

RH70-A,C

11/70 - 11/74 CTRLR
CERHAE0

AH-8001E-MC

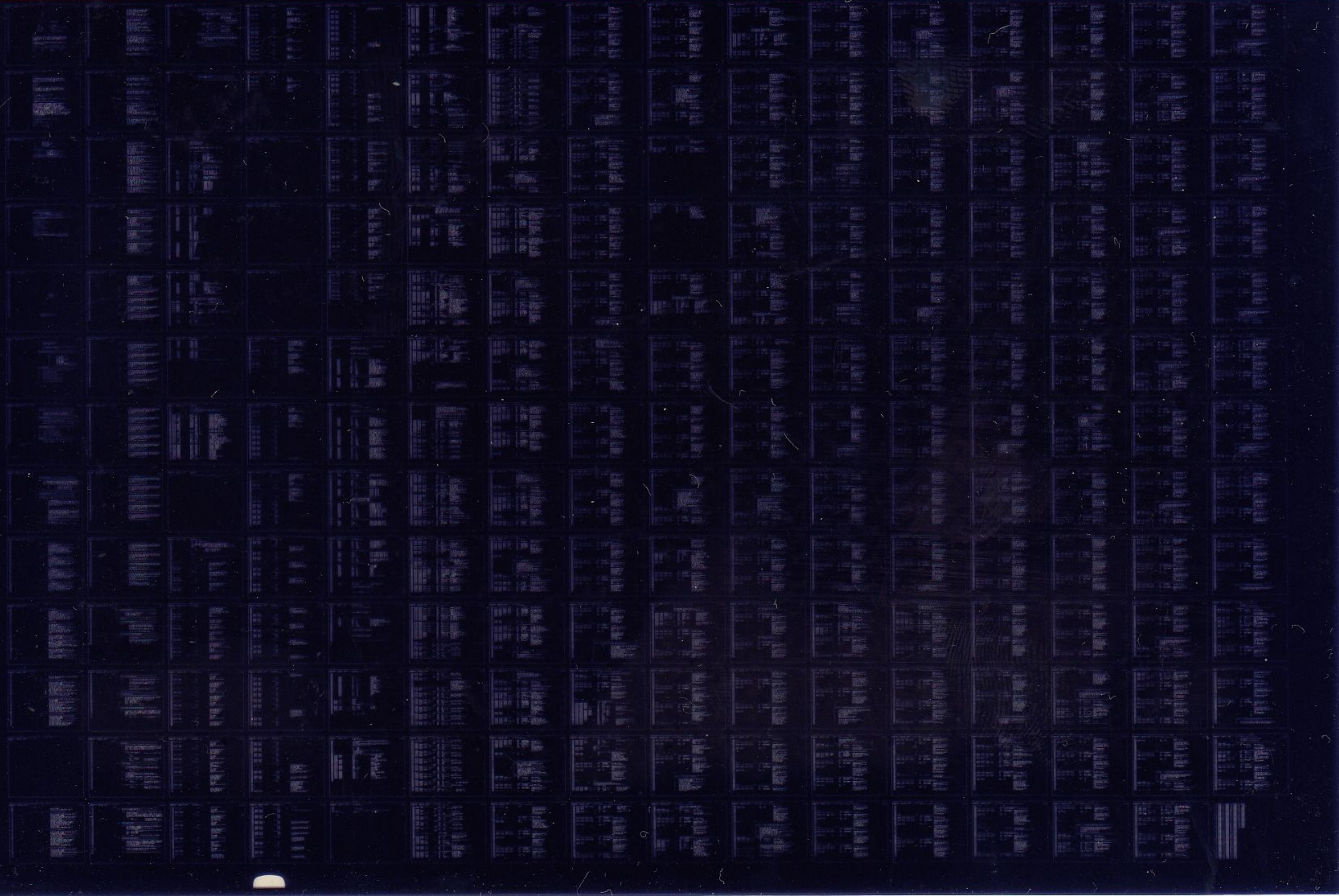
COPYRIGHT '75-80

FICHE 1 OF 2

JAN 1980

digital

MADE IN USA



RH70-A,C

11/70 - 11/74 CTRLR
CERHAE0

AH-8001E-MC

COPYRIGHT '75-80
FICHE 2 OF 2

JAN 1980

digital

MADE IN USA

.REM %

3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49

IDENTIFICATION

PRODUCT CODE: AC-7999E-MC
PRODUCT NAME: CERHAE0 11/70-11/74 CTRLR DIAG
PRODUCT DATE: MAY, 1979
MAINTAINER: DIAGNOSTIC ENGINEERING

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

NO RESPONSIBILITY IS ASSUMED FOR THE USE OR RELIABILITY OF SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL OR ITS AFFILIATED COMPANIES.

COPYRIGHT (C) 1975, 1979 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL	PDP	UNIBUS	MASSBUS
DEC	DECUS	DECTAPE	

50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105

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 RP04, RM03, RS03, RS04, TU16). THE PERIPHERAL MUST BE COMPLETE AND WORKING. FOR EXAMPLE THE RP04, RM03, RS03, RS04 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.

2.4 PROGRAMMABLE DRIVES (DUAL PORT: ENABLED)

THIS REV INCORPORATES A SAFEGUARD TO PREVENT INADVERTENT CORRUPTION OF MEDIA IN PROGRAMMABLE DRIVES. THIS IS A POTENTIAL HAZARD IN RUNNING THIS PROGRAM IN A MULTI-PROCESSOR SYSTEM. FOR THE STANDARD STARTING ADDRESS OF 200 THE PROGRAM HAS BEEN MODIFIED TO PREVENT INITIALIZING DRIVES FOUND TO BE PROGRAMMABLE. THIS MODIFICATION APPLIES ONLY TO THE FIELD ENVIRONMENT (XXDP CHAIN, STANDALONE) WHERE LOCATION 42 DOES NOT EQUAL LOCATION 46. FOR THE MANUFACTURING ENVIRONMENT (WHERE LOCATION 42 EQUALS LOCATION 46) PROGRAMMABLE DRIVES WILL NOT BE INHIBITED. IF THE OPERATOR DESIRES TO RUN THIS PROGRAM USING PROGRAMMABLE DRIVES IN A FIELD ENVIRONMENT USE STARTING ADDRESS 220, WHERE 220 IS THE SAME AS 200 WITHOUT INHIBITING PROGRAMMABLE DRIVES. SEE SECTION 4.2 FOR A SUMMARY OF ALL STARTING ADDRESSES.

3. LOADING PROCEDURE

USE STANDARD LOADING PROCEDURES FOR .BIN TAPES.

4. STARTING PROCEDURE

4.1 OPERATIONAL SWITCH SETTINGS

SWITCH	USE
-----	-----
15	HALT ON ERROR

106	14	LOOP ON TEST
107	13	INHIBIT ERROR TYPEOUTS
108	11	INHIBIT ITERATIONS
109	10	BELL ON ERROR
110	9	LOOP ON ERROR
111	8	LOOP ON TEST IN SWR<7:0>

4.2 STARTING ADDRESS

112		
113		
114		
115		START AT ADDRESS 200 --- FOR STANDARD RH70.
116		PROGRAMMABLE DRIVES INHIBITED
117		SEE SEC 2.4
118		START AT ADDRESS 210 --- FOR NONSTANDARD RH70.
119		PROGRAMMABLE DRIVES NOT INHIBITED
120		STARTING ADDRESS 210 SHOULD BE USED TO TEST
121		ANY ONE RH INSTEAD OF ALL THE RH GIVEN BY 200 START
122		START AT ADDRESS 220--- SAME AS 200 BUT WITH NO INHIBITIONS
123		

124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153

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, RM - 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.

- 154
- 155
- 156
- 157
- 158
- 159
- 160
- 161
- 162
- 163
- 164
- 165
- 166
- 167
- 168
- 169
- 170
- 171
- 172
- 173
- 174
- 175
- 176
- 177
- 178
- 179
- 180
- 181
- 182
- 183
- 184
- 185
- 186
- 187
- 188
- 189
- 190
- 191
- 192
- 193
- 194
- 195
- 196
- 197
- 198
- 199

DOCUMENT
.....
CERHAEO
.....

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



200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244

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 SW8 LOW

27 *****
BASIC DEFINITIONS

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

245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282

```

*****
139 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.

```


283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336

CERHAEO

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 0 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

337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391

CERHAEO

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!BITS!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.

392 CERHAEO
393
394 2618 TEST 7 BIT BANG RHBA
395 2620 TEST LOADING AND READING OF ALL POSSIBLE BITS
396 IN RHBA REGISTER.
397 USE A PATTERN OF WALKING 1'S (1,2,4,10 ETC)
398 AND WALKING 0'S (-2,-3,-5 ETC)
399 IN THIS TEST 0 BITS WILL NOT BE WRITTEN INTO
400 AND 0 BITS WILL ALWAYS BE SET
401 AND BIT00 BITS WILL ALWAYS BE CLEARED
402 IF SWITCH 14 OR 9 OR 8 ARE SET FOR DEBUGGING
403 THEN A RH CLEAR (RHCS2 BIT #5) WILL BE
404 GIVEN TO AID SCOPE SYNC'S ON THE CLEAR
405 SIGNAL.
406
407 2699 TEST 10 BIT BANG RHBAE
408
409 2701 TEST LOADING AND READING OF ALL POSSIBLE BITS
410 IN RHBAE REGISTER.
411 USE A PATTERN OF WALKING 1'S (1,2,4,10 ETC)
412 AND WALKING 0'S (-2,-3,-5 ETC)
413 IN THIS TEST 177700 BITS WILL NOT BE WRITTEN INTO
414 AND 0 BITS WILL ALWAYS BE SET
415 AND 177700 BITS WILL ALWAYS BE CLEARED
416 IF SWITCH 14 OR 9 OR 8 ARE SET FOR DEBUGGING
417 THEN A RH CLEAR (RHCS2 BIT #5) WILL BE
418 GIVEN TO AID SCOPE SYNC'S ON THE CLEAR
419 SIGNAL.
420
421 2784 TEST 11 SILO TEST 1 (ONE WORD WRITE)
422 AFTER A RH CLEAR IS GIVEN (SET BIT #5 IN RHCS2)
423 RHCS2 IS CHECKED TO HAVE "IR" (BIT #6) HIGH
424 TOGETHER WITH UNIT NUMBER
425 ONE WORD OF ALL ZEROS IS WRITTEN INTO RHDB
426 BY "CLR"
427 "DR" (BIT #7) IS WAITED FOR BY A TRAP INSTRUCTION
428 CALLED "WAT" (NO TIMING IS DONE)
429 HOWEVER IF "DR" DOES NOT SET WITHIN "WAT" COUNT
430 DOWN AN ERROR IS REPORTED
431 RHDB IS READ AND CHECKED TO CONTAIN ZEROS
432 RHCS2 WILL BE CHECKED TO HAVE "IR" AND UNIT NUMBER
433 RHCS1, RHCS3, RHBA, RHBAE, RHWC, WILL BE CHECKED TO HAVE
434 APPROPRIATE VALUES
435
436 2973 TEST 12 SILO TEST 2 (ONE WORD WRITE)
437 AFTER A RH CLEAR IS GIVEN (SET BIT #5 IN RHCS2)
438 RHCS2 IS CHECKED TO HAVE "IR" (BIT #6) HIGH
439 TOGETHER WITH UNIT NUMBER
440 ONE WORD OF ALL ONES IS WRITTEN INTO RHDB
441 BY "CLR"
442 "DR" (BIT #7) IS WAITED FOR BY A TRAP INSTRUCTION
443 CALLED "WAT" (NO TIMING IS DONE)
444 HOWEVER IF "DR" DOES NOT SET WITHIN "WAT" COUNT
445 DOWN AN ERROR IS REPORTED
446 RHDB IS READ AND CHECKED TO CONTAIN ALL ONES
447

CERHAEO

448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503

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 12525
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 (SILO SIZE AND 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
A COUNT PATTERN 0,1,2,... IS WRITTEN INTO RHDB
UNTIL "IR" IS LOW IN RHCS2. THIS DETERMINES
THE WORD SIZE OF THE SILO, WHICH IS
THEN TYPED OUT (IN DECIMAL).
'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 READS.
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
A PATTERN OF FLOATING ONES (1,2,4,10,20,40,100,200,...)
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 READS.
THEN RHCS2 IS CHECKED TO HAVE "IR"

CERHAEO MACY11 30A(1052) 02-MAY-79 14:35 PAGE 14
CERHAE.P11 02-MAY-79 14:08

M 1

SEQ 0012

504

CERHAEO

505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559

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

4227 TEST 16 SILO TEST 6 (FLOATING ONES)

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
A PATTERN OF FLOATING ONES STARTING 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 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)

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
'N' 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 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)

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
A PATTERN OF FLOATING ZEROS STARTING 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

CERHAEO

560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613

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

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)

614 CERHAEO
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668

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 6 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' INSTRUCTI
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 10 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

CERHAEO

669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724

6424

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

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)

CERHAEO

725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775

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

776	CERHAEO	8349	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
777			
778			
779			
780			
781			
782			
783			
784			
785			
786			
787			
788			
789		8581	TEST 40 TEST DBL (RHCS3 BIT #10) TEST A
790			
791		8583	CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
792			
793		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
794			
795			
796			
797			
798			
799		8694	TEST 41 TEST DBL (RHCS3 BIT #10) TEST B
800			
801		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
802			
803			
804			
805			
806			
807			
808		8808	TEST 42 TEST DBL (RHCS3 BIT #10) TEST C
809			
810		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
811			
812			
813			
814			
815			
816			
817		8921	TEST 43 TEST DBL (RHCS3 BIT #10) TEST D
818			
819		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
820			
821			
822			
823			
824			
825			

826	CERHAEO	9034	TEST 44 TEST DBL (RHCS3 BIT #10) TEST E
827		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
828		9153	TEST 45 TEST DBL (RHCS3 BIT #10) TEST F
829		9155	CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5) SET UP FOR A ELEVEN WORD WRITE FROM AN ODD WORD BOUNDAR 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
830		9268	TEST 46 TEST DBL (RHCS3 BIT #10) TEST G
831		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
832		9381	TEST 47 TEST DBL (RHCS3 BIT #10) TEST H
833		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
834		9495	TEST 50 TEST DBL (RHCS3 BIT #10) TEST I
835		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
836		9502	CHECK RHCS1,RHCS2,RHBA,RHBAE,RHWC
837		9608	TEST 51 TEST DBL (RHCS3 BIT #10) TEST J
838			
839			
840			
841			
842			
843			
844			
845			
846			
847			
848			
849			
850			
851			
852			
853			
854			
855			
856			
857			
858			
859			
860			
861			
862			
863			
864			
865			
866			
867			
868			
869			
870			
871			
872			
873			
874			
875			

876	CERHAEO	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
877			
878			
879			
880			
881			
882			
883			
884		9721	TEST 52 TEST DBL (RHCS3 BIT #10) TEST K
885			
886		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
887			
888			
889			
890			
891			
892			
893		9835	TEST 53 TEST DBL (RHCS3 BIT #10) TEST L
894			
895		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
896			
897			
898			
899			
900			
901			
902			
903		9955	TEST 54 TEST DBL (RHCS3 BIT #10) TEST M
904			
905		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
906			
907			
908			
909			
910			
911			
912		10068	TEST 55 TEST DBL (RHCS3 BIT #10) TEST N
913			
914		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
915			
916			
917			
918			
919			
920			
921		10181	TEST 56 TEST DBL (RHCS3 BIT #10) TEST O
922			
923		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
924			
925			
926			
927			
928			
929			

930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979

CERHAEO

10295 TEST 57 TEST DBL (RHCS3 BIT #10) TEST P

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

10409 TEST 60 TEST DBL (RHCS3 BIT #10) TEST Q

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

10522 TEST 61 TEST DBL (RHCS3 BIT #10) TEST R

10524 CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
 SET UP FOR A TWO WORD WRITE REVERSE FROM AN EVEN WORD BO
 WITH BAI IN RHCS2 BIT #3 SET
 DO A TWO WORD WRITE REVERSE FROM AN EVEN WORD BOUNDARY

10528 THIS SHOULD NOT SET RHCS3 BIT #10 DBL
 CHECK RHCS3
 CHECK RHCS1,RHCS2,RHBA,RHBAE,RHWC

10642 TEST 62 TEST DBL (RHCS3 BIT #10) TEST S

10644 CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
 SET UP FOR A NINE WORD READ REVERSE FROM AN EVEN WORD BO
 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

10761 TEST 63 TEST DBL (RHCS3 BIT #10) TEST T

10763 CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
 SET UP FOR A TEN WORD READ REVERSE FROM AN EVEN WORD BOU
 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

CERHAEO

980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028

10879 TEST 64 TEST DBL (RHCS3 BIT #10) TEST U

10881 CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
SET UP FOR A TEN WORD READ REVERSE FROM AN ODD WORD BOUN
DO A TEN WORD READ REVERSE FROM AN ODD 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

10998 TEST 65 TEST DBL (RHCS3 BIT #10) TEST V

11000 CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
SET UP FOR A ELEVEN WORD READ REVERSE FROM AN EVEN WORD
DO A ELEVEN 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

11117 TEST 66 TEST DBL (RHCS3 BIT #10) TEST W

11119 CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
SET UP FOR A ELEVEN WORD READ REVERSE FROM AN ODD WORD B
DO A ELEVEN WORD READ REVERSE FROM AN ODD WORD BOUNDARY

11122 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

11235 TEST 67 TEST DBL (RHCS3 BIT #10) TEST X

11237 CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
SET UP FOR A TEN WORD READ REVERSE FROM AN ODD WORD BOUN
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, RHWC
IF MAG. TAPE NOT USED, THIS TEST IS NOT DONE

11362 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

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

CERHAEO

```

*****
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  RP04,5,6/RM03 DATA TRANSFER SUBROUTINE
*****

11706  THIS SUBROUTINE WRITE/READ/WRITE CHECK ON THE RP04 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.

```


1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116

CERHAEO

```
11758 *****  
TMO2 DATA TRANSFER SUBROUTINE  
*****  
11760 THIS SUBROUTINE WRITES/READS/WRITE CHECKS ON THE TMO2-TU  
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 R0,@COMND  
11915 THIS ROUTINE WILL ALLOW THE CHANGE OF THE BASE  
ADDRESS FROM 176700 TO ANY TYPED VALUE  
11982 *****  
11985 *****  
SCOPE HANDLER ROUTINE  
*****  
11987 THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT  
AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7  
AND LOAD THE ERROR FLAG (SERFLG) 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
```

1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169

CERHAEO

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:
RDLIN ;:INPUT A STRING FROM THE TTY
RETURN HERE ;:ADDRESS OF FIRST CHARACTER WIL
;:TERMINATOR WILL BE A BYTE OF A

1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263

CERHAEO

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.

1264
1265
1266
1267
1268
1269
1270
1271
1272

CERHAEO

12746

POWER DOWN AND UP ROUTINES

12786

%

1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328

001000

000011

000012

000015

000200

177776

177774

177772

177570

177570

000000

000001

000002

000003

000004

000005

000006

000007

000006

000007

000000

```

.TITLE CERHAEO
.*COPYRIGHT (C) 1975-1977
.*DIGITAL EQUIPMENT CORP.
.*MAYNARD, MASS. 01754
.*
.*
.*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
.*PACKAGE (MAINDEC-11-DZQAC-C1),MAR 24, 1976.
.*
.SBTTL 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>
.*      7              STOP FURTHER COMPARES IF SW08 IS LOW
.*      6              TYPE ALL REG. WITH ERROR IF SW8 LOW
.SBTTL BASIC DEFINITIONS
.*INITIAL ADDRESS OF THE STACK POINTER *** 1000 ***
STACK= 1000
.EQUIV EMT,ERROR      ;;BASIC DEFINITION OF ERROR CALL
.EQUIV IOT,SCOPE      ;;BASIC DEFINITION OF SCOPE CALL

.*MISCELLANEOUS DEFINITIONS
HT= 11                ;;CODE FOR HORIZONTAL TAB
LF= 12                ;;CODE FOR LINE FEED
CR= 15                ;;CODE FOR CARRIAGE RETURN
CRLF= 200             ;;CODE FOR CARRIAGE RETURN-LINE FEED
PS= 177776           ;;PROCESSOR STATUS WORD
.EQUIV PS,PSW
STKLMT= 177774        ;;STACK LIMIT REGISTER
PIRQ= 177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
DSWR= 177570          ;;HARDWARE SWITCH REGISTER
DDISP= 177570         ;;HARDWARE DISPLAY REGISTER

.*GENERAL PURPOSE REGISTER DEFINITIONS
R0= %0                ;;GENERAL REGISTER
R1= %1                ;;GENERAL REGISTER
R2= %2                ;;GENERAL REGISTER
R3= %3                ;;GENERAL REGISTER
R4= %4                ;;GENERAL REGISTER
R5= %5                ;;GENERAL REGISTER
R6= %6                ;;GENERAL REGISTER
R7= %7                ;;GENERAL REGISTER
R6=SP                 ;;STACK POINTER
R7=PC                 ;;PROGRAM COUNTER

.*PRIORITY LEVEL DEFINITIONS
PRO= 0                ;;PRIORITY LEVEL 0

```


CERHAEO MACY11 30A(1052) 02-MAY-79 14:35 PAGE 32
 CERHAE.P11 02-MAY-79 14:08 BASIC DEFINITIONS

SEQ 0030

```

1329      000040      PR1=      40      ::PRIORITY LEVEL 1
1330      000100      PR2=     100      ::PRIORITY LEVEL 2
1331      000140      PR3=     140      ::PRIORITY LEVEL 3
1332      000200      PR4=     200      ::PRIORITY LEVEL 4
1333      000240      PR5=     240      ::PRIORITY LEVEL 5
1334      000300      PR6=     300      ::PRIORITY LEVEL 6
1335      000340      PR7=     340      ::PRIORITY LEVEL 7
1336
1337
1338      100000      :*'SWITCH REGISTER' SWITCH DEFINITIONS
1339      040000      SW15=   100000
1340      020000      SW14=   40000
1341      010000      SW13=   20000
1342      004000      SW12=   10000
1343      002000      SW11=   4000
1344      001000      SW10=   2000
1345      000400      SW09=   1000
1346      000200      SW08=   400
1347      000100      SW07=   200
1348      000040      SW06=   100
1349      000020      SW05=   40
1350      000010      SW04=   20
1351      000004      SW03=   10
1352      000002      SW02=   4
1353      000001      SW01=   2
1354      .EQUIV      SW00=   1
1355      .EQUIV      SW09,SW9
1356      .EQUIV      SW08,SW8
1357      .EQUIV      SW07,SW7
1358      .EQUIV      SW06,SW6
1359      .EQUIV      SW05,SW5
1360      .EQUIV      SW04,SW4
1361      .EQUIV      SW03,SW3
1362      .EQUIV      SW02,SW2
1363      .EQUIV      SW01,SW1
1364      .EQUIV      SW00,SW0
1365
1366      100000      :*DATA BIT DEFINITIONS (BIT00 TO BIT15)
1367      040000      BIT15=  100000
1368      020000      BIT14=  40000
1369      010000      BIT13=  20000
1370      004000      BIT12=  10000
1371      002000      BIT11=  4000
1372      001000      BIT10=  2000
1373      000400      BIT09=  1000
1374      000200      BIT08=  400
1375      000100      BIT07=  200
1376      000040      BIT06=  100
1377      000020      BIT05=  40
1378      000010      BIT04=  20
1379      000004      BIT03=  10
1380      000002      BIT02=  4
1381      000001      BIT01=  2
1382      .EQUIV      BIT00=  1
1383      .EQUIV      BIT09,BIT9
1384      .EQUIV      BIT08,BIT8
1385      .EQUIV      BIT07,BIT7

```

CERHAEO MACY11 30A(1052) 02-MAY-79 14:35 PAGE 33
 CERHA.E.P11 02-MAY-79 14:08 BASIC DEFINITIONS

SEQ 0031

```

1385      .EQUIV BIT06,BIT6
1386      .EQUIV BIT05,BIT5
1387      .EQUIV BIT04,BIT4
1388      .EQUIV BIT03,BIT3
1389      .EQUIV BIT02,BIT2
1390      .EQUIV BIT01,BIT1
1391      .EQUIV BIT00,BIT0
1392
1393      ;*BASIC "CPU" TRAP VECTOR ADDRESSES
1394      ERRVEC= 4          ;; TIME OUT AND OTHER ERRORS
1395      RESVEC= 10        ;; RESERVED AND ILLEGAL INSTRUCTIONS
1396      TBITVEC=14       ;; "T" BIT
1397      TRTVEC= 14       ;; TRACE TRAP
1398      BPTVEC= 14       ;; BREAKPOINT TRAP (BPT)
1399      IOTVEC= 20        ;; INPUT/OUTPUT TRAP (IOT) **SCOPE**
1400      PWRVEC= 24       ;; POWER FAIL
1401      EMTVEC= 30       ;; EMULATOR TRAP (EMT) **ERROR**
1402      TRAPVEC=34       ;; "TRAP" TRAP
1403      TKVEC= 60        ;; TTY KEYBOARD VECTOR
1404      TPVEC= 64        ;; TTY PRINTER VECTOR
1405      PIRQVEC=240      ;; PROGRAM INTERRUPT REQUEST VECTOR
1406
1407      .SBTTL TRAP CATCHER
1408
1409      .=0
1410      ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
1411      ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
1412      ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
1413      .=174
1414      000174 000000     DISPREG: .WORD 0          ;; SOFTWARE DISPLAY REGISTER
1415      000176 000000     SWREG:   .WORD 0          ;; SOFTWARE SWITCH REGISTER
1416      .SBTTL STARTING ADDRESS(ES)
1417      000200 000137 007230  JMP @MBEGIN ;; JUMP TO STARTING ADDRESS OF PROGRAM
1418      .=46
1419      000046 040454     SENDAD
1420      .=52
1421      000052 000000     000000
1422      .=210
1423      000210 000137 007244  JMP @MBEGIN2   ; JUMP SELECT TEST
1424      .=220
1425      000220 000137 007260  JMP @MBEGIN3   ; LIKE 200 BUT ENABLE PROGRAMMABLE DRIVES
1426      .SBTTL MEMORY MANAGEMENT DEFINITIONS
1427
1428      ;*KT11 VECTOR ADDRESS
1429
1430      MMVEC= 250
1431
1432      ;*KT11 STATUS REGISTER ADDRESSES
1433
1434      177572     SR0= 177572
1435      177574     SR1= 177574
1436      177576     SR2= 177576
1437      172516     SR3= 172516
1438
1439      ;*KERNEL "I" PAGE DESCRIPTOR REGISTERS
1440

```


1441	172300	KIPDR0= 172300
1442	172302	KIPDR1= 172302
1443	172304	KIPDR2= 172304
1444	172306	KIPDR3= 172306
1445	172310	KIPDR4= 172310
1446	172312	KIPDR5= 172312
1447	172314	KIPDR6= 172314
1448	172316	KIPDR7= 172316

;*KERNEL 'I' PAGE ADDRESS REGISTERS

1451		
1452	172340	KIPAR0= 172340
1453	172342	KIPAR1= 172342
1454	172344	KIPAR2= 172344
1455	172346	KIPAR3= 172346
1456	172350	KIPAR4= 172350
1457	172352	KIPAR5= 172352
1458	172354	KIPAR6= 172354
1459	172356	KIPAR7= 172356

1460		::*****
1461		
1462	001110	. =1110

1463
1464
1465
1466
1467
1468
1469 001100
1470 001100 001100
1471 001100 000000
1472 001102 000
1473 001103 000
1474 001104 000000
1475 001106 000000
1476 001110 000000
1477 001112 000000
1478 001114 000
1479 001115 001
1480 001116 000000
1481 001120 000000
1482 001122 000000
1483 001124 000000
1484 001126 000000
1485 001130 000000
1486 001132 000000
1487 001134 000
1488 001135 000
1489 001136 000000
1490 001140 177570
1491 001142 177570
1492 001144 177560
1493 001146 177562
1494 001150 177564
1495 001152 177566
1496 001154 000
1497 001155 002
1498 001156 012
1499 001157 000
1500 001160 000000
1501
1502 001162 000000
1503 001164 000000
1504 001166 000000
1505 001170 000000
1506 001172 000000
1507 001174 000000
1508 001176 000000
1509 001200 000000
1510 001202 000000
1511 001204 000000
1512 001206 000000
1513 001210 000000
1514 001212 000000
1515 001214 000000
1516 001216 177607 000377
1517 001222 077
1518 001223 015

.SBTTL COMMON TAGS

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

```

.=1100
SCMTAG:          .WORD      0      ;; START OF COMMON TAGS
SPASS:          .WORD      0      ;; CONTAINS PASS COUNT
STSTNM:         .BYTE      0      ;; CONTAINS THE TEST NUMBER
SERFLG:         .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
                .WORD      0
SAUTOB:         .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)
$REGAD:        .WORD      0      ;; CONTAINS THE ADDRESS FROM
                .WORD      0      ;; WHICH ($REGO) WAS OBTAINED
$REG0:         .WORD      0      ;; CONTAINS (($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)+10)
$REG5:         .WORD      0      ;; CONTAINS (($REGAD)+12)
$TMP0:         .WORD      0      ;; USER DEFINED
$TMP1:         .WORD      0      ;; USER DEFINED
$TMP2:         .WORD      0      ;; USER DEFINED
$TMP3:         .WORD      0      ;; USER DEFINED
$TMP4:         .WORD      0      ;; USER DEFINED
$TMP5:         .WORD      0      ;; USER DEFINED
$TIMES:        0                  ;; MAX. NUMBER OF ITERATIONS
$ESCAPE:       0                  ;; ESCAPE ON ERROR ADDRESS
$BELL:         .ASCIZ   <207><377><377> ;; CODE FOR BELL
$QUES:         .ASCII   '?!'      ;; QUESTION MARK
$CRLF:         .ASCII   <15>      ;; CARRIAGE RETURN

```


CERHAEO MACY11 30A(1052) 02-MAY-79 14:35 PAGE 36
CERHAE.P11 02-MAY-79 14:08 COMMON TAGS

SEQ 0034

1519 001224 000012
1520

\$LF: .ASCIZ <12> ;:LINE FEED
:.....


```
1521 .SBTTL ERROR POINTER TABLE
1522
1523 : *THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
1524 : *THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
1525 : *LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
1526 : *NOTE1: IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
1527 : *NOTE2: EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
1528
1529 : * EM ::POINTS TO THE ERROR MESSAGE
1530 : * DH ::POINTS TO THE DATA HEADER
1531 : * DT ::POINTS TO THE DATA
1532 : * DF ::POINTS TO THE DATA FORMAT
1533
1534
1535 001226 $ERRTB:
1536
1537 :*****
1538 :
1539
1540
1541
1542
1543 :ITEM 1
1544 001226 047636 EM1 :THE RH BASE ADDRESS WAS 772040
1545 :INDICATING A RS04 OR RS03
1546 :BUT THE DRIVE TYPE DID NOT
1547 :CONTAIN 0,1,2,3 OR 4
1548 001230 066102 DH1 :PC
1549 :RHDT
1550 001232 067604 DT1 :$ERRPC,$BDDAT,0
1551 001234 067776 DF1 :0
1552
1553 :ITEM 2
1554 001236 050010 EM2 :THE RH BASE ADDRESS WAS 776700
1555 :INDICATING AN RP04, RP05, OR RP06
1556 :BUT THE DRIVE TYPE DID NOT
1557 :CONTAIN 20020 24020, 20021, 24021,
1558 :20022, OR 24022
1559
1560 001240 066102 DH1 :PC
1561 :RHDT
1562 001242 067604 DT1 :$ERRPC,$BDDAT,0
1563 001244 067776 DF1 :0
1564
1565 :ITEM 3
1566 001246 050212 EM3 :THE RH BASE ADDRESS WAS 772440
1567 :INDICATING A TM02. BUT THE
1568 :DRIVE TYPE DID NOT CONTAIN
1569 :142010 INDICATING NO
1570 :TM02
1571 001250 066102 DH1 :PC
1572 :RHDT
1573 001252 067604 DT1 :$ERRPC,$BDDAT,0
1574 001254 067776 DF1 :0,0
1575
1576 :ITEM 4
```


Line	Code	Address	Pointer	Description
1577	001256	050346	EM4	:THE RH BASE ADDRESS WAS 776300
1578				:INDICATING MORE THAN ONE
1579				:RH DEVICE, BUT THE DRIVE
1580				:TYPE DID NOT CONTAIN
1581				:0,1,2,3,4,20020,24020,20021,24021,
1582				:20022,24022, OR 142010 INDICATING NO
1583				:RH DEVICE IS PRESENT
1584	001260	066102	DH1	:PC
1585				:RHDT
1586	001262	067604	DT1	:\$ERRPC,\$BDDAT,0
1587	001264	067776	DF1	:0,0
1588				
1589			:ITEM 5	
1590	001266	050645	EM5	:THE UNIBUS TIMED OUT
1591				:FOR EACH OF THE FOLLOWING
1592				:ADDRESSES, INDICATING
1593				:NO RH ON THE SYSTEM
1594				:
1595				:SO ABORT PROGRAM
1596	001270	066111	DH5	:PC
1597				:UNIBUS ADDRESSES ON WHICH
1598				:TIME OUT OCCURRED
1599	001272	067612	DT5	:\$ERRPC,\$RCS1,\$PCS1,\$MCS1,\$MIXCS1
1600	001274	070000	DF5	:0,0,0,0,0
1601				
1602			:ITEM 6	
1603	001276	051020	EM6	:AFTER AN RH CLEAR (BIT #5
1604				:IN RHCS2) RHCS2 DOES NOT
1605				:HAVE ONLY IR AND UNIT
1606				:NUMBER
1607	001300	066165	DH6	:PC
1608				:TEST NO
1609				:RHCS2 GOOD
1610				:RHCS2 BAD
1611	001302	067626	DT6	:\$ERRPC,\$TSTNM,\$GDDAT,\$BDDAT
1612	001304	070005	DF6	:0,0,0,0
1613				
1614			:ITEM 7	
1615	001306	051136	EM7	:AFTER CLEARING THE RH AND
1616				:WRITING ONE WORD INTO
1617				:RHDB AND READING IT
1618				:BACK CAUSED RHDB TO
1619				:HAVE WRONG VALUE GIVEN IN BAD RHDB
1620	001310	066230	DH7	:PC
1621				:TEST NO
1622				:RHDB GOOD
1623				:RHDB BAD
1624	001312	067626	DT6	:\$ERRPC,\$TSTNM,\$GDDAT,\$BDDAT
1625	001314	070005	DF6	:0,0,0,0
1626				
1627			:ITEM 10	
1628	001316	051330	EM10	:AFTER CLEARING THE RH AND
1629				:WRITING ONE WORD INTO
1630				:RHDB AND READING IT
1631				:BACK, CAUSED RHCS2 TO
1632				:HAVE WRONG DATA

1633					:GIVEN IN BAD RHCS2
1634	001320	066165		DH6	:PC
1635					:TEST NUMBER
1636					:RHCS2 GOOD
1637					:RHCS2 BAD
1638	001322	067626		DT6	:SERRPC,TSTNM,\$GDDAT,\$BDDAT
1639	001324	070005		DF6	:0,0,0,0
1640					
1641			:ITEM 11		
1642	001326	051523		EM11	:AFTER CLEARING THE RH AND
1643					:WRITING ONE WORD INTO
1644					:RHDB AND READING IT
1645					:BACK, CAUSED RHCS1 TO
1646					:HAVE WRONG DATA
1647					:GIVEN IN BAD RHCS1
1648	001330	066270		DH11	:PC
1649					:TEST NUMBER
1650					:RHCS1 GOOD
1651					:RHCS1 BAD
1652	001332	067626		DT6	:SERRPC,TSTNM,\$GDDAT,\$BDDAT
1653	001334	070005		DF6	:0,0,0,0
1654					
1655			:ITEM 12		
1656	001336	051716		EM12	:AFTER CLEARING THE RH AND
1657					:WRITING ONE WORD INTO
1658					:RHDB AND READING IT
1659					:BACK, CAUSED RHCS3 TO
1660					:HAVE WRONG DATA
1661					:GIVEN IN BAD RHCS3
1662	001340	066332		DH12	:PC
1663					:TEST NUMBER
1664					:RHCS3 GOOD
1665					:RHCS3 BAD
1666	001342	067626		DT6	:SERRPC,TSTNM,\$GDDAT,\$BDDAT
1667	001344	070005		DF6	:0,0,0,0
1668					
1669			:ITEM 13		
1670	001346	052111		EM13	:AFTER CLEARING THE RH AND
1671					:WRITING ONE WORD INTO
1672					:RHDB AND READING IT
1673					:BACK, CAUSED RHBA TO
1674					:HAVE WRONG DATA
1675					:GIVEN IN BAD RHBA
1676	001350	066374		DH13	:PC
1677					:TEST NUMBER
1678					:RHBA GOOD
1679					:RHBA BAD
1680	001352	067626		DT6	:SERRPC,TSTNM,\$GDDAT,\$BDDAT
1681	001354	070005		DF6	:0,0,0,0
1682					
1683			:ITEM 14		
1684	001356	052303		EM14	:AFTER CLEARING THE RH AND
1685					:WRITING ONE WORD INTO
1686					:RHDB AND READING IT
1687					:BACK, CAUSED RHBAE TO
1688					:HAVE WRONG DATA

1689					:GIVEN IN BAD RHBAE
1690	001360	066434		DH14	:PC
1691					:TEST NUMBER
1692					:RHBAE GOOD
1693					:RHBAE BAD
1694	001362	067626		DT6	:SERRPC,TSTNM,\$GDDAT,\$BDDAT
1695	001364	070005		DF6	:0,0,0,0
1696					
1697					
1698	001366	052476		:ITEM 15 EM15	:AFTER CLEARING THE RH AND
1699					:WRITING ONE WORD INTO
1700					:RHDB AND READING IT
1701					:BACK, CAUSED RHWC TO
1702					:HAVE WRONG DATA
1703					:GIVEN IN BAD RHWC
1704	001370	066476		DH15	:PC
1705					:TEST NUMBER
1706					:RHWC GOOD
1707					:RHWC BAD
1708	001372	067626		DT6	:SERRPC,TSTNM,\$GDDAT,\$BDDAT
1709	001374	070005		DF6	:0,0,0,0
1710					
1711					
1712	001376	052670		:ITEM 16 EM16	:AFTER CLEARING THE RH AND
1713					:WRITING ONE WORD INTO
1714					:RHDB
1715					:CAUSED RHCS2 TO
1716					:HAVE WRONG DATA
1717					:GIVEN IN BAD RHCS2
1718	001400	066165		DH6	:PC
1719					:TEST NUMBER
1720					:RHCS2 GOOD
1721					:RHCS2 BAD
1722	001402	067626		DT6	:SERRPC,TSTNM,\$GDDAT,\$BDDAT
1723	001404	070005		DF6	:0,0,0,0
1724					
1725					
1726	001406	053063		:ITEM 17 EM17	:AFTER CLEARING THE RH AND
1727					:WRITING TWO WORD INTO
1728					:RHDB AND READING IT
1729					:BACK, CAUSED RHDB TO
1730					:HAVE WRONG DATA
1731					:GIVEN IN BAD RHDB
1732	001410	066230		DH7	:PC
1733					:TEST NUMBER
1734					:RHDB GOOD
1735					:RHDB BAD
1736	001412	067626		DT6	:SERRPC,TSTNM,\$GDDAT,\$BDDAT
1737	001414	070005		DF6	:0,0,0,0
1738					
1739					
1740	001416	053255		:ITEM 20 EM20	:AFTER CLEARING THE RH AND
1741					:WRITING TWO WORD INTO
1742					:RHDB AND READING ONE
1743					:BACK, CAUSED RHCS2 TO
1744					:HAVE WRONG DATA

1745				:GIVEN IN BAD RHCS?
1746	001420	066165	DH6	:PC
1747				:TEST NUMBER
1748				:RHCS2 GOOD
1749				:RHCS2 BAD
1750	001422	067626	DT6	:SERRPC,TSTNM,\$GDDAT,\$BDDAT
1751	001424	070005	DF6	:0,0,0,0
1752				
1753			:ITEM 21	
1754	001426	053451	EM21	:AFTER CLEARING THE RH AND
1755				:WRITING TWO WORD INTO
1756				:RHDB AND READING IT
1757				:BACK TWICE, CAUSED RHDB TO
1758				:HAVE WRONG DATA,
1759				:GIVEN IN BAD RHDB
1760	001430	066230	DH7	:PC
1761				:TEST NUMBER
1762				:RHDB GOOD
1763				:RHDB BAD
1764	001432	067626	DT6	:SERRPC,TSTNM,\$GDDAT,\$BDDAT
1765	001434	070005	DF6	:0,0,0,0
1766				
1767			:ITEM 22	
1768	001436	053651	EM22	:AFTER CLEARING THE RH AND
1769				:WRITING TWO WORD INTO
1770				:RHDB AND READING IT
1771				:BACK TWICE, CAUSED RHCS2 TO
1772				:HAVE WRONG DATA
1773				:GIVEN IN BAD RHCS2
1774	001440	066165	DH6	:PC
1775				:TEST NUMBER
1776				:RHCS2 GOOD
1777				:RHCS2 BAD
1778	001442	067626	DT6	:SERRPC,TSTNM,\$GDDAT,\$BDDAT
1779	001444	070005	DF6	:0,0,0,0
1780				
1781			:ITEM 23	
1782	001446	054052	EM23	:AFTER CLEARING THE RH AND
1783				:WRITING INTO
1784				:RHDB
1785				:CAUSED RHCS2 TO
1786				:HAVE WRONG DATA
1787				:GIVEN IN BAD RHCS2
1788	001450	066165	DH6	:PC
1789				:TEST NUMBER
1790				:RHCS2 GOOD
1791				:RHCS2 BAD
1792	001452	067626	DT6	:SERRPC,TSTNM,\$GDDAT,\$BDDAT
1793	001454	070005	DF6	:0,0,0,0
1794				
1795			:ITEM 24	
1796	001456	054234	EM24	:THE RH WAS CLEARED
1797				:AND A PATTERN OF WORDS
1798				:WERE WRITTEN INTO RHDB
1799				:READING RHDB FOR THE
1800				: 'N' TH. TIME GAVE WRONG

1801				:VALUE IN RHDB
1802				:N IS GIVEN IN 'WORD NO'
1803	001460	066536	DH24	:PC
1804				:TEST NO
1805				:WORD NO
1806				:RHDB GOOD
1807				:RHDB BAD
1808	001462	067640	DT24	:\$ERRPC,TSTNM,SILONM,\$GDDAT,\$BDDAT
1809	001464	070011	DF24	:0,0,0,0,0
1810				
1811			:ITEM 25	
1812	001466	054462	EM25	:THE RH WAS CLEARED AND
1813				:A PATTERN OF WORDS WERE
1814				:WRITTEN INTO RHDB
1815				:AFTER READING ALL WORDS
1816				:FOLLOWING REGISTER CONTAINED WRONG
1817				:VALUE
1818	001470	066165	DH6	:PC
1819				:TEST NO
1820				:RHCS2 GOOD
1821				:RHCS2 BAD
1822	001472	067626	DT6	:\$ERRPC,TSTNM,\$GDDAT,\$BDDAT
1823	001474	070005	DF6	:0,0,0,0
1824				
1825			:ITEM 26	
1826	001476	054462	EM25	
1827	001500	066270	DH11	
1828	001502	067626	DT6	
1829	001504	070005	DF6	:0,0
1830				
1831			:ITEM 27	
1832	001506	054462	EM25	
1833	001510	066332	DH12	:PC
1834				:TEST NO
1835				:RHCS3 GOOD
1836				:RHCS3 BAD
1837	001512	067626	DT6	:\$ERRPC,TSTNM,\$GDDAT,\$BDDAT
1838	001514	070005	DF6	:0,0,0,0
1839				
1840			:ITEM 30	
1841	001516	054462	EM25	
1842	001520	066374	DH13	:PC
1843				:TEST NO
1844				:RHBA GOOD
1845				:RHBA BAD
1846	001522	067626	DT6	:\$ERRPC,TSTNM,\$GDDAT,\$BDDAT
1847	001524	070005	DF6	:0,0,0,0
1848				
1849			:ITEM 31	
1850	001526	054462	EM25	:PC
1851	001530	066434	DH14	:TEST NO
1852				:RHBAE GOOD
1853				:RHBAE BAD
1854				:\$ERRPC,TSTNM,\$GDDAT,\$BDDAT
1855	001532	067626	DT6	:0,0,0,0
1856	001534	070005	DF6	

1857
 1858
 1859 001536 054462
 1860 001540 066476
 1861
 1862
 1863
 1864 001542 067626
 1865 001544 070005
 1866
 1867
 1868 001546 054705
 1869
 1870
 1871
 1872 001550 066606
 1873
 1874
 1875

:ITEM 32
 EM25
 DH15

DT6
 DF6

:ITEM 33
 EM33

DH33

:PC
 :TEST NO
 :RHWC GOOD
 :RHWC BAD
 :\$ERRPC,\$TSTNM,\$GDDAT,\$BDDAT
 :0,0,0,0

:SETTING RH CLEAR BIT #5
 :IN RHCS2 CAUSED
 :ERROR REGISTER 1 TO HAVE
 :WRONG VALUE
 :PC
 :TEST NO
 :RHER1 GOOD
 :RHER1 BAD

1876 001552 067626
1877 001554 070005

DT6
DF6

;\$ERRPC,TSTNM,\$GDDAT,\$BDDAT

CERHAEO MACY11 30A(1052) 02-MAY-79 14:35 PAGE 45
CERHAE.P11 02-MAY-79 14:08 ERROR PCINTER TABLE

SEQ 0043

1878

CERHAEO MACY11 30A(1052) 02-MAY-79 14:35 PAGE 46
CERHAE.P11 02-MAY-79 14:08 ERROR POINTER TABLE

F 4

SEQ 0044

1879

:ITEM 34

1880	001556	055021	EM34	:AN RH LCLEAR WAS GIVEN
1881				:RHER1 WAS CHECKED TO HAVE ZERO
1882				:PAT (BIT #4 RHCS2) WAS SET
1883				:TO INVERT PARITY CHECKING
1884				:RHER1 WAS READ BUT DID NOT
1885				:CONTAIN WHAT IS IN RHER1 GOOD
1886	001560	066606	DH33	:PC
1887				:TEST NO
1888				:RHER1 GOOD
1889				:RHER1 BAD
1890	001562	067626	DT6	:SERRPC,TSTNM,\$GDDAT,\$BDDAT
1891	001564	070005	DF6	:0,0,0,0
1892				
1893				:ITEM 35
1894	001566	055263	EM35	:RH CLEAR WAS GIVEN
1895				:RHER1 WAS CHECKED
1896				:PAT (BIT #4 RHCS2) WAS SET
1897				:AND RHER1 WAS READ
1898				:RHCS1 SHOULD HAVE RDY
1899				:SC AND MCPE SET
1900	001570	066270	DH11	:PC
1901				:TEST NO
1902				:RHCS1 GOOD
1903				:RHCS1 BAD
1904	001572	067626	DT6	:SERRPC,TSTNM,\$GDDAT,\$BDDAT
1905	001574	070005	DF6	:0,0,0,0
1906				
1907				:ITEM 36
1908	001576	055461	EM36	:RH CLEAR WAS GIVEN
1909				:RHER1 WAS CHECKED
1910				:PAT (RHCS2-BIT #4) WAS SET
1911				:RHER1 WAS READ AND CHECKED
1912				: "1" WAS WRITTEN INTO "TRE"-RHCS1
1913				:ON CHECKING RHCS1
1914				:IT DID NOT CONTAIN SC,MCPE
1915				:AND RDY

1916	001600	066270	DH11	:PC
1917				:TEST NUMBER
1918				:RHCS1 GOOD
1919				:RHCS1 BAD
1920	001602	067626	DT6	:SERRPC,TSTNM,\$GDDAT,\$BDDAT
1921	001604	070005	DF6	:0,0,0,0
1922				
1923			:ITEM 37	
1924	001606	055736	EM37	:RH WAS CLEARED
1925				:RHER1 WAS CHECKED
1926				:PAT (RHCS2-BIT #4) WAS SET
1927				:RHER1 WAS READ
1928				: '1' WAS WRITTEN IN 'TRE' RHCS1
1929				: AN RH CLEAR WAS TO
1930				: GIVE WHAT IS IN 'GOOD'
1931				: BUT GAVE WHAT IS IN 'BAD'
1932	001610	066270	DH11	:PC
1933				:TEST NO
1934				:RHCS1 GOOD
1935				:RHCS1 BAD
1936	001612	067626	DT6	:SERRPC,TSTNM,\$GDDAT,\$BDDAT
1937	001614	070005	DF6	:0,0,0,0
1938				
1939			:ITEM 40	
1940	001616	055736	EM37	
1941	001620	066165	DH6	
1942	001622	067626	DT6	
1943	001624	070005	DF6	
1944				
1945			:ITEM 41	
1946	001626	055736	EM37	
1947	001630	066332	DH12	
1948	001632	067626	DT6	
1949	001634	070005	DF6	
1950				
1951			:ITEM 42	
1952	001636	055736	EM37	
1953	001640	066374	DH13	
1954	001642	067626	DT6	
1955	001644	070005	DF6	
1956				
1957			:ITEM 43	
1958	001646	055736	EM37	
1959	001650	066434	DH14	
1960	001652	067626	DT6	
1961	001654	070005	DF6	
1962				
1963			:ITEM 44	
1964	001656	055736	EM37	
1965	001660	066476	DH15	
1966	001662	067626	DT6	
1967	001664	070005	DF6	
1968				
1969			:ITEM 45	
1970	001666	056240	EM45	:AFTER AN RH CLEAR
1971				: '1' WAS WRITTEN IN THE DISK

1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027

001670 066650

001672 067626
001674 070005

001676 056476

001700 066270

001702 067626
001704 070005

001706 056476
001710 066165
001712 067626
001714 070005

001716 056476
001720 066332
001722 067626
001724 070005

001726 056476
001730 066374
001732 067626
001734 070005

001736 056476
001740 066434
001742 067626
001744 070005

DH45

DT6
DF6

:ITEM 46
EM46

DH11

DT6
DF6

:ITEM 47
EM46
DH6
DT6
DF6

:ITEM 50
EM46
DH12
DT6
DF6

:ITEM 51
EM46
DH13
DT6
DF6

:ITEM 52
EM46
DH14
DT6
DF6

:ITEM 53

```

:ADDRESS REGISTER (IN TAPE
:DRIVES CALLED FRAME COUNT )
:PAT IN RHCS1 WAS SET
:ON READING RHDA (IN TAPE
:DRIVES RHFC ) IT DID NOT
:CONTAIN '1'
:RHDA=RHFC
:PC
:TEST NO
:RHDA GOOD
:RHDA BAD
:SERRPC,TSTNM,$GDDAT,$BDDAT
:0,0,0,0

:AFTER AN RH CLEAR
:'1' WAS WRITTEN IN THE
:DISK ADDRESS REGISTER (IN
:TAPE
:PAT IN RHCS1 WAS SET
:ON READING RHDA (IN TAPE
: ) FOLLOWING
:REGISTER DID NOT CONTAIN
:WHAT IS IN 'GOOD'
:PC
:TEST NO
:RHCS1 GOOD
:RHCS1 BAD
:SERRPC,TSTNM,$GDDAT,$BDDAT
  
```


2028	001746	056476	EM46	
2029	001750	066476	DH15	
2030	001752	067626	DT6	
2031	001754	070005	DF6	
2032				
2033			:ITEM 54	
2034	001756	056777	EM54	: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"
2035				
2036				
2037				
2038				
2039				
2040	001760	066270	DH11	
2041	001762	067626	DT6	
2042	001764	070005	DF6	
2043				
2044			:ITEM 55	
2045	001766	056777	EM54	
2046	001770	066434	DH14	
2047	001772	067626	DT6	
2048	001774	070005	DF6	
2049				
2050			:ITEM 56	
2051	001776	057210	EM56	: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"
2052				
2053				
2054				
2055				
2056				
2057	002000	066434	DH14	
2058	002002	067626	DT6	
2059	002004	070005	DF6	
2060				
2061			:ITEM 57	
2062	002006	057210	EM56	
2063	002010	066270	DH11	
2064	002012	067626	DT6	
2065	002014	070005	DF6	
2066				
2067			:ITEM 60	
2068	002016	057446	EM60	: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"
2069				
2070				
2071				
2072				
2073				
2074	002020	066270	DH11	
2075	002022	067626	DT6	
2076	002024	070005	DF6	
2077				
2078			:ITEM 61	
2079	002026	057446	EM60	
2080	002030	066434	DH14	
2081	002032	067626	DT6	
2082	002034	070005	DF6	
2083				

2084			:ITEM 62		
2085	002036	057657	EM62		:AN RH CLEAR (RHCS2 BIT #5)
2086					:WAS GIVEN
2087					:A17 WAS WRITTEN IN RHCS1
2088					:ALL ZEROS WERE WRITTEN IN RHBAE
2089					:THE FOLLOWING REGISTER DID
2090					:NOT CONTAIN WHAT IS IN 'GOOD'
2091	002040	066434	DH14		
2092	002042	067626	DT6		
2093	002044	070005	DF6		
2094					
2095			:ITEM 63		
2096	002046	057657	EM62		
2097	002050	066270	DH11		
2098	002052	067626	DT6		
2099	002054	070005	DF6		
2100					
2101			:ITEM 64		
2102	002056	060114	EM64		:AN RH CLEAR (RHCS2 BIT #5)
2103					:WAS GIVEN
2104					:IE (RHCS1 BIT #6) WAS SET
2105					:ON READING THE FOLLOWING
2106					:REGISTER IT DID NOT
2107					:CONTAIN WHAT IS IN 'GOOD'
2108	002060	066270	DH11		
2109	002062	067626	DT6		
2110	002064	070005	DF6		
2111					
2112			:ITEM 65		
2113	002066	060114	EM64		
2114	002070	066332	DH12		
2115	002072	067626	DT6		
2116	002074	070005	DF6		
2117					
2118			:ITEM 66		
2119	002076	060114	EM64		:AN RH CLEAR (RHCS2 BIT #5)
2120					:WAS GIVEN
2121					:A16 WAS WRITTEN IN RHCS1
2122					:ALL ZEROS WERE WRITTEN IN RHBAE
2123					:THE FOLLOWING REGISTER DID
2124					:NOT CONTAIN WHAT IS IN 'GOOD'
2125	002100	066332	DH12		
2126	002102	067626	DT6		
2127	002104	070005	DF6		
2128					
2129			:ITEM 67		
2130	002106	060114	EM64		
2131	002110	066270	DH11		
2132	002112	067626	DT6		
2133	002114	070005	DF6		
2134					
2135			:ITEM 70		
2136	002116	060563	EM70		:RH CLEAR WAS GIVEN
2137					:TWO SUCCESSIVE 'GO' (RHCS1 BIT #0)
2138					:WAS GIVEN WITHOUT GIVING
2139					:TIME FOR FIRST 'GO' TO

2140			
2141			
2142			
2143			
2144	002120	066270	DH11
2145	002122	067626	DT6
2146	002124	070005	DF6
2147			
2148			:ITEM 71
2149	002126	060563	EM70
2150	002130	066165	DH6
2151	002132	067626	DT6
2152	002134	070005	DF6
2153			
2154			:ITEM 72
2155	002136	060563	EM70
2156	002140	066332	DH12
2157	002142	067626	DT6
2158	002144	070005	DF6
2159			
2160			:ITEM 73
2161	002146	060563	EM70
2162	002150	066374	DH13
2163	002152	067626	DT6
2164	002154	070005	DF6
2165			
2166			:ITEM 74
2167	002156	060563	EM70
2168	002160	066434	DH14
2169	002162	067626	DT6
2170	002164	070005	DF6
2171			
2172			:ITEM 75
2173	002166	060563	EM70
2174	002170	066476	DH15
2175	002172	067626	DT6
2176	002174	070005	DF6
2177			
2178			:ITEM 76
2179	002176	061037	EM76
2180			
2181			
2182			
2183			
2184			
2185			
2186			
2187	002200	066270	DH11
2188	002202	067626	DT6
2189	002204	070005	DF6
2190			
2191			:ITEM 77
2192	002206	061037	EM76
2193	002210	066165	DH6
2194	002212	067626	DT6
2195	002214	070005	DF6

:COMPLETE
:THE FOLLOWING REGISTER
:DID NOT CONTAIN WHAT IS
:IN 'GOOD'

:RH CLEAR WAS GIVEN A 10. WORD
:WRITE WAS DONE FROM A
:LOCATION TAGED WRFROM
:AND WITH BAI BIT SET
:AT THE END OF THE WRITE
:THE FOLLOWING REGISTER DID
:NOT CONTAIN WHAT IS
:IN GOOD

2196
 2197
 2198 002216 061037
 2199 002220 066332
 2200 002222 067626
 2201 002224 070005
 2202
 2203
 2204 002226 061037
 2205 002230 066374
 2206 002232 067626
 2207 002234 070005
 2208
 2209
 2210 002236 061037
 2211 002240 066434
 2212 002242 067626
 2213 002244 070005
 2214
 2215
 2216 002246 061037
 2217 002250 066476
 2218 002252 067626
 2219 002254 070005
 2220
 2221
 2222 002256 061334
 2223
 2224
 2225
 2226
 2227
 2228
 2229 002260 066710
 2230 002262 067654
 2231 002264 070016
 2232
 2233
 2234 002266 061334
 2235 002270 066760
 2236 002272 067670
 2237 002274 070023
 2238
 2239
 2240 002276 061572
 2241
 2242
 2243
 2244
 2245
 2246
 2247
 2248
 2249
 2250 002300 067030
 2251 002302 067704

:ITEM 100
 EM76
 DH12
 DT6
 DF6

 :ITEM 101
 EM76
 DH13
 DT6
 DF6

 :ITEM 102
 EM76
 DH14
 DT6
 DF6

 :ITEM 103
 EM76
 DH15
 DT6
 DF6

 :ITEM 104
 EM104

 DH104
 DT104
 DF104

 :ITEM 105
 EM104
 DH105
 DT105
 DF105

 :ITEM 106
 EM106

 DH106
 DT106

: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
 :0,0,0,0,0

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

 :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
 :IS IN GOOD
 :PC, TSTNM, IPCK, RHCS1 GOOD, RHCS1 BAD
 :\$ERRPC, TSTNM, IP, \$GDDAT, \$BDDAT

