	047040	052117	041440	
14851	055004	047117	040524	047111	
14852	055012	053440	040510	020124	
14853	055020	051511	044440	020116	
14854	055026	047507	042117	000	
14855					
14856	055033	122	020110	040527	EM37: .ASCII /RH WAS CLEARED. RHER1 WAS CHECKED. PAT (RHCS2-BIT #4) WAS SET/<15><12>
14857	055040	020123	046103	040505	
14858	055046	042522	027104	051040	
14859	055054	042510	030522	053440	
14860	055062	051501	041440	042510	
14861	055070	045503	042105	020056	
14862	055076	040520	020124	051050	
14863	055104	041510	031123	041055	
14864	055112	052111	021440	024464	
14865	055120	053440	051501	051440	
14866	055126	052105	005015		
14867	055132	044122	051105	020061	.ASCII /RHER1 WAS READ. "1" WAS WRITTEN IN "TRE" IN RHCS1/<15><12>
14868	055140	040527	020123	042522	
14869	055146	042101	020056	030442	
14870	055154	020042	040527	020123	
14871	055162	051127	052111	042524	
14872	055170	020116	047111	021040	
14873	055176	051124	021105	044440	
14874	055204	020116	044122	051503	
14875	055212	006461	012		
14876	055215	101	020116	044122	.ASCII /AN RH CLEAR WAS GIVEN BUT FOLLOWING REGISTER DID NOT/<15><12>
14877	055222	041440	042514	051101	
14878	055230	053440	051501	043440	
14879	055236	053111	047105	041040	
14880	055244	052125	043040	046117	
14881	055252	047514	044527	043516	
14882	055260	051040	043505	051511	
14883	055266	042524	020122	044504	
14884	055274	020104	047516	006524	
14885	055302	012			
14886	055303	103	047117	040524	.ASCIZ /CONTAIN WHAT IS IN "GOOD"/
14887	055310	047111	053440	040510	
14888	055316	020124	051511	044440	
14889	055324	020116	043442	047517	
14890	055332	021104	000		
14891					
14892	055335	101	052106	051105	EM45: .ASCII /AFTER AN RH CLEAR "1" WAS WRITTEN IN THE DISK (TAPE) ADDRESS/<15><12>

14893	055342	040440	020116	044122
14894	055350	041440	042514	051101
14895	055356	021040	021061	053440
14896	055364	051501	052440	044522
14897	055372	052124	047105	044440
14898	055400	020116	044124	020105
14899	055406	044504	045523	024040
14900	055414	040524	042520	020051
14901	055422	042101	051104	051505
14902	055430	006523	012	
14903	055433	122	043505	051511
14904	055440	042524	027122	050040
14905	055446	052101	044440	020116
14906	055454	044122	051503	020061
14907	055462	040527	020123	042523
14908	055470	027124	047440	020116
14909	055476	042522	042101	047111
14910	055504	020107	044504	045523
14911	055512	024040	040524	042520
14912	055520	020051	042101	051104
14913	055526	051505	006523	012
14914	055533	122	043505	051511
14915	055540	042524	020122	052111
14916	055546	042040	042111	047040
14917	055554	052117	041440	047117
14918	055562	040524	047111	021040
14919	055570	021061	000	
14920				
14921	055573	101	052106	051105
14922	055600	040440	020116	044122
14923	055606	041440	042514	051101
14924	055614	021040	021061	053440
14925	055622	051501	053440	044522
14926	055630	052124	047105	044440
14927	055636	020116	044124	020105
14928	055644	044504	045523	024040
14929	055652	040524	042520	020051
14930	055660	042101	051104	051505
14931	055666	020123	042522	044507
14932	055674	052123	051105	005015
14933	055702	040520	020124	047111
14934	055710	051040	041510	030523
14935	055716	053440	051501	051440
14936	055724	052105	040440	052106
14937	055732	051105	051040	040505
14938	055740	044504	043516	042040
14939	055746	051511	020113	052050
14940	055754	050101	024505	040440
14941	055762	042104	042522	051523
14942	055770	051040	043505	051511
14943	055776	042524	020122	044124
14944	056004	020105	047506	046114
14945	056012	053517	047111	006507
14946	056020	012		
14947	056021	122	043505	051511
14948	056026	042524	020122	044504

.ASCII /REGISTER. PAT IN RHCSI WAS SET. ON READING DISK (TAPE) ADDRESS/<15><12>

.ASCIZ /REGISTER IT DID NOT CONTAIN "1"/

EM46: .ASCII /AFTER AN RH CLEAR "1" WAS WRITTEN IN THE DISK (TAPE) ADDRESS REGISTER/<

.ASCII /PAT IN RHCSI WAS SET AFTER READING DISK (TAPE) ADDRESS REGISTER THE FOL

.ASCIZ /REGISTER DID NOT CONTAIN WHAT IS IN "GOOD"/

14949	056034	020104	047516	020124
14950	056042	047503	052116	044501
14951	056050	020116	044127	052101
14952	056056	044440	020123	047111
14953	056064	021040	047507	042117
14954	056072	000042		
14955				
14956	056074	047101	051040	020110
14957	056102	046103	040505	020122
14958	056110	051050	041510	031123
14959	056116	041040	052111	021440
14960	056124	024465	053440	051501
14961	056132	043440	053111	047105
14962	056140	020056	030501	020066
14963	056146	051050	041510	030523
14964	056154	041040	052111	021440
14965	056162	024470	053440	051501
14966	056170	051440	052105	005015
14967	056176	047117	051040	040505
14968	056204	044504	043516	052040
14969	056212	042510	043040	046117
14970	056220	047514	044527	043516
14971	056226	051040	043505	051511
14972	056234	042524	020122	052111
14973	056242	042040	042111	047040
14974	056250	052117	041440	047117
14975	056256	040524	047111	053440
14976	056264	040510	020124	051511
14977	056272	044440	020116	043442
14978	056300	047517	021104	000
14979				
14980	056305	101	020116	044122
14981	056312	041440	042514	051101
14982	056320	024040	044122	051503
14983	056326	020062	044502	020124
14984	056334	032443	020051	040527
14985	056342	020123	044507	042526
14986	056350	027116	040440	033061
14987	056356	024040	044122	051503
14988	056364	020061	044502	020124
14989	056372	034043	020051	040527
14990	056400	020123	042523	006524
14991	056406	012		
14992	056407	101	046114	055040
14993	056414	051105	051517	053440
14994	056422	051105	020105	051127
14995	056430	052111	042524	020116
14996	056436	047111	051040	041110
14997	056444	042501	020056	044124
14998	056452	047105	052040	042510
14999	056460	043040	046117	047514
15000	056466	044527	043516	051040
15001	056474	043505	051511	042524
15002	056502	020122	044504	020104
15003	056510	047516	020124	040510
15004	056516	042526	053440	040510

EM54: .ASCII /AN RH CLEAR (RHCS2 BIT #5) WAS GIVEN. A16 (RHCS1 BIT #8) WAS SET/<15><1

.ASCIZ /ON READING THE FOLLOWING REGISTER IT DID NOT CONTAIN WHAT IS IN "GOOD"/

EM56: .ASCII /AN RH CLEAR (RHCS2 BIT #5) WAS GIVEN. A16 (RHCS1 BIT #8) WAS SET/<15><1

.ASCIZ /ALL ZEROS WERE WRITTEN IN RHBAE. THEN THE FOLLOWING REGISTER DID NOT HA

15005	056524	020124	051511	044440
15006	056532	020116	043442	047517
15007	056540	021104	000	
15008				
15009	056543	101	020116	044122
15010	056550	041440	042514	051101
15011	056556	024040	044122	051503
15012	056564	020062	044502	020124
15013	056572	032443	020051	040527
15014	056600	020123	044507	042526
15015	056606	027116	040440	033461
15016	056614	024040	044122	051503
15017	056622	020061	044502	020124
15018	056630	034443	020051	040527
15019	056636	020123	042523	006524
15020	056644	012		
15021	056645	117	020116	042522
15022	056652	042101	047111	020107
15023	056660	044124	020105	047506
15024	056666	046114	053517	047111
15025	056674	020107	042522	044507
15026	056702	052123	051105	044440
15027	056710	020124	044504	020104
15028	056716	047516	020124	047503
15029	056724	052116	044501	020116
15030	056732	044127	052101	044440
15031	056740	020123	047111	021040
15032	056746	047507	042117	000042
15033				
15034	056754	047101	051040	020110
15035	056762	046103	040505	020122
15036	056770	051050	041510	031123
15037	056776	041040	052111	021440
15038	057004	024470	053440	051501
15039	057012	043440	053111	047105
15040	057020	020056	030501	020067
15041	057026	051050	041510	030523
15042	057034	041040	052111	021440
15043	057042	024471	053440	051501
15044	057050	051440	052105	005015
15045	057056	046101	020114	042532
15046	057064	047522	020123	042527
15047	057072	042522	053440	044522
15048	057100	052124	047105	044440
15049	057106	020116	044122	040502
15050	057114	020105	044124	047105
15051	057122	052040	042510	043040
15052	057130	046117	047514	044527
15053	057136	043516	051040	043505
15054	057144	051511	042524	020122
15055	057152	044504	020104	047516
15056	057160	020124	040510	042526
15057	057166	053440	040510	020124
15058	057174	051511	044440	020116
15059	057202	047507	042117	005015
15060	057210	000		

