

.REM 8

IDENTIFICATION

PRODUCT CODE: AC 1853B MC
PRODUCT NAME: CORDABO RDJ11 B CLUSTER DIAG
PRODUCT DATE: APRIL, 1984
MAINTAINER: DIAGNOSTIC ENGINEERING

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

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

COPYRIGHT (C) 1984 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL	DEC	DECUS	ENTRUS	MASSBUS
DEC			DECTAPE	

43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73

TABLE OF CONTENTS

1. PROGRAM ABSTRACT
2. SYSTEM REQUIREMENTS
3. LOADING AND STARTING PROCEDURES
4. SPECIAL ENVIRONMENTS
5. PROGRAM OPTIONS
6. EXECUTION TIMES
7. ERROR INFORMATION
8. EXAMPLES
9. PROGRAM DESCRIPTION
 - 9.1 J11 CODE
 - 9.2 CACHE CODE
 - 9.3 ON BOARD ROM CODE
 - 9.4 LINE TIME CLOCK CODE
 - 9.5 SERIAL LINE UNIT CODE
 - 9.6 Q22BE CODE

D1

SEQ 0003

94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129

1. PROGRAM ABSTRACT

THIS PROGRAM TESTS OUT KDJ11-B CPU BOARD, INCLUDING THE J11 CHIP SET, ON BOARD CACHE, ON BOARD ROM'S, SERIAL LINE UNIT, AND LINE TIME CLOCKS.

THE KDJ11-B IS A PDP-11 CPU THAT INCORPORATES THE J11 CHIP SET AS THE HEART OF THE PROCESSOR. IT IS A QUAD HEIGHT Q22 BUS MODULE. IT HAS ON BOARD CACHE, SOME OF THE FUNCTIONALITY OF THE CACHE IS HIDDEN INSIDE THE J11 AND THE REST OF THE FUNCTIONS IMPLEMENTED IN TWO ON BOARD GATE ARRAYS. THE STORAGE CAPACITY OF THE CACHE IS 4K BYTES OF RAM, CALLED DATA RAM'S. THE CACHE IS IMPLEMENTED AS A DIRECT MAPPED CACHE WITH ADDRESS BITS 21 THROUGH 13 STORED IN A DIFFERENT SET OF RAM'S CALLED TAG STORE.

THE KDJ11-B ALSO HAS TWO ON-BOARD ROM'S. ONE OF THEM, THE 16-BIT ADDRESSABLE ROM, CONTAINS THE SELF TEST AND THE BOOT CODES. THE OTHER ROM, THE 8-BIT ADDRESSABLE ONE, CONTAINS THE BASE AREA WITH HARDWARE SELECTION PARAMETERS, OPTIONAL BOOTSTRAPS, OPTIONAL UFD (USER FRIENDLY DIAGNOSTIC) SYSTEM DESCRIPTION AREA, AND OPTIONAL FOREIGN LANGUAGE TEXT.

THE SERIAL LINE UNIT IS IMPLEMENTED THRU A DLART CHIP WHICH PROVIDES THE STANDARD CONSOLE INTERFACE TO THE CPU. IT HAS INTERNAL LOOP BACK MODE AND PROVIDES WITH THREE CLOCK LINES: 800HZ, 60HZ, AND 50HZ.

THE LINE TIME CLOCK FUNCTIONS ARE IMPLEMENTED USING THOSE THREE LINES AND THE BEVENT LINE. THE LINE CLOCK STATUS REGISTER IS IMPLEMENTED IN ONE OF THE GATE ARRAYS.

2. SYSTEM REQUIREMENTS

HARDWARE REQUIREMENTS

TO RUN SUCCESSFULLY THE DIAGNOSTIC NEEDS:

1. KDJ11-B CPU MODULE
2. CONSOLE TERMINAL
3. AT LEAST 28K OF MEMORY

IN DVT, AND STAGE ONE MANUFACTURING (MODULE ASSEMBLY) THE Q22 BUS EXERCISER IS NEEDED TO CHECK Q22 BUS LOGIC.

3. LOADING AND STARTING PROCEDURES

TO START UP THIS PROGRAM:

1. BOOT XXDP.
2. TYPE PR NAME, WHERE NAME IS THE NAME OF THE BIN OR BIC FILE FOR THIS PROGRAM.

THE STARTING ADDRESS OF THE PROGRAM IS 200.
 NOTE: IF TRYING TO RESTART THE PROGRAM IN AN ARBITRARY PLACE AFTER HALT OR BREAK THE FOLLOWING REGISTERS SHOULD BE SET UP:
 1777572*0 TO DISABLE MEMMORY MANAGEMENT

130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185

17777520-1000 TO CLEAR DIAGNOSTIC MODE (BIT 8), BUT STILL SAVE
 HALT ON BREAK
 1777746-400 TO FLUSH THE CACHE

4. SPECIAL ENVIRONMENTS

THE PROGRAM IS APT COMPATIBLE. IT CAN ALSO BE RUN UNDER THE UFD MONITOR.
 IN THOSE CASES NONE OF THE STANDARD ERROR PRINTOUTS OCCUR. REFER
 TO CORRESPONDING DOCUMENTS ON RUNNING PROCEDURES IN APT AND UNDER
 UFD MONITOR.

5. PROGRAM OPTIONS

THE Q22 BUS EXERCISER IS UTILIZED IF IT IS PRESENT IN THE SYSTEM AND
 THE DIAGNOSTIC IS NOT RUNNING IN UFD MODE.
 STANDARD CAPABILITIES OF LOOPING ON TEST AND ON ERROR ARE PROVIDED.
 IN ORDER TO RUN THE EXTENSIVE CACHE DATA RAM TEST BIT 7 HAS TO BE SET
 IN THE SOFTWARE SWITCH REGISTER. TO RUN THE EXTENSIVE CACHE TAG RAM TEST
 BIT 6 HAS TO BE SET IN THE SOFTWARE SWITCH REGISTER.

SWITCH REGISTER SELECTION:

BIT NUMBER	USE
15	HALT ON ERROR
14	LOOP ON PRESENT TEST
13	INHIBIT ERROR TYPEOUTS
11	INHIBIT ITERATIONS
10	BELL ON ERROR
9	LOOP ON ERROR
8	LOOP ON TEST IN SWRCS OF
7	DO EXTENSIVE DATA RAM TEST
6	DO EXTENSIVE TAG RAM TEST

6. EXECUTION TIMES

WITHOUT EXTENSIVE RAM TESTS, THE DIAGNOSTIC RUNS IN UNDER 15 SECONDS.
 WITH THEM, IT TAKES ABOUT 2 MINUTE.

7. ERROR INFORMATION

IN THE CASE OF ERRORS, A FAILING PC AND TEST NUMBERS ARE GIVEN.
 WHERE IT IS POSSIBLE, EXPECTED AND RECEIVED DATA ARE GIVEN.
 FOR AN EXAMPLE, SEE SECTION 8.

8. EXAMPLES

AFTER BOOTING KXDP, AND STARTING THE PROGRAM, THE FOLLOWING
 WILL APPEAR ON THE TERMINAL:

KDJ11 B CPU DIAGNOSTIC

SWR XXXXXX NEW

WHERE XXXXXX CORRESPOND TO PRESENT SOFTWARE SWITCH REGISTER

186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236

SETTING.
 AFTER "NEW" AN OPERATOR CAN DO ONE OF THE FOLLOWING:
 1) TYPE IN A NEW SOFTWARE SWITCH REGISTER SETTING FOLLOWED
 BY CARRIAGE RETURN OR
 2) JUST TYPE IN CARRIAGE RETURN IN WHICH CASE THE SOFTWARE
 REGISTER WILL REMAIN UNCHANGED.

EXAMPLE OF ERROR PRINTOUT:
 ERROR IN TAG STORE

TEST	ERROR	ADDRESS	ADDRESS
0	PC	<1-16>	<15-0>
27	105620	66600	000000

NOTE: THIS MAY NOT CORRESPOND TO THE ACTUAL PROGRAM COUNTER.

9. PROGRAM DESCRIPTION

9.1 J11 CODE

THIS PORTION OF THE CODE TESTS OUT THE J11 CHIP SET. IT IS BROKEN
 INTO 3 PIECES: CPU TESTS, WHICH VERIFY DIFFERENT INSTRUCTIONS IN
 DIFFERENT MODES AND DIFFERENT TRAP CONDITIONS; MMU TESTS, WHICH
 VERIFY DIFFERENT FUNCTIONS OF MMU; AND FFP TESTS, WHICH DO
 DIFFERENT FLOATING POINT INSTRUCTIONS.
 THIS PORTION OF THE CODE HAVE BEEN WRITTEN IN CLOSE RELATIONSHIP
 WITH THE J11 MICROCODE. THEREFORE, EVEN THOUGH NOT ALL POSSIBLE
 INSTRUCTIONS IN ALL POSSIBLE ADDRESSING MODE HAVE BEEN TESTED, AN
 ATTEMPT HAS BEEN MADE TO EXERCISE ALL OF THE MICROCODE.

9.2 CACHE CODE

THIS PORTION OF THE DIAGNOSTIC VERIFIES ALL CACHE FUNCTIONS.
 DATA AND TAG RAM'S TESTS ARE ALSO INCLUDED.

 NOTE: IN ORDER TO RUN EXTENSIVE CACHE RAM'S TESTS THE CORRESPONDING
 BITS IN THE SOFTWARE SWITCH REGISTER SHOULD BE SET. SEE SECTION 5.

9.3 ON BOARD ROM'S CODE

THESE TESTS VERIFY THE CHECKSUMS OF THE 16-BIT ROM AND THE BASE
 AREA OF THE 8 BIT ROM.

9.4 LINE TIME CLOCKS CODE

THIS PART OF THE PROGRAM VERIFIES THE FUNCTIONS OF ALL 4 CLOCK
 LINES: 3 OF THEM FROM THE SERIAL LINE CHIP (50HZ, 60HZ, 800HZ)
 AND BEVENT LINE.

NOTE: IN FPD MODE ONLY FUNCTIONS CORRESPONDING TO BOOT ROM
 SELECTION ARE CHECKED.

9.5 SERIAL LINE UNIT CODE

THESE TESTS VERIFY THE FUNCTIONALITY OF THE SLC CHIP UTILISING THE
 MAINTENANCE MODE OF THE CHIP.

CORDARO KDJ11 B COUNTER MAC11 30010461 05 APR 84 16:45 PAGE 7
CORDAR.P11 05 APR 84 16:45

(5]

SEQ 0006

237
238
239
240
241
242
243

9.6 Q2PBE CODE
THESE TESTS VERIFY INTERRUPT ARBITRATION OF THE KDJ11 B, DMA
PROTOCOL, AND CACHE FUNCTIONS RELATED TO THE DMA ACTIVITIES,
INCLUDING PMG COUNTER.

&

H I

SEQ 0007

```

244          167400          $SWR=167400
245          000300          $SWRMK=300
246
247          .TITLE COKDABO KDJ11 B CLUSTER
248          ;*COPYRIGHT (C) DEC 83
249          ;*DIGITAL EQUIPMENT CORP.
250          ;*MAYNARD, MASS. 01754
251          ;*
252          ;*PROGRAM BY DIAG. ENG.
253          ;*
254          ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP 11 MAINDEC SYSMAC
255          ;*PACKAGE (MAINDEC-11-DZQAC-C8), OCT, 1982.
256          ;*
257          $IN 1
258          .SBTTL OPERATIONAL SWITCH SETTINGS
259          ;*
260          ;*          SWITCH          USE
261          ;*          -----
262          ;*          15          HALT ON ERROR
263          ;*          14          LOOP ON TEST
264          ;*          13          INHIBIT ERROR TIMEOUTS
265          ;*          11          INHIBIT ITERATIONS
266          ;*          10          BELL ON ERROR
267          ;*           9          LOOP ON ERROR
268          ;*           8          LOOP ON TEST IN SWR<5:0>
269          ;*           7          DO EXTENSIVE DATA RAM TEST
270          ;*           6          DO EXTENSIVE TAG RAM TEST
271          .SBTTL MEMORY MANAGEMENT DEFINITIONS
272          ;*
273          ;*KT11 VECTOR ADDRESS
274          ;*
275          MMVEC= 250
276          ;*
277          ;*KT11 STATUS REGISTER ADDRESSES
278          ;*
279          SR0= 177572
280          SR1= 177574
281          SR2= 177576
282          SR3= 172516
283          ;*
284          ;*USER "I" PAGE DESCRIPTOR REGISTERS
285          ;*
286          UIPDR0= 177600
287          UIPDR1= 177602
288          UIPDR2= 177604
289          UIPDR3= 177606
290          UIPDR4= 177610
291          UIPDR5= 177612
292          UIPDR6= 177614
293          UIPDR7= 177616
294          ;*
295          ;*USER "D" PAGE DESCRIPTOR REGISTERS
296          ;*
297          UDPDR0= 177620
298          UDPDR1= 177622
299          UDPDR2= 177624
300          UDPDR3= 177626

```

300	177630	UDPDR4 = 177630
301	177632	UDPDR5 = 177632
302	177634	UDPDR6 = 177634
303	177636	UDPDR7 = 177636
304		
305		; *USER "I" PAGE ADDRESS REGISTERS
306		
307	177640	UIPAR0 = 177640
308	177642	UIPAR1 = 177642
309	177644	UIPAR2 = 177644
310	177646	UIPAR3 = 177646
311	177650	UIPAR4 = 177650
312	177652	UIPAR5 = 177652
313	177654	UIPAR6 = 177654
314	177656	UIPAR7 = 177656
315		
316		; *USER "D" PAGE ADDRESS REGISTERS
317		
318	177660	UDPAR0 = 177660
319	177662	UDPAR1 = 177662
320	177664	UDPAR2 = 177664
321	177666	UDPAR3 = 177666
322	177670	UDPAR4 = 177670
323	177672	UDPAR5 = 177672
324	177674	UDPAR6 = 177674
325	177676	UDPAR7 = 177676
326		
327		; *SUPERVISOR "I" PAGE DESCRIPTOR REGISTERS
328		
329	172200	SIPDR0 = 172200
330	172202	SIPDR1 = 172202
331	172204	SIPDR2 = 172204
332	172206	SIPDR3 = 172206
333	172210	SIPDR4 = 172210
334	172212	SIPDR5 = 172212
335	172214	SIPDR6 = 172214
336	172216	SIPDR7 = 172216
337		
338		; *SUPERVISOR "D" PAGE DESCRIPTOR REGISTERS
339		
340	172220	SDPDR0 = 172220
341	172222	SDPDR1 = 172222
342	172224	SDPDR2 = 172224
343	172226	SDPDR3 = 172226
344	172230	SDPDR4 = 172230
345	172232	SDPDR5 = 172232
346	172234	SDPDR6 = 172234
347	172236	SDPDR7 = 172236
348		
349		; *SUPERVISOR "I" PAGE ADDRESS REGISTERS
350		
351	172240	SIPAR0 = 172240
352	172242	SIPAR1 = 172242
353	172244	SIPAR2 = 172244
354	172246	SIPAR3 = 172246
355	172250	SIPAR4 = 172250

356	172252	SIPAR5 = 172252
357	172254	SIPAR6 = 172254
358	172256	SIPAR7 = 172256
359		
360		; *SUPERVISOR "D" PAGE ADDRESS REGISTERS
361		
362	172260	SDPAR0 = 172260
363	172262	SDPAR1 = 172262
364	172264	SDPAR2 = 172264
365	172266	SDPAR3 = 172266
366	172270	SDPAR4 = 172270
367	172272	SDPAR5 = 172272
368	172274	SDPAR6 = 172274
369	172276	SDPAR7 = 172276
370		
371		; *KERNEL "I" PAGE DESCRIPTOR REGISTERS
372		
373	172300	KIPDR0 = 172300
374	172302	KIPDR1 = 172302
375	172304	KIPDR2 = 172304
376	172306	KIPDR3 = 172306
377	172310	KIPDR4 = 172310
378	172312	KIPDR5 = 172312
379	172314	KIPDR6 = 172314
380	172316	KIPDR7 = 172316
381		
382		; *KERNEL "D" PAGE DESCRIPTOR REGISTERS
383		
384	172320	KDPDR0 = 172320
385	172322	KDPDR1 = 172322
386	172324	KDPDR2 = 172324
387	172326	KDPDR3 = 172326
388	172330	KDPDR4 = 172330
389	172332	KDPDR5 = 172332
390	172334	KDPDR6 = 172334
391	172336	KDPDR7 = 172336
392		
393		; *KERNEL "I" PAGE ADDRESS REGISTERS
394		
395	172340	KIPAR0 = 172340
396	172342	KIPAR1 = 172342
397	172344	KIPAR2 = 172344
398	172346	KIPAR3 = 172346
399	172350	KIPAR4 = 172350
400	172352	KIPAR5 = 172352
401	172354	KIPAR6 = 172354
402	172356	KIPAR7 = 172356
403		
404		; *KERNEL "D" PAGE ADDRESS REGISTERS
405		
406	172360	KDPAR0 = 172360
407	172362	KDPAR1 = 172362
408	172364	KDPAR2 = 172364
409	172366	KDPAR3 = 172366
410	172370	KDPAR4 = 172370
411	172372	KDPAR5 = 172372

412	172374	KDPAF6= 172374
413	172376	KDPAF7= 172376
414		
415		.SBTTL BASIC DEFINITIONS
416		
417		;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
418	001100	STACK= 1100
419	104000	ERROR= EMI ;:BASIC DEFINITION OF ERROR CALL
420	000004	SCOPE= IOT ;:BASIC DEFINITION OF SCOPE CALL
421		
422		;*MISCELLANEOUS DEFINITIONS
423	000011	HT= 11 ;:CODE FOR HORIZONTAL TAB
424	000012	LF= 12 ;:CODE FOR LINE FEED
425	000015	CR= 15 ;:CODE FOR CARRIAGE RETURN
426	000200	CRLF= 200 ;:CODE FOR CARRIAGE RETURN-LINE FEED
427	177776	PS= 177776 ;:PROCESSOR STATUS WORD
428	177776	PSW= PS
429	177774	STKLMT= 177774 ;:STACK LIMIT REGISTER
430	177772	PIRQ= 177772 ;:PROGRAM INTERRUPT REQUEST REGISTER
431	177570	DSWR= 177570 ;:HARDWARE SWITCH REGISTER
432	177570	DDISP= 177570 ;:HARDWARE DISPLAY REGISTER
433		
434		;*GENERAL PURPOSE REGISTER DEFINITIONS
435	000000	R0= *0 ;:GENERAL REGISTER
436	000001	R1= *1 ;:GENERAL REGISTER
437	000002	R2= *2 ;:GENERAL REGISTER
438	000003	R3= *3 ;:GENERAL REGISTER
439	000004	R4= *4 ;:GENERAL REGISTER
440	000005	R5= *5 ;:GENERAL REGISTER
441	000006	R6= *6 ;:GENERAL REGISTER
442	000007	R7= *7 ;:GENERAL REGISTER
443	000006	SP= *6 ;:STACK POINTER
444	000007	PC= *7 ;:PROGRAM COUNTER
445		
446		;*PRIORITY LEVEL DEFINITIONS
447	000000	PR0= 0 ;:PRIORITY LEVEL 0
448	000040	PR1= 40 ;:PRIORITY LEVEL 1
449	000100	PR2= 100 ;:PRIORITY LEVEL 2
450	000140	PR3= 140 ;:PRIORITY LEVEL 3
451	000200	PR4= 200 ;:PRIORITY LEVEL 4
452	000240	PR5= 240 ;:PRIORITY LEVEL 5
453	000300	PR6= 300 ;:PRIORITY LEVEL 6
454	000340	PR7= 340 ;:PRIORITY LEVEL 7
455		
456		;*SWITCH REGISTER SWITCH DEFINITIONS
457	100000	SW15= 100000
458	040000	SW14= 40000
459	020000	SW13= 20000
460	010000	SW12= 10000
461	004000	SW11= 4000
462	002000	SW10= 2000
463	001000	SW09= 1000
464	000400	SW08= 400
465	000200	SW07= 200
466	000100	SW06= 100
467	000040	SW05= 40

468	000020	SW04=	20
469	000010	SW03=	10
470	000004	SW02=	4
471	000002	SW01=	2
472	000001	SW00=	1
473	001000	SW9=	SW09
474	000400	SW8=	SW08
475	000200	SW7=	SW07
476	000100	SW6=	SW06
477	000040	SW5=	SW05
478	000020	SW4=	SW04
479	000010	SW3=	SW03
480	000004	SW2=	SW02
481	000002	SW1=	SW01
482	000001	SW0=	SW00

483		; *DATA BIT DEFINITIONS (BIT00 TO BIT15)	
484		BIT15=	100000
485	100000	BIT14=	40000
486	040000	BIT13=	20000
487	020000	BIT12=	10000
488	010000	BIT11=	4000
489	004000	BIT10=	2000
490	002000	BIT09=	1000
491	001000	BIT08=	400
492	000400	BIT07=	200
493	000200	BIT06=	100
494	000100	BIT05=	40
495	000040	BIT04=	20
496	000020	BIT03=	10
497	000010	BIT02=	4
498	000004	BIT01=	2
499	000002	BIT00=	1
500	000001	BIT9=	BIT09
501	001000	BIT8=	BIT08
502	000400	BIT7=	BIT07
503	000200	BIT6=	BIT06
504	000100	BIT5=	BIT05
505	000040	BIT4=	BIT04
506	000020	BIT3=	BIT03
507	000010	BIT2=	BIT02
508	000004	BIT1=	BIT01
509	000002	BIT0=	BIT00
510	000001		

511		; *BASIC "CPU" TRAP VECTOR ADDRESSES	
512		ERRVEC=	4
513	000004	RESVEC=	10
514	000010	IBITVEC=	14
515	000014	IRIVEC=	14
516	000014	BPTVEC=	14
517	000014	IOTVEC=	20
518	000020	PWRVEC=	24
519	000024	EMIVEC=	30
520	000030	TRAPVEC=	34
521	000034	IKVEC=	40
522	000040	IPVEC=	64
523	000064		

;; TIME OUT AND OTHER ERRORS
 ;; RESERVED AND ILLEGAL INSTRUCTIONS
 ;; "I" BIT
 ;; TRACE TRAP
 ;; BREAKPOINT TRAP (BPT)
 ;; INPUT/OUTPUT TRAP (IOT) **SCOPE**
 ;; POWER FAIL
 ;; EMULATOR TRAP (EMT) **ERROR**
 ;; TRAP- TRAP
 ;; ITY KEYBOARD VECTOR
 ;; ITY PRINTER VECTOR

```

524          000240      PIRGVEC=240          ;;PROGRAM INTERRUPT REQUEST VECTOR
525          000001      UFDSET=          1          ;;FLAG FOR UFD
526          .SBTTL TRAP CATCHER
527
528          000000      .+0
529          ;;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
530          ;;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
531          ;;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
532          000174      .+174
533 000174 000000      DISPREG: .WORD 0          ;;SOFTWARE DISPLAY REGISTER
534 000176 000000      SWREG: .WORD 0          ;;SOFTWARE SWITCH REGISTER
535          000200      .+200
536 000200 005037 001160      CLR      $TMP0
537 000204 000137 003764      JMP      @#START
538          000220      .+220
539 000220 012737 000777 001160      MOV      @777,$TMP0
540 000226 000137 003764      JMP      @#START
541
542          .SBTTL ACT11 HOOKS
543
544          ;;*****
545          ;;HOOKS REQUIRED BY ACT11
546          000232      $SVPC=.          ;;SAVE PC
547          000046      .+46
548 000046 133610      $ENDAD          ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .SEOP
549          000052      .+52
550 000052 000000      .WORD 0          ;;2)SET LOC.52 TO ZERO
551          000232      .+$SVPC          ;; RESTORE PC
552          .SBTTL APT PARAMETER BLOCK
553
554          ;;*****
555          ;;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
556          ;;*****
557          000232      . $X=.          ;;SAVE CURRENT LOCATION
558          000024      .+24          ;;SET POWER FAIL TO POINT TO START OF PROGRAM
559 000024 000200      200          ;;FOR APT START UP
560          000044      .+44          ;;POINT TO APT INDIRECT ADDRESS PTR.
561 000044 000232      $APTHDR          ;;POINT TO APT HEADER BLOCK
562          000232      .+ $X          ;;RESET LOCATION COUNTER
563          ;;*****
564          ;;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
565          ;;INTERFACE SPEC.
566
567 000232      $APTHD:
568 000232 000000      $HIBTS: .WORD 0          ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
569 000234 001200      $MBADR: .WORD $MAIL      ;;ADDRESS OF APT MAILBOX (BITS 0-15)
570 000236 000000      $TSTM: .WORD          ;;RUN TIME OF LONGEST TEST
571 000240 000000      $PASIM: .WORD          ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
572 000242 000000      $UNITM: .WORD          ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
573 000244 000052      .WORD $FTEND $MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)