ERROR POINTER TABLE

2252	002304	070030	DF106	:0,0,0,0,0
2253				
2254			:ITEM 107	
2255	002306	061572	EM106	
2256	002310	066760	DH105	:PC,TSTNM,IPCK,RHCS2 GOOD,RHCS2 BAD
2257	002312	067670	DT105	
2258	002314	070023	DF105	
2259				
2260			:ITEM 110	
2261	002316	061572	EM106	
2262	002320	066710	DH104	
2263	002322	067654	DT104	
2264	002324	070016	DF104	
2265				
2266			:ITEM 111	
2267	002326	061572	EM106	
2268	002330	067100	DH111	:PC,TSTNM,IPCK,RHBA GOOD,RHBA BAD
2269	002332	067654	DT104	:SERRPC,TSTNM,IP,\$GDDAT,\$BDDAT
2270	002334	070016	DF104	:0,0,0,0,0
2271				
2272			:ITEM 112	
2273	002336	061572	EM106	
2274	002340	067146	DH112	:PC,TSTNM,IPCK,RHBAE GOOD,RHBAE BAD
2275	002342	067654	DT104	:SERRPC,TSTNM,IP,\$GDDAT,\$BDDAT
2276	002344	070016	DF104	:0,0,0,0,0
2277				
2278			:ITEM 113	
2279	002346	061572	EM106	
2280	002350	067216	DH113	:PC,TSTNM,IPCK,RHWC GOOD,RHWC BAD
2281	002352	067654	DT104	:SERRPC,TSTNM,IP,\$GDDAT,\$BDDAT
2282	002354	070016	DF104	:0,0,0,0,0
2283				
2284			:ITEM 114	
2285	002356	062063	EM114	:RH CLEAR WAS GIVEN AN
2286				:A WRITE CHECK WAS DONE
2287				:THE FOLLOWING REGISTER
2288				:DID NOT CONTAIN WHAT IS
2289				:IN 'GOOD'
2290	002360	066165	DH6	
2291	002362	067626	DT6	
2292	002364	070005	DF6	
2293				
2294			:ITEM 115	
2295	002366	062063	EM114	:RH CLEAR WAS GIVEN AN
2296				:A WRITE CHECK WAS DONE
2297				:THE FOLLOWING REGISTER
2298				:DID NOT CONTAIN WHAT IS
2299				:IN 'GOOD'
2300	002370	066332	DH12	
2301	002372	067626	DT6	
2302	002374	070005	DF6	
2303				
2304			:ITEM 116	
2305	002376	062232	EM116	:RH CLEAR WAS GIVEN AN
2306				:A WRITE CHECK WAS DONE
2307				:THEN ANOTHER RH CLEAR WAS

2308			
2309			
2310			
2311	002400	066270	DH11
2312	002402	067626	DT6
2313	002404	070005	DF6
2314			
2315			:ITEM 117
2316	002406	062232	EM116
2317	002410	066165	DH6
2318	002412	067626	DT6
2319	002414	070005	DF6
2320			
2321			:ITEM 120
2322	002416	062232	EM116
2323	002420	066332	DH12
2324	002422	067626	DT6
2325	002424	070005	DF6
2326			
2327			:ITEM 121
2328	002426	062232	EM116
2329	002430	066374	DH13
2330	002432	067626	DT6
2331	002434	070005	DF6
2332			
2333			:ITEM 122
2334	002436	062232	EM116
2335	002440	066434	DH14
2336	002442	067626	DT6
2337	002444	070005	DF6
2338			
2339			:ITEM 123
2340	002446	062232	EM116
2341	002450	066476	DH15
2342	002452	067626	DT6
2343	002454	070005	DF6
2344			
2345			:ITEM 124
2346	002456	062442	EM124
2347			
2348			
2349			
2350			
2351	002460	066332	DH12
2352	002462	067626	DT6
2353	002464	070005	DF6
2354			
2355			:ITEM 125
2356	002466	062442	EM124
2357	002470	066270	DH11
2358	002472	067626	DT6
2359	002474	070005	DF6
2360			
2361			:ITEM 126
2362	002476	062442	EM124
2363	002500	066165	DH6

: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'

2364 002502 067626
 2365 002504 070005
 2366
 2367
 2368 002506 062442
 2369 002510 066374
 2370 002512 067626
 2371 002514 070005
 2372
 2373
 2374 002516 062442
 2375 002520 066434
 2376 002522 067626
 2377 002524 070005
 2378
 2379
 2380 002526 062442
 2381 002530 066476
 2382 002532 067626
 2383 002534 070005
 2384
 2385 002536 062605
 2386
 2387
 2388
 2389
 2390
 2391 002540 067264
 2392
 2393
 2394
 2395 002542 067626
 2396 002544 070005
 2397
 2398 002546 062763
 2399
 2400
 2401
 2402
 2403
 2404 002550 067264
 2405
 2406
 2407
 2408 002552 067626
 2409 002554 070005
 2410
 2411 002556 063141
 2412
 2413
 2414
 2415
 2416
 2417 002560 067264
 2418 002562 067626
 2419 002564 070005

DT6
 DF6
 :ITEM 127
 EM124
 DH13
 DT6
 DF6
 :ITEM 130
 EM124
 DH14
 DT6
 DF6
 :ITEM 131
 EM124
 DH15
 DT6
 DF6
 :ITEM132
 EM132
 DH132
 DT6
 DF6
 :ITEM133
 EM133
 DH132
 DT6
 DF6
 :ITEM 134
 EM134
 DH132
 DT6
 DF6

:BEFORE DATA TRANSFER
 :COMMAND WAS TO BE GIVEN
 :THE RP DRIVE
 :STATUS REGISTER DID NOT
 :CONTAIN WHAT IS IN GOOD

:PC
 :TEST NO.
 :RHDS1 GOOD
 :RHDS1 BAD

:BEFORE DATA TRANSFER
 :COMMAND WAS TO BE GIVEN
 :THE RS DRIVE
 :STATUS REGISTER DID NOT
 :CONTAIN WHAT IS IN GOOD

:PC
 :TEST NO.
 :RHDS1 GOOD
 :RHDS1 BAD

:BEFORE DATA TRANSFER COMMAND
 :WAS TO BE GIVEN THE
 :MAG TAPE DRIVE STATUS
 :REGISTER DID NOT CONTAIN WHAT
 :IS IN GOOD

2420			:ITEM135		
2421	002566	063331	EM135		: WAS WAITING FOR A BIT TO SET
2422					: BIT IN QUESTION IS IN 'BIT WAITED FOR'
2423					: REGISTER IN QUESTION IN 'REG ADDR'
2424	002570	067326	DH135		: PC
2425					: TEST NO
2426					: PC OF WAT
2427					: BIT
2428					: REG ADDR
2429	002572	067720	DT135		: \$ERRPC, TSTNM, WATIPC, WAITBT, WAITRE
2430	002574	070035	DF135		: 0,0,0,0,0
2431			:ITEM136		
2432	002576	063531	EM136		: WAS WAITING FOR A BIT TO RESET
2433					: BIT IN QUESTION IS IN 'BIT WAITED FOR'
2434					: REGISTER IN QUESTION IN 'REG ADDR'
2435	002600	067326	DH135		: PC
2436					: TEST NO
2437					: PC OF WAT
2438					: BIT
2439					: REG ADDR
2440	002602	067720	DT135		: \$ERRPC, TSTNM, WATIPC, WAITBT, WAITRE
2441	002604	070035	DF135		: 0,0,0,0,0
2442			:ITEM 137		
2443	002606	063733	EM137		: ON WRITING AND READING
2444					: THE REGISTER # IN 'REG. ADDR'
2445					: IT DID NOT CONTAIN
2446					: EXPECTED VALUE.
2447	002610	067406	DH137		: PC
2448					: TEST NO
2449					: REG ADDR
2450					: GOOD DATA
2451					: BAD DATA
2452	002612	067734	DT137		: \$ERRPC, TSTNM, \$BDADR, \$GDDAT, \$BDDAT
2453	002614	070042	DF137		: 0,0,0,0,0
2454					
2455			:ITEM 140		
2456			EM140		
2457	002616	064061	DH106		
2458	002620	067030	DT106		
2459	002622	067704	DF106		
2460	002624	070030			
2461			:ITEM 141		
2462	002626	064313	EM141		: AFTER CLEARING THE RH AND
2463					: WRITING TWO WORD INTO
2464					: RHDB AND READING IT
2465					: BACK TWICE, CAUSED RHCS1 TO
2466					: HAVE WRONG DATA,
2467					: GIVEN IN BAD RHCS1
2468	002630	066270	DH11		: PC
2469					: TEST NUMBER
2470					: RHCS1 GOOD
2471					: RHCS1 BAD
2472	002632	067626	DT6		: \$ERRPC, TSTNM, \$GDDAT, \$BDDAT
2473	002634	070005	DF6		: 0,0,0,0
2474					
2475					

Line	Code	Code	Code	Code	Description
2476					:ITEM 142
2477	002636	064514		EM142	: AFTER CLEARING THE RH AND
2478					: WRITING TWO WORD INTO
2479					: RHDB AND READING IT
2480					: BACK TWICE, CAUSED RHCS3 TO
2481					: HAVE WRONG DATA,
2482					: GIVEN IN BAD RHCS3
2483	002640	066332		DH12	: PC
2484					: TEST NUMBER
2485					: RHCS3 GOOD
2486					: RHCS3 BAD
2487	002642	067626		DT6	: \$ERRPC, TSTNM, \$GDDAT, \$BDDAT
2488	002644	070005		DF6	: 0,0,0,0
2489					
2490					
2491					:ITEM 143
2492	002646	064715		EM143	: AFTER CLEARING THE RH AND
2493					: WRITING TWO WORD INTO
2494					: RHDB AND READING IT
2495					: BACK TWICE, CAUSED RHBA TO
2496					: HAVE WRONG DATA,
2497					: GIVEN IN BAD RHBA
2498	002650	066374		DH13	: PC
2499					: TEST NUMBER
2500					: RHBA GOOD
2501					: RHBA BAD
2502	002652	067626		DT6	: \$ERRPC, TSTNM, \$GDDAT, \$BDDAT
2503	002654	070005		DF6	: 0,0,0,0
2504					
2505					
2506					:ITEM 144
2507	002656	065115		EM144	: AFTER CLEARING THE RH AND
2508					: WRITING TWO WORD INTO
2509					: RHDB AND READING IT
2510					: BACK TWICE, CAUSED RHBAE TO
2511					: HAVE WRONG DATA,
2512					: GIVEN IN BAD RHBAE
2513	002660	066434		DH14	: PC
2514					: TEST NUMBER
2515					: RHBAE GOOD
2516					: RHBAE BAD
2517	002662	067626		DT6	: \$ERRPC, TSTNM, \$GDDAT, \$BDDAT
2518	002664	070005		DF6	: 0,0,0,0
2519					
2520					
2521					:ITEM 145
2522	002666	065316		EM145	: AFTER CLEARING THE RH AND
2523					: WRITING TWO WORD INTO
2524					: RHDB AND READING IT
2525					: BACK TWICE, CAUSED RHWC TO
2526					: HAVE WRONG DATA,
2527					: GIVEN IN BAD RHWC
2528	002670	066476		DH15	: PC
2529					: TEST NUMBER
2530					: RHWC GOOD
2531					: RHWC BAD

2532	002672	067626	DT6	;\$ERRPC,TSTNM,\$GDDAT,\$BDDAT
2533	002674	070005	DF6	:0,0,0,0
2534			:ITEM146	
2535	002676	065516	EM146	:A DEVICE BASE ADDRESS DID NOT
2536				:TIME OUT BUT CORRESPONDING
2537				:VECTOR ADDRESS DID TIME OUT
2538	002700	067460	DH146	:PC
2539				:BASE
2540				:VECTOR
2541	002702	067750	DT146	;\$ERRPC,TESTDV,TESTVC,0
2542	002704	070047	DF146	:0,0,0
2543				
2544			:ITEM147	
2545	002706	065710	EM147	:A DEVICE ADDRESS DID NOT
2546				:TIME OUT BUT NO UNITS HAD
2547				:APPROPRIATE DRIVE TYPE
2548	002710	067507	DH147	:PC
2549				:BASE ADDRESS
2550	002712	067760	DT147	;\$ERRPC,TESTDV
2551	002714	070052	DF147	
2552				
2553			:ITEM150	
2554	002716	066040	EM150	:THE SILO DID NOT SIZE AS EXPECTED
2555	002720	067534	DH150	:PC
2556				:TEST NUMBER
2557				:SILO SIZE
2558	002722	067766	DT150	;\$ERRPC,TSTNM,SILOSZ
2559	002724	070054	DF150	:0,0,1

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
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

002726 000254

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

000001
000002
000004
000010
000100
002000
004000
010000
020000
040000
100000

: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 #8)
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 BOUNDRY (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)

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
2643
2644
2645
2646
2647
2648
2649
2650
2651
2652
2653
2654
2655
2656
2657
2658
2659
2660
2661
2662
2663
2664
2665
2666
2667
2668
2669
2670
2671

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)

DFF5= 1 :DRIVE FORWARD 5"/SEC. (BIT #0)
BOT= 2 :BEGINING OF TAPE (BIT #1)
DFF20= 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)

2672	020000	OPI= 20000	:OPERATION INCOMPLETE (BIT #13)
2673	040000	UNS= 40000	:DRIVE UNSAFE (BIT #14)
2674	100000	DCK= 100000	:DATA CHECK ERROR (BIT 15)
2675			
2676		:MAINTAINABILITY REGISTER (RHMP) (#03)	
2677			
2678	000001	DMD= 1	:DIAGINOSTIC MODE (BIT #0)
2679	000002	MCLK= 2	:MAINTAINABILITY CLOCK (BIT #1)
2680	000004	MINX= 4	:MAINTAINABILITY INDEX (BIT #2)
2681	000010	MSTCK= 10	:MAINTAINABILITY SECTOR CLOCK (BIT #3)
2682	000020	MRD= 20	:MAINTAINABILITY READ (BIT #4)
2683	000040	MWR= 40	:MAINTAINABILITY WRITE (BIT #5)
2684	001000	DTSY= 1000	:MAINTAINABILITY SYNC DETECTED (BIT #9)
2685			
2686		:ATTENTION SUMMARY PSEUDO-REGISTER (RHAS) (#04)	
2687			
2688	000001	AT0= 1	:DEVICE 0 (BIT #0)
2689	000002	AT1= 2	:DEVICE 1 (BIT #1)
2690	000004	AT2= 4	:DEVICE 2 (BIT #2)
2691	000010	AT3= 10	:DEVICE 3 (BIT #3)
2692	000020	AT4= 20	:DEVICE 4 (BIT #4)
2693	000040	AT5= 40	:DEVICE 5 (BIT #5)
2694	000100	AT6= 100	:DEVICE 6 (BIT #6)
2695	000200	AT7= 200	:DEVICE 7 (BIT #7)
2696			
2697		:DESIRED SECTOR/TRACK ADDRESS REGISTER (RHDST) (#1)	
2698		:EACH BIT IS CALLED BY BIT NUMBER	
2699		:DRIVE TYPE REGISTER (RHDT) (#06)	
2700		:EACH BIT IS CALLED BY BIT NUMBER	
2701		:LOOK-AHEAD REGISTER (RHLA) (#07)	
2702			
2703	000001	EXT1= 1	:EXTENSION 1 (BIT #0)
2704	000002	EXT2= 2	:EXTENSION 2 (BIT #1)
2705	000004	EXT4= 4	:EXTENSION 3 (BIT #2)
2706	000010	EXT10= 10	:EXTENSION 4 (BIT #3)
2707	000020	EXT20= 20	:EXTENSION 5 (BIT #4)
2708	000040	EXT40= 40	:EXTENSION 6 (BIT #5)
2709	000100	SC1= 100	:SECTOR COUNT FIELD 0 (BIT #6)
2710	000200	SC2= 200	:SECTOR COUNT FIELD 1 (BIT #7)
2711	000400	SC4= 400	:SECTOR COUNT FIELD 2 (BIT #8)
2712	001000	SC10= 1000	:SECTOR COUNT FIELD 3 (BIT #9)
2713	002000	SC20= 2000	:SECTOR COUNT FIELD 4 (BIT #10)
2714	004000	TRK1= 4000	:TRACK FIELD 1 (BIT #11)
2715	010000	TRK2= 10000	:TRACK FIELD 2 (BIT #12)
2716	020000	TRK4= 20000	:TRACK FIELD 3 (BIT #13)
2717	040000	TRK10= 40000	:TRACK FIELD 4 (BIT #14)
2718	100000	TRK20= 100000	:TRACK FIELD 5 (BIT #15)
2719			
2720		:ERROR REGISTER #2 (RHER2) (#10)	
2721			
2722	000001	WCU= 1	:WRITE CURRENT UNSAFE (BIT #0)
2723	000002	CSF= 2	:CURRENT SINK FAILURE (BIT #1)
2724	000004	WSU= 4	:WRITE SELECT UNSAFE (BIT #2)
2725	000010	CSU= 10	:CURRENT SWITCH UNSAFE (BIT #3)
2726	000020	MSE= 20	:MOTOR SEQUENCE ERROR (BIT #4)
2727	000040	TDF= 40	:TRANSITIONS DETECTOR FAILURE (BIT #5)

2728	000100	TUF= 100	:TRANSITIONS UNSAFE (BIT #6)
2729	000200	FEN= 200	:FAILSAFE ENABLED (BIT #7)
2730	000400	WRU= 400	:WRITE READY UNSAFE (BIT #8)
2731	001000	MHS= 1000	:MULTIPLE HEAD SELECT (BIT #9)
2732	002000	NHS= 2000	:NO HEAD SELECTION (BIT #10)
2733	004000	IXE= 4000	:INDEX ERROR (BIT #11)
2734	010000	VU30= 10000	:30VOLT UNSAFE (BIT #12)
2735	020000	PLU= 20000	:PLO UNSAFE (BIT #13)
2736	100000	ACU= 100000	:ACUNSAFE (BIT #15)
2737			
2738		:OFFSET REGISTER (RHOF) (#11)	
2739			
2740	000001	OF25= 1	:OFFSET 25 MICRO INCHES (BIT #0)
2741	000002	OF50= 2	:OFFSET 50 MICRO INCHES (BIT #1)
2742	000004	OF100= 4	:OFFSET 100 MICRO INCHES (BIT #2)
2743	000010	OF200= 10	:OFFSET 200 MICRO INCHES (BIT #3)
2744	000020	OF400= 20	:OFFSET 400 MICRO INCHES (BIT #4)
2745	000040	OF800= 40	:OFFSET 800 MICRO INCHES (BIT #5)
2746			
2747	000200	OFREV= 200	:OFFSET NEGATIVE (REVERSE) (BIT #5)
2748	002000	HCI= 2000	:HEADER COMPARE INHIBIT (BIT #10)
2749	004000	ECI= 4000	:ERROR CORRECTION CODE INHIBIT (BIT #11)
2750	010000	FMT22= 10000	:FORMAT BIT (BIT #12)
2751			
2752		:TAPE CONTROL REGISTER (RHTC)	
2753			
2754	000001	S1= 1	:SLAVE NUMBER
2755	000002	S2= 2	:SLAVE NUMBER
2756	000004	S4= 4	:SLAVE NUMBER
2757	000010	EPAR= 10	:EVEN PARITY
2758	000020	FMT1= 20	:FORMAT
2759	000040	FMT2= 40	:FORMAT
2760	000100	FMT4= 100	:FORMAT
2761	000200	FMT8= 200	:FORMAT
2762	000400	DEN1= 400	:DENSITY
2763	001000	DEN2= 1000	:DENSITY
2764	002000	DEN4= 2000	:DENSITY
2765	001400	BP18= 1400	:800 BPI
2766	000300	NML= 300	:NORMAL - 2 FRAMES PER WORD
2767			
2768			
2769			
2770			
2771			
2772		:DRIVE TYPE REGISTER	
2773			
2774	002000	SLVPR= 2000	:SLAVE PRESENT (BIT #10)
2775	010000	CH7= 10000	:CHANNEL 7 (BIT #12)
2776			
2777		:DESIRED CYLINDER ADDRESS (RHCA) (#12)	
2778		:EACH BIT IS CALLED BY BIT NUMBER.	
2779			
2780			
2781			
2782			
2783		:CURRENT CYLINDER ADDRESS (RHCC) (#13)	

2784
2785
2786
2787
2788
2789
2790
2791
2792
2793
2794
2795
2796
2797
2798
2799
2800
2801
2802
2803
2804
2805
2806
2807
2808
2809
2810
2811
2812
2813
2814
2815
2816
2817
2818
2819
2820
2821

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

::.....

2822
2823
2824
2825
2826
2827
2828
2829
2830
2831
2832
2833
2834
2835
2836
2837
2838
2839
2840
2841
2842
2843
2844
2845
2846
2847
2848
2849
2850
2851
2852
2853
2854
2855
2856
2857
2858
2859
2860
2861
2862
2863
2864
2865

000000
000002
000004
000006
000006
000010
000012
000014
000016
000020
000022
000024
000026
000030
000032
000032
000034
000036
000040
000042
000044
000046

003000
003000 000000
003002 000422

004046 172040
004050 176700
004052 172440
004054 176300
004056 000000

CS1= 0
WC= 2
BA= 4
FC= 6
DA= 6
CS2= 10
DS= 12
ER1= 14
AS= 16
LA= 20
DB= 22
MR= 24
DT= 26
SN= 30
OF= 32
TC= 32
DC= 34
CC= 36
ER2= 40
ER3= 42
EC1= 44
EC2= 46

:BUFFERS
.=3000
BUFEW:
BFEVEN: 0
BUFOW:
BFODD: .BLKW 274.
:STARTING ADDRESS OF REGISTERS

RSCS1: 172040
RPCS1: 176700
TMCS1: 172440
MIXCS1: 176300
PRESENT: 0

:CONTROL STATUS 1
:WORD COUNT
:BUS ADDRESS
:FRAME COUNT
:DESIRED SECTOR/TRACK
:CONTROL STATUS 2
:DRIVE STATUS
:ERROR 1
:ATTENTION SUMMERY
:LOOK AHEAD
:DATA BUFFER
:MAINTENANCE REGISTER
:DRIVE TYPE
:SERIAL NUMBER
:OFFSET
:TAPE CONTROL
:DESIRED CYLINDER
:CURRENT CYLINDER
:ERROR 2
:ERROR 3
:ECC 1
:ECC 2

:BUFFER

:RS ALONG
:RP ALONE
:TM ALONE
:MIXED SYSTEM

.SBTTL REGISTER ADDRESSES

2866
2867
2868
2869
2870
2871
2872
2873
2874
2875
2876
2877
2878
2879
2880
2881
2882 004060 000000
2883 004062 000000
2884 004064 000000
2885 004066
2886 004066
2887 004066 000000
2888 004070 000000
2889 004072 000000
2890 004074 000000
2891 004076 000000
2892 004100
2893 004100 000000
2894 004102 000000
2895 004104 000000
2896 004106 000000
2897 004110 000000
2898 004112
2899 004112 000000
2900 004114 000000
2901 004116 000000
2902 004120 000000
2903 004122 000000
2904 004124 000000
2905 004126 000000
2906 004130 000000
2907 004132 000000
2908
2909
2910

: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) RP04,5,6
:4) TM02
:THE CONTENTS OF RHCS1 WHICH IS THE BASE ADDRESS IS
:172040 FOR RS03/04
:176700 FOR RP04,5,6
:172440 FOR TM02
: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: :FRAME COUNT
RHDA: :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: :CHECK CHARACTER
RHLA: 0 :LOOK-AHEAD
RHDB: 0 :DATA BUFFER
RHMR: 0 :MAINTAINABILITY
RHDT: 0 :DRIVE TYPE
RHSN: 0 :SERIAL NUMBER
RHTC: :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

```

2911
2912
2913           ;FUNCTION EQUATES
2914
2915           ;TABLE OF FUNCTIONS FOR RHCS1 THEN 'GO' BIT HAS TO BE SET
2916 004134     FUTABL:
2917 004134 000000 NOPERA: 0           ;NO OPERATION

```


2918	004136	000002	UNLOAD: 2	:UNLOAD (STAND BY)
2919	004140	000006	RECALI: 6	:RECALIBRATE
2920	004142	000010	DCLEAR: 10	:DRIVE CLEAR
2921	004144	000012	RELEAS: 12	:RELEASE (DUAL-PORT OPERATION)
2922	004146	000030	SERCH: 30	:SEARCH COMMAND
2923	004150	000050	WRCHK: 50	:WRITE CHECK DATA
2924	004152	000052	WRCHDT: 52	:WRITE CHECK HEADER AND DATA
2925	004154	000060	WRIDAT: 60	:WRITE DATA
2926	004156	000062	WRIFOR: 62	:WRITE HEADER AND DATA (FORMAT)
2927	004160	000070	READAT: 70	:READ DATA
2928	004162	000072	REFOR: 72	:READ HEADER AND DATA
2929	004164	000004	SEECOM: 4	:SEEK COMMAND
2930	004166	000014	OFSETC: 14	:OFFSET COMMAND
2931	004170	000016	RETCL: 16	:RETURN TO CENTERLINE
2932	004172	000022	PKACK: 22	:PACK ACKNOWLEDGE
2933	004174	000020	READIN: 20	:READ IN
2934	004176	000006	REWIND: 6	:REWIND
2935	004200	000076	REVRED: 76	:REVERSE READ
2936	004202	000030	SPACFD: 30	:SPACE FORWARD
2937	004204	000066	REVWRT: 66	:REVERSE WRITE
2938	004206	000000	ILLEGL: .WORD 0	:COMPUTED ILLEGAL FUNCTION
2939				
2940			:DATA BUFFER FOR READ WRITE	
2941	004210	000412	WRFROM: .BLKW 266.	:WRITE FROM THIS BUFFER
2942	005234	000412	REINTO: .BLKW 266.	:READ INTO THIS BUFFER
2943	006260	000000	COMND: 0	:STORE COMMAND FOR COMMAND ROUTINE
2944	006262	000000	COMAND: 0	:STORE COMMAND
2945	006264	000000	NOGO: 0	:IF ZERO GO IS TO BE GIVEN IN COMMAND ROUTINE
2946				:IF ONES GO IS NOT TO BE GIVEN IN COMMAND ROUTINE
2947				:USED IN PGE TEST
2948	006266	000000	WATO: 0	:IF ZERO WAIT FOR BIT TO SET
2949				:IF ONES WAIT FOR BIT TO RESET
2950				:USED IN WAIT TRAP
2951	006270	000000	WRTBIT: 0	:BITS NOT WRITTEN INTO
2952	006272	000000	SETBIT: 0	:BITS ALWAYS SET
2953	006274	000000	CLRBIT: 0	:BITS ALWAYS CLEARED
2954				
2955				
2956				
2957			:RESERVED LOCATIONS	
2958	006276	000000	TSTNM: 0	:TEST NUMBER
2959	006300	000000	SLAVE: 0	:KEEP SLAVE NO.
2960	006302	000000	IP: 0	:IPCK NUMBER FOR ERROR PRINTOUT
2961	006304	000000	SILONM: 0	:SILO WORD NUMBER FOR ERROR PRINT OUT
2962	006306	000000	WAITPC: 0	:WAIT TRAP PC
2963	006310	000000	WAITRE: 0	:WAITING FOR REGISTER ADDRESS IN WAIT TRAP
2964	006312	000000	WAITBT: 0	:WAITING FOR BIT IN WAIT TRAP
2965				
2966			:TABLE FOR ATTENTION BITS	
2967			:ATTENTION TABLE	
2968	006314	001	002	004
2969	006317	010	020	040
2970	006322	100	200	
2971				
2972				
2973				

CERHAEO MACY11 30A(1052) 02-MAY-79 14:35 PAGE 69
 CERHAE.P11 02-MAY-79 14:08 REGISTER ADDRESSES

SEQ 0067

```

2974          ;RESERVED LOCATIONS FOR UNIT SELECT
2975 006324 172040 BASEAD: 172040          ;RS BASE ADDRESS
2976 006326 176700 BASPAD: 176700          ;RP BASE ADDRESS
2977 006330 172440 BASTAD: 172440          ;TU BASE ADDRESS
2978 006332 176300 BASEMAD:          176300 ;MIXED SYSTEM BASE ADDRESS
2979
2980
2981 006334 000000 BASEVC: 0          ;RS VECTOR
2982 006336 000254 BASP: 254          ;RP VECTOR
2983 006340 000001 BAST: 1          ;TU VECTOR
2984 006342 000002 BASM: 2          ;MIXED VECTOR
2985
2986          ;TABLE FOR GIVEN BASE ADDRESSES
2987 006344 000010 BSGIVA: .BLKW 8.
2988
2989
2990          ;TABLE FOR GIVEN VECTOR
2991 006364 000010 BSGIVV: .BLKW 8.
2992
2993
2994 006404 000004 NMRHS: 4          ;NUMBER OF RH
2995 006406 000000 BASINX: 0          ;INDEX FOR BASE
2996 006410 000000 WORKBS: 0          ;WORKING BASE
2997 006412 000000 WORKNM: 0          ;WORKING NUMBER FOR RH
2998 006414 000000 WORKVC: 0          ;WORKING VECTOR FOR RH
2999 006416 000000 FORBAE: 0          ;TOTAL NO. OF REG. FOR BAE CALCULATION
3000 006420 000000 LOPCT: 0          ;LOOP COUNT FOR 200 START
3001 006422 000000 LOPCT1: 0          ;LOOP COUNT FOR 210 START
3002 006424 000000 USEVEC: 0          ;USE VECTOR
3003 006426 000000 SILOSZ: 0          ;SILO SIZE
3004 006430 000000 WDCT1: 0          ;USED TO SET RHW
3005 006432 000000 WDCT2: 0          ;
3006 006434 000000 WDCT3: 0          ;
3007 006436 000000 TSTPGM: 0          ;=1, ALLOW TESTING PROGRAMMABLE DRIVES
3008
3009
3010
3011 006440 000000 GIVE: 0          ;IF ONES OPERATOR WILL GIVE ADDRESS
3012 006442 000000 UNIT: 0          ;UNIT UNDER TEST
3013 006444 000000 ST200: 0          ;ALL ONES INDICATE STARTING FROM 200
3014          ;TESTING TABLE
3015 006446 000010 TSRHNO: .BLKW 8.          ;RH NO TO BE TESTED IN ORDER OF TEST
3016
3017 006466 000010 TSDEVICE: .BLKW 8.          ;DEVICE TO BE TESTED IN ORDER OF TEST
3018          ;1=RS03,3/L,3/LA 2=RS04,4/L 4=RP04,5,6 SINGLE PORT
3019          ;10=RP04,5,6 DUAL PORT 20=TM02
3020
3021
3022 006506 000010 TSUNIT: .BLKW 8.          ;UNIT NO. TO BE TESTED IN ORDER OF TEST
3023
3024
3025 006526 000010 TSSLAV: .BLKW 8.          ;SLAVE NO TO BE TESTED IN ORDER OF TEST
3026
3027
3028 006546 000000 TSTOTL: 0          ;TOTAL NO. OF UNITS TO BE TESTED
3029 006550 000000 TSINDX: 0          ;INDEX TO ONE UNDER TEST

```


3030					
3031					
3032	006552	000010	TSVEC: .BLKW 8.		;VECTOR ADDRESS IN ORDER OF TEST
3033					
3034					
3035	006572	000010	TSBAE: .BLKW 8.		;RHBAE ADDRESS IN ORDER OF TEST
3036					
3037					
3038	006612	000010	TSCS3: .BLKW 8.		;RHCS3 ADDRESS IN ORDER OF TEST
3039					
3040					
3041	006632	000010	TSCOMD: .BLKW 8.		;COMMAND ADDRESS IN ORDER OF TEST
3042					
3043					
3044	006652	000010	TSBASE: .BLKW 8.		;BASE ADDRESS IN ORDER OF TEST
3045					
3046					
3047					
3048					
3049					
3050					
3051	006672	000000	ERFLG\$: 0		;ERROR FLAG
3052	006674	000000	FIRST: 0		;IF ZERO WILL TYPE HEADER
3053					;IF ONES WILL NOT TYPE HEADER
3054					
3055					
3056					
3057	006676	000000	ATTENT: 0		;ATTENTION BIT FOR PRESENT UNIT
3058	006700	000000	TOTALAT: 0		;TATAL ATTENTION BITS
3059					
3060	006702	000000	TMP0: .WORD 0		;TEMP STORAGE
3061	006704	000000	TMP1: .WORD 0		
3062	006706	000000	TMP4: .WORD 0		;TEMP STORAGE
3063					;PERMANENT TABLE
3064	006710	172040	PERRS: 172040		;RS BASE ADDRESS
3065	006712	176700	PERRP: 176700		;RP BASE ADDRESS
3066	006714	172440	PERTU: 172440		;TU BASE ADDRESS
3067	006716	176300	PERMX: 176300		;MIXED BASE ADDRESS OR RM03 BASE ADDRESS
3068					
3069	006720	000204	PERRSV: 204		;RS VECTOR ADDRESS
3070	006722	000254	PERRPV: 254		;RP VECTOR ADDRESS
3071	006724	000224	PERTUV: 224		;TU VECTOR ADDRESS
3072	006726	000150	PERMXV: 150		;MIXED VECTOR ADDRESS OR RM03 BASE ADDRESS
3073					
3074	006730	000004	PERNUM: 4		;NUMBER OF RH
3075					
3076					;WORKING TABLE
3077					
3078	006732	000004	DEVPNT: .BLKW 4		;BASE ADDRESS TO BE TESTED
3079	006742	000004	VECPNT: .BLKW 4		;VECTOR ADDRESS TO BE TESTED
3080	006752	000000	LTRY: 0		;NO. OF UNITS TO BE TESTED
3081	006754	000000	TFOUND: 0		;TWICE NUMBER OF RH FOUND
3082	006756	000000	TESTDV: 0		;STORES BASE ADDRESS OF TEST DEVICE
3083	006760	000000	TESTVC: 0		;STORES VECTOR OF TEST DEVICE
3084					;THE ABOVE TWO IS BEFORE ANY
3085					;DEVICE IS FOUND.

```
3086 :TABLE FOR DRIVE TYPES
3087 : 20024= SINGLE PORT RM03, 24024= DUAL PORT RM03
3088 : 0=RS03, 1=RS03/L, 2=RS04, 3=RS04/L, 4=RS03/LA
3089 : 20020=SINGLE PORT RP04, 24020=DUAL PORT RP04
3090 : 20021=SINGLE PORT RP05, 24021=DUAL PORT RP05
3091 : 20022=SINGLE PORT RP06, 24022=DUAL PORT RP06
3092 : 142010=TU16
3093
3094 006762 020024 024024 000000 TYPNT: .WORD 20024,24024,0,1,2,3,4,20020,24020,20021,24021,20022,24022,142010
3095 006770 000001 000002 000003
3096 006776 000004 020020 024020
3097 007004 020021 024021 020022
3098 007012 024022 142010
3099 007016 000000 POINTT: 0 ;POITER TO ABOVE TABLE
3100 007020 000016 NTYPNT: 16 ;NUMBER OF ABOVE TYPES
3101
3102 007022 000000 TEMPLV: 0 ;TEMPORARY SLAVE COUNTER
3103
3104 :TABLE OF DATA FOR DEVICES FOUND
3105 007024 000006 FDEVIC: .BLKW 6 ;DEVICE FOUND, 1=TU, 2=RP DUAL
3106 ;3=RP SINGLE, 4=RS04, 5=RS03
3107 ;6=RM03 DUAL,7=RM03 SINGLE
3108 007040 000006 FUNITN: .BLKW 6 ;UNIT NUMBER FOUND
3109 007054 000006 FTYPE: .BLKW 6 ;DDRIVE TYPE FOUND
3110 007070 000006 FVECTR: .BLKW 6 ;VECTOR FOUND
3111 007104 000006 FBASEA: .BLKW 6 ;BASE ADDRESS FOUND
3112 007120 000006 FRHNM: .BLKW 6 ;RH NUMBER FOUND
3113 007134 000006 FSLAVE: .BLKW 6 ;TU16 SLAVE NUMBER FOUND
3114 007150 000006 FDRIVR: .BLKW 6 ;DRIVER ADDRESS FOUND
3115 007164 000006 FBAE: .BLKW 6 ;BAE ADDRESS FOUND
3116 007200 000006 FCS3: .BLKW 6 ;CS3 ADDRESS FOUND
3117
3118 007214 000000 TSTUNT: 0 ;TOTAL NUMBER OF RH FOUND AT
3119 ;END OF SIZE
3120 007216 000000 WORUNT: 0 ;SAME AS ABOVE BUT TO BE
3121 ;DECREASED AT END PROGRAM
3122
3123 007220 000000 VECTOR: 0 ;VECTOR UNDER TEST
3124 007222 177740 LERADD: 177740 ;LOW ERROR ADDRESS REG.
3125 007224 177742 HERADD: 177742 ;HIGH ERROR ADDRESS REG
3126 007226 177744 MEMERR: 177744 ;MEMORY SYSTEM ADDRESS REG
3127 000114 PARE70=114 ;PARITY ERROR VECTOR FOR 70
```



```

3128 .SBTTL REGISTER TEST
3129 007230 012737 177777 006444 BEGIN: MOV #-1,ST200
3130 007236 005037 006436 CLR TSTPGM ;DISABLE PROGRAMMABLE DRIVES
3131 007242 000414 BR START
3132 007244 005037 006444 BEGIN2: CLR ST200
3133 007250 012737 000001 006436 MOV #1, TSTPGM ;ENABLE PROGRAMMABLE DRIVES
3134 007256 000406 BR START
3135 007260 012737 000001 006436 BEGIN3: MOV #1, TSTPGM ;ENABLE PROGRAMMABLE DRIVES
3136 007266 012737 177777 006444 MOV #-1, ST200 ;LIKE A 200 START
3137
3138 007274 005037 007024 START: CLR FDEVIC ; CLEAR DEVICE FOUND INDICATOR
3139 .SBTTL INITIALIZE THE COMMON TAGS
3140 ::CLEAR THE COMMON TAGS ($CMTAG) AREA.
3141 007300 012706 001100 MOV #$CMTAG,R6 ;;FIRST LOCATION TO BE CLEARED
3142 007304 005026 CLR (R6)+ ;;CLEAR MEMORY LOCATION
3143 007306 022706 001140 CMP #SWR,R6 ;;DONE?
3144 007312 001374 BNE -.6 ;;LOOP BACK IF NO
3145 007314 012706 001000 MOV #STACK,SP ;;SETUP THE STACK POINTER
3146 ::INITIALIZE A FEW VECTORS
3147 007320 012737 044044 000020 MOV #SCOPE,@IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
3148 007326 012737 000340 000022 MOV #340,@IOTVEC+2 ;;LEVEL 7
3149 007334 012737 045670 000030 MOV #ERROR,@EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
3150 007342 012737 000340 000032 MOV #340,@EMTVEC+2 ;;LEVEL 7
3151 007350 012737 047374 000034 MOV #STRAP,@TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
3152 007356 012737 000340 000036 MOV #340,@TRAPVEC+2;LEVEL 7
3153 007364 012737 047454 000024 MOV #SPWRDN,@PWRVEC ;;POWER FAILURE VECTOR
3154 007372 012737 000340 000026 MOV #340,@PWRVEC+2 ;;LEVEL 7
3155 007400 005037 001212 CLR $TIMES ;;INITIALIZE NUMBER OF ITERATIONS
3156 007404 005037 001214 CLR $ESCAPE ;;CLEAR THE ESCAPE ON ERROR ADDRESS
3157 007410 112737 000001 001115 MOVB #1,$ERMAX ;;ALLOW ONE ERROR PER TEST
3158 007416 012737 007416 001106 MOV #,$SLPADR ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
3159 007424 012737 007424 001110 MOV #,$SLPERR ;;SETUP THE ERROR LOOP ADDRESS
3160 ::SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
3161 ::EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
3162 007432 013746 000004 MOV @ERRVEC,-(SP) ;;SAVE ERROR VECTOR
3163 007436 012737 007472 000004 MOV #64$,@ERRVEC ;;SET UP ERROR VECTOR
3164 007444 012737 177570 001140 MOV #DSWR,SWR ;;SETUP FOR A HARDWARE SWICH REGISTER
3165 007452 012737 177570 001142 MOV #DDISP,DISPLAY ;;AND A HARDWARE DISPLAY REGISTER
3166 007460 022777 177777 171452 CMP #-1,@SWR ;;TRY TO REFERENCE HARDWARE SWR
3167 007466 001012 BNE 66$ ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
3168 ;;AND THE HARDWARE SWR IS NOT = -1
3169 007470 000403 BR 65$ ;;BRANCH IF NO TIMEOUT
3170 007472 012716 007500 64$: MOV #65$, (SP) ;;SET UP FOR TRAP RETURN
3171 007476 000002 RTI
3172 007500 012737 000176 001140 65$: MOV #SWREG,SWR ;;POINT TO SOFTWARE SWR
3173 007506 012737 000174 001142 MOV #DISPREG,DISPLAY
3174 007514 012637 000004 66$: MOV (SP)+,@ERRVEC ;;RESTORE ERROR VECTOR
3175
3176
3177
3178 007520 012737 000000 177776 MOV #0,PS ;SET PROCESSOR STATUS TO 0
3179 007526 012737 000200 000036 MOV #200,@TRAPVEC+2 ;TRAP PRIORITY = 4
3180 007534 013700 002726 MOV @RPTVEC,R0 ;GET RP VECTOR ADDRESS
3181 007540 012720 044002 MOV #RPTVEC,(R0)+ ;THIS IS FOR UNTIMELY INTERRUPTS
3182 007544 012710 000340 MOV #340,(R0) ;RPO4 INTERRUPT SERVICE ROUTINE
3183 ;PRIORITY = 7
    
```

```

3184 007550 004737 045010 JSR PC,@#STKINT ;INITILIZE THE TK
3185 007554 005037 046024 CLR PRITEM ;CLEAR FOR ERROR MESSAGE PRINTOUT
3186 007560 005737 006674 TST @#FIRST ;IS THIS FIRST TIME ROUND
3187 007564 001001 BNE 1$ ;BRANCH IF NOT
3188 007566 0C0402 BR 2$
3189 007570 000137 007654 1$: JMP @#SND2
3190 007574 023737 000042 000046 2$: CMP @#42,@#46 ;ARE WE IN QV OR AUTO ACCEPT MODE?
3191 007602 001421 BEQ SND1 ;IF YES, SKIP HEADER
3192 007604 104401 007612 TYPE ,68$ ;;TYPE ASCIZ STRING
3193 007610 000415 BR 67$ ;;GET OVER THE ASCIZ
3194 ;;68$: .ASCIZ <15><12>/CERHAEO RH70 CTRLR DIAG/
3195 67$:
3196 007644 BR SND2
3197 007646 000403 MOV #1, TSTPGM ;ENABLE PROGRAMMABLE DRIVES
3198 007654 012737 000001 006436 SND1: MOV #-1,@#FIRST ;NEXT TIME DO NOT GIVE HEADER
3199 007662 012737 177777 006674 SND2: MOV #PARITY,@#PARE70 ;PARITY VECTOR
3200 007670 012737 040646 000114 MOV #340,@#PARE70+2 ;PRIORITY 7
3201 007676 012737 000340 000116 MOV #TIEOUT,@#ERRVEC ;TIMEOUT VECTOR
3202 007704 012737 040510 000004 MOV #340,@#ERRVEC+2 ;PRIORITY 7
3203 007712 005737 006436 TST TSTPGM ;PROGRAMMABLE DRIVES ENABLED?
3204 007716 001431 BEQ 6$ ;BRANCH IF NO
3205 007720 104401 007726 TYPE ,65$ ;;TYPE ASCIZ STRING
3206 007724 000426 BR 64$ ;;GET OVER THE ASCIZ
3207 ;;65$: .ASCIZ <15><12>/WARNING: PROGRAMMABLE DRIVES MAY BE USED/
3208 010002 64$:
3209 010002 6$:

```

3210
3211
3212
3213
3214
3215
3216
3217
3218
3219
3220
3221
3222
3223
3224
3225
3226
3227
3228
3229
3230
3231
3232
3233
3234
3235
3236
3237
3238
3239

```

*****
*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
*

```



```

3296 010306          72$:
3297 010306 013746 006704      MOV     TMP1,-(SP)
3298 010312 104405              TYPDS
3299 010314 104401 000200      TYPE     ,CRLF
3300 010320 104410              RDOCT
3301 010322 012660 006742      MOV     (SP)+,VECPNT(R0) ;FILL WORKING TABLE WITH VECTOR ADDRESS
3302 010326 005720              TST     (R0)+ ;ADD 2 TO R0
3303 010330 005237 006704      INC     TMP1 ;GET NEXT RH NUMBER
3304 010334 005337 006702      DEC     TMP0 ;COUNT DOWN RH TO BE TESTED
3305 010340 001302              BNE     3$ ;BRANCH IF ALL NOT DONE
3306
3307                      ;R0 WILL HAVE DEVICE POINTER, R1 WILL HAVE VECTOR POINTER
3308 010342 012700 006732      SETSIZ: MOV     #DEVPNT,R0 ;DEVICE POINTER
3309 010346 012701 006742      MOV     #VECPNT,R1 ;VECTOR PMINTER
3310 010352 005037 006754      CLR     TFOUND ;WILL BE USED AS POINTER
3311 010356                      ;TO FOUND TABLE.
3312 010356 012737 012554 000004  SS2:  MOV     #SS4,ERRVEC ;TIME OUT VECTOR TO REDUCE LTRY
3313 010364 012037 006756      MOV     (R0)+,TESTDV ;ADDRESS OF DEVICE TO BE TESTED
3314 010370 012137 006760      MOV     (R1)+,TESTVC ;VECTOR TO BE TESTED
3315 010374 005777 176356      TST     @TESTDV ;TRY TIME OUT ON DEVICE BASE
3316                      ;IF NO TIME OUT OCCURS THEN DEVICE IN TESTDV IS PRESENT
3317                      ;NOW FILL ALL REGISTER ADDRESS TABLE
3318 010400 012702 000026      MOV     #22,R2 ;COUNT 22
3319 010404 012703 004060      MOV     #RHCS1,R3 ;GET BASE LOCATION
3320 010410 013704 006756      MOV     TESTDV,R4 ;GET BASE ADDRESS
3321 010414 010423          2$:  MOV     R4,(R3)+ ;FILL PROPER ADDRESS
3322 010416 062704 000002      ADD     #2,R4 ;INCREMENT ADDRESS BY 2
3323 010422 005302              DEC     R2 ;COUNT DOWN
3324 010424 001373              BNE     2$ ;BRANCH IF 22 NOT DONE.
3325 010426 005077 173436      CLR     @RHCS2 ;SET UNIT NUMBER
3326 010432
3327 010432 012737 006762 007016  SS6:  MOV     #TYPNT,POINTT ;POINTER TO DRIVE TYPE
3328 010440 012737 000016 007020  3$:  MOV     #16,NTYPNT ;NUMBER OF DRIVE TYPES
3329
3330 010446          SS7:
3331 010446 022737 000001 007020  4$:  CMP     #1,NTYPNT ;IS IT FOR TU
3332 010454 001424              BEQ     5$ ;IF FOR TU BRANCH
3333 010456 017737 173424 001126      MOV     @RHDT,$BDDAT ;READ DRIVE TYPE
3334 010464 012702 000016              MOV     #16,R2 ;GET NUMBER FOR DT NOT DONE
3335 010470 163702 007020      SUB     NTYPNT,R2 ;GET DT TO DO
3336 010474 006302              ASL     R2 ;MULTIPLY R2 BY 2
3337 010476 026237 006762 001126      CMP     TYPNT(R2),$BDDAT ;IS DRIVE TYPE FOUND
3338 010504 001402              BEQ     16$ ;DRIVE TYPE FOUND SO BRANCH
3339 010506 000137 012600              JMP     SS8
3340 010512 032777 010000 173352  16$:  BIT     #MOL,@RHDS1 ;IS IT NON EXISTANT DRIVE
3341 010520 001043              BNE     8$ ;MOL SET, DRIVE TYPE FOUND SO BRANCH
3342                      ;A DEVICE HAS NOT BEEN FOUND
3343 010522 000137 012572              JMP     SS5 ;A DEVICE HAS NOT BEEN FOUND SO BRANCH
3344
3345                      ;TRYING THE TU16 DRIVE TYPE
3346 010526 005037 007022          5$:  CLR     TMPSLV ;SLAVE 0
3347 010532 013777 007022 173352  6$:  MOV     TMPSLV,@RHTC ;MOVE SLAVE NUMBER IN TAPE CONTROL
3348 010540 017737 173342 001126      MOV     @RHDT,$BDDAT ;READ DRIVE TYPE
3349 010546 032737 002000 001126      BIT     #SLVPR,$BDDAT ;IS SLAVE PRESENT
3350 010554 001014              BNE     7$
3351 010556 022737 000007 007022  30$:  CMP     #7,TMPSLV ;IS 7 DONE
    
```



```

3352 010564 001002          BNE      17$          ;BRANCH IF 7 DONE AND NONE FOUND
3353 010566 000137 012600    JMP      SS8
3354 010572 005237 007022    INC      TEMPLV      ;GET NEXT SLAVE
3355 010576 012777 040000 173254 17$:  MOV      #TRE,@RHCS1 ;CLEAR CONTROLLER ERRORS
3356 010604 000752          BR       6$          ;TRY NEXT SLAVE
3357          ;A DEVICE HAS BEEN FOUND SAVE SLAVE
3358 010606 032777 010000 173256 7$:  BIT      #MOL,@RHDS1 ;IS THE DRIVE ON-LINE?
3359 010614 001760          BEQ     30$          ;BRANCH IF NOT
3360 010616 013702 006754          MOV      TFOUND,R2  ;GET NO OF RH FOUND X2 FOR INDEX
3361 010622 013762 007022 007134  MOV      TEMPLV,FSLAVE(R2)
3362          ;A DEVICE HAS BEEN FOUND TEST VECTOR
3363 010630 012737 010644 000004 8$:  MOV      #9$,ERRVEC ;TIME OUT VECTOR TO REPORT ERROR
3364 010636 005777 176116          TST     @TESTVC     ;TRY TIME OUT ON VECTOR
3365 010642 000403          BR      10$         ;A VECTOR EXISTS SO CONTINUE
3366 010644 104146          9$:  ERROR  146         ;A DEVICE BASE ADDRESS DID NOT
3367          ;TIME OUT BUT ITS CORRESPONDING
3368          ;VECTOR ADDRESS TIMED OUT
3369          ;HIT CONTINUE TO REPEAT THIS
3370          ;TEST
3371 010646 000000          HALT
3372 010650 000767          BR      8$          ;REPEAT THIS TEST
3373          ;A DEVICE HAS BEEN FOUND AND VECTOR RESPONDS SO STORE RESULTS
3374 010652 005737 006436          10$: TST     TSTPGM     ;PROGRAMMABLE DRIVES OK?
3375 010656 001006          BNE     20$         ;BRANCH IF YES
3376 010660 032777 001000 173204  BIT      #BIT9, @RHDS1 ;IS THE DRIVE PROGRAMMABLE?
3377 010666 001402          BEQ     20$         ;BRANCH IF NO
3378 010670 000137 012600          JMP     SS8         ;ELSE SKIP THIS DRIVE
3379 010674          20$:
3380 010674 013702 006754          MOV      TFOUND,R2  ;GET NO OF RH FOUND X2 FOR INDEX
3381 010700 013762 007020 007024  MOV      NTPNT,FDEVIC(R2) ;DEVICE 1=TU; 2,3,4,5,6,7=RP;
3382          ;10,11,12,13,14=RS , 15,16=RM
3383 010706 017762 173156 007040  MOV      @RHCS2,FUNITN(R2) ;GET RHCS2
3384 010714 042762 177770 007040  BIC     #177770,FUNITN(R2) ;KEEP UNIT NUMBER
3385 010722 017762 173160 007054  MOV      @RHDT,FTYPE(R2) ;DRIVE BYTE
3386 010730 013762 006760 007070  MOV      TESTVC,FVECTR(R2) ;VECTOR
3387 010736 013762 006756 007104  MOV      TESTDV,FBASEA(R2) ;BASE ADDRESS
3388 010744 013762 006754 007120  MOV      TFOUND,FRHM(R2) ;RH NUMBER X2 MINUS ONE
3389 010752 006262 007120          ASR     FRHM(R2)    ;RH NUMBER MINUS ONE
3390 010756 005262 007120          INC     FRHM(R2)    ;RH NUMBER
3391 010762 022762 000001 007024  CMP     #1,FDEVIC(R2) ;IS IT TU
3392 010770 001016          BNE     11$         ;BRANCH IF NOT TU
3393 010772 012762 042634 007150  MOV      #CMNDTM,FDRIVR(R2) ;DRIVER ADDRESS FOR TU
3394 011000 013746 006756          MOV      TESTDV,-(SP) ;BASE ADDRESS
3395 011004 062716 000034          ADD     #<2*14.>,(SP) ;15TH ADDRESS IS BAE FOR TU
3396 011010 011662 007164          MOV      (SP),FBAE(R2) ;SAVE BAE FOR TU
3397 011014 062716 000002          ADD     #2,(SP)     ;16TH ADDRESS IS CS3 FOR TU
3398 011020 012662 007200          MOV      (SP)+,FCS3(R2) ;SAVE CS3 FOR TU
3399
3400 011024 000521          BR      SS1
3401 011026 022762 000002 007024 11$:  CMP     #2,FDEVIC(R2) ;IS IT RP DUAL PORT
3402 011034 001434          BEQ     12$         ;BRANCH IF RP FOUND
3403 011036 022762 000003 007024  CMP     #3,FDEVIC(R2) ;IS IT RP
3404 011044 001430          BEQ     12$         ;BRANCH IF RP FOUND
3405 011046 022762 000004 007024  CMP     #4,FDEVIC(R2) ;IS IT RP DUAL PORT
3406 011054 001424          BEQ     12$         ;BRANCH IF RP FOUND
3407 011056 022762 000005 007024  CMP     #5,FDEVIC(R2) ;IS IT RP

```

```

3408 011064 001420 BEQ 12$ ;BRANCH IF RP FOUND
3409 011066 022762 000006 007024 CMP #6,FDEVIC(R2) ;IS IT RP DUAL PORT
3410 011074 001414 BEQ 12$ ;BRANCH IF RP FOUND
3411 011076 022762 000007 007024 CMP #7,FDEVIC(R2) ;IS IT RP
3412 011104 001410 BEQ 12$ ;BRANCH IF RP
3413 011106 022762 000015 007024 CMP #15,FDEVIC(R2) ; RM03 DUAL PORT ??????
3414 011114 001404 BEQ 12$ ; BRANCH IF IT IS,
3415 011116 022762 000016 007024 CMP #16,FDEVIC(R2) ; RM03 SINGLE PORT ????????
3416 011124 001016 BNE 13$ ; BRANCH IF IT IS NOT
3417 011126 012762 042500 007150 12$: MOV #CMNDRP,FDRIVR(R2) ;DRIVER ADDRESS FOR RP
3418 011134 013746 006756 MOV TESTDV,-(SP) ;BASE ADDRESS
3419 011140 062716 000050 ADD #<2*20>,(SP) ;21ST ADDRESS IS BAE FOR RP
3420 011144 011662 007164 MOV (SP),FBAE(R2) ;SAVE BAE FOR RP
3421 011150 062716 000002 ADD #2,(SP) ;22ND ADDRESS IS CS3 FOR RP
3422 011154 012662 007200 MOV (SP)+,FCS3(R2) ;SAVE CS3 FOR RP
3423
3424 011160 000443 BR SS1
3425 011162 022762 000010 007024 13$: CMP #10,FDEVIC(R2) ;IS IT RS03/LA
3426 011170 001420 BEQ 14$ ;BRANCH IF RS03/LA
3427 011172 022762 000011 007024 CMP #11,FDEVIC(R2) ;IS IT RS04/L
3428 011200 001414 BEQ 14$ ;BRANCH IF RS04/L
3429 011202 022762 000012 007024 CMP #12,FDEVIC(R2) ;IS IT RS04
3430 011210 001410 BEQ 14$ ;BRANCH IF RS04
3431 011212 022762 000013 007024 CMP #13,FDEVIC(R2) ;IS IT RS03/L
3432 011220 001404 BEQ 14$ ;BRANCH IF RS03/L
3433 011222 022762 000014 007024 CMP #14,FDEVIC(R2) ;IS IT RS03
3434 011230 001016 BNE 15$ ;BRANCH IF NOT RS03
3435 011232 012762 042410 007150 14$: MOV #CMNDRS,FDRIVR(R2) ;DRIVER ADDRESS FOR RS
3436 011240 013746 006756 MOV TESTDV,-(SP) ;BASE ADDRESS
3437 011244 062716 000030 ADD #<2*12>,(SP) ;13TH ADDRESS IS BAE FOR RS
3438 011250 011662 007164 MOV (SP),FBAE(R2) ;SAVE BAE FOR RS
3439 011254 062716 000002 ADD #2,(SP) ;14TH ADDRESS IS CS3 FOR RS
3440 011260 012662 007200 MOV (SP)+,FCS3(R2) ;SAVE CS3 FOR RS
3441 011264 000401 BR SS1 ;CONTINUE
3442 011266 000000 15$: HALT ;PROGRAM ERROR
3443
3444 ;NOW TYPE RESULT
3445
3446 011270 SS1:
3447 011270 104401 011276 TYPE ,65$ ;:TYPE ASCIZ STRING
3448 011274 000406 BR ,64$ ;:GET OVER THE ASCIZ
3449 ;:65$: .ASCIZ <15><12>/ON RH NO/
3450 64$:
3451 011312 MOV FRHNM(R2),-(SP) ;TYPE RH NUMBER
3452 011312 016246 007120 TYPDS
3453 011316 104405 TYPE ,67$ ;:TYPE ASCIZ STRING
3454 011320 104401 011326 BR ,66$ ;:GET OVER THE ASCIZ
3455 ;:67$: .ASCIZ / BASE /
3456 66$:
3457 011350 MOV FBASEA(R2),-(SP) ;TYPE BASE ADDRESS
3458 011350 016246 007104 TYPDC
3459 011354 104402 TYPE ,69$ ;:TYPE ASCIZ STRING
3460 011362 104401 011364 BR ,68$ ;:GET OVER THE ASCIZ
3461 ;:69$: .ASCIZ / FOUND/
3462 68$:
3463 011374 022762 000001 007024 CMP #1,FDEVIC(R2) ;IS IT TU