EM60: .ASCII /AN RH CLEAR (RHCS2 BIT #5) WAS GIVEN. A17 (RHCS1 BIT #9) WAS SET/<15><1

.ASCIZ /ON READING THE FOLLOWING REGISTER IT DID NOT CONTAIN WHAT IS IN "GOOD"/

EM62: .ASCII /AN RH CLEAR (RHCS2 BIT #8) WAS GIVEN. A17 (RHCS1 BIT #9) WAS SET/<15><1

.ASCIZ /ALL ZEROS WERE WRITTEN IN RHBAE THEN THE FOLLOWING REGISTER DID NOT HAV

15061						
15062	057211	101	020116	044122	EM64:	.ASCII /AN RH CLEAR (RHCS2-BIT #5) WAS GIVEN. IE (RHCS1 BIT #6) WAS SET/<15><12
15063	057216	041440	042514	051101		
15064	057224	024040	044122	051503		
15065	057232	026462	044502	020124		
15066	057240	032443	020051	040527		
15067	057246	020123	044507	042526		
15068	057254	027116	044440	020105		
15069	057262	051050	041510	030523		
15070	057270	041040	052111	021440		
15071	057276	024466	053440	051501		
15072	057304	051440	052105	005015		
15073	057312	047117	051040	040505	.ASCIZ	/ON READING THE FOLLOWING REGISTER IT DID NOT CONTAIN WHAT IS IN "GOOD"/
15074	057320	044504	043516	052040		
15075	057326	042510	043040	046117		
15076	057334	047514	044527	043516		
15077	057342	051040	043505	051511		
15078	057350	042524	020122	052111		
15079	057356	042040	042111	047040		
15080	057364	052117	041440	047117		
15081	057372	040524	047111	053440		
15082	057400	040510	020124	051511		
15083	057406	044440	020116	043442		
15084	057414	047517	021104	000		
15085						
15086	057421	101	020116	044122	EM66:	.ASCII /AN RH CLEAR (RHCS2-BIT #5) WAS GIVEN. IE (RHCS1 BIT #6) WAS SET/<15><12
15087	057426	041440	042514	051101		
15088	057434	024040	044122	051503		
15089	057442	026462	044502	020124		
15090	057450	032443	020051	040527		
15091	057456	020123	044507	042526		
15092	057464	027116	044440	020105		
15093	057472	051050	041510	030523		
15094	057500	041040	052111	021440		
15095	057506	024466	053440	051501		
15096	057514	051440	052105	005015		
15097	057522	046101	020114	042532	.ASCIZ	/ALL ZEROS WERE WRITTEN IN RHCS3 THEN THE FOLLOWING REGISTER DID NOT CON
15098	057530	047522	020123	042527		
15099	057536	042522	053440	044522		
15100	057544	052124	047105	044440		
15101	057552	020116	044122	051503		
15102	057560	020063	044124	047105		
15103	057566	052040	042510	043040		
15104	057574	046117	047514	044527		
15105	057602	043516	051040	043505		
15106	057610	051511	042524	020122		
15107	057616	044504	020104	047516		
15108	057624	020124	047503	052116		
15109	057632	044501	020116	044127		
15110	057640	052101	044440	020123		
15111	057646	047111	021040	047507		
15112	057654	042117	000042			
15113						
15114	057660	044122	041440	042514	EM70:	.ASCII /RH CLEAR WAS GIVEN. TWO SUCCESSIVE "GO" (RHCS1-BIT #0) WAS GIVEN/<15><1
15115	057666	051101	053440	051501		
15116	057674	043440	053111	047105		

15117	057702	020056	053524	020117
15118	057710	052523	041503	051505
15119	057716	044523	042526	021040
15120	057724	047507	020042	051050
15121	057732	041510	030523	041055
15122	057740	052111	021440	024460
15123	057746	053440	051501	043440
15124	057754	053111	047105	005015
15125	057762	044527	044124	052517
15126	057770	020124	040527	052111
15127	057776	047111	020107	047506
15128	060004	020122	044124	020105
15129	060012	044506	051522	020124
15130	060020	043442	021117	052040
15131	060026	020117	047503	050115
15132	060034	042514	042524	020056
15133	060042	044124	020105	047506
15134	060050	046114	053517	047111
15135	060056	020107	042522	044507
15136	060064	052123	051105	005015
15137	060072	044504	020104	047516
15138	060100	020124	047503	052116
15139	060106	044501	020116	044127
15140	060114	052101	044440	020123
15141	060122	047111	021040	047507
15142	060130	042117	000042	
15143				
15144	060134	044122	041440	042514
15145	060142	051101	053440	051501
15146	060150	043440	053111	047105
15147	060156	040440	031040	053440
15148	060164	051117	020104	051127
15149	060172	052111	020105	040527
15150	060200	020123	047504	042516
15151	060206	043040	047522	020115
15152	060214	020101	047514	040503
15153	060222	044524	047117	005015
15154	060230	040524	042507	020104
15155	060236	051127	051106	046517
15156	060244	040440	042116	053440
15157	060252	052111	020110	040502
15158	060260	020111	051050	041510
15159	060266	031123	041040	052111
15160	060274	021440	024463	051440
15161	060302	052105	040440	020124
15162	060310	044124	020105	047105
15163	060316	006504	012	
15164	060321	117	020106	044124
15165	060326	020105	051127	052111
15166	060334	020105	044124	020105
15167	060342	047506	046114	053517
15168	060350	047111	020107	042522
15169	060356	044507	052123	051105
15170	060364	042040	042111	047040
15171	060372	052117	041440	047117
15172	060400	040524	047111	053440

.ASCII /WITHOUT WAITING FOR THE FIRST "GO" TO COMPLETE. THE FOLLOWING REGISTER/

.ASCIZ /DID NOT CONTAIN WHAT IS IN "GOOD"/

EM76: .ASCII /RH CLEAR WAS GIVEN A 2 WORD WRITE WAS DONE FROM A LOCATION/<15><12>

.ASCII /TAGED WRFROM AND WITH BAI (RHCS2 BIT #3) SET AT THE END/<15><12>

.ASCIZ /OF THE WRITE THE FOLLOWING REGISTER DID NOT CONTAIN WHAT IS IN "GOOD"/

15173	060406	040510	020124	051511
15174	060414	044440	020116	043442
15175	060422	047517	021104	000
15176				
15177	060427	122	020110	046103
15178	060434	040505	020122	040527
15179	060442	020123	044507	042526
15180	060450	020116	047101	044440
15181	060456	041520	020113	044502
15182	060464	020124	044123	053517
15183	060472	020116	047111	021040
15184	060500	050111	045503	020042
15185	060506	040527	020123	042523
15186	060514	006524	012	
15187	060517	132	051105	051517
15188	060524	053440	051105	020105
15189	060532	047515	042526	020104
15190	060540	047111	047524	051040
15191	060546	042110	027102	047440
15192	060554	020116	042522	042101
15193	060562	047111	020107	044122
15194	060570	041104	052040	042510
15195	060576	043040	046117	047514
15196	060604	044527	043516	005015
15197	060612	042522	044507	052123
15198	060620	051105	042040	042111
15199	060626	047040	052117	041440
15200	060634	047117	040524	047111
15201	060642	053440	040510	020124
15202	060650	051511	044440	020116
15203	060656	043442	047517	021104
15204	060664	000		
15205				
15206	060665	122	020110	046103
15207	060672	040505	020122	040527
15208	060700	020123	044507	042526
15209	060706	020116	047101	044440
15210	060714	041520	020113	044502
15211	060722	020124	044123	053517
15212	060730	020116	047111	021040
15213	060736	050111	045503	020042
15214	060744	040527	020123	042523
15215	060752	006524	012	
15216	060755	132	051105	051517
15217	060762	053440	051105	020105
15218	060770	047515	042526	020104
15219	060776	047111	047524	051040
15220	061004	042110	027102	051040
15221	061012	042110	020102	040527
15222	061020	020123	042522	042101
15223	061026	020056	047101	051040
15224	061034	020110	046103	040505
15225	061042	020122	040527	020123
15226	061050	044507	042526	006516
15227	061056	012		
15228	061057	124	042510	043040