```


N1

COKDABO RDJ11 B CLUSTER MAC111 30(1046) 05-APR-84 16:45 PAGE 14
 COKDAB.P11 05-APR-84 16:45 COMMON TAGS

SEQ 0013

```

574      .SBTTL  COMMON TAGS
575
576      ;*****
577      ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
578      ;*USED IN THE PROGRAM.
579
580      001100      .=1100
581      001100      $CMTAG:      .;START OF COMMON TAGS
582      001100      000000      .WORD      0
583      001102      000      $TSTNM: .BYTE      0      ;CONTAINS THE TEST NUMBER
584      001103      000      $ERFLG: .BYTE      0      ;CONTAINS ERROR FLAG
585      001104      000000      $ICNT: .WORD      0      ;CONTAINS SUBTEST ITERATION COUNT
586      001106      000000      $IPADR: .WORD      0      ;CONTAINS SCOPE LOOP ADDRESS
587      001110      000000      $IPERR: .WORD      0      ;CONTAINS SCOPE RETURN FOR ERRORS
588      001112      000000      $ERTTL: .WORD      0      ;CONTAINS TOTAL ERRORS DETECTED
589      001114      000      $ITEMB: .BYTE      0      ;CONTAINS ITEM CONTROL BYTE
590      001115      001      $ERMAX: .BYTE      1      ;CONTAINS MAX. ERRORS PER TEST
591      001116      000000      $ERRPC: .WORD      0      ;CONTAINS PC OF LAST ERROR INSTRUCTION
592      001120      000000      $GADR: .WORD      0      ;CONTAINS ADDRESS OF 'GOOD' DATA
593      001122      000000      $BADADR: .WORD      0      ;CONTAINS ADDRESS OF 'BAD' DATA
594      001124      000000      $GDDAT: .WORD      0      ;CONTAINS 'GOOD' DATA
595      001126      000000      $BDDAT: .WORD      0      ;CONTAINS 'BAD' DATA
596      001130      000000      .WORD      0      ;RESERVED--NOT TO BE USED
597      001132      000000      .WORD      0
598      001134      000      $AUTOB: .BYTE      0      ;AUTOMATIC MODE INDICATOR
599      001135      000      $INTAG: .BYTE      0      ;INTERRUPT MODE INDICATOR
600      001136      000000      .WORD      0
601      001140      177570      $SWR: .WORD      DSWR      ;ADDRESS OF SWITCH REGISTER
602      001142      177570      $DISPLAY: .WORD      DDISP      ;ADDRESS OF DISPLAY REGISTER
603      001144      177560      $TKS: 177560      ;TTY KBD STATUS
604      001146      177562      $TKB: 177562      ;TTY KBD BUFFER
605      001150      177564      $TPS: 177564      ;TTY PRINTER STATUS REG. ADDRESS
606      001152      177566      $TPB: 177566      ;TTY PRINTER BUFFER REG. ADDRESS
607      001154      000      $NULL: .BYTE      0      ;CONTAINS NULL CHARACTER FOR FILLS
608      001155      002      $FILLS: .BYTE      2      ;CONTAINS # OF FILLER CHARACTERS REQUIRED
609      001156      012      $FILLC: .BYTE      12      ;INSERT FILL CHARS. AFTER A "LINE FEED"
610      001157      000      $TPFLG: .BYTE      0      ;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
611      001160      000000      $TMP0: .WORD      0      ;USER DEFINED
612      001162      000000      $TMP1: .WORD      0      ;USER DEFINED
613      001164      000000      $TIMES: 0      ;MAX. NUMBER OF ITERATIONS
614      001166      000000      $ESCAPE: 0      ;ESCAPE ON ERROR ADDRESS
615      001170      177607      000377      $BELL: .ASCIIZ <207><377><377> ;CODE FOR BELL
616      001174      077      $QUES: .ASCIIZ <?> ;QUESTION MARK
617      001175      015      $CRIF: .ASCIIZ <15> ;CARRIAGE RETURN
618      001176      000012      $LF: .ASCIIZ <12> ;LINE FEED
619      ;*****
620      .SBTTL  APT MAILBOX TABLE
621
622      ;*****
623      .EVEN
624      001200      $MAIL: ;APT MAILBOX
625      001200      000000      $MSGTY: .WORD      AMSGTY ;MESSAGE TYPE CODE
626      001202      000000      $FATAL: .WORD      AFATAL ;FATAL ERROR NUMBER
627      001204      000000      $TESTN: .WORD      ATESTN ;TEST NUMBER
628      001206      000000      $PASS: .WORD      APASS ;PASS COUNT
629      001210      000000      $DEVCT: .WORD      ADEVCT ;DEVICE COUNT

```

630	001212	000000	\$UNIT: .WORD	AUNIT	;; I/O UNIT NUMBER
631	001214	000000	\$MSGAD: .WORD	AMSGAD	;; MESSAGE ADDRESS
632	001216	000000	\$MSGLG: .WORD	AMSLG	;; MESSAGE LENGTH
633	001220		\$ETABLE: .WORD		;; APT ENVIRONMENT TABLE
634	001220	000	\$ENV: .BYTE	AENV	;; ENVIRONMENT BYTE
635	001221	000	\$ENVM: .BYTE	AENVM	;; ENVIRONMENT MODE BITS
636	001222	000000	\$SWREG: .WORD	ASWREG	;; APT SWITCH REGISTER
637	001224	000000	\$USWR: .WORD	AUSWR	;; USER SWITCHES
638	001226	000000	\$CPUOP: .WORD	ACPUOP	;; CPU TYPE, OPTIONS
639			;;		BITS 15-11=CPU TYPE
640			;;		11/04=01, 11/05=02, 11/20=03, 11/40=04, 11/45=05
641			;;		11/70=06, PDQ=07, Q=10
642			;;		BIT 10=REAL TIME CLOCK
643			;;		BIT 9=FLOATING POINT PROCESSOR
644			;;		BIT 8=MEMORY MANAGEMENT
645	001230	000	\$MAMS1: .BYTE	AMAMS1	;; HIGH ADDRESS, M.S. BYTE
646	001231	000	\$MTYP1: .BYTE	AMTYP1	;; MEM. TYPE, BLK#1
647			;;		MEM. TYPE BYTE (HIGH BYTE)
648			;;		900 NSEC CORE=001
649			;;		300 NSEC BIPOLAR=002
650			;;		500 NSEC MOS=003
651	001232	000000	\$MADR1: .WORD	AMADR1	;; HIGH ADDRESS, BLK#1
652			;;		MEM. LAST ADDR.=3 BYTES, THIS WORD AND LOW OF "TYPE" ABOVE
653	001234	000	\$MAMS2: .BYTE	AMAMS2	;; HIGH ADDRESS, M.S. BYTE
654	001235	000	\$MTYP2: .BYTE	AMTYP2	;; MEM. TYPE, BLK#2
655	001236	000000	\$MADR2: .WORD	AMADR2	;; MEM. LAST ADDRESS, BLK#2
656	001240	000	\$MAMS3: .BYTE	AMAMS3	;; HIGH ADDRESS, M.S. BYTE
657	001241	000	\$MTYP3: .BYTE	AMTYP3	;; MEM. TYPE, BLK#3
658	001242	000000	\$MADR3: .WORD	AMADR3	;; MEM. LAST ADDRESS, BLK#3
659	001244	000	\$MAMS4: .BYTE	AMAMS4	;; HIGH ADDRESS, M.S. BYTE
660	001245	000	\$MTYP4: .BYTE	AMTYP4	;; MEM. TYPE, BLK#4
661	001246	000000	\$MADR4: .WORD	AMADR4	;; MEM. LAST ADDRESS, BLK#4
662	001250	000000	\$VECT1: .WORD	AVECT1	;; INTERRUPT VECTOR#1, BUS PRIORITY#1
663	001252	000000	\$VECT2: .WORD	AVECT2	;; INTERRUPT VECTOR#2, BUS PRIORITY#2
664	001254	000000	\$BASE: .WORD	ABASE	;; BASE ADDRESS OF EQUIPMENT UNDER TEST
665	001256	000000	\$DEVN: .WORD	ADEVN	;; DEVICE MAP
666	001260	000000	\$CDW1: .WORD	ACDW1	;; CONTROLLER DESCRIPTION WORD#1
667	001262	000000	\$CDW2: .WORD	ACDW2	;; CONTROLLER DESCRIPTION WORD#2
668	001264	000000	\$DDW0: .WORD	ADDW0	;; DEVICE DESCRIPTOR WORD#0
669	001266	000000	\$DDW1: .WORD	ADDW1	;; DEVICE DESCRIPTOR WORD#1
670	001270	000000	\$DDW2: .WORD	ADDW2	;; DEVICE DESCRIPTOR WORD#2
671	001272	000000	\$DDW3: .WORD	ADDW3	;; DEVICE DESCRIPTOR WORD#3
672	001274	000000	\$DDW4: .WORD	ADDW4	;; DEVICE DESCRIPTOR WORD#4
673	001276	000000	\$DDW5: .WORD	ADDW5	;; DEVICE DESCRIPTOR WORD#5
674	001300	000000	\$DDW6: .WORD	ADDW6	;; DEVICE DESCRIPTOR WORD#6
675	001302	000000	\$DDW7: .WORD	ADDW7	;; DEVICE DESCRIPTOR WORD#7
676	001304	000000	\$DDW8: .WORD	ADDW8	;; DEVICE DESCRIPTOR WORD#8
677	001306	000000	\$DDW9: .WORD	ADDW9	;; DEVICE DESCRIPTOR WORD#9
678	001310	000000	\$DDW10: .WORD	ADDW10	;; DEVICE DESCRIPTOR WORD#10
679	001312	000000	\$DDW11: .WORD	ADDW11	;; DEVICE DESCRIPTOR WORD#11
680	001314	000000	\$DDW12: .WORD	ADDW12	;; DEVICE DESCRIPTOR WORD#12
681	001316	000000	\$DDW13: .WORD	ADDW13	;; DEVICE DESCRIPTOR WORD#13
682	001320	000000	\$DDW14: .WORD	ADDW14	;; DEVICE DESCRIPTOR WORD#14
683	001322	000000	\$DDW15: .WORD	ADDW15	;; DEVICE DESCRIPTOR WORD#15
684					
685					

CORDARO RD.11 B CLUSTER MAC111 5001046) 05 APR 84 16:45 PAGE 16
CORDAR.P11 05 APR 84 16:45 API MAILBOX-ETABLE

889 001324
88

\$ETEND:

SEQ 0015

688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743

001324

001324 122614
001326 130360
001330 131430
001332 000000

001334 122650
001336 130360
001340 131430
001342 000000

001344 122662
001346 130360
001350 131430
001352 000000

001354 122674
001356 130405
001360 131436
001362 000000

001364 122734
001366 130472
001370 131450
001372 000000

001374 122767
001376 130472
001400 131450
001402 000000

001404 123032
001406 130571
001410 131464
001412 000000

001414 123066
001416 130571

.SBTTL ERROR POINTER TABLE

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

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

\$ERRTB:

.SBTTL ERROR DEFINITIONS

;ITEM 1
EM1 ;CPU ERROR
DH1 ;TEST 0, ERROR PC
DT1 ;\$TMP1,\$ERRPC
0

;ITEM 2
EM2 ;MMU ERROR
DH1 ;TEST 0, ERROR PC
DT1 ;\$TMP1,\$ERRPC
0

;ITEM 3
EM3 ;FPP ERROR
DH1 ;TEST 0, ERROR PC
DT1 ;\$TMP1,\$ERRPC
0

;ITEM 4
EM4 ;ERROR IN READ WRITE BITS OF CCR
DH4 ;TEST 0, PC, EXPECTED DATA, RECEIVED DATA
DT4 ;\$TMP1,\$ERRPC,R1,CCR
0

;ITEM 5
EM5 ;FORCE MISS WRITES TO CACHE
DH5 ;TEST 0, PC, HIT/MISS, DATA IN CACHE, DATA IN MEMORY
DT5 ;\$TMP1,\$ERRPC, R2, R1, \$GDDAT
0

;ITEM 6
EM6 ;FORCE MISS WRITE INVALIDATES CACHE
DH5 ;TEST 0, PC, HIT/MISS, DATA IN CACHE, DATA IN MEMORY
DT5 ;\$TMP1,\$ERRPC, R2, R1, \$GDDAT
0

;ITEM 7
EM7 ;UNEXPECTED PARITY INTERRUPT
DH7 ;TEST 0, PC, ADDRESS ACCESSED, MSER
DT7 ;\$TMP1,\$ERRPC,\$BDADR,MSER
0

;ITEM 10
EM10 ;TAG PARITY ERROR
DH7 ;TEST 0, PC, ADDRESS ACCESSED, MSER

744	001420	131464	DT7	;TMP1,\$ERRPC,\$BDADR,MSEB
745	001422	000000	0	
746			;ITEM 11	
747	001424	123107	EM11	;DATA PARITY ERROR
748	001426	130571	DH7	;TEST #, PC, ADDRESS ACCESSED, MSEB
749	001430	131464	DT7	;TMP1,\$ERRPC,\$BDADR,MSEB
750	001432	000000	0	
751			;ITEM 12	
752	001434	123131	EM12	LOW BYTE PARITY ERROR
753	001436	130571	DH7	;TEST #, PC, ADDRESS ACCESSED, MSEB
754	001440	131464	DT7	;TMP1,\$ERRPC,\$BDADR,MSEB
755	001442	000000	0	
756			;ITEM 13	
757	001444	123157	EM13	;HIGH BYTE PARITY ERROR
758	001446	130571	DH7	;TEST #, PC, ADDRESS ACCESSED, MSEB
759	001450	131464	DT7	;TMP1,\$ERRPC,\$BDADR,MSEB
760	001452	000000	0	
761			;ITEM 14	
762	001454	123206	EM14	;ERROR DATA PATH
763	001456	130405	DH4	;TEST #, PC, EXPECTED DATA, RECEIVED DATA
764	001460	131476	DT14	;TMP1,\$ERRPC,\$GDDAT,TSFLOC
765	001462	000000	0	
766			;ITEM 15	
767	001464	123231	EM15	;FORCE MISS READS FROM CACHE
768	001466	130472	DH5	;TEST #, PC, HIT/MISS, DATA IN CACHE, DATA IN MEMORY
769	001470	131450	DT5	;TMP1,\$ERRPC, R2, R1, \$GDDAT
770	001472	000000	0	
771			;ITEM 16	
772	001474	123265	EM16	;FORCE MISS READS FROM CACHE AND MISS
773	001476	130472	DH5	;TEST #, PC, HIT/MISS, DATA IN CACHE, DATA IN MEMORY
774	001500	131450	DT5	;TMP1,\$ERRPC, R2, R1, \$GDDAT
775	001502	000000	0	
776			;ITEM 17	
777	001504	123332	EM17	;ERROR IN RECORDING HITS IN HIT/MISS
778	001506	130405	DH4	;TEST #, PC, EXPECTED DATA, RECEIVED DATA
779	001510	131510	DT17	;TMP1,\$ERRPC,\$GDDAT,RECDAT
780	001512	000000	0	
781			;ITEM 20	
782	001514	123376	EM20	;WRITE BYTE ALLOCATES CACHE
783	001516	130360	DH1	;TEST #, PC
784	001520	131430	DT1	;TMP1,\$ERRPC
785	001522	000000	0	
786			;ITEM 21	
787	001524	123431	EM21	;WRITE BYTE HIT DOES NOT RECORD HIT
788	001526	130360	DH1	;TEST #, PC
789	001530	131430	DT1	;TMP1,\$ERRPC
790	001532	000000	0	
791			;ITEM 22	
792	001534	123474	EM22	;BYTES REVERSED ON WRITE CYCLES
793	001536	130360	DH1	;TEST #, PC
794	001540	131430	DT1	;TMP1,\$ERRPC
795	001542	000000	0	
796			;ITEM 23	
797	001544	123533	EM23	;CONDITIONAL BYPASS DOESN'T INVALIDATE CACHE
798	001546	130472	DH5	;TEST #, PC, HIT/MISS, DATA IN CACHE, DATA IN MEMORY
799	001550	131450	DT5	;TMP1,\$ERRPC, R2, R1, \$GDDAT

800	001552	000000	0	
801			;ITEM 24	
802	001554	123607	EM24	;HITS RECORDED AFTER FLUSHING CACHE
803	001556	130644	DH24	;TEST 0, PC, NUMBER OF HITS
804	001560	131522	DT24	;TMP1,\$ERRPC,R3
805	001562	000000	0	
806			;ITEM 25	
807	001564	123652	EM25	;BYPASS DOESN'T INVALIDATE CACHE
808	001566	130360	DH1	;TEST 0, PC
809	001570	131430	DT1	;TMP1,\$ERRPC
810	001572	000000	0	
811			;ITEM 26	
812	001574	123712	EM26	;MSER DOES NOT CLEAR ON WRITE REFERENCE
813	001576	130360	DH1	;TEST 0, PC
814	001600	131430	DT1	;TMP1,\$ERRPC
815	001602	000000	0	
816			;ITEM 27	
817	001604	123761	EM27	;PARITY ERROR DON'T CAUSE A MISS
818	001606	130710	DH27	;TEST 0, PC, MSER, HIT/MISS
819	001610	131532	DT27	;TMP1,\$ERRPC,MSER,R3,0
820	001612	000000	0	
821			;ITEM 30	
822	001614	124021	EM30	;PARITY ERROR DON'T SET MSER WITH CCR<7>=0
823	001616	130710	DH27	;TEST 0, PC, MSER, HIT/MISS
824	001620	131532	DT27	;TMP1,\$ERRPC,MSER,R3,0
825	001622	000000	0	
826			;ITEM 31	
827	001624	124073	EM31	;PARITY ERROR IGNORED
828	001626	130360	DH1	;TEST 0, PC
829	001630	131430	DT1	;TMP1,\$ERRPC
830	001632	000000	0	
831			;ITEM 32	
832	001634	124120	EM32	;PARITY ERROR IGNORED ON LOW BYTE
833	001636	130360	DH1	;TEST 0, PC
834	001640	131430	DT1	;TMP1,\$ERRPC
835	001642	000000	0	
836			;ITEM 33	
837	001644	124161	EM33	;PARITY ERROR IGNORED ON HIGH BYTE
838	001646	130360	DH1	;TEST 0, PC
839	001650	131430	DT1	;TMP1,\$ERRPC
840	001652	000000	0	
841			;ITEM 34	
842	001654	124223	EM34	;PARITY ABORT LOGIC DOESN'T WORK
843	001656	130360	DH1	;TEST 0, PC
844	001660	131430	DT1	;TMP1,\$ERRPC
845	001662	000000	0	
846			;ITEM 35	
847	001664	124263	EM35	;MSER NOT SET PROPERLY
848	001666	130405	DH4	;TEST 0, PC, EXPECTED DATA, RECEIVED DATA
849	001670	131544	DT35	;TMP1,\$ERRPC,\$GDDAT,MSER,0
850	001672	000000	0	
851			;ITEM 36	
852	001674	124311	EM36	;PARITY INTERRUPT DOESN'T WORK
853	001676	130360	DH1	;TEST 0, PC
854	001700	131430	DT1	;TMP1,\$ERRPC
855	001702	000000	0	


```

856      ;ITEM 37
857      EM37      ;NXM AND PARITY ABORT DIN'T HAPPEN
858      DH1      ;TEST 0, PC
859      DT1      ;$TMP1,$ERRPC
860      0
861      ;ITEM 40
862      EM40      ;PARITY ABORT NOT BLOCKED BY NXM TRAP
863      DH1      ;TEST 0, PC
864      DT1      ;$TMP1,$ERRPC
865      0
866      ;ITEM 41
867      EM41      ;MULTI-PROCESSOR HOOK INSTRUCTION DOESN'T CAUSE MISS
868      DH41      ;TEST 0, PC, INSTRUCTION OPCODE
869      DT41      ;$TMP1,$ERRPC,$BDDAT
870      0
871      ;ITEM 42
872      EM42      ;ERROR IN PARITY LOGIC
873      DH1      ;TEST 0, PC
874      DT1      ;$TMP1,$ERRPC
875      0
876      ;ITEM 43
877      EM43      ;ERROR IN CACHE DATA RAMS
878      DH43      ;TEST 0, PC, EXPECTED DATA, RECEIVED DATA, CACHE LOCATION
879      DT43      ;$TMP1,$ERRPC,R1,RECDAT,$BDADR
880      0
881      ;ITEM 44
882      EM44      ;ERROR IN NXM IN STANDALONE MODE
883      DH1      ;TEST 0, PC
884      DT1      ;$TMP1,$ERRPC
885      0
886      ;ITEM 45
887      EM45      ;HITS NOT RECORDED PROPERLY THRU HIT/MISS
888      DH1      ;TEST 0, PC
889      DT1      ;$TMP1,$ERRPC
890      0
891      ;ITEM 46
892      EM46      ;HIT RECORDED FOR A LOCATION NOT IN CACHE
893      DH47      ;TEST 0, PC, LOCATION ACCESSED
894      DT47      ;$TMP1,$ERRPC,KIPAR6,$BDADR
895      0
896      ;ITEM 47
897      EM47      ;MISS RECORDED FOR A LOCATION THAT SHOULD BE IN CACHE
898      DH47      ;TEST 0, PC, LOCATION ACCESSED
899      DT47      ;$TMP1,$ERRPC,KIPAR6,$BDADR
900      0
901      ;ITEM 50
902      EM50      ;ERROR IN TAG STORE
903      DH47      ;TEST 0, PC, ADDRESS
904      DT50      ;$TMP1,$ERRPC,R1,$BDADR
905      0
906      ;ITEM 51
907      EM51      ;ERROR PCR READ WRITE BITS
908      DH4      ;TEST 0, PC, EXPECTED DATA, RECEIVED DATA
909      DT51      ;$TMP1,$ERRPC,$BDDAT,PCR
910      0
911      ;ITEM 52

```

912	002034	125203	FM52	;ERROR IN BCSR READ WRITE BITS
913	002036	130405	DH4	;TEST #, PC, EXPECTED DATA, RECEIVED DATA
914	002040	131640	DT52	; \$TMP1, \$ERROC, \$GDDAT, BCSR
915	002042	000000	0	
916			; ITEM 53	
917	002044	125241	FM53	;RESET DOESN'T CLEAR BCSR<4>
918	002046	130360	DH1	;TEST #, PC
919	002050	131430	DT1	; \$TMP1, \$ERRPC
920	002052	000000	0	
921			; ITEM 54	
922	002054	125275	FM54	;CHECKSUM ERROR IN 16-BIT ROM
923	002056	130360	DH1	;TEST #, PC
924	002060	131430	DT1	; \$TMP1, \$ERRPC
925	002062	000000	0	
926			; ITEM 55	
927	002064	125333	FM55	;CHKSUM ERROR IN 8-BIT ROM
928	002066	130360	DH1	;TEST #, PC
929	002070	131430	DT1	; \$TMP1, \$ERRPC
930	002072	000000	0	
931			; ITEM 56	
932	002074	125367	FM56	;TIMEOUT READING LKS
933	002076	130360	DH1	;TEST #, PC
934	002100	131430	DT1	; \$TMP1, \$ERRPC
935	002102	000000	0	
936			; ITEM 57	
937	002104	125413	FM57	;LKS<07> DOES NOT BECOME 1
938	002106	130360	DH1	;TEST #, PC
939	002110	131430	DT1	; \$TMP1, \$ERRPC
940	002112	000000	0	
941			; ITEM 60	
942	002114	125445	FM60	;WRITE REFERENCE DOESN'T CLEAR LKS<07>
943	002116	130360	DH1	;TEST #, PC
944	002120	131430	DT1	; \$TMP1, \$ERRPC
945	002122	000000	0	
946			; ITEM 61	
947	002124	125513	FM61	;ILLEGAL LKS INTERRUPTS
948	002126	130360	DH1	;TEST #, PC
949	002130	131430	DT1	; \$TMP1, \$ERRPC
950	002132	000000	0	
951			; ITEM 62	
952	002134	125542	FM62	;PROCESSOR INTERRUPTS DON'T CLEAR LKS<07>
953	002136	130360	DH1	;TEST #, PC
954	002140	131430	DT1	; \$TMP1, \$ERRPC
955	002142	000000	0	
956			; ITEM 63	
957	002144	125613	FM63	;LKS READY DOESN'T GO LOW
958	002146	130360	DH1	;TEST #, PC
959	002150	131430	DT1	; \$TMP1, \$ERRPC
960	002152	000000	0	
961			; ITEM 64	
962	002154	125644	FM64	;WRONG NUMBER OF LKS INTERRUPTS
963	002156	130360	DH1	;TEST #, PC
964	002160	131430	DT1	; \$TMP1, \$ERRPC
965	002162	000000	0	
966			; ITEM 65	
967	002164	125703	FM65	;LKS INTERRUPTS HAPPEN AT WRONG PRIORITIES