```



```

3464 011402 001020          BNE      1$          ;BRANCH IF NOT TU
3465 011404 104401 011412  TYPE      71$       ;;TYPE ASCIZ STRING
3466 011410 000410          BR       70$       ;;GET OVER THE ASCIZ
3467          ;;71$: .ASCIZ / TU16 AT SLAVE/
3468          70$:
3469 011432 016246 007134  MOV     FSLAVE(R2),-(SP) ;TYPE SLAVE NUMBER
3470 011436 104405          TYPDS
3471 011440 000137 012372  JMP     6$
3472          ;THE FOLLOWING TYPES OUT
3473          ;THE DEVICE AND REDEFINES
3474          ;FDEVIC: 1=TU, 2=RP DUAL PORT
3475          ;3=RP SINGLE PORT
3476          ;4=RS04/L,RS04
3477          ;5=RS03/LA,RS03/L,RS03
3478          ;6= RM03 DUAL PORT,7=RM03 SINGLE PORT
3479 011444 022762 000002 007024 1$:    CMP     #2,FDEVIC(R2) ;IS IT RP06 DUAL PORT
3480 011452 001016          BNE     2$          ;BRANCH IF NOT RP06 DUAL PORT
3481 011454 104401 011462  TYPE     73$       ;;TYPE ASCIZ STRING
3482 011460 000411          BR      72$       ;;GET OVER THE ASCIZ
3483          ;;73$: .ASCIZ / RP06 DUAL PORT /
3484          72$:
3485 011504 000137 012372  JMP     6$
3486 011510 022762 000003 007024 2$:    CMP     #3,FDEVIC(R2) ;IS IT RP06 SINGLE PORT
3487 011516 001016          BNE     3$          ;BRANCH IF NOT RP06 SINGLE PORT
3488 011520 104401 011526  TYPE     75$       ;;TYPE ASCIZ STRING
3489 011524 000411          BR      74$       ;;GET OVER THE ASCIZ
3490          ;;75$: .ASCIZ / RP06 SINGLE PORT/
3491          74$:
3492 011550 000137 012372  JMP     6$
3493 011554 022762 000004 007024 3$:    CMP     #4,FDEVIC(R2) ;IS IT RP05 DUAL PORT
3494 011562 001020          BNE     4$          ;BRANCH IF NOT RP05 DUAL PORT
3495 011564 012762 000002 007024  MOV     #2,FDEVIC(R2)
3496 011572 104401 011600  TYPE     77$       ;;TYPE ASCIZ STRING
3497 011576 000410          BR      76$       ;;GET OVER THE ASCIZ
3498          ;;77$: .ASCIZ / RP05 DUAL PORT/
3499          76$:
3500 011620 000137 012372  JMP     6$
3501 011624 022762 000005 007024 4$:    CMP     #5,FDEVIC(R2) ;IS IT RP05 SINGLE PORT
3502 011632 001021          BNE     8$          ;BRANCH IF NOT RP05 SINGLE PORT
3503 011634 012762 000003 007024  MOV     #3,FDEVIC(R2)
3504 011642 104401 011650  TYPE     79$       ;;TYPE ASCIZ STRING
3505 011646 000411          BR      78$       ;;GET OVER THE ASCIZ
3506          ;;79$: .ASCIZ / RP05 SINGLE PORT/
3507          78$:
3508 011672 000137 012372  JMP     6$
3509 011676 022762 000006 007024 8$:    CMP     #6,FDEVIC(R2) ;IS IT RP04 DUAL PORT
3510 011704 001020          BNE     9$          ;BRANCH IF NOT RP04 DUAL PORT
3511 011706 012762 000002 007024  MOV     #2,FDEVIC(R2)
3512 011714 104401 011722  TYPE     81$       ;;TYPE ASCIZ STRING
3513 011720 000410          BR      80$       ;;GET OVER THE ASCIZ
3514          ;;81$: .ASCIZ / RP04 DUAL PORT/
3515          80$:
3516 011742 000137 012372  JMP     6$
3517 011746 022762 000007 007024 9$:    CMP     #7,FDEVIC(R2) ;IS IT RP04 SINGLE PORT
3518 011754 001021          BNE    10$         ;BRANCH IF NOT RP04 SINGLE PORT
3519 011756 012762 000003 007024  MOV     #3,FDEVIC(R2)

```

```

3520 011764 104401 011772      TYPE      83$      ;;TYPE ASCIZ STRING
3521 011770 000411      BR      82$      ;;GET OVER THE ASCIZ
3522      ;;83$: .ASCIZ / RP04 SINGLE PORT/
3523 012014      82$:
3524 012014 000137 012372      JMP      6$
3525 012020 022762 000010 007024 10$: CMP #10,FDEVIC(R2) ;IS IT RS03-LA
3526 012026 001015      BNE     11$      ;BRANCH IF NOT RS03-LA
3527 012030 012762 000005 007024      MOV #5,FDEVIC(R2)
3528 012036 104401 012044      TYPE     85$      ;;TYPE ASCIZ STRING
3529 012042 000405      BR      84$      ;;GET OVER THE ASCIZ
3530      ;;85$: .ASCIZ / RS03-LA/
3531      84$:
3532 012056      JMP      6$
3533 012062 000137 012372 000011 007024 11$: CMP #11,FDEVIC(R2) ;IS IT RS04-L
3534 012070 001013      BNE     12$      ;BRANCH IF NOT RS04-L
3535 012072 012762 000004 007024      MOV #4,FDEVIC(R2)
3536 012100 104401 012106      TYPE     87$      ;;TYPE ASCIZ STRING
3537 012104 000404      BR      86$      ;;GET OVER THE ASCIZ
3538      ;;87$: .ASCIZ / RS04-L/
3539      86$:
3540 012116      BR      6$
3541 012120 000525 000012 007024 12$: CMP #12,FDEVIC(R2) ;IS IT RS04
3542 012126 001012      BNE     13$      ;BRANCH IF NOT RS04
3543 012130 012762 000004 007024      MOV #4,FDEVIC(R2)
3544 012136 104401 012144      TYPE     89$      ;;TYPE ASCIZ STRING
3545 012142 000403      BR      88$      ;;GET OVER THE ASCIZ
3546      ;;89$: .ASCIZ / RS04/
3547      88$:
3548 012152      BR      6$
3549 012154 000507 000013 007024 13$: CMP #13,FDEVIC(R2) ;IS IT RS03-L
3550 012162 001013      BNE     14$      ;BRANCH IF NOT RS03-L
3551 012164 012762 000005 007024      MOV #5,FDEVIC(R2)
3552 012172 104401 012200      TYPE     91$      ;;TYPE ASCIZ STRING
3553 012176 000404      BR      90$      ;;GET OVER THE ASCIZ
3554      ;;91$: .ASCIZ / RS03-L/
3555      90$:
3556 012210      BR      6$
3557 012210 000470 000014 007024 14$: CMP #14,FDEVIC(R2) ;IS IT RS03
3558 012220 001012      BNE     15$      ;BRANCH IF NOT RS03
3559 012222 012762 000005 007024      MOV #5,FDEVIC(R2)
3560 012230 104401 012236      TYPE     93$      ;;TYPE ASCIZ STRING
3561 012234 000403      BR      92$      ;;GET OVER THE ASCIZ
3562      ;;93$: .ASCIZ / RS03/
3563      92$:
3564 012244      BR      6$
3565 012244 000452 000015 007024 15$: CMP #15,FDEVIC(R2) ; IS IT RM03 DUAL PORT?
3566 012254 001020      BNE     16$      ; BR IF NOT A RM03 DUAL
3567 012256 012762 000006 007024      MOV #6, FDEVIC(R2)
3568 012264 104401 012272      TYPE     95$      ;;TYPE ASCIZ STRING
3569 012270 000411      BR      94$      ;;GET OVER THE ASCIZ
3570      ;;95$: .ASCIZ / RM03 DUAL PORT/
3571      94$:
3572 012314      BR 6$
3573 012314 000426 000016 007024 16$: CMP #16,FDEVIC(R2) ; RM03 SINGLE PORT
3574 012324 001021      BNE     5$      ; BR IF NOT
3575 012326 012762 000007 007024      MOV #7, FDEVIC(R2) ; RM03 REDEFINED

```



```

3576 012334 104401 012342          TYPE      ,97$          ;;TYPE ASCIZ STRING
3577 012340 000412          BR        ,96$          ;;GET OVER THE ASCIZ
3578          ;;97$: .ASCIZ /          RM03 SINGLE PORT/
3579 012366          96$:          BR 6$
3580 012366 000401          5$:          HALT          ;PROGRAM ERROR
3581 012370 000000          6$:
3582 012372          TYPE      ,99$          ;;TYPE ASCIZ STRING
3583 012372 104401 012400          BR        ,98$          ;;GET OVER THE ASCIZ
3584 012376 000411          ;;99$: .ASCIZ <15><12>/AT UNIT NUMBER/
3585          98$:
3586 012422          MOV      FUNITN(R2),-(SP) ;TYPE UNIT NUMBER
3587 012422 016246 007040          TYPDS
3588 012426 104405          TYPE      ,101$         ;;TYPE ASCIZ STRING
3589 012430 104401 012436          BR        ,100$         ;;GET OVER THE ASCIZ
3590 012434 000411          ;;101$: .ASCIZ /          VECTOR /
3591          100$:
3592 012460          MOV      FVECTR(R2),-(SP) ;TYPE VECTOR
3593 012460 016246 007070          TYPOC
3594 012464 104402          TYPE      ,103$         ;;TYPE ASCIZ STRING
3595 012466 104401 012474          BR        ,102$         ;;GET OVER THE ASCIZ
3596 012472 000402          ;;103$: .ASCIZ <15><12>/ /
3597          102$:
3598 012500          TYPE      ,105$         ;;TYPE ASCIZ STRING
3599 012500 104401 012506          BR        ,104$         ;;GET OVER THE ASCIZ
3600 012504 000402          ;;105$: .ASCIZ <15><12>/ /
3601          104$:
3602 012512
3603
3604          ;UPDATE TFOUND
3605 012512 062737 000002 006754          ADD      #2,TFOUND          ;GET READY FOR NEXT DEVICE
3606
3607          ;ALL UNITS DONE? IF LTRY BECOMES ZERO YES ALL DONE
3608 012520 005337 006752          DEC      LTRY          ;REDUCE DEVICES LEFT TO TRY
3609 012524 001402          BEQ      7$          ;BRANCH IF ALL COMPLETE
3610 012526 000137 010356          JMP      SS2          ;ALL NOT DONE
3611 012532          7$:
3612 012532 013746 006754          SS3: MOV      TFOUND,-(SP)          ;GET TWICE NUMBER OF RH FOUND
3613 012536 006216          ASR      (SP)          ;DIVIDE BY 2
3614 012540 011637 007214          MOV      (SP),TSTUNT
3615 012544 012637 007216          MOV      (SP)+,WORUNT
3616 012550 000137 012664          JMP      TST2          ;JUMP TO NEXT TEST
3617
3618          ;A RH TIMED OUT SO TRY NEXT UNIT OR END TRYING
3619 012554 005337 006752          SS4: DEC      LTRY          ;RH TIMES OUT SO DECREASE RH TYPED
3620 012560 001402          BEQ      1$          ;BRANCH IF ALL DONE
3621 012562 000137 010356          JMP      SS2          ;ALL NOT DONE SO JUMP
3622 012566 000137 012532          1$: JMP      SS3          ;ALL DONE SO STORE NUMBER OF RH FOUND
3623
3624          ;A DRIVE TYPE DID NOT MATCH SO TRY NEXT
3625 012572 012777 040000 171260          SS5: MOV      #TRE,@RHCS1          ;CLEAR NED ERROR
3626 012600 005337 007020          SS8: DEC      NTYPNT          ;DECREASE NUMBER OF DRIVE TYPES TRIED
3627 012604 001402          BEQ      1$          ;BRANCH IF ALL DONE
3628 012606 000137 010446          JMP      SS7          ;ALL NOT DONE
3629 012612 005277 171252          1$: INC      @RHCS2          ;GET NEXT UNIT NO.
3630 012616 012777 040000 171234          MOV      #TRE,@RHCS1          ;CLEAR NED ERROR
3631 012624 017746 171240          MOV      @RHCS2,-(SP)          ;GET RHCS2
    
```

```

3632 012630 042716 177770      BIC    #177770,(SP)  :GET UNIT NO
3633 012634 022726 000000      CMP    #0.,(SP)+   :ARE 7 DONE
3634 012640 001402                BEQ    2$           :BRANCH IF 7 DONE
3635 012642 000137 010432      JMP    SS6         :7 NOT DONE SO TRY NEXT UNIT NO
3636
3637 012646 005737 007024      2$:   TST    FDEVIC  : TEST DEVICE FOUND
3638 012652 001340                BNE SS4           : BR IF AT LEAST ONE DRIVE FOUND
3639 012654 104147                ERROR  147       :DEVICE ADDRESS DID NOT TIME
3640 012656 000000                HALT              :OUT BUT NO UNITS HAD
3641
3642 012660 000137 012554      JMP    SS4         :APPROPRIATE DRIVE TYPE.
3643
3644
3645
3646
3647
3648
3649
3650

```

```

3651
3652
3653
3654
3655 012664 000004
3656 012666 012737 000001 001212
3657 012674 012737 040510 000004
3658 012702 012737 000340 000006
3659
3660 012710 013746 007216
3661 012714 013702 007214
3662 012720 162602
3663 012722 006302
3664 012724 104401 012732
3665 012730 000402
3666
3667 012736
3668 012736 104401 012744
3669 012742 000410
3670
3671 012764
3672 012764 016246 007120
3673 012770 104405
3674
3675 012772 104401 013000
3676 012776 000410
3677
3678 013020
3679 013020 104401 013026
3680 013024 000404
3681
3682 013036
3683 013036 016246 007104
3684 013042 104402
3685 013044 022762 000001 007024
3686 013052 001017
3687 013054 104401 013062

```

```

:*****
:*TEST 2          UNIT UNDER TEST
:*              THIS TYPES THE UNIT TO BE TESTED
:*              AND IS THE FIRST TEST AFTER THE END OF THE PROGRAM
:*****
TST2:  SCOPE
      MOV    #1,$TIMES      ;;DO 1 ITERATION
      MOV    #TIEOUT,@#ERRVEC ;TIMEOUT VECTOR
      MOV    #340,@#ERRVEC+2 ;PRIORITY 7
      MOV    WORUNT,-(SP)   ;GET WORKING RH NUMBER LEFT
      MOV    TSTUNT,R2     ;GET TOTAL NO OF RH FOUND
      SUB    (SP)+,R2      ;NO OF RH TO BE TESTED
      ASL    R2             ;INDEX FOR RH TO BE TESTED.
      TYPE   ,65$          ;;TYPE ASCIZ STRING
      BR     64$           ;;GET OVER THE ASCIZ
::65$: .ASCIZ <15><12>/ /
64$:   TYPE   ,67$          ;;TYPE ASCIZ STRING
      BR     66$           ;;GET OVER THE ASCIZ
::67$: .ASCIZ <15><12>/TESTING RH NO/
66$:   MOV    FRHNM(R2),-(SP) ;TYPE RH NO.
      TYPDS
      TYPE   ,69$          ;;TYPE ASCIZ STRING
      BR     68$           ;;GET OVER THE ASCIZ
::69$: .ASCIZ /          USING /
68$:   TYPE   ,71$          ;;TYPE ASCIZ STRING
      BR     70$           ;;GET OVER THE ASCIZ
::71$: .ASCIZ / BASE /
70$:   MOV    FBASEA(R2),-(SP) ;TYPE BASE ADDRESS
      TYPOC
      CMP    #1,FDEVIC(R2) ;IS IT TU
      BNE    1$           ;BRANCH IF NOT TU
      TYPE   ,73$          ;;TYPE ASCIZ STRING

```



```

3688 013060 000410 BR 72$ ::GET OVER THE ASCIZ
3689 ::73$: .ASCIZ / TU16 AT SLAVE/
3690 013102 72$:
3691 013102 016246 007134 MOV FSLAVE(R2),-(SP) ;TYPE SLAVE NUMBER
3692 013106 104405 TYPDS
3693 013110 000526 BR 6$
3694 013112 022762 000002 007024 1$: CMP #2,FDEVIC(R2) ;IS IT RP DUAL PORT
3695 013120 001014 BNE 2$ ;BRANCH IF NOT RP DUAL PORT
3696 013122 104401 013130 TYPE ,75$ ;:TYPE ASCIZ STRING
3697 013126 000410 BR 74$ ;:GET OVER THE ASCIZ
3698 ::75$: .ASCIZ / RP DUAL PORT /
3699 74$:
3700 013150 000506 BR 6$
3701 013152 022762 000003 007024 2$: CMP #3,FDEVIC(R2) ;IS IT RO SINGLE PORT
3702 013160 001014 BNE 3$ ;BRANCH IF NOT RP
3703 013162 104401 013170 TYPE ,77$ ;:TYPE ASCIZ STRING
3704 013166 000410 BR 76$ ;:GET OVER THE ASCIZ
3705 ::77$: .ASCIZ / RP SINGLE PORT/
3706 76$:
3707 013210 000466 BR 6$
3708 013212 022762 000004 007024 3$: CMP #4,FDEVIC(R2) ;IS IT RS04
3709 013220 001007 BNE 4$ ;BRANCH IF NOT RS04
3710 013222 104401 013230 TYPE ,79$ ;:TYPE ASCIZ STRING
3711 013226 000403 BR 78$ ;:GET OVER THE ASCIZ
3712 ::79$: .ASCIZ / RS04/
3713 78$:
3714 013236 000453 BR 6$
3715 013240 022762 000005 007024 4$: CMP #5,FDEVIC(R2) ;IS IT RS03
3716 013246 001007 BNE 5$ ;BRANCH IF NOT RS03
3717 013250 104401 013256 TYPE ,81$ ;:TYPE ASCIZ STRING
3718 013254 000403 BR 80$ ;:GET OVER THE ASCIZ
3719 ::81$: .ASCIZ / RS03/
3720 80$:
3721 013264 000440 BR 6$
3722 013266 022762 000006 007024 5$: CMP #6,FDEVIC(R2) ; RM03 DUAL PORT?
3723 013274 001013 BNE 666$ ; BR IF NOT
3724 013276 104401 013304 TYPE ,83$ ;:TYPE ASCIZ STRING
3725 013302 000407 BR 82$ ;:GET OVER THE ASCIZ
3726 ::83$: .ASCIZ / RM DUAL PORT/
3727 82$:
3728 013322 000421 BR 6$
3729 013324 022762 000007 007024 666$: CMP #7,FDEVIC(R2) ; RM03 SINGLE PORT ???
3730 013332 001014 BNE 555$ ; BRANCH IF NOT
3731 013334 104401 013342 TYPE ,85$ ;:TYPE ASCIZ STRING
3732 013340 000410 BR 84$ ;:GET OVER THE ASCIZ
3733 ::85$: .ASCIZ / RM SINGLE PORT/
3734 84$:
3735 013362 000401 BR 6$ ;CONTINUE WITH DEVICE FOUND
3736
3737 013364 000000 555$: HALT ;PROGRAM ERROR
3738
3739 013366 6$:
3740 013366 104401 013374 TYPE ,87$ ;:TYPE ASCIZ STRING
3741 013372 000411 BR 86$ ;:GET OVER THE ASCIZ
3742 ::87$: .ASCIZ <15><12>/AT UNIT NUMBER/
3743 86$:

```

```
3744 013416 016246 007040      MOV      FUNITN(R2),-(SP) ;TYPE UNIT NUMBER
3745 013422 104405              TYPDS
3746 013424 104401 013432      TYPE      ,89$           ;;TYPE ASCIZ STRING
3747 013430 000411              BR        88$           ;;GET OVER THE ASCIZ
3748                               ;;89$: .ASCIZ /      VECTOR /
3749 013454                      88$:
3750 013454 016246 007070      MOV      FVECTR(R2),-(SP) ;TYPE VECTOR
3751 013460 104402              TYPOC
3752 013462 012703 000024      MOV      #20,,R3        ;COUNT 20
3753 013466 012704 004060      MOV      #RHCS1,R4      ;GET BASE LOCATION
3754 013472 016205 007104      MOV      FBASEA(R2),R5  ;GET BASE ADDRESS
3755 013476 010524              7$: MOV      R5,(R4)+     ;FILL PROPER ADDRESS
3756 013500 062705 000002      ADD      #2,R5          ;INCREMENT BY 2
3757 013504 005303              DEC      R3              ;COUNT DOWN
3758 013506 001373              BNE      7$             ;BRANCH IF 20 NOT DONE
3759
3760 013510 016237 007040 006442      MOV      FUNITN(R2),UNIT ;UNIT NUMBER
3761 013516 016237 007070 007220      MOV      FVECTR(R2),VECTOR ;VECTOR
3762 013524 016237 007134 006300      MOV      FSLAVE(R2),SLAVE ;SLAVE NO
3763 013532 016237 007150 006260      MOV      FDRIVR(R2),COMND ;DRIVER ADDRESS
3764 013540 016237 007164 004130      MOV      FBAE(R2),RHBAE ;BAE ADDRESS
3765 013546 016237 007200 004132      MOV      FCS3(R2),RHCS3 ;CS3 ADDRESS
3766
3767 013554 005037 046024      CLR      PRITEM
```

3768
3769
3770
3771
3772
3773
3774
3775
3776
3777

*TEST 3 BIT BANG RHCS1

* 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 WILL NOT BE WRITTEN INTO
* 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 SYNCs ON THE CLEAR
* SIGNAL.

3790
3791
3792
3793
3794
3795
3796
3797
3798
3799

```
*****  
TST3: SCOPE  
MOV      #STACK,SP      ;RESET STACK  
MOV      #3,@#TSTNM     ;SAVE TEST NUMBER  
JSR      PC,@#CLDISK    ;GIVE RH INITIALIZE  
;SETUP UNIT NUBER  
;CLEAR RHWC AND FUNCTION BITS IN RHCS1
```



```

3800
3801 013600 012737 176201 006270      MOV      #SC!TRE!MCPE!DVA!BIT12!BIT10!RDY!GO,WRTBIT      ;SC!TRE!MCPE!DVA!BIT12:B
3802 013606 012737 004200 006272      MOV      #RDY!DVA,SETBIT ;RDY!DVA ARE BITS ALWAYS SET
3803 013614 012737 002000 006274      MOV      #BIT10,CLRBIT ;BIT10 ARE BITS ALWAYS CLEARED
3804
3805                                     ;FLOAT 1'S THRU THE RHCS1 REGISTER
3806 013622 012737 013646 001110      MOV      #2$,SLPERR ;SET LOOP ON ERROR ADDRESS
3807 013630 012705 000001                MOV      #1,R5 ;GETTING READY TO FLOAT A ONE
3808 013634 010537 001124                MOV      R5,$GDDAT ;START WITH DATA
3809 013640 042737 176201 001124      BIC      #SC!TRE!MCPE!DVA!BIT12!BIT10!RDY!GO,$GDDAT ;CLEAR BITS NOT WRITTEN
3810 013646 032777 041400 165264      BIT      #BIT14!BIT9!BIT8,@SWR ;ARE SWITCHES 14 OR 9 OR 8 SET
3811 013654 001406                BEQ      3$ ;BRANCH IF ANY ONE SET
3812 013656 052777 000040 170204      BIS      #CLR,@RHCS2 ;GIVE CLEAR FOR SCOPE
3813                                     ;SYNC IF IN DEBUG:
3814 013664 013777 006442 170176      MOV      UNIT,@RHCS2
3815 013672 013777 001124 170160      MOV      $GDDAT,@RHCS1 ;WRITE RHCS1 REGISTER
3816 013700 017737 170154 001126      MOV      @RHCS1,$BDDAT ;READ RHCS1 REGISTER
3817 013706 043737 006274 001124      BIC      CLRBIT,$GDDAT ;CLEAR ALWAYS CLEARED BITS
3818 013714 053737 006272 001124      BIS      SETBIT,$GDDAT ;SET ALWAYS SET BITS
3819 013722 023737 001124 001126      CMP      $GDDAT,$BDDAT ;TEST
3820 013730 001404                BEQ      4$ ;BRANCH IF GOOD
3821 013732 013737 004060 001122      MOV      RHCS1,$BDADR ;GET REGISTER ADDRESS
3822 013740 104137                ERROR    137 ;ONE WAS BEING FLOATED
3823                                     ;THRU RHCS1 REGISTER
3824                                     ;ON READING RHCS1 BACK
3825                                     ;IT DID NOT CONTAIN WHAT
3826                                     ;WAS EXPECTED.
3827 013742 000241                4$:     CLC ;CLEAR CARRY
3828 013744 006305                ASL      R5 ;GET 1 ONE LEFT
3829 013746 103332                BCC      1$ ;BRANCH IF 16 NOT DONE
3830
3831                                     ;FLOAT A ZERO THRU RHCS1 REGISTER.
3832
3833 013750 012737 013774 001110      MOV      #6$,SLPERR ;SET LOOP BACK POINT.
3834 013756 012705 177776                MOV      #177776,R5 ;GET READY TO FLOAT A ZERO
3835 013762 010537 001124                MOV      R5,$GDDAT ;GET READY TO WRITE DATA
3836 013766 042737 176201 001124      BIC      #SC!TRE!MCPE!DVA!BIT12!BIT10!RDY!GO,$GDDAT ;CLEAR BITS NOT WRITTEN
3837 013774 032777 041400 165136      BIT      #BIT14!BIT9!BIT8,@SWR ;ARE SWITCHES 14 OR 9 OR 8 SET
3838 014002 001406                BEQ      7$ ;BRANCH IF ANY OF THE ABOVE SET
3839 014004 012777 000040 170056      MOV      #CLR,@RHCS2 ;CLEAR FOR SCOPE AID TO
3840                                     ;SYNC IF IN DEBUG
3841 014012 013777 006442 170050      MOV      UNIT,@RHCS2
3842 014020 013777 001124 170032      MOV      $GDDAT,@RHCS1 ;WRITE INTO RH'XA.
3843 014026 017737 170026 001126      MOV      @RHCS1,$BDDAT ;READ RHCS1
3844 014034 053737 006272 001124      BIS      SETBIT,$GDDAT ;SET BITS ALWAYS SET
3845 014042 043737 006274 001124      BIC      CLRBIT,$GDDAT ;CLEAR BITS ALWAYS 0
3846 014050 023737 001124 001126      CMP      $GDDAT,$BDDAT ;TEST
3847 014056 001404                BEQ      8$ ;BRANCH IF GOOD
3848 014060 013737 004060 001122      MOV      RHCS1,$BDADR ;GET REGISTER ADDRESS
3849 014066 104137                ERROR    137 ;ZERO WAS BEING FLOATED
3850                                     ;THRU RHCS1 REGISTER
3851                                     ;ON READING IT BACK IT
3852                                     ;DID NOT CONTAIN WHAT
3853                                     ;WAS EXPECTED
3854
3855 014070 000261                8$:     SEC
    
```

3856 014072 006105
 3857 014074 103732

ROL R5
 BCS 5\$

3858
 3859
 3860
 3861
 3862
 3863
 3864
 3865
 3866
 3867
 3868
 3869
 3870
 3871
 3872
 3873

 :*TEST 4 BIT BANG RHCS2

:* 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.

3874
 3875 014076 000004
 3876 014100 012706 001000
 3877 014104 012737 000004 006276
 3878
 3879 014112 004737 041222
 3880
 3881
 3882
 3883 014116 012737 177740 006270
 3884 014124 012737 000100 006272
 3885
 3886
 3887 014132 012737 014156 001110
 3888 014140 012705 000001
 3889 014144 010537 001124
 3890 014150 042737 177740 001124
 3891 014156 032777 041400 164754
 3892 014164 001406
 3893 014166 052777 000040 167674
 3894
 3895 014174 013777 006442 167666
 3896 014202 013777 001124 167660
 3897 014210 017737 167654 001126
 3898 014216 053737 006272 001124
 3899 014224 023737 001124 001126
 3900 014232 001404
 3901 014234 013737 004070 001122
 3902 014242 104137
 3903
 3904
 3905
 3906
 3907 014244 000241
 3908 014246 006305
 3909 014250 103335
 3910
 3911

 TST4: SCOPE
 MOV #STACK.SP ;RESET STACK
 MOV #4,@TSTNM ;SAVE TEST NUMBER
 JSR PC,@CLDISK ;GIVE RH INITIALIZE
 ;SETUP UNIT NUBER
 ;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\$,SLPERR ;SET LOOP ON ERROR ADDRESS
 MOV #1,R5 ;GETTING READY TO FLOAT A ONE
 1\$: MOV R5,\$GDDAT ;START WITH DATA
 BIC #177740,\$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 UNIT,@RHCS2
 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,\$BDADR ;GET REGISTER ADDRESS
 ERROR 137 ;ONE WAS BEING FLOATED
 ;THRU RHCS2 REGISTER
 ;ON READING RHCS2 BACK
 ;IT DID NOT CONTAIN WHAT
 ;WAS EXPECTED.
 4\$: CLC ;CLEAR CARRY
 ASL R5 ;GET 1 ONE LEFT
 BCC 1\$;BRANCH IF 16 NOT DONE

:FLOAT A ZERO THRU RHCS2 REGISTER.


```

3912
3913 014252 012737 014276 001110      MOV    #6$, $LPERR      ;SET LOOP BACK POINT.
3914 014260 012705 177776              MOV    #177776, R5      ;GET READY TO FLOAT A ZERO
3915 014264 010537 001124      5$:   MOV    R5, $GDDAT      ;GET READY TO WRITE DATA
3916 014270 042737 177740 001124      BIC    #177740, $GDDAT  ;CLEAR BITS NOT WRITTEN INTO
3917 014276 032777 041400 164634      6$:   BIT    #BIT14!BIT9!BIT8, @SWR ;ARE SWITCHES 14 OR 9 OR 8 SET
3918 014304 001406              BEQ    7$              ;BRANCH IF ANY OF THE ABOVE SET
3919 014306 012777 000040 167554      MOV    #CLR, @RHCS2    ;CLEAR FOR SCOPE AID TO
3920                                ;SYNC IF IN DEBUG
3921 014314 013777 006442 167546      MOV    UNIT, @RHCS2
3922 014322 013777 001124 167540      7$:   MOV    $GDDAT, @RHCS2  ;WRITE INTO RH'XA.
3923 014330 017737 167534 001126      MOV    @RHCS2, $BDDAT  ;READ RHCS2
3924 014336 053737 006272 001124      BIS    SETBIT, $GDDAT  ;SET BITS ALWAYS SET
3925 014344 023737 001124 001126      CMP    $GDDAT, $BDDAT ;TEST
3926 014352 001404              BEQ    8$              ;BRANCH IF GOOD
3927 014354 013737 004070 001122      MOV    RHCS2, $BDADR   ;GET REGISTER ADDRESS
3928 014362 104137              ERROR  137            ;ZERO WAS BEING FLOATED
3929                                ;THRU RHCS2 REGISTER
3930                                ;ON READING IT BACK IT
3931                                ;DID NOT CONTAIN WHAT
3932                                ;WAS EXPECTED
3933
3934 014364 000261      8$:   SEC
3935 014366 006105      ROL    R5
3936 014370 103735      BCS    5$
3937
3938
3939
3940
3941
3942
3943
3944
3945
3946
3947
3948
3949
3950
3951
3952
3953
3954
3955 014372 000004      TST5: SCOPE
3956 014374 012706 001000      MOV    #STACK, SP      ;RESET STACK
3957 014400 012737 000005 006276      MOV    #5, @TSTNM      ;SAVE TEST NUMBER
3958
3959 014406 004737 041222      JSR    PC, @#CLDISK    ;GIVE RH INITIALIZE
3960                                ;SETUP UNIT NUBER
3961                                ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
3962
3963 014412 012737 177660 006270      MOV    #177660, WRTBIT ;177660 ARE BITS NOT WRITTEN INTO
3964 014420 012737 000000 006272      MOV    #0, SETBIT      ;0 ARE BITS ALWAYS SET
3965 014426 012737 001660 006274      MOV    #BIT9!BIT8!BIT7!BIT5!BIT4, CLRBIT ;BIT9!BIT8!BIT7!BIT5!BIT4 ARE BITS ALWAY
3966
3967                                ;FLOAT 1'S THRU THE RHCS3 REGISTER
    
```

```
3968 014434 012737 014460 001110      MOV    #2$, $LPERR      ;SET LOOP ON ERROR ADDRESS
3969 014442 012705 000001              MOV    #1, R5           ;GETTING READY TO FLOAT A ONE
3970 014446 010537 001124              1$:   MOV    R5, $GDDAT      ;START WITH DATA
3971 014452 042737 177660 001124      BIC    #177660, $GDDAT  ;CLEAR BITS NOT WRITTEN INTO
3972 014460 032777 041400 164452      2$:   BIT    #BIT14!BIT9!BIT8, @SWR ;ARE SWITCHES 14 OR 9 OR 8 SET
3973 014466 001406                    BEQ    3$               ;BRANCH IF ANY ONE SET
3974 014470 052777 000040 167372      BIS    #CLR, @RHCS2    ;GIVE CLEAR FOR SCOPE
3975                                ;SYNC IF IN DEBUG:
3976 014476 013777 006442 167364      MOV    UNIT, @RHCS2
3977 014504 013777 001124 167420      3$:   MOV    $GDDAT, @RHCS3  ;WRITE RHCS3 REGISTER
3978 014512 017737 167414 001126      MOV    @RHCS3, $BDDAT  ;READ RHCS3 REGISTER
3979 014520 043737 006274 001124      BIC    CLRBIT, $GDDAT  ;CLEAR ALWAYS CLEARED BITS
3980 014526 053737 006272 001124      BIS    SETBIT, $GDDAT  ;SET ALWAYS SET BITS
3981 014534 023737 001124 001126      CMP    $GDDAT, $BDDAT ;TEST
3982 014542 001404                    BEQ    4$               ;BRANCH IF GOOD
3983 014544 013737 004132 001122      MOV    RHCS3, $BDADR  ;GET REGISTER ADDRESS
3984 014552 104137                    ERROR  137             ;ONE WAS BEING FLOATED
3985                                ;THRU RHCS3 REGISTER
3986                                ;ON READING RHCS3 BACK
3987                                ;IT DID NOT CONTAIN WHAT
3988                                ;WAS EXPECTED.
3989 014554 000241                    4$:   CLC                    ;CLEAR CARRY
3990 014556 006305                    ASL    R5               ;GET 1 ONE LEFT
3991 014560 103332                    BCC    1$               ;BRANCH IF 16 NOT DONE
3992                                ;FLOAT A ZERO THRU RHCS3 REGISTER.
3993
3994
3995 014562 012737 014606 001110      MOV    #6$, $LPERR      ;SET LOOP BACK POINT.
3996 014570 012705 177776              MOV    #177776, R5     ;GET READY TO FLOAT A ZERO
3997 014574 010537 001124              5$:   MOV    R5, $GDDAT      ;GET READY TO WRITE DATA
3998 014600 042737 177660 001124      BIC    #177660, $GDDAT ;CLEAR BITS NOT WRITTEN INTO
3999 014606 032777 041400 164324      6$:   BIT    #BIT14!BIT9!BIT8, @SWR ;ARE SWITCHES 14 OR 9 OR 8 SET
4000 014614 001406                    BEQ    7$               ;BRANCH IF ANY OF THE ABOVE SET
4001 014616 012777 000040 167244      MOV    #CLR, @RHCS2    ;CLEAR FOR SCOPE AID TO
4002                                ;SYNC IF IN DEBUG
4003 014624 013777 006442 167236      MOV    UNIT, @RHCS2
4004 014632 013777 001124 167272      7$:   MOV    $GDDAT, @RHCS3 ;WRITE INTO RH'XA.
4005 014640 017737 167266 001126      MOV    @RHCS3, $BDDAT ;READ RHCS3
4006 014646 053737 006272 001124      BIS    SETBIT, $GDDAT ;SET BITS ALWAYS SET
4007 014654 043737 006274 001124      BIC    CLRBIT, $GDDAT ;CLEAR BITS ALWAYS 0
4008 014662 023737 001124 001126      CMP    $GDDAT, $BDDAT ;TEST
4009 014670 001404                    BEQ    8$               ;BRANCH IF GOOD
4010 014672 013737 004132 001122      MOV    RHCS3, $BDADR  ;GET REGISTER ADDRESS
4011 014700 104137                    ERROR  137             ;ZERO WAS BEING FLOATED
4012                                ;THRU RHCS3 REGISTER
4013                                ;ON READING IT BACK IT
4014                                ;DID NOT CONTAIN WHAT
4015                                ;WAS EXPECTED
4016
4017 014702 000261                    8$:   SEC                    ;
4018 014704 006105                    ROL    R5               ;
4019 014706 103732                    BCS    5$               ;
4020
4021
4022
4023
*****
; *TEST 6 BIT BANG RHWC
```



```

4024
4025      :*      TEST LOADING AND READING OF ALL POSSIBLE BITS
4026      :*      IN RHCW REGISTER.
4027      :*      USE A PATTERN OF WALKING 1'S (1,2,4,10 ETC)
4028      :*      AND WALKING 0'S (-2,-3,-5 ETC)
4029      :*      IN THIS TEST 0 BITS WILL NOT BE WRITTEN INTO
4030      :*      AND 0 BITS WILL ALWAYS BE SET
4031      :*      IF SWITCH 14 OR 9 OR 8 ARE SET FOR DEBUGGING
4032      :*      THEN A RH CLEAR (RHCS2 BIT #5) WILL BE
4033      :*      GIVEN TO AID SCOPE SYNC'S ON THE CLEAR
4034      :*      SIGNAL.
4035
4036      :*****
4037      TST6:  SCOPE
4038      014710 000004      MOV      #STACK,SP      ;RESET STACK
4039      014712 012706 001000      MOV      #6,@#TSTNM      ;SAVE TEST NUMBER
4040      014716 012737 000006 006276
4041      014724 004737 041222      JSR      PC,@#CLDISK      ;GIVE RH INITIALIZE
4042      ;SETUP UNIT NUBER
4043      ;CLEAR RHCW AND FUNCTION BITS IN RHCS1
4044
4045      014730 012737 000000 006270      MOV      #0,WRTBIT      ;0 ARE BITS NOT WRITTEN INTO
4046      014736 012737 000000 006272      MOV      #0,SETBIT      ;0 ARE BITS ALWAYS SET
4047
4048      ;FLOAT 1'S THRU THE RHCW REGISTER
4049      014744 012737 014770 001110      MOV      #2$, $LPERR      ;SET LOOP ON ERROR ADDRESS
4050      014752 012705 000001      MOV      #1,R5      ;GETTING READY TO FLOAT A ONE
4051      014756 010537 001124      1$:  MOV      R5,$GDDAT      ;START WITH DATA
4052      014762 042737 000000 001124      BIC      #0,$GDDAT      ;CLEAR BITS NOT WRITTEN INTO
4053      014770 032777 041400 164142      2$:  BIT      #BIT14!BIT9!BIT8,@SWR ;ARE SWITCHES 14 OR 9 OR 8 SET
4054      014776 001406      BEQ      3$      ;BRANCH IF ANY ONE SET
4055      015000 052777 000040 167062      BIS      #CLR,@RHCS2      ;GIVE CLEAR FOR SCOPE
4056      ;SYNC IF IN DEBUG:
4057      015006 013777 006442 167054      MOV      UNIT,@RHCS2
4058      015014 013777 001124 167040      3$:  MOV      $GDDAT,@RHCW      ;WRITE RHCW REGISTER
4059      015022 017737 167034 001126      MOV      @RHCW,$BDDAT      ;READ RHCW REGISTER
4060      015030 053737 006272 001124      BIS      SETBIT,$GDDAT      ;SET ALWAYS SET BITS
4061      015036 023737 001124 001126      CMP      $GDDAT,$BDDAT      ;TEST
4062      015044 001404      BEQ      4$      ;BRANCH IF GOOD
4063      015046 013737 004062 001122      MOV      RHCW,$BDDADR      ;GET REGISTER ADDRESS
4064      015054 104137      ERROR    137      ;ONE WAS BEING FLOATED
4065      ;THRU RHCW REGISTER
4066      ;ON READING RHCW BACK
4067      ;IT DID NOT CONTAIN WHAT
4068      ;WAS EXPECTED.
4069      015056 000241      4$:  CLC      ;CLEAR CARRY
4070      015060 006305      ASL      R5      ;GET 1 ONE LEFT
4071      015062 103335      BCC      1$      ;BRANCH IF 16 NOT DONE
4072
4073      ;FLOAT A ZERO THRU RHCW REGISTER.
4074
4075      015064 012737 015110 001110      MOV      #6$, $LPERR      ;SET LOOP BACK POINT.
4076      015072 012705 177776      MOV      #177776,R5      ;GET READY TO FLOAT A ZERO
4077      015076 010537 001124      5$:  MOV      R5,$GDDAT      ;GET READY TO WRITE DATA
4078      015102 042737 000000 001124      BIC      #0,$GDDAT      ;CLEAR BITS NOT WRITTEN INTO
4079      015110 032777 041400 164022      6$:  BIT      #BIT14!BIT9!BIT8,@SWR ;ARE SWITCHES 14 OR 9 OR 8 SET
    
```

```

4080 015116 001406          BEQ      7$          ;BRANCH IF ANY OF THE ABOVE SET
4081 015120 012777 000040 166742  MOV      #CLR,@RHCS2 ;CLEAR FOR SCOPE AID TO
4082                                     ;SYNC IF IN DEBUG
4083 015126 013777 006442 166734  MOV      UNIT,@RHCS2
4084 015134 013777 001124 166720 7$:      MOV      $GDDAT,@RHWC ;WRITE INTO RH'XA.
4085 015142 017737 166714 001126  MOV      @RHWC,$BDDAT ;READ RHWC
4086 015150 053737 006272 001124  BIS      SETBIT,$GDDAT ;SET BITS ALWAYS SET
4087 015156 023737 001124 001126  CMP      $GDDAT,$BDDAT ;TEST
4088 015164 001404          BEQ      8$          ;BRANCH IF GOOD
4089 015166 013737 004062 001122  MOV      RHWC,$BDDADR ;GET REGISTER ADDRESS
4090 015174 104137          ERROR    137        ;ZERO WAS BEING FLOATED
4091                                     ;THRU RHWC REGISTER
4092                                     ;ON READING IT BACK IT
4093                                     ;DID NOT CONTAIN WHAT
4094                                     ;WAS EXPECTED
4095
4096 015176 000261          8$:      SEC
4097 015200 006105          ROL      R5
4098 015202 103735          BCS      5$
4099
4100
4101 .....
4102 :*TEST 7          BIT BANG RHBA
4103
4104 :*          TEST LOADING AND READING OF ALL POSSIBLE BITS
4105 :*          IN RHBA REGISTER.
4106 :*          USE A PATTERN OF WALKING 1'S (1,2,4,10 ETC)
4107 :*          AND WALKING 0'S (-2,-3,-5 ETC)
4108 :*          IN THIS TEST 0 BITS WILL NOT BE WRITTEN INTO
4109 :*          AND 0 BITS WILL ALWAYS BE SET
4110 :*          AND BIT00 BITS WILL ALWAYS BE CLEARED
4111 :*          IF SWITCH 14 OR 9 OR 8 ARE SET FOR DEBUGGING
4112 :*          THEN A RH CLEAR (RHCS2 BIT #5) WILL BE
4113 :*          GIVEN TO AID SCOPE SYNC'S ON THE CLEAR
4114 :*          SIGNAL.
4115
4116 .....
4117 TST7:          SCOPE
4118 015204 000004          MOV      #STACK,SP ;RESET STACK
4119 015206 012706 001000  MOV      #7,@TSTNM ;SAVE TEST NUMBER
4120 015212 012737 000007 006276
4121 015220 004737 041222          JSR      PC,@CLDISK ;GIVE RH INITIALIZE
4122                                     ;SETUP UNIT NUBER
4123                                     ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
4124
4125 015224 012737 000000 006270  MOV      #0,WRTBIT ;0 ARE BITS NOT WRITTEN INTO
4126 015232 012737 000000 006272  MOV      #0,SETBIT ;0 ARE BITS ALWAYS SET
4127 015240 012737 000001 006274  MOV      #BIT00,CLRBIT ;BIT00 ARE BITS ALWAYS CLEARED
4128
4129 :FLOAT 1'S THRU THE RHBA REGISTER
4130 015246 012737 015272 001110  MOV      #2$,SLPERR ;SET LOOP ON ERROR ADDRESS
4131 015254 012705 000001          MOV      #1,R5 ;GETTING READY TO FLOAT A ONE
4132 015260 010537 001124          MOV      R5,$GDDAT ;START WITH DATA
4133 015264 042737 000000 001124 1$:      BIC      #0,$GDDAT ;CLEAR BITS NOT WRITTEN INTO
4134 015272 032777 041400 163640 2$:      BIT      #BIT14!BIT9!BIT8,@SWR ;ARE SWITCHES 14 OR 9 OR 8 SET
4135 015300 001406          BEQ      3$          ;BRANCH IF ANY ONE SET
    
```


BIT BANG RHBA

```

4136 015302 052777 000040 166560      BIS      #CLR,@RHCS2      ;GIVE CLEAR FOR SCOPE
4137                                     ;SYNC IF IN DEBUG:
4138 015310 013777 006442 166552      MOV      UNIT,@RHCS2
4139 015316 013777 001124 166540 3$:    MOV      $GDDAT,@RHBA    ;WRITE RHBA REGISTER
4140 015324 017737 166534 001126      MOV      @RHBA,$BDDAT    ;READ RHBA REGISTER
4141 015332 043737 006274 001124      BIC      CLRBIT,$GDDAT   ;CLEAR ALWAYS CLEARED BITS
4142 015340 053737 006272 001124      BIS      SETBIT,$GDDAT   ;SET ALWAYS SET BITS
4143 015346 023737 001124 001126      CMP      $GDDAT,$BDDAT   ;TEST
4144 015354 001404                                     BEQ      4$              ;BRANCH IF GOOD
4145 015356 013737 004064 001122      MOV      RHBA,$BDADR     ;GET REGISTER ADDRESS
4146 015364 104137                                     ERROR    137            ;ONE WAS BEING FLOATED
4147                                     ;THRU RHBA REGISTER
4148                                     ;ON READING RHBA BACK
4149                                     ;IT DID NOT CONTAIN WHAT
4150                                     ;WAS EXPECTED.
4151 015366 000241 4$:    CLC                                     ;CLEAR CARRY
4152 015370 006305      ASL      R5              ;GET 1 ONE LEFT
4153 015372 103332      BCC      1$              ;BRANCH IF 16 NOT DONE
4154
4155                                     ;FLOAT A ZERO THRU RHBA REGISTER.
4156
4157 015374 012737 015420 001110      MOV      #6$,$LPERR     ;SET LOOP BACK POINT.
4158 015402 012705 177776                                     MOV      #177776,R5     ;GET READY TO FLOAT A ZERO
4159 015406 010537 001124 5$:    MOV      R5,$GDDAT      ;GET READY TO WRITE DATA
4160 015412 042737 000000 001124      BIC      #0,$GDDAT      ;CLEAR BITS NOT WRITTEN INTO
4161 015420 032777 041400 163512 6$:    BIT      #BIT14!BIT9!BIT8,@SWR ;ARE SWITCHES 14 OR 9 OR 8 SET
4162 015426 001406                                     BEQ      7$              ;BRANCH IF ANY OF THE ABOVE SET
4163 015430 012777 000040 166432      MOV      #CLR,@RHCS2    ;CLEAR FOR SCOPE AID TO
4164                                     ;SYNC IF IN DEBUG
4165 015436 013777 006442 166424      MOV      UNIT,@RHCS2
4166 015444 013777 001124 166412 7$:    MOV      $GDDAT,@RHBA    ;WRITE INTO RH'XA.
4167 015452 017737 166406 001126      MOV      @RHBA,$BDDAT   ;READ RHBA
4168 015460 053737 006272 001124      BIS      SETBIT,$GDDAT   ;SET BITS ALWAYS SET
4169 015466 043737 006274 001124      BIC      CLRBIT,$GDDAT   ;CLEAR BITS ALWAYS 0
4170 015474 023737 001124 001126      CMP      $GDDAT,$BDDAT   ;TEST
4171 015502 001404                                     BEQ      8$              ;BRANCH IF GOOD
4172 015504 013737 004064 001122      MOV      RHBA,$BDADR     ;GET REGISTER ADDRESS
4173 015512 104137                                     ERROR    137            ;ZERO WAS BEING FLOATED
4174                                     ;THRU RHBA REGISTER
4175                                     ;ON READING IT BACK IT
4176                                     ;DID NOT CONTAIN WHAT
4177                                     ;WAS EXPECTED
4178
4179 015514 000261 8$:    SEC
4180 015516 006105      ROL      R5
4181 015520 103732      BCS      5$
4182
4183
4184
4185
4186
4187
4188
4189
4190
4191

```

 ;*TEST 10 BIT BANG RHBAE
 ;* 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)
 ;* IN THIS TEST 177700 BITS WILL NOT BE WRITTEN INTO

```

4192          : *      AND 0 BITS WILL ALWAYS BE SET
4193          : *      AND 177700 BITS WILL ALWAYS BE CLEARED
4194          : *      IF SWITCH 14 OR 9 OR 8 ARE SET FOR DEBUGGING
4195          : *      THEN A RH CLEAR (RHCS2 BIT #5) WILL BE
4196          : *      GIVEN TO AID SCOPE SYNC'S ON THE CLEAR
4197          : *      SIGNAL.
4198
4199          :*****
4200 015522 000004          TST10: SCOPE
4201 015524 012706 001000      MOV      #STACK,SP          ;RESET STACK
4202 015530 012737 000010 006276  MOV      #10,@#TSTNM        ;SAVE TEST NUMBER
4203
4204 015536 004737 041222      JSR      PC,@#CLDISK        ;GIVE RH INITIALIZE
4205                                ;SETUP UNIT NUBER
4206                                ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
4207
4208 015542 012737 177700 006270  MOV      #177700,WRTBIT      ;177700 ARE BITS NOT WRITTEN INTO
4209 015550 012737 000000 006272  MOV      #0,SETBIT           ;0 ARE BITS ALWAYS SET
4210 015556 012737 177700 006274  MOV      #177700,CLRBIT      ;177700 ARE BITS ALWAYS CLEARED
4211
4212          :FLOAT 1'S THRU THE RHBAE REGISTER
4213 015564 012737 015610 001110  MOV      #2$,SLPERR          ;SET LOOP ON ERROR ADDRESS
4214 015572 012705 000001          MOV      #1,R5               ;GETTING READY TO FLOAT A ONE
4215 015576 010537 001124          1$:  MOV      R5,$GDDAT           ;START WITH DATA
4216 015602 042737 177700 001124  BIC      #177700,$GDDAT      ;CLEAR BITS NOT WRITTEN INTO
4217 015610 032777 041400 163322  2$:  BIT      #BIT14!BIT9!BIT8,@SWR ;ARE SWITCHES 14 OR 9 OR 8 SET
4218 015616 001406          BEQ      3$                  ;BRANCH IF ANY ONE SET
4219 015620 052777 000040 166242  BIS      #CLR,@RHCS2         ;GIVE CLEAR FOR SCOPE
4220                                ;SYNC IF IN DEBUG:
4221 015626 013777 006442 166234  MOV      UNIT,@RHCS2
4222 015634 013777 001124 166266  3$:  MOV      $GDDAT,@RHBAE       ;WRITE RHBAE REGISTER
4223 015642 017737 166262 001126  MOV      @RHBAE,$BDDAT       ;READ RHBAE REGISTER
4224 015650 043737 006274 001124  BIC      CLRBIT,$GDDAT       ;CLEAR ALWAYS CLEARED BITS
4225 015656 053737 006272 001124  BIS      SETBIT,$GDDAT       ;SET ALWAYS SET BITS
4226 015664 023737 001124 001126  CMP      $GDDAT,$BDDAT       ;TEST
4227 015672 001404          BEQ      4$                  ;BRANCH IF GOOD
4228 015674 013737 004130 001122  MOV      RHBAE,$BDDADR       ;GET REGISTER ADDRESS
4229 015702 104137          ERROR 137                   ;ONE WAS BEING FLOATED
4230                                ;THRU RHBAE REGISTER
4231                                ;ON READING RHBAE BACK
4232                                ;IT DID NOT CONTAIN WHAT
4233                                ;WAS EXPECTED.
4234 015704 000241          4$:  CLC                          ;CLEAR CARRY
4235 015706 006305          ASL      R5                  ;GET 1 ONE LEFT
4236 015710 103332          BCC      1$                  ;BRANCH IF 16 NOT DONE
4237
4238          :FLOAT A ZERO THRU RHBAE REGISTER.
4239
4240 015712 012737 015736 001110  MOV      #6$,SLPERR          ;SET LOOP BACK POINT.
4241 015720 012705 177776          MOV      #177776,R5         ;GET READY TO FLOAT A ZERO
4242 015724 010537 001124          5$:  MOV      R5,$GDDAT           ;GET READY TO WRITE DATA
4243 015730 042737 177700 001124  BIC      #177700,$GDDAT      ;CLEAR BITS NOT WRITTEN INTO
4244 015736 032777 041400 163174  6$:  BIT      #BIT14!BIT9!BIT8,@SWR ;ARE SWITCHES 14 OR 9 OR 8 SET
4245 015744 001406          BEQ      7$                  ;BRANCH IF ANY OF THE ABOVE SET
4246 015746 012777 000040 166114  MOV      #CLR,@RHCS2         ;CLEAR FOR SCOPE AID TO
4247                                ;SYNC IF IN DEBUG

```



```

4248 015754 013777 006442 166106      MOV    UNIT,@RHCS2
4249 015762 013777 001124 166140 7$:    MOV    $GDDAT,@RHBAE      ;WRITE INTO RH'XA.
4250 015770 017737 166134 001126      MOV    @RHBAE,$BDDAT     ;READ RHBAE
4251 015776 053737 006272 001124      BIS    SETBIT,$GDDAT     ;SET BITS ALWAYS SET
4252 016004 043737 006274 001124      BIC   CLRBIT,$GDDAT     ;CLEAR BITS ALWAYS 0
4253 016012 023737 001124 001126      CMP    $GDDAT,$BDDAT    ;TEST
4254 016020 001404      BEQ    8$                ;BRANCH IF GOOD
4255 016022 013737 004130 001122      MOV    RHBAE,$BDADR     ;GET REGISTER ADDRESS
4256 016030 104137      ERROR  137              ;ZERO WAS BEING FLOATED
4257                                     ;THRU RHBAE REGISTER
4258                                     ;ON READING IT BACK IT
4259                                     ;DID NOT CONTAIN WHAT
4260                                     ;WAS EXPECTED
    
```

```

4261
4262 016032 000261      8$:    SEC
4263 016034 006105      ROL    R5
4264 016036 103732      BCS    5$
    
```

```

*****
*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"
*      "DR" (BIT #7) IS WAITED FOR BY A TRAP INSTRUCTION
*      CALLED "WAT" (NO TIMING IS DONE)
*      HOWEVER IF "DR" 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
*****
    
```

```

4286
4287 016040 000004      TST11: SCOPE
4288 016042 012706 001000      MOV    #STACK,SP      ;RESET STACK
4289 016046 012737 000011 006276      MOV    #11,@TSTNM     ;SAVE TEST NUMBER
4290
4291 016054 004737 041222      JSR    PC,@#CLDISK    ;GIVE RH INITIALIZE
4292                                     ;SETUP UNIT NUBER
4293                                     ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
4294
4295                                     ;CHECK THAT RHCS2 HAS IR
4296 016060 012737 000100 001124      MOV    #IR,$GDDAT     ;GET GOOD = 100
4297 016066 053737 006442 001124      BIS    UNIT,$GDDAT    ;INCLUDE UNIT NUMBER
4298
4299 016074 017737 165770 001126      MOV    @RHCS2,$BDDAT  ;READ RHCS2 FOR COMPARISON
4300 016102 023737 001124 001126      CMP    $GDDAT,$BDDAT  ;COMPARE EXPECTED
4301                                     ;DATA WITH DATA READ FROM
4302                                     ;RHCS2
4303 016110 001401      BEQ    65$            ;BRANCH IF GOOD
    
```

```

4304 016112 104006          ERROR 6
4305
4306
4307
4308
4309
4310
4311
4312
4313 016114          65$:
4314
4315
4316 016114 005077 165762
4317
4318
4319 016120 104411
4320 016122 004070
4321 016124 000200
4322
4323
4324
4325
4326
4327
4328
4329
4330
4331 016126 012737 000000 001124
4332
4333 016134 017737 165742 001126
4334 016142 023737 001124 001126
4335
4336
4337 016150 001401
4338 016152 104007
4339
4340
4341
4342
4343
4344
4345
4346
4347 016154          67$:
4348
4349 016154 012737 000100 001124
4350 016162 053737 006442 001124
4351
4352 016170 017737 165674 001126
4353 016176 023737 001124 001126
4354
4355
4356 016204 001401
4357 016206 104010
4358
4359

;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

;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
RHCS2 ;AND WAIT FOR OR BIT IN
OR ;RHCS2 REGISTER
;IF ERROR OCCURS HERE
;IT MEANS 'OR' DID NOT
;SET FOR THE FULL COUNT
;DOWN OF THE WAIT.T SUBROUTINE
;THIS TIME IS APPROXIMATELY
;LARGER THAN 500 MILLISECONDS

;CHECK THAT RHDB HAS 0
MOV #0,$GDDAT ;GET GOOD =
MOV @RHDB,$BDDAT ;READ RHDB FOR COMPARISON
CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
;DATA WITH DATA READ FROM
;RHDB
BEQ 67$ ;BRANCH IF GOOD
ERROR 7

;AFTER CLEARING THE RH
;AND WRITING ALL ZEROS
;INTO RHDB
;THEN READING IT BACK
;RHDB SHOULD HAVE 0
;=
;BUT CONTAINED WHAT IS
;GIVEN IN BAD RHDB

;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 69$ ;BRANCH IF GOOD
ERROR 10

;AFTER CLEARING THE RH
;AND WRITING ALL ZEROS
    
```



```
4416                                     :RHBA SHOULD HAVE 0
4417                                     :=
4418                                     :BUT CONTAINED WHAT IS
4419                                     :GIVEN IN BAD RHBA
4420 016312                               75$:
4421                                     :CHECK THAT RHBAE HAS 0
4422 016312 012737 000000 001124         MOV    #0,$GDDAT    :GET GOOD =
4423                                     :
4424 016320 017737 165604 001126         MOV    @RHBAE,$BDDAT :READ RHBAE FOR COMPARISON
4425 016326 023737 001124 001126         CMP    $GDDAT,$BDDAT :COMPARE EXPECTED
4426                                     :DATA WITH DATA READ FROM
4427                                     :RHBAE
4428 016334 001401                         BEQ    77$          :BRANCH IF GOOD
4429 016336 104014                         ERROR  14
4430                                     :AFTER CLEARING THE RH
4431                                     :AND WRITING ALL ZEROS
4432                                     :INTO RHDB
4433                                     :THEN READING IT BACK
4434                                     :RHBAE SHOULD HAVE 0
4435                                     :=
4436                                     :BUT CONTAINED WHAT IS
4437                                     :GIVEN IN BAD RHBAE
4438 016340                               77$:
4439                                     :CHECK THAT RHWC HAS 0
4440 016340 012737 000000 001124         MOV    #0,$GDDAT    :GET GOOD =
4441                                     :
4442 016346 017737 165510 001126         MOV    @RHWC,$BDDAT :READ RHWC FOR COMPARISON
4443 016354 023737 001124 001126         CMP    $GDDAT,$BDDAT :COMPARE EXPECTED
4444                                     :DATA WITH DATA READ FROM
4445                                     :RHWC
4446 016362 001401                         BEQ    79$          :BRANCH IF GOOD
4447 016364 104015                         ERROR  15
4448                                     :AFTER CLEARING THE RH
4449                                     :AND WRITING ALL ZEROS
4450                                     :INTO RHDB
4451                                     :THEN READING IT BACK
4452                                     :RHWC SHOULD HAVE 0
4453                                     :=
4454                                     :BUT CONTAINED WHAT IS
4455                                     :GIVEN IN BAD RHWC
4456 016366                               79$:
```

```
4457
4458
4459
4460 *****
4461 *TEST 12      SILO TEST 2 (ONE WORD WRITE)
4462 *      AFTER A RH CLEAR IS GIVEN (SET BIT #5 IN RHCS2)
4463 *      RHCS2 IS CHECKED TO HAVE "IR" (BIT #6) HIGH
4464 *      TOGETHER WITH UNIT NUMBER
4465 *      ONE WORD OF ALL ONES IS WRITTEN INTO RHDB
4466 *      BY "CLR"
4467 *      "OR" (BIT #7) IS WAITED FOR BY A TRAP INSTRUCTION
4468 *      CALLED "WAT" (NO TIMING IS DONE)
4469 *      HOWEVER IF "OR" DOES NOT SET WITHIN "WAT" COUNT
4470 *      DOWN AN ERROR IS REPORTED
4471 *      RHDB IS READ AND CHECKED TO CONTAIN ALL ONES
```



```
4472      :*      RHCS2 WILL BE CHECKED TO HAVE 'IR' AND UNIT NUMBER
4473      :*      RHCS1,RHCS3,RHBA,RHBAE,RHWC, WILL BE CHECKED TO HAVE
4474      :*      APPROPRIATE VALUES
4475      :*****
4476 016366 000004      TST12: SCOPE
4477 016370 012706 001000      MOV      #STACK,SP      ;RESET STACK
4478 016374 012737 000012 006276      MOV      #12,@#TSTNM    ;SAVE TEST NUMBER
4479
4480 016402 004737 041222      JSR      PC,@#CLDISK    ;GIVE RH INITIALIZE
4481                                     ;SETUP UNIT NUBER
4482                                     ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
4483
4484
4485      :CHECK THAT RHCS2 HAS IR
4486 016406 012737 000100 001124      MOV      #IR,$GDDAT     ;GET GOOD = 100
4487 016414 053737 006442 001124      BIS      UNIT,$GDDAT    ;INCLUDE UNIT NUMBER
4488
4489 016422 017737 165442 001126      MOV      @RHCS2,$BDDAT  ;READ RHCS2 FOR COMPARISON
4490 016430 023737 001124 001126      CMP      $GDDAT,$BDDAT  ;COMPARE EXPECTED
4491                                     ;DATA WITH DATA READ FROM
4492                                     ;RHCS2
4493 016436 001401      BEQ      65$           ;BRANCH IF GOOD
4494 016440 104006
4495
4496      :AFTER SETTING CLR BIT #5
4497      :IN RHCS2 '0' INIT THE RH
4498      :AND HAV. 'G' DONE NOTHING ELSE
4499      :RHCS2 SHOULD HAVE IR
4500      :=100
4501      :TOGETHER WITH UNIT NUMBER
4502      :BUT CONTAINED WHAT IS
4503 016442      65$:
4504
4505      :WRITE ONE WORD OF ALL ONES INTO RHDB
4506 016442 012777 177777 165432      MOV      #177777,@RHDB ;WRITE 1 IN RHDB
4507
4508      :WAIT FOR OR BIT IN RHCS2 REGISTER TO SET
4509 016450 104411      WAT
4510 016452 004070      RHCS2      ;TRAP TO WAIT.T SUBROUTINE
4511 016454 000200      OR         ;AND WAIT FOR OR BIT IN
4512                                     ;RHCS2 REGISTER
4513                                     ;IF ERROR OCCURS HERE
4514                                     ;IT MEANS 'OR' DID NOT
4515                                     ;SET FOR THE FULL COUNT
4516                                     ;DOWN OF THE WAIT.T SUBROUTINE
4517                                     ;THIS TIME IS APPROXIMATELY
4518                                     ;LARGER THAN 500 MILLISECONDS
4519
4520      :CHECK THAT RHDB HAS 177777
4521 016456 012737 177777 001124      MOV      #177777,$GDDAT ;GET GOOD = 0
4522
4523 016464 017737 165412 001126      MOV      @RHDB,$BDDAT   ;READ RHDB FOR COMPARISON
4524 016472 023737 001124 001126      CMP      $GDDAT,$BDDAT  ;COMPARE EXPECTED
4525                                     ;DATA WITH DATA READ FROM
4526                                     ;RHDB
4527 016500 001401      BEQ      67$           ;BRANCH IF GOOD
```