EM104: .ASCII /RH CLEAR WAS GIVEN AN IPCK BIT SHOWN IN "IPCK" WAS SET/<15><12>

.ASCII /ZEROS WERE MOVED INTO RHDB. ON READING RHDB THE FOLLOWING/<15><12>

.ASCIZ /REGISTER DID NOT CONTAIN WHAT IS IN "GOOD"/

EM106: .ASCII /RH CLEAR WAS GIVEN AN IPCK BIT SHOWN IN "IPCK" WAS SET/<15><12>

.ASCII /ZEROS WERE MOVED INTO RHDB. RHDB WAS READ. AN RH CLEAR WAS GIVEN/<15><1

.ASCIZ /THE FOLLOWING REGISTER DID NOT CONTAIN WHAT IS GIVEN IN "GCOD"/



15229	061064	046117	047514	044527
15230	061072	043516	051040	043505
15231	061100	051511	042524	020122
15232	061106	044504	020104	047516
15233	061114	020124	047503	052116
15234	061122	044501	020116	044127
15235	061130	052101	044440	020123
15236	061136	044507	042526	020116
15237	061144	047111	021040	047507
15238	061152	042117	000042	
15239				
15240	061156	044122	041440	042514
15241	061164	051101	053440	051501
15242	061172	043440	053111	047105
15243	061200	020056	040440	053440
15244	061206	044522	042524	041440
15245	061214	042510	045503	005015
15246	061222	040527	020123	047504
15247	061230	042516	020056	044124
15248	061236	020105	047506	046114
15249	061244	053517	047111	020107
15250	061252	042522	044507	052123
15251	061260	051105	042040	042111
15252	061266	047040	052117	041440
15253	061274	047117	040524	047111
15254	061302	053440	040510	020124
15255	061310	051511	044440	020116
15256	061316	043442	047517	021104
15257	061324	000		
15258				
15259	061325	122	020110	046103
15260	061332	040505	020122	040527
15261	061340	020123	044507	042526
15262	061346	027116	020040	020101
15263	061354	051127	052111	020105
15264	061362	044103	041505	006513
15265	061370	012		
15266	061371	127	051501	042040
15267	061376	047117	020105	044124
15268	061404	047105	040440	047516
15269	061412	044124	051105	051040
15270	061420	020110	046103	040505
15271	061426	020122	040527	020123
15272	061434	044507	042526	027116
15273	061442	052040	042510	043040
15274	061450	046117	047514	044527
15275	061456	043516	051040	043505
15276	061464	051511	042524	006522
15277	061472	012		
15278	061473	104	042111	047040
15279	061500	052117	041440	047117
15280	061506	040524	047111	053440
15281	061514	040510	020124	051511
15282	061522	044440	020116	043442
15283	061530	047517	021104	000
15284				

EM114: .ASCII /RH CLEAR WAS GIVEN. A WRITE CHECK/<15><12>

.ASCIZ /WAS DONE. THE FOLLOWING REGISTER DID NOT CONTAIN WHAT IS IN "GOOD"/

EM116: .ASCII /RH CLEAR WAS GIVEN. A WRITE CHECK/<15><12>

.ASCII /WAS DONE THEN ANOTHER RH CLEAR WAS GIVEN. THE FOLLOWING REGISTER/<15><1

.ASCIZ /DID NOT CONTAIN WHAT IS IN "GOOD"/

15285	061535	117	020116	020101	EM124: .ASCII /ON A SILO TEST TO TEST DBL RHCS3 BIT #10 THE FOLLOWING/<15><12>
15286	061542	044523	047514	052040	
15287	061550	051505	020124	047524	
15288	061556	052040	051505	020124	
15289	061564	041104	020114	044122	
15290	061572	051503	020063	044502	
15291	061600	020124	030443	020060	
15292	061606	044124	020105	047506	
15293	061614	046114	053517	047111	
15294	061622	006507	012		
15295	061625	122	043505	051511	.ASCIZ /REGISTER DID NOT CONTAIN WHAT IS IN "GOOD"/
15296	061632	042524	020122	044504	
15297	061640	020104	047516	020124	
15298	061646	047503	052116	044501	
15299	061654	020116	044127	052101	
15300	061662	044440	020123	047111	
15301	061670	021040	047507	042117	
15302	061676	000042			
15303	061700	042502	047506	042522	EM132: .ASCII /BEFORE DATA TRANSFER COMMAND WAS TO/<15><12>
15304	061706	042040	052101	020101	
15305	061714	051124	047101	043123	
15306	061722	051105	041440	046517	
15307	061730	040515	042116	053440	
15308	061736	051501	052040	006517	
15309	061744	012			
15310	061745	102	020105	044507	.ASCII /BE GIVEN THE RP DRIVE STATUS REGISTER DID/<15><12>
15311	061752	042526	020116	044124	
15312	061760	020105	050122	042040	
15313	061766	044522	042526	051440	
15314	061774	040524	052524	020123	
15315	062002	042522	044507	052123	
15316	062010	051105	042040	042111	
15317	062016	005015			
15318	062020	047516	020124	047503	.ASCIZ /NOT CONTAIN WHAT IS IN "GOOD"/
15319	062026	052116	044501	020116	
15320	062034	044127	052101	044440	
15321	062042	020123	047111	021040	
15322	062050	047507	042117	000042	
15323	062056	042502	047506	042522	EM133: .ASCII /BEFORE DATA TRANSFER COMMAND WAS TO/<15><12>
15324	062064	042040	052101	020101	
15325	062072	051124	047101	043123	
15326	062100	051105	041440	046517	
15327	062106	040515	042116	053440	
15328	062114	051501	052040	006517	
15329	062122	012			
15330	062123	102	020105	044507	.ASCII /BE GIVEN THE RS DRIVE STATUS REGISTER DID/<15><12>
15331	062130	042526	020116	044124	
15332	062136	020105	051522	042040	
15333	062144	044522	042526	051440	
15334	062152	040524	052524	020123	
15335	062160	042522	044507	052123	
15336	062166	051105	042040	042111	
15337	062174	005015			
15338	062176	047516	020124	047503	.ASCIZ /NOT CONTAIN WHAT IS IN "GOOD"/
15339	062204	052116	044501	020116	
15340	062212	044127	052101	044440	

15341	062220	020123	047111	021040	
15342	062226	047507	042117	000042	
15343	062234	042502	047506	042522	EM134: .ASCII /BEFORE DATA TRANSFER COMMAND WAS TO BE GIVEN/<15><12>
15344	062242	042040	052101	020101	
15345	062250	051124	047101	043123	
15346	062256	051105	041440	046517	
15347	062264	040515	042116	053440	
15348	062272	051501	052040	020117	
15349	062300	042502	043440	053111	
15350	062306	047105	005015		
15351	062312	044124	020105	040515	.ASCII /THE MAG TAPE DRIVE STATUS REGISTER DID NOT/<15><12>
15352	062320	020107	040524	042520	
15353	062326	042040	044522	042526	
15354	062334	051440	040524	052524	
15355	062342	020123	042522	044507	
15356	062350	052123	051105	042040	
15357	062356	042111	047040	052117	
15358	062364	005015			
15359	062366	047503	052116	044501	.ASCIZ /CONTAIN WHAT IS IN RHDS1-GOOD/
15360	062374	020116	044127	052101	
15361	062402	044440	020123	047111	
15362	062410	051040	042110	030523	
15363	062416	043455	047517	000104	
15364	062424	040527	020123	040527	EM135: .ASCII /WAS WAITING FOR A BIT TO SET IN A REGISTER/<15><12>
15365	062432	052111	047111	020107	
15366	062440	047506	020122	020101	
15367	062446	044502	020124	047524	
15368	062454	051440	052105	044440	
15369	062462	020116	020101	042522	
15370	062470	044507	052123	051105	
15371	062476	005015			
15372	062500	044502	020124	047111	.ASCIZ /BIT IN QUESTION IS IN "BIT WAITED"/<15><12>
15373	062506	050440	042525	052123	
15374	062514	047511	020116	051511	
15375	062522	044440	020116	041042	
15376	062530	052111	053440	044501	
15377	062536	042524	021104	005015	
15378	062544	000			
15379	062545	122	043505	051511	.ASCIZ /REGISTER ADDRESS IN QUESTION IS IN "REG. ADDR"/
15380	062552	042524	020122	042101	
15381	062560	051104	051505	020123	
15382	062566	047111	050440	042525	
15383	062574	052123	047511	020116	
15384	062602	051511	044440	020116	
15385	062610	051042	043505	020056	
15386	062616	042101	051104	000042	
15387	062624	040527	020123	040527	EM136: .ASCII /WAS WAITING FOR A BIT TO RESET IN A REGISTER/<15><12>
15388	062632	052111	047111	020107	
15389	062640	047506	020122	020101	
15390	062646	044502	020124	047524	
15391	062654	051040	051505	052105	
15392	062662	044440	020116	020101	
15393	062670	042522	044507	052123	
15394	062676	051105	005015		
15395	062702	044502	020124	047111	.ASCIZ /BIT IN QUESTION IS IN "BIT WAITED"/<15><12>
15396	062710	050440	042525	052123	