968	002166	131211	DH65	;TEST #, PC, PRIORITY
969	002170	131664	DT65	;TMP1,\$ERRPC,\$GDDAT
970	002172	000000	0	
971			;ITEM 66	
972	002174	125753	EM66	;BCSR<12> DOES NOT DISABLES LKS
973	002176	130360	DH1	;TEST #, PC
974	002200	131430	DT1	;TMP1,\$ERRPC
975	002202	000000	0	
976			;ITEM 67	
977	002204	126012	EM67	;BCSR<13> DOESN'T SET LKS<06>
978	002206	130360	DH1	;TEST #, PC
979	002210	131430	DT1	;TMP1,\$ERRPC
980	002212	000000	0	
981			;ITEM 70	
982	002214	126047	EM70	;RESET DOESN'T SET LKS<7>
983	002216	130360	DH1	;TEST #, PC
984	002220	131430	DT1	;TMP1,\$ERRPC
985	002222	000000	0	
986			;ITEM 71	
987	002224	126100	EM71	;RESET DOESN'T CLEAR LKS<06>
988	002226	130360	DH1	;TEST #, PC
989	002230	131430	DT1	;TMP1,\$ERRPC
990	002232	000000	0	
991			;ITEM 72	
992	002234	126134	EM72	;TIMEOUT READING SLU REGISTERS
993	002236	131255	DH72	;TEST #, PC, ADDRESS FAILED
994	002240	131556	DT41	;TMP1,\$ERRPC,\$BDDAT
995	002242	000000	0	
996			;ITEM 73	
997	002244	126172	EM73	;XMIT READY DIDN'T GO LOW
998	002246	130360	DH1	;TEST #, PC
999	002250	131430	DT1	;TMP1,\$ERRPC
1000	002252	000000	0	
1001			;ITEM 74	
1002	002254	126216	EM74	;BCSR DOESN'T BECOME 1
1003	002256	130360	DH1	;TEST #, PC
1004	002260	131430	DT1	;TMP1,\$ERRPC
1005	002262	000000	0	
1006			;ITEM 75	
1007	002264	126247	EM75	;WRONG CHARACTER RECEIVED
1008	002266	130405	DH4	;TEST #, PC, EXPECTED DATA, RECEIVED DATA
1009	002270	131674	DT75	;TMP1,\$ERRPC,\$GDDAT,\$BDDAT
1010	002272	000000	0	
1011			;ITEM 76	
1012	002274	126300	EM76	;BCSR<07> NOT CLEARED AFTER READING RBUF
1013	002276	130360	DH1	;TEST #, PC
1014	002300	131430	DT1	;TMP1,\$ERRPC
1015	002302	000000	0	
1016			;ITEM 77	
1017	002304	126350	EM77	;BCSR<07> NOT SET ON RESET
1018	002306	130360	DH1	;TEST #, PC
1019	002310	131430	DT1	;TMP1,\$ERRPC
1020	002312	000000	0	
1021			;ITEM 100	
1022	002314	126407	EM100	;BCSR<07> NOT CLEARED ON RESET
1023	002316	130360	DH1	;TEST #, PC

```

1024 002320 131430          DT1          ;$TMP1,$ERRPC
1025 002322 000000          0
1026          ;ITEM 101
1027 002324 126440          EM101         ;SLU INTERRUPTS HAPPEN AT 4
1028 002326 130360          DH1          ;TEST #, PC
1029 002330 131430          DT1          ;$TMP1,$ERRPC
1030 002332 000000          0
1031          ;ITEM 102
1032 002334 126473          EM102         ;RESET DOES NOT CLEAR XCSR<6> AND RSCR<6>
1033 002336 130360          DH1          ;TEST #, PC
1034 002340 131430          DT1          ;$TMP1,$ERRPC
1035 002342 000000          0
1036          ;ITEM 103
1037 002344 126555          EM103         ;TRANSMIT INTERRUPT DOES NOT CLEAR XCSR<07>
1038 002346 130360          DH1          ;TEST #, PC
1039 002350 131430          DT1          ;$TMP1,$ERRPC
1040 002352 000000          0
1041          ;ITEM 104
1042 002354 126630          EM104         ;RECEIVE INTERRUPTS DON'T CLEAR RCSR<07>
1043 002356 130360          DH1          ;TEST #, PC
1044 002360 131430          DT1          ;$TMP1,$ERRPC
1045 002362 000000          0
1046          ;ITEM 105
1047 002364 126700          EM105         ;BREAK CONDITION DOES NOT SET RBUF PROPERLY
1048 002366 131321          DH105        ;TEST #, PC, RBUF
1049 002370 131706          DT105        ;$TMP1,$ERRPC,RBUF
1050 002372 000000          0
1051          ;ITEM 106
1052 002374 126753          EM106         ;RBUF WASN'T CLEARED ON NEXT CHAR.
1053 002376 131321          DH105        ;TEST #, PC, RBUF
1054 002400 131706          DT105        ;$TMP1,$ERRPC,RBUF
1055 002402 000000          0
1056          ;ITEM 107
1057 002404 127031          EM107         ;ERROR IN WRITING TO XCSR<0>
1058 002406 130360          DH1          ;TEST #, PC
1059 002410 131430          DT1          ;$TMP1,$ERRPC
1060 002412 000000          0
1061          ;ITEM 110
1062 002414 127066          EM110         ;RESET DOES NOT CLEAR XCSR<00>
1063 002416 130360          DH1          ;TEST #, PC
1064 002420 131430          DT1          ;$TMP1,$ERRPC
1065 002422 000000          0
1066          ;ITEM 111
1067 002424 127130          EM111         ;FIRST CHARACTER WAS NOT OVERRUN BY THE SECOND
1068 002426 130405          DH4          ;TEST #, PC, EXPECTED DATA, RECEIVED DATA
1069 002430 131674          DT15        ;$TMP1,$ERRPC,$GDDAT,$BDDAT
1070 002432 000000          0
1071          ;ITEM 112
1072 002434 127206          EM112         ;OVERRUN CONDITION DOES NOT SET PROPER BITS IN RBUF
1073 002436 131321          DH105        ;TEST #, PC, RBUF
1074 002440 131706          DT105        ;$TMP1,$ERRPC,RBUF
1075 002442 000000          0
1076          ;ITEM 113
1077 002444 127271          EM113         ;RBUF WAS NOT CLEARED ON THE NEXT CHARACTER
1078 002446 131321          DH105        ;TEST #, PC, RBUF
1079 002450 131706          DT105        ;$TMP1,$ERRPC,RBUF

```

```

1080 002452 000000      0
1081      ;ITEM 114      EM114      ;ERROR IN XCSR<2>
1082      DH1            ;TEST 0, PC
1083      DT1            ;$TMP1,$ERRPC
1084      0
1085      ;ITEM 115      EM115      ;ERROR IN TAG STORE FROM STANDALONE MODE
1086      DH115          ;TEST 0, PC, MSER, ADDRESS ACCESSED
1087      DT115          ;$TMP1,$ERRPC,$BDDAT,KIPAR6,$BDAOR
1088      0
1089      ;ITEM 116      EM116      ;DMA TAG PARITY DOES NOT SET MSER<4>
1090      DH1            ;TEST 0, PC
1091      DT1            ;$TMP1,$ERRPC
1092      0
1093      ;ITEM 117      EM117      ;MSER<13>NOT SET IN STANDALONE MODE
1094      DH1            ;TEST 0, PC
1095      DT1            ;$TMP1,$ERRPC
1096      0
1097      ;ITEM 120      EM120      ;DMA WRITE HITS DON'T INVALIDATE CACHE
1098      DH1            ;TEST 0, PC
1099      DT1            ;$TMP1,$ERRPC
1100      0
1101      ;ITEM 121      EM121      ;IN BLOCK MODE ON WRITE DMA HITS NOT EVERYTHING IS INVALIDATED
1102      DH1            ;TEST 0, PC
1103      DT1            ;$TMP1,$ERRPC
1104      0
1105      ;ITEM 122      EM122      ;READ DMA HIT IS MESSED UP
1106      DH1            ;TEST 0, PC
1107      DT1            ;$TMP1,$ERRPC
1108      0
1109      ;ITEM 123      EM123      ;ERROR IN DMA CYCLES FROM Q2PBE
1110      DH1            ;TEST 0, PC
1111      DT1            ;$TMP1,$ERRPC
1112      0
1113      ;ITEM 124      EM124      ;PIRQ INTERRUPTS DON'T TAKE PRIORITY OVER Q BUS INTERRUPTS
1114      DH1            ;TEST 0, PC
1115      DT1            ;$TMP1,$ERRPC
1116      0
1117      ;ITEM 125      EM125      ;NO POWER DOWN TRAP TO 24 OCCUR
1118      DH1            ;TEST 0, PC
1119      DT1            ;$TMP1,$ERRPC
1120      0
1121      ;ITEM 126      EM126      ;ERROR IN INTERRUPTS FROM Q2PBE
1122      DH1            ;TEST 0, PC
1123      DT1            ;$TMP1,$ERRPC
1124      0
1125      ;ITEM 127      EM127      ;NO POWER DOWN TRAP TO 24 OCCUR
1126      DH1            ;TEST 0, PC
1127      DT1            ;$TMP1,$ERRPC
1128      0
1129      ;ITEM 128      EM128      ;ERROR IN INTERRUPTS FROM Q2PBE
1130      DH1            ;TEST 0, PC
1131      DT1            ;$TMP1,$ERRPC
1132      0
1133      ;ITEM 129      EM129      ;NO POWER DOWN TRAP TO 24 OCCUR
1134      DH1            ;TEST 0, PC
1135      DT1            ;$TMP1,$ERRPC
1136      0

```

```

1136      ;ITEM 127
1137      002604 130204      EM127      ;ERROR IN PMG COUNTER
1138      002606 130360      DH1        ;TEST #, PC
1139      002610 131430      DT1        ;$TMP1,$ERRPC
1140      002612 000C00      0
1141      ;ITEM 130
1142      002614 130246      EM130      ;UNEXPECTED TIMEOUT
1143      002616 130360      DH1        ;TEST #, PC
1144      002620 131732      DT130     ;$TMP1,$ERRPC
1145      002622 000000      0
1146      ;ITEM 131
1147      002624 130273      EM131      ;ERROR WRITING TO LKS<6>
1148      002626 130360      DH1        ;TEST #, PC
1149      002630 131430      DT1        ;$TMP1,$ERRPC
1150      002632 000000      0
1151      ;ITEM 132
1152      002634 130323      EM132      ;MAINTENANCE REGISTER ERROR
1153      002636 130360      DH1        ;TEST #, PC
1154      002640 131430      DT1        ;$TMP1,$ERRPC
1155      002642 000000      0
1156
1157      .SBTTL GLOBAL VARIABLES AND REGISTER NAMES
1158
1159      ;REGISTERS FOR THE FIRST Q22BE
1160      002644 000000      CSR1: .WORD 0      ;CONTROL REGISTER 1 FOR Q22BE
1161      002646 000000      CSR2: .WORD 0      ;CONTROL/STATUS REGISTER 2
1162      002650 000000      BA: .WORD 0        ;DMA ADDRESS FOR Q22BE
1163      002652 000000      WC: .WORD 0        ;WORD COUNT REGISTER
1164      002654 000000      DATA: .WORD 0     ;DMA DATA FOR Q22BE
1165      002656 000000      VQPR1: .WORD 0     ;ADDRESS OF VECTOR FOR Q22BE
1166      002660 000000      VQPR2: .WORD 0     ;PRIORITY
1167      002662 000000      SIMGOA: .WORD 0    ;SIMULTANEOUS GO ADDRESS REGISTER
1168
1169      ;REGISTERS FOR THE SECOND Q22BE
1170      002664 000000      CSR12: .WORD 0     ;CONTROL REGISTER 1 FOR Q22BE
1171      002666 000000      CSR22: .WORD 0     ;CONTROL/STATUS REGISTER 2
1172      002670 000000      BA2: .WORD 0       ;DMA ADDRESS FOR Q22BE
1173      002672 000000      WC2: .WORD 0       ;WORD COUNT REGISTER
1174      002674 000000      DATA2: .WORD 0    ;DMA DATA FOR Q22BE
1175      002676 000000      VQBE2: .WORD 0     ;ADDRESS OF VECTOR FOR Q22BE
1176      002700 000000      VQPR2: .WORD 0     ;PRIORITY
1177
1178      002702 000000      LKSEL: .WORD 0      ;
1179      002704 000000      ACTCHS: .WORD 0     ;ACTUAL CHECKSUM
1180      002706 000000      SAVPCR: .WORD 0     ;
1181      002710 000000      SAVBR: .WORD 0     ;
1182      002710 002710      .-2710
1183      002710 000000      TEMP: .WORD 0       ;
1184      002712 000017      .BLKW 15          ;RESERVED FOR BLOCK MODE TRANSFER
1185      002750 000000      TIMEOUT: .WORD 0    ;
1186      002752 000032 000012 000006 Q22EN: .WORD 32,12,1,2 ;PRIORITY 7-4 FOR Q22BE
1187      002760 000002
1188
1189
1190      177524      BCR: 177524      ;BOOT/DIAGNOSTICS CONFIGURATION
1191      177524      BDR: 177524      ;BOOT/DIAGNOSTICS DISPLAY

```

M₂

COKDABO KDJ11-H CLOSTER MACY11 30(1046) 05 APR 84 16:45 PAGE 26
 COKDAB.P11 05-APR-84 16:45 GLOBAL VARIABLES AND REGISTER NAMES

SEQ 0025

1192	177520	BCSR=	177520	;BOOT/DIAGNOSTICS STATUS
1193	177746	CCR=	177746	;CACHE CONTROL REGISTER
1194	177752	HITMIS=	177752	;HIT OR MISS REGISTER
1195	177734	KMC=	177734	;UNIBUS CONFIGURATION REGISTER
1196	177546	LKS=	177546	;CLOCK STATUS REGISTER
1197	177750	MAIREG=	177750	;MAINTENANCE REGISTER
1198	177744	MSER=	177744	;MEMORY SYSTEM ERROR
1199	177522	PCR=	177522	;PAGE CONTROL REGISTER
1200	177772	PIR=	177772	;PROGRAM INTERRUPT REQUEST
1201	177562	RBUF=	177562	;RECEIVER DATA BUFFER
1202	177560	RCSR=	177560	;RECEIVER STATUS REGISTER
1203	177566	XBUF=	177566	;TRANSMITTER DATA BUFFER
1204	177564	XCSR=	177564	;TRANSMITTER STATUS REGISTER
1205				
1206	177766	CPEREG=	177766	;CPU ERROR REGISTER
1207				
1208	177572	MMR0=SR0		;MEMORY MANAGEMENT REG. 0
1209	177574	MMR1=SR1		;MEMORY MANAGEMENT REG. 1
1210	177576	MMR2=SR2		;MEMORY MANAGEMENT REG. 2
1211	177516	MMR3=SR3		;MEMORY MANAGEMENT REG. 3
1212	120001	POLY=	120001	
1213	000000	NULL=	0	
1214				
1215		.SBTTL	GLOBAL DATA SECTION	
1216				
1217		;		
1218		; THE GLOBAL DATA SECTION CONTAINS DATA THAT ARE USED		
1219		; IN MORE THAN ONE TEST.		
1220		;		
1221				
1222		;THESE LOCATIONS ARE USED IN MORE THAN ONE TEST TO STORE VECTOR DATA		
1223		;WHEN THE TEST NEEDS TO HAVE AN ERROR CONDITION RESPOND DIFFERENTLY		
1224		;FROM THE DEFAULT RESPONSE.		
1225	002762	000000	SLOC00: .WORD	0
1226	002764	000000	SLOC01: .WORD	0
1227				
1228				
1229	002766	000000		;THESE LOCATIONS ARE USED IN MORE THAN ONE TEST TO STORE WORKING DATA.
1230	002770	000000	LOWADD: .WORD	0 ;STORES LOW ADDRESS FOR RAM TESTS
1231	002772	000000	GOODADD: .WORD	0 ;STORES GOOD ADDRESS FOR RAM TESTS
1232	002774	000000	TSTADD: .WORD	0 ;ADDRESS STORE FOR RAM TESTS
1233	002776	000000	NEWADD: .WORD	0 ;ADDRESS STORE FOR RAM TEST
1234	003000	000000	FLAG: .WORD	0 ;USED TO STORE "FLAG" CONDITIONS
1235	003002	000000	SAVSUP: .WORD	0 ;USED TO STORE SUPERVISOR STACK VALUE
1236	003004	000000	SAVUSE: .WORD	0 ;USED TO STORE USER STACK VALUE
1237	003006	000000	SAVMR0: .WORD	0 ;USED TO STORE MMU STATUS REGISTER 0 DATA
1238	003010	000000	SAVMR1: .WORD	0 ;USED TO STORE MMU STATUS REGISTER 1 DATA
1239	003012	000004	SAVMR2: .WORD	0 ;USED TO STORE MMU STATUS REGISTER 2 DATA
1240	003022	000004	FLOAT: .BLKW	4 ;USED TO STORE VALUES FOR MMU TESTS
1241	003032	000000	FLO: .BLKW	4 ;USED TO STORE VALUES FOR MMU TESTS
1242	003034	000000	SEQ: .WORD	0 ;STORES SEQUENCE NUMBER FOR JUMP TESTS
1243	003036	000000	SPS: .WORD	0 ;STORES STACK POINTER FOR JUMP TESTS
1244	003040	000004	SPSJ: .WORD	0 ;STORES STACK POINTER FOR JUMP TESTS
1245	003050	000004	BTEXP: .BLKW	4 ;STORES EXPONENT DURING BIT TESTS
1246	003060	000000	BTRES: .BLKW	4 ;STORES RECEIVED DATA FOR BIT TESTS
1247	003062	000004	COUNT: .WORD	0 ;ERROR INDICATOR FOR FLOATING POINT TESTS
			RECIEC: .BLKW	4 ;RECEIVED FLOATING POINT EXCEPTION CODE

N2

COKDABO KDJ11-B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 27
 COKDAB.F11 05-APR-84 16:45 GLOBAL DATA SECTION

SEQ 0026

1248	003072	000004	RECST: .BLKW 4	;RECIEVED FLOATING POINT STATUS
1249	003102	000004	RECDST: .BLKW 4	;DESTINATION ADDRESS FOR FLOATING POINT TESTS
1250				
1251				;THESE LOCATIONS ARE USED BY MORE THAN ONE TEST AS LOOP COUNTERS
1252	003112	000000	ALLCTR: .WORD 0	
1253	003114	000000	LOOPIN: .WORD 0	
1254				
1255				;SOME MORE TEMPORARY STORAGE FOR RAM TESTS
1256	003116	000000	SAVPOS: .WORD 0	;STORES TEMPORARY BIT POSITIONS FOR RAM TESTS
1257	003120	000000	MASK: .WORD 0	;STORES BIT MASK FOR ERROR ISOLATION
1258				
1259				;!!!!!!THIS IS IT. THE PROGRAM TEST LOCATION!!!!!!!!!!!!!!!!!!!!!!
1260	003122	000000	TSTLOC: .WORD 0	
1261	003124	000024	BLKW 20.	
1262				;FPP REGISTER DEFINITIONS
1263		000000	AC0 = *0	
1264		000001	AC1 = *1	
1265		000002	AC2 = *2	
1266		000003	AC3 = *3	
1267		000004	AC4 = *4	
1268		000005	AC5 = *5	
1269		000006	AC6 = *6	
1270		000007	AC7 = *7	
1271				
1272				;FPP INTERRUPT VECTOR
1273				
1274		000244	FPVEC=244	
1275				
1276				
1277		001000	STBOT= 1000	
1278				
1279				
1280	003174	123456	TAB1: .WORD 123456	
1281	003176	000000	.WORD 000000	
1282	003200	000000	.WORD 0	
1283	003202	000001	.WORD 1	
1284	003204	055555	TAB2: .WORD 055555	
1285	003206	177777	.WORD 177777	
1286	003210	145671	.WORD 145671	
1287	003212	100000	.WORD 100000	
1288	003214	003000	TAB3: .WORD 003000	
1289	003216	123456	.WORD 123456	
1290	003220	000000	.WORD 0	
1291	003222	000000	.WORD 0	
1292	003224	055555	TAB4: .WORD 55555	
1293	003226	177777	.WORD 1	
1294	003230	000000	.WORD 0	
1295	003232	000000	.WORD 0	
1296	003234	043243	TAB5: .WORD 43243	
1297	003236	000000	.WORD 0	
1298	003240	000000	.WORD 0	
1299	003242	000000	.WORD 0	
1300	003244	162400	TAB5A: .WORD 162400	
1301	003246	000000	.WORD 0	
1302	003250	000000	.WORD 0	
1303	003252	000000	.WORD 0	

CORDARO RDJ11 B CLUSTER MACY11 30(1046) 05 APR 84 16:45 PAGE 28
 CORDAR,P11 05-APR 84 16:45 GLOBAL DATA SECTION

SEQ 0027

1304	003254	000000			TAB6:	.WORD	0
1305	003256	000000				.WORD	0
1306	003260	000000				.WORD	0
1307	003262	000000				.WORD	0
1308	003264	047050			TAB6A:	.WORD	47050
1309	003266	010000				.WORD	10000
1310	003270	000000				.WORD	0
1311	003272	000000				.WORD	0
1312	003274	000200			TAB7:	.WORD	200
1313	003276	000000				.WORD	0
1314	003300	000000				.WORD	0
1315	003302	000000				.WORD	0
1316	003304	000200			TAB8:	.WORD	200
1317	003306	000000				.WORD	0
1318	003310	000000				.WORD	0
1319	003312	000001				.WORD	1
1320	003314	000400	000000	000000	TAB9:	.WORD	400,0,0,0
1321	003322	000000					
1322	003324	030000			TAB10:	.WORD	30000
1323	003326	003000				.WORD	3000
1324	003330	000000				.WORD	0
1325	003332	000000				.WORD	0
1326	003334	016400			TAB11:	.WORD	16400
1327	003336	000000				.WORD	0
1328	003340	000000				.WORD	0
1329	003342	000000				.WORD	0
1330	003344	030000	003000	000002	TAB11A:	.WORD	30000,3000,2,0
1331	003352	000000					
1332	003354	016100	000000	000000	TAB12:	.WORD	16100,0,0,1
1333	003362	000001					
1334	003364	016200			TAB13:	.WORD	16200
1335	003366	000000				.WORD	0
1336	003370	000000				.WORD	0
1337	003372	000001				.WORD	1
1338	003374	030000	003000	000000	TAB13B:	.WORD	30000,3000,0,140000
1339	003402	140000					
1340	003404	030000			TAB14:	.WORD	30000
1341	003406	000000				.WORD	0
1342	003410	000000				.WORD	0
1343	003412	000000				.WORD	0
1344	003414	024700			TAB15:	.WORD	24700
1345	003416	000000				.WORD	0
1346	003420	000000				.WORD	0
1347	003422	000000				.WORD	0
1348	003424	025000			TAB16:	.WORD	25000
1349	003426	175363				.WORD	175363
1350	003430	123456				.WORD	123456
1351	003432	123456				.WORD	123456
1352	003434	030000			TAB17:	.WORD	30000
1353	003436	007020				.WORD	7020
1354	003440	000000	000000			.WORD	0,0
1355	003444	023456			TAB18:	.WORD	23456
1356	003446	000000				.WORD	0
1357	003450	000000				.WORD	0
1358	003452	000001				.WORD	1
1359	003454	100200	000000	000000	TAB21:	.WORD	100200,0,0,0