4528	016502	104007			ERROR	7		
4529								:AFTER CLEARING THE RH
4530								:AND WRITING ALL ONES
4531								:INTO RHDB
4532								:THEN READING IT BACK
4533								:RHDB SHOULD HAVE 177777
4534								:BUT CONTAINED WHAT IS
4535								:GIVEN IN BAD RHDB
4536	016504						67\$:	
4537								:CHECK THAT RHCS2 HAS IR
4538	016504	012737	000100	001124	MOV	#IR,\$GDDAT		:GET GOOD = 100
4539	016512	053737	006442	001124	BIS	UNIT,\$GDDAT		:INCLUDE UNIT NUMBER
4540								
4541	016520	017737	165344	001126	MOV	@RHCS2,\$BDDAT		:READ RHCS2 FOR COMPARISON
4542	016526	023737	001124	001126	CMP	\$GDDAT,\$BDDAT		:COMPARE EXPECTED
4543								:DATA WITH DATA READ FROM
4544								:RHCS2
4545	016534	001401			BEQ	69\$:BRANCH IF GOOD
4546	016536	104010			ERROR	10		
4547								:AFTER CLEARING THE RH
4548								:AND WRITING ALL ONES
4549								:INTO RHDB
4550								:THEN READING IT BACK
4551								:RHCS2 SHOULD HAVE IR
4552								: =100
4553								:TOGETHER WITH UNIT NUMBER
4554								:BUT CONTAINED WHAT IS
4555								:GIVEN IN BAD RHCS2
4556	016540						69\$:	
4557								:CHECK THAT RHCS3 HAS 0
4558	016540	012737	000000	001124	MOV	#0,\$GDDAT		:GET GOOD =
4559								
4560	016546	017737	165360	001126	MOV	@RHCS3,\$BDDAT		:READ RHCS3 FOR COMPARISON
4561	016554	023737	001124	001126	CMP	\$GDDAT,\$BDDAT		:COMPARE EXPECTED
4562								:DATA WITH DATA READ FROM
4563								:RHCS3
4564	016562	001401			BEQ	71\$:BRANCH IF GOOD
4565	016564	104012			ERROR	12		
4566								:AFTER CLEARING THE RH
4567								:AND WRITING ALL ONES
4568								:INTO RHDB
4569								:THEN READING IT BACK
4570								:RHCS3 SHOULD HAVE 0
4571								: =
4572								:BUT CONTAINED WHAT IS
4573								:GIVEN IN BAD RHCS3
4574	016566						71\$:	
4575								:CHECK THAT RHBA HAS 0
4576	016566	012737	000000	001124	MOV	#0,\$GDDAT		:GET GOOD =
4577								
4578	016574	017737	165264	001126	MOV	@RHBA,\$BDDAT		:READ RHBA FOR COMPARISON
4579	016602	023737	001124	001126	CMP	\$GDDAT,\$BDDAT		:COMPARE EXPECTED
4580								:DATA WITH DATA READ FROM
4581								:RHBA
4582	016610	001401			BEQ	73\$:BRANCH IF GOOD
4583	016612	104013			ERROR	13		


```

4584                                     ;AFTER CLEARING THE RH
4585                                     ;AND WRITING ALL ONES
4586                                     ;INTO RHDB
4587                                     ;THEN READING IT BACK
4588                                     ;RHBA SHOULD HAVE 0
4589                                     ;=
4590                                     ;BUT CONTAINED WHAT IS
4591                                     ;GIVEN IN BAD RHBA
4592 016614                               73$:
4593                                     ;CHECK THAT RHBAE HAS 0
4594 016614 012737 000000 001124         MOV    #0,$GDDAT    ;GET GOOD =
4595
4596 016622 017737 165302 001126         MOV    @RHBAE,$BDDAT ;READ RHBAE FOR COMPARISON
4597 016630 023737 001124 001126         CMP    $GDDAT,$BDDAT ;COMPARE EXPECTED
4598                                     ;DATA WITH DATA READ FROM
4599                                     ;RHBAE
4600 016636 001401                         BEQ    75$          ;BRANCH IF GOOD
4601 016640 104014                         ERROR  14
4602                                     ;AFTER CLEARING THE RH
4603                                     ;AND WRITING ALL ONES
4604                                     ;INTO RHDB
4605                                     ;THEN READING IT BACK
4606                                     ;RHBAE SHOULD HAVE 0
4607                                     ;=
4608                                     ;BUT CONTAINED WHAT IS
4609                                     ;GIVEN IN BAD RHBAE
4610 016642                               75$:
4611                                     ;CHECK THAT RHWC HAS 0
4612 016642 012737 000000 001124         MOV    #0,$GDDAT    ;GET GOOD =
4613
4614 016650 017737 165206 001126         MOV    @RHWC,$BDDAT ;READ RHWC FOR COMPARISON
4615 016656 023737 001124 001126         CMP    $GDDAT,$BDDAT ;COMPARE EXPECTED
4616                                     ;DATA WITH DATA READ FROM
4617                                     ;RHWC
4618 016664 001401                         BEQ    77$          ;BRANCH IF GOOD
4619 016666 104015                         ERROR  15
4620                                     ;AFTER CLEARING THE RH
4621                                     ;AND WRITING ALL ONES
4622                                     ;INTO RHDB
4623                                     ;THEN READING IT BACK
4624                                     ;RHWC SHOULD HAVE 0
4625                                     ;=
4626                                     ;BUT CONTAINED WHAT IS
4627                                     ;GIVEN IN BAD RHWC
4628 016670                               77$:

```

```

4631 .....
4632 ;*TEST 13      SILO TEST 3 (TWO WORD WRITE)
4633
4634 ;*      AFTER A RH CLEAR IS GIVEN (SET BIT #5 IN RHCS2)
4635 ;*      RHCS2 IS CHECKED TO HAVE "IR" (BIT #6) HIGH.
4636 ;*      TOGETHER WITH UNIT NUMBER
4637 ;*      ONE WORD = 52525 IS WRITTEN INTO RHDB
4638 ;*      RHCS2 IS CHECKED TO HAVE "IR" AND UNIT NUMBER
4639 ;*      A SECOND WORD = 12525 IS WRITTEN INTO RHDB

```

```

4640 :* 'OR' (BIT #7 IN RHCS2) IS WAITED FOR BY A TRAP
4641 :* INSTRUCTION CALLED 'WAT'
4642 :* RHDB IS READ AND CHECKED TO CONTAIN 52525
4643 :* RHCS2 IS CHECKED TO HAVE 'IR', 'OR' AND UNIT NUMBER
4644 :* RHDB IS READ A SECOND TIME AND CHECKED TO
4645 :* CONTAIN 125252
4646 :* RHCS2 IS CHECKED TO HAVE 'IR' AND UNIT NUMBER
4647 :* THEN ALL REGISTERS RHCS1, RHCS3, RHBA, RHBAE, RHWC
4648 :* ARE CHECKED TO HAVE APPROPRIATE VALUE
4649
4650
4651

```

```

4652 016670 000004          TST13: SCOPE
4653 016672 012706 001000  MOV    #STACK,SP      ;RESET STACK
4654 016676 012737 000013 006276  MOV    #13,@#TSTNM    ;SAVE TEST NUMBER
4655
4656 016704 004737 041222    JSR    PC,@#CLDISK    ;GIVE RH INITIALIZE
4657                                ;SETUP UNIT NUBER
4658                                ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
4659
4660                                ;CHECK THAT RHCS2 HAS IR
4661 016710 012737 000100 001124  MOV    #IR,$GDDAT     ;GET GOOD = 100
4662 016716 053737 006442 001124  BIS    UNIT,$GDDAT    ;INCLUDE UNIT NUMBER
4663
4664 016724 017737 165140 001126  MOV    @RHCS2,$BDDAT  ;READ RHCS2 FOR COMPARISON
4665 016732 023737 001124 001126  CMP    $GDDAT,$BDDAT ;COMPARE EXPECTED
4666                                ;DATA WITH DATA READ FROM
4667                                ;RHCS2
4668 016740 001401          BEQ    65$            ;BRANCH IF GOOD
4669 016742 104006
4670                                ;AFTER SETTING 'CLR' BIT #5
4671                                ;IN RHCS2 TO INIT THE RH
4672                                ;AND HAVING DONE NOTHING ELSE
4673
4674                                ;RHCS2 SHOULD HAVE IR
4675                                ;=100
4676                                ;TOGETHER WITH UNIT NUMBER
4677                                ;BUT CONTAINED WHAT IS
4678                                ;GIVEN IN BAD RHCS2
4679 016744          65$:
4680
4681                                ;WRITE ONE WORD = 52525 INTO RHDB
4682 016744 012777 052525 165130  MOV    #52525,@RHDB  ;WRITE IN RHDB
4683
4684                                ;WAIT FOR OR BIT IN RHCS2 REGISTER TO SET
4685 016752 104411          WAT                                ;TRAP TO WAIT.T SUBROUTINE
4686 016754 004070          RHCS2                            ;AND WAIT FOR OR BIT IN
4687 016756 000200          OR                                ;RHCS2 REGISTER
4688                                ;IF ERROR OCCURS HERE
4689                                ;IT MEANS 'OR' DID NOT
4690                                ;SET FOR THE FULL COUNT
4691                                ;DOWN OF THE WAIT.T SUBROUTINE
4692                                ;THIS TIME IS APPROXIMATELY
4693                                ;LARGER THAN 500 MILLISECONDS
4694
4695

```



```

4864 017276      81$:
4865
4866 017276 012737 000000 001124      :CHECK THAT RHBAE HAS 0
4867
4868 017304 017737 164620 001126      MOV      @RHBAE,$BDDAT      ;GET GOOD = 0
4869 017312 023737 001124 001126      MOV      @RHBAE,$BDDAT      ;READ RHBAE FOR COMPARISON
4870
4871
4872 017320 001401
4873 017322 104144
4874
4875
4876
4877
4878
4879
4880
4881
4882
4883
4884 017324      83$:
4885
4886 017324 012737 000000 001124      :CHECK THAT RHWC HAS 0
4887
4888 017332 017737 164524 001126      MOV      @RHWC,$BDDAT      ;GET GOOD = 0
4889 017340 023737 001124 001126      MOV      @RHWC,$BDDAT      ;READ RHWC FOR COMPARISON
4890
4891
4892
4893
4894
4895
4896
4897
4898
4899
4900 017346 001401
4901 017350 104145
4902
4903
4904
4905
4906
4907
4908
4909
4910
4911
4912
4913
4914
4915
4916
4917
4918
4919

```

```

      BEQ      83$
      ERROR   144
      ;AFTER CLEARING RH THEN
      ;WRITING TWO WORDS INTO
      ;RHDB 52525 AND 125252
      ;THEN READING RHDB THE
      ;SECOND TIME
      ;RHBAE SHOULD HAVE 0
      ;BUT CONTAINED WHAT IS
      ;GIVEN IN BAD RHBAE

      BEQ      85$
      ERROR   145
      ;AFTER CLEARING RH THEN
      ;WRITING TWO WORDS INTO
      ;RHDB 52525 AND 125252
      ;THEN READING RHDB THE
      ;SECOND TIME
      ;RHWC SHOULD HAVE 0
      ;BUT CONTAINED WHAT IS
      ;GIVEN IN BAD RHWC

```

```

4900 017352      85$:
4901
4902
4903
4904
4905
4906
4907
4908
4909
4910
4911
4912
4913
4914
4915
4916
4917
4918
4919

```

```

*****
*TEST 14      SILO TEST 4 (SILO SIZE AND 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
*          A COUNT PATTERN 0,1,2,... IS WRITTEN INTO RHDB
*          UNTIL "IR" IS LOW IN RHCS2. THIS DETERMINES
*          THE WORD SIZE OF THE SILO, WHICH IS
*          THEN TYPED OUT (IN DECIMAL).
*          "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 RHBIS READ AND COMPARED TO HAVE THE RIGHT VALUE
*          AFTER EACH OF THE READS.
*          THEN RHCS2, RHCS1, RHCS3, RHBA, RHBAE, RHWC ARE

```



```

4920                                     :*
4921                                     :***** CHECKED TO HAVE THE APPROPRIATE VALUE
4922 017352 000004 TST14: SCOPE
4923 017354 012706 001000 MOV #STACK,SP ;RESET STACK
4924 017360 012737 000014 006276 MOV #14,@#TSTNM ;SAVE TEST NUMBER
4925
4926 017366 004737 041222 JSR PC,@#CLDISK ;GIVE RH INITIALIZE
4927                                     ;SETUP UNIT NUBER
4928                                     ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
4929
4930                                     ;CHECK THAT RHCS2 HAS IR
4931 017372 012737 000100 001124 MOV #IR,$GDDAT ;GET GOOD = 100
4932 017400 053737 006442 001124 BIS UNIT,$GDDAT ;INCLUDE UNIT NUMBER
4933
4934 017406 017737 164456 001126 MOV @RHCS2,$BDDAT ;READ RHCS2 FOR COMPARISON
4935 017414 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
4936                                     ;DATA WITH DATA READ FROM
4937                                     ;RHCS2
4938 017422 001401 BEQ 65$ ;BRANCH IF GOOD
4939 017424 104006 ERROR 6
4940                                     ;AFTER SETTING "CLR" BIT #5
4941                                     ;IN RHCS2 TO INIT THE RH
4942                                     ;AND HAVING DONE NOTHING ELSE
4943                                     ;RHCS2 SHOULD HAVE IR
4944                                     ;=100
4945                                     ;TOGETHER WITH UNIT NUMBER
4946                                     ;BUT CONTAINED WHAT IS
4947                                     ;GIVEN IN BAD RHCS2
4948 017426 65$:
4949
4950                                     ;WRITE INTO RHDB TO FILL IT
4951                                     ;THIS IS A COUNT PATTERN 0,1,2...
4952 017426 005037 006426 CLR SILOSZ ;SET UP TO COUNT SILO SIZE
4953 017432 005037 006430 CLR WDCT1
4954 017436 005037 006432 CLR WDCT2
4955 017442 005037 006434 CLR WDCT3
4956 017446 013777 006426 164426 1$: MOV SILOSZ,@RHDB ;PUT THE FIRST WORD IN
4957 017454 005237 006426 INC SILOSZ ;KEEP COUNT
4958 017460 032777 000100 164402 BIT #IR,@RHCS2 ;IS THE SILO FULL?
4959 017466 001367 BNE 1$ ;BRANCH IF NO
4960 017470 013737 006426 006430 MOV SILOSZ,WDCT1 ;SET UP 3 LOCATIONS FOR LATER USE
4961 017476 013737 006426 006432 MOV SILOSZ,WDCT2
4962 017504 013737 006426 006434 MOV SILOSZ,WDCT3
4963 017512 062737 000001 006430 ADD #1,WDCT1 ;THEY WILL BE USED
4964 017520 062737 000002 006432 ADD #2,WDCT2 ;TO SET RHWC
4965 017526 062737 000003 006434 ADD #3,WDCT3
4966 017534 005437 006430 NEG WDCT1
4967 017540 005437 006432 NEG WDCT2
4968 017544 005437 006434 NEG WDCT3
4969 017550 012701 017616 MOV #RH70A,R1 ;CHECKING FOR EXPECTED SILO SIZE
4970 017554 023721 006426 2$: CMP SILOSZ,(R1)+ ;IS THIS SIZE EXPECTED?
4971 017560 001405 BEQ 3$ ;BRANCH IF YES
4972 017562 022701 017622 CMP #RH70X,R1 ;DID WE CHECK ALL POSSIBILITIES?
4973 017566 103372 BHIS 2$ ;BRANCH IF NO
4974 017570 104150 ERROR 150 ;ELSE REPORT ERROR
4975 017572 000414 BR WAIT ;CONTINUE

```

```

4976 017574 005227 177777      3$:  INC      #-1                ;TYPE OUT THE SILO SIZE
4977 017600 001011                BNE      WAIT                ;DO IT ONLY ONCE
4978 017602 104401 067556        TYPE     ,SILMSG
4979 017606 013746 006426        MOV      SILOSZ, -(SP)
4980 017612 104405                TYPDS
4981 017614 000403                BR       WAIT                ;CONTINUE
4982
4983
4984 017616 000010                ;TABLE OF POSSIBLE SILO SIZES
4985 017620 000406                RH70A:  8.                   ;
4986 017622 000000                RH70C: 262.                  ;
4987 017624                        RH70X:  0                     ;FOR FUTURE USE
4988
4989                                ;WAIT FOR OR BIT IN RHCS2 REGISTER TO SET
4990 017624 104411                WAT
4991 017626 004070                RHCS2
4992 017630 000200                OR
4993                                ;TRAP TO WAIT.T SUBROUTINE
4994                                ;AND WAIT FOR OR BIT IN
4995                                ;RHCS2 REGISTER
4996                                ;IF ERROR OCCURS HERE
4997                                ;IT MEANS 'OR' DID NOT
4998                                ;SET FOR THE FULL COUNT
4999                                ;DOWN OF THE WAIT.T SUBROUTINE
5000                                ;THIS TIME IS APPROXIMATELY
5001                                ;LARGER THAN 500 MILLISECONDS
5002
5003                                ;CHECK THAT RHCS2 HAS OR
5004 017632 012737 000200 00:124    MOV      #OR,$GDDAT          ;GET GOOD = 200
5005 017640 053737 006442 001124    BIS      UNIT,$GDDAT        ;INCLUDE UNIT NUMBER
5006
5007                                ;READ RHCS2 FOR COMPARISON
5008 017646 017737 164216 001126    MOV      @RHCS2,$BDDAT      ;COMPARE EXPECTED
5009 017654 023737 001124 001126    CMP      $GDDAT,$BDDAT      ;DATA WITH DATA READ FROM
5010                                ;RHCS2
5011                                ;BRANCH IF GOOD
5012                                ;AFTER SETTING "CLR" BIT #5
5013                                ;IN RHCS2 TO INIT THE RH
5014                                ;AND HAVING WRITTEN A
5015                                ;COUNT PATTERN 0 THRU 7
5016                                ;INTO RHDB TO FILL IT
5017                                ;RHCS2 SHOULD HAVE OR
5018                                ;=200
5019                                ;TOGETHER WITH UNIT NUMBER
5020                                ;BUT CONTAINED WHAT IS
5021                                ;GIVEN IN BAD RHCS2
5022 017666                        65$:
5023                                ;READ RHDB UNTIL THE SILO IS EMPTY
5024                                ;VERIFY THE COUNT PATTERN AS READ FROM THE SILO
5025                                ;IS THE SAME PATTERN AS PREVIOUSLY WRITTEN INTO THE SILO
5026                                ;ELSE REPORT THE ERROR
5027 017666 005037 006304                CLR      SILONM
5028 017672 012737 000000 001124    MOV      #0, $GDDAT
5029 017700 005237 006304                INC      SILONM
5030 017704 023737 006304 006426    CMP      SILONM, SILOSZ
5031 017712 101013                BHI     68$
5032                                ;INITIALIZE SILO WORD #
5033                                ;INITIALIZE EXPECTED DATA
5034                                ;COUNT ONE SILO WORD
5035                                ;IS THE SILO EMPTY?
5036                                ;BRANCH IF YES

```



```

5032 017714 017737 164162 001126      MOV    @RHDB, $BDDAT      ;GET 0 WORD FROM THE SILO
5033 017722 023737 001124 001126      CMP    $GDDAT, $BDDAT    ;EXPECTED=ACTUAL?
5034 017730 001401                      BEQ    67$                ;BRANCH IF YES
5035 017732 104024                      ERROR  24                ;REPORT THE ERROR
5036 017734                      67$:
5037 017734 005237 001124      INC    $GDDAT            ;GET NEXT EXPECTED DATA
5038 017740 000757                      BR     66$                ;CONTINUE
5039 017742                      68$:
5040
5041
5042
5043                      ;CHECK THAT RHCS2 HAS IR
5044 017742 012737 000100 001124      MOV    #IR, $GDDAT      ;GET GOOD = 100
5045 017750 053737 006442 001124      BIS    UNIT, $GDDAT     ;INCLUDE UNIT NUMBER
5046
5047 017756 017737 164106 001126      MOV    @RHCS2, $BDDAT   ;READ RHCS2 FOR COMPARISON
5048 017764 023737 001124 001126      CMP    $GDDAT, $BDDAT  ;COMPARE EXPECTED
5049                      ;DATA WITH DATA READ FROM
5050                      ;RHCS2
5051 017772 001401                      BEQ    70$                ;BRANCH IF GOOD
5052 017774 104025                      ERROR  25
5053                      ;AFTER SETTING 'CLR' BIT #5
5054                      ;IN RHCS2 TO INIT THE RH
5055                      ;AND HAVING WRITTEN A
5056                      ;COUNT PATTERN 0 THRU 7
5057                      ;INTO RHDB TO FILL IT
5058                      ;RHCS2 SHOULD HAVE IR
5059                      ;=100
5060                      ;TOGETHER WITH UNIT NUMBER
5061                      ;BUT CONTAINED WHAT IS
5062                      ;GIVEN IN BAD RHCS2
5063 017776                      70$:
5064                      ;CHECK THAT RHCS1 HAS DVA!RDY
5065 017776 012737 004200 001124      MOV    #DVA!RDY, $GDDAT ;GET GOOD = 4200
5066
5067 020004 017737 164050 001126      MOV    @RHCS1, $BDDAT   ;READ RHCS1 FOR COMPARISON
5068 020012 023737 001124 001126      CMP    $GDDAT, $BDDAT  ;COMPARE EXPECTED
5069                      ;DATA WITH DATA READ FROM
5070                      ;RHCS1
5071 020020 001401                      BEQ    72$                ;BRANCH IF GOOD
5072 020022 104026                      ERROR  26
5073                      ;AFTER SETTING 'CLR' BIT #5
5074                      ;IN RHCS2 TO INIT THE RH
5075                      ;AND HAVING WRITTEN A
5076                      ;COUNT PATTERN 0 THRU 7
5077                      ;INTO RHDB TO FILL IT
5078                      ;RHCS1 SHOULD HAVE DVA!RDY
5079                      ;=4200
5080                      ;BUT CONTAINED WHAT IS
5081                      ;GIVEN IN BAD RHCS1
5082 020024                      72$:
5083                      ;CHECK THAT RHCS3 HAS 0
5084 020024 012737 000000 001124      MOV    #0, $GDDAT      ;GET GOOD = 0
5085
5086 020032 017737 164074 001126      MOV    @RHCS3, $BDDAT   ;READ RHCS3 FOR COMPARISON
5087 020040 023737 001124 001126      CMP    $GDDAT, $BDDAT  ;COMPARE EXPECTED

```

```
5088                                     :DATA WITH DATA READ FROM
5089                                     :RHCS3
5090 020046 001401                       BEQ      74$
5091 020050 104027                       ERROR    27
5092                                     :AFTER SETTING 'CLR' BIT #5
5093                                     :IN RHCS2 TO INIT THE RH
5094                                     :AND HAVING WRITTEN A
5095                                     :COUNT PATTERN 0 THRU 7
5096                                     :INTO RHDB TO FILL IT
5097                                     :RHCS3 SHOULD HAVE 0
5098                                     :BUT CONTAINED WHAT IS
5099                                     :GIVEN IN BAD RHCS3
5100 020052                               74$:
5101                                     :CHECK THAT RHBA HAS 0
5102 020052 012737 000000 001124         MOV      #0,$GDDAT      :GET GOOD = 0
5103
5104 020060 017737 164000 001126         MOV      @RHBA,$BDDAT   :READ RHBA FOR COMPARISON
5105 020066 023737 001124 001126         CMP      $GDDAT,$BDDAT :COMPARE EXPECTED
5106                                     :DATA WITH DATA READ FROM
5107                                     :RHBA
5108 020074 001401                       BEQ      76$
5109 020076 104030                       ERROR    30
5110                                     :AFTER SETTING 'CLR' BIT #5
5111                                     :IN RHCS2 TO INIT THE RH
5112                                     :AND HAVING WRITTEN A
5113                                     :COUNT PATTERN 0 THRU 7
5114                                     :INTO RHDB TO FILL IT
5115                                     :RHBA SHOULD HAVE 0
5116                                     :BUT CONTAINED WHAT IS
5117                                     :GIVEN IN BAD RHBA
5118 020100                               76$:
5119                                     :CHECK THAT RHBAE HAS 0
5120 020100 012737 000000 001124         MOV      #0,$GDDAT      :GET GOOD = 0
5121
5122 020106 017737 164016 001126         MOV      @RHBAE,$BDDAT :READ RHBAE FOR COMPARISON
5123 020114 023737 001124 001126         CMP      $GDDAT,$BDDAT :COMPARE EXPECTED
5124                                     :DATA WITH DATA READ FROM
5125                                     :RHBAE
5126 020122 001401                       BEQ      78$
5127 020124 104031                       ERROR    31
5128                                     :AFTER SETTING 'CLR' BIT #5
5129                                     :IN RHCS2 TO INIT THE RH
5130                                     :AND HAVING WRITTEN A
5131                                     :COUNT PATTERN 0 THRU 7
5132                                     :INTO RHDB TO FILL IT
5133                                     :RHBAE SHOULD HAVE 0
5134                                     :BUT CONTAINED WHAT IS
5135                                     :GIVEN IN BAD RHBAE
5136 020126                               78$:
5137                                     :CHECK THAT RHWC HAS 0
5138 020126 012737 000000 001124         MOV      #0,$GDDAT      :GET GOOD = 0
5139
5140 020134 017737 163722 001126         MOV      @RHWC,$BDDAT   :READ RHWC FOR COMPARISON
5141 020142 023737 001124 001126         CMP      $GDDAT,$BDDAT :COMPARE EXPECTED
5142                                     :DATA WITH DATA READ FROM
5143                                     :RHWC
```



```

5200                                     :=100
5201                                     ;TOGETHER WITH UNIT NUMBER
5202                                     ;BUT CONTAINED WHAT IS
5203                                     ;GIVEN IN BAD RHCS2
5204 020230                             65$:
5205
5206                                     ;WRITE INTO RHDB TO FILL IT
5207                                     ;DATA IS 1,2,4,10,20,40,100,200...
5208 020230 012701 000001                MOV #1,R1          ;GETTING READY TO FLOAT ONES
5209 020234 013702 006426                MOV SILOSZ,R2       ;COUNTER DATA WORDS
5210 020240 010177 163636                1$: MOV R1,@RHDB    ;WRITE INTO RHDB
5211 020244 006301                       ASL R1           ;SHIFT 1 ONE POSITION LEFT
5212 020246 001002                       BNE 2$           ;BRANCH IF NOT LAST WORD IN PATTERN

```



```

5227
5228
5229 020266 012737 000200 001124      :CHECK THAT RHCS2 HAS OR
5230 020274 053737 006442 001124      MOV #OR,$GDDAT      :GET GOOD = 200
5231                                     BIS UNIT,$GDDAT     :INCLUDE UNIT NUMBER
5232 020302 017737 163562 001126      MOV @RHCS2,$BDDAT   :READ RHCS2 FOR COMPARISON
5233 020310 023737 001124 001126      CMP $GDDAT,$BDDAT  :COMPARE EXPECTED
5234                                     :DATA WITH DATA READ FROM
5235                                     :RHCS2
5236 020316 001401                                     BEQ 67$             :BRANCH IF GOOD
5237 020320 104023                                     ERROR 23
5238
5239                                     :AFTER SETTING 'CLR' BIT #5
5240                                     :IN RHCS2 TO INIT THE RH
5241                                     :AND HAVING WRITTEN A
5242                                     :FLOATING ONES PATTERN
5243                                     :1,2,4,10,20,40,100,200
5244                                     :INTO RHDB TO FILL IT
5245                                     :RHCS2 SHOULD HAVE OR
5246                                     :=200
5247                                     :TOGETHER WITH UNIT NUMBER
5248                                     :BUT CONTAINED WHAT IS
5249                                     :GIVEN IN BAD RHCS2
5250 020322      67$:
5251
5252      :READ RHDB UNTIL THE SILO IS EMPTY
5253      :VERIFY THE COUNT PATTERN AS READ FROM THE SILO
5254      :IS THE SAME PATTERN AS PREVIOUSLY WRITTEN INTO THE SILO
5255      :ELSE REPORT THE ERROR
5256
5257 020322 005037 006304                                     CLR SILOM           :INITIALIZE SILO WORD #
5258 020326 012737 000001 001124      MOV #1,$GDDAT      :INITIALIZE EXPECTED DATA
5259 020334 005237 006304      68$:      INC SILOM           :COUNT ONE SILO WORD
5260 020340 023737 006304 006426      CMP SILOM,SILOSZ   :IS THE SILO EMPTY?
5261 020346 101017                                     BHI 70$            :BRANCH IF YES
5262 020350 017737 163526 001126      MOV @RHDB,$BDDAT   :GET 1 WORD FROM THE SILO
5263 020356 023737 001124 001126      CMP $GDDAT,$BDDAT  :EXPECTED=ACTUAL?
5264 020364 001401                                     BEQ 69$            :BRANCH IF YES
5265 020366 104024                                     ERROR 24           :REPORT THE ERROR
5266 020370      69$:
5267 020370 006337 001124      ASL $GDDAT          :GET NEXT EXPECTED DATA
5268 020374 001357                                     BNE 68$            :BRANCH IF NOT LAST WORD IN PATTERN
5269 020376 012737 000001 001124      MOV #1,$GDDAT      :ELSE START PATTERN OVER
5270 020404 000753                                     BR 68$             :CONTINUE
5271 020406      70$:
5272

```



```

5273
5274
5275
5276 020406 012737 000100 001124      :CHECK THAT RHCS2 HAS IR
5277 020414 053737 006442 001124      MOV #IR,$GDDAT ;GET GOOD = 100
5278                                     BIS UNIT,$GDDAT ;INCLUDE UNIT NUMBER
5279 020422 017737 163442 001126      MOV @RHCS2,$BDDAT ;READ RHCS2 FOR COMPARISON
5280 020430 023737 001124 001126      CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
5281                                     ;DATA WITH DATA READ FROM
5282                                     ;RHCS2
5283 020436 001401                                     BEQ 72$ ;BRANCH IF GOOD
5284 020440 104025                                     ERROR 25
5285
5286                                     ;AFTER SETTING "CLR" BIT #5
5287                                     ;IN RHCS2 TO INIT THE RH
5288                                     ;AND HAVING WRITTEN A
5289                                     ;FLOATING ONES PATTERN
5290                                     ;1,2,4,10,20,40,100,200
5291                                     ;INTO RHDB TO FILL IT
5292                                     ;RHCS2 SHOULD HAVE IR
5293                                     ;=100
5294                                     ;TOGETHER WITH UNIT NUMBER
5295                                     ;BUT CONTAINED WHAT IS
5296                                     ;GIVEN IN BAD RHCS2
5297 020442                                     72$:
5298                                     ;CHECK THAT RHCS1 HAS DVA:RDY
5299 020442 012737 004200 001124      MOV #DVA:RDY,$GDDAT ;GET GOOD = 4200
5300
5301 020450 017737 163404 001126      MOV @RHCS1,$BDDAT ;READ RHCS1 FOR COMPARISON
5302 020456 023737 001124 001126      CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
5303                                     ;DATA WITH DATA READ FROM
5304                                     ;RHCS1
5305 020464 001401                                     BEQ 74$ ;BRANCH IF GOOD
5306 020466 104026                                     ERROR 26
5307
5308                                     ;AFTER SETTING "CLR" BIT #5
5309                                     ;IN RHCS2 TO INIT THE RH
5310                                     ;AND HAVING WRITTEN A
5311                                     ;FLOATING ONES PATTERN
5312                                     ;1,2,4,10,20,40,100,200
5313                                     ;INTO RHDB TO FILL IT
5314                                     ;RHCS1 SHOULD HAVE DVA:RDY
5315                                     ;=4200
5316                                     ;BUT CONTAINED WHAT IS
5317                                     ;GIVEN IN BAD RHCS1
5318 020470                                     74$:
5319                                     ;CHECK THAT RHCS3 HAS 0
5320 020470 012737 000000 001124      MOV #0,$GDDAT ;GET GOOD = 0
5321
5322 020476 017737 163430 001126      MOV @RHCS3,$BDDAT ;READ RHCS3 FOR COMPARISON
5323 020504 023737 001124 001126      CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
5324                                     ;DATA WITH DATA READ FROM
5325                                     ;RHCS3
5326 020512 001401                                     BEQ 76$ ;BRANCH IF GOOD
5327 020514 104027                                     ERROR 27
5328

```



```

5329                                     ;AFTER SETTING 'CLR' BIT #5
5330                                     ;IN RHCS2 TO INIT THE RH
5331                                     ;AND HAVING WRITTEN A
5332                                     ;FLOATING ONES PATTERN
5333                                     ;1,2,4,10,20,40,100,200
5334                                     ;INTO RHDB TO FILL IT
5335                                     ;RHCS3 SHOULD HAVE 0
5336                                     ;BUT CONTAINED WHAT IS
5337                                     ;GIVEN IN BAD RHCS3

```

```

5338 020516                               76$:
5339                                     ;CHECK THAT RHBA HAS 0
5340 020516 012737 000000 001124      MOV     #0,$GDDAT
5341                                     ;GET GOOD = 0
5342 020524 017737 163334 001126      MOV     @RHBA,$BDDAT
5343 020532 023737 001124 001126      CMP     $GDDAT,$BDDAT
5344                                     ;READ RHBA FOR COMPARISON
5345                                     ;COMPARE EXPECTED
5346 020540 001401                       BEQ     78$
5347 020542 104030                       ERROR   30

```

```

;AFTER SETTING 'CLR' BIT #5
;IN RHCS2 TO INIT THE RH
;AND HAVING WRITTEN A
;FLOATING ONES PATTERN
;1,2,4,10,20,40,100,200
;INTO RHDB TO FILL IT
;RHBA SHOULD HAVE 0
;BUT CONTAINED WHAT IS
;GIVEN IN BAD RHBA

```

```

5348
5349                                     ;AFTER SETTING 'CLR' BIT #5
5350                                     ;IN RHCS2 TO INIT THE RH
5351                                     ;AND HAVING WRITTEN A
5352                                     ;FLOATING ONES PATTERN
5353                                     ;1,2,4,10,20,40,100,200
5354                                     ;INTO RHDB TO FILL IT
5355                                     ;RHBA SHOULD HAVE 0
5356                                     ;BUT CONTAINED WHAT IS
5357                                     ;GIVEN IN BAD RHBA

```

```

5358 020544                               78$:
5359                                     ;CHECK THAT RHBAE HAS 0
5360 020544 012737 000000 001124      MOV     #0,$GDDAT
5361                                     ;GET GOOD = 0
5362 020552 017737 163352 001126      MOV     @RHBAE,$BDDAT
5363 020560 023737 001124 001126      CMP     $GDDAT,$BDDAT
5364                                     ;READ RHBAE FOR COMPARISON
5365                                     ;COMPARE EXPECTED
5366 020566 001401                       BEQ     80$
5367 020570 104031                       ERROR   31

```

```

;AFTER SETTING 'CLR' BIT #5
;IN RHCS2 TO INIT THE RH
;AND HAVING WRITTEN A
;FLOATING ONES PATTERN
;1,2,4,10,20,40,100,200
;INTO RHDB TO FILL IT
;RHBAE SHOULD HAVE 0
;BUT CONTAINED WHAT IS
;GIVEN IN BAD RHBAE

```

```

5368
5369                                     ;AFTER SETTING 'CLR' BIT #5
5370                                     ;IN RHCS2 TO INIT THE RH
5371                                     ;AND HAVING WRITTEN A
5372                                     ;FLOATING ONES PATTERN
5373                                     ;1,2,4,10,20,40,100,200
5374                                     ;INTO RHDB TO FILL IT
5375                                     ;RHBAE SHOULD HAVE 0
5376                                     ;BUT CONTAINED WHAT IS
5377                                     ;GIVEN IN BAD RHBAE

```

```

5378 020572                               80$:
5379                                     ;CHECK THAT RHWC HAS 0
5380 020572 012737 000000 001124      MOV     #0,$GDDAT
5381                                     ;GET GOOD = 0
5382 020600 017737 163256 001126      MOV     @RHWC,$BDDAT
5383 020606 023737 001124 001126      CMP     $GDDAT,$BDDAT
5384                                     ;READ RHWC FOR COMPARISON

```

```

;COMPARE EXPECTED
;DATA WITH DATA READ FROM

```



```

5385
5386 020614 001401          BEQ      82$          :RHCW
5387 020616 104032          ERROR    32          :BRANCH IF GOOD
5388
5389
5390
5391
5392
5393
5394
5395
5396
5397
5398 020620          82$:
5399
5400
5401
5402
5403
5404
5405
5406
5407
5408
5409
5410
5411
5412
5413
5414
5415
5416
5417
5418
5419
5420
5421
5422 020620 000004
5423 020622 012706 001000
5424 020626 012737 000016 006276
5425
5426 020634 004737 041222
5427
5428
5429
5430
5431
5432 020640 012737 000100 001124
5433 020646 053737 006442 001124
5434
5435 020654 017737 163210 001126
5436 020662 023737 001124 001126
5437
5438
5439 020670 001401          BEQ      65$          :RHCW
5440 020672 104006          ERROR    6          :BRANCH IF GOOD

```

```

:AFTER SETTING 'CLR' BIT #5
:IN RHCS2 TO INIT THE RH
:AND HAVING WRITTEN A
:FLOATING ONES PATTERN
:1,2,4,10,20,40,100,200
:INTO RHDB TO FILL IT
:RHCW SHOULD HAVE 0
:BUT CONTAINED WHAT IS
:GIVEN IN BAD RHCW

```

```

*****
:TEST 16          SILO TEST 6 (FLOATING ONES)

```

```

:*
:* AFTER A RH CLEAR IS GIVEN (SET BIT #5 IN RHCS2)
:* RHCS2 IS CHECKED TO HAVE 'IR' (BIT #6) HIGH,
:* TOGETHER WITH UNIT NUMBER
:* A PATTERN OF FLOATING ONES STARTING 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 READS.
:* THEN RHCS2 IS CHECKED TO HAVE 'IR'
:* AND UNIT NUMBER
:* THEN RHCS1, RHCS3, RHBA, RHBAE, RHCW ARE
:* CHECKED TO HAVE THE APPROPRIATE VALUE

```

```

*****
TST16: SCOPE
MOV      #STACK,SP          :RESET STACK
MOV      #16,@WTSTNM        :SAVE TEST NUMBER
JSR      PC,@#CLDISK        :GIVE RH INITIALIZE
:SETUP UNIT NUBER
:CLEAR RHCW 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

```

5441
5442
5443
5444
5445
5446
5447
5448
5449
5450
5451
5452
5453
5454
5455
5456
5457
5458
5459
5460
5461
5462
5463
5464
5465
5466
5467
5468
5469
5470
5471
5472
5473
5474
5475
5476
5477
5478
5479
5480
5481
5482
5483
5484
5485
5486
5487
5488
5489
5490
5491
5492
5493
5494
5495
5496

020674

65\$:

012701 000400
020700 013702 006426
020704 010177 163172
020710 006301
020712 001002
020714 012701 000001
020720 005302
020722 001370
020724 104411
020726 004070
020730 000200

1\$:

2\$:

;WRITE INTO RHDB TO FILL IT
;DATA IS 400,1000,2000,4000,10000,20000,40000
;100000,1,2,4,....
MOV #400,R1 ;GETTING READY TO FLOAT ONES
MOV SILOSZ,R2 ;COUNTER DATA WORDS
MOV R1,@RHDB ;WRITE INTO RHDB
ASL R1 ;SHIFT 1 ONE POSITION LEFT
BNE 2\$;BRANCH IF NOT LAST WORD IN PATTERN
MOV #1,R1 ;ELSE START PATTERN OVER
DEC R2 ;COUNT DOWN
BNE 1\$;BRANCH IF NOT DONE
;WAIT FOR OR BIT IN RHCS2 REGISTER TO SET
WAT ;TRAP TO WAIT.T SUBROUTINE
RHCS2 ;AND WAIT FOR OR BIT IN
OR ;RHCS2 REGISTER
;IF ERROR OCCURS HERE
;IT MEANS 'OR' DID NOT
;SET FOR THE FULL COUNT
;DOWN OF THE WAIT.T SUBROUTINE
;THIS TIME IS APPROXIMATELY
;LARGER THAN 500 MILLISECONDS

;CHECK THAT RHCS2 HAS OR
MOV #OR,\$GDDAT ;GET GOOD = 200
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 67\$;BRANCH IF GOOD
ERROR 23

;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

;AFTER SETTING 'CLR' BIT #5
;IN RHCS2 TO INIT THE RH
;AND HAVING WRITTEN A
;FLOATING ONES PATTERN INTO UPPER BYTE
;400,1000,2000,4000,10000,20000
;40000,100000
;INTO RHDB TO FILL IT
;RHCS2 SHOULD HAVE OR
;=200
;TOGETHER WITH UNIT NUMBER
;BUT CONTAINED WHAT IS
;GIVEN IN BAD RHCS2


```

5497 020766          67$:
5498
5499                :READ RHDB UNTIL THE SILO IS EMPTY
5500                :VERIFY THE COUNT PATTERN AS READ FROM THE SILO
5501                :IS THE SAME PATTERN AS PREVIOUSLY WRITTEN INTO THE SILO
5502                :ELSE REPORT THE ERROR
5503
5504 020766 005037 006304          CLR      SILONM          ;INITIALIZE SILO WORD #
5505 020772 012737 000400 001124  MOV      #400, $GDDAT    ;INITIALIZE EXPECTED DATA
5506 021000 005237 006304          INC      SILONM          ;COUNT ONE SILO WORD
5507 021004 023737 006304 006426 68$:  CMP      SILONM, SILOSZ  ;IS THE SILO EMPTY?
5508 021012 101017          BHI      70$            ;BRANCH IF YES
5509 021014 017737 163062 001126  MOV      @RHDB, $BDDAT   ;GET 400 WORD FROM THE SILO
5510 021022 023737 001124 001126  CMP      $GDDAT, $BDDAT ;EXPECTED=ACTUAL?
5511 021030 001401          BEQ      69$            ;BRANCH IF YES
5512 021032 104024          ERROR    24          ;REPORT THE ERROR
5513 021034          69$:
5514 021034 006337 001124          ASL      $GDDAT          ;GET NEXT EXPECTED DATA
5515 021040 001357          BNE      68$            ;BRANCH IF NOT LAST WORD IN PATTERN
5516 021042 012737 000001 001124  MOV      #1, $GDDAT     ;ELSE START PATTERN OVER
5517 021050 000753          BR       68$            ;CONTINUE
5518 021052          70$:
5519
5520
5521
5522                :CHECK THAT RHCS2 HAS IR
5523 021052 012737 000100 001124  MOV      #IR, $GDDAT    ;GET GOOD = 100
5524 021060 053737 006442 001124  BIS      UNIT, $GDDAT   ;INCLUDE UNIT NUMBER
5525
5526 021066 017737 162776 001126  MOV      @RHCS2, $BDDAT ;READ RHCS2 FOR COMPARISON
5527 021074 023737 001124 001126  CMP      $GDDAT, $BDDAT ;COMPARE EXPECTED
5528                :DATA WITH DATA READ FROM
5529                :RHCS2
5530 021102 001401          BEQ      72$            ;BRANCH IF GOOD
5531 021104 104025          ERROR    25
5532
5533                :AFTER SETTING "CLR" BIT #5
5534                :IN RHCS2 TO INIT THE RH
5535                :AND HAVING WRITTEN A
5536                :FLOATING ONES PATTERN INTO UPPER BYTE
5537                :400,1000,2000,4000,10000,20000
5538                :40000,100000
5539                :INTO RHDB TO FILL IT
5540                :RHCS2 SHOULD HAVE IR
5541                :=100
5542                :TOGETHER WITH UNIT NUMBER
5543                :BUT CONTAINED WHAT IS
5544                :GIVEN IN BAD RHCS2
5544 021106          72$:
5545
5546 021106 012737 004200 001124  MOV      #DVA!RDY, $GDDAT ;GET GOOD = 4200
5547
5548 021114 017737 162740 001126  MOV      @RHCS1, $BDDAT ;READ RHCS1 FOR COMPARISON
5549 021122 023737 001124 001126  CMP      $GDDAT, $BDDAT ;COMPARE EXPECTED
5550                :DATA WITH DATA READ FROM
5551                :RHCS1
5552 021130 001401          BEQ      74$            ;BRANCH IF GOOD

```

```

5553 021132 104026          ERROR 26
5554
5555
5556
5557
5558
5559
5560
5561
5562
5563
5564
5565 021134          74$:
5566
5567 021134 012737 000000 001124      :CHECK THAT RHCS3 HAS 0
5568
5569 021142 017737 162764 001126      MOV #0,$GDDAT      :GET GOOD = 0
5570 021150 023737 001124 001126      MOV @RHCS3,$BDDAT :READ RHCS3 FOR COMPARISON
5571
5572
5573 021156 001401      CMP $GDDAT,$BDDAT :COMPARE EXPECTED
5574 021160 104027      :DATA WITH DATA READ FROM
5575
5576
5577
5578
5579
5580
5581
5582
5583
5584
5585 021162          76$:
5586
5587 021162 012737 000000 001124      BEQ 76$           :BRANCH IF GOOD
5588
5589 021170 017737 162670 001126      ERROR 27
5590 021176 023737 001124 001126
5591
5592
5593 021204 001401
5594 021206 104030
5595
5596
5597
5598
5599
5600
5601
5602
5603
5604
5605 021210          78$:
5606
5607 021210 012737 000000 001124      :CHECK THAT RHBAE HAS 0
5608

```

```

:AFTER SETTING 'CLR' BIT #5
:IN RHCS2 TO INIT THE RH
:AND HAVING WRITTEN A
:FLOATING ONES PATTERN INTO UPPER BYTE
:400,1000,2000,4000,10000,20000
:40000,100000
:INTO RHDB TO FILL IT
: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 HAVING WRITTEN A
:FLOATING ONES PATTERN INTO UPPER BYTE
:400,1000,2000,4000,10000,20000
:40000,100000
:INTO RHDB TO FILL IT
:RHCS3 SHOULD HAVE 0
:BUT CONTAINED WHAT IS
:GIVEN IN BAD RHCS3

```

```

:AFTER SETTING 'CLR' BIT #5
:IN RHCS2 TO INIT THE RH
:AND HAVING WRITTEN A
:FLOATING ONES PATTERN INTO UPPER BYTE
:400,1000,2000,4000,10000,20000
:40000,100000
:INTO RHDB TO FILL IT
:RHBA SHOULD HAVE 0
:BUT CONTAINED WHAT IS
:GIVEN IN BAD RHBA

```



```

5609 021216 017737 162706 001126      MOV    @RHBAE,$BDDAT    ;READ RHBAE FOR COMPARISON
5610 021224 023737 001124 001126      CMP    $GDDAT,$BDDAT   ;COMPARE EXPECTED
5611                                     ;DATA WITH DATA READ FROM
5612                                     ;RHBAE
5613 021232 001401                      BEQ    80$              ;BRANCH IF GOOD
5614 021234 104031                      ERROR  31
5615                                     ;AFTER SETTING 'CLR' BIT #5
5616                                     ;IN RHCS2 TO INIT THE RH
5617                                     ;AND HAVING WRITTEN A
5618                                     ;FLOATING ONES PATTERN INTO UPPER BYTE
5619                                     ;400,1000,2000,4000,10000,20000
5620                                     ;40000,100000
5621                                     ;INTO RHDB TO FILL IT
5622                                     ;RHBAE SHOULD HAVE 0
5623                                     ;BUT CONTAINED WHAT IS
5624                                     ;GIVEN IN BAD RHBAE
5625 021236                               80$:
5626                                     ;CHECK THAT RHWC HAS 0
5627 021236 012737 000000 001124      MOV    #0,$GDDAT       ;GET GOOD = 0
5628
5629 021244 017737 162612 001126      MOV    @RHWC,$BDDAT    ;READ RHWC FOR COMPARISON
5630 021252 023737 001124 001126      CMP    $GDDAT,$BDDAT   ;COMPARE EXPECTED
5631                                     ;DATA WITH DATA READ FROM
5632                                     ;RHWC
5633 021260 001401                      BEQ    82$              ;BRANCH IF GOOD
5634 021262 104032                      ERROR  32
5635                                     ;AFTER SETTING 'CLR' BIT #5
5636                                     ;IN RHCS2 TO INIT THE RH
5637                                     ;AND HAVING WRITTEN A
5638                                     ;FLOATING ONES PATTERN INTO UPPER BYTE
5639                                     ;400,1000,2000,4000,10000,20000
5640                                     ;40000,100000
5641                                     ;INTO RHDB TO FILL IT
5642                                     ;RHWC SHOULD HAVE 0
5643                                     ;BUT CONTAINED WHAT IS
5644                                     ;GIVEN IN BAD RHWC
5645 021264                               82$:
5646
5647
5648
5649
5650

```

```

*****
:*TEST 17      SILO TEST 7 (FLOATING 0 )

```

```

5651      :*      AFTER A RH CLEAR IS GIVEN (SET BIT #5 IN RHCS2)
5652      :*      RHCS2 IS CHECKED TO HAVE 'IR' (BIT #6) HIGH,
5653      :*      TOGETHER WITH UNIT NUMBER
5654      :*      'N' WORDS OF FLOATING ZEROS 177776, 177775, 177773,
5655      :*      177767, 177757, 177737, 177677, 177577...
5656      :*      IS WRITTEN INTO RHDB
5657      :*      'OR' (BIT #7 IN RHCS2) IS WAITED FOR BY A TRAP
5658      :*      INSTRUCTION CALLED 'WAT'
5659      :*      THEN RHCS2 IS CHECKED TO HAVE 'IR' LOW AND
5660      :*      'OR' HIGH TOGETHER WITH THE UNIT NUMBER
5661      :*      THEN RHDB IS READ AND COMPARED TO HAVE THE
5662      :*      RIGHT VALUE AFTER EACH OF THE READS.
5663      :*      THEN RHCS2 IS CHECKED TO HAVE 'IR'
5664      :*      AND UNIT NUMBER

```

```

5665          : * THEN RHCS1, RHCS3, RHBA, RHBAE, RHWC ARE
5666          : * CHECKED TO HAVE THE APPROPRIATE VALUE
5667          : * *****
5668          TST17: SCOPE
5669 021264 000004      MOV #STACK,SP      ;RESET STACK
5670 021266 012706 001000  MOV #17,@#TSTNM    ;SAVE TEST NUMBER
5671 021272 012737 000017 006276
5672 021300 004737 041222      JSR PC,@#CLDISK    ;GIVE RH INITIALIZE
5673          ;SETUP UNIT NUBER
5674          ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
5675
5676          ;CHECK THAT RHCS2 HAS IR
5677          MOV #IR,$GDDAT      ;GET GOOD = 100
5678 021304 012737 000100 001124  BIS UNIT,$GDDAT    ;INCLUDE UNIT NUMBER
5679 021312 053737 006442 001124
5680          MOV @RHCS2,$BDDAT   ;READ RHCS2 FOR COMPARISON
5681 021320 017737 162544 001126  CMP $GDDAT,$BDDAT  ;COMPARE EXPECTED
5682 021326 023737 001124 001126  ;DATA WITH DATA READ FROM
5683          ;RHCS2
5684          BEQ ERROR 65$      ;BRANCH IF GOOD
5685 021334 001401
5686 021336 104006
5687          ;AFTER SETTING 'CLR' BIT #5
5688          ;IN RHCS2 TO INIT THE RH
5689          ;AND HAVING DONE NOTHING
5690          ;ELSE
5691          ;RHCS2 SHOULD HAVE IR
5692          ;=100
5693          ;TOGETHER WITH UNIT NUMBER
5694          ;BUT CONTAINED WHAT IS
5695          ;GIVEN IN BAD RHCS2
5696 021340          65$:
5697
5698          ;WRITE INTO RHDB TO FILL IT
5699          ;DATA IS FLOATING ZEROS 177776, 177775, 177773, 177767, 177757, 177737,
5700          ;177677, 177577...
5701 021340 012701 177776      MOV #177776,R1     ;GETTING READY TO FLOAT ZEROS
5702 021344 013702 006426      MOV SILOSZ,R2      ;COUNTER DATA WORDS
5703 021350 010177 162526      MOV R1,@RHDB      ;WRITE INTO RHDB
5704 021354 000261          SEC                ;SET CARRY
5705 021356 006101          ROL R1            ;GET ZERO ONE BIT LEFT
5706 021360 102402          BVS 2$           ;BRANCH IF NOT LAST WORD IN PATTERN
5707 021362 012701 177776      MOV #177776, R1   ;ELSE START PATTERN OVER
5708 021366 005302          DEC R2           ;COUNT DOWN
5709 021370 001367          BNE 1$           ;BRANCH IF NOT DONE
5710          ;WAIT FOR OR BIT IN RHCS2 REGISTER TO SET
5711 021372 104411          WAT            ;TRAP TO WAIT.T SUBROUTINE
5712 021374 004070          RHCS2         ;AND WAIT FOR OR BIT IN
5713 021376 000200          OR             ;RHCS2 REGISTER
5714          ;IF ERROR OCCURS HERE
5715          ;IT MEANS 'OR' DID NOT
5716          ;SET FOR THE FULL COUNT
5717          ;DOWN OF THE WAIT.T SUBROUTINE
5718          ;THIS TIME IS APPROXIMATELY
5719          ;LARGER THAN 500 MILLISECONDS
5720

```



```

5721
5722
5723 021400 012737 000200 001124
5724 021406 053737 006442 001124
5725
5726 021414 017737 162450 001126
5727 021422 023737 001124 001126
5728
5729
5730 021430 001401
5731 021432 104023
5732
5733
5734
5735
5736
5737
5738
5739
5740
5741
5742
5743 021434
5744
5745
5746
5747
5748
5749
5750 021434 005037 006304
5751 021440 012737 177776 001124
5752 021446 005237 006304
5753 021452 023737 006304 006426
5754 021460 101017
5755 021462 017737 162414 001126
5756 021470 023737 001124 001126
5757 021476 001401
5758 021500 104024
5759 021502
5760 021502 006137 001124
5761 021506 102757
5762 021510 012737 177776 001124
5763 021516 000753
5764 021520
5765
5766
5767
5768
5769 021520 012737 000100 001124
5770 021526 053737 006442 001124
5771
5772 021534 017737 162330 001126
5773 021542 023737 001124 001126
5774
5775
5776 021550 001401
    
```

```

;CHECK THAT RHCS2 HAS OR
MOV #OR,$GDDAT ;GET GOOD = 200
BIS UNIT,$GDDAT ;INCLUDE UNIT NUMBER

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

BEQ 67$
ERROR 23

;AFTER SETTING 'CLR' BIT #5
;IN RHCS2 TO INIT THE RH
;AND HAVING WRITTEN A PATTERN
;OF FLOATING ZEROS - 177776, 177775,
;177773, 177767, 177757, 177737, 177677, 177577
;INTO RHDB TO FILL IT
;RHCS2 SHOULD HAVE OR
;=200
;TOGETHER WITH UNIT NUMBER
;BUT CONTAINED WHAT IS
;GIVEN IN BAD RHCS2

67$:
;READ RHDB UNTIL THE SILO IS EMPTY
;VERIFY THE COUNT PATTERN AS READ FROM THE SILO
;IS THE SAME PATTERN AS PREVIOUSLY WRITTEN INTO THE SILO
;ELSE REPORT THE ERROR

CLR SILONM ;INITIALIZE SILO WORD #
MOV #177776,$GDDAT ;INITIALIZE EXPECTED DATA
68$: INC SILONM ;COUNT ONE SILO WORD
CMP SILONM,SILOSZ ;IS THE SILO EMPTY?
BHI 70$ ;BRANCH IF YES
MOV @RHDB,$BDDAT ;GET 177776 WORD FROM THE SILO
CMP $GDDAT,$BDDAT ;EXPECTED=ACTUAL?
BEQ 69$ ;BRANCH IF YES
ERROR 24 ;REPORT THE ERROR

69$: ROL $GDDAT ;GET NEXT EXPECTED DATA
BVS 68$ ;BRANCH IF NOT LAST WORD IN PATTERN
MOV #177776,$GDDAT ;ELSE START PATTERN OVER
BR 68$ ;CONTINUE

70$:

;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
;BRANCH IF GOOD

BEQ 72$
    
```

```

5777 021552 104025          ERROR 25
5778
5779
5780
5781
5782
5783
5784
5785
5786
5787
5788
5789 021554          72$:
5790
5791 021554 012737 004200 001124
5792
5793 021562 017737 162272 001126
5794 021570 023737 001124 001126
5795
5796
5797 021576 001401
5798 021600 104026
5799
5800
5801
5802
5803
5804
5805
5806
5807
5808
5809 021602          74$:
5810
5811 021602 012737 000000 001124
5812
5813 021610 017737 162316 001126
5814 021616 023737 001124 001126
5815
5816
5817 021624 001401
5818 021626 104027
5819
5820
5821
5822
5823
5824
5825
5826
5827
5828 021630          76$:
5829
5830 021630 012737 000000 001124
5831
5832 021636 017737 162222 001126