15397	062716	047511	020116	051511	
15398	062724	044440	020116	041042	
15399	062732	052111	053440	044501	
15400	062740	042524	021104	005015	
15401	062746	000			
15402	062747	122	043505	051511	.ASCIZ /REGISTER ADDRESS IN QUESTION IS IN "REG. ADDR"/
15403	062754	042524	020122	042101	
15404	062762	051104	051505	020123	
15405	062770	047111	050440	042525	
15406	062776	052123	047511	020116	
15407	063004	051511	044440	020116	
15408	063012	051042	043505	020056	
15409	063020	042101	051104	000042	
15410	063026	047117	053440	044522	EM137: .ASCII /ON WRITING AND READING THE REGISTER IN "REG. ADDR"/<15><12>
15411	063034	044524	043516	040440	
15412	063042	042116	051040	040505	
15413	063050	044504	043516	052040	
15414	063056	042510	051040	043505	
15415	063064	051511	042524	020122	
15416	063072	047111	021040	042522	
15417	063100	027107	040440	042104	
15418	063106	021122	005015		
15419	063112	052111	042040	042111	.ASCIZ /IT DID NOT CONTAIN EXPECTED VALUE/
15420	063120	047040	052117	041440	
15421	063126	047117	040524	047111	
15422	063134	042440	050130	041505	
15423	063142	042524	020104	040526	
15424	063150	052514	000105		
15425	063154	044122	041440	042514	EM140: .ASCII /RH CLEAR WAS GIVEN AN IPCK BIT SHOWN IN "IPCK" WAS SET/<15><12>
15426	063162	051101	053440	051501	
15427	063170	043440	053111	047105	
15428	063176	040440	020116	050111	
15429	063204	045503	041040	052111	
15430	063212	051440	047510	047127	
15431	063220	044440	020116	044442	
15432	063226	041520	021113	053440	
15433	063234	051511	051440	052105	
15434	063242	005015			
15435	063244	042532	047522	020123	.ASCII /ZEROS WERE MOVED INTO RHDB. ON READING THE FOLLOWING/<15><12>
15436	063252	042527	042522	046440	
15437	063260	053117	042105	044440	
15438	063266	052116	020117	044122	
15439	063274	041104	020056	047117	
15440	063302	051040	040505	044504	
15441	063310	043516	020040	044124	
15442	063316	020105	047506	046114	
15443	063324	053517	047111	006507	
15444	063332	012			
15445	063333	122	043505	051511	.ASCIZ /REGISTER DID NOT CONTAIN WHAT IS IN "GOOD"/
15446	063340	042524	020122	044504	
15447	063346	020104	047516	020124	
15448	063354	047503	052116	044501	
15449	063362	020116	044127	052101	
15450	063370	044440	020123	047111	
15451	063376	021040	047507	042117	
15452	063404	000042			

15453	063406	043101	042524	020122	EM141: .ASCII /AFTER CLEARING THE RH AND WRITING TWO WORD INTO RHDB AND/<15><12>
15454	063414	046103	040505	044522	
15455	063422	043516	052040	042510	
15456	063430	051040	020110	047101	
15457	063436	020104	051127	052111	
15458	063444	047111	020107	053524	
15459	063452	020117	047527	042122	
15460	063460	044440	052116	020117	
15461	063466	044122	041104	040440	
15462	063474	042116	005015		
15463	063500	042522	042101	047111	.ASCIZ /READING IT BACK TWICE, CAUSED RHCS1 TO HAVE WRONG VALUE GIVEN IN "BAD"/
15464	063506	020107	052111	041040	
15465	063514	041501	020113	053524	
15466	063522	041511	026105	041440	
15467	063530	052501	042523	020104	
15468	063536	044122	051503	020061	
15469	063544	047524	044040	053101	
15470	063552	020105	051127	047117	
15471	063560	020107	040526	052514	
15472	063566	020105	044507	042526	
15473	063574	020116	047111	021040	
15474	063602	040502	021104	000	
15475	063607	101	052106	051105	EM142: .ASCII /AFTER CLEARING THE RH AND WRITING TWO WORD INTO RHDB AND/<15><12>
15476	063614	041440	042514	051101	
15477	063622	047111	020107	044124	
15478	063630	020105	044122	040440	
15479	063636	042116	053440	044522	
15480	063644	044524	043516	052040	
15481	063652	047527	053440	051117	
15482	063660	020104	047111	047524	
15483	063666	051040	042110	020102	
15484	063674	047101	006504	012	
15485	063701	122	040505	044504	.ASCIZ /READING IT BACK TWICE, CAUSED RHCS3 TO HAVE WRONG VALUE GIVEN IN "BAD"/
15486	063706	043516	044440	020124	
15487	063714	040502	045503	052040	
15488	063722	044527	042503	020054	
15489	063730	040503	051525	042105	
15490	063736	051040	041510	031523	
15491	063744	052040	020117	040510	
15492	063752	042526	053440	047522	
15493	063760	043516	053040	046101	
15494	063766	042525	043440	053111	
15495	063774	047105	044440	020116	
15496	064002	041042	042101	000042	
15497	064010	043101	042524	020122	EM143: .ASCII /AFTER CLEARING THE RH AND WRITING TWO WORD INTO RHDB AND/<15><12>
15498	064016	046103	040505	044522	
15499	064024	043516	052040	042510	
15500	064032	051040	020110	047101	
15501	064040	020104	051127	052111	
15502	064046	047111	020107	053524	
15503	064054	020117	047527	042122	
15504	064062	044440	052116	020117	
15505	064070	044122	041104	040440	
15506	064076	042116	005015		
15507	064102	042522	042101	047111	.ASCIZ /READING IT BACK TWICE, CAUSED RHBA TO HAVE WRONG VALUE GIVEN IN "BAD"/
15508	064110	020107	052111	041040	

15509	064116	041501	020113	053524	
15510	064124	041511	026105	041440	
15511	064132	052501	042523	020104	
15512	064140	044122	040502	052040	
15513	064146	020117	040510	042526	
15514	064154	053440	047522	043516	
15515	064162	053040	046101	042525	
15516	064170	043440	053111	047105	
15517	064176	044440	020116	041042	
15518	064204	042101	000042		
15519	064210	043101	042524	020122	EM144: .ASCII /AFTER CLEARING THE RH AND WRITING TWO WORD INTO RHDB AND/<15><12>
15520	064216	046103	040505	044522	
15521	064224	043516	052040	042510	
15522	064232	051040	020110	047101	
15523	064240	020104	051127	052111	
15524	064246	047111	020107	053524	
15525	064254	020117	047527	042122	
15526	064262	044440	052116	020117	
15527	064270	044122	041104	040440	
15528	064276	042116	005015		
15529	064302	042522	042101	047111	.ASCIZ /READING IT BACK TWICE, CAUSED RHBAE TO HAVE WRONG VALUE GIVEN IN "BAD"/
15530	064310	020107	052111	041040	
15531	064316	041501	020113	053524	
15532	064324	041511	026105	041440	
15533	064332	052501	042523	020104	
15534	064340	044122	040502	020105	
15535	064346	047524	044040	053101	
15536	064354	020105	051127	047117	
15537	064362	020107	040526	052514	
15538	064370	020105	044507	042526	
15539	064376	020116	047111	021040	
15540	064404	040502	021104	000	
15541	064411	101	052106	051105	EM145: .ASCII /AFTER CLEARING THE RH AND WRITING TWO WORD INTO RHDB AND/<15><12>
15542	064416	041440	042514	051101	
15543	064424	047111	020107	044124	
15544	064432	020105	044122	040440	
15545	064440	042116	053440	044522	
15546	064446	044524	043516	052040	
15547	064454	047527	053440	051117	
15548	064462	020104	047111	047524	
15549	064470	051040	042110	020102	
15550	064476	047101	006504	012	
15551	064503	122	040505	044504	.ASCIZ /READING IT BACK TWICE, CAUSED RHWC TO HAVE WRONG VALUE GIVEN IN "BAD"/
15552	064510	043516	044440	020124	
15553	064516	040502	045503	052040	
15554	064524	044527	042503	020054	
15555	064532	040503	051525	042105	
15556	064540	051040	053510	020103	
15557	064546	047524	044040	053101	
15558	064554	020105	051127	047117	
15559	064562	020107	040526	052514	
15560	064570	020105	044507	042526	
15561	064576	020116	047111	021040	
15562	064604	040502	021104	000	
15563					
15564	064611	101	042040	053105	EM146: .ASCII /A DEVICE BASE ADDRESS DID NOT TIME OUT BUT/<15><12>