C 4

SEQ 0028

1360	003462	000000				
1361	003464	100400	000000	000000	TAB22: .WORD	100400,0,0,0
1362	003472	000000				
1363	003474	000200	000000	000000	TAB23: .WORD	200,0,0,1
1364	003502	000001				
1365	003504	062400	000000	000000	TAB24: .WORD	62400,0,0,0
1366	003512	000000				
1367	003514	001100	000000	000000	TAB25: .WORD	1100,0,0,0
1368	003522	000000				
1369	003524	100600	000000	000000	TAB26: .WORD	100600,0,0,0
1370	003532	000000				
1371	003534	001000	000000	000000	TAB27: .WORD	1000,0,0,0
1372	003542	000000				
1373	003544	000600	000000	000000	TAB28: .WORD	600,0,0,0
1374	003552	000000				
1375	003554	010100	000000	000000	TAB29: .WORD	10100,0,0,0
1376	003562	000000				
1377	003564	010100	000000	002000	TAB29A: .WORD	10100,0,2000,0
1378	003572	000000				
1379						
1380	003574	000500	000000	000000	TAB30: .WORD	500,0,0,0
1381	003602	000000				
1382	003604	100400	000000	000000	TAB31: .WORD	100400,0,0,0
1383	003612	000000				
1384	003614	016000	000000	000000	TAB32: .WORD	16000,0,0,0
1385	003622	000000				
1386	003624	011600	000000	000000	TAB33: .WORD	11600,0,0,0
1387	003632	000000				
1388	003634	000640	000000	000000	TAB34: .WORD	640,0,0,0
1389	003642	000000				
1390	003644	077600	000000	000000	TAB40: .WORD	77600,0,0,0
1391	003652	000000				
1392	003654	100200	000000	000000	TAB41: .WORD	100200,0,0,1
1393	003662	000001				
1394	003664	000340	000000	000000	TAB42: .WORD	340,0,0,0
1395	003672	000000				
1396	003674	000077	177777	177777	TAB43: .WORD	77,177777,177777,177776
1397	003702	177776				
1398	003704	000577	177777	177777	TAB45: .WORD	577, 1, 1, 1
1399	003712	177777				
1400	003714	000577	177777	000000	TAB46: .WORD	577, 1,0,0
1401	003722	000000				
1402	003724	173737	124242	052525	TAB47: .WORD	173737,124242,052525,12346
1403	003732	012346				
1404	003734	000000	000000	052525	TAB47A: .WORD	0,0,052525,12346
1405	003742	012346				
1406	003744	173737	124242	000000	TAB48: .WORD	173737,124242,0,0
1407	003752	000000				
1408	003754	000600	000000	000000	TAB49: .WORD	600,0,0,0
1409	003762	000000				
1410						
1411	003764				START:	
1412					;; LCP/ORTION ROUTINE TO SAVE EMULATOR AND PRIORITY	
1413						
1414	003764	005737	004052		EMTSAB: LST	;; FIRST TIME THROUGH ?
1415	003770	001034			BNE VMKOR	;; BRANCH IF BEEN HERE ALREADY


```

1416 003772 032737 000040 000052      BIT      0BIT5,0052      ;; ARE WE IN UFD MODE ?
1417 004000 001430                BEQ      VMKOR          ;; LEAVE IF NOT
1418 004002 012737 177777 004056      MOV      0 1,UFDPLG      ;; SET UFD FLAG
1419 004010 032737 000100 000052      BIT      0BIT6,0052      ;; ARE WE IN QUIET MODE ?
1420 004016 001403                BEQ      1$          ;; BR IF NOT
1421 004020 012737 177777 004060      MOV      0 1,UQUIET      ;; SET QUIET MODE
1422 004026 104042      1$:      EMT      42          ;; GET ADDRESS OF XXDP DCA TABLE
1423 004030 005060 000042                CLR      42(R0)      ;; CLR XXDP+ "DRSERR"
1424 004034 013737 000030 004052      MOV      30,SAV30      ;; SAVE EMULATOR ADDRESS
1425 004042 013737 000032 004054      MOV      32,SAV32      ;; SAVE EMULATOR PRIORITY LEVEL
1426 004050 000404                BR      VMKOR          ;; GET AROUND TAG AREA
1427 004052 000000      SAV30:  .WORD    0          ;; PUT EMULATOR INFO HERE
1428 004054 000000      SAV32:  .WORD    0          ;; PUT PRIORITY LOCATION      HERE
1429 004056 000000      UFDPLG: .WORD    0          ;; USER FRIENDLY MODE FLAG
1430 004060 000000      UQUIET: .WORD    0          ;; UFD QUIET MODE FLAG
1431 004062      VMKOR:
1432      ;*****
1433
1434
1435 004062      1$:
1436      .SBTTL  INITIALIZE THE COMMON TAGS
1437      ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
1438 004062 012706 001100      MOV      0$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
1439 004066 005026                CLR      (R6)+        ;;CLEAR MEMORY LOCATION
1440 004070 022706 001140      CMP      0$SWR,R6      ;;DONE?
1441 004074 001374                BNE      -6          ;;LOOP BACK IF NO
1442 004076 012706 001100      MOV      0$STACK,SP      ;;SETUP THE STACK POINTER
1443      ;;INITIALIZE A FEW VECTORS
1444 004102 012737 133644 000020      MOV      0$SCOPE,0$IOTVEC  ;;IOT VECTOR FOR SCOPE ROUTINE
1445 004110 012737 000340 000022      MOV      0340,0$IOTVEC+2  ;;LEVEL 7
1446 004116 012737 134132 000030      MOV      0$ERROR,0$EMTVEC  ;;EMT VECTOR FOR ERROR ROUTINE
1447 004124 012737 000340 000032      MOV      0340,0$EMTVEC+2  ;;LEVEL 7
1448 004132 012737 136656 000034      MOV      0$TRAP,0$TRAPVEC  ;;TRAP VECTOR FOR TRAP CALLS
1449 004140 012737 000340 000036      MOV      0340,0$TRAPVEC+2  ;;LEVEL 7
1450 004146 012737 136740 000024      MOV      0$PWRDN,0$PWRVEC  ;;POWER FAILURE VECTOR
1451 004154 012737 000340 000026      MOV      0340,0$PWRVEC+2  ;;LEVEL 7
1452 004162 013737 135556 135550      MOV      0$ENDCT,0$EOPCT  ;;SETUP END-OF PROGRAM COUNTER
1453 004170 005037 001164                CLR      0$TIMES      ;;INITIALIZE NUMBER OF ITERATIONS
1454 004174 005037 001166                CLR      0$ESCAPE      ;;CLEAR THE ESCAPE ON ERROR ADDRESS
1455 004200 112737 000001 001115      MOV      01,0$TERMAX      ;;ALLOW ONE ERROR PER TEST
1456 004206 012737 004206 001106      MOV      0,,0$LPADR      ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
1457 004214 012737 004214 001110      MOV      0,,0$LPERR      ;;SETUP THE ERROR LOOP ADDRESS
1458      ;;SIZE FOR A HARDWARE SWITCH REGISTER, IF NOT FOUND OR IT IS
1459      ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
1460 004222 013746 000004                MOV      0$ERRVEC, (SP)  ;;SAVE ERROR VECTOR
1461 004226 012737 004262 000004      MOV      0641,0$ERRVEC  ;;SET UP ERROR VECTOR
1462 004234 012737 177570 001140      MOV      0$SWR,SWR      ;;SETUP FOR A HARDWARE SWICH REGISTER
1463 004242 012737 177570 001142      MOV      0$DISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
1464 004250 022777 177777 174662      CMP      0 1,0$SWR      ;;TRY TO REFERENCE HARDWARE SWR
1465 004256 001012                BNE      66$          ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
1466
1467 004260 000403                BR      65$          ;;BRANCH IF NO TIMEOUT
1468 004262 012716 004270      64$:      MOV      0651,(SP)      ;;SET UP FOR TRAP RETURN
1469 004266 000002      65$:      RTI
1470 004270 012737 000176 001140      MOV      0$SWREG,SWR      ;;POINT TO SOFTWARE SWR
1471 004276 012737 000174 001142      MOV      0$DISPREG,DISPLAY

```

```

1472 004304 012637 000004      66$: MOV      (SP)+, @ERRVEC      ;RESTORE ERROR VECTOR
1473
1474 004310 005037 001206      CLR      $PASS          ;CLEAR PASS COUNT
1475 004314 132737 000200 001221  BITH     @APTSIZE, $ENVM      ;TEST USER SIZE UNDER APT
1476 004322 001403      BEQ      67$          ;YES, USE NON-APT SWITCH
1477 004324 012737 001222 001140  MOV      @SWREG, SWR        ;NO, USE APT SWITCH REGISTER
1478 004332
1479 004332 013737 004056 004060 67$: MOV      UFDFLG, UQUIET      ;ABORT IN UFD ON ERROR
1480 004340 012737 133124 000004  MOV      @TOUT, @ERRVEC      ;POINT TO TIMEOUT ROUTINE
1481 004346 012737 000340 000006  MOV      @340, @ERRVEC+2    ;AT PRIORITY 7
1482 004354 012737 132374 000114  MOV      @RAMPAR, @0114     ;POINT PARITY ABORT
1483 004362 012737 000340 000116  MOV      @340, @0116       ;AT PRIORITY 7
1484 004370 012737 133306 000250  MOV      @MMUTRP, @0250     ;POINT MMU TRAP VECTOR
1485 004376 012737 000340 000252  MOV      @340, @0252       ;
1486 004404 005037 177766      CLR      @017766          ;CLEAR CPU ERROR REGISTER
1487 004410 032737 000100 000052  BIT      @BIT06, @052      ;IN UFD QUIET MODE ?
1488 004416 001061      BNE      LOOP          ;IF SO, SKIP PRINTOUT
1489 004420 012701 004432      MOV      @+, 12, R1       ;STORE LOCATION POINTER
1490 004424 012711 177777      MOV      @-1, (R1)        ;MAKE SURE TO PRINT NOT ONLY AT FIRST TIME
1491
1492      .SBTTL  TYPE PROGRAM NAME
1493      ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
1494 004430 005227 177777      INC      @-1          ;FIRST TIME?
1495 004434 001044      BNE      68$          ;BRANCH IF NO
1496 004436 022737 133610 000042  CMP      @ENDAD, @042      ;ACT-11?
1497 004444 001440      BEQ      68$          ;BRANCH IF YES
1498 004446 104401 004514      TYPE     ,69$          ;TYPE ASCII STRING
1499      .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
1500 004452 005737 000042      TST      @042          ;ARE WE RUNNING UNDER XXDP/ACT?
1501 004456 001012      BNE      70$          ;BRANCH IF YES
1502 004460 123727 001220 000001  CMPB     $ENV, @1          ;ARE WE RUNNING UNDER APT?
1503 004466 001406      BEQ      70$          ;BRANCH IF YES
1504 004470 023727 001140 000176  CMP      SWR, @SWREG       ;SOFTWARE SWITCH REG SELECTED?
1505 004476 001005      BNE      71$          ;BRANCH IF NO
1506 004500 104406      GTSWR          ;GET SOFT-SWR SETTINGS
1507 004502 000403      BR      71$
1508 004504 112737 000001 001134 70$: MOVB     @1, $AUTOB      ;SET AUTO-MODE INDICATOR
1509 004512 000415      71$:
1510      BR      68$          ;GET OVER THE ASCII?
1511      ;;69$: .ASCIIZ <CRLF>*KDJ11 B CPU DIAGNOSTIC* <CRLF>
1512      68$:
1513 004546 032737 000200 000052  BIT      @BIT07, @052      ;UFD MODE?
1514 004554 001002      BNE      LOOP          ;IF YES, BRANCH
1515 004556 004737 132452      JSR      PC, Q22SI2      ;SIZE FOR Q22BE
1516 004562
1517      LOOP:
1518      .DSABLE AMA
1519      *****
1520      IST1:  COPE
1521      .SBTTL  BASE INSTRUCTION SET TESTS
1522      *****
1523      BEGIN BASE INSTRUCTION SET TESTING
1524      *****
1525      FRSTST:
1526      ;TEST BEQ BNE INSTRUCTIONS
1527

```

```

1528
1529
1530 004564 000277
1531 004566 000244
1532 004570 001401
1533 004572 001001
1534
1535
1536
1537
1538
1539 004574 104001
1540 004576 000257
1541 004600 000264
1542 004602 001001
1543 004604 001401
1544
1545 004606 104001
1546 004610
1547 004610
1548
1549
1550
1551
1552 004610 000257
1553 004612 103001
1554
1555 004614 104001
1556 004616 000261
1557 004620 103401
1558
1559 004622 104001
1560 004624
1561
1562 004624
1563
1564
1565 004624 005000
1566 004626 001005
1567
1568 004630 005010
1569 004632 001003
1570 004634 005737 000000
1571 004640 001401
1572
1573 004642 104001
1574 004644
1575 004644
1576
1577
1578 004644 012737 125252 000000
1579 004652 022737 125252 000000
1580 004660 001401
1581
1582
1583

;*****
;THESE TWO INSTRUCTIONS ARE FUNDAMENTAL TO RECOGNIZING ERROR CONDITIONS
;
;SCC
;CLZ
;CC=0100 - Z BIT CLEARED
;*TEST INSTR - TRY TO CAUSE A BEQ ERROR
;BRANCH IF GOOD
;THE Z FLAG DIDNT CLEAR OR BRANCH FAILED.
;FAILURE AT THIS LOCATION
;COULD MEAN A BUS PROBLEM, MICRO-CODE PROBLEM
;CONDITION CODE PROBLEM OR JUST ABOUT ANYTHING
;ELSE.
;CPU ERROR
;
;COND CODES = 0100 (ZERO)
;*TEST INSTR* TRY TO BRANCH ON ZERO FLAG
;*TEST INSTR* BRANCH IF GOOD
;BRANCH FAILURE WITH Z BIT SET
;CPU ERROR
;END OF TEST

;
;TEST BRANCH ON CARRY
;*****
;THIS IS A TEST TO SEE IF THE MODULE FORM ANTICIPATED IS FEASIBLE.
;
;CCC
;BCC 2$
;CC=0000
;*TEST INSTR*
;BRANCH CARRY CLEAR FAILED
;CPU ERROR
;CC=1111
;*TEST INSTR*
;BRANCH CARRY SET FAILED
;CPU ERROR

;
;TEST DATA PATHS
;CLR R0
;BNE 1$
;TRY TO INSURE WE ARE TESTING
;THE DATA PATH AND NOT THE "CLR R0" INSTRUCTION
;FORCE LOCATION TO ZERO
;TRY TO INSURE 0=0
;AGAIN, TRY TO INSURE THAT 0=0
;BRANCH IF GOOD
;LOCATION 0 NOT SETUP PROPERLY
;CPU ERROR

;
;TEST DATA PATHS - ONES AND ZEROS
;MOV 0125252,000
;CMP 0125252,000
;BEQ 2$
;0=125252
;SEE IF DATA MADE IT
;BRANCH IF IF DATA IS GOOD
;ERROR! EITHER THE BUS IS BAD,
;OR THE MOV OR COMPARE
;INSTRUCTIONS FAILED

```

CORDBO RDU11 B CLUSTER MAC11 30(1046) 05 APR 84 16:45 PAGE 33
 CORDAB,P11 05-APR 84 16:45 BASE INSTRUCTION SET TESTS

SEQ 0032

```

1584 004662 104001 1$: ERROR +1 ;CPU ERROR
1585 004664 2$: ;END OF TEST
1586
1587
1588 004664 ;M5:
1589
1590 ; TEST DATA PATHS DATA 0'S AND 1'S
1591 004664 012737 052525 000000 MOV #052525,R0 ;SETUP DATA
1592 004672 023727 000000 052525 CMP #0,052525 ; TEST FOR CORRECT DATA
1593 004700 001401 BEQ 2$
1594 004702 104001 1$: ERROR +1 ;CPU ERROR
1595 004704 2$:
1596
1597 004704 ;M6:
1598 ; TEST DATA PATHS - 1'S
1599 004704 005037 000000 CLR #0
1600 004710 005137 000000 COM #0 ;SET UP MEMORY LOCATION 0 = 111111
1601 004714 023727 000000 177777 CMP #0,0177777 ; TEST DATA
1602 004722 001401 BEQ 2$ ;BRANCH IF NO ERROR
1603 004724 104001 1$: ERROR +1 ;CPU ERROR
1604 004726 2$:
1605
1606 ;GPROTS:
1607 004726 ;
1608 ; R0 BIT TESTS
1609 004726 012700 177777 MOV #177777,R0 ;R0=177777
1610 004732 020027 177777 CMP R0,0177777 ;DOES R0=177777
1611 004736 001401 BEQ 1$ ;YES GO ON
1612 ;NO GO TO ERROR
1613 004740 104001 1$: ERROR +1 ;CPU ERROR
1614 004742 005000 CLR R0 ;R0=0
1615 004744 020027 000000 CMP R0,#0 ;DOES R0=0
1616 004750 001401 BEQ 2$ ;YES GO ON
1617 ;NO GO TO ERROR
1618 004752 104001 2$: ERROR +1 ;CPU ERROR
1619 004754 012700 125252 MOV #125252,R0 ;R0=125252
1620 004760 020027 125252 CMP R0,0125252 ;DOES R0=125252
1621 004764 001401 BEQ 3$ ;YES GO ON
1622 ;NO GO TO ERROR
1623 004766 104001 3$: ERROR +1 ;CPU ERROR
1624 004770 012700 052525 MOV #052525,R0 ;R0=52525
1625 004774 020027 052525 CMP R0,052525 ;DOES R0=52525
1626 005000 001401 BEQ 4$ ;YES GO ON
1627 ;NO GO TO ERROR
1628 005002 104001 4$: ERROR +1 ;CPU ERROR
1629 005004
1630 ;GPR1TS:
1631 005004 ;
1632 ; R1 BIT TESTS
1633 005004 012701 177777 MOV #177777,R1 ;R1=177777
1634 005010 020127 177777 CMP R1,0177777 ;DOES R1=177777
1635 005014 001401 BEQ 1$ ;YES GO ON
1636 ;NO GO TO ERROR
1637 005016 104001 1$: ERROR +1 ;CPU ERROR
1638 005020 005001 CLR R1 ;R1=0
1639 005022 020127 000000 CMP R1,#0 ;DOES R1=0

```

CORDARO KDJ11-B CLUSTER MACY11 30(1046) 05 APR 84 16:45 PAGE 34
 CCKDAB.P11 05-APR-84 16:45 BASE INSTRUCTION SET TESTS

SEQ 0033

1640	005026	001401			BEG	2\$		;YES GO ON
1641								;NO GO TO ERROR
1642	005030	104001			ERROR	+1		;CPU ERROR
1643	005032	012701	125252	2\$:	MOV	#125252,R1		;R1=125252
1644	005036	020127	125252		CMP	R1,#125252		;DOES R1=125252
1645	005042	001401			BEG	3\$		;YES GO ON
1646								;NO GO TO ERROR
1647	005044	104001			ERROR	+1		;CPU ERROR
1648	005046	012701	052525	3\$:	MOV	#52525,R1		;R1=52525
1649	005052	020127	052525		CMP	R1,#52525		;DOES R1=52525
1650	005056	001401			BEG	4\$		;YES GO ON
1651								;NO GO TO ERROR
1652	005060	104001			ERROR	+1		;CPU ERROR
1653	005062			4\$:				
1654								
1655	005062				GPR2TS:			
1656								
1657	005062	012702	177777			R2 BIT TESTS		
1658	005066	020227	177777		MOV	#177777,R2		;R2=177777
1659	005072	001401			CMP	R2,#177777		;DOES R2=177777
1660					BEG	1\$		;YES GO ON
1661	005074	104001						;NO GO TO ERROR
1662	005076	005002		1\$:	ERROR	+1		;CPU ERROR
1663	005100	020227	000000		CLR	R2		;R2=0
1664	005104	001401			CMP	R2,#0		;DOES R2=0
1665					BEG	2\$		;YES GO ON
1666	005106	104001						;NO GO TO ERROR
1667	005110	012702	125252	2\$:	ERROR	+1		;CPU ERROR
1668	005114	020227	125252		MOV	#125252,R2		;R2=125252
1669	005120	001401			CMP	R2,#125252		;DOES R2=125252
1670					BEG	3\$		;YES GO ON
1671	005122	104001						;NO GO TO ERROR
1672	005124	012702	052525	3\$:	ERROR	+1		;CPU ERROR
1673	005130	020227	052525		MOV	#52525,R2		;R2=52525
1674	005134	001401			CMP	R2,#52525		;DOES R2=52525
1675					BEG	4\$		;YES GO ON
1676	005136	104001						;NO GO TO ERROR
1677	005140			4\$:	ERROR	+1		;CPU ERROR
1678								
1679	005140				GPR3TS:			
1680								
1681	005140	012703	177777			R3 BIT TESTS		
1682	005144	020327	177777		MOV	#177777,R3		;R3=177777
1683	005150	001401			CMP	R3,#177777		;DOES R3=177777
1684					BEG	1\$		;YES GO ON
1685	005152	104001						;NO GO TO ERROR
1686	005154	005003		1\$:	ERROR	+1		;CPU ERROR
1687	005156	020327	000000		CLR	R3		;R3=0
1688	005162	001401			CMP	R3,#0		;DOES R3=0
1689					BEG	2\$		;YES GO ON
1690	005164	104001						;NO GO TO ERROR
1691	005166	012703	125252	2\$:	ERROR	+1		;CPU ERROR
1692	005172	020327	125252		MOV	#125252,R3		;R3=125252
1693	005176	001401			CMP	R3,#125252		;DOES R3=125252
1694					BEG	3\$		;YES GO ON
1695	005200	104001						;NO GO TO ERROR
					ERROR	+1		;CPU ERROR

1696	005202	012703	052525	3\$:	MOV	#52525,R3	;R3=52525
1697	005206	020327	052525		CMP	R3,#52525	;DOES R3=52525
1698	005212	001401			BEQ	4\$	;YES GO ON
1699							;NO GO TO ERROR
1700	005214	104001			ERROR	+1	;CPU ERROR
1701	005216			4\$:			
1702							
1703	005216						
1704							
1705	005216	012704	177777				
1706	005222	020427	177777				
1707	005226	001401					
1708							
1709	005230	104001					
1710	005232	005004					
1711	005234	020427	000000	1\$:	CLR	R4	;R4=0
1712	005240	001401			CMP	R4,#0	;DOES R4=0
1713					BEQ	2\$	;YES GO ON
1714	005242	104001					;NO GO TO ERROR
1715	005244	012704	125252		ERROR	+1	;CPU ERROR
1716	005250	020427	125252	2\$:	MOV	#125252,R4	;R4=125252
1717	005254	001401			CMP	R4,#125252	;DOES R4=125252
1718					BEQ	3\$	;YES GO ON
1719	005256	104001					;NO GO TO ERROR
1720	005260	012704	052525		ERROR	+1	;CPU ERROR
1721	005264	020427	052525	3\$:	MOV	#52525,R4	;R4=52525
1722	005270	001401			CMP	R4,#52525	;DOES R4=52525
1723					BEQ	4\$	;YES GO ON
1724	005272	104001					;NO GO TO ERROR
1725	005274				ERROR	+1	;CPU ERROR
1726				4\$:			
1727	005274						
1728							
1729	005274	012705	177777				
1730	005300	020527	177777				
1731	005304	001401					
1732							
1733	005306	104001					
1734	005310	005005					
1735	005312	020527	000000	1\$:	CLR	R5	;R5=0
1736	005316	001401			CMP	R5,#0	;DOES R5=0
1737					BEQ	2\$	;YES GO ON
1738	005320	104001					;NO GO TO ERROR
1739	005322	012705	125252		ERROR	+1	;CPU ERROR
1740	005326	020527	125252	2\$:	MOV	#125252,R5	;R5=125252
1741	005332	001401			CMP	R5,#125252	;DOES R5=125252
1742					BEQ	3\$	;YES GO ON
1743	005334	104001					;NO GO TO ERROR
1744	005336	012705	052525		ERROR	+1	;CPU ERROR
1745	005342	020527	052525	3\$:	MOV	#52525,R5	;R5=52525
1746	005346	001401			CMP	R5,#52525	;DOES R5=52525
1747					BEQ	4\$	;YES GO ON
1748	005350	104001					;NO GO TO ERROR
1749	005352				ERROR	+1	;CPU ERROR
1750				4\$:			
1751	005352						


```

1752      ;      R6 BIT TESTS
1753 005352 012706 177777      MOV      #177777,R6      ;R6=177777
1754 005356 020627 177777      CMP      R6,#177777      ;DOES R6=177777
1755 005362 001401              BEQ      1$              ;YES GO ON
1756              ;NO GO TO ERROR
1757 005364 104001              ERROR    +1              ;CPU ERROR
1758 005366 005006              1$:      CLR      R6              ;R6=0
1759 005370 020627 000000      CMP      R6,#0              ;DOES R6=0
1760 005374 001401              BEQ      2$              ;YES GO ON
1761              ;NO GO TO ERROR
1762 005376 104001              ERROR    +1              ;CPU ERROR
1763 005400 012706 125252      2$:      MOV      #125252,R6      ;R6=125252
1764 005404 020627 125252      CMP      R6,#125252      ;DOES R6=125252
1765 005410 001401              BEQ      3$              ;YES GO ON
1766              ;NO GO TO ERROR
1767 005412 104001              ERROR    +1              ;CPU ERROR
1768 005414 012706 052525      3$:      MOV      #52525,R6      ;R6=52525
1769 005420 020627 052525      CMP      R6,#52525      ;DOES R6=52525
1770 005424 001401              BEQ      4$              ;YES GO ON
1771              ;NO GO TO ERROR
1772 005426 104001              ERROR    +1              ;CPU ERROR
1773 005430 012706 001000      4$:      MOV      #STBOT,R6      ;RESTORE SP
1774
1775      ;
1776 005434      PSWBTS:
1777      ; PSW LOW BYTE BIT TESTS
1778 005434 012737 000377 177776      MOV      #377,#177776      ;PS=357 T BIT SHOULDN'T SET
1779 005442 022737 000357 177776      CMP      #357,#177776      ;DOES PS=357
1780 005450 001401              BEQ      1$              ;YES GO ON
1781              ;NO GO TO ERROR
1782 005452 104001              ERROR    +1              ;CPU ERROR
1783 005454 005037 177776              1$:      CLR      #177776      ;PS=0
1784 005460 022737 000000 177776      CMP      #0,#177776      ;DOES PS=0
1785 005466 001401              BEQ      2$              ;YES GO ON
1786              ;NO GO TO ERROR
1787 005470 104001              ERROR    +1              ;CPU ERROR
1788 005472 012737 000105 177776      2$:      MOV      #105,#177776      ;PS=105
1789 005500 022737 000105 177776      CMP      #105,#177776      ;DOES PS=105
1790 005506 001401              BEQ      3$              ;YES GO ON
1791              ;NO GO TO ERROR
1792 005510 104001              ERROR    +1              ;CPU ERROR
1793 005512 012737 000252 177776      3$:      MOV      #252,#177776      ;PS=252
1794 005520 022737 000252 177776      CMP      #252,#177776      ;DOES PS=252
1795 005526 001401              BEQ      4$              ;YES GO ON
1796              ;NO GO TO ERROR
1797 005530 104001              ERROR    +1              ;CPU ERROR
1798 005532              4$:
1799
1800      NSPO:
1801
1802      ; TEST SINGLE OPERAND INSTRUCTIONS MODE 0
1803      ;*****
1804      ;THE INC, COM, CLR, AND DECREMENT INSTRUCTIONS ARE VERIFIED.
1805 005532 005004              CLR      R4              ;INITIALIZE R4 WITH DATA
1806 005534 005106              COM      R4              ;
1807 005536 005004              CLR      R4              ;TEST INSTRUCTION
  