```

```

;AFTER SETTING 'CLR' BIT #5
;IN RHCS2 TO INIT THE RH
;AND HAVING WRITTEN A PATTERN
;OF FLOATING ZEROS - 177776, 177775,
;177773, 177767, 177757, 177737, 177677, 177577
;INTO RHDB TO FILL IT
;RHCS2 SHOULD HAVE IR
;=100
;TOGETHER WITH UNIT NUMBER
;BUT CONTAINED WHAT IS
;GIVEN IN BAD RHCS2

;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 74$ ;BRANCH IF GOOD
ERROR 26

;AFTER SETTING 'CLR' BIT #5
;IN RHCS2 TO INIT THE RH
;AND HAVING WRITTEN A PATTERN
;OF FLOATING ZEROS - 177776, 177775,
;177773, 177767, 177757, 177737, 177677, 177577
;INTO RHDB TO FILL IT
;RHCS1 SHOULD HAVE DVA!RDY
;=4200
;BUT CONTAINED WHAT IS
;GIVEN IN BAD RHCS1

;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 76$ ;BRANCH IF GOOD
ERROR 27

;AFTER SETTING 'CLR' BIT #5
;IN RHCS2 TO INIT THE RH
;AND HAVING WRITTEN A PATTERN
;OF FLOATING ZEROS - 177776, 177775,
;177773, 177767, 177757, 177737, 177677, 177577
;INTO RHDB TO FILL IT
;RHCS3 SHOULD HAVE 0
;BUT CONTAINED WHAT IS
;GIVEN IN BAD RHCS3

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

MOV @RHBA,$BDDAT ;READ RHBA FOR COMPARISON

```



```

5833 021644 023737 001124 001126      CMP      $GDDAT,$BDDAT      ;COMPARE EXPECTED
5834                                     ;DATA WITH DATA READ FROM
5835                                     ;RHBA
5836 021652 001401                      BEQ      78$                ;BRANCH IF GOOD
5837 021654 104030                      ERROR    30
5838                                     ;AFTER SETTING "CLR" BIT #5
5839                                     ;IN RHCS2 TO INIT THE RH
5840                                     ;AND HAVING WRITTEN A PATTERN
5841                                     ;OF FLOATING ZEROS - 177776, 177775,
5842                                     ;177773, 177767, 177757, 177737, 177677, 177577
5843                                     ;INTO RHDB TO FILL IT
5844                                     ;RHBA SHOULD HAVE 0
5845                                     ;BUT CONTAINED WHAT IS
5846                                     ;GIVEN IN BAD RHBA
5847 021656                                78$:
5848                                     ;CHECK THAT RHBAE HAS 0
5849 021656 012737 000000 001124      MOV      #0,$GDDAT        ;GET GOOD = 0
5850
5851 021664 017737 162240 001126      MCV      @RHBAE,$BDDAT    ;READ RHBAE FOR COMPARISON
5852 021672 023737 001124 001126      CMP      $GDDAT,$BDDAT    ;COMPARE EXPECTED
5853                                     ;DATA WITH DATA READ FROM
5854                                     ;RHBAE
5855 021700 001401                      BEQ      80$                ;BRANCH IF GOOD
5856 021702 104031                      ERROR    31
5857                                     ;AFTER SETTING "CLR" BIT #5
5858                                     ;IN RHCS2 TO INIT THE RH
5859                                     ;AND HAVING WRITTEN A PATTERN
5860                                     ;OF FLOATING ZEROS - 177776, 177775,
5861                                     ;177773, 177767, 177757, 177737, 177677, 177577
5862                                     ;INTO RHDB TO FILL IT
5863                                     ;RHBAE SHOULD HAVE 0
5864                                     ;BUT CONTAINED WHAT IS
5865                                     ;GIVEN IN BAD RHBAE
5866 021704                                80$:
5867                                     ;CHECK THAT RHWC HAS 0
5868 021704 012737 000000 001124      MOV      #0,$GDDAT        ;GET GOOD = 0
5869
5870 021712 017737 162144 001126      MOV      @RHWC,$BDDAT     ;READ RHWC FOR COMPARISON
5871 021720 023737 001124 001126      CMP      $GDDAT,$BDDAT    ;COMPARE EXPECTED
5872                                     ;DATA WITH DATA READ FROM
5873                                     ;RHWC
5874 021726 001401                      BEQ      82$                ;BRANCH IF GOOD
5875 021730 104032                      ERROR    32
5876                                     ;AFTER SETTING "CLR" BIT #5
5877                                     ;IN RHCS2 TO INIT THE RH
5878                                     ;AND HAVING WRITTEN A PATTERN
5879                                     ;OF FLOATING ZEROS - 177776, 177775,
5880                                     ;177773, 177767, 177757, 177737, 177677, 177577
5881                                     ;INTO RHDB TO FILL IT
5882                                     ;RHWC SHOULD HAVE 0
5883                                     ;BUT CONTAINED WHAT IS
5884                                     ;GIVEN IN BAD RHWC
5885 021732                                82$:
5886
5887
5888

```

::.....

```

5889          : *TEST 20          SILO TEST 8 (FLOATING ZEROS )
5890
5891          : *
5892          : *   AFTER A RH CLEAR IS GIVEN (SET BIT #5 IN RHCS2)
5893          : *   RHCS2 IS CHECKED TO HAVE "IR" (BIT #6) HIGH,
5894          : *   TOGETHER WITH UNIT NUMBER
5895          : *   A PATTERN OF FLOATING ZEROS STARTING IN UPPER BYTE
5896          : *   177377, 177677, 175777, 173777, 167777, 157777, 137777, 77777....
5897          : *   IS WRITTEN INTO RHDB
5898          : *   "OR" (BIT #7 IN RHCS2) IS WAITED FOR BY A TRAP
5899          : *   INSTRUCTION CALLED "WAT"
5900          : *   THEN RHCS2 IS CHECKED TO HAVE "IR" LOW AND
5901          : *   "OR" HIGH TOGETHER WITH THE UNIT NUMBER
5902          : *   THEN RHDB IS READ AND COMPARED TO HAVE THE
5903          : *   RIGHT VALUE AFTER EACH OF THE READS.
5904          : *   THEN RHCS2 IS CHECKED TO HAVE "IR"
5905          : *   AND UNIT NUMBER
5906          : *   THEN RHCS1, RHCS3, RHBA, RHBAE, RHWC ARE
5907          : *   CHECKED TO HAVE THE APPROPRIATE VALUE
5908          : *
5908 021732 000004          : *****
5909 021734 012706 001000  TST20: SCOPE
5910 021740 012737 000020 006276  MOV    #STACK,SP          :RESET STACK
5911          :
5912 021746 004737 041222  MOV    #20,@TSTNM        :SAVE TEST NUMBER
5913          :
5914          :
5915          :
5916          :
5917          :
5918          :
5918 021752 012737 000100 001124  JSR    PC,@WCLDISK      :GIVE RH INITIALIZE
5919 021760 053737 006442 001124  :SETUP UNIT NUBER
5920          :
5921          :
5921 021766 017737 162076 001126  :CLEAR RHWC AND FUNCTION BITS IN RHCS1
5922 021774 023737 001124 001126  :
5923          :
5924          :
5925 022002 001401          :CHECK THAT RHCS2 HAS IR
5926 022004 104006          MOV    #IR,$GDDAT      :GET GOOD = 100
5927          :
5928          :
5929          :
5930          :
5931          :
5932          :
5933          :
5934          :
5935          :
5936 022006          :
5937          :
5938          :
5939          :
5940          :
5941          :
5942 022006 012701 177377  BEQ    65$             :BRANCH IF GOOD
5943          :
5944          :
5944 022012 013702 006426  ERROR 6              :AFTER SETTING "CLR" BIT #5
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          :
6000          :
6001          :
6002          :
6003          :
6004          :
6005          :
6006          :
6007          :
6008          :
6009          :
6010          :
6011          :
6012          :
6013          :
6014          :
6015          :
6016          :
6017          :
6018          :
6019          :
6020          :
6021          :
6022          :
6023          :
6024          :
6025          :
6026          :
6027          :
6028          :
6029          :
6030          :
6031          :
6032          :
6033          :
6034          :
6035          :
6036          :
6037          :
6038          :
6039          :
6040          :
6041          :
6042          :
6043          :
6044          :
6045          :
6046          :
6047          :
6048          :
6049          :
6050          :
6051          :
6052          :
6053          :
6054          :
6055          :
6056          :
6057          :
6058          :
6059          :
6060          :
6061          :
6062          :
6063          :
6064          :
6065          :
6066          :
6067          :
6068          :
6069          :
6070          :
6071          :
6072          :
6073          :
6074          :
6075          :
6076          :
6077          :
6078          :
6079          :
6080          :
6081          :
6082          :
6083          :
6084          :
6085          :
6086          :
6087          :
6088          :
6089          :
6090          :
6091          :
6092          :
6093          :
6094          :
6095          :
6096          :
6097          :
6098          :
6099          :
6100          :
6101          :
6102          :
6103          :
6104          :
6105          :
6106          :
6107          :
6108          :
6109          :
6110          :
6111          :
6112          :
6113          :
6114          :
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          :
6171          :
6172          :
6173          :
6174          :
6175          :
6176          :
6177          :
6178          :
6179          :
6180          :
6181          :
6182          :
6183          :
6184          :
6185          :
6186          :
6187          :
6188          :
6189          :
6190          :
6191          :
6192          :
6193          :
6194          :
6195          :
6196          :
6197          :
6198          :
6199          :
6200          :
6201          :
6202          :
6203          :
6204          :
6205          :
6206          :
6207          :
6208          :
6209          :
6210          :
6211          :
6212          :
6213          :
6214          :
6215          :
6216          :
6217          :
6218          :
6219          :
6220          :
6221          :
6222          :
6223          :
6224          :
6225          :
6226          :
6227          :
6228          :
6229          :
6230          :
6231          :
6232          :
6233          :
6234          :
6235          :
6236          :
6237          :
6238          :
6239          :
6240          :
6241          :
6242          :
6243          :
6244          :
6245          :
6246          :
6247          :
6248          :
6249          :
6250          :
6251          :
6252          :
6253          :
6254          :
6255          :
6256          :
6257          :
6258          :
6259          :
6260          :
6261          :
6262          :
6263          :
6264          :
6265          :
6266          :
6267          :
6268          :
6269          :
6270          :
6271          :
6272          :
6273          :
6274          :
6275          :
6276          :
6277          :
6278          :
6279          :
6280          :
6281          :
6282          :
6283          :
6284          :
6285          :
6286          :
6287          :
6288          :
6289          :
6290          :
6291          :
6292          :
6293          :
6294          :
6295          :
6296          :
6297          :
6298          :
6299          :
6300          :
6301          :
6302          :
6303          :
6304          :
6305          :
6306          :
6307          :
6308          :
6309          :
6310          :
6311          :
6312          :
6313          :
6314          :
6315          :
6316          :
6317          :
6318          :
6319          :
6320          :
6321          :
6322          :
6323          :
6324          :
6325          :
6326          :
6327          :
6328          :
6329          :
6330          :
6331          :
6332          :
6333          :
6334          :
6335          :
6336          :
6337          :
6338          :
6339          :
6340          :
6341          :
6342          :
6343          :
6344          :
6345          :
6346          :
6347          :
6348          :
6349          :
6350          :
6351          :
6352          :
6353          :
6354          :
6355          :
6356          :
6357          :
6358          :
6359          :
6360          :
6361          :
6362          :
6363          :
6364          :
6365          :
6366          :
6367          :
6368          :
6369          :
6370          :
6371          :
6372          :
6373          :
6374          :
6375          :
6376          :
6377          :
6378          :
6379          :
6380          :
6381          :
6382          :
6383          :
6384          :
6385          :
6386          :
6387          :
6388          :
6389          :
6390          :
6391          :
6392          :
6393          :
6394          :
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          :
6451          :
6452          :
6453          :
6454          :
6455          :
6456          :
6457          :
6458          :
6459          :
6460          :
6461          :
6462          :
6463          :
6464          :
6465          :
6466          :
6467          :
6468          :
6469          :
6470          :
6471          :
6472          :
6473          :
6474          :
6475          :
6476          :
6477          :
6478          :
6479          :
6480          :
6481          :
6482          :
6483          :
6484          :
6485          :
6486          :
6487          :
6488          :
6489          :
6490          :
6491          :
6492          :
6493          :
6494          :
6495          :
6496          :
6497          :
6498          :
6499          :
6500          :
6501          :
6502          :
6503          :
6504          :
6505          :
6506          :
6507          :
6508          :
6509          :
6510          :
6511          :
6512          :
6513          :
6514          :
6515          :
6516          :
6517          :
6518          :
6519          :
6520          :
6521          :
6522          :
6523          :
6524          :
6525          :
6526          :
6527          :
6528          :
6529          :
6530          :
6531          :
6532          :
6533          :
6534          :
6535          :
6536          :
6537          :
6538          :
6539          :
6540          :
6541          :
6542          :
6543          :
6544          :
6545          :
6546          :
6547          :
6548          :
6549          :
6550          :
6551          :
6552          :
6553          :
6554          :
6555          :
6556          :
6557          :
6558          :
6559          :
6560          :
6561          :
6562          :
6563          :
6564          :
6565          :
6566          :
6567          :
6568          :
6569          :
6570          :
6571          :
6572          :
6573          :
6574          :
6575          :
6576          :
6577          :
6578          :
6579          :
6580          :
6581          :
6582          :
6583          :
6584          :
6585          :
6586          :
6587          :
6588          :
6589          :
6590          :
6591          :
6592          :
6593          :
6594          :
6595          :
6596          :
6597          :
6598          :
6599          :
6600          :
6601          :
6602          :
6603          :
6604          :
6605          :
6606          :
6607          :
6608          :
6609          :
6610          :
6611          :
6612          :
6613          :
6614          :
6615          :
6616          :
6617          :
6618          :
6619          :
6620          :
6621          :
6622          :
6623          :
6624          :
6625          :
6626          :
6627          :
6628          :
6629          :
6630          :
6631          :
6632          :
6633          :
6634          :
6635          :
6636          :
6637          :
6638          :
6639          :
6640          :
6641          :
6642          :
6643          :
6644          :
6645          :
6646          :
6647          :
6648          :
6649          :
6650          :
6651          :
6652          :
6653          :
6654          :
6655          :
6656          :
6657          :
6658          :
6659          :
6660          :
6661          :
6662          :
6663          :
6664          :
6665          :
6666          :
6667          :
6668          :
6669          :
6670          :
6671          :
6672          :
6673          :
6674          :
6675          :
6676          :
6677          :
6678          :
6679          :
6680          :
6681          :
6682          :
6683          :
6684          :
6685          :
6686          :
6687          :
6688          :
6689          :
6690          :
6691          :
6692          :
6693          :
6694          :
6695          :
6696          :
6697          :
6698          :
6699          :
6700          :
6701          :
6702          :
6703          :
6704          :
6705          :
6706          :
6707          :
6708          :
6709          :
6710          :
6711          :
6712          :
6713          :
6714          :
6715          :
6716          :
6717          :
6718          :
6719          :
6720          :
6721          :
6722          :
6723          :
6724          :
6725          :
6726          :
6727          :
6728          :
6729          :
6730          :
6731          :
6732          :
6733          :
6734          :
6735          :
6736          :
6737          :
6738          :
6739          :
6740          :
6741          :
6742          :
6743          :
6744          :
6745          :
6746          :
6747          :
6748          :
6749          :
6750          :
6751          :
6752          :
6753          :
6754          :
6755          :
6756          :
6757          :
6758          :
6759          :
6760          :
6761          :
6762          :
6763          :
6764          :
6765          :
6766          :
6767          :
6768          :
6769          :
6770          :
6771          :
6772          :
6773          :
6774          :
6775          :
6776          :
6777          :
6778          :
6779          :
6780          :
6781          :
6782          :
6783          :
6784          :
6785          :
6786          :
6787          :
6788          :
6789          :
6790          :
6791          :
6792          :
6793          :
6794          :
6795          :
6796          :
6797          :
6798          :
6799          :
6800          :
6801          :
6802          :
6803          :
6804          :
6805          :
6806          :
6807          :
6808          :
6809          :
6810          :
6811          :
6812          :
6813          :
6814          :
6815          :
6816          :
6817          :
6818          :
6819          :
6820          :
6821          :
6822          :
6823          :
6824          :
6825          :
6826          :
6827          :
6828          :
6829          :
6830          :
6831          :
6832          :
6833          :
6834          :
6835          :
6836          :
6837          :
6838          :
6839          :
6840          :
6841          :
6842          :
6843          :
6844          :
6845          :
6846          :
6847          :
6848          :
6849          :
6850          :
6851          :
6852          :
6853          :
6854          :
6855          :
6856          :
6857          :
6858          :
6859          :
6860          :
6861          :
6862          :
6863          :
6864          :
6865          :
6866          :
6867          :
6868          :
6869          :
6870          :
6871          :
6872          :
6873          :
6874          :
6875          :
6876          :
6877          :
6878          :
6879          :
6880          :
6881          :
6882          :
6883          :
6884          :
6885          :
6886          :
6887          :
6888          :
6889          :
6890          :
6891          :
6892          :
6893          :
6894          :
6895          :
6896          :
6897          :
6898          :
6899          :
6900          :
6901          :
6902          :
6903          :
6904          :
6905          :
6906          :
6907          :
6908          :
6909          :
6910          :
6911          :
6912          :
6913          :
6914          :
6915          :
6916          :
6917          :
6918          :
6919          :
6920          :
6921          :
6922          :
6923          :
6924          :
6925          :
6926          :
6927          :
6928          :
6929          :
6930          :
6931          :
6932          :
6933          :
6934          :
6935          :
6936          :
6937          :
6938          :
6939          :
6940          :
6941          :
6942          :
6943          :
6944          :
6945          :
6946          :
6947          :
6948          :
6949          :
6950          :
6951          :
6952          :
6953          :
6954          :
6955          :
6956          :
6957          :
6958          :
6959          :
6960          :
6961          :
6962          :
6963          :
6964          :
6965          :
6966          :
6967          :
6968          :
6969          :
6970          :
6971          :
6972          :
6973          :
6974          :
6975          :
6976          :
6977          :
6978          :
6979          :
6980          :
6981          :
6982          :
6983          :
6984          :
6985          :
6986          :
6987          :
6988          :
6989          :
6990          :
6991          :
6992          :
6993          :
6994          :
6995          :
6996          :
6997          :
6998          :
6999          :
7000          :
7001          :
7002          :
7003          :
7004          :
7005          :
7006          :
7007          :
7008          :
7009          :
7010          :
7011          :
7012          :
7013          :
7014          :
7015          :
7016          :
7017          :
7018          :
7019          :
7020          :
7021          :
7022          :
7023          :
7024          :
7025          :
7026          :
7027          :
7028          :
7029          :
7030          :
7031          :
7032          :
7033          :
7034          :
7035          :
7036          :
7037          :
7038          :
7039          :
7040          :
7041          :
7042          :
7043          :
7044          :
7045          :
7046          :
7047          :
7048          :
7049          :
7050          :
7051          :
7052          :
7053          :
7054          :
7055          :
7056          :
7057          :
7058          :
7059          :
7060          :
7061          :
7062          :
7063          :
7064          :
7065          :
7066          :
7067          :
7068          :
7069          :
7070          :
7071          :
7072          :
7073          :
7074          :
7075          :
7076          :
7077          :
7078          :
7079          :
7080          :
7081          :
7082          :
7083          :
7084          :
7085          :
7086          :
7087          :
7088          :
7089          :
7090          :
7091          :
7092          :
7093          :
7094          :
7095          :
7096          :
7097          :
7098          :
7099          :
7100          :
7101          :
7102          :
7103          :
7104          :
7105          :
7106          :
7107          :
7108          :
7109          :
7110          :
7111          :
7112          :
7113          :
7114          :
7115          :
7116          :
7117          :
7118          :
7119          :
7120          :
7121          :
7122          :
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          :
7179          :
7180          :
7181          :
7182          :
7183          :
7184          :
7185          :
7186          :
7187          :
7188          :
7189          :
7190          :
7191          :
7192          :
7193          :
7194          :
7195          :
7196          :
7197          :
7198          :
7199          :
7200          :
7201          :
7202          :
7203          :
7204          :
7205          :
7206          :
7207          :
7208          :
7209          :
7210          :
7211          :
7212          :
7213          :
7214          :
7215          :
7216          :
7217          :
7218          :
7219          :
7220          :
7221          :
7222          :
7223          :
7224          :
7225          :
7226          :
7227          :
7228          :
7229          :
7230          :
7231          :
7232          :
7233          :
7234          :
7235          :
7236          :
7237          :
7238          :
7239          :
7240          :
7241          :
7242          :
7243          :
7244          :
7245          :
7246          :
7247          :
7248          :
7249          :
7250          :
7251          :
7252          :
7253          :
7254          :
7255          :
7256          :
7257          :
7258          :
7259          :
7260          :
7261          :
7262          :
7263          :
7264          :
7265          :
7266          :
7267          :
7268          :
7269          :
7270          :
7271          :
7272          :
7273          :
7274          :
7275          :
7276          :
7277          :
7278          :
7279          :
7280          :
7281          :
7282          :
7283          :
7284          :
7285          :
7286          :
7287          :
7288          :
7289          :
7290          :
7291          :
7292          :
7293          :
7294          :
7295          :
7296          :
7297          :
7298          :
7299          :
7300          :
7301          :
7302          :
7303          :
7304          :
7305          :
7306          :
7307          :
7308          :
7309          :
7310          :
7311          :
7312          :
7313          :
7314          :
7315          :
7316          :
7317          :
7318          :
7319          :
7320          :
7321          :
7322          :
7323          :
7324          :
7325          :
7326          :
7327          :
7328          :
7329          :
7330          :
7331          :
7332          :
7333          :
7334          :
7335          :
7336          :
7337          :
7338          :
7339          :
7340          :
7341          :
7342          :
7343          :
7344          :
7345          :
7346          :
7347          :
7348          :
7349          :
7350          :
7351          :
7352          :
7353          :
7354          :
7355          :
7356          :
7357          :
7358          :
7359          :
7360          :
7361          :
7362          :
7363          :
7364          :
7365          :
7366          :
7367          :
7368          :
7369          :
7370          :
7371          :
7372          :
7373          :
7374          :
7375          :
7376          :
7377          :
7378          :
7379          :
7380          :
7381          :
7382          :
7383          :
7384          :
7385          :
7386          :
7387          :
7388          :
7389          :
7390          :
7391          :
7392          :
7393          :
7394          :
7395          :
7396          :
7397          :
7398          :
7399          :
7400          :
7401          :
7402          :
7403          :
7404          :
7405          :
7406          :
7407          :
7408          :
7409          :
7410          :
7411          :
7412          :
7413          :
7414          :
7415          :
7416          :
7417          :
7418          :
7419          :
7420          :
7421          :
7422          :
7423          :
7424          :
7425          :
7426          :
7427          :
7428          :
7429          :
7430          :
7431          :
7432          :
7433          :
7434          :
7435          :
7436          :
7437          :
7438          :
7439          :
7440          :
7441          :
7442          :
7443          :
7444          :
7445          :
7446          :
7447          :
7448          :
7449          :
7450          :
7451          :
7452          :
7453          :
7454          :
7455          :
7456          :
7457          :
7458          :
7459          :
7460          :
7461          :
7462          :
7463          :
7464          :
7465          :
7466          :
7467          :
7468          :
7469          :
7470          :
7471          :
7472          :
7473          :
7474          :
7475          :
7476          :
7477          :
7478          :
7479          :
7480          :
7481          :
7482          :
7483          :
7484          :
7485          :
7486          :
7487          :
7488          :
7489          :
7490          :
7491          :
7492          :
7493          :
7494          :
7495          :
7496          :
7497          :
7498          :
7499          :
7500          :
7501          :
7502          :
7503          :
7504          :
7505          :
7506          :
7507          :
7508          :
7509          :
7510          :
7511          :
7512          :
7513          :
7514          :
7515          :
7516          :
7517          :
7518          :
7519          :
752
```



```

5945 022016 010177 162060 1$: MOV R1,@RHDB ;WRITE INTO RHDB
5946 022022 000261 SEC ;SET CARRY
5947 022024 006101 ROL R1 ;GET ZERO ONE BIT LEFT
5948 022026 102402 BVS 2$ ;BRANCH IF NOT LAST WORD IN PATTERN
5949 022030 012701 177776 MOV #177776, R1 ;ELSE START PATTERN OVER
5950 022034 005302 2$: DEC R2 ;COUNT DOWN
5951 022036 001367 BNE 1$ ;BRANCH IF NOT DONE
5952 ;WAIT FOR OR BIT IN RHCS2 REGISTER TO SET
5953 022040 104411 WAT ;TRAP TO WAIT.T SUBROUTINE
5954 022042 004070 RHCS2 ;AND WAIT FOR OR BIT IN
5955 022044 000200 OR ;RHCS2 REGISTER
5956 ;IF ERROR OCCURS HERE
5957 ;IT MEANS 'OR' DID NOT
5958 ;SET FOR THE FULL COUNT
5959 ;DOWN OF THE WAIT.T SUBROUTINE
5960 ;THIS TIME IS APPROXIMATELY
5961 ;LARGER THAN 500 MILLISECONDS
5962
5963
5964 ;CHECK THAT RHCS2 HAS OR
5965 022046 012737 000200 001124 MOV #OR,$GDDAT ;GET GOOD = 200
5966 022054 053737 006442 001124 BIS UNIT,$GDDAT ;INCLUDE UNIT NUMBER
5967
5968 022062 017737 162002 001126 MOV @RHCS2,$BDDAT ;READ RHCS2 FOR COMPARISON
5969 022070 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
5970 ;DATA WITH DATA READ FROM
5971 ;RHCS2
5972 022076 001401 BEQ 67$ ;BRANCH IF GOOD
5973 022100 104023 ERROR 23
5974 ;AFTER SETTING 'CLR' BIT #5
5975 ;IN RHCS2 TO INIT THE RH
5976 ;AND HAVING WRITTEN A
5977 ;PATTERN OF FLOATING ZEROS IN UPPER BYTE
5978 ;177377, 176777, 175777, 173777,
5979 ;167777, 157777, 137777, 77777
5980 ;INTO RHDB TO FILL IT
5981 ;RHCS2 SHOULD HAVE OR
5982 ;=200
5983 ;TOGETHER WITH UNIT NUMBER
5984 ;BUT CONTAINED WHAT IS
5985 ;GIVEN IN BAD RHCS2
5986 022102 67$:
5987
5988 ;READ RHDB UNTIL THE SILO IS EMPTY
5989 ;VERIFY THE COUNT PATTERN AS READ FROM THE SILO
5990 ;IS THE SAME PATTERN AS PREVIOUSLY WRITTEN INTO THE SILO
5991 ;ELSE REPORT THE ERROR
5992
5993 022102 005037 006304 CLR SILOM ;INITIALIZE SILO WORD #
5994 022106 012737 177377 001124 MOV #177377, $GDDAT ;INITIALIZE EXPECTED DATA
5995 022114 005237 006304 68$: INC SILOM ;COUNT ONE SILO WORD
5996 022120 023737 006304 006426 CMP SILOM, SILOSZ ;IS THE SILO EMPTY?
5997 022126 101017 BHI 70$ ;BRANCH IF YES
5998 022130 017737 161746 001126 MOV @RHDB, $BDDAT ;GET 177377 WORD FROM THE SILO
5999 022136 023737 001124 001126 CMP $GDDAT, $BDDAT ;EXPECTED=ACTUAL?
6000 022144 001401 BEQ 69$ ;BRANCH IF YES
    
```

```

6001 022146 104024          ERROR 24          ;REPORT THE ERROR
6002 022150          69$:  ROL    $GDDAT          ;GET NEXT EXPECTED DATA
6003 022150 006137 001124  BVS    68$          ;BRANCH IF NOT LAST WORD IN PATTERN
6004 022154 102757          MOV    #177776,    $GDDAT ;ELSE START PATTERN OVER
6005 022156 012737 177776 001124 BR     68$          ;CONTINUE
6006 022164 000753
6007 022166          70$:
6008
6009
6010
6011          ;CHECK THAT RHCS2 HAS IR
6012 022166 012737 000100 001124 MOV    #IR,$GDDAT ;GET GOOD = 100
6013 022174 053737 006442 001124 BIS    UNIT,$GDDAT ;INCLUDE UNIT NUMBER
6014
6015 022202 017737 161662 001126 MOV    @RHCS2,$BDDAT ;READ RHCS2 FOR COMPARISON
6016 022210 023737 001124 001126 CMP    $GDDAT,$BDDAT ;COMPARE EXPECTED
6017          ;DATA WITH DATA READ FROM
6018          ;RHCS2
6019 022216 001401          BEQ    72$          ;BRANCH IF GOOD
6020 022220 104025          ERROR 25
6021          ;AFTER SETTING 'CLR' BIT #5
6022          ;IN RHCS2 TO INIT THE RH
6023          ;AND HAVING WRITTEN A
6024          ;PATTERN OF FLOATING ZEROS IN UPPER BYTE
6025          ;177377, 176777, 175777, 173777,
6026          ;167777, 157777, 137777, 77777
6027          ;INTO RHDB TO FILL IT
6028          ;RHCS2 SHOULD HAVE IR
6029          ;=100
6030          ;TOGETHER WITH UNIT NUMBER
6031          ;BUT CONTAINED WHAT IS
6032          ;GIVEN IN BAD RHCS2
6033 022222          72$:
6034          ;CHECK THAT RHCS1 HAS DVA!RDY
6035 022222 012737 004200 001124 MOV    #DVA!RDY,$GDDAT ;GET GOOD = 4200
6036
6037 022230 017737 161624 001126 MOV    @RHCS1,$BDDAT ;READ RHCS1 FOR COMPARISON
6038 022236 023737 001124 001126 CMP    $GDDAT,$BDDAT ;COMPARE EXPECTED
6039          ;DATA WITH DATA READ FROM
6040          ;RHCS1
6041 022244 001401          BEQ    74$          ;BRANCH IF GOOD
6042 022246 104026          ERROR 26
6043          ;AFTER SETTING 'CLR' BIT #5
6044          ;IN RHCS2 TO INIT THE RH
6045          ;AND HAVING WRITTEN A
6046          ;PATTERN OF FLOATING ZEROS IN UPPER BYTE
6047          ;177377, 176777, 175777, 173777,
6048          ;167777, 157777, 137777, 77777
6049          ;INTO RHDB TO FILL IT
6050          ;RHCS1 SHOULD HAVE DVA!RDY
6051          ;=4200
6052          ;BUT CONTAINED WHAT IS
6053          ;GIVEN IN BAD RHCS1
6054 022250          74$:
6055          ;CHECK THAT RHCS3 HAS 0
6056 022250 012737 000000 001124 MOV    #0,$GDDAT ;GET GOOD = 0

```



```

6113                                     :GIVEN IN BAD RHBAE
6114 022352                               80$:
6115                                     :CHECK THAT RHWC HAS 0
6116 022352 012737 000000 001124         MOV    #0,$GDDAT      :GET GOOD = 0
6117
6118 022360 017737 161476 001126         MOV    @RHWC,$BDDAT   :READ RHWC FOR COMPARISON
6119 022366 023737 001124 001126         CMP    $GDDAT,$BDDAT :COMPARE EXPECTED
6120                                     :DATA WITH DATA READ FROM
6121                                     :RHWC
6122 022374 001401                         BEQ    82$            :BRANCH IF GOOD
6123 022376 104032                         ERROR  32
6124                                     :AFTER SETTING 'CLR' BIT #5
6125                                     :IN RHCS2 TO INIT THE RH
6126                                     :AND HAVING WRITTEN A
6127                                     :PATTERN OF FLOATING ZEROS IN UPPER BYTE
6128                                     :177377, 176777, 175777, 173777,
6129                                     :167777, 157777, 137777, 77777
6130                                     :INTO RHDB TO FILL IT
6131                                     :RHWC SHOULD HAVE 0
6132                                     :BUT CONTAINED WHAT IS
6133                                     :GIVEN IN BAD RHWC

```

```

6134 022400                               82$:
6135
6136
6137
6138
6139
6140
6141
6142
6143
6144
6145
6146
6147
6148
6149
6150
6151
6152
6153
6154
6155

```

```

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

```

```

6156 022400 000004                         TST21: SCOPE
6157 022402 012706 001000                 MOV    #STACK,SP     :RESET STACK
6158 022406 012737 000021 006276         MOV    #21,@TSTNM    :SAVE TEST NUMBER
6159
6160 022414 004737 041222                 JSR    PC,@#CLDISK   :GIVE RH INITIALIZE
6161                                     :SETUP UNIT NUBER
6162                                     :CLEAR RHWC AND FUNCTION BITS IN RHCS1
6163
6164
6165                                     :CHECK THAT RHER1 HAS 0
6166 022420 012737 000000 001124         MOV    #0,$GDDAT     :GET GOOD = 0
6167
6168 022426 017737 161442 001126         MOV    @RHER1,$BDDAT :READ RHER1 FOR COMPARISON

```



```

6225 022530          69$:
6226
6227                ;WRITE A ONE INTO TRE (RHCS1 - BIT #14)
6228                ;THIS SHOULD NOT CLEAR MCPE OR SC
6229 022530 012777 040000 161322  MOV    #TRE,@RHCS1    ;WRITE '1' INTO TRE IN RHCS1
6230
6231
6232
6233                ;CHECK THAT RHCS1 HAS 104200
6234 022536 012737 104200 001124  MOV    #104200,$GDDAT ;GET GOOD = 124200
6235
6236 022544 017737 161310 001126  MOV    @RHCS1,$BDDAT  ;READ RHCS1 FOR COMPARISON
6237 022552 023737 001124 001126  CMP    $GDDAT,$BDDAT ;COMPARE EXPECTED
6238                ;DATA WITH DATA READ FROM
6239                ;RHCS1
6240 022560 001401          BEQ    71$          ;BRANCH IF GOOD
6241 022562 104036          ERROR  36
6242
6243                ;AFTER CLEARING THE RH AND
6244                ;DEVICE REGISTERS BY AN
6245                ;RH CLEAR
6246                ;RHER1 WAS CHECKED TO HAVE
6247                ;ZEROS
6248                ;WRONG PARITY CHECKING WAS
6249                ;SET BY 'PAT' BIT IN RHCS2
6250                ;RHER1 WAS READ TO SET
6251                ;MCPE AND SC IN RHCS1
6252                ;ON WRITING '1' INTO TRE
6253                ;RHCS1 SHOULD HAVE 104200
6254                ;=124200
6255                ;BUT CONTAINED WHAT IS
6256                ;GIVEN IN BAD RHCS1
6256 022564          71$:
6257
6258                ;NOW ERRORS WILL BE CLEARED BY RH CLEAR
6259                ;SET 'CLR' BIT #5 IN RHCS2
6260 022564 004737 041222  JSR    PC,@#CLDISK  ;CLEAR BY RH INIT
6261
6262
6263                ;CHECK THAT RHCS1 HAS DVA!RDY
6264 022570 012737 004200 001124  MOV    #DVA!RDY,$GDDAT ;GET GOOD = 4200
6265
6266 022576 017737 161256 001126  MOV    @RHCS1,$BDDAT  ;READ RHCS1 FOR COMPARISON
6267 022604 023737 001124 001126  CMP    $GDDAT,$BDDAT ;COMPARE EXPECTED
6268                ;DATA WITH DATA READ FROM
6269                ;RHCS1
6270 022612 001401          BEQ    73$          ;BRANCH IF GOOD
6271 022614 104037          ERROR  37
6272
6273                ;AFTER CLEARING THE RH AND
6274                ;DEVICE REGISTERS BY AN
6275                ;RH CLEAR
6276                ;RHER1 WAS CHECKED TO HAVE
6277                ;ZEROS
6278                ;WRONG PARITY CHECKING WAS
6279                ;SET BY 'PAT' BIT IN RHCS2
6280                ;RHER1 WAS READ TO SET
                ;MCPE AND SC IN RHCS1

```


6393
6394 022776 001401
6395 023000 104044
6396
6397
6398
6399
6400
6401
6402
6403
6404
6405
6406
6407
6408
6409

BEG 83\$
ERROR 44

;RHC
;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
;RHC SHOULD HAVE 0
;BUT CONTAINED WHAT IS
;GIVEN IN BAD RHC

023002

83\$:

6410
6411
6412
6413
6414
6415
6416
6417
6418
6419
6420
6421
6422
6423
6424
6425
6426

:*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 RDY SET
:* CHECK RHCS2, RHCS3, RHBA, RHBAE, RHC TO HAVE APPROPRIATE
:* VALUES

6427 023002 000004
6428 023004 012706 001000
6429 023010 012737 000022 006276
6430
6431 023016 004737 041222

TST22: SCOPE
MOV #STACK,SP ;RESET STACK
MOV #22,@TSTNM ;SAVE TEST NUMBER
JSR PC,@CLDISK ;GIVE RH INITIALIZE
;SETUP UNIT NUBER
;CLEAR RHC AND FUNCTION BITS IN RHCS1

6432
6433
6434
6435
6436
6437
6438

;WRITE A ONE INTO RHDA - DISK ADDRESS REGISTER
;THIS REGISTER IN TAPE DRIVES IS CALLED 'RHFC FRAME COUNT'
;AFTER WRITING THIS '1' THEN ON READING THIS REGISTER
;THE CONTROL PARITY LINE WILL HAVE ZERO
MOV #BIT0,@RHDA ;WRITE '1' INTO RHDA
;IN TAPE DRIVES CALLED RHFC

6439 023022 012777 000001 161036
6440
6441
6442
6443 023030 052777 000020 161032
6444
6445

;INVERT THE PARITY TO BE CHECKED BY SETTING 'PAT' BIT #4 IN RHCS2
BIS #PAT,@RHCS2 ;SET PAT IN RHCS2

6446
6447 023036 012737 000001 001124
6448

;CHECK THAT RHDA HAS BIT0
MOV #BIT0,\$GDDAT ;GET GOOD = 1

```

6449 023044 017737 161016 001126      MOV    @RHDA,$BDDAT    ;READ RHDA FOR COMPARISON
6450 023052 023737 001124 001126      CMP    $GDDAT,$BDDAT  ;COMPARE EXPECTED
6451                                     ;DATA WITH DATA READ FROM
6452                                     ;RHDA
6453 023060 001401                      BEQ    65$             ;BRANCH IF GOOD
6454 023062 104045                      ERROR  45
6455                                     ;AFTER AN RH CLEAR '1'
6456                                     ;WAS WRITTEN INTO ADDRESS REGISTER
6457                                     ;(FRAME COUNT IN TAPE)
6458                                     ;PAT IN RHCS2 WAS SET
6459                                     ;THEN THE ADDRESS REGISTER
6460                                     ;WAS READ BACK
6461                                     ;RHDA SHOULD HAVE BIT0
6462                                     ;=1
6463                                     ;BUT CONTAINED WHAT IS
6464                                     ;GIVEN IN BAD RHDA
6465 023064                                65$:
6466
6467                                     ;CHECK THAT RHCS1 HAS 4200
6468 023064 012737 004200 001124      MOV    #4200,$GDDAT   ;GET GOOD = 124000
6469
6470 023072 017737 160762 001126      MOV    @RHCS1,$BDDAT ;READ RHCS1 FOR COMPARISON
6471 023100 023737 001124 001126      CMP    $GDDAT,$BDDAT ;COMPARE EXPECTED
6472                                     ;DATA WITH DATA READ FROM
6473                                     ;RHCS1
6474 023106 001401                      BEQ    67$             ;BRANCH IF GOOD
6475 023110 104046                      ERROR  46
6476                                     ;AFTER AN RH CLEAR '1'
6477                                     ;WAS WRITTEN INTO ADDRESS REGISTER
6478                                     ;(FRAME COUNT IN TAPE)
6479                                     ;PAT IN RHCS2 WAS SET
6480                                     ;THEN THE ADDRESS REGISTER
6481                                     ;WAS READ BACK
6482                                     ;RHCS1 SHOULD HAVE 4200
6483                                     ;=124000
6484                                     ;BUT CONTAINED WHAT IS
6485                                     ;GIVEN IN BAD RHCS1
6486 023112                                67$:
6487                                     ;CHECK THAT RHCS2 HAS IR!PAT
6488 023112 012737 000120 001124      MOV    #IR!PAT,$GDDAT ;GET GOOD = 120
6489 023120 053737 006442 001124      BIS    UNIT,$GDDAT    ;INCLUDE UNIT NUMBER
6490
6491 023126 017737 160736 001126      MOV    @RHCS2,$BDDAT ;READ RHCS2 FOR COMPARISON
6492 023134 023737 001124 001126      CMP    $GDDAT,$BDDAT ;COMPARE EXPECTED
6493                                     ;DATA WITH DATA READ FROM
6494                                     ;RHCS2
6495 023142 001401                      BEQ    69$             ;BRANCH IF GOOD
6496 023144 104047                      ERROR  47
6497                                     ;AFTER AN RH CLEAR '1'
6498                                     ;WAS WRITTEN INTO ADDRESS REGISTER
6499                                     ;(FRAME COUNT IN TAPE)
6500                                     ;PAT IN RHCS2 WAS SET
6501                                     ;THEN THE ADDRESS REGISTER
6502                                     ;WAS READ BACK
6503                                     ;RHCS2 SHOULD HAVE IR!PAT
6504                                     ;=120

```



```

6561                                     :WAS READ BACK
6562                                     :RHBAE SHOULD HAVE 0
6563                                     :BUT CONTAINED WHAT IS
6564                                     :GIVEN IN BAD RHBAE
6565 023250 75$:
6566                                     :CHECK THAT RHCW HAS 0
6567 023250 012737 000000 001124 MOV #0,$GDDAT ;GET GOOD = 0
6568
6569 023256 017737 160600 001126 MOV @RHCW,$BDDAT ;READ RHCW FOR COMPARISON
6570 023264 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
6571                                     :DATA WITH DATA READ FROM
6572                                     :RHCW
6573 023272 001401 BEQ 77$ ;BRANCH IF GOOD
6574 023274 104053 ERROR 53
6575                                     :AFTER AN RH CLEAR '1'
6576                                     :WAS WRITTEN INTO ADDRESS REGISTER
6577                                     : (FRAME COUNT IN TAPE)
6578                                     :PAT IN RHCS2 WAS SET
6579                                     :THEN THE ADDRESS REGISTER
6580                                     :WAS READ BACK
6581                                     :RHCW SHOULD HAVE 0
6582                                     :BUT CONTAINED WHAT IS
6583                                     :GIVEN IN BAD RHCW
6584 023276 77$:
6585
6586
6587
6588 *****
6589 :*TEST 23 TEST DUPLICATED A16 (RHCS1 BIT #8)
6590
6591 :* CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
6592 :* MOVE 400 (ONE INTO A16) IN RHCS1
6593 :* READ RHCS1 TO CONTAIN 600 (BIT #8 AND RDY)
6594 :* READ RHBAE TO CONTAIN '1' (BIT #0 HIGH)
6595
6596 :* MOVE 0 INTO RHBAE (WRITING 0 INTO RHBAE BIT #0)
6597 :* READ RHBAE TO CONTAIN 0
6598 :* READ RHCS1 TO CONTAIN ONLY RDY (BIT #8 IN RHCS1 IS ZERO)
6599
6600
6601 *****
6602 023276 000004 TST23: SCOPE
6603 023300 012706 001000 MOV #STACK,SP ;RESET STACK
6604 023304 012737 000023 006276 MOV #23,@TSTNM ;SAVE TEST NUMBER
6605
6606 023312 004737 041222 JSR PC,@CLDISK ;GIVE RH INITIALIZE
6607 ;SETUP UNIT NUBER
6608 ;CLEAR RHCW AND FUNCTION BITS IN RHCS1
6609
6610 ;WRITE '1' INTO A16 IN RHCS1
6611 023316 012777 000400 160534 MOV #A16,@RHCS1 ;MOVE 400 INTO RHCS1
6612
6613
6614
6615
6616 023324 012737 004600 001124 ;CHECK THAT RHCS1 HAS A16!DVA!RDY
MOV #A16!DVA!RDY,$GDDAT ;GET GOOD = 4600

```



```

6617
6618 023332 017737 160522 001126      MOV    @RHCS1,$BDDAT    ;READ RHCS1 FOR COMPARISON
6619 023340 023737 001124 001126      CMP    $GDDAT,$BDDAT   ;COMPARE EXPECTED
6620                                     ;DATA WITH DATA READ FROM
6621                                     ;RHCS1
6622 023346 001401      BEQ    65$              ;BRANCH IF GOOD
6623 023350 104054      ERROR  54
6624                                     ;AFTER SETTING 'CLR' BIT #5
6625                                     ;IN RHCS2 TO INIT THE RH
6626                                     ;A16 WAS WRITTEN INTO RHCS1
6627                                     ;RHCS1 WAS READ
6628                                     ;RHCS1 SHOULD HAVE A16!DVA!RDY
6629                                     ;=4600
6630                                     ;BUT CONTAINED WHAT IS
6631                                     ;GIVEN IN BAD RHCS1
6632 023352                65$:
6633
6634
6635                                     ;CHECK THAT RHBAE HAS BIT0
6636 023352 012737 000001 001124      MOV    #BIT0,$GDDAT    ;GET GOOD = 1
6637
6638 023360 017737 160544 001126      MOV    @RHBAE,$BDDAT   ;READ RHBAE FOR COMPARISON
6639 023366 023737 001124 001126      CMP    $GDDAT,$BDDAT   ;COMPARE EXPECTED
6640                                     ;DATA WITH DATA READ FROM
6641                                     ;RHBAE
6642 023374 001401      BEQ    67$              ;BRANCH IF GOOD
6643 023376 104055      ERROR  55
6644                                     ;AFTER SETTING 'CLR' BIT #5
6645                                     ;IN RHCS2 TO INIT THE RH
6646                                     ;A16 WAS WRITTEN INTO RHCS1
6647                                     ;RHCS1 WAS READ
6648                                     ;THEN RHBAE WAS READ
6649                                     ;RHBAE SHOULD HAVE BIT0
6650                                     ;=1
6651                                     ;BUT CONTAINED WHAT IS
6652                                     ;GIVEN IN BAD RHBAE
6653 023400                67$:
6654
6655                                     ;NOW MOVE 0 INTO RHBAE (WRITING 0 INTO RHBAE BIT #0)
6656 023400 005077 160524      CLR    @RHBAE          ;WRITE ZERO INTO RHBAE
6657
6658
6659                                     ;CHECK THAT RHBAE HAS 0
6660 023404 012737 000000 001124      MOV    #0,$GDDAT      ;GET GOOD = 0
6661
6662 023412 017737 160512 001126      MOV    @RHBAE,$BDDAT   ;READ RHBAE FOR COMPARISON
6663 023420 023737 001124 001126      CMP    $GDDAT,$BDDAT   ;COMPARE EXPECTED
6664                                     ;DATA WITH DATA READ FROM
6665                                     ;RHBAE
6666 023426 001401      BEQ    69$              ;BRANCH IF GOOD
6667 023430 104056      ERROR  56
6668                                     ;AFTER SETTING 'CLR' BIT #5
6669                                     ;IN RHCS2 TO INIT THE RH
6670                                     ;A16 WAS WRITTEN INTO RHCS1
6671                                     ;RHCS1 WAS READ
6672                                     ;RHBAE WAS READ

```

```

6673                                     ; THEN ZERO WAS WRITTEN
6674                                     ; INTO RHBAE
6675                                     ; ON READING RHBAE
6676                                     ; RHBAE SHOULD HAVE 0
6677                                     ; BUT CONTAINED WHAT IS
6678                                     ; GIVEN IN BAD RHBAE
6679 023432                               69$:
6680
6681
6682                                     ; CHECK THAT RHCS1 HAS DVA!RDY
6683 023432 012737 004200 001124          MOV   #DVA!RDY,$GDDAT ; GET GOOD = 4200
6684
6685 023440 017737 160414 001126          MOV   @RHCS1,$BDDAT ; READ RHCS1 FOR COMPARISON
6686 023446 023737 001124 001126          CMP   $GDDAT,$BDDAT ; COMPARE EXPECTED
6687                                     ; DATA WITH DATA READ FROM
6688                                     ; RHCS1
6689 023454 001401                          BEQ   71$ ; BRANCH IF GOOD
6690 023456 104057                          ERROR 57
6691
6692                                     ; AFTER SETTING 'CLR' BIT #5
6693                                     ; IN RHCS2 TO INIT THE RH
6694                                     ; A16 WAS WRITTEN INTO RHCS1
6695                                     ; RHCS1 WAS READ
6696                                     ; RHBAE WAS READ
6697                                     ; THEN ZERO WAS WRITTEN
6698                                     ; INTO RHBAE
6699                                     ; RHBAE WAS READ
6700                                     ; ON READING RHCS1
6701                                     ; RHCS1 SHOULD HAVE DVA!RDY
6702                                     ; =4200
6703                                     ; BUT CONTAINED WHAT IS
6704 023460                               71$:
6705
6706
6707
6708
6709
6710
6711
6712
6713
6714
6715
6716
6717
6718
6719
6720
6721

```

```

*****
*TEST 24          TEST DUPLICATED A17 (RHCS1 BIT #9)

```

```

* 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)

```

```

6722 023460 000004          TST24: SCOPE
6723 023462 012706 001000          MOV   #STACK,SP ; RESET STACK
6724 023466 012737 000024 006276          MOV   #24,@#TSTNM ; SAVE TEST NUMBER
6725
6726 023474 004737 041222          JSR   PC,@#CLDISK ; GIVE RH INITIALIZE
6727                                     ; SETUP UNIT NUBER
6728                                     ; CLEAR RHWC AND FUNCTION BITS IN RHCS1

```



```

6785                                     :RHBAE
6786 023610 001401                       BEQ 69$      :BRANCH IF GOOD
6787 023612 104062                       ERROR 62
6788                                     :AFTER SETTING 'CLR' BIT #5
6789                                     :IN RHCS2 TO INIT THE RH
6790                                     :A17 WAS WRITTEN INTO RHCS1
6791                                     :RHCS1 WAS READ
6792                                     :RHBAE WAS READ
6793                                     :THEN ZERO WAS WRITTEN
6794                                     :INTO RHBAE
6795                                     :ON READING RHBAE
6796                                     :RHBAE SHOULD HAVE 0
6797                                     :BUT CONTAINED WHAT IS
6798                                     :GIVEN IN BAD RHBAE

```

6799 023614 69\$:

```

6800
6801
6802                                     :CHECK THAT RHCS1 HAS DVA!RDY
6803 023614 012737 004200 001124         MOV #DVA!RDY,$GDDAT :GET GOOD = 4200
6804
6805 023622 017737 160232 001126         MOV @RHCS1,$BDDAT   :READ RHCS1 FOR COMPARISON
6806 023630 023737 001124 001126         CMP $GDDAT,$BDDAT  :COMPARE EXPECTED
6807                                     :DATA WITH DATA READ FROM
6808                                     :RHCS1
6809 023636 001401                       BEQ 71$      :BRANCH IF GOOD
6810 023640 104063                       ERROR 63

```

```

6811                                     :AFTER SETTING 'CLR' BIT #5
6812                                     :IN RHCS2 TO INIT THE RH
6813                                     :A17 WAS WRITTEN INTO RHCS1
6814                                     :RHCS1 WAS READ
6815                                     :RHBAE WAS READ
6816                                     :THEN ZERO WAS WRITTEN
6817                                     :INTO RHBAE
6818                                     :RHBAE WAS READ
6819                                     :ON READING RHCS1
6820                                     :RHCS1 SHOULD HAVE DVA!RDY
6821                                     :=4200
6822                                     :BUT CONTAINED WHAT IS
6823                                     :GIVEN IN BAD RHCS1

```

6824 023642 71\$:

```

6825
6826
6827 .....
6828 :*TEST 25 TEST DUPLICATED IE (RHCS1 BIT #6)
6829
6830 :* CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
6831 :* MOVE 100 (ONE INTO IE) IN RHCS1
6832 :* READ RHCS1 TO CONTAIN 300 (BIT #6 AND RDY)
6833 :* READ RHCS3 TO CONTAIN '100' (BIT #6 HIGH)
6834 :*
6835 :*
6836 :* MOVE 0 INTO RHCS3 (WRITING 0 INTO RHCS3 BIT #6)
6837 :* READ RHCS3 TO CONTAIN 100
6838 :* READ RHCS1 TO CONTAIN ONLY RDY (BIT #6 IN RHCS1 IS ZERO)
6839 :*
6840 :*

```



```

6841 .....
6842 023642 000004          TST25: SCOPE
6843 023644 012706 001000  MOV      #STACK,SP      ;RESET STACK
6844 023650 012737 000025 006276  MOV      #25,@#TSTNM    ;SAVE TEST NUMBER
6845
6846 023656 004737 041222    JSR      PC,@#CLDISK    ;GIVE RH INITIALIZE
6847                                ;SETUP UNIT NUBER
6848                                ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
6849
6850                                ;WRITE '1' INTO IE IN RHCS1
6851 023662 012777 000100 160170  MOV      #IE,@RHCS1    ;MOVE 100 INTO RHCS1
6852
6853
6854
6855                                ;CHECK THAT RHCS1 HAS IE!DVA!RDY
6856 023670 012737 004300 001124  MOV      #IE!DVA!RDY,$GDDAT ;GET GOOD = 4300
6857
6858 023676 017737 160156 001126  MOV      @RHCS1,$BDDAT   ;READ RHCS1 FOR COMPARISON
6859 023704 023737 001124 001126  CMP      $GDDAT,$BDDAT  ;COMPARE EXPECTED
6860                                ;DATA WITH DATA READ FROM
6861                                ;RHCS1
6862 023712 001401          BEQ      65$            ;BRANCH IF GOOD
6863 023714 104064          ERROR    64
6864                                ;AFTER SETTING 'CLR' BIT #5
6865                                ;IN RHCS2 TO INIT THE RH
6866                                ;IE WAS WRITTEN INTO RHCS1
6867                                ;RHCS1 WAS READ
6868                                ;RHCS1 SHOULD HAVE IE!DVA!RDY
6869                                ;=4300
6870                                ;BUT CONTAINED WHAT IS
6871                                ;GIVEN IN BAD RHCS1
6872 023716          65$:
6873
6874
6875                                ;CHECK THAT RHCS3 HAS IE
6876 023716 012737 000100 001124  MOV      #IE,$GDDAT     ;GET GOOD = 1
6877
6878 023724 017737 160202 001126  MOV      @RHCS3,$BDDAT  ;READ RHCS3 FOR COMPARISON
6879 023732 023737 001124 001126  CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED
6880                                ;DATA WITH DATA READ FROM
6881                                ;RHCS3
6882 023740 001401          BEQ      67$            ;BRANCH IF GOOD
6883 023742 104065          ERROR    65
6884                                ;AFTER SETTING 'CLR' BIT #5
6885                                ;IN RHCS2 TO INIT THE RH
6886                                ;IE WAS WRITTEN INTO RHCS1
6887                                ;RHCS1 WAS READ
6888                                ;THEN RHCS3 WAS READ
6889                                ;RHCS3 SHOULD HAVE IE
6890                                ;=1
6891                                ;BUT CONTAINED WHAT IS
6892                                ;GIVEN IN BAD RHCS3
6893 023744          67$:
6894
6895
6896 023744 005077 160162    ;NOW MOVE 0 INTO RHCS3 (WRITING 0 INTO RHCS3 BIT #1)
        CLR      @RHCS3    ;WRITE ZERO INTO RHCS3
    
```

```
6897
6898
6899
6900 023750 012737 000000 001124      :CHECK THAT RHCS3 HAS 0
      MOV      #0,$GDDAT      :GET GOOD = 0
6901
6902 023756 017737 160150 001126      MOV      @RHCS3,$BDDAT      :READ RHCS3 FOR COMPARISON
6903 023764 023737 001124 001126      CMP      $GDDAT,$BDDAT      :COMPARE EXPECTED
6904                                     :DATA WITH DATA READ FROM
6905                                     :RHCS3
6906 023772 001401      BEQ      69$      :BRANCH IF GOOD
6907 023774 104066      ERROR    66
6908                                     :AFTER SETTING "CLR" BIT #5
6909                                     :IN RHCS2 TO INIT THE RH
6910                                     :IE WAS WRITTEN INTO RHCS1
6911                                     :RHCS1 WAS READ
6912                                     :RHCS3 WAS READ
6913                                     :THEN ZERO WAS WRITTEN
6914                                     :INTO RHCS3
6915                                     :ON READING RHCS3
6916                                     :RHCS3 SHOULD HAVE 0
6917                                     :BUT CONTAINED WHAT IS
6918                                     :GIVEN IN BAD RHCS3
```

6919 023776 69\$:

```
6920
6921
6922                                     :CHECK THAT RHCS1 HAS DVA!RDY
6923 023776 012737 004200 001124      MOV      #DVA!RDY,$GDDAT :GET GOOD = 4200
6924
6925 024004 017737 160050 001126      MOV      @RHCS1,$BDDAT      :READ RHCS1 FOR COMPARISON
6926 024012 023737 001124 001126      CMP      $GDDAT,$BDDAT      :COMPARE EXPECTED
6927                                     :DATA WITH DATA READ FROM
6928                                     :RHCS1
6929 024020 001401      BEQ      71$      :BRANCH IF GOOD
6930 024022 104067      ERROR    67
```

```
6931                                     :AFTER SETTING "CLR" BIT #5
6932                                     :IN RHCS2 TO INIT THE RH
6933                                     :IE WAS WRITTEN INTO RHCS1
6934                                     :RHCS1 WAS READ
6935                                     :RHCS3 WAS READ
6936                                     :THEN ZERO WAS WRITTEN
6937                                     :INTO RHCS3
6938                                     :RHCS3 WAS READ
6939                                     :ON READING RHCS1
6940                                     :RHCS1 SHOULD HAVE DVA!RDY
6941                                     :=4200
6942                                     :BUT CONTAINED WHAT IS
6943                                     :GIVEN IN BAD RHCS1
```

6944 024024 71\$:

```
6945
6946
6947 :*****
6948 :*TEST 26      RHCS1 PROGRAM ERROR (PGE BIT #10)
6949
6950 :*      CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
6951 :*      SET UP FOR A 6 WORD WRITE
6952 :*      SET "GO" (RHCS1 BIT #0) TWICE IN TWO SUCCESSIVE
```


6953
6954
6955
6956
6957
6958
6959
6960
6961
6962
6963
6964
6965
6966
6967
6968
6969
6970
6971
6972
6973
6974
6975
6976
6977
6978
6979
6980
6981
6982
6983
6984
6985
6986
6987
6988
6989
6990
6991
6992
6993
6994
6995
6996
6997
6998
6999
7000
7001
7002
7003
7004
7005
7006
7007
7008

024024 000004
024026 012706 001000
024032 012737 000026 006276
024040 004737 041222
024044 012777 177772 160010
024052 012777 004210 160004
024060 012737 177777 006264
024066 004077 162166
024072 004154
024074 052777 000001 157756
024102 000240
024104 052777 000001 157746
024112 000240
024114 000240