15565	064616	041511	020105	040502				
15566	064624	042523	040440	042104				
15567	064632	042522	051523	042040				
15568	064640	042111	047040	052117				
15569	064646	052040	046511	020105				
15570	064654	052517	020124	052502				
15571	064662	006524	012					
15572	064665	124	042510	041440		.ASCII	/THE CORRESPONDING VECTOR DID TIME OUT/	<<15><12>
15573	064672	051117	042522	050123				
15574	064700	047117	044504	043516				
15575	064706	053040	041505	047524				
15576	064714	020122	044504	020104				
15577	064722	044524	042515	047440				
15578	064730	052125	006456	012				
15579	064735	110	052111	041440		.ASCIZ	/HIT CONTINUE TO REPEAT VECTOR TIMEOUT/	
15580	064742	047117	044524	052516				
15581	064750	020105	047524	051040				
15582	064756	050105	040505	020124				
15583	064764	042526	052103	051117				
15584	064772	052040	046511	047505				
15585	065000	052125	000					
15586	065003	101	042040	053105	EM147:	.ASCII	/A DEVICE BASE ADDRESS DID NOT TIME OUT/	<<15><12>
15587	065010	041511	020105	040502				
15588	065016	042523	040440	042104				
15589	065024	042522	051523	042040				
15590	065032	042111	047040	052117				
15591	065040	052040	046511	020105				
15592	065046	052517	006524	012				
15593	065053	102	052125	047040		.ASCIZ	/BUT NO UNIT NUMBERS HAD APPROPRIATE DRIVE TYPES/	
15594	065060	020117	047125	052111				
15595	065066	047040	046525	042502				
15596	065074	051522	044040	042101				
15597	065102	040440	050120	047522				
15598	065110	051120	040511	042524				
15599	065116	042040	044522	042526				
15600	065124	052040	050131	051505				
15601	065132	000						
15602								
15603								
15604	065133	120	004503	044122	DH1:	.ASCII	/PC RMDT/	
15605	065140	052104						
15606								
15607	065142	041520	052411	044516	DH5:	.ASCII	/PC UNIBUS ADDRESS ON WHICH TIME OUT OCCURRED/	
15608	065150	052502	020123	042101				
15609	065156	051104	051505	020123				
15610	065164	047117	057440	044510				
15611	065172	044103	052040	046511				
15612	065200	020105	052517	020124				
15613	065206	041517	052503	051122				
15614	065214	042105						
15615								
15616	065216	041520	052011	051505	DH6:	.ASCII	/PC TEST NO RHCS2 RHCS2/	<<15><12>
15617	065224	020124	047516	051040				
15618	065232	041510	031123	051011				
15619	065240	041510	031123	005015				
15620	065246	004411	047507	042117		.ASCIZ	/	GOOD BAD/

15621	065254	041011	042101	000					
15622									
15623	065261	120	004503	042524	DH7:	.ASCII	/PC	TEST	RHDB RHDB/<15><12>
15624	065266	052123	051011	042110					
15625	065274	004502	044122	041104					
15626	065302	005015							
15627	065304	047011	004517	047507		.ASCIZ	/	NO	GOOD BAD/
15628	065312	042117	041011	042101					
15629	065320	000							
15630									
15631	065321	120	004503	042524	DH11:	.ASCII	/PC	TEST	RHCS1 RHCS1/<15><12>
15632	065326	052123	051011	041510					
15633	065334	030523	051011	041510					
15634	065342	030523	005015						
15635	065346	047011	004517	047507		.ASCIZ	/	NO	GOOD BAD/
15636	065354	042117	041011	042101					
15637	065362	000							
15638									
15639	065363	120	004503	042524	DH12:	.ASCII	/PC	TEST	RHCS3 RHCS3/<15><12>
15640	065370	052123	051011	041510					
15641	065376	031523	051011	041510					
15642	065404	031523	005015						
15643	065410	047011	004517	047507		.ASCIZ	/	NO	GOOD BAD/
15644	065416	042117	041011	042101					
15645	065424	000							
15646									
15647	065425	120	004503	042524	DH13:	.ASCII	/PC	TEST	RHBA RHBA/<15><12>
15648	065432	052123	051011	041110					
15649	065440	004501	044122	040502					
15650	065446	005015							
15651	065450	047011	004517	047507		.ASCIZ	/	NO	GOOD BAD/
15652	065456	042117	041011	042101					
15653	065464	000							
15654									
15655	065465	120	004503	042524	DH14:	.ASCII	/PC	TEST	RHBAE RHBAE/<15><12>
15656	065472	052123	051011	041110					
15657	065500	042501	051011	041110					
15658	065506	042501	005015						
15659	065512	047011	004517	047507		.ASCIZ	/	NO	GOOD BAD/
15660	065520	042117	041011	042101					
15661	065526	000							
15662									
15663	065527	120	004503	042524	DH15:	.ASCII	/PC	TEST	RHWC RHWC/<15><12>
15664	065534	052123	051011	053510					
15665	065542	004503	044122	041527					
15666	065550	005015							
15667	065552	047011	004517	047507		.ASCIZ	/	NO	GOOD BAD/
15668	065560	042117	041011	042101					
15669	065566	000							
15670									
15671	065567	120	004503	042524	DH24:	.ASCII	/PC	TEST	WORD RHDB RHDB/<15><12>
15672	065574	052123	053411	051117					
15673	065602	004504	044122	041104					
15674	065610	051011	042110	006502					
15675	065616	012							
15676	065617	011	047516	047011		.ASCIZ	/	NO	NO GOOD BAD/



K07

MAINDEC-11-DERHAC-A MACY11 27(732) 09-SEP-76 07:55 PAGE 295  
 DERHAC.P11 POWER DOWN AND UP ROUTINES

15677	065624	004517	047507	042117								
15678	065632	041011	042101	000								
15679												
15680	065637	120	004503	042524	DH33:	.ASCII	/PC	TEST	RHER1	RHER1/<15><12>		
15681	065644	052123	051011	042510								
15682	065652	030522	051011	042510								
15683	065660	030522	005015									
15684	065664	047011	004517	047507		.ASCIZ	/	NO	GOOD	BAD/		
15685	065672	042117	041011	042101								
15686	065700	000										
15687												
15688	065701	120	004503	042524	DH45:	.ASCII	/PC	TEST	RHDA	RHDA/<15><12>		
15689	065706	052123	051011	042110								
15690	065714	004501	044122	040504								
15691	065722	005015										
15692	065724	047011	004517	047507		.ASCIZ	/	NO	GOOD	BAD/		
15693	065732	042117	041011	042101								
15694	065740	000										
15695												
15696	065741	120	004503	042524	DH104:	.ASCII	/PC	TEST	IPCK	RHCS3	RHCS3/<15><12>	
15697	065746	052123	044411	041520								
15698	065754	004513	044122	051503								
15699	065762	004463	044122	051503								
15700	065770	006463	012									
15701	065773	011	047516	004411		.ASCIZ	/	NO	GOOD	BAD/		
15702	066000	047507	042117	041011								
15703	066006	042101	000									
15704												
15705	066011	120	004503	042524	DH105:	.ASCII	/PC	TEST	IPCK	RHCS2	RHCS2/<15><12>	
15706	066016	052123	044411	041520								
15707	066024	004513	044122	051503								
15708	066032	004462	044122	051503								
15709	066040	006462	012									
15710	066043	011	047516	004411		.ASCIZ	/	NO	GOOD	BAD/		
15711	066050	047507	042117	041011								
15712	066056	042101	000									
15713												
15714	066061	120	004503	042524	DH106:	.ASCII	/PC	TEST	IPCK	RHCS1	RHCS1/<15><12>	
15715	066066	052123	044411	041520								
15716	066074	004513	044122	051503								
15717	066102	004461	044122	051503								
15718	066110	006461	012									
15719	066113	011	047516	004411		.ASCIZ	/	NO	GOOD	BAD/		
15720	066120	047507	042117	041011								
15721	066126	042101	000									
15722												
15723	066131	120	004503	042524	DH111:	.ASCII	/PC	TEST	IPCK	RHBA	RHBA/<15><12>	
15724	066136	052123	044411	041520								
15725	066144	004513	044122	040502								
15726	066152	051011	041110	006501								
15727	066160	012										
15728	066161	011	047516	004411		.ASCIZ	/	NO	GOOD	BAD/		
15729	066166	047507	042117	041011								
15730	066174	042101	000									
15731												
15732	066177	120	004503	042524	DH112:	.ASCII	/PC	TEST	IPCK	RHBAE	RHBAE/<15><12>	