```

K3

CORDBO RDJ11-B CLUSTER MACY11 30(1046) 05 APR-84 16:45 PAGE 37
COKDAB.P11 05-APR-84 16:45 BASE INSTRUCTION SET TESTS

SEQ 0036

1808	005540	001401		BEQ	2\$		;BRANCH IF R4 CLEARED
1809	005542	104001	1\$:	ERROR	+1		;CPU ERROR
1810	005544	005104	2\$:	COM	R4		;*TEST COMPLIMENT INSTRUCTION
1811	005546	005204		INC	R4		;*TEST INCREMENT INSTRUCTION
1812	005550	001401		BEQ	4\$		;BRANCH IF R4 =0
1813							;COMPLIMENT OR INCREMENT FAILED
1814	005552	104001	3\$:	ERROR	+1		;CPU ERROR
1815	005554		4\$:				
1816							
1817							
1818	005554						
1819							
1820							
1821	005554	005004		CLR	R4		;TEST SINGLE OPS - EVEN BYTE OF CLRB, DECB, AND COMB
1822	005556	005104		COM	R4		;SETUP TEST REGISTER
1823	005560	105004		CLRB	R4		;*TEST CLEAR BYTE INSTRUCTION
1824	005562	001401		BEQ	2\$		;BRANCH IF GOOD
1825							;CLEAR EVEN BYTE FAILED
1826	005564	104001	1\$:	ERROR	+1		;CPU ERROR
1827	005566	105304	2\$:	DECB	R4		;*TEST DECREMENT BYTE
1828	005570	100002		BPL	3\$		;DECREMENT BYTE FAILED
1829	005572	105104		COMB	R4		;*TEST COMPLIMENT BYTE
1830	005574	001401		BEQ	4\$		;BRANCH IF GOOD
1831							;COMPLIMENT OR DECREMENT FAILED TO WORK
1832	005576	104001	3\$:	ERROR	+1		;CPU ERROR
1833	005600		4\$:				
1834							
1835							
1836	005600						
1837							
1838	005600	005004		CLR	R4		;TEST SINGLE OPS - MODE 1 CLRB, COMB, AND INCB
1839	005602	005014		CLR	(R4)		
1840	005604	005114		COM	(R4)		;SETUP TEST DATA
1841	005606	005014		CLR	(R4)		;*TEST INSTRUCTION
1842	005610	001401		BEQ	2\$		;BRANCH IF GOOD
1843							;MODE 1 FAILED
1844	005612	104001	1\$:	ERROR	+1		;CPU ERROR
1845	005614	005114	2\$:	COM	(R4)		;*TEST INSTRUCTION
1846	005616	001403		BEQ	3\$		; (C) SHOULD * -1
1847	005620	100002		BPL	3\$		
1848	005622	005214		INC	(R4)		;*TEST INSTRUCTION
1849	005624	001401		BEQ	4\$		;BRANCH IF GOOD
1850							;COM OR INC FAILED TO ALTER LOC 0 CORRECTLY
1851	005626	104001	3\$:	ERROR	+1		;CPU ERROR
1852	005630		4\$:				
1853							
1854							
1855	005630						
1856							
1857							
1858	005630	005004		CLR	R4		;TEST SINGLE OPS MODE1-EVEN BYTE CLRB,COMB,INCB
1859	005632	005014		CLR	(R4)		
1860	005634	005114		COM	(R4)		;SETUP TEST DATA
1861	005636	105014		CLRB	(R4)		;*TEST INSTRUCTION
1862	005640	105014		CLRB	(R4)		;*TEST INSTRUCTION
1863	005642	001401		BEQ	2\$		;BRANCH IF GOOD


```

1864
1865 005644 104001 1$: ERROR +1 ;CLEAR (0) EVEN BYTE FAILED
1866 005646 105214 2$: INCB (R4) ;CPU ERROR
1867 005650 100405 BMI 3$ ;*TEST INSTRUCTION
1868 005652 001404 BEQ 3$ ; TEST FLAGS
1869 005654 105114 COMB (R4) ;*TEST INSTRUCTION
1870 005656 105214 INCB (R4)
1871 005660 105214 INCB (R4)
1872 005662 001401 BEQ 4$ ;BRANCH IF GOOD
1873 ;COMB OR INCB FAILED
1874 005664 104001 3$: ERROR +1 ;CPU ERROR
1875 005666 4$:
1876
1877
1878 005666 ;MSPE0:
1879
1880 ; TEST SINGLE OPS - ODD BYTE - CLRB, COMB, DECB
1881 005666 005004 CLR R4
1882 005670 005014 CLR (R4)
1883 005672 005114 COM (R4) ;SETUP TEST DATA
1884 005674 005204 INC R4 ;POINT TO ODD BYTE
1885 005676 105014 CLRB (R4) ;*TEST INSTRUCTION
1886 005700 001401 BEQ 1$ ;BRANCH IF GOOD
1887 ;CLEAR ODD BYTE FAILED
1888 005702 104001 1$: ERROR +1 ;CPU ERROR
1889 005704 005304 DEC R4 ;POINT TO EVEN BYTE
1890 005706 005214 JNC (R4) ;LOC 0+1 0
1891 005710 005204 INC R4 ;POINT TO ODD BYTE
1892 005712 105114 COMB (R4) ;*TEST INSTRUCTION
1893 005714 105214 INCB (R4) ;LOC 0+1 0
1894 005716 100003 RPL 2$ ;BRANCH IF ERROR
1895 005720 001402 BEQ 2$ ;
1896 005722 105214 INCB (R4) ;*TEST INSTRUCTION
1897 005724 001401 BEQ 3$ ;BRANCH IF GOOD
1898 ;MODE 1, ODD BYTE FAILED
1899 005726 104001 2$: ERROR +1 ;CPU ERROR
1900 005730 3$:
1901
1902
1903 005730 ;MSPF:
1904
1905 005730 005004 ; TEST SINGLE OP - MODE 2 - CLR, COM, INC
1906 005732 105104 COMB R4
1907 005734 005204 INC R4 ;R4=400
1908 005736 005014 CLR (R4) ;400=0
1909 005740 005114 COM (R4) ;400=1
1910 005742 005024 CLR (R4) ;*TEST INSTRUCTION
1911 005744 001401 BEQ 1$ ;BRANCH IF GOOD
1912 ;MODE 2 CLEAR FAILED
1913 005746 104001 1$: ERROR +1 ;CPU ERROR
1914 005750 005304 DEC R4
1915 005752 005304 DEC R4 ;R4=400
1916 005754 005124 COM (R4) ;*TEST INSTRUCTION
1917 005756 100004 RPL 2$ ;BRANCH IF FAILURE
1918 005760 005304 DEC R4
1919 005762 005304 DEC R4 ;R4 400

```

COKDABO KDJ11 B CLUSTER MAC111 30(1046) 05-APR-84 16:45 PAGE 39
COKDAB.P11 05-APR-84 16:45 BASE INSTRUCTION SET TESTS

SLQ 0038

```

1920 005764 005224      INC      (R4)+      ;*TEST INSTRUCTION
1921 005766 001401      BEQ      3$      ;BRANCH IF GOOD
1922                                ;MODE 2 FAILURE
1923 005770 104001      2$:      ERROR    +1      ;CPU ERROR
1924 005772      3$:
1925
1926      ;
1927 005772      ; MSPG:
1928
1929      ;      TEST CLRB, COMB, DECB, MODE 2 - EVEN BYTE
1930 005772 005004      CLR      R4
1931 005774 105104      COMB     R4
1932 005776 005204      INC      R4      ;R4=400
1933 006000 005014      CLR      (R4)
1934 006002 005114      COM      (R4)      ;400=-1
1935 006004 105024      CLRB     (R4)+      ;*TEST INSTRUCTION
1936 006006 001401      BEQ      1$      ;BRANCH IF GOOD
1937                                ;MODE 2 EVEN BYTE FAILED
1938 006010 104001      1$:      ERROR    +1      ;CPU ERROR
1939 006012 005304      DEC      R4
1940 006014 105324      DECB     (R4)+      ;*TEST INSTRUCTION
1941 006016 100003      BPL      2$      ;BRANCH IF BAD
1942 006020 005304      DEC      R4      ;POINT TO EVEN BYTE
1943 006022 105124      COMB     (R4)+      ;*TEST INSTRUCTION
1944 006024 001401      BEQ      3$      ;BRANCH IF GOOD
1945                                ;MODE 2, EVEN BYTE FAILED
1946 006026 104001      2$:      ERROR    +1      ;CPU ERROR
1947 006030      3$:
1948
1949      ;
1950 006030      ; MSPH:
1951
1952      ;      TEST CLRB, COMB, INCB MODE 2 - ODD BYTE
1953 006030 005004      CLR      R4
1954 006032 105104      COMB     R4
1955 006034 005204      INC      R4      ;R4=400
1956 006036 005014      CLR      (R4)
1957 006040 005114      COM      (R4)      ;400=-1 -1
1958 006042 005214      INC      (R4)      ;POINT TO ODD BYTE
1959 006044 105024      CLRB     (R4)+      ;*TEST INSTRUCTION
1960 006046 001401      BEQ      1$      ;BRANCH IF GOOD
1961                                ;MODE 2, ODD BYTE FAILED
1962 006050 104001      1$:      ERROR    +1      ;CPU ERROR
1963 006052 005304      DEC      R4      ;POINT TO ODD BYTE
1964 006054 005304      DEC      R4
1965 006056 105224      INCB     (R4)+      ;400=1 0
1966 006060 105124      COMB     (R4)+      ;*TEST INSTRUCTION
1967 006062 100003      BPL      2$      ;BRANCH IF MODE 2 FAILED
1968 006064 005304      DEC      R4      ;POINT TO ODD BYTE
1969 006066 105224      INCB     (R4)+      ;BRANCH IF GOOD
1970 006070 001401      BEQ      3$      ;MODE 2, ODD BYTE FAILED
1971                                ;CPU ERROR
1972 006072 104001      2$:      ERROR    +1
1973 006074      3$:
1974
1975      ;

```

COKDAB0 KDJ11-8 CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 40
 COKDAB.P11 05-APR-84 16:45 BASE INSTRUCTION SET TESTS

SEQ 0039

```

1976 006074 MSPI:
1977
1978 ; TEST CLR, COM, INC - MODE 3
1979 006074 005004 CLR R4 ;
1980 006076 005014 CLR (R4) ;0=0
1981 006100 105114 COMB (R4) ;
1982 006102 005214 INC (R4) ;0=400
1983 006104 005034 CLR @ (R4)+ ;*TEST INSTRUCTION
1984 006106 001401 BEQ 1$ ;BRANCH IF GOOD
1985 ;MODE 3 FAILED, 400 SHOULD=0
1986 006110 104001 1$: ERROR +1 ;CPU ERROR
1987 006112 005304 DEC R4
1988 006114 005304 DEC R4 ;R4=0
1989 006116 005134 COM @ (R4)+ ;*TEST INSTRUCTION
1990 006120 100004 BPL 2$ ;BRANCH IF BAD
1991 006122 005304 DEC R4
1992 006124 005304 DEC R4 ;REPOSITION POINTER
1993 006126 005234 INC @ (R4)+ ;*TEST INSTRUCTION
1994 006130 001401 BEQ 3$ ;BRANCH IF GOOD
1995 ;MODE 3 FAILED
1996 006132 104001 2$: ERROR +1 ;CPU ERROR
1997 006134 3$:
1998
1999 ;
2000 006134 MSPJ:
2001
2002 ; TEST CLRB, COMB, INCB - MODE 3, EVEN/ODD BYTE
2003 006134 005004 CLR R4 ;R4=0
2004 006136 005001 CLR R1
2005 006140 105101 COMB R1
2006 006142 005201 INC R1 ;R1=400
2007 006144 005011 CLR (R1)
2008 006146 005121 COM (R1)+ ;400=-1
2009 006150 005011 CLR (R1)
2010 006152 105111 COMB (R1) ;402=000 377
2011 006154 005014 CLR (R4)
2012 006156 105114 COMB (R4)
2013 006160 005214 INC (R4) ;0=400
2014 006162 105034 CLRB @ (R4)+ ;*TEST INSTRUCTION 400=377 000
2015 006164 001401 BEQ 1$ ;BRANCH IF MODE 3 EVEN BYTE CLEARED
2016 ; TEST INSTRUCTION FAILED
2017 006166 104001 1$: ERROR +1 ;CPU ERROR
2018 006170 005304 DEC R4 ;REPOSITION POINTER
2019 006172 005304 DEC R4
2020 006174 105134 COMB @ (R4)+ ;*TEST INSTRUCTION
2021 006176 005304 DEC R4
2022 006200 005304 DEC R4 ;REPOSITION POINTER
2023 006202 105234 INCB @ (R4)+ ;*TEST INSTRUCTION
2024 006204 001401 BEQ 3$ ;BRANCH IF GOOD
2025 ;MODE 3, EVEN BYTE FAILED
2026 006206 104001 2$: ERROR +1 ;CPU ERROR
2027 006210 005304 3$: DEC R4
2028 006212 005304 DEC R4
2029 006214 005214 INC (R4) ;R4=401
2030 006216 105234 INCB @ (R4)+ ;*TEST INSTRUCTION
2031 006220 001004 BNE 4$ ;BRANCH IF 402 NEQ 0

```

CORDBO RDJ11 B CLUSTER MACY11 30(1046) 05 APR 84 16:45 PAGE 41
 CORDAB,P11 05-APR 84 16:45 BASE INSTRUCTION SET TESTS

SEQ 0040

2032	006222	005304		DEC	R4	
2033	006224	005304		DEC	R4	;R4=401
2034	006226	105034		CLRB	8(R4),	;401=0
2035	006230	001401		BEQ	5:	;BRANCH IF GOOD
2036						;ODD BYTE FAILED
2037	006232	104001	4:	ERROR	+1	;CPU ERROR
2038	006234	005304	5:	DEC	R4	
2039	006236	005304		DEC	R4	;R4=401
2040	006240	105134		COMB	8(R4),	;403=377
2041	006242	005304		DEC	R4	
2042	006244	005304		DEC	R4	
2043	006246	105234		INCB	8(R4),	;*TEST INSTRUCTION
2044	006250	001401		BEQ	7:	;BRANCH IF GOOD
2045						;MODE3 ODD BYTE FAILED.
2046	006252	104001	6:	ERROR	+1	;CPU ERROR
2047	006254		7:			
2048						
2049						
2050	006254					
2051						
2052						
2053	006254	005004		TEST CLR, COM, DEC - MODE 4		
2054	006256	105104		CLR	R4	
2055	006260	005204		COMB	R4	
2056	006262	005014		INC	R4	;R4=400
2057	006264	005124		CLR	(R4)	
2058	006266	005014		COM	(R4),	;400=1
2059	006270	005224		CLR	(R4)	
2060	006272	005044		INC	(R4),	;402=1
2061	006274	001401		CLR	(R4)	;*TEST INSTRUCTION
2062				BEQ	1:	;BRANCH IF GOOD
2063	006276	104001				;MODE 4 FAILED
2064	006300	005344	1:	ERROR	+1	;CPU ERROR
2065	006302	005114		DEC	(R4)	;*TEST INSTRUCTION 400=1
2066	006304	001405		COM	(R4)	;400=1
2067	006306	100404		BEQ	2:	;BRANCH IF BAD
2068	006310	005204		BMI	2:	;BRANCH IF BAD
2069	006312	005204		INC	R4	
2070	006314	005344		INC	R4	;R4=400
2071	006316	001401		DEC	(R4)	;*TEST INSTRUCTION
2072				BEQ	3:	;BRANCH IF GOOD
2073	006320	104001	2:	ERROR	+1	;MODE 4 FAILED
2074	006322		3:			;CPU ERROR
2075						
2076						
2077	006322					
2078						
2079						
2080	006322	005004		TEST COMB, INCH, CLRB - MODE 4, ODD BYTE		
2081	006324	105104		CLR	R4	
2082	006326	005204		COMB	R4	
2083	006330	005044		INC	R4	;R4=400
2084	006332	105114		CLR	(R4)	;400=0
2085	006334	005224		COMB	(R4)	
2086	006336	005014		INC	(R4),	;400=001 000
2087	006340	005124		CLR	(R4)	
				COM	(R4),	;400=1

C4

SEQ 0041

2088	006342	005204		INC	R4		
2089	006344	105044		CLRB	(R4)		
2090	006346	001401		BEQ	25		
2091							
2092	006350	104001	15:	ERROR	+1		
2093	006352	005204	25:	INC	R4		
2094	006354	005204		INC	R4		
2095	006356	105144		COMB	(R4)		
2096	006360	005304		DEC	R4		
2097	006362	005304		DEC	R4		
2098	006364	105244		INCB	(R4)		
2099	006366	001401		BEQ	45		
2100							
2101	006370	104001	35:	ERROR	+1		
2102	006372	105344	45:	DECB	(R4)		
2103	006374	001401		BEQ	65		
2104							
2105	006376	104001	55:	ERROR	+1		
2106	006400		65:				
2107							
2108							
2109	006400						
2110							
2111							
2112	006400	005004		TEST CLR, COM, INC	MODE 5		
2113	006402	005014		CLR	R4		
2114	006404	105114		CLR	(R4)		
2115	006406	005224		COMB	(R4)		
2116	006410	005054		INC	(R4)+		
2117	006412	001401		CLR	2 (R4)		
2118				BEQ	15		
2119	006414	104001					
2120	006416	005204	15:	ERROR	+1		
2121	006420	005204		INC	R4		
2122	006422	005154		INC	R4		
2123	006424	001401		COM	2 (R4)		
2124	006426	005204		BEQ	25		
2125	006430	005204		INC	R4		
2126	006432	005354		INC	R4		
2127	006434	001403		DEC	2 (R4)		
2128	006436	005224		BEQ	25		
2129	006440	105254		INC	(R4)+		
2130	006442	001401		INCB	2 (R4)		
2131				BEQ	35		
2132	006444	104001	25:	ERROR	+1		
2133	006446		35:				
2134							
2135							
2136	006446						
2137							
2138							
2139	006446	005004		TEST NEG MODE 5			
2140	006450	105104		CLR	R4		
2141	006452	005204		COMB	R4		
2142	006454	005001		INC	R4		
2143	006456	105101		CLR	R1		
				COMB	R1		

R4=403
 TEST INST. CLEAR ODD BYTE (401)
 BRANCH IF GOOD
 MODE 4 BYTE FAILED
 CPU ERROR

R4=403
 TEST INST. 401=377

TEST INST. 401=0
 BRANCH IF GOOD
 MODE 4 ODD BYTE FAILED
 CPU ERROR
 TEST INST.
 BRANCH IF GOOD
 MODE 4 DECREMENT ODD BYTE FAILED
 CPU ERROR

IO=400
 TEST INST. 400=0
 BRANCH IF GOOD
 MODE 5 FAILED
 CPU ERROR
 RESET POINTER TO 0
 TEST INST. 376=1
 BRANCH IF BAD
 REPOSITION POINTER
 TEST INST. 376=0
 BRANCH IF BAD
 IO=401 R4=2
 TEST INST. 400=0 376
 BRANCH IF GOOD
 MODE 5 FAILED
 CPU ERROR

R4=400

2144	006460	005301		DEC	R1		R1=376
2145	006462	005002		CLR	R2		R2=0
2146	006464	005012		CLR	(R2)		R0=0
2147	006466	005014		CLR	(R4)		
2148	006470	005114		COM	(R4)		R400=-1
2149	006472	005011		CLR	(R1)		R376=0
2150	006474	005454		NEG	0-(R4)		R0=0
2151	006476	001401		BEQ	2\$		BRANCH IF GOOD
2152							NEG FAILED
2153	006500	104001	1\$:	ERROR	+1		CPU ERROR
2154	006502	005334	2\$:	DEC	0(R4)+		R0=1
2155	006504	005454		NEG	0-(R4)		R0=1
2156	006506	001403		BEQ	3\$		BRANCH IF BAD
2157	006510	102402		BVS	3\$		BRANCH IF BAD
2158	006512	100401		BMI	3\$		BRANCH IF BAD
2159	006514	103401		BCS	4\$		BRANCH IF GOOD
2160							NEG FAILED
2161	006516	104001	3\$:	ERROR	+1		CPU ERROR
2162	006520	005334	4\$:	DEC	0(R4)+		TEST RESULT OF NEGATE
2163	006522	001401		BEQ	6\$		BRANCH IF GOOD
2164							RESULT OF NEGATE BAD
2165	006524	104001	5\$:	ERROR	+1		CPU ERROR
2166	006526	105212	6\$:	INCB	(R2)		R0=1
2167	006530	005454		NEG	0-(R4)		R0=-1
2168	006532	001403		BEQ	7\$		
2169	006534	102402		BVS	7\$		
2170	006536	103001		BCC	7\$		
2171	006540	100401		BMI	8\$		BRANCH IF GOOD
2172							BAD NEGATE
2173	006542	104001	7\$:	ERROR	+1		CPU ERROR
2174	006544	105212	8\$:	INCB	(R2)		R0=0
2175	006546	001401		BEQ	10\$		BRANCH IF GOOD
2176	006550	104001	9\$:	ERROR	+1		CPU ERROR
2177	006552		10\$:				
2178							
2179							
2180							
2181	006552		MSPP:				
2182							
2183				TEST CLR, COM, INC	MODE 6		
2184	006552	005004		CLR	R4		
2185	006554	005204		INC	R4		
2186	006556	005204		INC	R4		R4+2
2187	006560	005001		CLR	R1		
2188	006562	105101		COMB	R1		
2189	006564	005201		INC	R1		R1+400
2190	006566	005011		CLR	(R1)		
2191	006570	005121		COM	(R1)+		R400=1
2192	006572	005011		CLR	(R1)		R1=402
2193	006574	005211		INC	(R1)		R402=1
2194	006576	005002		CLR	R2		R2=0
2195	006600	005012		CLR	(R2)		R0=0
2196	006602	005064	000376	CLR	376(R4)		R400=0
2197	006606	001401		BEQ	2\$		BRANCH IF GOOD
2198	006610	104001	1\$:	ERROR	+1		CPU ERROR
2199	006612	005364	2\$:	DEC	376(R4)		R400=-1