024116 012737 144061 001124
024124 017737 157730 001126
024132 023737 001124 001126

024140 001401
024142 104070

024144
024144 012737 002300 001124
024152 053737 006442 001124

TST26: SCOPE
MOV #STACK,SP ;RESET STACK
MOV #26,@TSTNM ;SAVE TEST NUMBER
JSR PC,@WCLDISK ;GIVE RH INITIALIZE
;SETUP UNIT NUBER
;CLEAR RHWC AND FUNCTION BITS IN RHCS1

;SET UP RH FOR A 6 WORD WRITE
MOV #-6,@RHWC ;WORD COUNT REGISTER=6
MOV #WRFROM,@RHBA ;BUS ADDRESS REGISTER=WRITBUF
;WRITE 6 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,@RHCS1 ;SET GO
NOP ;DELAY
BIS #GO,@RHCS1 ;SET GO WITHOUT TIME TO COMPLETE LAST GO
NOP

;CHECK THAT RHCS1 HAS SC!DVA!TRE!61
MOV #SC!DVA!TRE!61,\$GDDAT ;GET GOOD = 144000

MOV @RHCS1,\$BDDAT ;READ RHCS1 FOR COMPARISON
CMP \$GDDAT,\$BDDAT ;COMPARE EXPECTED
;DATA WITH DATA READ FROM
;RHCS1
BEQ 65\$;BRANCH IF GOOD
ERROR 70

;AFTER SETTING 'CLR' BIT #5
;IN RHCS2 TO INIT THE RH
;AND GIVING TWO SUCCESSIVE
;'GO' WITHOUT GIVING TIME FOR
;THE FIRST TO COMPLETE
;RHCS1 SHOULD HAVE SC!DVA!TRE!61
;=144000
;BUT CONTAINED WHAT IS
;GIVEN IN BAD RHCS1

65\$:
;CHECK THAT RHCS2 HAS PGE!OR!IR
MOV #PGE!OR!IR,\$GDDAT ;GET GOOD = 2300
BIS UNIT,\$GDDAT ;INCLUDE UNIT NUMBER

```

7009 024160 017737 157704 001126      MOV    @RHCS2,$BDDAT  ;READ RHCS2 FOR COMPARISON
7010 024166 023737 001124 001126      CMP    $GDDAT,$BDDAT  ;COMPARE EXPECTED
7011                                     ;DATA WITH DATA READ FROM
7012                                     ;RHCS2
7013 024174 001401      BEQ    67$            ;BRANCH IF GOOD
7014 024176 104071      ERROR  71
7015                                     ;AFTER SETTING 'CLR' BIT #5
7016                                     ;IN RHCS2 TO INIT THE RH
7017                                     ;AND GIVING TWO SUCCESSIVE
7018                                     ;'GO' WITHOUT GIVING TIME FOR
7019                                     ;THE FIRST TO COMPLETE
7020                                     ;RHCS2 SHOULD HAVE PGE!OR!IR
7021                                     ;=2300
7022                                     ;TOGETHER WITH UNIT NUMBER
7023                                     ;BUT CONTAINED WHAT IS
7024                                     ;GIVEN IN BAD RHCS2

```

```

7025 024200      67$:
7026                                     ;CHECK THAT RHCS3 HAS DBL
7027 024200 012737 002000 001124      MOV    #DBL,$GDDAT    ;GET GOOD = 0
7028
7029 024206 017737 157720 001126      MOV    @RHCS3,$BDDAT  ;READ RHCS3 FOR COMPARISON
7030 024214 023737 001124 001126      CMP    $GDDAT,$BDDAT  ;COMPARE EXPECTED
7031                                     ;DATA WITH DATA READ FROM
7032                                     ;RHCS3
7033 024222 001401      BEQ    69$            ;BRANCH IF GOOD
7034 024224 104072      ERROR  72
7035                                     ;AFTER SETTING 'CLR' BIT #5
7036                                     ;IN RHCS2 TO INIT THE RH
7037                                     ;AND GIVING TWO SUCCESSIVE
7038                                     ;'GO' WITHOUT GIVING TIME FOR
7039                                     ;THE FIRST TO COMPLETE
7040                                     ;RHCS3 SHOULD HAVE DBL
7041                                     ;BUT CONTAINED WHAT IS
7042                                     ;GIVEN IN BAD RHCS3

```

```

7043 024226      69$:
7044
7045
7046 .....
7047 *TEST 27      RHCS2 - BUS ADDRESS INHIBIT BIT #3
7048
7049 *
7050 * CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #6)
7051 * SET UP FOR A 10. WORD WRITE FROM A BUFFER TAGGED 'WRFROM'
7052 * SET BUS ADDRESS INHIBIT RHCS2 BIT #3 'BAI'
7053 * AT THE END OF WRITE CHECK
7054 * RHCS1 TO HAVE ONLY RDY AND COMMAND
7055 * RHCS2 TO HAVE BAI
7056 * RHCS3 TO HAVE 0
7057 * RHBA TO HAVE ADDRESS OF 'WRFROM'
7058 * RHBAE AND RHWC TO HAVE ZEROS
7059
7060 * NOW SET UP READ FOR THE SAME DATA WITH
7061 * BAI SET TO READ INTO A BUFFER TAGGED 'REINTO'
7062 * AFTER READ CHECK
7063 * RHCS1 TO HAVE ONLY RDY AND COMMAND
7064 * RHCS2 TO HAVE BAI
7065 * RHCS3 TO HAVE 0

```



```

7065      : *      RHBA TO HAVE ADDRESS OF 'REINTO'
7066      : *      RHBAE AND RHWC TO HAVE ZEROS
7067      : *      DATA IN REINTO BUFFER IS CHECKED WITH DATA IN
7068      : *      WRFROM BUFFER
7069      :*****
7070 024226 000004      TST27: SCOPE
7071 024230 012706 001000      MOV      #STACK,SP      ;RESET STACK
7072 024234 012737 000027 006276      MOV      #27,@#TSTNM    ;SAVE TEST NUMBER
7073
7074 024242 004737 041222      JSR      PC,@#CLDISK    ;GIVE RH INITIALIZE
7075                                ;SETUP UNIT NUBER
7076                                ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
7077
7078                                ;SET UP 10. WORD WRITE FROM BUFFER TAGGED 'WRFROM'
7079                                ;WITH BAI IN RHCS2
7080 024246 012777 177766 157606      MOV      #-10.,@RHWC    ;WORD COUNT REGISTER=10.
7081 024254 052777 000010 157606      BIS      #BAI,@RHCS2    ;BUS ADDRESS INHIBIT IN RHCS2
7082 024262 012777 004210 157574      MOV      #WRFROM,@RHBA  ;BUS ADDRESS REGISTER=WRFROM
7083 024270 004077 161764      JSR      R0,@COMND      ;GO TO DO COMMAND
7084 024274 004154      WRIDAT      ;WRITE DATA
7085
7086
7087                                ;CHECK THAT RHCS1 HAS DVA!RDY!60
7088 024276 012737 004260 001124      MOV      #DVA!RDY!60,$GDDAT ;GET GOOD = 4252
7089
7090 024304 017737 157550 001126      MOV      @RHCS1,$BDDAT    ;READ RHCS1 FOR COMPARISON
7091 024312 023737 001124 001126      CMP      $GDDAT,$BDDAT    ;COMPARE EXPECTED
7092                                ;DATA WITH DATA READ FROM
7093                                ;RHCS1
7094 024320 001401      BEQ      65$              ;BRANCH IF GOOD
7095 024322 104076      ERROR    76
7096
7097                                ;AFTER SETTING 'CLR' BIT #5
7098                                ;IN RHCS2 TO INIT THE RH
7099                                ;A 10. WORD WRITE WAS DONE
7100                                ;WITH BAI BIT IN RHCS2 SET
7101                                ;AT END OF WRITE
7102                                ;RHCS1 SHOULD HAVE DVA!RDY!60
7103                                ;=4252
7104                                ;BUT CONTAINED WHAT IS
7105                                ;GIVEN IN BAD RHCS1
7106
7107 024324 012737 000110 001124      65$: ;CHECK THAT RHCS2 HAS BAI!IR
7108 024332 053737 006442 001124      MOV      #BAI!IR,$GDDAT ;GET GOOD = 10
7109                                BIS      UNIT,$GDDAT    ;INCLUDE UNIT NUMBER
7110 024340 017737 157524 001126      MOV      @RHCS2,$BDDAT    ;READ RHCS2 FOR COMPARISON
7111 024346 023737 001124 001126      CMP      $GDDAT,$BDDAT    ;COMPARE EXPECTED
7112                                ;DATA WITH DATA READ FROM
7113                                ;RHCS2
7114 024354 001401      BEQ      67$              ;BRANCH IF GOOD
7115 024356 104077      ERROR    77
7116                                ;AFTER SETTING 'CLR' BIT #5
7117                                ;IN RHCS2 TO INIT THE RH
7118                                ;A 10. WORD WRITE WAS DONE
7119                                ;WITH BAI BIT IN RHCS2 SET
7120                                ;AT END OF WRITE
    
```



```

7177                                     :RHBAE SHOULD HAVE 0
7178                                     :BUT CONTAINED WHAT IS
7179                                     :GIVEN IN BAD RHBAE
7180 024462                               73$:
7181                                     :CHECK THAT RHWC HAS 0
7182 024462 012737 000000 001124      MOV    #0,$GDDAT      :GET GOOD = 0
7183
7184 024470 017737 157366 001126      MOV    @RHWC,$BDDAT  :READ RHWC FOR COMPARISON
7185 024476 023737 001124 001126      CMP    $GDDAT,$BDDAT :COMPARE EXPECTED
7186                                     :DATA WITH DATA READ FROM
7187                                     :RHWC
7188 024504 001401                       BEQ    75$           :BRANCH IF GOOD
7189 024506 104103                       ERROR  103
7190                                     :AFTER SETTING 'CLR' BIT #5
7191                                     :IN RHCS2 TO INIT THE RH
7192                                     :A 10. WORD WRITE WAS DONE
7193                                     :WITH BAI BIT IN RHCS2 SET
7194                                     :AT END OF WRITE
7195                                     :RHWC SHOULD HAVE 0
7196                                     :BUT CONTAINED WHAT IS
7197                                     :GIVEN IN BAD RHWC
7198 024510                               75$:
7199
7200
7201
7202
7203
7204
7205
7206
7207
7208
7209
7210
7211
7212
7213
7214
7215 024510 000004                       TST30: SCOPE
7216 024512 012706 001000               MOV    #STACK,SP    :RESET STACK
7217 024516 012737 000030 006276       MOV    #30,@TSTNM  :SAVE TEST NUMBER
7218
7219 024524 004737 041222               JSR    PC,@WCLDISK  :GIVE RH INITIALIZE
7220                                     :SETUP UNIT NUBER
7221                                     :CLEAR RHWC AND FUNCTION BITS IN RHCS1
7222 024530 012737 000000 006302       MOV    #0,IP        :IPCK BIT NO. FOR PRINTOUT
7223
7224 024536 052777 000001 157366       BIS    #IPCK0,@RHCS3 :SET IPCK0 IN RHCS3-BIT #0
7225 024544 005037 004210               CLR    WRFROM       :WRITE ZERO INTO LOCATION
7226 024550 013777 004210 157324       MOV    WRFROM,@RHDB :WRITE ZEROS INTO RHDB ONCE
7227
7228                                     :CHECK THAT RHCS1 HAS SC!TRE!DVA!RDY
7229 024556 012737 144200 001124       MOV    #SC!TRE!DVA!RDY,$GDDAT :GET GOOD = 144200
7230
7231 024564 017737 157270 001126       MOV    @RHCS1,$BDDAT :READ RHCS1 FOR COMPARISON
7232 024572 023737 001124 001126       CMP    $GDDAT,$BDDAT :COMPARE EXPECTED

```

```

*****
*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
*****

```



```

7401 ; IN RHCS2 TO INIT THE RH
7402 ; IPCK0 BIT #0 RHCS3 WAS SET
7403 ; ZEROS WERE MOVED INTO
7404 ; RHDB
7405 ; RHDB WAS READ THEN
7406 ; AN RH CLEAR (RHCS2 BIT #5)
7407 ; WAS GIVEN THIS SHOULD
7408 ; CLEAR ALL ERRORS
7409 ; RHBAE SHOULD HAVE 0
7410 ; BUT CONTAINED WHAT IS
7411 ; GIVEN IN BAD RHBAE

```

```

7412 025064          79$:
7413 ; CHECK THAT RHWC HAS 0
7414 025064 012737 000000 001124 MOV #0,$GDDAT
7415
7416 025072 017737 156764 001126 MOV @RHWC,$BDDAT
7417 025100 023737 001124 001126 CMP $GDDAT,$BDDAT
7418
7419
7420 025106 001401 BEQ 81$
7421 025110 104113 ERROR 113

```

; GET GOOD = 0

```

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

```

```

7422 ; AFTER SETTING "CLR" BIT #5
7423 ; IN RHCS2 TO INIT THE RH
7424 ; IPCK0 BIT #0 RHCS3 WAS SET
7425 ; ZEROS WERE MOVED INTO
7426 ; RHDB
7427 ; RHDB WAS READ THEN
7428 ; AN RH CLEAR (RHCS2 BIT #5)
7429 ; WAS GIVEN THIS SHOULD
7430 ; CLEAR ALL ERRORS
7431 ; RHWC SHOULD HAVE 0
7432 ; BUT CONTAINED WHAT IS
7433 ; GIVEN IN BAD RHWC

```

```

7434 025112          81$:
7435
7436
7437
7438
7439
7440
7441
7442
7443
7444
7445
7446
7447
7448
7449
7450

```

```

.....
* 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
.....

```

```

7451 025112 000004 T$131: SCOPE
7452 025114 012706 001000 MOV #STACK,SP ; RESET STACK
7453 025120 012737 000031 006276 MOV #31,@TSTNM ; SAVE TEST NUMBER
7454
7455 025126 004737 041222 JSR PC,@CLDISK ; GIVE RH INITIALIZE
7456 ; SETUP UNIT NUMBER

```



```
7569 ; IN RHCS2 TO INIT THE RH
7570 ; IPCK1 BIT #1 RHCS3 WAS SET
7571 ; ZEROS WERE MOVED INTO
7572 ; RHDB
7573 ; RHDB WAS READ THEN
7574 ; AN RH CLEAR (RHCS2 BIT #5)
7575 ; WAS GIVEN THIS SHOULD
7576 ; CLEAR ALL ERRORS
7577 ; RHCS2 SHOULD HAVE IR
7578 ; =100
7579 ; TOGETHER WITH UNIT NUMBER
7580 ; BUT CONTAINED WHAT IS
7581 ; GIVEN IN BAD RHCS2
7582 025364 73$:
7583 ; CHECK THAT RHCS3 HAS 0
7584 025364 012737 000000 001124 MOV #0,$GDDAT ; GET GOOD = 0
7585
7586 025372 017737 156534 001126 MOV @RHCS3,$BDDAT ; READ RHCS3 FOR COMPARISON
7587 025400 023737 001124 001126 CMP $GDDAT,$BDDAT ; COMPARE EXPECTED
7588 ; DATA WITH DATA READ FROM
7589 ; RHCS3
7590 025406 001401 BEQ 75$ ; BRANCH IF GOOD
7591 025410 104110 ERROR 110
7592 ; AFTER SETTING 'CLR' BIT #5
7593 ; IN RHCS2 TO INIT THE RH
7594 ; IPCK1 BIT #1 RHCS3 WAS SET
7595 ; ZEROS WERE MOVED INTO
7596 ; RHDB
7597 ; RHDB WAS READ THEN
7598 ; AN RH CLEAR (RHCS2 BIT #5)
7599 ; WAS GIVEN THIS SHOULD
7600 ; CLEAR ALL ERRORS
7601 ; RHCS3 SHOULD HAVE 0
7602 ; BUT CONTAINED WHAT IS
7603 ; GIVEN IN BAD RHCS3
7604 025412 75$:
7605 ; CHECK THAT RHBA HAS 0
7606 025412 012737 000000 001124 MOV #0,$GDDAT ; GET GOOD = 0
7607
7608 025420 017737 156440 001126 MOV @RHBA,$BDDAT ; READ RHBA FOR COMPARISON
7609 025426 023737 001124 001126 CMP $GDDAT,$BDDAT ; COMPARE EXPECTED
7610 ; DATA WITH DATA READ FROM
7611 ; RHBA
7612 025434 001401 BEQ 77$ ; BRANCH IF GOOD
7613 025436 104111 ERROR 111
7614 ; AFTER SETTING 'CLR' BIT #5
7615 ; IN RHCS2 TO INIT THE RH
7616 ; IPCK1 BIT #1 RHCS3 WAS SET
7617 ; ZEROS WERE MOVED INTO
7618 ; RHDB
7619 ; RHDB WAS READ THEN
7620 ; AN RH CLEAR (RHCS2 BIT #5)
7621 ; WAS GIVEN THIS SHOULD
7622 ; CLEAR ALL ERRORS
7623 ; RHBA SHOULD HAVE 0
7624 ; BUT CONTAINED WHAT IS
```



```
7625 ;GIVEN IN BAD RHBA
7626 025440 77$:
7627 ;CHECK THAT RHBAE HAS 0
7628 025440 012737 000000 001124 MOV #0,$GDDAT ;GET GOOD = 0
7629
7630 025446 017737 156456 001126 MOV @RHBAE,$BDDAT ;READ RHBAE FOR COMPARISON
7631 025454 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
7632 ;DATA WITH DATA READ FROM
7633 ;RHBAE
7634 025462 001401 BEQ 79$ ;BRANCH IF GOOD
7635 025464 104112 ERROR 112
7636 ;AFTER SETTING 'CLR' BIT #5
7637 ;IN RHCS2 TO INIT THE RH
7638 ;IPCK1 BIT #1 RHCS3 WAS SET
7639 ;ZEROS WERE MOVED INTO
7640 ;RHDB
7641 ;RHDB WAS READ THEN
7642 ;AN RH CLEAR (RHCS2 BIT #5)
7643 ;WAS GIVEN THIS SHOULD
7644 ;CLEAR ALL ERRORS
7645 ;RHBAE SHOULD HAVE 0
7646 ;BUT CONTAINED WHAT IS
7647 ;GIVEN IN BAD RHBAE
7648 025466 79$:
7649 ;CHECK THAT RHWC HAS 0
7650 025466 012737 000000 001124 MOV #0,$GDDAT ;GET GOOD = 0
7651
7652 025474 017737 156362 001126 MOV @RHWC,$BDDAT ;READ RHWC FOR COMPARISON
7653 025502 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
7654 ;DATA WITH DATA READ FROM
7655 ;RHWC
7656 025510 001401 BEQ 81$ ;BRANCH IF GOOD
7657 025512 104113 ERROR 113
7658 ;AFTER SETTING 'CLR' BIT #5
7659 ;IN RHCS2 TO INIT THE RH
7660 ;IPCK1 BIT #1 RHCS3 WAS SET
7661 ;ZEROS WERE MOVED INTO
7662 ;RHDB
7663 ;RHDB WAS READ THEN
7664 ;AN RH CLEAR (RHCS2 BIT #5)
7665 ;WAS GIVEN THIS SHOULD
7666 ;CLEAR ALL ERRORS
7667 ;RHWC SHOULD HAVE 0
7668 ;BUT CONTAINED WHAT IS
7669 ;GIVEN IN BAD RHWC
7670 025514 81$:
```

```
7671
7672
7673
7674
7675 *****
7676 *TEST 32 RHSC2 MDPE BIT #8 AND RHCS3 IPCK2 BIT #2
7677 * CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
7678 * SET IPCK2 (RHCS3 BIT #2)
7679 * MOVE ALL ZEROS INTO RHDB TWICE
7680 * THIS SHOULD NOT SET TRE SC IN RHCS1
* READ RHDB ONCE THIS SHOULD SET MDPE (RHCS2 BIT #8)
```

```

7681      :*      CHECK RHCS1 FOR RDY (BIT #7), TRE (BIT #14), SC (BIT #15)
7682      :*      READ RHDB A SECOND TIME. CHECK RHCS1, RHCS2, RHCS3
7683      :*      "CLR" (RHCS2 BIT #5) IS GIVEN
7684      :*      ALL ERRORS ARE CLEARED
7685      :*      CHECK RHCS1, RHCS2, RHCS3, RHBA, RHBAE, RHWC
7686
7687      :*****
7688 025514 000004      TST32: SCOPE
7689 025516 012706 001000      MOV      #STACK,SP      ;RESET STACK
7690 025522 012737 000032 006276      MOV      #32,@#TSTNM    ;SAVE TEST NUMBER
7691
7692 025530 004737 041222      JSR      PC,@#CLDISK    ;GIVE RH INITIALIZE
7693      :*      :SETUP UNIT NUBER
7694      :*      :CLEAR RHWC AND FUNCTION BITS IN RHCS1
7695 025534 012737 000002 006302      MOV      #2,IP          ;IPCK BIT NO. FOR PRINTOUT
7696
7697 025542 052777 000004 156362      BIS      #IPCK2,@RHCS3 ;SET IPCK2 IN RHCS3-BIT #2
7698 025550 005037 004210      CLR      WRFROM          ;WRITE ZERO INTO LOCATION
7699 025554 013777 004210 156320      MOV      WRFROM,@RHDB    ;WRITE ZEROS INTO RHDB ONCE
7700 025562 013777 004210 156312      MOV      WRFROM,@RHDB    ;WRITE ZERO SECOND TIME
7701
7702      :CHECK THAT RHCS1 HAS DVA!RDY
7703 025570 012737 004200 001124      MOV      #DVA!RDY,$GDDAT ;GET GOOD = 4200
7704
7705 025576 017737 156256 001126      MOV      @RHCS1,$BDDAT   ;READ RHCS1 FOR COMPARISON
7706 025604 023737 001124 001126      CMP      $GDDAT,$BDDAT   ;COMPARE EXPECTED
7707      :*      :DATA WITH DATA READ FROM
7708      :*      :RHCS1
7709 025612 001401      BEQ      65$             ;BRANCH IF GOOD
7710 025614 104140      ERROR    140
7711      :*      :AFTER SETTING "CLR" BIT #5
7712      :*      :IN RHCS2 TO INIT THE RH
7713      :*      :IPCK2 BIT #2 RHCS3 WAS SET
7714      :*      :ZEROS WERE MOVED INTO RHDB
7715      :*      :TWICE
7716      :*      :RHCS1 SHOULD HAVE DVA!RDY
7717      :*      :=4200
7718      :*      :BUT CONTAINED WHAT IS
7719      :*      :GIVEN IN BAD RHCS1
7720 025616      65$:
7721 025616 017737 156260 001200      MOV      @RHDB,$TMP1     ;READ RHDB ONCE
7722
7723      :CHECK THAT RHCS3 HAS 4
7724 025624 012737 000004 001124      MOV      #4,$GDDAT       ;GET GOOD = 4
7725
7726 025632 017737 156274 001126      MOV      @RHCS3,$BDDAT   ;READ RHCS3 FOR COMPARISON
7727 025640 023737 001124 001126      CMP      $GDDAT,$BDDAT   ;COMPARE EXPECTED
7728      :*      :DATA WITH DATA READ FROM
7729      :*      :RHCS3
7730 025646 001401      BEQ      67$             ;BRANCH IF GOOD
7731 025650 104104      ERROR    104
7732
7733      :*      :AFTER SETTING "CLR" BIT #5
7734      :*      :IN RHCS2 TO INIT THE RH
7735      :*      :IPCK2 BIT #2 RHCS3 WAS SET
7736      :*      :ZEROS WERE MOVED INTO
  
```



```
7737 ;RHDB
7738 ;RHDB WAS READ THEN
7739 ;RHCS3 SHOULD HAVE 4
7740 ;=4
7741 ;BUT CONTAINED WHAT IS
7742 ;GIVEN IN BAD RHCS3
7743 025652 67$:
7744
7745 ;CHECK THAT RHCS2 HAS MDPE!OR!IR
7746 025652 012737 000700 001124 MOV #MDPE!OR!IR,$GDDAT ;GET GOOD = 700
7747 025660 053737 006442 001124 BIS UNIT,$GDDAT ;INCLUDE UNIT NUMBER
7748
7749 025666 017737 156176 001126 MOV @RHCS2,$BDDAT ;READ RHCS2 FOR COMPARISON
7750 025674 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
7751 ;DATA WITH DATA READ FROM
7752 ;RHCS2
7753 025702 001401 BEQ 69$ ;BRANCH IF GOOD
7754 025704 104105 ERROR 105
7755 ;AFTER SETTING "CLR" BIT #5
7756 ;IN RHCS2 TO INIT THE RH
7757 ;IPCK2 BIT #2 RHCS3 WAS SET
7758 ;ZEROS WERE MOVED INTO
7759 ;RHDB
7760 ;RHDB WAS READ THEN
7761 ;RHCS2 SHOULD HAVE MDPE!OR!IR
7762 ;=700
7763 ;TOGETHER WITH UNIT NUMBER
7764 ;BUT CONTAINED WHAT IS
7765 ;GIVEN IN BAD RHCS2
7766 025706 69$:
7767
7768 025706 017737 156170 001202 MOV @RHDB,$TMP2 ;READ RHDB SECOND TIME
7769 ;CHECK THAT RHCS1 HAS SC!TRE!DVA!RDY
7770 025714 012737 144200 001124 MOV #SC!TRE!DVA!RDY,$GDDAT ;GET GOOD = 144200
7771
7772 025722 017737 156132 001126 MOV @RHCS1,$BDDAT ;READ RHCS1 FOR COMPARISON
7773 025730 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
7774 ;DATA WITH DATA READ FROM
7775 ;RHCS1
7776 025736 001401 BEQ 71$ ;BRANCH IF GOOD
7777 025740 104140 ERROR 140
7778 ;AFTER SETTING "CLR" BIT #5
7779 ;IN RHCS2 TO INIT THE RH
7780 ;IPCK2 BIT #2 RHCS3 WAS SET
7781 ;ZEROS WERE MOVED INTO
7782 ;RHDB
7783 ;RHDB WAS READ THEN
7784 ;RHCS1 SHOULD HAVE SC!TRE!DVA!RDY
7785 ;=144200
7786 ;BUT CONTAINED WHAT IS
7787 ;GIVEN IN BAD RHCS1
7788 025742 71$:
7789
7790 025742 012737 000500 001124 ;CHECK THAT RHCS2 HAS MDPE!IR
7791 025750 053737 006442 001124 MOV #MDPE!IR,$GDDAT ;GET GOOD = 500
7792 BIS UNIT,$GDDAT ;INCLUDE UNIT NUMBER
```



```

7961                                     ;IN RHCS2 TO INIT THE RH
7962                                     ;IPCK2 BIT #2 RHCS3 WAS SET
7963                                     ;ZEROS WERE MOVED INTO
7964                                     ;RHDB
7965                                     ;RHDB WAS READ THEN
7966                                     ;AN RH CLEAR (RHCS2 BIT #5)
7967                                     ;WAS GIVEN THIS SHOULD
7968                                     ;CLEAR ALL ERRORS
7969                                     ;RHW C SHOULD HAVE 0
7970                                     ;BUT CONTAINED WHAT IS
7971                                     ;GIVEN IN BAD RHW C

```

026242

87\$:

```

7972
7973
7974
7975
7976
7977
7978
7979
7980
7981
7982
7983
7984
7985
7986
7987
7988
7989

```

```

:*****
:*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)
:* 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, RHW C

```

```

7990 026242 000004
7991 026244 012706 001000
7992 026250 012737 000033 006276
7993
7994 026256 004737 041222
7995
7996
7997 026262 012737 000003 006302
7998
7999 026270 052777 000010 155634
8000 026276 005037 004210
8001 026302 013777 004210 155572
8002 026310 013777 004210 155564
8003
8004
8005 026316 012737 004200 001124
8006
8007 026324 017737 155530 001126
8008 026332 023737 001124 001126
8009
8010
8011 026340 001401
8012 026342 104140
8013
8014
8015
8016

```

```

TST33: SCOPE
MOV      #STACK,SP      ;RESET STACK
MOV      #33,@TSTNM     ;SAVE TEST NUMBER

JSR      PC,@#CLDISK    ;GIVE RH INITIALIZE
                        ;SETUP UNIT NUBER
                        ;CLEAR RHW C AND FUNCTION BITS IN RHCS1
MOV      #3,IP          ;IPCK BIT NO. FOR PRINTOUT

BIS      #IPCK3,@RHCS3  ;SET IPCK3 IN RHCS3-BIT #3
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    140

;AFTER SETTING 'CLR' BIT #5
;IN RHCS2 TO INIT THE RH
;IPCK3 BIT #3 RHCS3 WAS SET
;ZEROS WERE MOVED INTO RHDB

```

```

8017                                     ;TWICE
8018                                     ;RHCS1 SHOULD HAVE DVA!RDY
8019                                     ;=4200
8020                                     ;BUT CONTAINED WHAT IS
8021                                     ;GIVEN IN BAD RHCS1
8022 026344 017737 155532 001200 65$:  MOV    @RHDB,$TMP1    ;READ RHDB ONCE
8023 026344 017737 155532 001200
8024
8025
8026                                     ;CHECK THAT RHCS3 HAS 10
8027 026352 012737 000010 001124  MOV    #10,$GDDAT    ;GET GOOD = 10
8028
8029 026360 017737 155546 001126  MOV    @RHCS3,$BDDAT ;READ RHCS3 FOR COMPARISON
8030 026366 023737 001124 001126  CMP    $GDDAT,$BDDAT ;COMPARE EXPECTED
8031                                     ;DATA WITH DATA READ FROM
8032                                     ;RHCS3
8033 026374 001401  BEQ    67$    ;BRANCH IF GOOD
8034 026376 104104  ERROR   104
8035
8036                                     ;AFTER SETTING "CLR" BIT #5
8037                                     ;IN RHCS2 TO INIT THE RH
8038                                     ;IPCK3 BIT #3 RHCS3 WAS SET
8039                                     ;ZEROS WERE MOVED INTO
8040                                     ;RHDB
8041                                     ;RHDB WAS READ THEN
8042                                     ;RHCS3 SHOULD HAVE 10
8043                                     ;=10
8044                                     ;BUT CONTAINED WHAT IS
8045 026400 67$:                                     ;GIVEN IN BAD RHCS3
8046
8047                                     ;CHECK THAT RHCS2 HAS MDPE!OR!IR
8048 026400 012737 000700 001124  MOV    #MDPE!OR!IR,$GDDAT ;GET GOOD = 700
8049 026406 053737 006442 001124  BIS    UNIT,$GDDAT    ;INCLUDE UNIT NUMBER
8050
8051 026414 017737 155450 001126  MOV    @RHCS2,$BDDAT ;READ RHCS2 FOR COMPARISON
8052 026422 023737 001124 001126  CMP    $GDDAT,$BDDAT ;COMPARE EXPECTED
8053                                     ;DATA WITH DATA READ FROM
8054                                     ;RHCS2
8055 026430 001401  BEQ    69$    ;BRANCH IF GOOD
8056 026432 104105  ERROR   105
8057
8058                                     ;AFTER SETTING "CLR" BIT #5
8059                                     ;IN RHCS2 TO INIT THE RH
8060                                     ;IPCK3 BIT #3 RHCS3 WAS SET
8061                                     ;ZEROS WERE MOVED INTO
8062                                     ;RHDB
8063                                     ;RHDB WAS READ THEN
8064                                     ;RHCS2 SHOULD HAVE MDPE!OR!IR
8065                                     ;=700
8066                                     ;TOGETHER WITH UNIT NUMBER
8067                                     ;BUT CONTAINED WHAT IS
8068 026434 69$:                                     ;GIVEN IN BAD RHCS2
8069
8070 026434 017737 155442 001202  MOV    @RHDB,$TMP2    ;READ RHDB SECOND TIME
8071                                     ;CHECK THAT RHCS1 HAS SC!TRE!DVA!RDY
8072 026442 012737 144200 001124  MOV    #SC!TRE!DVA!RDY,$GDDAT ;GET GOOD = 144200

```


8241 ; IN RHCS2 TO INIT THE RH
8242 ; IPCK3 BIT #3 RHCS3 WAS SET
8243 ; ZEROS WERE MOVED INTO
8244 ; RHDB
8245 ; RHDB WAS READ THEN
8246 ; AN RH CLEAR (RHCS2 BIT #5)
8247 ; WAS GIVEN THIS SHOULD
8248 ; CLEAR ALL ERRORS
8249 ; RHBAE SHOULD HAVE 0
8250 ; BUT CONTAINED WHAT IS
8251 ; GIVEN IN BAD RHBAE

8252 026742 85\$:
8253 ; CHECK THAT RHWC HAS 0
8254 026742 012737 000000 001124 MOV #0,\$GDDAT ; GET GOOD = 0
8255
8256 026750 017737 155106 001126 MOV @RHWC,\$BDDAT ; READ RHWC FOR COMPARISON
8257 026756 023737 001124 001126 CMP \$GDDAT,\$BDDAT ; COMPARE EXPECTED
8258 ; DATA WITH DATA READ FROM
8259 ; RHWC
8260 026764 001401 BEQ 87\$; BRANCH IF GOOD
8261 026766 104113 ERROR 113

8262 ; AFTER SETTING 'CLR' BIT #5
8263 ; IN RHCS2 TO INIT THE RH
8264 ; IPCK3 BIT #3 RHCS3 WAS SET
8265 ; ZEROS WERE MOVED INTO
8266 ; RHDB
8267 ; RHDB WAS READ THEN
8268 ; AN RH CLEAR (RHCS2 BIT #5)
8269 ; WAS GIVEN THIS SHOULD
8270 ; CLEAR ALL ERRORS
8271 ; RHWC SHOULD HAVE 0
8272 ; BUT CONTAINED WHAT IS
8273 ; GIVEN IN BAD RHWC

8274 026770 87\$:
8275
8276
8277
8278

8279
8280 *TEST 34 RHCS2-WCE BIT #14 AND RHCS3-WCE OW BIT #12
8281
8282 * CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
8283 * WRITE NINE WORD OF ALL NINES ON TO THE DEVICE
8284 * SET UP TO DO WRITE CHECK ON THAT WORD FROM AN
8285 * ODD WORD BOUNDARY
8286 * DO A ONE WORD WRITE CHECK WITH RHBA FROM AN
8287 * ODD WORD BOUNDARY
8288 * THIS SHOULD SET WCE OW (RHCS3 BIT #12)
8289 * AND WCE (RHCS2 BIT #14)
8290 * 'CLR' (RHCS2 BIT #5) IS GIVEN
8291 * ALL ERRORS ARE CLEARED
8292 * CHECK RHCS1, RHCS2, RHCS3, RHBA, RHBAE, RHWC

8293
8294
8295 026770 000004 TST34: SCOPE
8296 026772 012706 001000 MOV #STACK,SP ; RESET STACK


```

8297 026776 012737 000034 006276      MOV    #34,@#TSTNM      ;SAVE TEST NUMBER
8298
8299 027004 004737 041222      JSR    PC,@#CLDISK      ;GIVE RH INITIALIZE
8300                                     ;SETUP UNIT NUBER
8301                                     ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
8302
8303                                     ;WRITE ONE WORD OF ALL ONES ON THE DEVICE
8304 027010 012737 177777 004210      MOV    #-1,WRFROM       ;ALL ONES INTO WRITE FROM
8305 027016 012777 177767 155036      MOV    #-9,@RHWC        ;NINE WORD FOR WORD COUNT REGISTER
8306 027024 012777 004210 155032      MOV    #WRFROM,@RHBA    ;WRITE FROM BUFFER INTO BUS ADDRESS
8307
8308 027032 004077 157222      JSR    RO,@COMND        ;GO TO DO COMMAND
8309 027036 004154      WRIDAT      ;WRITE DATA
8310
8311                                     ;SET UP FOR WRITE CHECK
8312 027040 005037 003002      CLR    BUFW            ;WRITE DATA FOR WRITE CHECK ERROR
8313 027044 012777 177767 155010      MOV    #-9,@RHWC        ;NINE WORD FOR WORD COUNT REGISTER
8314 027052 012777 003002 155004      MOV    #BUFW,@RHBA     ;WRITE FROM BUFFER INTO BUS ADDRESS
8315                                     ;FROM ODD WORD BOUNDARY
8316                                     ;WRITE CHECK ON ODD WORD BOUNDARY
8317 027060 004077 157174      JSR    RO,@COMND        ;GO TO DO COMMAND
8318 027064 004150      WRCHK      ;WRITE CHECK DATA
8319
8320
8321
8322
8323                                     ;CHECK THAT RHCS2 HAS WCE!OR!IR
8324 027066 012737 040300 001124      MOV    #WCE!OR!IR,$GDDAT ;GET GOOD = 40300
8325 027074 053737 006442 001124      BIS    UNIT,$GDDAT      ;INCLUDE UNIT NUMBER
8326
8327 027102 017737 154762 001126      MOV    @RHCS2,$BDDAT    ;READ RHCS2 FOR COMPARISON
8328 027110 023737 001124 001126      CMP    $GDDAT,$BDDAT    ;COMPARE EXPECTED
8329                                     ;DATA WITH DATA READ FROM
8330                                     ;RHCS2
8331 027116 001401      BEQ    65$             ;BRANCH IF GOOD
8332 027120 104114      ERROR  114
8333                                     ;AFTER SETTING "CLR" BIT #5
8334                                     ;IN RHCS2 TO INIT THE RH
8335                                     ;ONE WORD OF ALL ONES IS
8336                                     ;WRITTEN ON THE DEVICE
8337                                     ;A WRITE CHECK WAS DONE
8338                                     ;ON AN ODD WORD BOUNDARY THEN
8339                                     ;RHCS2 SHOULD HAVE WCE!OR!IR
8340                                     ;=40300
8341                                     ;TOGETHER WITH UNIT NUMBER
8342                                     ;BUT CONTAINED WHAT IS
8343                                     ;GIVEN IN BAD RHCS2
8344 027122      65$:
8345
8346                                     ;CHECK THAT RHCS3 HAS DBL!WCEOW
8347 027122 012737 012000 001124      MOV    #DBL!WCEOW,$GDDAT ;GET GOOD = 12000
8348
8349 027130 017737 154776 001126      MOV    @RHCS3,$BDDAT    ;READ RHCS3 FOR COMPARISON
8350 027136 023737 001124 001126      CMP    $GDDAT,$BDDAT    ;COMPARE EXPECTED
8351                                     ;DATA WITH DATA READ FROM
8352                                     ;RHCS3

```



```

8409                                     ;WRITTEN ON THE DEVICE
8410                                     ;A WRITE CHECK WAS DONE
8411                                     ;ON AN ODD WORD BOUNDARY THEN
8412                                     ;AN RH CLEAR (RHCS2 BIT #5)
8413                                     ;WAS GIVEN THIS SHOULD
8414                                     ;CLEAR ALL ERRORS
8415                                     ;RHCS2 SHOULD HAVE IR
8416                                     ;=100
8417                                     ;TOGETHER WITH UNIT NUMBER
8418                                     ;BUT CONTAINED WHAT IS
8419                                     ;GIVEN IN BAD RHCS2
8420 027236                               71$:
8421                                     ;CHECK THAT RHCS3 HAS 0
8422 027236 012737 000000 001124      MOV    #0,$GDDAT      ;GET GOOD = 0
8423
8424 027244 017737 154662 001126      MOV    @RHCS3,$BDDAT ;READ RHCS3 FOR COMPARISON
8425 027252 023737 001124 001126      CMP    $GDDAT,$BDDAT ;COMPARE EXPECTED
8426                                     ;DATA WITH DATA READ FROM
8427                                     ;RHCS3
8428 027260 001401                       BEQ    73$           ;BRANCH IF GOOD
8429 027262 104120                       ERROR  120
8430
8431                                     ;AFTER SETTING "CLR" BIT #5
8432                                     ;IN RHCS2 TO INIT THE RH
8433                                     ;ONE WORD OF ALL ONES IS
8434                                     ;WRITTEN ON THE DEVICE
8435                                     ;A WRITE CHECK WAS DONE
8436                                     ;ON AN ODD WORD BOUNDARY THEN
8437                                     ;AN RH CLEAR (RHCS2 BIT #5)
8438                                     ;WAS GIVEN THIS SHOULD
8439                                     ;CLEAR ALL ERRORS
8440                                     ;RHCS3 SHOULD HAVE 0
8441                                     ;BUT CONTAINED WHAT IS
8442                                     ;GIVEN IN BAD RHCS3
8442 027264                               73$:
8443                                     ;CHECK THAT RHBA HAS 0
8444 027264 012737 000000 001124      MOV    #0,$GDDAT      ;GET GOOD = 0
8445
8446 027272 017737 154566 001126      MOV    @RHBA,$BDDAT  ;READ RHBA FOR COMPARISON
8447 027300 023737 001124 001126      CMP    $GDDAT,$BDDAT ;COMPARE EXPECTED
8448                                     ;DATA WITH DATA READ FROM
8449                                     ;RHBA
8450 027306 001401                       BEQ    75$           ;BRANCH IF GOOD
8451 027310 104121                       ERROR  121
8452
8453                                     ;AFTER SETTING "CLR" BIT #5
8454                                     ;IN RHCS2 TO INIT THE RH
8455                                     ;ONE WORD OF ALL ONES IS
8456                                     ;WRITTEN ON THE DEVICE
8457                                     ;A WRITE CHECK WAS DONE
8458                                     ;ON AN ODD WORD BOUNDARY THEN
8459                                     ;AN RH CLEAR (RHCS2 BIT #5)
8460                                     ;WAS GIVEN THIS SHOULD
8461                                     ;CLEAR ALL ERRORS
8462                                     ;RHBA SHOULD HAVE 0
8463                                     ;BUT CONTAINED WHAT IS
8464 027312                               75$:
    
```

```
8465 ;CHECK THAT RHBAE HAS 0
8466 027312 012737 000000 001124 MOV #0,$GDDAT ;GET GOOD = 0
8467
8468 027320 017737 154604 001126 MOV @RHBAE,$BDDAT ;READ RHBAE FOR COMPARISON
8469 027326 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
8470 ;DATA WITH DATA READ FROM
8471 ;RHBAE
8472 027334 001401 BEQ 77$ ;BRANCH IF GOOD
8473 027336 104122 ERROR 122
8474 ;AFTER SETTING 'CLR' BIT #5
8475 ;IN RHCS2 TO INIT THE RH
8476 ;ONE WORD OF ALL ONES IS
8477 ;WRITTEN ON THE DEVICE
8478 ;A WRITE CHECK WAS DONE
8479 ;ON AN ODD WORD BOUNDARY THEN
8480 ;AN RH CLEAR (RHCS2 BIT #5)
8481 ;WAS GIVEN THIS SHOULD
8482 ;CLEAR ALL ERRORS
8483 ;RHBAE SHOULD HAVE 0
8484 ;BUT CONTAINED WHAT IS
8485 ;GIVEN IN BAD RHBAE
8486 027340 77$:
8487 ;CHECK THAT RHWC HAS 0
8488 027340 012737 000000 001124 MOV #0,$GDDAT ;GET GOOD = 0
8489
8490 027346 017737 154510 001126 MOV @RHWC,$BDDAT ;READ RHWC FOR COMPARISON
8491 027354 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
8492 ;DATA WITH DATA READ FROM
8493 ;RHWC
8494 027362 001401 BEQ 79$ ;BRANCH IF GOOD
8495 027364 104123 ERROR 123
8496 ;AFTER SETTING 'CLR' BIT #5
8497 ;IN RHCS2 TO INIT THE RH
8498 ;ONE WORD OF ALL ONES IS
8499 ;WRITTEN ON THE DEVICE
8500 ;A WRITE CHECK WAS DONE
8501 ;ON AN ODD WORD BOUNDARY THEN
8502 ;AN RH CLEAR (RHCS2 BIT #5)
8503 ;WAS GIVEN THIS SHOULD
8504 ;CLEAR ALL ERRORS
8505 ;RHWC SHOULD HAVE 0
8506 ;BUT CONTAINED WHAT IS
8507 ;GIVEN IN BAD RHWC
8508 027366 79$:
8509
8510
8511
8512 ;*****
8513 ;*TEST 35 RHCS2-WCE BIT #14 AND RHCS3-WCE OW BIT #12
8514
8515 ;* CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
8516 ;* WRITE NINE WORD OF ALL NINES ON 10 THE DEVICE
8517 ;* SET UP TO DO WRITE CHECK ON THAT WORD FROM AN
8518 ;* ODD WORD BOUNDARY
8519 ;* DO A ONE WORD WRITE CHECK WITH RHBA FROM AN
8520 ;* ODD WORD BOUNDARY
```



```

8521      : * THIS SHOULD SET WCE OW (RHCS3 BIT #12)
8522      : * AND WCE (RHCS2 BIT #14)
8523      : * "CLR" (RHCS2 BIT #5) IS GIVEN
8524      : * ALL ERRORS ARE CLEARED
8525      : * CHECK RHCS1, RHCS2, RHCS3, RHBA, RHBAE, RHWC
8526
8527      : *****
8528 027366 000004      TST35: SCOPE
8529 027370 012706 001000      MOV #STACK,SP ;RESET STACK
8530 027374 012737 000035 006276      MOV #35,@#TSTNM ;SAVE TEST NUMBER
8531
8532 027402 004737 041222      JSR PC,@#CLDISK ;GIVE RH INITIALIZE
8533 ;SETUP UNIT NUBER
8534 ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
8535
8536 ;WRITE ONE WORD OF ALL ONES ON THE DEVICE
8537 027406 012737 177777 004210      MOV #-1,WRFROM ;ALL ONES INTO WRITE FROM
8538 027414 012777 177767 154440      MOV #-9,@RHWC ;NINE WORD FOR WORD COUNT REGISTER
8539 027422 012777 004210 154434      MOV #WRFROM,@RHBA ;WRITE FROM BUFFER INTO BUS ADDRESS
8540
8541 027430 004077 156624      JSR RO,@COMND ;GO TO DO COMMAND
8542 027434 004154      WRIDAT ;WRITE DATA
8543
8544 ;SET UP FOR WRITE CHECK
8545 027436 005037 003002      CLR BUFOW ;WRITE DATA FOR WRITE CHECK ERROR
8546 027442 012777 177767 154412      MOV #-9,@RHWC ;NINE WORD FOR WORD COUNT REGISTER
8547 027450 012777 003002 154406      MOV #BUFOW,@RHBA ;WRITE FROM BUFFER INTO BUS ADDRESS
8548 ;FROM ODD WORD BOUNDARY
8549 ;WRITE CHECK ON ODD WORD BOUNDARY
8550 027456 004077 156576      JSR RO,@COMND ;GO TO DO COMMAND
8551 027462 004150      WRCHK ;WRITE CHECK DATA
8552
8553
8554
8555
8556 ;CHECK THAT RHCS2 HAS WCE!OR!IR
8557 027464 012737 040300 001124      MOV #WCE!OR!IR,$GDDAT ;GET GOOD = 40300
8558 027472 053737 006442 001124      BIS UNIT,$GDDAT ;INCLUDE UNIT NUMBER
8559
8560 027500 017737 154364 001126      MOV @RHCS2,$BDDAT ;READ RHCS2 FOR COMPARISON
8561 027506 023737 001124 001126      CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
8562 ;DATA WITH DATA READ FROM
8563 ;RHCS2
8564 027514 001401      BEQ 65$ ;BRANCH IF GOOD
8565 027516 104114      ERROR 114
8566 ;AFTER SETTING "CLR" BIT #5
8567 ;IN RHCS2 TO INIT THE RH
8568 ;ONE WORD OF ALL ONES IS
8569 ;WRITTEN ON THE DEVICE
8570 ;A WRITE CHECK WAS DONE
8571 ;ON AN ODD WORD BOUNDARY THEN
8572 ;RHCS2 SHOULD HAVE WCE!OR!IR
8573 ;=40300
8574 ;TOGETHER WITH UNIT NUMBER
8575 ;BUT CONTAINED WHAT IS
8576 ;GIVEN IN BAD RHCS2

```

```

8577 027520          65$:
8578
8579                ;CHECK THAT RHCS3 HAS DBL!WCEOW
8580 027520 012737 012000 001124    MOV      #DBL!WCEOW,$GDDAT      ;GET GOOD = 12000
8581
8582 027526 017737 154400 001126    MOV      @RHCS3,$BDDAT        ;READ RHCS3 FOR COMPARISON
8583 027534 023737 001124 001126    CMP      $GDDAT,$BDDAT        ;COMPARE EXPECTED
8584                                     ;DATA WITH DATA READ FROM
8585                                     ;RHCS3
8586 027542 001401          BEQ      67$                    ;BRANCH IF GOOD
8587 027544 104115          ERROR    115
8588                                     ;AFTER SETTING "CLR" BIT #5
8589                                     ;IN RHCS2 TO INIT THE RH
8590                                     ;ONE WORD OF ALL ONES IS
8591                                     ;WRITTEN ON THE DEVICE
8592                                     ;A WRITE CHECK WAS DONE
8593                                     ;ON AN ODD WORD BOUNDARY THEN
8594                                     ;RHCS3 SHOULD HAVE DBL!WCEOW
8595                                     ;=12000
8596                                     ;BUT CONTAINED WHAT IS
8597                                     ;GIVEN IN BAD RHCS3
8598 027546          67$:
8599
8600
8601 027546 004737 041222          JSR      PC,@#CLDISK          ;GIVE RH INITIALIZE
8602                                     ;SETUP UNIT NUMBER
8603                                     ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
8604
8605
8606                ;CHECK THAT RHCS1 HAS RDY!DVA
8607 027552 012737 004200 001124    MOV      #RDY!DVA,$GDDAT      ;GET GOOD = 4200
8608
8609 027560 017737 154274 001126    MOV      @RHCS1,$BDDAT        ;READ RHCS1 FOR COMPARISON
8610 027566 023737 001124 001126    CMP      $GDDAT,$BDDAT        ;COMPARE EXPECTED
8611                                     ;DATA WITH DATA READ FROM
8612                                     ;RHCS1
8613 027574 001401          BEQ      69$                    ;BRANCH IF GOOD
8614 027576 104116          ERROR    116
8615                                     ;AFTER SETTING "CLR" BIT #5
8616                                     ;IN RHCS2 TO INIT THE RH
8617                                     ;ONE WORD OF ALL ONES IS
8618                                     ;WRITTEN ON THE DEVICE
8619                                     ;A WRITE CHECK WAS DONE
8620                                     ;ON AN ODD WORD BOUNDARY THEN
8621                                     ;AN RH CLEAR (RHCS2 BIT #5)
8622                                     ;WAS GIVEN THIS SHOULD
8623                                     ;CLEAR ALL ERRORS
8624                                     ;RHCS1 SHOULD HAVE RDY!DVA
8625                                     ;=4200
8626                                     ;BUT CONTAINED WHAT IS
8627                                     ;GIVEN IN BAD RHCS1
8628 027600          69$:
8629
8630 027600 012737 000100 001124    MOV      #IR,$GDDAT           ;GET GOOD = 100
8631 027606 053737 006442 001124    BIS      UNIT,$GDDAT          ;INCLUDE UNIT NUMBER
8632
    
```


8689
8690
8691
8692
8693
8694
8695
8696
8697
8698
8699
8700
8701
8702
8703
8704
8705
8706
8707
8708
8709
8710
8711
8712
8713
8714
8715
8716
8717
8718
8719
8720
8721
8722
8723
8724
8725
8726
8727
8728
8729
8730
8731
8732
8733
8734
8735
8736
8737
8738
8739
8740
8741
8742
8743
8744

027710

75\$:

027710 012737 000000 001124

027716 017737 154206 001126

027724 023737 001124 001126

027732 001401

027734 104122

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

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

BEQ 77\$
ERROR 122

:A WRITE CHECK WAS DONE
:ON AN ODD 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

: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 ODD 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

027736

77\$:

027736 012737 000000 001124

027744 017737 154112 001126

027752 023737 001124 001126

027760 001401

027762 104123

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

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

BEQ 79\$
ERROR 123

: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
: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

027764

79\$:

8745
8746
8747
8748
8749
8750
8751
8752
8753
8754
8755
8756
8757
8758
8759
8760
8761
8762
8763
8764
8765
8766
8767
8768
8769
8770
8771
8772
8773
8774
8775
8776
8777
8778
8779
8780
8781
8782
8783
8784
8785
8786
8787
8788
8789
8790
8791
8792
8793
8794
8795
8796
8797
8798
8799
8800

```
*****
: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
*****
```

```
*****
TST36:  SCOPE
MOV     #STACK,SP      ;RESET STACK
MOV     #36,@#TSTNM    ;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     R0,@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 BOUNDRY
:WRITE CHECK ON EVEN WORD BOUNDARY
JSR     R0,@COMND      ;GO TO DO COMMAND
WRCKEK                                ;WRITE CHECK DATA
:CHECK THAT RHCS2 HAS WCE!OR!IR
MOV     #WCE!OR!IR,$GDDAT ;GET GOOD = 40300
BIS     UNIT,$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   114
                        ;AFTER SETTING "CLR" BIT #5
                        ;IN RHCS2 TO INIT THE RH
*****
```



```

8857                                     ;RHCS1 SHOULD HAVE RDY!DVA
8858                                     ;=4200
8859                                     ;BUT CONTAINED WHAT IS
8860                                     ;GIVEN IN BAD RHCS1
8861 030176                               69$:
8862                                     ;CHECK THAT RHCS2 HAS IR
8863 030176 012737 000100 001124      MOV #IR,$GDDAT ;GET GOOD = 100
8864 030204 053737 006442 001124      BIS UNIT,$GDDAT ;INCLUDE UNIT NUMBER
8865
8866 030212 017737 153652 001126      MOV @RHCS2,$BDDAT ;READ RHCS2 FOR COMPARISON
8867 030220 023737 001124 001126      CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
8868                                     ;DATA WITH DATA READ FROM
8869                                     ;RHCS2
8870 030226 001401                       BEQ 71$ ;BRANCH IF GOOD
8871 030230 104117                       ERROR 117
8872
8873                                     ;AFTER SETTING 'CLR' BIT #5
8874                                     ;IN RHCS2 TO INIT THE RH
8875                                     ;ONE WORD OF ALL ONES IS
8876                                     ;WRITTEN ON THE DEVICE
8877                                     ;A WRITE CHECK WAS DONE
8878                                     ;ON AN EVEN WORD BOUNDARY THEN
8879                                     ;AN RH CLEAR (RHCS2 BIT #5)
8880                                     ;WAS GIVEN THIS SHOULD
8881                                     ;CLEAR ALL ERRORS
8882                                     ;RHCS2 SHOULD HAVE IR
8883                                     ;=100
8884                                     ;TOGETHER WITH UNIT NUMBER
8885                                     ;BUT CONTAINED WHAT IS
8886                                     ;GIVEN IN BAD RHCS2
8886 030232                               71$:
8887                                     ;CHECK THAT RHCS3 HAS 0
8888 030232 012737 000000 001124      MOV #0,$GDDAT ;GET GOOD = 0
8889
8890 030240 017737 153666 001126      MOV @RHCS3,$BDDAT ;READ RHCS3 FOR COMPARISON
8891 030246 023737 001124 001126      CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
8892                                     ;DATA WITH DATA READ FROM
8893                                     ;RHCS3
8894 030254 001401                       BEQ 73$ ;BRANCH IF GOOD
8895 030256 104120                       ERROR 120
8896
8897                                     ;AFTER SETTING 'CLR' BIT #5
8898                                     ;IN RHCS2 TO INIT THE RH
8899                                     ;ONE WORD OF ALL ONES IS
8900                                     ;WRITTEN ON THE DEVICE
8901                                     ;A WRITE CHECK WAS DONE
8902                                     ;ON AN EVEN WORD BOUNDARY THEN
8903                                     ;AN RH CLEAR (RHCS2 BIT #5)
8904                                     ;WAS GIVEN THIS SHOULD
8905                                     ;CLEAR ALL ERRORS
8906                                     ;RHCS3 SHOULD HAVE 0
8907                                     ;BUT CONTAINED WHAT IS
8908                                     ;GIVEN IN BAD RHCS3
8908 030260                               73$:
8909                                     ;CHECK THAT RHBA HAS 0
8910 030260 012737 000000 001124      MOV #0,$GDDAT ;GET GOOD = 0
8911
8912 030266 017737 153572 001126      MOV @RHBA,$BDDAT ;READ RHBA FOR COMPARISON
    
```


8969
8970
8971
8972
8973
8974
8975
8976
8977
8978
8979
8980
8981
8982
8983
8984
8985
8986
8987
8988
8989
8990
8991
8992
8993
8994
8995
8996
8997
8998
8999
9000
9001
9002
9003
9004
9005
9006
9007
9008
9009
9010
9011
9012
9013
9014
9015
9016
9017
9018
9019
9020
9021
9022
9023
9024

030362

798:

:WAS GIVEN THIS SHOULD
 :CLEAR ALL ERRORS
 :RHWC SHOULD HAVE 0
 :BUT CONTAINED WHAT IS
 :GIVEN IN BAD RHWC

.....
 :TEST 37 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

.....
 :TEST37: SCOPE
 MOV #STACK,SP :RESET STACK
 MOV #37,@TSTNM :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 R0,@COMND :GO TO DO COMMAND
 WRIDAT :WRITE DATA

:SET UP FOR WRITE CHECK
 CLR B'JFEW :WRITE DATA FOR WRITE CHECK ERROR
 MOV #-9,@RHWC :NINE WORD FOR WORD COUNT REGISTER
 MOV #BUF EW,@RHBA :WRITE FROM BUFFER INTO BUS ADDRESS
 :FROM EVEN WORD BOUNDARY