15733	066204	052123	044411	041520						
15734	066212	004513	044122	040502						
15735	066220	004505	044122	040502						
15736	066226	006505	012							
15737	066231	011	047516	004411	.ASCIZ /	NO	GOOD	BAD/		
15738	066236	047507	042117	041011						
15739	066244	042101	000							
15740										
15741	066247	120	004503	042524	DH113: .ASCII /PC	TEST	IPCK	RHWC	RHWC/<15><12>	
15742	066254	052123	044411	041520						
15743	066262	004513	044122	041527						
15744	066270	051011	053510	006503						
15745	066276	012								
15746	066277	011	047516	004411	.ASCIZ /	NO	GOOD	BAD/		
15747	066304	047507	042117	041011						
15748	066312	042101	000							
15749	066315	120	004505	042524	DH132: .ASCII /PE	TEST	RHDS1	RHDS1/<15><12>		
15750	066322	052123	051011	042110						
15751	066330	030523	051011	042110						
15752	066336	030523	005015							
15753	066342	047011	004517	047507	.ASCIZ /	NO	GOOD	BAD/		
15754	066350	042117	041040	042101						
15755	066356	000								
15756	066357	120	004503	042524	DH135: .ASCII /PC	TEST	PC OF	BIT WAIT	REG/<15><12>	
15757	066364	052123	050011	020103						
15758	066372	043117	041011	052111						
15759	066400	053440	044501	020124						
15760	066406	051040	043505	005015						
15761	066414	047011	004517	040527	.ASCIZ /	NO	WAT	FOR	ADDR/	
15762	066422	004524	047506	004522						
15763	066430	020040	042101	051104						
15764	066436	000								
15765	066437	120	004503	042524	DH137: .ASCII /PC	TEST	REG.	GOOD	BAD/<15><12>	
15766	066444	052123	051011	043505						
15767	066452	004456	047507	042117						
15768	066460	041011	042101	005015						
15769	066466	047011	004517	042101	.ASCIZ /	NO	ADDR	DATA	DATA/	
15770	066474	051104	042011	052101						
15771	066502	004501	040504	040524						
15772	066510	000								
15773										
15774	066511	120	020103	020040	DH146: .ASCIZ /PC	BASE	VECTOR/			
15775	066516	020040	041040	051501						
15776	066524	020105	020040	053040						
15777	066532	041505	047524	000122						
15778	066540	041520	020040	020040	DH147: .ASCIZ /PC	BASE	ADDRESS/			
15779	066546	020040	040502	042523						
15780	066554	040440	042104	042522						
15781	066562	051523	000							
15782										
15783										
15784		066566			.EVEN					
15785										
15786	066566	001116	001126	000000	DT1: .WORD	\$ERRPC, \$BDDAT, 0				
15787	066574	001116	004046	004050	DT5: .WORD	\$ERRPC, \$RCSI, \$PCSI, \$TMCSI, \$MIXCSI, 0				
15788	066602	004052	004054	000000						

15789	066610	001116	004656	001124	DT6:	.WORD	\$ERRPC, TSTNM, \$GDDAT, \$BDDAT, 0
15790	066616	001126	000000				
15791	066622	001116	004656	004664	DT24:	.WORD	\$ERRPC, TSTNM, \$ILONM, \$GDDAT, \$BDDAT, 0
15792	066630	001124	001126	000000			
15793	066636	001116	004656	004662	DT104:	.WORD	\$ERRPC, TSTNM, IP, \$GDDAT, \$BDDAT, 0
15794	066644	001124	001126	000000			
15795	066652	001116	004656	004662	DT105:	.WORD	\$ERRPC, TSTNM, IP, \$GDDAT, \$BDDAT, 0
15796	066660	001124	001126	000000			
15797	066666	001116	004656	004662	DT106:	.WORD	\$ERRPC, TSTNM, IP, \$GDDAT, \$BDDAT, 0
15798	066674	001124	001126	000000			
15799	066702	001116	004656	004672	DT135:	.WORD	\$ERRPC, TSTNM, WAITBT, WAITPC, WAITRE, 0
15800	066710	004666	004670	000000			
15801	066716	001116	004656	001122	DT137:	.WORD	\$ERRPC, TSTNM, \$BDADR, \$GDDAT, \$BDDAT, 0
15802	066724	001124	001126	000000			
15803	066732	001116	005324	005326	DT146:	.WORD	\$ERRPC, TESTDV, TESTVC, 0
15804	066740	000000					
15805	066742	001116	005324	000000	DT147:	.WORD	\$ERRPC, TESTDV, 0
15806							
15807							
15808							
15809	066750	000	000		DF1:	.BYTE	0,0
15810	066752	000	000	000	DF5:	.BYTE	0,0,0,0,0
15811	066755	000	000				
15812	066757	000	000	000	DF6:	.BYTE	0,0,0,0
15813	066762	000					
15814	066763	000	000	000	DF24:	.BYTE	0,0,0,0,0
15815	066766	000	000				
15816	066770	000	000	000	DF104:	.BYTE	0,0,0,0,0
15817	066773	000	000				
15818	066775	000	000	000	DF105:	.BYTE	0,0,0,0,0
15819	067000	000	000				
15820	067002	000	000	000	DF106:	.BYTE	0,0,0,0,0
15821	067005	000	000				
15822	067007	000	000	000	DF135:	.BYTE	0,0,0,0,0
15823	067012	000	000				
15824	067014	000	000	000	DF137:	.BYTE	0,0,0,0,0
15825	067017	000	000				
15826	067021	000	000	000	DF146:	.BYTE	0,0,0
15827	067024	000	000		DF147:	.BYTE	0,0
15828						.EVEN	
15829		000001				.END	

ACL = 000040  
 ACU = 100000  
 AOE = 001000  
 APE = 100000  
 AS = 000016  
 ATA = 100000  
 ATABLE = 004674  
 ATTENT = 005244  
 ATO = 000001  
 AT1 = 000002  
 AT2 = 000004  
 AT3 = 000010  
 AT4 = 000020  
 AT5 = 000040  
 AT6 = 000100  
 AT7 = 000200  
 A16 = 000400  
 A17 = 001000  
 BA = 000004  
 BAI = 000010  
 BASEAD = 004704  
 BASECH = 042254  
 BASEMA = 004712  
 BASEVC = 004714  
 BASINX = 004766  
 BASM = 004722  
 BASP = 004716  
 BASPAD = 004706  
 BAST = 004720  
 BASTAD = 004710  
 BEGIN = 005522  
 BEGIN2 = 005532  
 BFEVEN = 003000  
 BF000 = 003002  
 BIT0 = 000001  
 BIT00 = 000001  
 BIT01 = 000002  
 BIT02 = 000004  
 BIT03 = 000010  
 BIT04 = 000020  
 BIT05 = 000040  
 BIT06 = 000100  
 BIT07 = 000200  
 BIT08 = 000400  
 BIT09 = 001000  
 BIT1 = 000002  
 BIT10 = 002000  
 BIT11 = 004000  
 BIT12 = 010000  
 BIT13 = 020000  
 BIT14 = 040000  
 BIT15 = 100000  
 BIT2 = 000004