2200	006616	005164	000400		COM	400(R4)	;402=-1
2201	006622	001405			BEQ	3\$	;BRANCH IF BAD
2202	006624	005264	000400		INC	400(R4)	;402=-2
2203	006630	005264	000400		INC	400(R4)	
2204	006634	001401			BEQ	4\$	;BRANCH IF GOOD
2205							;MODE 6 FAILED
2206	006636	104001		3\$:	ERROR	+1	;CPU ERROR
2207	006640	005261	177776	4\$:	INC	-2(R1)	;400=0
2208	006644	001401			BEQ	6\$	;BRANCH IF GOOD
2209							;INC MODE 6 FAILED
2210	006646	104001		5\$:	ERROR	+1	;CPU ERROR
2211	006650			6\$:			
2212							
2213				:			
2214				:			
2215				:			
2216	006650			MSPQ;			
2217							
2218				:	TEST NEG MODE 6		
2219	006650	005001			CLR	R1	;R1=0
2220	006652	005004			CLR	R4	
2221	006654	105104			COMB	R4	
2222	006656	005204			INC	R4	;R4=400
2223	006660	005014			CLR	(R4)	:
2224	006662	005114			COM	(R4)	;400=-1
2225	006664	005044			CLR	-(R4)	;376=0
2226	006666	005044			CLR	-(R4)	:
2227	006670	005224			INC	(R4)+	;374=1 R4=376
2228	006672	005464	000002		NEG	2(R4)	;400-1
2229	006676	001403			BEQ	1\$	;NEGATE FAILED
2230	006700	102402			BVS	1\$	
2231	006702	100401			BMI	1\$	
2232	006704	103401			BOS	2\$	;BRANCH IF GOOD
2233							;NEGATE FAILED
2234	006706	104001		1 :	ERROR	+1	;CPU ERROR
2235	006710	005364	000002	2\$:	DEC	2(R4)	; TEST RESULT OF NEGATE
2236	006714	001401			BEQ	4\$	;BRANCH IF GOOD
2237							;RESULT OF NEGATE FAILED
2238	006716	104001		3\$:	ERROR	+1	;CPU ERROR
2239	006720	005464	000000	4\$:	NEG	0(R4)	;+0=0
2240	006724	001401			BEQ	5\$	; BRANCH IF GOOD
2241							;NEGATE FAILED
2242	006726	104001			ERROR	+1	;CPU ERROR
2243	006730	105461	000374	5\$:	NEGB	374(R1)	;374=0 377
2244	006734	102403			BVS	6\$	:
2245	006736	001402			BEQ	6\$	
2246	006740	100901			BPL	6\$	
2247	006742	103401			BOS	7\$	;BRANCH IF GOOD
2248							;NEGATE FAILED
2249	006744	104001		6\$:	ERROR	+1	;CPU ERROR
2250	006746	105261	000374	7\$:	INCH	374(R1)	;374=0
2251	006752	001401			BEQ	9\$	;BRANCH IF GOOD
2252							;NEGATE FAILED
2253	006754	104001		8\$:	ERROR	+1	;CPU ERROR
2254	006756			9\$:			
2255							

CORDBO R0J11 R CLUSTER MACY11 SO(1046) 05 APR 84 16:45 PAGE 45
 CORDAB.P11 05-APR-84 16:45 BASE INSTRUCTION SET TESTS

SEQ 0044

```

2256
2257 006756      ; MSPR:
2258
2259      ; TEST CLR, COM, INC - MODE 7
2260 006756 005001 CLR R1 ;R1=0
2261 006760 005004 CLR R4 ;
2262 006762 105104 COMB R4 ;
2263 006764 005204 INC R4 ;R4=400
2264 006766 005011 CLR (R1)
2265 006770 105111 COMB (R1)
2266 006772 005211 INC (R1)
2267 006774 005211 INC (R1)
2268 006776 005211 INC (R1) ;:0=402
2269 007000 005014 CLR (R4) ;400=0
2270 007002 005064 000002 CLR 2(R4) ;
2271 007006 005164 000002 COM 2(R4) ;402=-1
2272 007012 005074 177400 CLR @-400(R4) ;402=0
2273 007016 001401 BEQ 2$ ;BRANCH IF GOOD
2274 007020 104001 1$: ERROR +1 ;CPU ERROR
2275 ;INSTRUCTION FAILED
2276 007022 005171 000000 2$: COM @0(R1) ;402=-1
2277 007026 100401 BMI 4$ ;BRANCH IF GOOD
2278 007030 104001 3$: ERROR +1 ;CPU ERROR
2279
2280 007032 005104 4$: COM R4
2281 007034 005274 000401 INC @401(R4) ;402=0
2282 007040 001401 BEQ 6$ ;BRANCH IF GOOD
2283 007042 104001 5$: ERROR +1 ;CPU ERROR
2284 ;MODE 7 FAILED
2285 007044 6$:
2286
2287 ; MSPS:
2288 007044
2289
2290      ; TEST NEG MODE 7
2291 007044 005004 CLR R4
2292 007046 005014 CLR (R4) ;0=0
2293 007050 005002 CLR R2 ;
2294 007052 105102 COMB R2
2295 007054 005202 INC R2 ;R2=400
2296 007056 005012 CLR (R2) ;400=0
2297 007060 005472 177400 NEG @-400(R2) ;NEG OF 0=0
2298 007064 103401 BCS 1$ ;****
2299 007066 001401 BEQ 2$ ;BRANCH IF GOOD
2300 007070 101001 1$: ERROR +1 ;CPU ERROR
2301
2302 007072 005314 2$: DEC (R4) ;0=-1
2303 007074 005474 000400 NEG @-400(R4) ;0=1
2304 007100 001403 BEQ 3$ ;BRANCH IF ERROR
2305 007102 102402 BVS 3$ ;
2306 007104 100401 BMI 3$ ;
2307 007106 103401 BCS 4$ ;BRANCH IF GOOD
2308 007110 104001 3$: ERROR +1 ;CPU ERROR
2309 ;NEGATE MODE 7 FAILED
2310 007112 4$:
2311

```


(54)

SEQ 0045

```

2312
2313 007112      ; MSPT:
2314
2315      ;      TEST SINGLE OPERAND MODE 2 REG 7
2316 007112 005004 CLR      R4
2317 007114 105104 COMB     R4
2318 007116 005204 INC      R4
2319 007120 005027 CLR      (R7)+
2320 007122 177777 1$:     .WORD  -1
2321 007124 001401 BEQ      3$
2322 007126 104001 2$:     ERROR  +1
2323 007130      3$:
2324
2325
2326
2327 007130      MSPU:
2328
2329      ;      TEST TST MODE 0
2330 007130 005004 CLR      R4
2331 007132 000277 SCC
2332 007134 000244 CLZ
2333 007136 005704 TST      R4
2334 007140 103403 BCS      1$
2335 007142 102402 BVS      1$
2336 007144 100401 BMI      1$
2337 007146 001401 BEQ      2$
2338 007150 104001 1$:     ERROR  +1
2339
2340 007152 005304 2$:     DEC      R4
2341 007154 000277 SCC
2342 007156 000250 CLN
2343 007160 005704 TST      R4
2344 007162 103403 BCS      3$
2345 007164 102402 BVS      3$
2346 007166 001401 BEQ      3$
2347 007170 100401 BMI      4$
2348 007172 104001 3$:     ERROR  +1
2349
2350 007174      4$:
2351
2352
2353 007174      MSPV0:
2354
2355      ;      TEST TST MODE 0 BYTE
2356 007174 005004 CLR      R4
2357 007176 105104 COMB     R4
2358 007200 000277 SCC
2359 007202 000250 CLN
2360 007204 105704 TSTB     R4
2361 007206 102403 BVS      1$
2362 007210 103402 BCS      1$
2363 007212 102401 BVS      1$
2364 007214 100401 BMI      2$
2365 007216 104001 1$:     ERROR  +1
2366
2367 007220 005204 2$:     INC      R4

```

;R4=400
 ;CLEAR NEXT LOCATION
 ;SETUP INITIAL DATA
 ;BRANCH IF GOOD
 ;CPU ERROR

;R4=0
 ;CONDITION CODES =1111
 ;CC=1011
 ;*TEST INSTRUCTION

;BRANCH IF ERROR
 ;BRANCH IF GOOD
 ;CPU ERROR
 ;TST MODE 0 FAILED
 ;R4=-1

;CC=0111
 ;*TEST INSTRUCTION MODE 0
 ;BRANCH IF ERROR

;BRANCH IF GOOD
 ;CPU ERROR
 ;TST FAILED

;0=000 3??

;CC=0111
 ;*TEST INSTRUCTION ON EVEN BYTE
 ;BRANCH IF ERROR

;BRANCH IF GOOD
 ;CPU ERROR

;POINT TO 1

CORRADO KDJ11-B CLUSTER MAC111 3001046) 05 APR-84 16:45 PAGE 47
 CORRADO.P11 05-APR-84 16:45 BASE INSTRUCTION SET TESTS

SEQ 0046

2368	007222	105704	TSTB	R4	; TEST INSTRUCTION
2369	007224	001401	BEQ	4\$	; BRANCH IF GOOD
2370	007226	104001	3\$: ERROR	+1	; CPU ERROR
2371					; TST FAILED ON BYTE
2372	007230		4\$:		
2373					
2374					
2375	007230				
2376					
2377					
2378	007230	005004		TEST TST MODE 1	
2379	007232	005014	CLR	R4	
2380	007234	000277	CLR	(R4)	; 0=0
2381	007236	000244	SCC		
2382	007240	005714	CLZ		; CC=1011
2383	007242	103403	TST	(R4)	; *TEST INSTRUCTION IN MODE 1
2384	007244	102402	BCS	1\$	; BRANCH IF ERROR
2385	007246	100401	BVS	1\$	
2386	007250	001401	BMI	1\$	
2387	007252	104001	BEQ	2\$	; BRANCH IF GOOD
2388			1\$: ERROR	+1	; CPU ERROR
2389	007254	005214	2\$: INC	(R4)	; 0=1
2390	007256	000277	SCC		
2391	007260	005714	TST	(R4)	; TEST INSTRUCTION
2392	007262	001403	BEQ	3\$	; BRANCH IF ERROR
2393	007264	102402	BVS	3\$	
2394	007266	103401	BCS	3\$	
2395	007270	100001	BPL	4\$	; BRANCH IF GOOD
2396	007272	104001	3\$: ERROR	+1	; CPU ERROR
2397					; TST FAILED MODE 1
2398	007274		4\$:		
2399					
2400					
2401	007274				
2402					
2403					
2404	007274	005004		TEST TST MODE 1 BYTE	
2405	007276	005014	CLR	R4	; R4=0
2406	007300	105114	CLR	(R4)	
2407	007302	005214	COMB	(R4)	
2408	007304	000277	INC	(R4)	; 0=001 000
2409	007306	000244	SCC		
2410	007310	105714	CLZ		; CC=1011
2411	007312	103403	TSTB	(R4)	; *TEST INSTRUCTION
2412	007314	102402	BCS	1\$	; BRANCH IF ERROR
2413	007316	100401	BVS	1\$	
2414	007320	001401	BMI	1\$	
2415	007322	104001	BEQ	2\$	; BRANCH IF GOOD
2416			1\$: ERROR	+1	; CPU ERROR
2417	007324	005204	2\$: INC	R4	; R4=1
2418	007326	000277	SCC		
2419	007330	105714	TSTB	(R4)	; TEST INSTRUCTION
2420	007332	001403	BEQ	3\$	; BRANCH IF ERROR
2421	007334	100402	BMI	3\$	
2422	007336	102401	BVS	3\$	
2423	007340	103001	SCC	4\$	; BRANCH IF GOOD

CORRADO R0J11 B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 48
 COKDAB.P11 05-APR-84 16:45 BASE INSTRUCTION SET TESTS

SEQ 0047

```

2424 007342 104001 3$: ERROR +1 ;CPU ERROR
2425
2426 007344 4$:
2427
2428 ;
2429 007344 ;MSPY:
2430
2431 ; TEST TST MODE 2
2432 007344 005004 CLR R4 ;
2433 007346 005024 CLR (R4)+ ;0-0
2434 007350 005014 CLR (R4)
2435 007352 005114 COM (R4) ;2+1
2436 007354 005004 CLR R4 ;R4+0
2437 007356 000277 SCC ;
2438 007360 000244 CLZ ;CC=1011
2439 007362 005724 TST (R4)+ ; TEST INSTRUCTION
2440 007364 103403 BCS 1$ ;BRANCH IF ERROR
2441 007366 102402 BVS 1$
2442 007370 100401 BMI 1$
2443 007372 001401 BEQ 2$ ;BRANCH IF GOOD
2444 007374 104001 1$: ERROR +1 ;CPU ERROR
2445 ;MODE 2 TEST FAILED
2446 007376 005724 2$: TST (R4)+ ;TST LOC2
2447 007400 103403 BCS 3$
2448 007402 102402 BVS 3$
2449 007404 001401 BEQ 3$
2450 007406 100401 BMI 4$
2451 007410 104001 3$: ERROR +1 ;CPU ERROR
2452 ;MODE 2 FAILED
2453 007412 4$:
2454
2455 ;
2456 007412 ;MSPZ:
2457
2458 ; TEST TST MODE 2 BYTE
2459 007412 005004 CLR R4
2460 007414 005024 CLR (R4)+ ;
2461 007416 105144 COMB (R4) ;0-377 000
2462 007420 005304 DEC R4 ;R4-0
2463 007422 000277 SCC
2464 007424 000244 CLZ ;CC=1011
2465 007426 105724 TSTB (R4)+ ;
2466 007430 102403 BVS 1$ ;BRANCH IF ERROR
2467 007432 103402 BCS 1$
2468 007434 100401 BMI 1$
2469 007436 001401 BEQ 2$ ;BRANCH IF GOOD
2470 007440 104001 1$: ERROR +1 ;CPU ERROR
2471 ;MODE 2 EVEN BYTE FAILED
2472 007442 000277 2$: SCC
2473 007444 000250 CLN ;CC=0111
2474 007446 105724 TSTB (R4)+ ;
2475 007450 001403 BEQ 3$ ;
2476 007452 103402 BCS 3$
2477 007454 102401 BVS 3$
2478 007456 100401 BMI 4$ ;BRANCH IF GOOD
2479 007460 104001 3$: ERROR +1 ;CPU ERROR

```

```

2480                                     ;MODE 2 ODD BYTE FAILED
2481 007462 4$:
2482
2483
2484 007462 MSPAA:
2485
2486 ; TEST TST MODE 3
2487 007462 005004 CLR R4
2488 007464 005014 CLR (R4)
2489 007466 105114 COMB (R4)
2490 007470 005214 INC (R4) ;0 400
2491 007472 005034 CLR @R4+ ;400=0
2492 007474 005004 CLR R4 ;R4=0
2493 007476 000277 SCC
2494 007500 000244 CLZ ;CC=1011
2495 007502 005734 TST @R4+ ; TEST MODE 3
2496 007504 103403 BCS 1$ ;BRANCH IF ERROR
2497 007506 102402 BVS 1$
2498 007510 100401 BMI 1$
2499 007512 001401 BEQ 2$ ;BRANCH IF GOOD
2500 007514 104001 1$: ERROR +1 ;CPU ERROR
2501                                     ;MODE 3 FAILED
2502 007516 005304 2$: DEC R4
2503 007520 005304 DEC R4 ;R4=0
2504 007522 005334 DEC @R4+ ;400=-1
2505 007524 005004 CLR R4
2506 007526 000277 SCC
2507 007530 000250 CLN ;CC=0111
2508 007532 005734 TST @R4+ ; TEST INSTRUCTION
2509 007534 103403 BCS 3$
2510 007536 001402 BEQ 3$ ;BRANCH IF ERROR
2511 007540 102401 BVS 3$
2512 007542 100401 BMI 4$ ;BRANCH IF GOOD
2513                                     ;ERROR MODE 3
2514 007544 104001 4$: ERROR +1 ;CPU ERROR
2515 007546
2516
2517
2518 007546 MSPBB:
2519
2520 ; TEST TST MODE 3 AUTO-INC
2521 007546 005004 CLR R4
2522 007550 005014 CLR (R4) ;0=0
2523 007552 105114 COMB (R4)
2524 007554 005214 INC (R4) ;0 400
2525 007556 005001 CLR R1
2526 007560 105101 COMB R1
2527 007562 005201 INC R1 ;R1=400
2528 007564 005011 CLR (R1) ;400=0
2529 007566 000277 SCC
2530 007570 005734 TST @R4+ ;400 0
2531 007572 103403 BCS 1$ ;ERROR IF CARRY
2532 007574 102402 BVS 1$ ;ERROR IF OVERFLOW
2533 007576 100401 BMI 1$ ;ERROR IF MINUS
2534 007600 001401 BEQ 2$ ;ERROR IF NOT EQUAL
2535 007602 104001 1$: ERROR +1 ;CPU ERROR

```

Address	Op-Code	Op-Code Hex	Instruction	Comments
2536	007604	005304	2\$: DEC R4	;CC SHOULD = 0100
2538	007606	005304	DEC R4	
2539	007610	005704	TST R4	;SEE IF AUTO-INC WORKED
2540	007612	001401	BEQ 4\$	;ERROR IF R4 NE 0
2541	007614	104001	3\$: ERROR +1	;CPU ERROR
2542				;AUTO-INC FAILED
2543	007616		4\$:	
2544				
2545				
2546	007616		;MSTB3:	
2547				
2548			; TEST TST MODE 3 BYTE	
2549	007616	005004	CLR R4	
2550	007620	005014	CLR (R4)	
2551	007622	105114	COMB (R4)	
2552	007624	005214	INC (R4)	
2553	007626	005214	INC (R4)	;0=401
2554	007630	005001	CLR R1	
2555	007632	105101	COMB R1	
2556	007634	005201	INC R1	;R1=400
2557	007636	005011	CLR (R1)	
2558	007640	005111	COM (R1)	
2559	007642	105011	CLRB (R1)	;400=377 000
2560	007644	105734	TSTB @ (R4)+	;** TEST INSTRUCTION
2561	007646	001403	BEQ 1\$	;ERROR IF EQUAL
2562	007650	103402	BCS 1\$	;ERROR IF CARRY SET
2563	007652	102401	BVS 1\$	;ERROR IF OVERFLOW
2564	007654	100401	BMI 2\$	;BRANCH IF MINUS
2565	007656	104001	1\$: ERROR +1	;CPU ERROR
2566				;CC ERROR
2567	007660	005304	2\$: DEC R4	
2568	007662	005304	DEC R4	
2569	007664	001401	BEQ 4\$	;BRANCH IF AUTO-INC WORKED
2570	007666	104001	3\$: ERROR +1	;CPU ERROR
2571				;AUTO-INC FAILED
2572	007670		4\$:	
2573				
2574				
2575	007670		;MST4:	
2576				
2577			; TEST TST MODE 4	
2578	007670	005004	CLR R4	
2579	007672	005014	CLR (R4)	;0=0
2580	007674	005204	INC R4	
2581	007676	005204	INC R4	;R4=2
2582	007700	000277	SCC	
2583	007702	000244	CLZ	;CC=1011
2584	007704	005744	TST (R4)	;**TEST INSTRUCTION
2585	007706	103403	BCS 1\$	;ERROR IF CARRY
2586	007710	102402	BVS 1\$	;ERROR IF OVERFLOW
2587	007712	100401	BMI 1\$	;ERROR IF MINUS
2588	007714	001401	BEQ 2\$	;BRANCH IF GOOD
2589	007716	104001	1\$: ERROR +1	;CPU ERROR
2590				;CC WRONG
2591	007720	005704	2\$: TST R4	;INSURE CORRECT AUTO-DEC

COKDABO RDJ11 B CLUSTER MACY11 30(1046) 05 APR 84 16:45 PAGE 51
COKDAB.P11 05 APR 84 16:45 BASE INSTRUCTION SET TESTS

SEQ 0050

```

2592 007722 001401      BEQ      4$      ;BRANCH IF GOOD AUTO-DEC
2593                                ;BAD AUTO-DEC
2594 007724 104001      3$:      ERROR    +1      ;CPU ERROR
2595 007726      4$:
2596
2597
2598 007726      ;
2599      MST4B:
2600      ;      TEST TST MODE 4 BYTE
2601 007726 005004      CLR      R4
2602 007730 005014      CLR      (R4)
2603 007732 005114      COM      (R4)
2604 007734 105114      COMB     (R4)      ;0=377 000
2605 007736 000277      SCC
2606 007740 005204      INC      R4
2607 007742 005204      INC      R4      ;R4=2
2608 007744 105744      TSTB     (R4)      ;**TEST INSTRUCTION
2609 007746 001403      BEQ      1$      ;ERROR IF EQUAL TO 0
2610 007750 103402      BCS      1$      ;ERROR IF CARRY
2611 007752 102401      BVS      1$      ;ERROR IF OVERFLOW
2612 007754 100401      BMI      2$      ;BRANCH IF MINUS
2613 007756 104001      1$:      ERROR    +1      ;CPU ERROR
2614                                ;CC SHOULD EQUAL 0100
2615 007760 105744      2$:      TSTB     -(R4)      ;**TEST EVEN BYTE
2616 007762 001401      BEQ      4$      ;BRANCH IF GOOD
2617 007764 104001      3$:      ERROR    +1      ;CPU ERROR
2618                                ;CC SHOULD EQUAL 0100 AND R4=-1
2619 007766      4$:
2620      ;
2621 007766      ;
2622      MST5:
2623      ;      TEST TST MODE 5
2624 007766 005004      CLR      R4
2625 007770 005024      CLR      (R4)+      ;0=0, R4=2
2626 007772 000277      SCC
2627 007774 000244      CLZ
2628 007776 005754      TST      @-(R4)      ;CC=1011
2629 010000 103403      BCS      1$      ;TEST INSTRUCTION
2630 010002 102402      BVS      1$      ;ERROR IF CARRY
2631 010004 100401      BMI      1$      ;ERROR IF OVERFLOW
2632 010006 001401      BEQ      2$      ;ERROR IF MINUS
2633 010010 104001      1$:      ERROR    +1      ;BRANCH IF GOOD
2634                                ;CPU ERROR
2635 010012 005704      2$:      TST      R4      ;CC WRONG, SHOULD = 0100
2636 010014 001401      BEQ      4$      ;BRANCH IF AUTO DEC WORKED
2637 010016 104001      3$:      ERROR    +1      ;CPU ERROR
2638                                ;AUTO-DEC FAILED
2639 010020      4$:
2640      ;
2641      ;
2642 010020      ;
2643      MST5B:
2644      ;      TEST TST MODE 5 BYTE
2645 010020 005004      CLR      R4
2646 010022 005014      CLR      (R4)
2647 010024 105114      COMB     (R4)

```


COKDABO KDJ11-B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 52
COKDAB.P11 05-APR-84 16:45 BASE INSTRUCTION SET TESTS

SEQ 0051

```

2648 010026 005214      INC      (R4)          ;0=400
2649 010030 005034      CLR      @-(R4)+        ;400=0, R4=2
2650 010032 005154      COM      @-(R4)          ;
2651 010034 105134      COMB     @-(R4)+        ;400=377 000 R4=2
2652 010036 105754      TSTB     @-(R4)          ;**TEST INSTRUCTION
2653 010040 103403      BCS      1$           ;ERROR IF CARRY
2654 010042 100402      BMI      1$           ;ERROR IF MINUS
2655 010044 102401      BVS      1$           ;ERROR IF OVERFLOW
2656 010046 001401      BEQ      2$           ;BRANCH IF GOOD
2657 010050 104001      1$:      ERROR    +1       ;CPU ERROR
2658                                     ;CC SHOULD = 0100
2659 010052 005224      2$:      INC      (R4)+        ;0=401
2660 010054 105754      TSTB     @-(R4)          ;**TEST INSTRUCTION
2661 010056 100401      BMI      4$           ;BRANCH IF GOOD
2662                                     ;EVEN BYTE FAILURE
2663 010060 104001      3$:      ERROR    +1       ;CPU ERROR
2664 010062      4$:
2665
2666      ;
2667 010062      MST6:
2668
2669      ;      TEST TST MODE 6
2670 010062 005004      CLR      R4
2671 010064 005014      CLR      (R4)          ;0=0
2672 010066 105104      COMB     R4
2673 010070 005204      INC      R4           ;R4=400
2674 010072 005014      CLR      (R4)
2675 010074 005114      COM      (R4)          ;400=-1
2676 010076 005764 177400      TST      -400(R4)      ;**TEST LOCATION 0
2677 010102 103403      BCS      1$           ;ERROR IF CARRY
2678 010104 102402      BVS      1$           ;ERROR IF OVERFLOW
2679 010106 100401      BMI      1$           ;ERROR IF MINUS
2680 010110 001401      BEQ      2$           ;BRANCH IF ZERO
2681 010112 104001      1$:      ERROR    +1       ;CPU ERROR
2682                                     ;CC ARE WRONG
2683 010114 005004      2$:      CLR      R4
2684 010116 005764 000400      TST      400(R4)      ;TST LOCATION 400
2685 010122 001401      BEQ      3$           ;ERROR IF EQUAL
2686 010124 100401      BMI      4$           ;BRANCH IF MINUS
2687 010126 104001      3$:      ERROR    +1       ;CPU ERROR
2688                                     ;CC ERROR
2689 010130      4$:
2690
2691      ;
2692 010130      MST7:
2693
2694      ;      TEST TST MODE 7
2695 010130 005004      CLR      R4
2696 010132 005014      CLR      (R4)
2697 010134 005124      COM      (R4)+        ;0=1
2698 010136 005014      CLR      (R4)          ;0=0
2699 010140 005002      CLR      R2          ;R2=0
2700 010142 005004      CLR      R4
2701 010144 105104      COMB     R4
2702 010146 005204      INC      R4           ;R4=400
2703 010150 005014      CLR      (R4)          ;400=0

```

N4

COKDABO KDJ11-B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 53
 COKDAB.P11 05-APR-84 16:45 BASE INSTRUCTION SET TESTS