:WRITE CHECK ON EVEN WORD BOUNDARY
 JSR R0,@COMND :GO TO DO COMMAND
 WRCHK :WRITE CHECK DATA

:CHECK THAT RHCS2 HAS WCE!OR!IR
 MOV #WCE!OR!IR,\$GDDAT :GET GOOD = 40300
 BIS UNIT,\$GDDAT :INCLUDE UNIT NUMBER

030362 000004
 030364 012706 001000
 030370 012737 000037 006276
 030376 004737 041222
 030402 012737 177777 004210
 030410 012777 177767 153444
 030416 012777 004210 153440
 030424 004077 155630
 030430 004154
 030432 005037 003000
 030436 012777 177767 153416
 030444 012777 003000 153412
 030452 004077 155602
 030456 004150
 030460 012737 040300 001124
 030466 053737 006442 001124

```

9025
9026 030474 017737 153370 001126      MOV    @RHCS2,$BDDAT    ;READ RHCS2 FOR COMPARISON
9027 030502 023737 001124 001126      CMP    $GDDAT,$BDDAT    ;COMPARE EXPECTED
9028                                     ;DATA WITH DATA READ FROM
9029                                     ;RHCS2
9030 030510 001401      BEQ    65$              ;BRANCH IF GOOD
9031 030512 104114      ERROR  114
9032                                     ;AFTER SETTING "CLR" BIT #5
9033                                     ;IN RHCS2 TO INIT THE RH
9034                                     ;ONE WORD OF ALL ONES IS
9035                                     ;WRITTEN ON THE DEVICE
9036                                     ;A WRITE CHECK WAS DONE
9037                                     ;ON AN EVEN WORD BOUNDARY THEN
9038                                     ;RHCS2 SHOULD HAVE WCE!OR!IR
9039                                     ;=40300
9040                                     ;TOGETHER WITH UNIT NUMBER
9041                                     ;BUT CONTAINED WHAT IS
9042                                     ;GIVEN IN BAD RHCS2
9043 030514      65$:
9044
9045                                     ;CHECK THAT RHCS3 HAS DBL!WCEEW
9046 030514 012737 006000 001124      MOV    #DBL!WCEEW,$GDDAT ;GET GOOD = 6000
9047
9048 030522 017737 153404 001126      MOV    @RHCS3,$BDDAT    ;READ RHCS3 FOR COMPARISON
9049 030530 023737 001124 001126      CMP    $GDDAT,$BDDAT    ;COMPARE EXPECTED
9050                                     ;DATA WITH DATA READ FROM
9051                                     ;RHCS3
9052 030536 001401      BEQ    67$              ;BRANCH IF GOOD
9053 030540 104115      ERROR  115
9054                                     ;AFTER SETTING "CLR" BIT #5
9055                                     ;IN RHCS2 TO INIT THE RH
9056                                     ;ONE WORD OF ALL ONES IS
9057                                     ;WRITTEN ON THE DEVICE
9058                                     ;A WRITE CHECK WAS DONE
9059                                     ;ON AN EVEN WORD BOUNDARY THEN
9060                                     ;RHCS3 SHOULD HAVE DBL!WCEEW
9061                                     ;=6000
9062                                     ;BUT CONTAINED WHAT IS
9063                                     ;GIVEN IN BAD RHCS3
9064 030542      67$:
9065
9066
9067 030542 004737 041222      JSR    PC,@#CLDISK      ;GIVE RH INITIALIZE
9068                                     ;SETUP UNIT NUBER
9069                                     ;CLEAR RHIC AND FUNCTION BITS IN RHCS1
9070
9071
9072                                     ;CHECK THAT RHCS1 HAS RDY!DVA
9073 030546 012737 004200 001124      MOV    #RDY!DVA,$GDDAT ;GET GOOD = 4200
9074
9075 030554 017737 153300 001126      MOV    @RHCS1,$BDDAT    ;READ RHCS1 FOR COMPARISON
9076 030562 023737 001124 001126      CMP    $GDDAT,$BDDAT    ;COMPARE EXPECTED
9077                                     ;DATA WITH DATA READ FROM
9078                                     ;RHCS1
9079 030570 001401      BEQ    69$              ;BRANCH IF GOOD
9080 030572 104116      ERROR  116

```



```

9081 ;AFTER SETTING 'CLR' BIT #5
9082 ;IN RHCS2 TO INIT THE RH
9083 ;ONE WORD OF ALL ONES IS
9084 ;WRITTEN ON THE DEVICE
9085 ;A WRITE CHECK WAS DONE
9086 ;ON AN EVEN WORD BOUNDARY THEN
9087 ;AN RH CLEAR (RHCS2 BIT #5)
9088 ;WAS GIVEN THIS SHOULD
9089 ;CLEAR ALL ERRORS
9090 ;RHCS1 SHOULD HAVE RDY:DVA
9091 ;=4200
9092 ;BUT CONTAINED WHAT IS
9093 ;GIVEN IN BAD RHCS1
9094 030574 69$:
9095 ;CHECK THAT RHCS2 HAS IR
9096 030574 012737 000100 001124 MOV #IR,$GDDAT ;GET GOOD = 100
9097 030602 053737 006442 001124 BIS UNIT,$GDDAT ;INCLUDE UNIT NUMBER
9098
9099 030610 017737 153254 001126 MOV @RHCS2,$BDDAT ;READ RHCS2 FOR COMPARISON
9100 030616 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
9101 ;DATA WITH DATA READ FROM
9102 ;RHCS2
9103 030624 001401 BEQ 71$ ;BRANCH IF GOOD
9104 030626 104117 ERROR 117
9105 ;AFTER SETTING 'CLR' BIT #5
9106 ;IN RHCS2 TO INIT THE RH
9107 ;ONE WORD OF ALL ONES IS
9108 ;WRITTEN ON THE DEVICE
9109 ;A WRITE CHECK WAS DONE
9110 ;ON AN EVEN WORD BOUNDARY THEN
9111 ;AN RH CLEAR (RHCS2 BIT #5)
9112 ;WAS GIVEN THIS SHOULD
9113 ;CLEAR ALL ERRORS
9114 ;RHCS2 SHOULD HAVE IR
9115 ;=100
9116 ;TOGETHER WITH UNIT NUMBER
9117 ;BUT CONTAINED WHAT IS
9118 ;GIVEN IN BAD RHCS2
9119 030630 71$:
9120 ;CHECK THAT RHCS3 HAS 0
9121 030630 012737 000000 001124 MOV #0,$GDDAT ;GET GOOD = 0
9122
9123 030636 017737 153270 001126 MOV @RHCS3,$BDDAT ;READ RHCS3 FOR COMPARISON
9124 030644 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
9125 ;DATA WITH DATA READ FROM
9126 ;RHCS3
9127 030652 001401 BEQ 73$ ;BRANCH IF GOOD
9128 030654 104120 ERROR 120
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 EVEN WORD BOUNDARY THEN
9135 ;AN RH CLEAR (RHCS2 BIT #5)
9136 ;WAS GIVEN THIS SHOULD

```

```

9137                                     ;CLEAR ALL ERRORS
9138                                     ;RHCS3 SHOULD HAVE 0
9139                                     ;BUT CONTAINED WHAT IS
9140                                     ;GIVEN IN BAD RHCS3
9141 030656                               73$:
9142                                     ;CHECK THAT RHBA HAS 0
9143 030656 012737 000000 001124      MOV    #0,$GDDAT      ;GET GOOD = 0
9144
9145 030664 017737 153174 001126      MOV    @RHBA,$BDDAT  ;READ RHBA FOR COMPARISON
9146 030672 023737 001124 001126      CMP    $GDDAT,$BDDAT ;COMPARE EXPECTED
9147                                     ;DATA WITH DATA READ FROM
9148                                     ;RHBA
9149 030700 001401                        BEQ    75$            ;BRANCH IF GOOD
9150 030702 104121
9151                                     ;AFTER SETTING "CLR" BIT #5
9152                                     ;IN RHCS2 TO INIT THE RH
9153                                     ;ONE WORD OF ALL ONES IS
9154                                     ;WRITTEN ON THE DEVICE
9155                                     ;A WRITE CHECK WAS DONE
9156                                     ;ON AN EVEN WORD BOUNDARY THEN
9157                                     ;AN RH CLEAR (RHCS2 BIT #5)
9158                                     ;WAS GIVEN THIS SHOULD
9159                                     ;CLEAR ALL ERRORS
9160                                     ;RHBA SHOULD HAVE 0
9161                                     ;BUT CONTAINED WHAT IS
9162                                     ;GIVEN IN BAD RHBA
9163 030704                               75$:
9164                                     ;CHECK THAT RHBAE HAS 0
9165 030704 012737 000000 001124      MOV    #0,$GDDAT      ;GET GOOD = 0
9166
9167 030712 017737 153212 001126      MOV    @RHBAE,$BDDAT ;READ RHBAE FOR COMPARISON
9168 030720 023737 001124 001126      CMP    $GDDAT,$BDDAT ;COMPARE EXPECTED
9169                                     ;DATA WITH DATA READ FROM
9170                                     ;RHBAE
9171 030726 001401                        BEQ    77$            ;BRANCH IF GOOD
9172 030730 104122
9173                                     ;AFTER SETTING "CLR" BIT #5
9174                                     ;IN RHCS2 TO INIT THE RH
9175                                     ;ONE WORD OF ALL ONES IS
9176                                     ;WRITTEN ON THE DEVICE
9177                                     ;A WRITE CHECK WAS DONE
9178                                     ;ON AN EVEN WORD BOUNDARY THEN
9179                                     ;AN RH CLEAR (RHCS2 BIT #5)
9180                                     ;WAS GIVEN THIS SHOULD
9181                                     ;CLEAR ALL ERRORS
9182                                     ;RHBAE SHOULD HAVE 0
9183                                     ;BUT CONTAINED WHAT IS
9184                                     ;GIVEN IN BAD RHBAE
9185 030732                               77$:
9186                                     ;CHECK THAT RHWC HAS 0
9187 030732 012737 000000 001124      MOV    #0,$GDDAT      ;GET GOOD = 0
9188
9189 030740 017737 153116 001126      MOV    @RHWC,$BDDAT  ;READ RHWC FOR COMPARISON
9190 030746 023737 001124 001126      CMP    $GDDAT,$BDDAT ;COMPARE EXPECTED
9191                                     ;DATA WITH DATA READ FROM
9192                                     ;RHWC

```



```

9193 030754 001401      BEQ      79$      ;BRANCH IF GOOD
9194 030756 104123      ERROR    123
9195                                     ;AFTER SETTING 'CLR' BIT #5
9196                                     ;IN RHCS2 TO INIT THE RH
9197                                     ;ONE WORD OF ALL ONES IS
9198                                     ;WRITTEN ON THE DEVICE
9199                                     ;A WRITE CHECK WAS DONE
9200                                     ;ON AN EVEN WORD BOUNDARY THEN
9201                                     ;AN RH CLEAR (RHCS2 BIT #5)
9202                                     ;WAS GIVEN THIS SHOULD
9203                                     ;CLEAR ALL ERRORS
9204                                     ;RHCW SHOULD HAVE 0
9205                                     ;BUT CONTAINED WHAT IS
9206                                     ;GIVEN IN BAD RHCW

```

```

9207 030760      79$:
9208
9209
9210
9211

```

```

*****
: *TEST 40      TEST DBL (RHCS3 BIT #10) TEST A

```

```

: *      CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
: *      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,RHCW

```

```

*****
TST40: SCOPE

```

```

9223 030760 000004      TST40: SCOPE
9224 030762 012706 001000      MOV      #STACK,SP      ;RESET STACK
9225 030766 012737 000040 006276      MOV      #40,@#TSTNM    ;SAVE TEST NUMBER
9226
9227 030774 004737 041222      JSR      PC,@#CLDISK    ;GIVE RH INITIALIZE
9228                                     ;SETUP UNIT NUBER
9229                                     ;CLEAR RHCW AND FUNCTION BITS IN RHCST
9230

```

```

;SET UP FOR A NINE WORD WRITE FROM AN EVEN WORD BOUNDARY

```

```

9231
9232 031000 012701 000003      MOV      #3,R1          ;COUNT OF THREE
9233 031004 012702 003000      MOV      #BFEBVEN,R2    ;START THREE WORD BUFFER
9234                                     ;FROM AN EVEN WORD BOUNDARY
9235 031010 012722 052525      1$:  MOV      #52525,(R2)+ ;MOVE THREE 52525 INTO BUFFER
9236 031014 005301      DEC      R1             ;COUNT
9237 031016 001374      BNE     1$             ;BRANCH IF THREE NOT DONE
9238 031020 012777 003000 153036      MOV      #BFEBVEN,@RHBA ;SET BUS ADDRESS TO START
9239                                     ;FROM AN EVEN WORD BOUNDARY
9240 031026 012777 177767 153026      MOV      #-9,@RHCW      ;WORD COUNT NINE
9241 031034 004077 155220      JSR      RO,@COMND      ;GO TO DO COMMAND
9242 031040 004154      WRIDAT                    ;WRITE DATA
9243
9244

```

```

;CHECK THAT RHCS3 HAS 0

```

```

9245
9246 031042 012737 000000 001124      MOV      #0,$GDDAT      ;GET GOOD = 0
9247
9248 031050 017737 153056 001126      MOV      @RHCS3,$BDDAT  ;READ RHCS3 FOR COMPARISON

```



```

9305 031160 012737 000000 001124      MOV    #0,$GDDAT      ;GET GOOD = 0
9306                                     ;
9307 031166 017737 152670 001126      MOV    @RHWC,$BDDAT   ;READ RHWC FOR COMPARISON
9308 031174 023737 001124 001126      CMP    $GDDAT,$BDDAT  ;COMPARE EXPECTED
9309                                     ;DATA WITH DATA READ FROM
9310                                     ;RHWC
9311 031202 001401      BEQ    71$            ;BRANCH IF GOOD
9312 031204 104131      ERROR  131
9313                                     ;AFTER SETTING 'CLR' BIT #5
9314                                     ;IN RHCS2 TO INIT THE RH
9315                                     ;A NINE WORD WRITE FROM AN
9316                                     ;EVEN WORD BOUNDARY WAS DONE
9317                                     ;THEN
9318                                     ;RHWC SHOULD HAVE 0
9319                                     ;BUT CONTAINED WHAT IS
9320                                     ;GIVEN IN BAD RHWC

```

9321 031206 71\$:

```

9322
9323
9324
9325
9326
9327
9328
9329
9330
9331
9332
9333
9334
9335
*****
: *TEST 41      TEST DBL (RHCS3 BIT #10) TEST B

```

```

: *
: * 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

```

```

9336 031206 000004      TST41: SCOPE
9337 031210 012706 001000      MOV    #STACK,SP     ;RESET STACK
9338 031214 012737 000041 006276      MOV    #41,@TSTNM    ;SAVE TEST NUMBER
9339
9340 031222 004737 041222      JSR    PC,@WCLD!SK   ;GIVE RH INITIALIZE
9341                                     ;SETUP UNIT NUBER
9342                                     ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
9343

```

```

9344                                     ;SET UP FOR A TEN WORD WRITE FROM AN EVEN WORD BOUNDARY
9345 031226 012701 000003      MOV    #3,R1          ;COUNT OF THREE
9346 031232 012702 003000      MOV    #BFEVEN,R2     ;START THREE WORD BUFFER
9347                                     ;FROM AN EVEN WORD BOUNDARY
9348 031236 012722 052525      1$: MOV    #52525,(R2)+ ;MOVE THREE 52525 INTO BUFFER
9349 031242 005301      DEC    R1              ;COUNT
9350 031244 001374      BNE    1$              ;BRANCH IF THREE NOT DONE
9351 031246 012777 003000 152610      MOV    #BFEVEN,@RHBA  ;SET BUS ADDRESS TO START
9352                                     ;FROM AN EVEN WORD BOUNDARY
9353 031254 012777 177766 152600      MOV    #-10,@RHWC     ;WORD COUNT TEN
9354 031262 004077 154772      JSR    R0,@COMND      ;GO TO DO COMMAND
9355 031266 004154      WRIDAT                ;WRITE DATA
9356

```

```

9357
9358                                     ;CHECK THAT RHCS3 HAS DBL
9359 031270 012737 002000 001124      MOV    #DBL,$GDDAT    ;GET GOOD = 2000
9360

```



```

9529                                     :GIVEN IN BAD RHCS2
9530 031634                               69$:
9531                                     :CHECK THAT RHWC HAS 0
9532 031634 012737 000000 001124        MOV     #0,$GDDAT      :GET GOOD = 0
9533
9534 031642 017737 152214 001126        MOV     @RHWC,$BDDAT   :READ RHWC FOR COMPARISON
9535 031650 023737 001124 001126        CMP     $GDDAT,$BDDAT :COMPARE EXPECTED
9536                                     :DATA WITH DATA READ FROM
9537                                     :RHWC
9538 031656 001401                         BEQ     71$            :BRANCH IF GOOD
9539 031660 104131                         ERROR   131
9540                                     :AFTER SETTING 'CLR' BIT #5
9541                                     :IN RHCS2 TO INIT THE RH
9542                                     :A TEN WORD WRITE FROM AN
9543                                     :ODD WORD BOUNDARY WAS DONE
9544                                     :THEN
9545                                     :RHWC SHOULD HAVE 0
9546                                     :BUT CONTAINED WHAT IS
9547                                     :GIVEN IN BAD RHWC
9548 031662                               71$:
9549
9550
9551
9552                                     :*****
9553                                     :*TEST 43      TEST DBL (RHCS3 BIT #10) TEST D
9554
9555                                     :*
9556                                     :* CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
9557                                     :* SET UP FOR A ELEVEN WORD WRITE FROM AN EVEN WORD BOUNDARY
9558                                     :* DO A ELEVEN WORD WRITE FROM AN EVEN WORD BOUNDARY
9559                                     :* THIS SHOULD NOT SET RHCS3 BIT #10 DBL
9560                                     :* CHECK RHCS3
9561                                     :* CHECK RHCS1,RHCS2,RHBA,RHBAE,RHWC
9562                                     :*****
9563 031662 000004                               TST43: SCOPE
9564 031664 012706 001000                    MOV     #STACK,SP      :RESET STACK
9565 031670 012737 000043 006276            MOV     #43,@#TSTNM    :SAVE TEST NUMBER
9566
9567 031676 004737 041222                    JSR     PC,@#CLDISK    :GIVE RH INITIALIZE
9568                                     :SETUP UNIT NUBER
9569                                     :CLEAR RHWC AND FUNCTION BITS IN RHCS1
9570
9571                                     :SET UP FOR A ELEVEN WORD WRITE FROM AN EVEN WORD BOUNDARY
9572 031702 012701 000003                    MOV     #3,R1          :COUNT OF THREE
9573 031706 012702 003000                    MOV     #BFEVEN,R2     :START THREE WORD BUFFER
9574                                     :FROM AN EVEN WORD BOUNDARY
9575 031712 012722 052525                               1$: MOV     #52525,(R2)+  :MOVE THREE 52525 INTO BUFFER
9576 031716 005301                         DEC     R1              :COUNT
9577 031720 001374                         BNE     1$             :BRANCH IF THREE NOT DONE
9578 031722 012777 003000 152134            MOV     #BFEVEN,@RHBA  :SET BUS ADDRESS TO START
9579                                     :FROM AN EVEN WORD BOUNDARY
9580 031730 012777 177765 152124            MOV     #-11,@RHWC     :WORD COUNT ELEVEN
9581 031736 004077 154316                    JSR     R0,@COMND      :GO TO DO COMMAND
9582 031742 004154                         WRIDAT                 :WRITE DATA
9583
9584
    
```

```

9585 ;CHECK THAT RHCS3 HAS 0
9586 031744 012737 000000 001124 MOV #0,$GDDAT ;GET GOOD = 0
9587
9588 031752 017737 152154 001126 MOV @RHCS3,$BDDAT ;READ RHCS3 FOR COMPARISON
9589 031760 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
9590 ;DATA WITH DATA READ FROM
9591 ;RHCS3
9592 031766 001401 BEQ 65$ ;BRANCH IF GOOD
9593 031770 104124 ERROR 124
9594 ;AFTER SETTING 'CLR' BIT #5
9595 ;IN RHCS2 TO INIT THE RH
9596 ;A ELEVEN WORD WRITE FROM AN
9597 ;EVEN WORD BOUNDARY WAS DONE
9598 ;THEN
9599 ;RHCS3 SHOULD HAVE 0
9600 ;BUT CONTAINED WHAT IS
9601 ;GIVEN IN BAD RHCS3
9602 031772 65$:
9603 031772 042777 000076 152060 BIC #76,@RHCS1 ;CLEAR FUNCTION BITS
9604 ;CHECK THAT RHCS1 HAS RDY!DVA
9605 032000 012737 004200 001124 MOV #RDY!DVA,$GDDAT ;GET GOOD = 4200
9606
9607 032006 017737 152046 001126 MOV @RHCS1,$BDDAT ;READ RHCS1 FOR COMPARISON
9608 032014 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
9609 ;DATA WITH DATA READ FROM
9610 ;RHCS1
9611 032022 001401 BEQ 67$ ;BRANCH IF GOOD
9612 032024 104125 ERROR 125
9613 ;AFTER SETTING 'CLR' BIT #5
9614 ;IN RHCS2 TO INIT THE RH
9615 ;A ELEVEN WORD WRITE FROM AN
9616 ;EVEN WORD BOUNDARY WAS DONE
9617 ;THEN
9618 ;RHCS1 SHOULD HAVE RDY!DVA
9619 ;=4200
9620 ;BUT CONTAINED WHAT IS
9621 ;GIVEN IN BAD RHCS1
9622 032026 67$:
9623 ;CHECK THAT RHCS2 HAS IR
9624 032026 012737 000100 001124 MOV #IR,$GDDAT ;GET GOOD = 100
9625 032034 053737 006442 001124 BIS UNIT,$GDDAT ;INCLUDE UNIT NUMBER
9626
9627 032042 017737 152022 001126 MOV @RHCS2,$BDDAT ;READ RHCS2 FOR COMPARISON
9628 032050 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
9629 ;DATA WITH DATA READ FROM
9630 ;RHCS2
9631 032056 001401 BEQ 69$ ;BRANCH IF GOOD
9632 032060 104126 ERROR 126
9633 ;AFTER SETTING 'CLR' BIT #5
9634 ;IN RHCS2 TO INIT THE RH
9635 ;A ELEVEN WORD WRITE FROM AN
9636 ;EVEN WORD BOUNDARY WAS DONE
9637 ;THEN
9638 ;RHCS2 SHOULD HAVE IR
9639 ;=100
9640 ;TOGETHER WITH UNIT NUMBER
    
```



```

9641                                     ;BUT CONTAINED WHAT IS
9642                                     ;GIVEN IN BAD RHCS2
9643 032062                               69$:
9644                                     ;CHECK THAT RHWC HAS 0
9645 032062 012737 000000 001124      MOV    #0,$GDDAT    ;GET GOOD = 0
9646
9647 032070 017737 151766 001126      MOV    @RHWC,$BDDAT ;READ RHWC FOR COMPARISON
9648 032076 023737 001124 001126      CMP    $GDDAT,$BDDAT ;COMPARE EXPECTED
9649                                     ;DATA WITH DATA READ FROM
9650                                     ;RHWC
9651 032104 001401                       BEQ    71$          ;BRANCH IF GOOD
9652 032106 104131
9653                                     ;AFTER SETTING 'CLR' BIT #5
9654                                     ;IN RHCS2 TO INIT THE RH
9655                                     ;A ELEVEN WORD WRITE FROM AN
9656                                     ;EVEN WORD BOUNDARY WAS DONE
9657                                     ;THEN
9658                                     ;RHWC SHOULD HAVE 0
9659                                     ;BUT CONTAINED WHAT IS
9660                                     ;GIVEN IN BAD RHWC
9661 032110                               71$:
9662
9663
9664
9665
9666 *****
9667 *TEST 44      TEST DBL (RHCS3 BIT #10) TEST E
9668
9669 **          CLEAR THE SUBSYSTEM BY A CONTROLLER (CLEAR (RHCS2 BIT #5)
9670 **          SET UP FOR A TEN WORD WRITE FROM AN EVEN WORD BOUNDARY
9671 **          WITH BAI IN RHCS2 BIT #3 SET
9672 **          DO A TEN WORD WRITE FROM AN EVEN WORD BOUNDARY
9673 **          THIS SHOULD NOT SET RHCS3 BIT #10 DBL
9674 **          CHECK RHCS3
9675 **          CHECK RHCS1,RHCS2,RHBA,RHBAE,RHWC
9676 *****
9677 032110 000004      TEST44: SCOPE
9678 032112 012706 001000      MOV    #STACK,SP    ;RESET STACK
9679 032116 012737 000044 006276  MOV    #44,@TSTNM   ;SAVE TEST NUMBER
9680
9681 032124 004737 041222      JSR    PC,@CLDISK   ;GIVE RH INITIALIZE
9682                                     ;SETUP UNIT NUBER
9683                                     ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
9684
9685                                     ;SET UP FOR A TEN WORD WRITE FROM AN EVEN WORD BOUNDARY
9686 032130 012701 000003      MOV    #3,R1        ;COUNT OF THREE
9687 032134 012702 003000      MOV    #BFEVEN,R2   ;START THREE WORD BUFFER
9688                                     ;FROM AN EVEN WORD BOUNDARY
9689 032140 012722 052525      1$:  MOV    #52525,(R2)+ ;MOVE THREE 52525 INTO BUFFER
9690 032144 005301                       DEC    R1            ;COUNT
9691 032146 001374                       BNE    1$           ;BRANCH IF THREE NOT DONE
9692 032150 012777 003000 151706      MOV    #BFEVEN,@RHBA ;SET BUS ADDRESS TO START
9693                                     ;FROM AN EVEN WORD BOUNDARY
9694 032156 012777 177766 151676      MOV    #-10,@RHWC   ;WORD COUNT TEN
9695 032164 052777 000010 151676      BIS    #BAI,@RHCS2  ;SET BAI IN RHCS2
9696 032172 004077 154062      JSR    R0,@COMND    ;GO TO DO COMMAND

```

```

9697 032176 004154 WRIDAT ;WRITE DATA
9698
9699
9700 ;CHECK THAT RHCS3 HAS 0
9701 032200 012737 000000 001124 MOV #0,$GDDAT ;GET GOOD = 0
9702
9703 032206 017737 151720 001126 MOV @RHCS3,$BDDAT ;READ RHCS3 FOR COMPARISON
9704 032214 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
9705 ;DATA WITH DATA READ FROM
9706 ;RHCS3
9707 032222 001401 BEQ 65$ ;BRANCH IF GOOD
9708 032224 104124 ERROR 124
9709 ;AFTER SETTING 'CLR' BIT #5
9710 ;IN RHCS2 TO INIT THE RH
9711 ;A TEN WORD WRITE FROM AN
9712 ;EVEN WORD BOUNDARY WAS DONE
9713 ;WITH BAI IN RHCS2 SET
9714 ;THEN
9715 ;RHCS3 SHOULD HAVE 0
9716 ;BUT CONTAINED WHAT IS
9717 ;GIVEN IN BAD RHCS3
9718 032226 65$:
9719 032226 042777 000076 151624 BIC #76,@RHCS1 ;CLEAR FUNCTION BITS
9720 ;CHECK THAT RHCS1 HAS RDY!DVA
9721 032234 012737 004200 001124 MOV #RDY!DVA,$GDDAT ;GET GOOD = 4200
9722
9723 032242 017737 151612 001126 MOV @RHCS1,$BDDAT ;READ RHCS1 FOR COMPARISON
9724 032250 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
9725 ;DATA WITH DATA READ FROM
9726 ;RHCS1
9727 032256 001401 BEQ 67$ ;BRANCH IF GOOD
9728 032260 104125 ERROR 125
9729 ;AFTER SETTING 'CLR' BIT #5
9730 ;IN RHCS2 TO INIT THE RH
9731 ;A TEN WORD WRITE FROM AN
9732 ;EVEN WORD BOUNDARY WAS DONE
9733 ;WITH BAI IN RHCS2 SET
9734 ;THEN
9735 ;RHCS1 SHOULD HAVE RDY!DVA
9736 ;=4200
9737 ;BUT CONTAINED WHAT IS
9738 ;GIVEN IN BAD RHCS1
9739 032262 67$:
9740 ;CHECK THAT RHCS2 HAS IR!BAI
9741 032262 012737 000110 001124 MOV #IR!BAI,$GDDAT ;GET GOOD = 110
9742 032270 053737 006442 001124 BIS UNIT,$GDDAT ;INCLUDE UNIT NUMBER
9743
9744 032276 017737 151566 001126 MOV @RHCS2,$BDDAT ;READ RHCS2 FOR COMPARISON
9745 032304 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
9746 ;DATA WITH DATA READ FROM
9747 ;RHCS2
9748 032312 001401 BEQ 69$ ;BRANCH IF GOOD
9749 032314 104126 ERROR 126
9750 ;AFTER SETTING 'CLR' BIT #5
9751 ;IN RHCS2 TO INIT THE RH
9752 ;A TEN WORD WRITE FROM AN

```


9753 ;EVEN WORD BOUNDARY WAS DONE
9754 ;WITH BAI IN RHCS2 SET
9755 ;THEN
9756 ;RHCS2 SHOULD HAVE IR!BAI
9757 ;=110
9758 ;TOGETHER WITH UNIT NUMBER
9759 ;BUT CONTAINED WHAT IS
9760 ;GIVEN IN BAD RHCS2

9761 032316 69\$:
9762 ;CHECK THAT RHWC HAS 0
9763 032316 012737 000000 001124 MOV #0,\$GDDAT ;GET GOOD = 0

9764
9765 032324 017737 151532 001126 MOV @RHWC,\$BDDAT ;READ RHWC FOR COMPARISON
9766 032332 023737 001124 001126 CMP \$GDDAT,\$BDDAT ;COMPARE EXPECTED
9767 ;DATA WITH DATA READ FROM
9768 ;RHWC
9769 032340 001401 BEQ 71\$;BRANCH IF GOOD
9770 032342 104131 ERROR 131

9771 ;AFTER SETTING "CLR" BIT #5
9772 ;IN RHCS2 TO INIT THE RH
9773 ;A TEN WORD WRITE FROM AN
9774 ;EVEN WORD BOUNDARY WAS DONE
9775 ;WITH BAI IN RHCS2 SET
9776 ;THEN
9777 ;RHWC SHOULD HAVE 0
9778 ;BUT CONTAINED WHAT IS
9779 ;GIVEN IN BAD RHWC

9780 032344 71\$:

9781
9782
9783
9784 :*****
9785 :*TEST 45 TEST DBL (RHCS3 BIT #10) TEST F

9786
9787 :* CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
9788 :* SET UP FOR A ELEVEN WORD WRITE FROM AN ODD WORD BOUNDARY
9789 :* DO A ELEVEN WORD WRITE FROM AN ODD WORD BOUNDARY
9790 :* THIS SHOULD SET RHCS3 BIT #10 DBL
9791 :* CHECK RHCS3
9792 :* CHECK RHCS1,RHCS2,RHBA,RHBAE,RHWC

9793
9794 :*****
9795 TST45: SCOPE
9796 032344 000004 MOV #STACK,SP ;RESET STACK
9797 032346 012706 001000 MOV #45,@TSTNM ;SAVE TEST NUMBER
9798 032352 012737 000045 006276
9799 032360 004737 041222 JSR PC,@CLDISK ;GIVE RH INITIALIZE
9800 ;SETUP UNIT NUBER
9801 ;CLEAR RHWC AND FUNCTION BITS IN RHCS1

9802
9803 ;SET UP FOR A ELEVEN WORD WRITE FROM AN ODD WORD BOUNDARY
9804 032364 012701 000003 MOV #3,R1 ;COUNT OF THREE
9805 032370 012702 003002 MOV #BFODD,R2 ;START THREE WORD BUFFER
9806 ;FROM AN ODD WORD BOUNDARY
9807 032374 012722 052525 1\$: MOV #52525,(R2)+ ;MOVE THREE 52525 INTO BUFFER
9808 032400 005301 DEC R1 ;COUNT

```

9809 032402 001374          BNE      1$          ;BRANCH IF THREE NOT DONE ;
9810 032404 012777 003002 151452  MOV      #BFODD,@RHBA ;SET BUS ADDRESS TO START ;
9811                                ;FROM AN ODD WORD BOUNDARY ;
9812 032412 012777 177765 151442  MOV      #-11,@RHWC ;WORD COUNT ELEVEN ;
9813 032420 004077 153634          JSR      RO,@COMND   ;GO TO DO COMMAND ;
9814 032424 004154          WRIDAT          ;WRITE DATA ;
9815
9816
9817                                ;CHECK THAT RHCS3 HAS DBL ;
9818 032426 012737 002000 001124  MOV      #DBL,$GDDAT ;GET GOOD = 2000 ;
9819
9820 032434 017737 151472 001126  MOV      @RHCS3,$BDDAT ;READ RHCS3 FOR COMPARISON ;
9821 032442 023737 001124 001126  CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED ;
9822                                ;DATA WITH DATA READ FROM ;
9823                                ;RHCS3 ;
9824 032450 001401          BEQ      65$          ;BRANCH IF GOOD ;
9825 032452 104124          ERROR     124
9826                                ;AFTER SETTING 'CLR' BIT #5 ;
9827                                ;IN RHCS2 TO INIT THE RH ;
9828                                ;A ELEVEN WORD WRITE FROM AN ;
9829                                ;ODD WORD BOUNDARY WAS DONE ;
9830                                ;THEN ;
9831                                ;RHCS3 SHOULD HAVE DBL ;
9832                                ;=2000 ;
9833                                ;BUT CONTAINED WHAT IS ;
9834                                ;GIVEN IN BAD RHCS3 ;
9835 032454          65$:
9836 032454 042777 000076 151376  BIC      #76,@RHCS1 ;CLEAR FUNCTION BITS ;
9837                                ;CHECK THAT RHCS1 HAS RDY!DVA ;
9838 032462 012737 004200 001124  MOV      #RDY!DVA,$GDDAT ;GET GOOD = 4200 ;
9839
9840 032470 017737 151364 001126  MOV      @RHCS1,$BDDAT ;READ RHCS1 FOR COMPARISON ;
9841 032476 023737 001124 001126  CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED ;
9842                                ;DATA WITH DATA READ FROM ;
9843                                ;RHCS1 ;
9844 032504 001401          BEQ      67$          ;BRANCH IF GOOD ;
9845 032506 104125          ERROR     125
9846                                ;AFTER SETTING 'CLR' BIT #5 ;
9847                                ;IN RHCS2 TO INIT THE RH ;
9848                                ;A ELEVEN WORD WRITE FROM AN ;
9849                                ;ODD WORD BOUNDARY WAS DONE ;
9850                                ;THEN ;
9851                                ;RHCS1 SHOULD HAVE RDY!DVA ;
9852                                ;=4200 ;
9853                                ;BUT CONTAINED WHAT IS ;
9854                                ;GIVEN IN BAD RHCS1 ;
9855 032510          67$:
9856                                ;CHECK THAT RHCS2 HAS IR ;
9857 032510 012737 000100 001124  MOV      #IR,$GDDAT   ;GET GOOD = 100 ;
9858 032516 053737 006442 001124  BIS      UNIT,$GDDAT ;INCLUDE UNIT NUMBER ;
9859
9860 032524 017737 151340 001126  MOV      @RHCS2,$BDDAT ;READ RHCS2 FOR COMPARISON ;
9861 032532 023737 001124 001126  CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED ;
9862                                ;DATA WITH DATA READ FROM ;
9863                                ;RHCS2 ;
9864 032540 001401          BEQ      69$          ;BRANCH IF GOOD ;
    
```


9865 032542 104126 ERROR 126
9866
9867
9868
9869
9870
9871
9872
9873
9874
9875
9876 032544
9877
9878 032544 012737 000000 001124
9879
9880 032552 017737 151304 001126
9881 032560 023737 001124 001126
9882
9883
9884 032566 001401
9885 032570 104131
9886
9887
9888
9889
9890
9891
9892
9893
9894 032572
9895
9896
9897
9898
9899
9900
9901
9902
9903
9904
9905
9906
9907
9908
9909
9910 032572 000004
9911 032574 012706 001000
9912 032600 012737 000046 006276
9913
9914 032606 004737 041222
9915
9916
9917
9918
9919 032612 012701 000003
9920 032616 012702 003000

69\$:

:CHECK THAT RHWC HAS 0
MOV #0,\$GDDAT

:AFTER SETTING 'CLR' BIT #5
:IN RHCS2 TO INIT THE RH
:A ELEVEN WORD WRITE FROM AN
:ODD WORD BOUNDARY WAS DONE
:THEN
:RHCS2 SHOULD HAVE IR
:=100
:TOGETHER WITH UNIT NUMBER
:BUT CONTAINED WHAT IS
:GIVEN IN BAD RHCS2

:GET GOOD = 0

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

:READ RHWC FOR COMPARISON
:COMPARE EXPECTED
:DATA WITH DATA READ FROM
:RHWC
:BRANCH IF GOOD

BEQ 71\$
ERROR 131

:AFTER SETTING 'CLR' BIT #5
:IN RHCS2 TO INIT THE RH
:A ELEVEN WORD WRITE FROM AN
:ODD WORD BOUNDARY WAS DONE
:THEN
:RHWC SHOULD HAVE 0
:BUT CONTAINED WHAT IS
:GIVEN IN BAD RHWC

71\$:

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

:* 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

TST46:

SCOPE

MOV #STACK,SP

:RESET STACK

MOV #46,@TSTNM

:SAVE TEST NUMBER

JSR PC,@WCLDISK

:GIVE RH INITIALIZE

:SETUP UNIT NUMBER

:CLEAR RHWC AND FUNCTION BITS IN RHCS1

:SET UP FOR A NINE WORD READ FROM AN EVEN WORD BOUNDARY

MOV #3,R1

:COUNT OF THREE

MOV #BFEEVEN,R2

:START THREE WORD BUFFER


```
9977                                     ;RHCS2
9978 032766 001401                       BEQ      69$
9979 032770 104126                       ERROR    126
9980                                     ;AFTER SETTING "CLR" BIT #5
9981                                     ;IN RHCS2 TO INIT THE RH
9982                                     ;A NINE WORD READ FROM AN
9983                                     ;EVEN WORD BOUNDARY WAS DONE
9984                                     ;THEN
9985                                     ;RHCS2 SHOULD HAVE IR
9986                                     ;=100
9987                                     ;TOGETHER WITH UNIT NUMBER
9988                                     ;BUT CONTAINED WHAT IS
9989                                     ;GIVEN IN BAD RHCS2
9990 032772                               69$:
9991                                     ;CHECK THAT RHCW HAS 0
9992 032772 012737 000000 001124          MOV      #0,$GDDAT ;GET GOOD = 0
9993
9994 033000 017737 151056 001126          MOV      @RHCW,$BDDAT ;READ RHCW FOR COMPARISON
9995 033006 023737 001124 001126          CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED
9996                                     ;DATA WITH DATA READ FROM
9997                                     ;RHCW
9998 033014 001401                       BEQ      71$
9999 033016 104131                       ERROR    131
10000                                     ;AFTER SETTING "CLR" BIT #5
10001                                     ;IN RHCS2 TO INIT THE RH
10002                                     ;A NINE WORD READ FROM AN
10003                                     ;EVEN WORD BOUNDARY WAS DONE
10004                                     ;THEN
10005                                     ;RHCW SHOULD HAVE 0
10006                                     ;BUT CONTAINED WHAT IS
10007                                     ;GIVEN IN BAD RHCW
10008 033020                               71$:
10009
10010
10011
10012
10013
10014
10015
10016
10017
10018
10019
10020
10021
10022
10023 033020 000004
10024 033022 012706 001000
10025 033026 012737 000047 006276
10026
10027 033034 004737 041222
10028
10029
10030
10031
10032 033040 012701 000003

*****
;*TEST 47 TEST DBL (RHCS3 BIT #10) TEST H
*****
;* 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,RHCW
*****
TST47: SCOPE
MOV      #STACK,SP ;RESET STACK
MOV      #47,@TSTNM ;SAVE TEST NUMBER
JSR      PC,@WCLDISK ;GIVE RH INITIALIZE
;SETUP UNIT NUMBER
;CLEAR RHCW AND FUNCTION BITS IN RHCS1
;SET UP FOR A TEN WORD READ FROM AN EVEN WORD BOUNDARY
MOV      #3,R1 ;COUNT OF THREE
```

```

10033 033044 012702 003000      MOV      #BFEVEN,R2      ;START THREE WORD BUFFER
10034                                ;FROM AN EVEN WORD BOUNDARY
10035 033050 012722 052525      1$: MOV      #52525,(R2)+  ;MOVE THREE 52525 INTO BUFFER
10036 033054 005301              DEC      R1              ;COUNT
10037 033056 001374              BNE     1$              ;BRANCH IF THREE NOT DONE
10038 033060 012777 003000 150776  MOV      #BFEVEN,@RHBA   ;SET BUS ADDRESS TO START
10039                                ;FROM AN EVEN WORD BOUNDARY
10040 033066 012777 177766 150766  MOV #-10,@RHWC ;WORD COUNT TEN
10041 033074 004077 153160      JSR     R0,@COMND      ;GO TO DO COMMAND
10042 033100 004160      READAT ;READ DATA
10043
10044
10045                                ;CHECK THAT RHCS3 HAS DBL
10046 033102 012737 002000 001124  MOV      #DBL,$GDDAT    ;GET GOOD = 2000
10047
10048 033110 017737 151016 001126  MOV      @RHCS3,$BDDAT  ;READ RHCS3 FOR COMPARISON
10049 033116 023737 001124 001126  CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED
10050                                ;DATA WITH DATA READ FROM
10051                                ;RHCS3
10052 033124 001401      BEQ     65$            ;BRANCH IF GOOD
10053 033126 104124      ERROR   124
10054                                ;AFTER SETTING 'CLR' BIT #5
10055                                ;IN RHCS2 TO INIT THE RH
10056                                ;A TEN WORD READ FROM AN
10057                                ;EVEN WORD BOUNDARY WAS DONE
10058                                ;THEN
10059                                ;RHCS3 SHOULD HAVE DBL
10060                                ;=2000
10061                                ;BUT CONTAINED WHAT IS
10062                                ;GIVEN IN BAD RHCS3
10063 033130      65$:
10064 033130 042777 000076 150722  BIC      #76,@RHCS1    ;CLEAR FUNCTION BITS
10065                                ;CHECK THAT RHCS1 HAS RDY!DVA
10066 033136 012737 004200 001124  MGV      #RDY!DVA,$GDDAT ;GET GOOD = 4200
10067
10068 033144 017737 150710 001126  MOV      @RHCS1,$BDDAT  ;READ RHCS1 FOR COMPARISON
10069 033152 023737 001124 001126  CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED
10070                                ;DATA WITH DATA READ FROM
10071                                ;RHCS1
10072 033160 001401      BEQ     67$            ;BRANCH IF GOOD
10073 033162 104125      ERROR   125
10074                                ;AFTER SETTING 'CLR' BIT #5
10075                                ;IN RHCS2 TO INIT THE RH
10076                                ;A TEN WORD READ FROM AN
10077                                ;EVEN WORD BOUNDARY WAS DONE
10078                                ;THEN
10079                                ;RHCS1 SHOULD HAVE RDY!DVA
10080                                ;=4200
10081                                ;BUT CONTAINED WHAT IS
10082                                ;GIVEN IN BAD RHCS1
10083 033164      67$:
10084                                ;CHECK THAT RHCS2 HAS IR
10085 033164 012737 000100 001124  MOV      #IR,$GDDAT    ;GET GOOD = 100
10086 033172 053737 006442 001124  BIS      UNIT,$GDDAT   ;INCLUDE UNIT NUMBER
10087
10088 033200 017737 150664 001126  MOV      @RHCS2,$BDDAT ;READ RHCS2 FOR COMPARISON
    
```



```

10089 033206 023737 001124 001126      CMP      $GDDAT,$BDDAT      :COMPARE EXPECTED
10090                                     :DATA WITH DATA READ FROM
10091                                     :RHCS2
10092 033214 001401                      BEQ      69$                 :BRANCH IF GO.
10093 033216 104126                      ERROR   126
10094                                     :AFTER SETTING 'CLR' BIT #5
10095                                     :IN RHCS2 TO INIT THE RH
10096                                     :A TEN WORD READ FROM AN
10097                                     :EVEN WORD BOUNDARY WAS DONE
10098                                     :THEN
10099                                     :RHCS2 SHOULD HAVE IR
10100                                     :=100
10101                                     :TOGETHER WITH UNIT NUMBER
10102                                     :BUT CONTAINED WHAT IS
10103                                     :GIVEN IN BAD RHCS2
10104 033220                               69$:
10105                                     :CHECK THAT RHCW HAS 0
10106 033220 012737 000000 001124      MOV      #0,$GDDAT         :GET GOOD = 0
10107
10108 033226 017737 150630 001126      MOV      @RHCW,$BDDAT      :READ RHCW FOR COMPARISON
10109 033234 023737 001124 001126      CMP      $GDDAT,$BDDAT    :COMPARE EXPECTED
10110                                     :DATA WITH DATA READ FROM
10111                                     :RHCW
10112 033242 001401                      BEQ      71$                 :BRANCH IF GOOD
10113 033244 104131                      ERROR   131
10114                                     :AFTER SETTING 'CLR' BIT #5
10115                                     :IN RHCS2 TO INIT THE RH
10116                                     :A TEN WORD READ FROM AN
10117                                     :EVEN WORD BOUNDARY WAS DONE
10118                                     :THEN
10119                                     :RHCW SHOULD HAVE 0
10120                                     :BUT CONTAINED WHAT IS
10121                                     :GIVEN IN BAD RHCW
10122 033246                               71$:
10123
10124
10125
10126
10127
10128
10129
10130
10131
10132
10133
10134
10135
10136
10137 033246 000004                      TST50: SCOPE
10138 033250 012706 001000                      MOV      #STACK,SP         :RESET STACK
10139 033254 012737 000050 006276      MOV      #50,@#TSTNM      :SAVE TEST NUMBER
10140
10141 033262 004737 041222                      JSR      PC,@#CLDISK      :GIVE RH INITIALIZE
10142                                     :SETUP UNIT NUBER
10143                                     :CLEAR RHCW AND FUNCTION BITS IN RHCS1
10144

```

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

: * 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
: * CHECK RHCS1,RHCS2,RHBA,RHBAE,RHCW

```

10145 ;SET UP FOR A TEN WORD READ FROM AN ODD WORD BOUNDARY
10146 033266 012701 000003 MOV #3,R1 ;COUNT OF THREE
10147 033272 012702 003002 MOV #BFODD,R2 ;START THREE WORD BUFFER
10148 ;FROM AN ODD WORD BOUNDARY
10149 033276 012722 052525 1$: MOV #52525,(R2)+ ;MOVE THREE 52525 INTO BUFFER
10150 033302 005301 DEC R1 ;COUNT
10151 033304 001374 BNE 1$ ;BRANCH IF THREE NOT DONE
10152 033306 012777 003002 150550 MOV #BFODD,@RHBA ;SET BUS ADDRESS TO START
10153 ;FROM AN ODD WORD BOUNDARY
10154 033314 012777 177766 150540 MOV #-10,@RHWC ;WORD COUNT TEN
10155 033322 004077 152732 JSR R0,@COMND ;GO TO DO COMMAND
10156 033326 004160 READAT ;READ DATA
10157
10158
10159 ;CHECK THAT RHCS3 HAS 0
10160 033330 012737 000000 001124 MOV #0,$GDDAT ;GET GOOD = 0
10161
10162 033336 017737 150570 001126 MOV @RHCS3,$BDDAT ;READ RHCS3 FOR COMPARISON
10163 033344 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
10164 ;DATA WITH DATA READ FROM
10165 ;RHCS3
10166 033352 001401 BEQ 65$ ;BRANCH IF GOOD
10167 033354 104124 ERROR 124
10168 ;AFTER SETTING 'CLR' BIT #5
10169 ;IN RHCS2 TO INIT THE RH
10170 ;A TEN WORD READ FROM AN
10171 ;ODD WORD BOUNDARY WAS DONE
10172 ;THEN
10173 ;RHCS3 SHOULD HAVE 0
10174 ;BUT CONTAINED WHAT IS
10175 ;GIVEN IN BAD RHCS3
10176 033356 042777 000076 150474 65$: BIC #76,@RHCS1 ;CLEAR FUNCTION BITS
10177 033356 ;CHECK THAT RHCS1 HAS RDY!DVA
10178 MOV #RDY!DVA,$GDDAT ;GET GOOD = 4200
10179 033364 012737 004200 001124
10180
10181 033372 017737 150462 001126 MOV @RHCS1,$BDDAT ;READ RHCS1 FOR COMPARISON
10182 033400 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
10183 ;DATA WITH DATA READ FROM
10184 ;RHCS1
10185 033406 001401 BEQ 67$ ;BRANCH IF GOOD
10186 033410 104125 ERROR 125
10187 ;AFTER SETTING 'CLR' BIT #5
10188 ;IN RHCS2 TO INIT THE RH
10189 ;A TEN WORD READ FROM AN
10190 ;ODD WORD BOUNDARY WAS DONE
10191 ;THEN
10192 ;RHCS1 SHOULD HAVE RDY!DVA
10193 ;=4200
10194 ;BUT CONTAINED WHAT IS
10195 ;GIVEN IN BAD RHCS1
10196 033412 67$:
10197 ;CHECK THAT RHCS2 HAS IR
10198 033412 012737 000100 001124 MOV #IR,$GDDAT ;GET GOOD = 100
10199 033420 053737 006442 001124 BIS UNIT,$GDDAT ;INCLUDE UNIT NUMBER
10200

```



```

10201 033426 017737 150436 001126      MOV    @RHCS2,$BDDAT  ;READ RHCS2 FOR COMPARISON
10202 033434 023737 001124 001126      CMP    $GDDAT,$BDDAT ;COMPARE EXPECTED
10203                                     ;DATA WITH DATA READ FROM
10204                                     ;RHCS2
10205 033442 001401      BEQ    69$           ;BRANCH IF GOOD
10206 033444 104126      ERROR  126
10207                                     ;AFTER SETTING 'CLR' BIT #5
10208                                     ;IN RHCS2 TO INIT THE RH
10209                                     ;A TEN WORD READ FROM AN
10210                                     ;ODD WORD BOUNDARY WAS DONE
10211                                     ;THEN
10212                                     ;RHCS2 SHOULD HAVE IR
10213                                     ;=100
10214                                     ;TOGETHER WITH UNIT NUMBER
10215                                     ;BUT CONTAINED WHAT IS
10216                                     ;GIVEN IN BAD RHCS2
10217 033446                                     69$:
10218                                     ;CHECK THAT RHCW HAS 0
10219 033446 012737 000000 001124      MOV    #0,$GDDAT     ;GET GOOD = 0
10220
10221 033454 017737 150402 001126      MOV    @RHCW,$BDDAT ;READ RHCW FOR COMPARISON
10222 033462 023737 001124 001126      CMP    $GDDAT,$BDDAT ;COMPARE EXPECTED
10223                                     ;DATA WITH DATA READ FROM
10224                                     ;RHCW
10225 033470 001401      BEQ    71$           ;BRANCH IF GOOD
10226 033472 104131      ERROR  131
10227                                     ;AFTER SETTING 'CLR' BIT #5
10228                                     ;IN RHCS2 TO INIT THE RH
10229                                     ;A TEN WORD READ FROM AN
10230                                     ;ODD WORD BOUNDARY WAS DONE
10231                                     ;THEN
10232                                     ;RHCW SHOULD HAVE 0
10233                                     ;BUT CONTAINED WHAT IS
10234                                     ;GIVEN IN BAD RHCW
10235 033474                                     71$:
10236
10237
10238
10239                                     ;*****
10240                                     ;*TEST 51      TEST DBL (RHCS3 BIT #10) TEST J
10241
10242                                     ;*
10243                                     ;*      CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
10244                                     ;*      SET UP FOR A ELEVEN WORD READ FROM AN EVEN WORD BOUNDARY
10245                                     ;*      DO A ELEVEN WORD READ FROM AN EVEN WORD BOUNDARY
10246                                     ;*      THIS SHOULD NOT SET RHCS3 BIT #10 DBL
10247                                     ;*      CHECK RHCS3
10248                                     ;*      CHECK RHCS1,RHCS2,RHBA,RHBAE,RHCW
10249                                     ;*****
10250 033474 000004      TST51: SCOPE
10251 033476 012706 001000      MOV    #STACK,SP     ;RESET STACK
10252 033502 012737 000051 006276      MOV    #51,@#TSTNM   ;SAVE TEST NUMBER
10253
10254 033510 004737 041222      JSR    PC,@#CLDISK   ;GIVE RH INITIALIZE
10255                                     ;SETUP UNIT NUBER
10256                                     ;CLEAR RHCW AND FUNCTION BITS IN RHCS1

```

```

10257
10258 ;SET UP FOR A ELEVEN WORD READ FROM AN EVEN WORD BOUNDARY
10259 033514 012701 000003 MOV #3,R1 ;COUNT OF THREE
10260 033520 012702 003000 MOV #BFEVEN,R2 ;START THREE WORD BUFFER
10261 ;FROM AN EVEN WORD BOUNDARY
10262 033524 012722 052525 1$: MOV #52525,(R2)+ ;MOVE THREE 52525 INTO BUFFER
10263 033530 005301 DEC R1 ;COUNT
10264 033532 001374 BNE 1$ ;BRANCH IF THREE NOT DONE
10265 033534 012777 003000 150322 MOV #BFEVEN,@RHBA ;SET BUS ADDRESS TO START
10266 ;FROM AN EVEN WORD BOUNDARY
10267 033542 012777 177765 150312 MOV #-11,@RHWC ;WORD COUNT ELEVEN
10268 033550 004077 152504 JSR R0,@COMND ;GO TO DO COMMAND
10269 033554 004160 READAT ;READ DATA
10270
10271
10272 ;CHECK THAT RHCS3 HAS 0
10273 033556 012737 000000 001124 MOV #0,$GDDAT ;GET GOOD = 0
10274
10275 033564 017737 150342 001126 MOV @RHCS3,$BDDAT ;READ RHCS3 FOR COMPARISON
10276 033572 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
10277 ;DATA WITH DATA READ FROM
10278 ;RHCS3
10279 033600 001401 BEQ 65$ ;BRANCH IF GOOD
10280 033602 104124 ERROR 124
10281 ;AFTER SETTING 'CLR' BIT #5
10282 ;IN RHCS2 TO INIT THE RH
10283 ;A ELEVEN WORD READ FROM AN
10284 ;EVEN WORD BOUNDARY WAS DONE
10285 ;THEN
10286 ;RHCS3 SHOULD HAVE 0
10287 ;BUT CONTAINED WHAT IS
10288 ;GIVEN IN BAD RHCS3
10289 033604 042777 000076 150246 65$: BIC #76,@RHCS1 ;CLEAR FUNCTION BITS
10290 033604 012737 004200 001124 ;CHECK THAT RHCS1 HAS RDY!DVA
10291 MOV #RDY!DVA,$GDDAT ;GET GOOD = 4200
10292 033612 017737 150234 001126 MOV @RHCS1,$BDDAT ;READ RHCS1 FOR COMPARISON
10293 033620 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
10294 ;DATA WITH DATA READ FROM
10295 ;RHCS1
10296 ;BRANCH IF GOOD
10297 BEQ 67$
10298 033634 001401 ERROR 125
10299 033636 104125
10300 ;AFTER SETTING 'CLR' BIT #5
10301 ;IN RHCS2 TO INIT THE RH
10302 ;A ELEVEN WORD READ FROM AN
10303 ;EVEN WORD BOUNDARY WAS DONE
10304 ;THEN
10305 ;RHCS1 SHOULD HAVE RDY!DVA
10306 ;=4200
10307 ;BUT CONTAINED WHAT IS
10308 ;GIVEN IN BAD RHCS1
10309 033640 053737 000100 001124 67$: ;CHECK THAT RHCS2 HAS IR
10310 MOV #IR,$GDDAT ;GET GOOD = 100
10311 033640 012737 006442 001124 BIS UNIT,$GDDAT ;INCLUDE UNIT NUMBER
10312 033646 053737 006442 001124

```



```

10425 034066 012737 000100 001124      MOV    #IR,$GDDAT      ;GET GOOD = 100
10426 034074 053737 006442 001124      BIS    UNIT,$GDDAT     ;INCLUDE UNIT NUMBER
10427
10428 034102 017737 147762 001126      MOV    @RHCS2,$BDDAT   ;READ RHCS2 FOR COMPARISON
10429 034110 023737 001124 001126      CMP    $GDDAT,$BDDAT   ;COMPARE EXPECTED
10430                                     ;DATA WITH DATA READ FROM
10431                                     ;RHCS2
10432 034116 001401      BEQ    69$             ;BRANCH IF GOOD
10433 034120 104126      ERROR  126
10434                                     ;AFTER SETTING 'CLR' BIT #5
10435                                     ;IN RHCS2 TO INIT THE RH
10436                                     ;A ELEVEN WORD READ FROM AN
10437                                     ;ODD WORD BOUNDARY WAS DONE
10438                                     ;THEN
10439                                     ;RHCS2 SHOULD HAVE IR
10440                                     ;=100
10441                                     ;TOGETHER WITH UNIT NUMBER
10442                                     ;BUT CONTAINED WHAT IS
10443                                     ;GIVEN IN BAD RHCS2
10444 034122      69$:
10445                                     ;CHECK THAT RHCW HAS 0
10446 034122 012737 000000 001124      MOV    #0,$GDDAT      ;GET GOOD = 0
10447
10448 034130 017737 147726 001126      MOV    @RHCW,$BDDAT   ;READ RHCW FOR COMPARISON
10449 034136 023737 001124 001126      CMP    $GDDAT,$BDDAT   ;COMPARE EXPECTED
10450                                     ;DATA WITH DATA READ FROM
10451                                     ;RHCW
10452 034144 001401      BEQ    71$             ;BRANCH IF GOOD
10453 034146 104131      ERROR  131
10454                                     ;AFTER SETTING 'CLR' BIT #5
10455                                     ;IN RHCS2 TO INIT THE RH
10456                                     ;A ELEVEN WORD READ FROM AN
10457                                     ;ODD WORD BOUNDARY WAS DONE
10458                                     ;THEN
10459                                     ;RHCW SHOULD HAVE 0
10460                                     ;BUT CONTAINED WHAT IS
10461                                     ;GIVEN IN BAD RHCW
10462 034150      71$:
10463
10464
10465
10466
10467
10468
10469
10470
10471
10472
10473
10474
10475
10476
10477
10478 034150 000004      TST53: SCOPE
10479 034152 012706 001000      MOV    #STACK,SP      ;RESET STACK
10480 034156 012737 000053 006276      MOV    #53,@#TSTNM    ;SAVE TEST NUMBER

```

```

*****
;*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,RHCW
*****

```

```

10481
10482 034164 004737 041222 JSR PC,@#CLDISK ;GIVE RH INITIALIZE
10483 ;SETUP UNIT NUBER
10484 ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
10485
10486 ;SET UP FOR A TEN WORD READ FROM AN EVEN WORD BOUNDARY
10487 034170 012701 000003 MOV #3,R1 ;COUNT OF THREE
10488 034174 012702 003000 MOV #BFEVEN,R2 ;START THREE WORD BUFFER
10489 ;FROM AN EVEN WORD BOUNDARY
10490 034200 012722 052525 1$: MOV #52525,(R2)+ ;MOVE THREE 52525 INTO BUFFER
10491 034204 005301 DEC R1 ;COUNT
10492 034206 001374 BNE 1$ ;BRANCH IF THREE NOT DONE
10493 034210 012777 003000 147646 MOV #BFEVEN,@RHBA ;SET BUS ADDRESS TO START
10494 ;FROM AN EVEN WORD BOUNDARY
10495 034216 012777 177766 147636 MOV #-10,@RHWC ;WORD COUNT TEN
10496 034224 052777 000010 147636 BIS #BAI,@RHCS2 ;SET BAI IN RHCS2
10497 034232 004077 152022 JSR RO,@COMND ;GO TO DO COMMAND
10498 034236 004160 READAT ;READ DATA
10499
10500
10501 ;CHECK THAT RHCS3 HAS 0
10502 034240 012737 000000 001124 MOV #0,$GDDAT ;GET GOOD = 0
10503
10504 034246 017737 147660 001126 MOV @RHCS3,$BDDAT ;READ RHCS3 FOR COMPARISON
10505 034254 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
10506 ;DATA WITH DATA READ FROM
10507 ;RHCS3
10508 034262 001401 BEQ 65$ ;BRANCH IF GOOD
10509 034264 104124 ERROR 124
10510 ;AFTER SETTING "CLR" BIT #5
10511 ;IN RHCS2 TO INIT THE RH
10512 ;A TEN WORD READ FROM AN
10513 ;EVEN WORD BOUNDARY WAS DONE
10514 ;WITH BAI IN RHCS2 SET
10515 ;THEN
10516 ;RHCS3 SHOULD HAVE 0
10517 ;BUT CONTAINED WHAT IS
10518 ;GIVEN IN BAD RHCS3
10519 034266 042777 000076 147564 65$: BIC #75,@RHCS1 ;CLEAR FUNCTION BITS
10520 034266 ;CHECK THAT RHCS1 HAS RDY!DVA
10521 MOV #RDY!DVA,$GDDAT ;GET GOOD = 4200
10522 034274 012737 004200 001124
10523
10524 034302 017737 147552 001126 MOV @RHCS1,$BDDAT ;READ RHCS1 FOR COMPARISON
10525 034310 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
10526 ;DATA WITH DATA READ FROM
10527 ;RHCS1
10528 034316 001401 BEQ 67$ ;BRANCH IF GOOD
10529 034320 104125 ERROR 125
10530 ;AFTER SETTING "CLR" BIT #5
10531 ;IN RHCS2 TO INIT THE RH
10532 ;A TEN WORD READ FROM AN
10533 ;EVEN WORD BOUNDARY WAS DONE
10534 ;WITH BAI IN RHCS2 SET
10535 ;THEN
10536 ;RHCS1 SHOULD HAVE RDY!DVA