BIT3 = 000010  
 BIT4 = 000020  
 BITS = 000040  
 BIT6 = 000100  
 BIT7 = 000200  
 BIT8 = 000400  
 BIT9 = 001000  
 BOT = 000002  
 BPI8 = 001400  
 BPTVEC = 000014  
 BSGIVA = 004724  
 BSGIVV = 004744  
 BUFEW = 003000  
 BUFOV = 003002  
 CC = 000036  
 CHECK = 040354  
 CHECKC = 040412  
 CHECKT = 040376  
 CH7 = 010000  
 CLDISK = 040322  
 CLR = 000040  
 CLRBIT = 004654  
 CMNORP = 041572  
 CMNDRS = 041502  
 CMNDTM = 041710  
 COMAND = 004642  
 COMND = 004640  
 CR = 000015  
 CRLF = 000200  
 CSF = 000002  
 CSU = 000010  
 CS1 = 000000  
 CS2 = 000010  
 CA = 000006  
 DB = 000022  
 DBL = 002000  
 DC = 000034  
 DCK = 100000  
 DCL = 000100  
 DCLEAR = 004142  
 DDISP = 177570  
 DEN1 = 000400  
 DEN2 = 001000  
 DEN4 = 002000  
 DEVPNT = 005300  
 DE1 = 000040  
 DFF20 = 000002  
 DFF5 = 000001  
 DF1 = 066750  
 DF104 = 066770  
 DF105 = 066775  
 DF106 = 067002  
 DF135 = 067007

DF137 = 067014  
 DF146 = 067021  
 DF147 = 067024  
 DF24 = 066763  
 DF5 = 066752  
 DF6 = 066757  
 DH1 = 065133  
 DH104 = 065741  
 DH105 = 066011  
 DH106 = 066061  
 DH11 = 065321  
 DH111 = 066131  
 DH112 = 066177  
 DH113 = 066247  
 DH12 = 065363  
 DH13 = 065425  
 DH132 = 066315  
 DH135 = 066357  
 DH137 = 066437  
 DH14 = 065465  
 DH146 = 066511  
 DH147 = 066540  
 DH15 = 065527  
 DH24 = 065567  
 DH33 = 065637  
 DH45 = 065701  
 DH5 = 065142  
 DH6 = 065216  
 DH7 = 065261  
 DIGB = 000004  
 DISPLA = 001142  
 DISPRE = 000174  
 DLT = 100000  
 DL64 = 000020  
 DMD = 000001  
 DPEEW = 020000  
 DPEOW = 040000  
 DPR = 000400  
 DRY = 000200  
 DS = 000012  
 CSWR = 177570  
 DT = 000026  
 DTE = 010000  
 DTSY = 001000  
 DT1 = 066566  
 DT104 = 066636  
 DT105 = 066652  
 DT106 = 066666  
 DT135 = 066702  
 DT137 = 066716  
 DT146 = 066732  
 DT147 = 066742  
 DT24 = 066622

DT5 = 066574  
 DT6 = 066610  
 DVA = 004000  
 ECH = 000100  
 ECI = 004000  
 EC1 = 000044  
 EC2 = 000046  
 EMTVEC = 000030  
 EM1 = 046712  
 EM10 = 050404  
 EM104 = 060427  
 EM106 = 060665  
 EM11 = 050577  
 EM114 = 061156  
 EM116 = 061325  
 EM12 = 050772  
 EM124 = 061535  
 EM13 = 051165  
 EM132 = 061700  
 EM33 = 062056  
 EM134 = 062234  
 EM135 = 062424  
 EM136 = 062624  
 EM137 = 063026  
 EM14 = 051357  
 EM140 = 063154  
 EM141 = 063406  
 EM142 = 063607  
 EM143 = 064010  
 EM144 = 064210  
 EM145 = 064411  
 EM146 = 064611  
 EM147 = 065003  
 EM15 = 051552  
 EM16 = 051744  
 EM17 = 052137  
 EM2 = 047064  
 EM20 = 052331  
 EM21 = 052525  
 EM22 = 052725  
 EM23 = 053126  
 EM24 = 053123  
 EM25 = 053553  
 EM3 = 047266  
 EM33 = 054002  
 EM34 = 054116  
 EM35 = 054360  
 EM36 = 054556  
 EM37 = 055033  
 EM4 = 047422  
 EM45 = 055335  
 EM46 = 055573  
 EMS = 047721

EM54 = 056074  
 EM56 = 056305  
 EM6 = 050074  
 EM60 = 056543  
 EM62 = 056754  
 EM64 = 057211  
 EM66 = 057421  
 EM7 = 050212  
 EM70 = 057660  
 EM76 = 060134  
 EPAR = 000010  
 ERFLGS = 005240  
 ERR = 040000  
 ERRVEC = 000004  
 ER1 = 000014  
 ER2 = 000040  
 ER3 = 000042  
 EXT1 = 000001  
 EXT10 = 000010  
 EXT2 = 000002  
 EXT20 = 000020  
 EXT4 = 000004  
 EXT40 = 000040  
 FBAE = 005466  
 FBASEA = 005426  
 FC = 000006  
 FCS3 = 005476  
 FDEVIC = 005366  
 FDRIVR = 005456  
 FEN = 000200  
 FER = 000020  
 FIRST = 005242  
 FMT1 = 000020  
 FMT2 = 000040  
 FMT22 = 010000  
 FMT4 = 000100  
 FMT8 = 000200  
 FORBAE = 004776  
 FRHNM = 005436  
 FSLAVE = 005446  
 FTYPE = 005406  
 FUNITN = 005376  
 FUTABL = 004134  
 FVECTR = 005416  
 GIVE = 005006  
 GO = 000001  
 GRV = 000010  
 HCE = 000200  
 HCI = 002000  
 HCRC = 000400  
 HERADD = 005516  
 HT = 000011  
 IAE = 002000

IE	=	000100	NHS	=	002000	PR7	=	000340	RO	=	%000000	SW08	=	000400
ILF	=	000001	NML	=	000300	PS	=	177776	R1	=	%000001	SW09	=	001000
ILLEGL	=	004206	NMRHS	=	004764	PSEL	=	002000	R2	=	%000002	SW1	=	000002
ILR	=	000002	NOGO	=	004644	PSU	=	000001	R3	=	%000003	SW10	=	002000
IOTVEC	=	000020	NOPERA	=	004134	PSW	=	177776	R4	=	%000004	SW11	=	004000
IP	=	004662	NTYPNT	=	005362	PUTREG	=	041442	R5	=	%000005	SW12	=	010000
IPCK0	=	000001	OCYL	=	100000	PWRVEC	=	000024	R6	=	%000006	SW13	=	020000
IPCK1	=	000002	OF	=	000032	RDCHR	=	104406	R7	=	%000007	SW14	=	040000
IPCK2	=	000004	OFREV	=	000200	RDLIN	=	104407	SC	=	100000	SW15	=	100000
IPCK3	=	000010	OFSETC	=	004166	RDOCT	=	104410	SC1	=	000100	SW2	=	000004
IR	=	000100	OF100	=	000004	RDY	=	000200	SC10	=	001000	SW3	=	000010
IXE	=	004000	OF200	=	000010	READAT	=	004160	SC2	=	000200	SW4	=	000020
KIPAR0	=	172340	OF25	=	000001	READIN	=	004174	SC20	=	002000	SW5	=	000040
KIPAR1	=	172342	OF400	=	000020	RECALI	=	004140	SC4	=	000400	SW6	=	000100
KIPAR2	=	172344	OF50	=	000002	REFOR	=	004162	SEECOM	=	004164	SW7	=	000200
KIPAR3	=	172346	OF800	=	000040	REINTO	=	004424	SERCH	=	004146	SW8	=	000400
KIPAR4	=	172350	OPERSE	=	040654	RELEAS	=	004144	SETBIT	=	004652	SW9	=	001000
KIPAR5	=	172352	OPI	=	020000	RESVEC	=	000010	SETSIZ	=	006522	S1	=	000001
KIPAR6	=	172354	OR	=	000200	RETCL	=	004170	SILONM	=	004664	S2	=	000002
KIPAR7	=	172356	PAR	=	000010	REVRED	=	004200	SKI	=	040000	S4	=	000004
KIPDR0	=	172300	PARE70	=	000114	REVWRT	=	004204	SLAVE	=	004660	TBITVE	=	000014
KIPDR1	=	172302	PARITY	=	037746	REWIND	=	004176	SLVPR	=	002000	TC	=	000032
KIPDR2	=	172304	PAT	=	000020	RHAS	=	004076	SN	=	000030	TDF	=	000040
KIPDR3	=	172306	PC	=	%000007	RHBA	=	004064	SND1	=	006124	TESTAD	=	040652
KIPDR4	=	172310	PCJSR	=	040352	RHBAE	=	004130	SP	=	%000006	TESTDV	=	005324
KIPDR5	=	172312	PE	=	020000	RHCA	=	004114	SPACFD	=	004202	TESTVC	=	005326
KIPDR6	=	172314	PERMX	=	005264	RHCC	=	004100	SRO	=	177572	TFOUND	=	005322
KIPDR7	=	172316	PERMXV	=	005274	RHCCA	=	004116	SR1	=	177574	TIEOUT	=	037610
LA	=	000020	PERNUM	=	005276	RHCS1	=	004060	SR2	=	177576	TIMCNT	=	040456
LBT	=	002000	PERRP	=	005260	RHCS2	=	004070	SR3	=	172516	TKVEC	=	000060
LERADD	=	005514	PERRPV	=	005270	RHCS3	=	004132	SS1	=	007376	TMCS1	=	004052
LF	=	000012	PERRS	=	005256	RHDA	=	004066	SS2	=	006536	TMPSLV	=	005364
LOPCT	=	005000	PERRSV	=	005266	RHDB	=	004102	SS3	=	010506	TMP0	=	005250
LOPCT1	=	005002	PERTU	=	005262	RHDST	=	004066	SS4	=	010530	TMP1	=	005252
LTRY	=	005320	PERTUV	=	005272	RHDS1	=	004072	SS5	=	010546	TMP4	=	005254
MCLK	=	000002	PGE	=	002000	RHDT	=	004106	SS6	=	006612	TN	=	000067
MCPE	=	020000	PIP	=	020000	RHEC1	=	004124	SS7	=	006626	TOTALA	=	005246
MDPE	=	000400	PIRQ	=	177772	RHEC2	=	004126	SS8	=	010554	TPVEC	=	000064
MEMERR	=	005520	PIRQVE	=	000240	RHER1	=	004074	STACK	=	001000	TRAPVE	=	000034
MHS	=	001000	PKACK	=	004172	RHER2	=	004120	START	=	005536	TRE	=	040000
MINX	=	000004	PLU	=	020000	RHER3	=	004122	STKLMT	=	177774	TRK1	=	004000
MIXCS1	=	004054	POINTT	=	005360	RHFC	=	004066	ST200	=	005012	TRK10	=	040000
MMVEC	=	000250	PRE	=	000020	RHLA	=	004100	SWR	=	001140	TRK2	=	010000
MOL	=	010000	PRESEN	=	004056	RHMR	=	004104	SWREG	=	000176	TRK20	=	100000
MPE	=	000400	PRITEM	=	045100	RHOF	=	004112	SW0	=	000001	TRK4	=	020000
MR	=	000024	PROG	=	001000	RHSN	=	004110	SW00	=	000001	TRTVEC	=	000014
MRD	=	000020	PRO	=	000000	RHTC	=	004112	SW01	=	000002	TSBAE	=	005140
MSE	=	000020	PR1	=	000040	RHWC	=	004062	SW02	=	000004	TSBASE	=	005220
MSTCK	=	000010	PR2	=	000100	RMR	=	000004	SW03	=	000010	TSCOMD	=	005200
MWR	=	000040	PR3	=	000140	RPCS1	=	004050	SW04	=	000020	TSCS3	=	005160
MXF	=	001000	PR4	=	000200	RPVEC	=	002716	SW05	=	000040	TSDEVI	=	005034
NEO	=	010000	PR5	=	000240	RPVECT	=	043056	SW06	=	000100	TSINDX	=	005116
NEM	=	004000	PR6	=	000300	RSCS1	=	004046	SW07	=	000200	TSRHNO	=	005014