SEQ 0052

```

2704 010152 005774 177402      IST      @-376(R4)      ;**TEST LOCATION 0
2705 010156 103403              BCS      1$          ;ERROR IF CARRY
2706 010160 102402              BVS      1$          ;ERROR IF OVERFLOW
2707 010162 001401              BEQ      1$          ;ERROR IF ZERO
2708 010164 100401              BMI      2$          ;BRANCH IF GOOD
2709 010166 104001              1$:      ERROR      +1      ;CPU ERROR
2710                                ;ERROR IN CC
2711 010170 005222              2$:      INC      (R2)+      ;0=-2 R2=2
2712 010172 005774 177402      TST      @-376(R4)      ;** CHECK CONTENTS OF LOCATION 2
2713 010176 100401              BMI      3$          ;ERROR IF MINUS
2714 010200 001401              BEQ      4$          ;BRANCH IF GOOD
2715 010202 104001              3$:      ERROR      +1      ;CPU ERROR
2716                                ;CC SHOULD = 0100
2717 010204              4$:
2718
2719
2720
2721
2722
2723
2724      .SBTTL ***** DOUBLE OPERAND TESTS *****
2725
2726 010204      MDMO:
2727
2728      ;      TEST MOVE MODE 0
2729 010204 005004      CLR      R4          ;R4=0
2730 010206 005001      CLR      R1
2731 010210 005101      COM      R1          ;R1=-1
2732 010212 010104      MOV      R1,R4      ;**TEST MOVE INSTRUCTION
2733 010214 001402      BEQ      1$          ;ERROR IF R4=0
2734 010216 102401      BVS      1$          ;ERROR IF OVERFLOW
2735 010220 100401      BMI      2$          ;BRANCH IF MINUS
2736 010222 104001      1$:      ERROR      +1      ;CPU ERROR
2737                                ;CC SHOULD = 1000, R4 SHOULD=-1
2738 010224 005204      2$:      INC      R4          ;R4 SHOULD = 0
2739 010226 001375      BNE      1$          ;ERROR IF R4 NE 0
2740
2741
2742 010230      MDAO:
2743
2744      ;      TEST ADD MODE 0
2745 010230 005004      CLR      R4          ;R4=0
2746 010232 005001      CLR      R1
2747 010234 005101      COM      R1          ;R1=-1
2748 010236 060104      ADD      R1,R4      ;** TEST ADD OF R1 TO R4
2749 010240 001403      BEQ      1$          ;ERROR IF ZERO
2750 010242 103402      BCS      1$          ;ERROR IF CARRY
2751 010244 102401      BVS      1$          ;ERROR IF OVERFLOW
2752 010246 100401      BMI      2$          ;BRANCH IF MINUS
2753 010250 104001      1$:      ERROR      +1      ;CPU ERROR
2754                                ;CC SHOULD = 1000, R4= 1
2755 010252 005204      2$:      INC      R4          ;R4 SHOULD = 0
2756 010254 001401      BEQ      4$          ;BRANCH IF R4=0
2757 010256 104001      3$:      ERROR      +1      ;CPU ERROR
2758                                ;R4 SHOULD = 0
2759 010260              4$:

```



```

2760
2761
2762 010260      ; MDS0:
2763
2764      1      TEST SUB MODE 0
2765 010260 005004 CLR      R4
2766 010262 005001 CLR      R1
2767 010264 005201 INC      R1
2768 010266 160104 SUB      R1,R4
2769 010270 103403 BVS      1$
2770 010272 103002 BCC      1$
2771 010274 001401 BEQ      1$
2772 010276 100401 BMI      2$
2773 010300 104001 1$: ERROR  +1
2774
2775 010302 005101 2$: COM      R1
2776 010304 005201      INC      R1
2777 010306 160104      SUB      R1,R4
2778 010310 001401      BEQ      4$
2779 010312 104001 3$: ERROR  +1
2780
2781 010314      4$:
2782
2783      ; MDM27:
2784 010314
2785
2786      1      TEST MOV MODE 27,00
2787 010314 000252 CCC
2788 010316 012704 125252 MOV      0125252,R4
2789 010322 001401      BEQ      1$
2790 010324 100401      BMI      2$
2791 010326 104001 1$: ERROR  +1
2792
2793 010330 012701 052525 2$: MOV      0052525,R1
2794 010334 100401      BMI      3$
2795 010336 001001      BNE      4$
2796 010340 104001 3$: ERROR  +1
2797
2798 010342 060104 4$: ADD      R1,R4
2799 010344 100401      BMI      6$
2800 010346 104001 5$: ERROR  +1
2801
2802 010350 005204 6$: INC      R4
2803 010352 001375      BNE      5$
2804
2805      ; MB100:
2806 010354
2807
2808      1      TEST BIC, BIC MODE 0,0
2809 010354 005004 CLR      R4
2810 010356 005104 COM      R4
2811 010360 012701 125252 MOV      0125252,R1
2812 010364 012702 052525 MOV      0052525,R2
2813 010370 000261 SEC
2814 010372 040104 BIC      R1,R4
2815 010374 103003 BCC      1$

```

```

;R1=1 R4=0
; **TEST OF R4 R1, R4=-1
; ERROR IF V SET
; ERROR IF NO CARRY
; ERROR IF =0
; BRANCH IF MINUS
; CPU ERROR
; CC SHOULD = 1001
;R1=1
; GET TWO'S COMPLEMENT, R1=-1
; **TEST R4 R1 (1-1=0)
; BRANCH IF ZERO
; CPU ERROR
; CC SHOULD = 0100

; CC=0000
; **TEST MOVE
; ERROR IF = 0
; BRANCH IF MINUS
; CPU ERROR
; CC SHOULD = 1000
; **TEST MOVE
; ERROR IF MINUS
; BRANCH IF NE 0
; CPU ERROR
; CC SHOULD = 0000
;R1,R4= 1
; BRANCH IF MINUS
; CPU ERROR
; MOV FAILED
;R4=1=0
; ERROR IF NOT ZERO

;R4= 1
; SETUP R1 TEST DATA
; R2=COMPLEMENT OF R1
; **TEST BIC WITH CARRY SET
; ERROR IF NO CARRY

```

2816	010376	102402		BVS	1:		ERROR IF OVERFLOW
2817	010400	001401		BEQ	1:		ERROR IF 0
2818	010402	100001		BPL	2:		BRANCH IF PLUS
2819	010404	104001	1:	ERROR	1:		CPU ERROR
2820							CC SHOULD = 0001
2821	010406	020402	2:	CMP	R4,R2		COMPARE CONTENTS OF R4 AND R2
2822	010410	001401		BEQ	4:		BRANCH IF EQUAL
2823	010412	104001	3:	ERROR	1:		CPU ERROR
2824							R4 AND R2 SHOULD BE EQUAL
2825	010414	005301	4:	DEC	R1		R1=125251
2826	010416	050201		BIS	R2,R1		BIS 052525 AND 125251=177775
2827	010420	100401		BMI	6:		BRANCH IF PLUS VALUE
2828	010422	104001	5:	ERROR	1:		CPU ERROR
2829							BAD BIS OPERATION
2830	010424	005201	6:	INC	R1		
2831	010426	005201		INC	R1		R1=0
2832	010430	005201		INC	R1		
2833	010432	001373		BNE	5:		ERROR IF NE 0
2834							
2835							
2836	010434						
2837							
2838							
2839	010434	012701	125252				
2840	010440	012704	100000				
2841	010444	012702	052525				
2842	010450	030401					
2843	010452	001401					
2844	010454	100401					
2845	010456	104001					
2846							
2847	010460	020401					
2848	010462	001402					
2849	010464	103001					
2850	010466	100401					
2851	010470	104001					
2852							
2853	010472	020104					
2854	010474	001403					
2855	010476	103402					
2856	010500	102401					
2857	010502	100001					
2858	010504	104001					
2859							
2860	010506	005004					
2861	010510	005204					
2862	010512	000277					
2863	010514	030401					
2864	010516	001401					
2865	010520	104001					
2866							
2867	010522						
2868							
2869							
2870	010522						
2871							

CORDBO RDJ11 B CLUSTER MACY11 50(1046)
COKDAB,P11 05 APR 84 16:45

05 APR 84 16:45 PAGE 56
***** DOUBLE OPERAND TESTS *****

SEQ 0055

```

2872 ; TEST MOV, MOV8 MODE 1,1 AND SIGN EXT ON MOV8 TO GPR
2873 010522 012704 000400 MOV #400,R4 ;R4=400
2874 010526 012701 000402 MOV #402,R1 ;R1=402
2875 010532 005014 CLR (R4) ;
2876 010534 005114 COM (R4) ;400=-1
2877 010536 005011 CLR (R1) ;
2878 010540 105111 COMB (R1) ;402=000 377
2879 010542 005002 CLR R2 ;R2=0
2880 010544 012703 000405 MOV #405,R3 ;R3=405
2881 010550 000277 SCC ;CC=1111
2882 010552 011412 MOV (R4),(R2) ;MOV 400 TO 0 ,0=-1
2883 010554 001403 BEQ 1$ ;ERROR IF 0
2884 010556 102402 BVS 1$ ;ERROR IF OVERFLOW
2885 010560 103001 BCC 1$ ;ERROR IF NO CARRY
2886 010562 100401 BMI 2$ ;BRANCH IF GOOD
2887 010564 104001 1$: ERROR +1 ;CPU ERROR
2888 ;CC SHOULD =1001
2889 010566 005212 2$: INC (R2) ;0=0
2890 010570 001004 BNE 3$ ;ERROR IF NOT 0
2891 010572 000257 CCC ;CC=0000
2892 010574 111113 MOV8 (R1),(R3) ;405=377
2893 010576 001401 BEQ 3$ ;ERROR IF EQUAL
2894 010600 100401 BMI 4$ ;BRANCH IF GOOD
2895 010602 104001 3$: ERROR +1 ;CPU ERROR
2896 ;
2897 010604 105213 4$: INCB (R3) ;405=0
2898 010606 001375 BNE 5$ ;ERROR IF 405 NOT 0
2899 ;CHECK THAT SIGN EXTENSION OCCURS ON A MOV8 TO GENERAL REGISTER.
2900 010610 005002 CLR R2 ;INIT R2 TO ZERO.
2901 010612 111102 MOV8 (R1),R2 ;MOVE 377 TO R2
2902 010614 100005 RPL 5$ ;ERROR! BIT 15 SHOULD BE SET.
2903 010616 102404 BVS 5$ ;V BIT SHOULD BE CLEARED
2904 010620 103403 BCS 5$ ;CARRY BIT SHOULD BE UNAFFECTED
2905 010622 022702 177777 CMP #177777,R2 ;TEST R2
2906 010626 001401 BEQ 6$ ;SIGN EXTENDED THROUGH UPPER BYTE
2907 ;ERROR! BYTE SHOULD HAVE
2908 ;SIGN EXTENDED THROUGH UPPER BYTE
2909 010630 104001 5$: ERROR +1 ;CPU ERROR
2910 010632 6$:
2911 ;
2912 ;
2913 010632 ;
2914 ;
2915 ;
2916 010632 012704 000400 ; TEST ADD MODE 1,1
2917 010636 012701 000402 MOV #400,R4 ;R4=400
2918 010642 012714 177753 MOV #402,R1 ;R1=402
2919 010646 012711 000024 MOV #25,(R4) ;400=25
2920 010652 061114 MOV #24,(R1) ;402=24
2921 010654 001404 ADD (R1),(R4) ;25+24=1
2922 010656 103403 BEQ 1$ ;ERROR IF 0
2923 010660 100002 BCS 1$ ;ERROR IF CARRY
2924 010662 005214 BPL 1$ ;ERROR IF POSITIVE RESULT
2925 010664 001401 INC (R4) ;1+1=0
2926 010666 104001 1$: BEQ 2$ ;BRANCH IF GOOD
2927 ;CPU ERROR
;CC SHOULD = 1000

```

```

2928 010670      2$:
2929
2930
2931 010670      ;
2932      MBB11:
2933      ;
2934 010670 012704 000400      ; TEST SUB MODE 1,1
2935 010674 012701 000404      MOV      #400,R4      ;R4=400
2936 010700 012714 000003      MOV      #404,R1      ;R1=404
2937 010704 012711 000006      MOV      #3,(R4)      ;400=3
2938 010710 000277      MOV      #6,(R1)      ;406=6
2939 010712 161411      SCC      ;CC=1111
2940 010714 001402      SUB      (R4),(R1)      ;6-3=3
2941 010716 100401      BEQ      1$      ;ERROR IF 0
2942 010720 103001      BMI      1$      ;ERROR IF MINUS
2943 010722 104001      BCC      2$      ;BRANCH IF GOOD
2944      1$:      ERROR      +1      ;CPU ERROR
2945      ;CC SHOULD = 0000
2946 010724 161411      2$:      SUB      (R4),(R1)      ;3-3=0
2947 010726 001375      BNE      1$      ;ERROR IF NOT 0
2948
2949 010730      ;
2950      MBB11:
2951      ;
2952 010730 012704 000400      ; TEST BIC, BIS MODE 1,1
2953 010734 012701 000402      MOV      #400,R4      ;R4=400
2954 010740 012714 052525      MOV      #402,R1      ;R1=402
2955 010744 012711 125252      MOV      #052525,(R4)      ;400=052525
2956 010750 051411      MOV      #125252,(R1)      ;402=125252
2957 010752 001401      BIS      (R4),(R1)      ;R4 V R1 = -1
2958 010754 100401      BEQ      1$      ;ERROR IF 0
2959 010756 104001      BMI      2$      ;BRANCH IF GOOD
2960      1$:      ERROR      +1      ;CPU ERROR
2961      ;CC SHOULD = 1000
2962 010760 005211      2$:      INC      (R1)      ;402=0
2963 010762 001401      BEQ      4$      ;BRANCH IF GOOD
2964 010764 104001      3$:      ERROR      +1      ;CPU ERROR
2965      ;CC SHOULD = 0100
2966 010766 005311      4$:      DEC      (R1)      ;402=-1
2967 010770 041411      BIC      (R4),(R1)      ;R1=125252
2968 010772 001401      BEQ      5$      ;ERROR IF 0
2969 010774 100401      BMI      6$      ;BRANCH IF GOOD
2970 010776 104001      5$:      ERROR      +1      ;CPU ERROR
2971      ;CC SHOULD = 1000
2972 011000 005111      6$:      COM      (R1)      ;402=052525
2973 011002 041114      BIC      (R1),(R4)      ;400=0
2974 011004 001401      BEQ      8$      ;BRANCH IF GOOD
2975 011006 104001      7$:      ERROR      +1      ;CPU ERROR
2976      ;CC SHOULD = 0100
2977      8$:
2978      ;
2979 011010      ;
2980      MBB11:
2981      ;
2982 011010 012704 000400      ; TEST BIT, CMP MODE 1,1
2983 011014 012714 052525      MOV      #400,R4      ;R4=400
2984      MOV      #052525,(R4)      ;400=052525

```

CORDBAB RDJ11 B CLUSTER MACY11 30(1046) 05 APR-84 16:45 PAGE 58
COKDAB.P11 05-APR 84 16:45 ***** DOUBLE OPERAND TESTS *****

SEQ 0057

Address	Hex	Dec	Label	Instruction	Comments
2984	011020	012701	000402	MOV #402,R1	;R1=402
2985	011024	012711	125252	MOV #125252,(R1)	;402+125252
2986	011030	000241		CLC	;CLEAR CARRY
2987	011032	031411		BIT (R4),(R1)	;**052525+125252=0
2988	011034	103401		BCS 1\$	;ERROR IF CARRY
2989	011036	001401		BEQ 2\$	;BRANCH IF GOOD
2990	011040	104001	1\$:	ERROR +1	;CPU ERROR
2991					;CC SHOULD = 0100
2992	011042	021411	2\$:	CMP (R4),(R1)	;400-402=125253
2993	011044	001403		BEQ 3\$	;ERROR IF ZERO
2994	011046	103002		BCC 3\$	;ERROR IF NO CARRY
2995	011050	102001		BVC 3\$	;ERROR IF NO OVERFLOW
2996	011052	100401		BMI 4\$	;BRANCH IF GOOD
2997	011054	104001	3\$:	ERROR +1	;CPU ERROR
2998					;CC SHOULD = 1000
2999	011056	005014	4\$:	CLR (R4)	
3000	011060	005214		INC (R4)	;400+1
3001	011062	031114		BIT (R1),(R4)	;125252+1=0
3002	011064	001401		BEQ 6\$	;BRANCH IF GOOD
3003	011066	104001	5\$:	ERROR +1	;CPU ERROR
3004					;CC SHOULD= 0100
3005	011070		6\$:		
3006					
3007					
3008	011070		MM22:		
3009					
3010				TEST MOV MODE 2,2	
3011	011070	012704	000400	MOV #400,R4	;R4=400
3012	011074	012701	000402	MOV #402,R1	;R1=402
3013	011100	012714	000005	MOV #5,(R4)	;400+5
3014	011104	005021		CLR (R1)+	;402=0
3015	011106	005011		CLR (R1)	
3016	011110	005111		COM (R1)	;404=-1
3017	011112	005741		TST -(R1)	;R1=402
3018	011114	000277		SCC	;CC=1111
3019	011116	012124		MOV (R1)+,(R4)+	;400+0 R4=402 R1=404
3020	011120	100403		BMI 1\$	;ERROR IF MINUS
3021	011122	103002		BCC 1\$	;ERROR IF NO CARRY
3022	011124	102401		BVS 1\$	;ERROR IF OVERFLOW
3023	011126	001401		BEQ 2\$	;BRANCH IF GOOD
3024	011130	104001	1\$:	ERROR +1	;CPU ERROR
3025					;CC SHOULD= 0101
3026	011132	005244	2\$:	INC -(R4)	;400+1 R4=400
3027					
3028	011134	061411		ADD (R4),(R1)	;1+ -1 =0
3029	011136	001401		BEQ 4\$	;BRANCH IF GOOD
3030	011140	104001	3\$:	ERROR +1	;CPU ERROR
3031					;CC SHOULD = 0100
3032	011142		4\$:		
3033					
3034					
3035	011142		MM22:		
3036					
3037				TEST SUB MODE 2,2	
3038	011142	012704	000400	MOV #400,R4	;R4=400
3039	011146	012701	000402	MOV #402,R1	;R1=402

Address	Hex	Hex	Hex	Assembly	Comments
3040	011152	012714	177760	MOV	#177760,(R4) ;400=177760
3041	011156	012711	177750	MOV	#177750,(R1) ;402=177750
3042	011162	162421		SUB	(R4)+,(R1)+ ;R1=177770
3043	011164	001403		BEQ	1\$;ERROR IF ZERO
3044	011166	102402		BVS	1\$;ERROR IF OVERFLOW
3045	011170	103001		BCC	1\$;ERROR IF NO CARRY
3046	011172	100401		BMI	2\$;BRANCH IF GOOD
3047	011174	104001		1\$: ERROR	+1 ;CPU ERROR
3048					;CC SHOULD=1000
3049	011176	005241		2\$: INC	-(R1) ;R1=177771
3050	011200	162721	177771	SUB	#177771,(R1)+ ;R1=0
3051	011204	100401		BMI	3\$;ERROR IF MINUS
3052	011206	001401		BEQ	4\$;BRANCH IF GOOD
3053	011210	104001		3\$: ERROR	+1 ;CPU ERROR
3054					;CCSHOULD = 0100
3055	011212			4\$:	
3056					
3057					
3058	011212			;	MBB22;
3059					
3060				;	TEST BIC, BICB, BIS, BISB MODE 2,2
3061	011212	012704	000400	MOV	#400,R4 ;R4=400
3062	011216	012701	000402	MOV	#402,R1 ;R1=402
3063	011222	012702	000404	MOV	#404,R2 ;R2=404
3064	011226	012714	141401	MOV	#141401,(R4) ;400=303 001
3065	011232	012711	177405	MOV	#177405,(R1) ;402=377 005
3066	011236	012722	000070	MOV	#70,(R2)+ ;404=2070
3067	011242	012722	177777	MOV	#-1,(R2)+ ;406=-1
3068	011246	042421		BIC	(R4)+,(R1)+ ;402=074004
3069	011250	001401		BEQ	1\$;ERROR IF ZERO
3070	011252	100001		BPL	2\$;BRANCH IF GOOD
3071					;CC SHOULD = 1000
3072	011254	104001		1\$: ERROR	+1 ;CPU ERROR
3073	011256	052421		2\$: BIS	(R4)+,(R1)+ ;404=074074
3074	011260	142421		BICB	(R4)+,(R1)+ ;406=074
3075	011262	005301		DEC	R1 ;R4=405 R1=406
3076	011264	152421		BISB	(R4)+,(R1)+ ;406=-1 R4=406 R1=407
3077	011266	100401		BMI	4\$;BRANCH IF GOOD
3078					;406 SHOULD=-1
3079	011270	104001		3\$: ERROR	+1 ;CPU ERROR
3080	011272	005214		4\$: INC	(R4) ;406 SHOULD=0
3081	011274	001401		REQ	6\$;BRANCH IF GOOD
3082					;ERROR! 406 NE 0
3083	011276	104001		5\$: ERROR	+1 ;CPU ERROR
3084	011300			6\$:	
3085					
3086					
3087	011300			;	MBC22;
3088					
3089				;	TEST BIT, CMP MODE 2,2
3090	011300	012704	000400	MOV	#400,R4 ;R4=400
3091	011304	012701	000402	MOV	#402,R1 ;R1=402
3092	011310	012714	125252	MOV	#125252,(R4) ;400=125252
3093	011314	012721	100001	MOV	#100001,(R1)+ ;402=100001
3094	011320	012711	100002	MOV	#100002,(R1) ;404=100002
3095	011324	005741		ISI	(R1) ;R1=402

H15

CORRADO RDJ11 B CLUSTER MACY11 30(1046) 05 APR-84 16:45 PAGE 60
COKDAB.F11 05-APR-84 16:45 ***** DOUBLE OPERAND TESTS *****

SEQ 0059

```
3096 011326 132421      BITB      (R4)+,(R1)+      ;**ANDED RESULT = 000
3097 011330 100401      BMI       1$              ;ERROR IF MINUS
3098 011332 001401      BEQ       2$              ;BRANCH IF GOOD
3099 011334 104001      1$:      ERROR      +1              ;CPU ERROR
3100                      ;CC SHOULD = 0100
3101 011336 132124      2$:      BITB      (R1)+,(R4)+      ;** ANDED RESULT = 200
3102 011340 001401      BEQ       3$              ;ERROR IF EQUAL
3103 011342 100401      BMI       4$              ;BRANCH IF GOOD
3104 011344 104001      3$:      ERROR      +1              ;CPU ERROR
3105                      ;CC SHOULD = 1000 R4=402 R1=404
3106 011346 022421      4$:      CMP       (R4)+,(R1)+      ;RESULT = +1
3107 011350 001402      BEQ       5$              ;ERROR IF EQUAL
3108 011352 103001      BCC       5$              ;ERROR IF NO CARRY
3109 011354 100401      BMI       6$              ;BRANCH IF GOOD
3110 011356 104001      5$:      ERROR      +1              ;CPU ERROR
3111                      ;CC SHOULD = 0000
3112 011360 005341      6$:      DEC       -(R1)              ;404-100001
3113 011362 005741      TST       -(R1)              ;R4=404 R1=402
3114 011364 022124      CMP       (R1)+,(R4)+      ;RESULT = 0
3115 011366 001401      BEQ       8$              ;BRANCH IF GOOD
3116                      ;CC SHOULD = 0100 R1=404 R4=406
3117 011370 104001      7$:      ERROR      +1              ;CPU ERROR
3118 011372      8$:
3119
3120
3121 011372      ;MS33:
3122
3123      ;
3124 011372 005004      ;
3125 011374 012701 000002      CLR       R4              ;R4=0
3126 011400 012702 000400      MOV       #2,R1              ;R1=2
3127 011404 012714 000400      MOV       #400,R2             ;R2=400
3128 011410 012711 000402      MOV       #400,(R4)           ;0=400
3129 011414 012722 000200      MOV       #402,(R1)           ;2=402
3130 011420 012712 054320      MOV       #200,(R2)+          ;400+200
3131 011424 163431      MOV       #54320,(R2)              ;402+54320
3132 011426 001402      SUB       @R4)+,@R1)+          ;54320 200+54120
3133 011430 103401      BEQ       1$              ;ERROR IF ZERO
3134 011432 100001      BCS       1$              ;ERROR IF CARRY
3135 011434 104001      BPL       2$              ;BRANCH IF GOOD
3136      1$:      ERROR      +1              ;CPU ERROR
3137 011436 022712 054120      2$:      CMP       #54120,(R2)      ;CC SHOULD =0001
3138 011442 001401      BEQ       4$              ;TEST R4 AUTO INC AND RESULT
3139 011444 104001      3$:      ERROR      +1              ;BRANCH IF GOOD
3140                      ;CPU ERROR
3141 011446 005067 166326      4$:      CLR       0              ;CC SHOULD = 0100 R4=2 R1=4
3142 011452 005067 166324      CLR       1              ;RESTORE VECTORS
3143
3144
3145
3146 011456      ;MCB44:
3147
3148
3149 011456 012704 000400      ;
3150 011462 012701 000402      TEST CMP, BIT MODE 4,4
3151 011466 012721 125366      MOV       #400,R4              ;R4=400
3152                      MOV       #402,R1              ;R1=402
3153                      MOV       #125366,(R1)+          ;402-125366 R1=404
```