```



```
10537                                     :=4200
10538                                     ;BUT CONTAINED WHAT IS
10539                                     ;GIVEN IN BAD RHCS1
10540 034322                               67$:
10541                                     ;CHECK THAT RHCS2 HAS IR!BAI
10542 034322 012737 000110 001124         MOV #IR!BAI,$GDDAT ;GET GOOD = 110
10543 034330 053737 006442 001124         BIS UNIT,$GDDAT   ;INCLUDE UNIT NUMBER
10544
10545 034336 017737 147526 001126         MOV @RHCS2,$BDDAT ;READ RHCS2 FOR COMPARISON
10546 034344 023737 001124 001126         CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
10547                                     ;DATA WITH DATA READ FROM
10548                                     ;RHCS2
10549 034352 001401                         BEQ 69$           ;BRANCH IF GOOD
10550 034354 104126                         ERROR 126
10551                                     ;AFTER SETTING "CLR" BIT #5
10552                                     ;IN RHCS2 TO INIT THE RH
10553                                     ;A TEN WORD READ FROM AN
10554                                     ;EVEN WORD BOUNDARY WAS DONE
10555                                     ;WITH BAI IN RHCS2 SET
10556                                     ;THEN
10557                                     ;RHCS2 SHOULD HAVE IR!BAI
10558                                     ;=110
10559                                     ;TOGETHER WITH UNIT NUMBER
10560                                     ;BUT CONTAINED WHAT IS
10561                                     ;GIVEN IN BAD RHCS2
10562 034356                               69$:
10563                                     ;CHECK THAT RHWC HAS 0
10564 034356 012737 000000 001124         MOV #0,$GDDAT    ;GET GOOD = 0
10565
10566 034364 017737 147472 001126         MOV @RHWC,$BDDAT ;READ RHWC FOR COMPARISON
10567 034372 023737 001124 001126         CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
10568                                     ;DATA WITH DATA READ FROM
10569                                     ;RHWC
10570 034400 001401                         BEQ 71$           ;BRANCH IF GOOD
10571 034402 104131                         ERROR 131
10572                                     ;AFTER SETTING "CLR" BIT #5
10573                                     ;IN RHCS2 TO INIT THE RH
10574                                     ;A TEN WORD READ FROM AN
10575                                     ;EVEN WORD BOUNDARY WAS DONE
10576                                     ;WITH BAI IN RHCS2 SET
10577                                     ;THEN
10578                                     ;RHWC SHOULD HAVE 0
10579                                     ;BUT CONTAINED WHAT IS
10580                                     ;GIVEN IN BAD RHWC
10581 034404                               71$:
10582
10583
10584
10585
10586 *****
10587 ;*TEST 54 TEST DBL (RHCS3 BIT #10) TEST M
10588
10589 ;* CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
10590 ;* SET UP FOR A ONE WORD WRITE REVERSE FROM AN EVEN WORD BOUNDARY
10591 ;* DO A ONE WORD WRITE REVERSE FROM AN EVEN WORD BOUNDARY
10592 ;* THIS SHOULD NOT SET RHCS3 BIT #10 DBL
```

```
10593      :*      CHECK RHCS3
10594      :*      CHECK RHCS1,RHCS2,RHBA,RHBAE,RHWC
10595
10596      :*****
10597 034404 000004      TST54: SCOPE
10598 034406 012706 001000      MOV #STACK,SP      ;RESET STACK
10599 034412 012737 000054 006276      MOV #54,@TSTNM      ;SAVE TEST NUMBER
10600
10601 034420 004737 041222      JSR PC,@#CLDISK      ;GIVE RH INITIALIZE
10602      ;SETUP UNIT NUBER
10603      ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
10604
10605      ;SET UP FOR A ONE WORD WRITE REVERSE FROM AN EVEN WORD BOUNDARY
10606 034424 012701 000003      MOV #3,R1      ;COUNT OF THREE
10607 034430 012702 003000      MOV #BFEVEN,R2      ;START THREE WORD BUFFER
10608      ;FROM AN EVEN WORD BOUNDARY
10609 034434 012722 052525      1$: MOV #52525,(R2)+      ;MOVE THREE 52525 INTO BUFFER
10610 034440 005301      DEC R1      ;COUNT
10611 034442 001374      BNE 1$      ;BRANCH IF THREE NOT DONE
10612 034444 012777 003000 147412      MOV #BFEVEN,@RHBA      ;SET BUS ADDRESS TO START
10613      ;FROM AN EVEN WORD BOUNDARY
10614 034452 012777 177777 147402      MOV #-1,@RHWC ;WORD COUNT ONE
10615 034460 004077 151574      JSR RO,@COMND      ;GO TO DO COMMAND
10616 034464 004204      REVWRT      ;REVERSEWRITE DATA
10617
10618
10619      ;CHECK THAT RHCS3 HAS 0
10620 034466 012737 000000 001124      MOV #0,$GDDAT      ;GET GOOD = 0
10621
10622 034474 017737 147432 001126      MOV @RHCS3,$BDDAT      ;READ RHCS3 FOR COMPARISON
10623 034502 023737 001124 001126      CMP $GDDAT,$BDDAT      ;COMPARE EXPECTED
10624      ;DATA WITH DATA READ FROM
10625      ;RHCS3
10626 034510 001401      BEQ 65$      ;BRANCH IF GOOD
10627 034512 104124      ERROR 124
10628
10629      ;AFTER SETTING 'CLR' BIT #5
10630      ;IN RHCS2 TO INIT THE RH
10631      ;A ONE WORD WRITE REVERSE FROM AN
10632      ;EVEN WORD BOUNDARY WAS DONE
10633      ;THEN
10634      ;RHCS3 SHOULD HAVE 0
10635      ;BUT CONTAINED WHAT IS
10636      ;GIVEN IN BAD RHCS3
10636 034514      65$: BIC #76,@RHCS1      ;CLEAR FUNCTION BITS
10637 034514 042777 000076 147336      ;CHECK THAT RHCS1 HAS SC!RDY!DVA
10638      MOV #SC!RDY!DVA,$GDDAT      ;GET GOOD = 104200
10639 034522 012737 104200 001124
10640
10641 034530 017737 147324 001126      MOV @RHCS1,$BDDAT      ;READ RHCS1 FOR COMPARISON
10642 034536 023737 001124 001126      CMP $GDDAT,$BDDAT      ;COMPARE EXPECTED
10643      ;DATA WITH DATA READ FROM
10644      ;RHCS1
10645 034544 001401      BEQ 67$      ;BRANCH IF GOOD
10646 034546 104125      ERROR 125
10647      ;AFTER SETTING 'CLR' BIT #5
10648      ;IN RHCS2 TO INIT THE RH
```


10649 ;A ONE WORD WRITE REVERSE FROM AN
10650 ;EVEN WORD BOUNDARY WAS DONE
10651 ;THEN
10652 ;RHCS1 SHOULD HAVE SC!RDY!DVA
10653 ;=104200
10654 ;BUT CONTAINED WHAT IS
10655 ;GIVEN IN BAD RHCS1

10656 034550 67\$: ;CHECK THAT RHCS2 HAS IR!OR
10657 ;MOV #IR!OR,\$GDDAT ;GET GOOD = 300
10658 034550 012737 000300 001124 ;BIS UNIT,\$GDDAT ;INCLUDE UNIT NUMBER
10659 034556 053737 006442 001124
10660
10661 034564 017737 147300 001126 ;MOV @RHCS2,\$BDDAT ;READ RHCS2 FOR COMPARISON
10662 034572 023737 001124 001126 ;CMP \$GDDAT,\$BDDAT ;COMPARE EXPECTED
10663 ;DATA WITH DATA READ FROM
10664 ;RHCS2
10665 034600 001401 ;BEQ 69\$;BRANCH IF GOOD
10666 034602 104126 ;ERROR 126

10667 ;AFTER SETTING 'CLR' BIT #5
10668 ;IN RHCS2 TO INIT THE RH
10669 ;A ONE WORD WRITE REVERSE FROM AN
10670 ;EVEN WORD BOUNDARY WAS DONE
10671 ;THEN
10672 ;RHCS2 SHOULD HAVE IR!OR
10673 ;=300
10674 ;TOGETHER WITH UNIT NUMBER
10675 ;BUT CONTAINED WHAT IS
10676 ;GIVEN IN BAD RHCS2

10677 034604 69\$: ;CHECK THAT RHWC HAS 0
10678 ;MOV #0,\$GDDAT ;GET GOOD = 0
10679 034604 012737 000000 001124
10680
10681 034612 017737 147244 001126 ;MOV @RHWC,\$BDDAT ;READ RHWC FOR COMPARISON
10682 034620 023737 001124 001126 ;CMP \$GDDAT,\$BDDAT ;COMPARE EXPECTED
10683 ;DATA WITH DATA READ FROM
10684 ;RHWC
10685 034626 001401 ;BEQ 71\$;BRANCH IF GOOD
10686 034630 104131 ;ERROR 131

10687 ;AFTER SETTING 'CLR' BIT #5
10688 ;IN RHCS2 TO INIT THE RH
10689 ;A ONE WORD WRITE REVERSE FROM AN
10690 ;EVEN WORD BOUNDARY WAS DONE
10691 ;THEN
10692 ;RHWC SHOULD HAVE 0
10693 ;BUT CONTAINED WHAT IS
10694 ;GIVEN IN BAD RHWC

10695 034632 71\$:

10696
10697
10698
10699
10700 ;*****
10701 ;*TEST 55 TEST DBL (RHCS3 BIT #10) TEST N

10702 ;* CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
10703 ;* SET UP FOR A TWO WORD WRITE REVERSE FROM AN EVEN WORD BOUNDARY
10704 ;* DO A TWO WORD WRITE REVERSE FROM AN EVEN WORD BOUNDARY

```
10705 :* THIS SHOULD NOT SET RHCS3 BIT #10 DBL
10706 :* CHECK RHCS3
10707 :* CHECK RHCS1,RHCS2,RHBA,RHBAE,RHWC
10708
10709 :*****
10710 034632 000004 T55: SCOPE
10711 034634 012706 001000 MOV #STACK,SP ;RESET STACK
10712 034640 012737 000055 006276 MOV #55,@TSTNM ;SAVE TEST NUMBER
10713
10714 034646 004737 041222 JSR PC,@#CLDISK ;GIVE RH INITIALIZE
10715 ;SETUP UNIT NUBER
10716 ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
10717
10718 ;SET UP FOR A TWO WORD WRITE REVERSE FROM AN EVEN WORD BOUNDARY
10719 034652 012701 000003 MOV #3,R1 ;COUNT OF THREE
10720 034656 012702 003000 MOV #BFEVEN,R2 ;START THREE WORD BUFFER
10721 ;FROM AN EVEN WORD BOUNDARY
10722 034662 012722 052525 1$: MOV #52525,(R2)+ ;MOVE THREE 52525 INTO BUFFER
10723 034666 005301 DEC R1 ;COUNT
10724 034670 001374 BNE 1$ ;BRANCH IF THREE NOT DONE
10725 034672 012777 003000 147164 MOV #BFEVEN,@RHBA ;SET BUS ADDRESS TO START
10726 ;FROM AN EVEN WORD BOUNDARY
10727 034700 012777 177776 147154 MOV #-2,@RHWC ;WORD COUNT TWO
10728 034706 004077 151346 JSR RO,@COMND ;GO TO DO COMMAND
10729 034712 004204 REVWRT ;REVERSEWRITE DATA
10730
10731
10732 ;CHECK THAT RHCS3 HAS 0
10733 034714 012737 000000 001124 MOV #0,$GDDAT ;GET GOOD = 0
10734
10735 034722 017737 147204 001126 MOV @RHCS3,$BDDAT ;READ RHCS3 FOR COMPARISON
10736 034730 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
10737 ;DATA WITH DATA READ FROM
10738 ;RHCS3
10739 034736 001401 BEQ 65$ ;BRANCH IF GOOD
10740 034740 104124 ERROR 124
10741 ;AFTER SETTING 'CLR' BIT #5
10742 ;IN RHCS2 TO INIT THE RH
10743 ;A TWO WORD WRITE REVERSE FROM AN
10744 ;EVEN WORD BOUNDARY WAS DONE
10745 ;THEN
10746 ;RHCS3 SHOULD HAVE 0
10747 ;BUT CONTAINED WHAT IS
10748 ;GIVEN IN BAD RHCS3
10749 034742 042777 000076 147110 65$: BIC #76,@RHCS1 ;CLEAR FUNCTION BITS
10750 034742 ;CHECK THAT RHCS1 HAS SC!RDY!DVA
10751 MOV #SC!RDY!DVA,$GDDAT ;GET GOOD = 104200
10752 034750 012737 104200 001124
10753
10754 034756 017737 147076 001126 MOV @RHCS1,$BDDAT ;READ RHCS1 FOR COMPARISON
10755 034764 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
10756 ;DATA WITH DATA READ FROM
10757 ;RHCS1
10758 034772 001401 BEQ 67$ ;BRANCH IF GOOD
10759 034774 104125 ERROR 125
10760 ;AFTER SETTING 'CLR' BIT #5
```



```

10761 ;IN RHCS2 TO INIT THE RH
10762 ;A TWO WORD WRITE REVERSE FROM AN
10763 ;EVEN WORD BOUNDARY WAS DONE
10764 ;THEN
10765 ;RHCS1 SHOULD HAVE SC!RDY!DVA
10766 ;=104200
10767 ;BUT CONTAINED WHAT IS
10768 ;GIVEN IN BAD RHCS1

```

```

10769 034776 67$: ;CHECK THAT RHCS2 HAS IR!OR
10770 ;MOV #IR!OR,$GDDAT ;GET GOOD = 300
10771 034776 012737 000300 001124 ;BIS UNIT,$GDDAT ;INCLUDE UNIT NUMBER
10772 035004 053737 006442 001124
10773
10774 035012 017737 147052 001126 ;MOV @RHCS2,$BDDAT ;READ RHCS2 FOR COMPARISON
10775 035020 023737 001124 001126 ;CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
10776 ;DATA WITH DATA READ FROM
10777 ;RHCS2
10778 035026 001401 ;BEQ 69$ ;BRANCH IF GOOD
10779 035030 104126 ;ERROR 126

```

```

10780 ;AFTER SETTING 'CLR' BIT #5
10781 ;IN RHCS2 TO INIT THE RH
10782 ;A TWO WORD WRITE REVERSE FROM AN
10783 ;EVEN WORD BOUNDARY WAS DONE
10784 ;THEN
10785 ;RHCS2 SHOULD HAVE IR!OR
10786 ;=300
10787 ;TOGETHER WITH UNIT NUMBER
10788 ;BUT CONTAINED WHAT IS
10789 ;GIVEN IN BAD RHCS2

```

```

10790 035032 69$: ;CHECK THAT RHWC HAS 0
10791 ;MOV #0,$GDDAT ;GET GOOD = 0
10792 035032 012737 000000 001124
10793
10794 035040 017737 147016 001126 ;MOV @RHWC,$BDDAT ;READ RHWC FOR COMPARISON
10795 035046 023737 001124 001126 ;CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
10796 ;DATA WITH DATA READ FROM
10797 ;RHWC
10798 035054 001401 ;BEQ 71$ ;BRANCH IF GOOD
10799 035056 104131 ;ERROR 131

```

```

10800 ;AFTER SETTING 'CLR' BIT #5
10801 ;IN RHCS2 TO INIT THE RH
10802 ;A TWO WORD WRITE REVERSE FROM AN
10803 ;EVEN WORD BOUNDARY WAS DONE
10804 ;THEN
10805 ;RHWC SHOULD HAVE 0
10806 ;BUT CONTAINED WHAT IS
10807 ;GIVEN IN BAD RHWC

```

```

10808 035060 71$:
10809
10810
10811
10812 ;*****
10813 ;*TEST 56 TEST DBL (RHCS3 BIT #10) TEST 0
10814
10815 ;* CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
10816 ;* SET UP FOR A TWO WORD WRITE REVERSE FROM AN ODD WORD BOUNDARY

```

```

10817      ;*      DO A TWO WORD WRITE REVERSE FROM AN ODD WORD BOUNDARY
10818      ;*      THIS SHOULD SET RHCS3 BIT #10 DBL
10819      ;*      CHECK RHCS3
10820      ;*      CHECK RHCS1,RHCS2,RHBA,RHBAE,RHWC
10821
10822      ;*****
10823      TST56: SCOPE
10824      MOV      #STACK,SP      ;RESET STACK
10825      MOV      #56,@#TSTNM    ;SAVE TEST NUMBER
10826
10827      JSR      PC,@#CLDISK    ;GIVE RH INITIALIZE
10828      ;SETUP UNIT NUBER
10829      ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
10830
10831      ;SET UP FOR A TWO WORD WRITE REVERSE FROM AN ODD WORD BOUNDARY
10832      MOV      #3,R1          ;COUNT OF THREE
10833      MOV      #BFODD,R2      ;START THREE WORD BUFFER
10834      ;FROM AN ODD WORD BOUNDARY
10835      1$:      MOV      #52525,(R2)+ ;MOVE THREE 52525 INTO BUFFER
10836      DEC      R1              ;COUNT
10837      BNE     1$              ;BRANCH IF THREE NOT DONE
10838      MOV      #BFODD,@RHBA    ;SET BUS ADDRESS TO START
10839      ;FROM AN ODD WORD BOUNDARY
10840      MOV      #-2,@RHWC ;WORD COUNT TWO
10841      JSR      RO,@COMND      ;GO TO DO COMMAND
10842      REVWRT ;REVERSEWRITE DATA
10843
10844
10845      ;CHECK THAT RHCS3 HAS DBL
10846      MOV      #DBL,$GDDAT    ;GET GOOD = 2000
10847
10848      MOV      @RHCS3,$BDDAT  ;READ RHCS3 FOR COMPARISON
10849      CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED
10850      ;DATA WITH DATA READ FROM
10851      ;RHCS3
10852      BEQ     65$            ;BRANCH IF GOOD
10853      ERROR   124
10854
10855      ;AFTER SETTING 'CLR' BIT #5
10856      ;IN RHCS2 TO INIT THE RH
10857      ;A TWO WORD WRITE REVERSE FROM AN
10858      ;ODD WORD BOUNDARY WAS DONE
10859      ;THEN
10860      ;RHCS3 SHOULD HAVE DBL
10861      ;=2000
10862      ;BUT CONTAINED WHAT IS
10863      ;GIVEN IN BAD RHCS3
10864      65$:      BIC      #76,@RHCS1 ;CLEAR FUNCTION BITS
10865      ;CHECK THAT RHCS1 HAS SC!RDY!DVA
10866      MOV      #SC!RDY!DVA,$GDDAT ;GET GOOD = 104200
10867
10868      MOV      @RHCS1,$BDDAT  ;READ RHCS1 FOR COMPARISON
10869      CMP      $GDDAT,$BDDAT ;COMPARE EXPECTED
10870      ;DATA WITH DATA READ FROM
10871      ;RHCS1
10872      BEQ     67$            ;BRANCH IF GOOD

```



```

10873 035222 104125          ERROR 125
10874
10875
10876
10877
10878
10879
10880
10881
10882
10883 035224
10884
10885 035224 012737 000300 001124
10886 035232 053737 006442 001124
10887
10888 035240 017737 146624 001126
10889 035246 023737 001124 001126
10890
10891
10892 035254 001401
10893 035256 104126
10894
10895
10896
10897
10898
10899
10900
10901
10902
10903
10904 035260
10905
10906 035260 012737 000000 001124
10907
10908 035266 017737 146570 001126
10909 035274 023737 001124 001126
10910
10911
10912 035302 001401
10913 035304 104131
10914
10915
10916
10917
10918
10919
10920
10921
10922 035306
10923
10924
10925
10926
10927
10928

```

67\$:

```

:CHECK THAT RHCS2 HAS IR!OR
MOV #IR!OR,$GDDAT :GET GOOD = 300
BIS UNIT,$GDDAT :INCLUDE UNIT NUMBER
MOV @RHCS2,$BDDAT :READ RHCS2 FOR COMPARISON
CMP $GDDAT,$BDDAT :COMPARE EXPECTED
:DATA WITH DATA READ FROM
:RHCS2
BEQ 69$ :BRANCH IF GOOD
ERROR 126

```

```

:AFTER SETTING 'CLR' BIT #5
:IN RHCS2 TO INIT THE RH
:A TWO WORD WRITE REVERSE FROM AN
:ODD WORD BOUNDARY WAS DONE
:THEN
:RHCS1 SHOULD HAVE SC!RDY!DVA
:=104200
:BUT CONTAINED WHAT IS
:GIVEN IN BAD RHCS1

```

```

:AFTER SETTING 'CLR' BIT #5
:IN RHCS2 TO INIT THE RH
:A TWO WORD WRITE REVERSE FROM AN
:ODD WORD BOUNDARY WAS DONE
:THEN
:RHCS2 SHOULD HAVE IR!OR
:=300
:TOGETHER WITH UNIT NUMBER
:BUT CONTAINED WHAT IS
:GIVEN IN BAD RHCS2

```

69\$:

```

:CHECK THAT RHCW HAS 0
MOV #0,$GDDAT :GET GOOD = 0
MOV @RHCW,$BDDAT :READ RHCW FOR COMPARISON
CMP $GDDAT,$BDDAT :COMPARE EXPECTED
:DATA WITH DATA READ FROM
:RHCW
BEQ 71$ :BRANCH IF GOOD
ERROR 131

```

```

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

```

71\$:

```

*****
:*TEST 57 TEST DBL (RHCS3 BIT #10) TEST P

```

```

10929      : * CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
10930      : * SET UP FOR A THREE WORD WRITE REVERSE FROM AN EVEN WORD BOUNDARY
10931      : * DO A THREE WORD WRITE REVERSE FROM AN EVEN WORD BOUNDARY
10932      : * THIS SHOULD SET RHCS3 BIT #10 DBL
10933      : * CHECK RHCS3
10934      : * CHECK RHCS1,RHCS2,RHBA,RHBAE,RHWC

```

```

10935
10936
10937      035306 000004
10938      035310 012706 001000
10939      035314 012737 000057 006276
10940
10941      035322 004737 041222
10942
10943
10944
10945      : *****
10946      : TST57: SCOPE
10947      : MOV #STACK,SP ;RESET STACK
10948      : MOV #57,@#TSTNM ;SAVE TEST NUMBER
10949
10950      JSR PC,@#CLDISK ;GIVE RH INITIALIZE
10951      ;SETUP UNIT NUBER
10952      ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
10953
10954      :SET UP FOR A THREE WORD WRITE REVERSE FROM AN EVEN WORD BOUNDARY
10955      MOV #3,R1 ;COUNT OF THREE
10956      MOV #BFEBVEN,R2 ;START THREE WORD BUFFER
10957      ;FROM AN EVEN WORD BOUNDARY
10958      1$: MOV #52525,(R2)+ ;MOVE THREE 52525 INTO BUFFER
10959      DEC R1 ;COUNT
10960      BNE 1$ ;BRANCH IF THREE NOT DONE
10961      MOV #BFEBVEN,@RHBA ;SET BUS ADDRESS TO START
10962      ;FROM AN EVEN WORD BOUNDARY
10963      MOV #-3,@RHWC ;WORD COUNT THREE
10964      JSR RO,@COMND ;GO TO DO COMMAND
10965      REVWRT ;REVERSEWRITE DATA
10966
10967
10968      :CHECK THAT RHCS3 HAS DBL
10969      MOV #DBL,$GDDAT ;GET GOOD = 2000
10970
10971      MOV @RHCS3,$BDDAT ;READ RHCS3 FOR COMPARISON
10972      CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
10973      ;DATA WITH DATA READ FROM
10974      ;RHCS3
10975      BEQ 65$ ;BRANCH IF GOOD
10976      ERROR 124
10977
10978      :AFTER SETTING "CLR" BIT #5
10979      :IN RHCS2 TO INIT THE RH
10980      :A THREE WORD WRITE REVERSE FROM AN
10981      :EVEN WORD BOUNDARY WAS DONE
10982      :THEN
10983      :RHCS3 SHOULD HAVE DBL
10984      :=2000
10985      :BUT CONTAINED WHAT IS
10986      :GIVEN IN BAD RHCS3
10987
10988      65$: BIC #76,@RHCS1 ;CLEAR FUNCTION BITS
10989      :CHECK THAT RHCS1 HAS SC!RDY!DVA
10990      MOV #SC!RDY!DVA,$GDDAT ;GET GOOD = 104200
10991
10992      MOV @RHCS1,$BDDAT ;READ RHCS1 FOR COMPARISON
10993      CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
10994      ;DATA WITH DATA READ FROM

```


10985
10986
10987
10988
10989
10990
10991
10992
10993
10994
10995
10996
10997
10998
10999
11000
11001
11002
11003
11004
11005
11006
11007
11008
11009
11010
11011
11012
11013
11014
11015
11016
11017
11018
11019
11020
11021
11022
11023
11024
11025
11026
11027
11028
11029
11030
11031
11032
11033
11034
11035
11036
11037
11038
11039
11040

035446 001401
035450 104125

035452
035452 012737 000300 001124
035' 0 053737 006442 001124
035466 017737 146376 001126
035474 023737 001124 001126

035502 001401
035504 104126

035506
035506 012737 000000 001124
035514 017737 146342 001126
035522 023737 001124 001126

035530 001401
035532 104131

035534

BEQ 67\$
ERROR 125

67\$:
:CHECK THAT RHCS2 HAS IR!OR
MOV #IR!OR,\$GDDAT
BIS UNIT,\$GDDAT

MOV @RHCS2,\$BDDAT
CMP \$GDDAT,\$BDDAT

BEQ 69\$
ERROR 126

69\$:
:CHECK THAT RHC HAS 0
MOV #0,\$GDDAT

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

BEQ 71\$
ERROR 131

:RHCS1
:BRANCH IF GOOD

:AFTER SETTING 'CLR' BIT #5
:IN RHCS2 TO INIT THE RH
:A THREE WORD WRITE REVERSE FROM AN
:EVEN WORD BOUNDARY WAS DONE
:THEN
:RHCS1 SHOULD HAVE SC!RDY!DVA
:=104200
:BUT CONTAINED WHAT IS
:GIVEN IN BAD RHCS1

:IR!OR
:GET GOOD = 300
:INCLUDE UNIT NUMBER

: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
:A THREE WORD WRITE REVERSE FROM AN
:EVEN WORD BOUNDARY WAS DONE
:THEN
:RHCS2 SHOULD HAVE IR!OR
:=300
:TOGETHER WITH UNIT NUMBER
:BUT CONTAINED WHAT IS
:GIVEN IN BAD RHCS2

: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
:A THREE WORD WRITE REVERSE FROM AN
:EVEN WORD BOUNDARY WAS DONE
:THEN
:RHC SHOULD HAVE 0
:BUT CONTAINED WHAT IS
:GIVEN IN BAD RHC

::*****

```

11041      :*TEST 60      TEST DBL (RHCS3 BIT #10) TEST Q
11042
11043      :*          CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
11044      :*          SET UP FOR A THREE WORD WRITE REVERSE FROM AN ODD WORD BOUNDARY
11045      :*          DO A THREE WORD WRITE REVERSE FROM AN ODD WORD BOUNDARY
11046      :*          THIS SHOULD NOT SET RHCS3 BIT #10 DBL
11047      :*          CHECK RHCS3
11048      :*          CHECK RHCS1,RHCS2,RHBA,RHBAE,RHWC
11049
11050      :*****
11051      TST60: SCOPE
11052      MOV      #STACK,SP      ;RESET STACK
11053      MOV      #60,@#TSTNM    ;SAVE TEST NUMBER
11054
11055      JSR      PC,@#CLDISK    ;GIVE RH INITIALIZE
11056      ;SETUP UNIT NUBER
11057      ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
11058
11059      ;SET UP FOR A THREE WORD WRITE REVERSE FROM AN ODD WORD BOUNDARY
11060      MOV      #3,R1          ;COUNT OF THREE
11061      MOV      #BFODD,R2      ;START THREE WORD BUFFER
11062      ;FROM AN ODD WORD BOUNDARY
11063      1$: MOV      #52525,(R2)+ ;MOVE THREE 52525 INTO BUFFER
11064      DEC      R1              ;COUNT
11065      BNE     1$              ;BRANCH IF THREE NOT DONE
11066      MOV      #BFODD,@RHBA    ;SET BUS ADDRESS TO START
11067      ;FROM AN ODD WORD BOUNDARY
11068      MOV      #-3,@RHWC ;WORD COUNT THREE
11069      JSR      R0,@COMND      ;GO TO DO COMMAND
11070      REVWRT ;REVERSEWRITE DATA
11071
11072
11073      ;CHECK THAT RHCS3 HAS 0
11074      MOV      #0,$GDDAT      ;GET GOOD = 0
11075
11076      MOV      @RHCS3,$BDDAT  ;READ RHCS3 FOR COMPARISON
11077      CMP     $GDDAT,$BDDAT  ;COMPARE EXPECTED
11078      ;DATA WITH DATA READ FROM
11079      ;RHCS3
11080      BEQ     65$            ;BRANCH IF GOOD
11081      ERROR  124
11082      ;AFTER SETTING "CLR" BIT #5
11083      ;IN RHCS2 TO INIT THE RH
11084      ;A THREE WORD WRITE REVERSE FROM AN
11085      ;ODD WORD BOUNDARY WAS DONE
11086      ;THEN
11087      ;RHCS3 SHOULD HAVE 0
11088      ;BUT CONTAINED WHAT IS
11089      ;GIVEN IN BAD RHCS3
11090      65$: BIC     #76,@RHCS1  ;CLEAR FUNCTION BITS
11091      ;CHECK THAT RHCS1 HAS SC!RDY!DVA
11092      MOV      #SC!RDY!DVA,$GDDAT ;GET GOOD = 104200
11093      035652 012737 104200 001124
11094
11095      MOV      @RHCS1,$BDDAT  ;READ RHCS1 FOR COMPARISON
11096      CMP     $GDDAT,$BDDAT  ;COMPARE EXPECTED

```


11153
11154
11155
11156
11157
11158
11159
11160
11161
11162
11163
11164
11165
11166
11167
11168
11169
11170
11171
11172
11173
11174
11175
11176
11177
11178
11179
11180
11181
11182
11183
11184
11185
11186
11187
11188
11189
11190
11191
11192
11193
11194
11195
11196
11197
11198
11199
11200
11201
11202
11203
11204
11205
11206
11207
11208

035762 000004
035764 012706 001000
035770 012737 000061 006276
035776 004737 041222
036002 012701 000003
036006 012702 003000
036012 012722 052525
036016 005301
036020 001374
036022 012777 003000 146034
036030 012777 177776 146024
036036 052777 000010 146024
036044 004077 150210
036050 004204
036052 012737 000000 001124
036060 017737 146046 001126
036066 023737 001124 001126
036074 001401
036076 104124
036100
036100 042777 000076 145752

```
*****
*TEST 61 TEST DBL (RHCS3 BIT #10) TEST R
*
* CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
* SET UP FOR A TWO WORD WRITE REVERSE FROM AN EVEN WORD BOUNDARY
* WITH BAI IN RHCS2 BIT #3 SET
* 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
*****
TST61: SCOPE
MOV #STACK,SP ;RESET STACK
MOV #61,@#TSTNM ;SAVE TEST NUMBER
JSR PC,@#CLDISK ;GIVE RH INITIALIZE
;SETUP UNIT NUBER
;CLEAR RHWC AND FUNCTION BITS IN RHCS1
;SET UP FOR A TWO WORD WRITE REVERSE FROM AN EVEN WORD BOUNDARY
MOV #3,R1 ;COUNT OF THREE
MOV #BFEBVEN,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 #BFEBVEN,@RHBA ;SET BUS ADDRESS TO START
;FROM AN EVEN WORD BOUNDARY
MOV #-2,@RHWC ;WORD COUNT TWO
BIS #BAI,@RHCS2 ;SET BAI IN RHCS2
JSR RO,@COMND ;GO TO DO COMMAND
REVVRT ;REVERSEWRITE 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 TWO WORD WRITE REVERSE 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$: BIC #76,@RHCS1 ;CLEAR FUNCTION BITS
;CHECK THAT RHCS1 HAS SC!RDY!DVA
```



```

11209 036106 012737 104200 001124      MOV      #SC!RDY!DVA,$GDDAT      ;GET GOOD = 104200
11210
11211 036114 017737 145740 001126      MOV      @RHCS1,$BDDAT      ;READ RHCS1 FOR COMPARISON
11212 036122 023737 001124 001126      CMP      $GDDAT,$BDDAT      ;COMPARE EXPECTED
11213                                     ;DATA WITH DATA READ FROM
11214                                     ;RHCS1
11215 036130 001401      BEQ      67$      ;BRANCH IF GOOD
11216 036132 104125      ERROR    125
11217                                     ;AFTER SETTING 'CLR' BIT #5
11218                                     ;IN RHCS2 TO INIT THE RH
11219                                     ;A TWO WORD WRITE REVERSE FROM AN
11220                                     ;EVEN WORD BOUNDARY WAS DONE
11221                                     ;WITH BAI IN RHCS2 SET
11222                                     ;THEN
11223                                     ;RHCS1 SHOULD HAVE SC!RDY!DVA
11224                                     ;=104200
11225                                     ;BUT CONTAINED WHAT IS
11226                                     ;GIVEN IN BAD RHCS1
11227 036134                                     67$:
11228                                     ;CHECK THAT RHCS2 HAS IR!OR!BAI
11229 036134 012737 000310 001124      MOV      #IR!OR!BAI,$GDDAT      ;GET GOOD = 310
11230 036142 053737 006442 001124      BIS      UNIT,$GDDAT      ;INCLUDE UNIT NUMBER
11231
11232 036150 017737 145714 001126      MOV      @RHCS2,$BDDAT      ;READ RHCS2 FOR COMPARISON
11233 036156 023737 001124 001126      CMP      $GDDAT,$BDDAT      ;COMPARE EXPECTED
11234                                     ;DATA WITH DATA READ FROM
11235                                     ;RHCS2
11236 036164 001401      BEQ      69$      ;BRANCH IF GOOD
11237 036166 104126      ERROR    126
11238                                     ;AFTER SETTING 'CLR' BIT #5
11239                                     ;IN RHCS2 TO INIT THE RH
11240                                     ;A TWO WORD WRITE REVERSE FROM AN
11241                                     ;EVEN WORD BOUNDARY WAS DONE
11242                                     ;WITH BAI IN RHCS2 SET
11243                                     ;THEN
11244                                     ;RHCS2 SHOULD HAVE IR!OR!BAI
11245                                     ;=310
11246                                     ;TOGETHER WITH UNIT NUMBER
11247                                     ;BUT CONTAINED WHAT IS
11248                                     ;GIVEN IN BAD RHCS2
11249 036170                                     69$:
11250                                     ;CHECK THAT RHWC HAS 0
11251 036170 012737 000000 001124      MOV      #0,$GDDAT      ;GET GOOD = 0
11252
11253 036176 017737 145660 001126      MOV      @RHWC,$BDDAT      ;READ RHWC FOR COMPARISON
11254 036204 023737 001124 001126      CMP      $GDDAT,$BDDAT      ;COMPARE EXPECTED
11255                                     ;DATA WITH DATA READ FROM
11256                                     ;RHWC
11257 036212 001401      BEQ      71$      ;BRANCH IF GOOD
11258 036214 104131      ERROR    131
11259                                     ;AFTER SETTING 'CLR' BIT #5
11260                                     ;IN RHCS2 TO INIT THE RH
11261                                     ;A TWO WORD WRITE REVERSE FROM AN
11262                                     ;EVEN WORD BOUNDARY WAS DONE
11263                                     ;WITH BAI IN RHCS2 SET
11264                                     ;THEN

```

:RHWC SHOULD HAVE 0
:BUT CONTAINED WHAT IS
:GIVEN IN BAD RHWC

11265
11266
11267
11268 036216 71\$:

:TEST 62 TEST DBL (RHCS3 BIT #10) TEST S

:* CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
:* SET UP FOR A NINE WORD READ REVERSE FROM AN EVEN WORD BOUNDARY
:* 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

TST62: SCOPE

11285 036216 000004
11286 036220 012706 001000
11287 036224 012737 000062 006276

MOV #STACK,SP ;RESET STACK
MOV #62,@#TSTNM ;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 TST63 ;BRANCH IF TU NOT THERE

11289 036232 004737 041222
11290
11291
11292 036236 022737 042634 006260
11293

:SET UP FOR A NINE 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 #-9.,@RHWC ;WORD COUNT NINE

11294 036244 001103
11295
11296

11298 036246 012701 000003
11299 036252 012702 003000
11300

JSR R0,@COMND ;GO TO DO COMMAND
REVRED ;REVERSE READ

11301 036256 012722 052525
11302 036262 005301
11303 036264 001374
11304 036266 012777 003000 145570

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

11305
11306 036274 012777 177767 145560
11307
11308 036302 004077 147752
11309 036306 004200

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

11310
11311
11312 036310 012737 002000 001124
11313

ERROR 124 ;AFTER SETTING 'CLR' BIT #5

11314 036316 017737 145610 001126
11315 036324 023737 001124 001126
11316
11317

11318 036332 001401
11319 036334 104124
11320


```

11321                                     ;IN RHCS2 TO INIT THE RH
11322                                     ;A NINE WORD READ REVERSE FROM AN
11323                                     ;EVEN WORD BOUNDARY WAS DONE
11324                                     ;THEN
11325                                     ;RHCS3 SHOULD HAVE DBL
11326                                     ;=2000
11327                                     ;BUT CONTAINED WHAT IS
11328                                     ;GIVEN IN BAD RHCS3
11329 036336                               65$: BIC #76,@RHCS1 ;CLEAR FUNCTION BITS
11330 036336 042777 000076 145514 ;CHECK THAT RHCS1 HAS RDY!DVA
11331                                     MOV #RDY!DVA,$GDDAT ;GET GOOD = 4200
11332 036344 012737 004200 001124
11333                                     MOV @RHCS1,$BDDAT ;READ RHCS1 FOR COMPARISON
11334 036352 017737 145502 001126                                     CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
11335 036360 023737 001124 001126 ;DATA WITH DATA READ FROM
11336                                     ;RHCS1
11337                                     BEQ 67$ ;BRANCH IF GOOD
11338 036366 001401
11339 036370 104125
11340                                     ;AFTER SETTING 'CLR' BIT #5
11341                                     ;IN RHCS2 TO INIT THE RH
11342                                     ;A NINE WORD READ REVERSE FROM AN
11343                                     ;EVEN WORD BOUNDARY WAS DONE
11344                                     ;THEN
11345                                     ;RHCS1 SHOULD HAVE RDY!DVA
11346                                     ;=4200
11347                                     ;BUT CONTAINED WHAT IS
11348                                     ;GIVEN IN BAD RHCS1
11349 036372                               67$: ;CHECK THAT RHCS2 HAS IR
11350                                     MOV #IR,$GDDAT ;GET GOOD = 100
11351 036372 012737 000100 001124                                     BIS UNIT,$GDDAT ;INCLUDE UNIT NUMBER
11352 036400 053737 006442 001124
11353                                     MOV @RHCS2,$BDDAT ;READ RHCS2 FOR COMPARISON
11354 036406 017737 145456 001126                                     CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
11355 036414 023737 001124 001126 ;DATA WITH DATA READ FROM
11356                                     ;RHCS2
11357                                     BEQ 69$ ;BRANCH IF GOOD
11358 036422 001401
11359 036424 104126
11360                                     ;AFTER SETTING 'CLR' BIT #5
11361                                     ;IN RHCS2 TO INIT THE RH
11362                                     ;A NINE WORD READ REVERSE FROM AN
11363                                     ;EVEN WORD BOUNDARY WAS DONE
11364                                     ;THEN
11365                                     ;RHCS2 SHOULD HAVE IR
11366                                     ;=100
11367                                     ;TOGETHER WITH UNIT NUMBER
11368                                     ;BUT CONTAINED WHAT IS
11369                                     ;GIVEN IN BAD RHCS2
11370 036426                               69$: ;CHECK THAT RHWC HAS 0
11371                                     MOV #0,$GDDAT ;GET GOOD = 0
11372 036426 012737 000000 001124
11373                                     MOV @RHWC,$BDDAT ;READ RHWC FOR COMPARISON
11374 036434 017737 145422 001126                                     CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
11375 036442 023737 001124 001126 ;DATA WITH DATA READ FROM
11376

```

```

11377
11378 036450 001401
11379 036452 104131
11380
11381
11382
11383
11384
11385
11386
11387
11388 036454
11389
11390
11391
11392
11393
11394
11395
11396
11397
11398
11399
11400
11401
11402
11403
11404 036454 000004
11405 036456 012706 001000
11406 036462 012737 000063 006276
11407
11408 036470 004737 041222
11409
11410
11411 036474 022737 042634 006260
11412
11413 036502 001103
11414
11415
11416
11417 036504 012701 000003
11418 036510 012702 003000
11419
11420 036514 012722 052525
11421 036520 005301
11422 036522 001374
11423 036524 012777 003000 145332
11424
11425 036532 012777 177766 145322
11426
11427 036540 004077 147514
11428 036544 004200
11429
11430
11431 036546 012737 000000 001124
11432

```

```

:RHC
:BRANCH IF GOOD
BEQ 71$
ERROR 131

```

```

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

```

71\$:

```

:*****
:*TEST 63 TEST DBL (RHCS3 BIT #10) TEST T

```

```

:* 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,RHC
:* IF MAG. TAPE NOT USED.THIS TEST IS NOT DONE

```

```

:*****
TST63: SCOPE

```

```

MOV #STACK,SP ;RESET STACK
MOV #63,@TSTNM ;SAVE TEST NUMBER
JSR PC,@CLDISK ;GIVE RH INITIALIZE
;SETUP UNIT NUBER
;CLEAR RHC AND FUNCTION BITS IN RHCS1
CMP #CMNDTM,COMND ;IS TU THERE
BNE TST64 ;BRANCH IF TU NOT THERE

```

```

;SET UP FOR A TEN 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
;MOVE THREE 52525 INTO BUFFER
1$: MOV #52525,(R2)+
DEC R1 ;COUNT
BNE 1$ ;BRANCH IF THREE NOT DONE
MOV #BFEVEN,@RHBA ;SET BUS ADDRESS TO START
;FROM AN EVEN WORD BOUNDARY

```

```

MOV #-10.,@RHC ;WORD COUNT TEN
JSR R0,@COMND ;GO TO DO COMMAND
REVRED ;REVERSE READ

```

```

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

```



```

11433 036554 017737 145352 001126      MOV    @RHCS3,$BDDAT  ;READ RHCS3 FOR COMPARISON
11434 036562 023737 001124 001126      CMP    $GDDAT,$BDDAT  ;COMPARE EXPECTED
11435                                     ;DATA WITH DATA READ FROM
11436                                     ;RHCS3
11437 036570 001401      BEQ    65$           ;BRANCH IF GOOD
11438 036572 104124      ERROR  124
11439                                     ;AFTER SETTING 'CLR' BIT #5
11440                                     ;IN RHCS2 TO INIT THE RH
11441                                     ;A TEN WORD READ REVERSE FROM AN
11442                                     ;EVEN WORD BOUNDARY WAS DONE
11443                                     ;THEN
11444                                     ;RHCS3 SHOULD HAVE 0
11445                                     ;BUT CONTAINED WHAT IS
11446                                     ;GIVEN IN BAD RHCS3
11447 036574                                     65$:
11448 036574 042777 000076 145256      BIC    #76,@RHCS1    ;CLEAR FUNCTION BITS
11449                                     ;CHECK THAT RHCS1 HAS RDY!DVA
11450 036602 012737 004200 001124      MOV    #RDY!DVA,$GDDAT ;GET GOOD = 4200
11451
11452 036610 017737 145244 001126      MOV    @RHCS1,$BDDAT  ;READ RHCS1 FOR COMPARISON
11453 036616 023737 001124 001126      CMP    $GDDAT,$BDDAT  ;COMPARE EXPECTED
11454                                     ;DATA WITH DATA READ FROM
11455                                     ;RHCS1
11456 036624 001401      BEQ    67$           ;BRANCH IF GOOD
11457 036626 104125      ERROR  125
11458                                     ;AFTER SETTING 'CLR' BIT #5
11459                                     ;IN RHCS2 TO INIT THE RH
11460                                     ;A TEN WORD READ REVERSE FROM AN
11461                                     ;EVEN WORD BOUNDARY WAS DONE
11462                                     ;THEN
11463                                     ;RHCS1 SHOULD HAVE RDY!DVA
11464                                     ;=4200
11465                                     ;BUT CONTAINED WHAT IS
11466                                     ;GIVEN IN BAD RHCS1
11467 036630                                     67$:
11468                                     ;CHECK THAT RHCS2 HAS IR
11469 036630 012737 000100 001124      MOV    #IR,$GDDAT    ;GET GOOD = 100
11470 036636 053737 006442 001124      BIS    UNIT,$GDDAT   ;INCLUDE UNIT NUMBER
11471
11472 036644 017737 145220 001126      MOV    @RHCS2,$BDDAT  ;READ RHCS2 FOR COMPARISON
11473 036652 023737 001124 001126      CMP    $GDDAT,$BDDAT  ;COMPARE EXPECTED
11474                                     ;DATA WITH DATA READ FROM
11475                                     ;RHCS2
11476 036660 001401      BEQ    69$           ;BRANCH IF GOOD
11477 036662 104126      ERROR  126
11478                                     ;AFTER SETTING 'CLR' BIT #5
11479                                     ;IN RHCS2 TO INIT THE RH
11480                                     ;A TEN WORD READ REVERSE FROM AN
11481                                     ;EVEN WORD BOUNDARY WAS DONE
11482                                     ;THEN
11483                                     ;RHCS2 SHOULD HAVE IR
11484                                     ;=100
11485                                     ;TOGETHER WITH UNIT NUMBER
11486                                     ;BUT CONTAINED WHAT IS
11487                                     ;GIVEN IN BAD RHCS2
11488 036664                                     69$:
    
```

```

11489 ;CHECK THAT RHWC HAS 0
11490 036664 012737 000000 001124 MOV #0,$GDDAT ;GET GOOD = 0
11491
11492 036672 017737 145164 001126 MOV @RHWC,$BDDAT ;READ RHWC FOR COMPARISON
11493 036700 023737 001124 001126 CMP $GDDAT,$BDDAT ;COMPARE EXPECTED
11494 ;DATA WITH DATA READ FROM
11495 ;RHWC
11496 036706 001401 BEQ 71$ ;BRANCH IF GOOD
11497 036710 104131 ERROR 131
11498 ;AFTER SETTING "CLR" BIT #5
11499 ;IN RHCS2 TO INIT THE RH
11500 ;A TEN WORD READ REVERSE FROM AN
11501 ;EVEN WORD BOUNDARY WAS DONE
11502 ;THEN
11503 ;RHWC SHOULD HAVE 0
11504 ;BUT CONTAINED WHAT IS
11505 ;GIVEN IN BAD RHWC
11506 036712 71$:
11507
11508
11509
11510 *****
11511 *TEST 64 TEST DBL (RHCS3 BIT #10) TEST U
11512
11513 ;* CLEAR THE SUBSYSTEM BY A CONTROLLER CLEAR (RHCS2 BIT #5)
11514 ;* SET UP FOR A TEN WORD READ REVERSE FROM AN ODD WORD BOUNDARY
11515 ;* DO A TEN WORD READ REVERSE FROM AN ODD WORD BOUNDARY
11516 ;* THIS SHOULD SET RHCS3 BIT #10 DBL
11517 ;* CHECK RHCS3
11518 ;* CHECK RHCS1,RHCS2,RHBA,RHBAE,RHWC
11519 ;* IF MAG. TAPE NOT USED.THIS TEST IS NOT DONE
11520
11521 *****
11522 036712 000004 TST64: SCOPE
11523 036714 012706 001000 MOV #STACK,SP ;RESET STACK
11524 036720 012737 000064 006276 MOV #64,@TSTNM ;SAVE TEST NUMBER
11525
11526 036726 004737 041222 JSR PC,@#CLDISK ;GIVE RH INITIALIZE
11527 ;SETUP UNIT NUBER
11528 ;CLEAR RHWC AND FUNCTION BITS IN RHCS1
11529 036732 022737 042634 006260 CMP #CMNDTM,COMND ;IS TU THERE
11530
11531 036740 001103 BNE TST65 ;BRANCH IF TU NOT THERE
11532
11533
11534 ;SET UP FOR A TEN WORD READ REVERSE FROM AN ODD WORD BOUNDARY
11535 036742 012701 000003 MOV #3,R1 ;COUNT OF THREE
11536 036746 012702 003002 MOV #BFODD,R2 ;START THREE WORD BUFFER
11537 ;FROM AN ODD WORD BOUNDARY
11538 036752 012722 052525 1$: MOV #52525,(R2)+ ;MOVE THREE 52525 INTO BUFFER
11539 036756 005301 DEC R1 ;COUNT
11540 036760 001374 BNE 1$ ;BRANCH IF THREE NOT DONE
11541 036762 012777 003002 145074 MOV #BFODD,@RHBA ;SET BUS ADDRESS TO START
11542 ;FROM AN ODD WORD BOUNDARY
11543 036770 012777 177766 145064 MOV #-10.,@RHWC ;WORD COUNT TEN
11544

```


CERHAEO MACY11 30A(1052) 02-MAY-79 14:35 PAGE 224
 CERHAE.P11 02-MAY-79 14:08 T64

TEST DBL (RHCS3 BIT #10) TEST U

SEQ 0222

11545	036776	004077	147256		JSR	RO,@COMND		:GO TO DO COMMAND
11546	037002	004200			REVRED			:REVERSE READ
11547								
11548								:CHECK THAT RHCS3 HAS DBL
11549	037004	012737	002000	001124	MOV	#DBL,\$GDDAT		:GET GOOD = 2000
11550								
11551	037012	017737	145114	001126	MOV	@RHCS3,\$BDDAT		:READ RHCS3 FOR COMPARISON
11552	037020	023737	001124	001126	CMP	\$GDDAT,\$BDDAT		:COMPARE EXPECTED
11553								:DATA WITH DATA READ FROM
11554								:RHCS3
11555	037026	001401			BEQ	65\$:BRANCH IF GOOD
11556	037030	104124			ERROR	124		
11557								:AFTER SETTING 'CLR' BIT #5
11558								:IN RHCS2 TO INIT THE RH
11559								:A TEN WORD READ REVERSE FROM AN
11560								:ODD WORD BOUNDARY WAS DONE
11561								:THEN
11562								:RHCS3 SHOULD HAVE DBL
11563								: =2000
11564								:BUT CONTAINED WHAT IS
11565								:GIVEN IN BAD RHCS3
11566	037032						65\$:	
11567	037032	042777	000076	145020	BIC	#76,@RHCS1		:CLEAR FUNCTION BITS
11568								:CHECK THAT RHCS1 HAS RDY!DVA
11569	037040	012737	004200	001124	MOV	#RDY!DVA,\$GDDAT		:GET GOOD = 4200
11570								
11571	037046	017737	145006	001126	MOV	@RHCS1,\$BDDAT		:READ RHCS1 FOR COMPARISON
11572	037054	023737	001124	001126	CMP	\$GDDAT,\$BDDAT		:COMPARE EXPECTED
11573								:DATA WITH DATA READ FROM
11574								:RHCS1
11575	037062	001401			BEQ	67\$:BRANCH IF GOOD
11576	037064	104125			ERROR	125		
11577								:AFTER SETTING 'CLR' BIT #5
11578								:IN RHCS2 TO INIT THE RH
11579								:A TEN WORD READ REVERSE FROM AN
11580								:ODD WORD BOUNDARY WAS DONE
11581								:THEN
11582								:RHCS1 SHOULD HAVE RDY!DVA
11583								: =4200
11584								:BUT CONTAINED WHAT IS
11585								:GIVEN IN BAD RHCS1
11586	037066						67\$:	
11587								:CHECK THAT RHCS2 HAS IR
11588	037066	012737	000100	001124	MOV	#IR,\$GDDAT		:GET GOOD = 100
11589	037074	053737	006442	001124	BIS	UNIT,\$GDDAT		:INCLUDE UNIT NUMBER
11590								
11591	037102	017737	144762	001126	MOV	@RHCS2,\$BDDAT		:READ RHCS2 FOR COMPARISON
11592	037110	023737	001124	001126	CMP	\$GDDAT,\$BDDAT		:COMPARE EXPECTED
11593								:DATA WITH DATA READ FROM
11594								:RHCS2
11595	037116	001401			BEQ	69\$:BRANCH IF GOOD
11596	037120	104126			ERROR	126		
11597								:AFTER SETTING 'CLR' BIT #5
11598								:IN RHCS2 TO INIT THE RH
11599								:A TEN WORD READ REVERSE FROM AN
11600								:ODD WORD BOUNDARY WAS DONE

```

11601                                     :THEN
11602                                     :RHCS2 SHOULD HAVE IR
11603                                     : =100
11604                                     :TOGETHER WITH UNIT NUMBER
11605                                     :BUT CONTAINED WHAT IS
11606                                     :GIVEN IN BAD RHCS2
11607 037122                               69$:
11608                                     :CHECK THAT RHCW HAS 0
11609 037122 012737 000000 001124         MOV    #0,$GDDAT      ;GET GOOD = 0
11610
11611 037130 017737 144726 001126         MOV    @RHCW,$BDDAT  ;READ RHCW FOR COMPARISON
11612 037136 023737 001124 001126         CMP    $GDDAT,$BDDAT ;COMPARE EXPECTED
11613                                     :DATA WITH DATA READ FROM
11614                                     :RHCW
11615 037144 001401                         BEQ    71$           ;BRANCH IF GOOD
11616 037146 104131                         ERROR  131
11617
11618                                     :AFTER SETTING "CLR" BIT #5
11619                                     :IN RHCS2 TO INIT THE RH
11620                                     :A TEN WORD READ REVERSE FROM AN
11621                                     :ODD WORD BOUNDARY WAS DONE
11622                                     :THEN
11623                                     :RHCW SHOULD HAVE 0
11624                                     :BUT CONTAINED WHAT IS
11625 037150                               71$:
11626
11627
11628
11629
11630
11631
11632
11633
11634
11635
11636
11637
11638
11639
11640

```

 :*TEST 65 TEST DBL (RHCS3 BIT #10) TEST V

```

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

```

		TST65:														
		SCOPE	#STACK	SP	:RESET STACK											
		MOV	#65,	@#TSTNM	:SAVE T	8531	8600	8764	8833	8997	9066	922				
		MOV	#65,	@#TSTNM	:SAVE T	8531	8600	8764	8833	8997	9066	922				
96803	037150	000004			10940	11054	11168	11288	11407	11525	11644					
	037152	001000			10940	11054	11168	11288	11407	11525	11644					
	097986	099737	000065	006276	10481	10600	10713	10826	10940	11054	11168	11288	11407	11525	11644	
	10026	10140	10253	10366	10481	10600	10713	10826	10940	11054	11168	11288	11407	11525	11644	
	11763	11882														
SAVTST	1283#															
SCOPE	1301#	3254	3655	3793	3875	3955	4037	4117	4200	4287	4476	4652	4922	5176	5422	
	5668	5908	6156	6427	6602	6722	6842	6963	7070	7215	7451	7688	7990	8295	8528	
	8761	8994	9223	9336	9450	9563	9677	9795	9910	10023	10137	10250	10363	10478	10597	
	10710	10823	10937	11051	11165	11285	11404	11522	11641	11760	11879	12000	12048			
SETPRI	1#	1406#	12979													
SB47RA	13488#	13469	13470	13471	13472	13475	13476									
SETUP	1#	1406#	3139													
SILO	1283#	5021	5251	5498	5744	5987										
SIZDEV	1283#															
SKIP	1#	1283#	1406#	11293	11412	11530	11649	11768	11887							
SLASH	1#	1406#														
SMORE	1283#	12703														
SPACE	1406#															
STARS	1#	1406#	1461	1465	1520	2561	2567	2619	2621	2820	3229	3253	3650	3654	3777	

SSDESCA	1#	1406#													
SSNEWT	1#	1406#	3229	3650	3777	3860	3939	4022	4101	4184	4271	4460	4631	4903	5157
	5402	5648	5888	6137	6413	6587	6707	6827	6947	7046	7202	7438	7674	7976	8279
45	8978	0212	9325	9439	9552	9665	9784	9899	10012	10126	10239	10352	10466		
	10586	10699	10812	10926	11040	11153	11273	11392	11510	11629	11748	11866	11993		
SSSET	13460#	13469	13470	13471	13472	13475	13476	13477	13478						
SSSKIP	1#	1406#													
.EQUAT	1#	1273#	1296												
.HEADE	1#	1273#													
.KT11	1#	1273#	1426												
.SETUP	1#	1273#	3139												
.SWRHI	1#	1273#	1283												
.SWRLO	1273#	1294#													
.SACT1	1#														
.SAPT8	1#														
.SAPTH	1#														
.SAPTY	1#														
.SASTA	1#														
.SCATC	1#	1273#	1407												
.SCMTA	1#	1273#	1463												
.SDB2D	1#														
.SDB20	1#														
.SDIV	1#														
.SEOP	1#	1273#	12039												
.SERRO	1#	1273#	13087												
.SERRT	1273#														
.SMULT	1#														
.SPOWE	1#	1273#	13482												
.SRAND	1#														
.SRDDE	1#														
.SRDOC	1#	1273#	13034												
.SREAD	1#	1273#	12890												
.SR2AZ	1#														
.SSAVE	1#														
.SSB2D	1#														
.SSB20	1#														
.SSCOP	1#	1273#	12688												
.SSIZE	1#														
.SSUPR	1#														
.STRAP	1#	1273#	13437												
.STYPB	1#														
.STYPD	1#	1273#	12753												
.STYPE	1#	1273#	12820												
.STYPO	1#	1273#	13360												
.S4OCA	1#														
.1170	1#														

. ABS. 070060 000

ERRORS DETECTED: 0

CERHAE.BIN,CERHAE.SEQ/CRF/SOL/NL:TOC=CERHAE.SML,CERHAE.P11
RUN-TIME: 104 151 9 SECONDS
RUN-TIME RATIO: 842/266=3.1

CERMAE0 MACY11 30A(1052) 02-MAY-79 14:35 PAG
CERMAE.P11 02-MAY-79 14:08 CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0315

CORE USED: 37K (73 PAGES)

CERHAEO MACY11 30A(1052) 02-MAY-79 14:35 PAGE 317
CERHAE.P11 02-MAY-79 14:08 CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0313

8298 8367