TSSLAV	005074	TST54	033504	WCE	= 040000	SERRPC	001116	\$RTNAD	037566
TSTNM	004656	TST55	033732	WCEEW	= 004000	\$ERRTB	001226	\$SAVR6	046700
TSTOTL	005114	TST56	034160	WCEOW	= 010000	\$ERTTY	045102	\$SCOPE	043120
TSTUNT	005506	TST57	034406	WCF	= 000040	\$ERTTL	001112	\$SETUP=	000017
TST1	006162	TST6	012514	WCU	= 000001	\$ESCAP	001214	\$SS1	= 000000
TST10	013276	TST60	034634	WLE	= 004000	\$FILLC	001156	\$STUP	= 177777
TST11	013600	TST61	035062	WORKBS	004770	\$FILLS	001155	\$SVLAD	043334
TST12	014126	TST62	035316	WORKNM	004772	\$GDADR	001120	\$SWR	= 167700
TST13	014430	TST63	035554	WORKVC	004774	\$GDDAT	001124	\$SWRMK=	000000
TST14	015112	TST64	036012	WORUNT	005510	\$GET42	037544	\$TIMES	001212
TST15	016022	TST65	036250	WRCHDT	004152	\$HD	= 000000	\$TKB	001146
TST16	016734	TST66	036506	WRCHK	004150	\$HIOCT	044742	\$TKCNT	044044
TST17	017646	TST67	036744	WRFROM	004210	\$ICNT	001104	\$TKINT	044064
TST2	010632	TST7	012774	WRIDAT	004154	\$ILLUP	046674	\$TKQEN=	044063
TST20	020562	TST70	037210	WRIFOR	004156	\$INTAG	001135	\$TKQIN	044046
TST21	021476	TSUNIT	005054	WRL	= 004000	\$ITEMB	001114	\$TKQOU	044050
TST22	022110	TSVEC	005120	WRTBIT	004650	\$LF	001224	\$TKGSR	044052
TST23	022404	TUF	= 000100	WRU	= 000400	\$LPADR	001106	\$TKS	001144
TST24	022566	TYPDS	= 104405	WSU	= 000004	\$LPERR	001110	\$TKSRV	044134
TST25	022750	TYPE	= 104401	\$AUTOB	001134	\$MNEW	044573	\$TMP0	001176
TST26	023132	TYPERR	046022	\$BDADR	001122	\$MSWR	044562	\$TMP1	001200
TST27	023326	TYPNT	005330	\$BDDAT	001126	\$MXCNT	043376	\$TMP2	001202
TST3	011430	TYPC	= 104402	\$BELL	001216	\$NULL	001154	\$TMP3	001204
TST30	023610	TYPON	= 104404	\$CHARC	044040	\$NWTST=	000001	\$TMP4	001206
TST31	024212	TYPOS	= 104403	\$CMTAG	001100	\$OCNT	046444	\$TMP5	001210
TST32	024614	UNIT	005010	\$CM1	= 000006	\$OMODE	046446	\$TN	= 000071
TST33	025342	UNLOAD	004136	\$CM2	= 000014	\$OVER	043362	\$TPB	001152
TST34	026070	UNS	= 040000	\$CM3	= 000006	\$PASS	001100	\$TPFLG	001157
TST35	026466	USEVEC	005004	\$CM4	= 000006	\$POWER	046702	\$TPS	001150
TST36	027064	US1	= 000001	\$CNTLC	044543	\$PWRAD	046670	\$TRAP	046450
TST37	027462	US2	= 000002	\$CNTLG	044555	\$PWRDN	046530	\$TRAP2	046472
TST4	011732	US4	= 000004	\$CNTLU	044550	\$PWRMG	046664	\$TRP	= 000012
TST40	030060	UWR	= 000010	\$CRLF	001223	\$PWRUP	046602	\$TRPAD	046504
TST41	030306	VECPNT	005310	\$DBLK	043614	\$QUES	001222	\$TSTNM	001102
TST42	030534	VECTOR	005512	\$DOAGN	037564	\$RDCHR	044334	\$TTYIN	044532
TST43	030762	VUF	= 000002	\$DTBL	043604	\$RDLIN	044424	\$TYPDS	043400
TST44	031210	VU30	= 010000	\$ENDAD	037554	\$RDOCT	044604	\$TYPE	043624
TST45	031444	VV	= 000100	\$ENDCT	037522	\$RDSZ	= 000011	\$TYPEC	043774
TST46	031672	WAITBT	004672	\$ENDMG	037573	\$REGAD	001160	\$TYPEX	044042
TST47	032120	WAITPC	004666	\$ENULL	037570	\$REG0	001162	\$TYPOC	046246
TST5	012212	WAITRE	004670	\$EOP	037466	\$REG1	001164	\$TYPON	046262
TST50	032346	WAIT.T	040460	\$EOPCT	037514	\$REG2	001166	\$TYPOS	046222
TST51	032574	WAT	= 104411	\$ERFLG	001103	\$REG3	001170	\$XTSTR	043140
TST52	033022	WATO	004646	\$ERMAX	001115	\$REG4	001172	\$SET4=	000000
TST53	033250	WC	= 000002	\$ERROR	044744	\$REG5	001174	\$OFILL	046445
.	= 067026								

ERRORS DETECTED: 0  
DEFAULT GLOBALS GENERATED: 0

\*DERHAC,DERHAC/SOL/PAGNUM=DERHAC.SML(400,4335),DERHAC.P11(400,4355)  
RUN-TIME: 105 157 2 SECONDS  
RUN-TIME RATIO: 327/265=1.2  
CORE USED: 38K (75 PAGES)