3152	011472	012724	173001	MOV	#173001,(R4)+	;400=173001 R4=402
3153	011476	024441		CMP	-(R4),-(R1)	;173001 - 125366=045603 CC=0000
3154	011500	103401		BCL	1\$	;ERROR IF CARRY
3155	011502	100001		BPL	2\$	;BRANCH IF GOOD
3156						;BAD COMPARE
3157	011504	104001		1\$: ERROR	+1	;CPU ERROR
3158	011506	005204		2\$: INC	R4	
3159	011510	005201		INC	R1	;R1=403
3160	011512	000261		SEC		;SET CARRY
3161	011514	134144		BITB	-(R1),-(R4)	;173+1=0
3162	011516	103001		BCC	3\$	;C SHOULD REMAIN SET
3163	011520	001401		BEQ	4\$	;BRANCH IF GOOD
3164						;INCORRECT COMPARE R4=400 R1=402
3165	011522	104001		3\$: ERROR	+1	;CPU ERROR
3166	011524	005724		4\$: TST	(R4)+	;R4=402
3167	011526	005201		INC	R1	;R1=403
3168	011530	124441		CMPB	-(R4),-(R1)	;173 - 173=0
3169	011532	001401		BEQ	6\$	;BRANCH IF GOOD
3170						;BAD COMPARE
3171	011534	104001		5\$: ERROR	+1	;CPU ERROR
3172	011536			6\$:		
3173						
3174	011536			MA55:		
3175						
3176					TEST ADD MODE 5,5	
3177	011536	012704	000400	MOV	#400,R4	
3178	011542	012724	000001	MOV	#1,(R4)+	;400=1
3179	011546	012724	177776	MOV	#2,(R4)+	;402=2
3180	011552	012724	000400	MOV	#400,(R4)+	;404=400
3181	011556	012714	000402	MOV	#402,(R4)	;406=402 R4=406
3182	011562	012701	000410	MOV	#410,R1	;R1=410
3183	011566	065451		ADD	#-(R4),#-(R1)	;1 - 2 = -1
3184	011570	001402		BEQ	1\$	;ERROR IF ZERO
3185	011572	100001		BPL	1\$	;ERROR IF PLUS
3186	011574	103001		BCC	2\$	;BRANCH IF GOOD
3187	011576	104001		1\$: ERROR	+1	;CPU ERROR
3188						;BAD ADD
3189	011600	062704	000004	2\$: ADD	#4,R4	;R4=410
3190	011604	065154		ADD	#(R1),#-(R4)	;1 - 1 = 0
3191	011606	001401		BEQ	4\$	;BRANCH IF GOOD
3192	011610	104001		3\$: ERROR	+1	;CPU ERROR
3193						;CC SHOULD = 0100 R4=406 R1=402
3194	011612			4\$:		
3195						
3196						
3197						
3198						
3199						
3200					TEST DOP BIT(B) MODE 6,6	
3201						
3202						
3203						
3204						
3205	011612			MB66:		
3206						
3207					TEST BIT, BITB MODE 6,6	


```

3208 011612 005004          CLR      R4              ;R4=0
3209 011614 012701 000400    MOV      @400,R1          ;
3210 011620 012721 125252    MOV      @125252,(R1)+     ;400=125252
3211 011624 012721 000001    MOV      @1,(R1)+         ;402=1
3212 011630 012721 100000    MOV      @100000,(R1)+      ;404=1000000 R1=406
3213 011634 036461 000400 177774 BIT      400(R4),-4(R1) ;(400)+(402)=0
3214 011642 001401          BEQ      2$              ;BRANCH IF GOOD
3215                      ;CC SHOULD = 0100
3216 011644 104001          1$:      ERROR      +1      ;CPU ERROR
3217 011646 136461 000405 177772 2$:      BITB     405(R4),-6(R1) ;(405)+(400)=200
3218 011654 001401          BEQ      3$              ;ERROR IF ZERO
3219 011656 100401          BMI      4$              ;BRANCH IF GOOD
3220                      ;CC SHOULD = 1000
3221 011660 104001          3$:      ERROR      +1      ;CPU ERROR
3222 011662          4$:
3223
3224                      ;
3225 011662          MS77:
3226
3227                      ;
3228 011662 012704 000400    ;      TEST SUB MODE 7,7
3229 011666 005001          MOV      @400,R4              ;
3230 011670 012724 177776    CLR      R1                ;
3231 011674 012724 177777    MOV      @-2,(R4)+          ;400=-2
3232 011700 012724 000400    MOV      @-1,(R4)+          ;402=-1
3233 011704 012711 000402    MOV      @400,(R4)+         ;404=400 R4=406
3234 011710 005201          MOV      @402,(R1)           ;0=402
3235 011712 167471 177372 000403 INC      R1            ;R1=1
3236 011720 001401          SUB      @ 406(R4),@403(R1)   ;-2 - -1 = -1
3237 011722 100401          BEQ      1$              ;ERROR IF ZERO
3238          BMI      2$              ;BRANCH IF GOOD
3239          ;CC SHOULD=1000
3240 011724 104001          1$:      ERROR      +1      ;CPU ERROR
3241 011726 167174 177777 177776 2$:      SUB      @-1(R1),@-2(R4) ; 1 - 1 = 0
3242 011734 001401          BEQ      4$              ;BRANCH IF GOOD
3243 011736 104001          3$:      ERROR      +1      ;CPU ERROR
3244          ;ERROR ON SUBTRACT 400=0
3245 011740 005067 166034 4$:      CLR      0              ;RESTORE VECTORS
3246 011744 005067 166032    CLR      2              ;
3247
3248                      ;
3249 011750          MRLO:
3250
3251                      ;
3252 011750 012704 125252    ;      TEST ROL, ROLB MODE 0
3253 011754 000277          MOV      @125252,R4          ;R4=125252
3254 011756 006104          ORC      R4                ;CC=1111
3255 011760 102004          ROL      R4                ;R4=052525 WITH CARRY SET
3256 011762 103003          BVC     1$              ;ERROR IF V CLEAR
3257 011764 022704 052525    RRC      R4                ;ERROR IF CARRY CLEAR
3258 011770 001401          CMP      @052525,R4          ;SEE IF RD = EXPECTED
3259 011772 104001          BEQ      2$              ;ERROR IF R4 NE EXPECTED
3260          1$:      ERROR      +1      ;CPU ERROR
3261 011774 012704 125252    2$:      MOV      @125252,R4 ;R4=125252
3262 012000 000257          ORC      R4                ;CC=0000
3263 012002 106104          ROLB     R4                ;ROTATE EVEN BYTE

```

3264	012004	103005		BCC	3\$		;ERROR IF NO CARRY
3265	012006	102004		BVC	3\$		;ERROR IF NO OVERFLOW
3266	012010	100403		BMI	3\$		;ERROR IF MINUS
3267	012012	022704	125124	CMP	0125124,R4		;SEE IF R4 > EXPECTED
3268	012016	001401		BEQ	4\$		;BRANCH IF GOOD
3269	012020	104001		3\$:	ERROR	+1	;CPU ERROR
3270							;ROLB FAILED, CC SHOULD=1011, R4=125125
3271	012022			4\$:			
3272							
3273							
3274	012022						
3275							
3276							
3277	012022	005004					
3278	012024	012714	052525				
3279	012030	006114					
3280	012032	100005					
3281	012034	102004					
3282	012036	103403					
3283	012040	021427	125252				
3284	012044	001401					
3285							
3286	012046	104001		1\$:	ERROR	+1	;CPU ERROR
3287	012050	012714	125252	2\$:	MOV	0125252,(R4)	;O=125252
3288	012054	005204			INC	R4	;R4=1
3289	012056	000277			SCC		;CC=1111
3290	012060	106114			ROLB	(R4)	;**TEST INSTRUCTION
3291	012062	100406			BMI	3\$	;ERROR IF RESULT IS POSITIVE
3292	012064	103005			BCC	3\$	;ERROR IF NO CARRY
3293	012066	102004			BVC	3\$	;ERROR IF V CLEAR
3294	012070	005304			DEC	R4	;R4=0
3295	012072	022714	052652		CMP	052652,(R4)	;ERROR IF 0 NE EXPECTED
3296	012076	001401			BEQ	4\$	;BRANCH IF GOOD
3297	012100	104001		3\$:	ERROR	+1	;CPU ERROR
3298							;BAD ROLB ODD BYTE,CC SHOULD=1011
3299	012102			4\$:			
3300							
3301							
3302	012102						
3303							
3304							
3305	012102	005004					
3306	012104	012714	100000				
3307	012110	000257					
3308	012112	006124					
3309	012114	103002					
3310	012116	102001					
3311	012120	001401					
3312							
3313	012122	104001		1\$:	ERROR	+1	;CPU ERROR
3314	012124	005304		2\$:	DEC	R4	
3315	012126	005304			DEC	R4	
3316	012130	001012			BNE	3\$	;ERROR IN AUTO DEC
3317	012132	012714	004040		MOV	04040,(R4)	;O=4040
3318	012136	000241			CLC		
3319	012140	106124			ROLB	(R4)	;**TEST INSTRUCTION

3320	012142	103405		BCS	3\$		;ERROR IF CARRY SET
3321	012144	102404		BVS	3\$		;ERROR IF V
3322	012146	005304		DEC	R4		
3323	012150	022714	004100	CMP	#04100,(R4)		;SEE IF 0- EXPECTED RESULT
3324	012154	001401		BEQ	4\$		;BRANCH IF GOOD
3325	012156	104001		3\$: ERROR	+1		;CPU ERROR
3326							;BAD ROL
3327	012160			4\$:			
3328							
3329							
3330	012160			MRL3:			
3331							
3332							
3333	012160	005004					
3334	012162	012714	052525	CLR	R4		;R4=0
3335	012166	000277		MOV	#052525,(R4)		;0=52525
3336	012170	006137	000000	SCC			;CC=1111
3337	012174	100005		ROL	#0		;**TEST INSTRUCTION MODE 3 WITH PC
3338	012176	102004		BPL	1\$		;ERROR IF PLUS
3339	012200	103403		BVC	1\$		;ERROR IF NO OVERFLOW
3340	012202	022714	125253	BCS	1\$		;ERROR IF CARRY
3341	012206	001401		CMP	#125253,(R4)		;COMPARE RESULT WITH EXPECTED
3342				BEQ	2\$		;BRANCH IF GOOD
3343	012210	104001					;BAD ROL. CC SHOULD=1010
3344	012212	012714	125252	1\$: ERROR	+1		;CPU ERROR
3345	012216	000261		2\$: MOV	#125252,(R4)		;0=125252
3346	012220	106137	000000	SEC			;CC=---1
3347	012224	100402		ROLB	#0		;**TEST INSTRUCTION
3348	012226	103001		BMI	3\$		;ERROR IF MINUS
3349	012230	102401		BCC	3\$		;ERROR IF NO CARRY
3350	012232	104001		HVS	4\$		;BRANCH IF OVERFLOW
3351				3\$: ERROR	+1		;CPU ERROR
3352	012234						;BAD ROL. CC SHOULD=1011
3353				4\$:			
3354							
3355	012234			MRL4:			
3356							
3357							
3358	012234	005001					
3359	012236	012704	000002	TEST ROL MODE 4			
3360	012242	012711	054321	CLR	R1		;R1=0
3361	012246	000277		MOV	#2,R4		;R4=2
3362	012250	006144		MOV	#54321,(R1)		;0=54321
3363	012252	100007		SCC			;CC=1111
3364	012254	102006		ROL	(R4)		;**TEST INSTRUCTION
3365	012256	103405		BPL	1\$		;ERROR IF PLUS
3366	012260	022711	130643	BVC	1\$		;ERROR IF NO OVERFLOW
3367	012264	001002		BCS	1\$		;ERROR IF CARRY
3368	012266	005704		CMP	#130643,(R1)		;SEE IF EXPECTED RESULT
3369	012270	001401		BNE	1\$		;BRANCH IF ROL FAILED
3370				TSI	R4		;SEE IF AUTO-DEC WORKED
3371	012272	104001		BEQ	2\$		;BRANCH IF GOOD
3372	012274						;ERROR! BAD ROL INST
3373				1\$: ERROR	+1		;CPU ERROR
3374				2\$:			
3375	012274						
				MRL5:			

COKDABO KDJ11 B CLUSTER MACY11 30(1046) 05 APR-84 16:45 PAGE 65
 COKDAB.P11 05-APR-84 16:45 ***** DOUBLE OPERAND TESTS *****

SEQ 0064

```

3376
3377 ; TEST ROL MODE 5
3378 012274 005004 ; CLR R4 ; R4=0
3379 012276 012714 000400 ; MOV #400,(R4) ; 0=400
3380 012302 012734 123456 ; MOV #123456,@(R4)+ ; 400=123465, R4=2
3381 012306 000277 ; SCC ; CC=1111
3382 012310 006154 ; ROL @-(R4) ; **TEST INSTRUCTION
3383 012312 100410 ; BMI 1$ ; ERROR IF RESULT IS MINUS
3384 012314 103007 ; BCC 1$ ; ERROR IF NO CARRY
3385 012316 102006 ; BVC 1$ ; ERROR IF NO OVERFLOW
3386 012320 005704 ; TST R4 ; SEE IF AUTO-DEC WORKED
3387 012322 001004 ; BNE 1$ ; ERROR OF AUTO-DEC
3388 012324 022737 047135 000400 ; CMP #47135,@#400 ; SEE IF CORRECT RESULT
3389 012332 001401 ; BEQ 2$ ; BRANCH IF GOOD
3390 ; BAD ROL MODE 5
3391 012334 104001 1$: ERROR +1 ; CPU ERROR
3392 012336 2$:
3393
3394 ;
3395 012336 ; MRL6:
3396
3397 ; TEST ROL MODE 6
3398 012336 012704 000400 ; MOV #400,R4 ; R4=400
3399 012342 005001 ; CLR R1 ; R1=0
3400 012344 012711 032525 ; MOV #32525,(R1) ; 0=32525
3401 012350 000277 ; SCC
3402 012352 006164 177400 ; ROL -400(R4) ; **TEST INSTRUCTION
3403 012356 100405 ; BMI 1$ ; ERROR IF MINUS
3404 012360 103404 ; BCS 1$ ; ERROR IF CARRY
3405 012362 102403 ; BVS 1$ ; ERROR IF OVERFLOW
3406 012364 022711 065253 ; CMP #65253,(R1) ; SEE IF CORRECT RESULT
3407 012370 001401 ; BEQ 2$ ; BRANCH IF GOOD
3408 ; BAD ROL MODE 6
3409 012372 104001 1$: ERROR +1 ; CPU ERROR
3410 012374 2$:
3411
3412 ;
3413 012374 ; MRL7:
3414
3415 ; TEST ROL MODE 7
3416 012374 012704 000400 ; MOV #400,R4 ; R4=400
3417 012400 005037 000402 ; CLR @#402 ; 402=0
3418 012404 012737 100000 000000 ; MOV #100000,@#0 ; 0=100000
3419 012412 006174 000002 ; ROL @2(R4) ; **TEST INSTRUCTION
3420 012416 100406 ; BMI 1$ ; ERROR IF MINUS
3421 012420 001005 ; BNE 1$ ; ERROR IF NOT ZERO
3422 012422 103004 ; BCC 1$ ; ERROR IF NO CARRY
3423 012424 102003 ; BVC 1$ ; ERROR IF NO OVERFLOW
3424 012426 005737 000000 ; TST @#0 ; CHECK RESULT
3425 012432 001401 ; BEQ 2$ ; BRANCH IF GOOD
3426 ; BAD ROL MODE 7
3427 012434 104001 1$: ERROR +1 ; CPU ERROR
3428 012436 2$:
3429
3430 ;
3431 012436 ; MSW37:

```

N5

CORRADO KDJ11-B CLUSTER MACY11 30(1046) 05 APR 84 16:45 PAGE 66
 CORRAD.P11 05-APR-84 16:45 ***** DOUBLE OPERAND TESTS *****

SEQ 0065

```

3432
3433      ;      TEST SWAB MODE 37
3434 012436 012704 000400      MOV      #400,R4      ;R4=400
3435 012442 012714 040700      MOV      #40700,(R4)    ;400= 101 300
3436 012446 000337 000400      SWAB      @#400        ;400 SHOULD = 300 101
3437 012452 100406      BMI      1$      ;ERROR IF MINUS
3438 012454 022714 140101      CMP      #140101,(R4)    ;SEE IF EXPECTED RESULT
3439 012460 001003      BNE      1$      ;BRANCH IF BAD
3440 012462 000337 000400      SWAB      @#400        ;400=101 300
3441 012466 100401      BMI      2$      ;BRANCH IF GOOD
3442      ;ERROR! BAD SWAB MODE 37
3443 012470 104001      1$:      ERROR      +1      ;CPU ERROR
3444 012472      2$:
3445
3446      ;
3447 012472      MRR0:
3448
3449      ;      TEST ROR MODE 0
3450 012472 012704 052525      MOV      #52525,R4      ;R4=52525
3451 012476 000257      CCC      ;CC=0000
3452 012500 006004      ROR      R4      ;R4 SHOULD = 25252 WITH CARRY
3453 012502 103003      BCC      1$      ;ERROR IF NO CARRY
3454 012504 022704 025252      CMP      #25252,R4      ;SEE IF R4= EXPECTED
3455 012510 001401      BEQ      2$      ;BRANCH IF GOOD
3456      ;ROR MODE 0 FAILED
3457 012512 104001      1$:      ERROR      +1      ;CPU ERROR
3458 012514      2$:
3459
3460      ;
3461 012514      MRRB1:
3462
3463      ;      TEST RORB MODE 1
3464 012514 005004      CLR      R4      ;R4=0
3465 012516 012714 000001      MOV      #1,(R4)      ;0=1
3466 012522 000277      SCC      ;CC 1111
3467 012524 106014      RORB      (R4)      ;0=000200, NO C
3468 012526 103004      BCC      1$      ;ERROR IF NO CARRY
3469 012530 100003      RPL      1$      ;ERROR IF PLUS
3470 012532 022714 000200      CMP      #200,(R4)      ;CHECK RESULT
3471 012536 001401      BEQ      2$      ;BRANCH IF GOOD
3472 012540 104001      1$:      ERROR      +1      ;CPU ERROR
3473      ;BAD RESULT
3474 012542      2$:
3475
3476      ;
3477
3478 012542      MJ:
3479
3480      ;      TEST JMP ALL MODES
3481 012542 012737 000001 003032      MOV      #1,@#SEQ      ;SETUP TEST SEQUENCER
3482 012550 012701 012614      MOV      @MJ01,R1      ;SET MODE 1 JUMP ADDRESS
3483 012554 000111      JMP      (R1)      ;JMP MODE 1
3484 012556 023727 003032 000002      MJ02:      CMP      @#SEQ,#2      ;CHECK FOR CORRECT SEQUENCE
3485 012564 001401      BEQ      MJ02A      ;BRANCH IF GOOD
3486      ;MODE 2 JUMP FAILED
3487 012566 104001      1$:      ERROR      +1      ;CPU ERROR

```

CORDBO RDJ11 B CLUSTER MACY11 30(1046) 05 APR 84 16:45 PAGE 67
 CORDAB.P11 05 APR 84 16:45 ***** DOUBLE OPERAND TESTS *****

SEQ 0056

3488	012570	020127	012560		MJU2A:	CMP	R1,0MJU2+2		!CHECK FOR AUTO-INCREMENT
3489	012574	001401				BEQ	MJU2B		!BRANCH IF GOOD
3490	012576	104001			24:	ERROR	+1		!CPU ERROR
3491									!AUTO-INCR FAILED
3492	012600	005237	003032		MJU2B:	INC	00SEQ		!UPDATE TEST SEQUENCER
3493	012604	012701	012612			MOV	0MJU2,R1		!SETUP MODE 3 JUMP
3494	012610	000131				JMP	0(R1)+		!JUMP MODE 3
3495	012612	012640			MJU2:	WORD	MJU3		!MODE 3 DESTINATION
3496	012614	023727	003032	000001	MJU1:	CMP	00SEQ,01		!TEST FOR CORRECT SEQUENCE
3497	012622	001401				BEQ	MJU1A		!BRANCH IF GOOD
3498	012624	104001			34:	ERROR	+1		!CPU ERROR
3499									!JUMP OUT OF SEQUENCE
3500	012626	005237	003032		MJU1A:	INC	00SEQ		!UPDATE SEQUENCE
3501	012632	012701	012556			MOV	0MJU2,R1		!SETUP MODE 2 DESTINATION
3502	012636	000121				JMP	(R1)+		!JUMP MODE 2
3503	012640	023727	003032	000003	MJU3:	CMP	00SEQ,03		!TEST FOR CORRECT SEQUENCE
3504	012646	001401				BEQ	MJU3A		!BRANCH IF GOOD
3505	012650	104001			44:	ERROR	+1		!CPU ERROR
3506									!JUMP OUT OF SEQUENCE
3507	012652	022701	012614		MJU3A:	CMP	0MJU2+2,R1		!TEST AUTO-INCREMENT
3508	012656	001401				BEQ	MJU3B		!BRANCH IF GOOD
3509	012660	104001			54:	ERROR	+1		!CPU ERROR
3510									!AUTO-INCREMENT FAILED MODE 3
3511	012662	005237	003032		MJU3B:	INC	00SEQ		!UPDATE SEQUENCER
3512	012666	012701	012736			MOV	0MJU4+2,R1		!SETUP DESTINATION MODE 4
3513	012672	000141				JMP	-(R1)		!EXECUTE JUMP MODE 4
3514	012674	000000			MJU5:	HALT			
3515	012676	022701	012772			CMP	0MJ5,R1		!CHECK AUTO DECREMENT
3516	012702	001401				BEQ	MJU5A		!BRANCH IF GOOD AUTO DEC
3517	012704	104001			64:	ERROR	+1		!CPU ERROR
3518									!AUTO-DEC FAILED MODE 5
3519	012706	023727	003032	000005	MJU5A:	CMP	00SEQ,05		!TEST CORRECT SEQUENCE
3520	012714	001401				BEQ	MJU5B		!BRANCH IF GOOD SEQUENCE
3521	012716	104001			74:	ERROR	+1		!CPU ERROR
3522									!JUMP OUT OF SEQUENCE
3523	012720	005237	003032		MJU5B:	INC	00SEQ		!UPDATE SEQUENCE COUNT
3524	012724	012701	012767			MOV	0MJU6+5,R1		!SETUP DESTINATION MODE 6
3525	012730	000161	000005			JMP	+5(R1)		!JUMP MODE 6
3526					84:				
3527	012734	000240			MJU4:	NOP			
3528	012736	022701	012734			CMP	0MJU4,R1		!TEST AUTO DEC
3529	012742	001401				BEQ	MJU4A		!BRANCH IF GOOD
3530	012744	104001			84:	ERROR	+1		!CPU ERROR
3531									!MODE 4 AUTO-DEC FAILED
3532	012746	023727	003032	000004	MJU4A:	CMP	00SEQ,04		!TEST FOR CORRECT SEQUENCE
3533	012754	001401				BEQ	MJU4B		!BRANCH IF CORRECT SEQUENCE
3534	012756	104001			94:	ERROR	+1		!CPU ERROR
3535									!INCORRECT JUMP SEQUENCE
3536	012760	005237	003032		MJU4B:	INC	00SEQ		!UPDATE SEQUENCE
3537	012764	012701	012774			MOV	0MJ5+2,R1		!SETUP MODE 5 POINTER
3538	012770	000151				JMP	2(R1)		!EXECUTE MODE 5 JUMP
3539					104:				
3540	012772	012676			MJU5:	WORD	MJU5+2		!POINTER MODE 5
3541					114:				
3542	012774	022737	000006	003032	MJU6:	CMP	00,00SEQ		!CHECK FOR CORRECT SEQUENCE
3543	013002	001401				BEQ	MJU6A		!BRANCH IF GOOD


```
3544 013004 104001 10$: ERROR +1 ;CPU ERROR
3545 ;INCORRECT SEQUENCE
3546 013006 005237 003032 MJU6A: INC @0SEQ ;UPDATE SEQUENCER
3547 013012 012701 013032 MOV @MJ7+10,R1 ;SETUP INDEX
3548 013016 000171 177770 JMP @-10(R1) ;EXECUTE MODE 7 JUMP
3549 013022 013026 MJ7: .WORD MJU7 ;POINTER FOR MODE 7
3550 013024 000000 HALT
3551 013026 022737 000007 003032 MJU7: CMP #7,@0SEQ ;TEST FOR CORRECT SEQUENCE
3552 013034 001401 BEQ MJU7E ;BRANCH IF GOOD SEQUENCE
3553 013036 104001 11$: ERROR +1 ;CPU ERROR
3554 ;TESTING OUT OF SEQUENCE
3555 013040 MJU7E:
3556 ;
3557 ;
3558 ;TEST THAT PRE-FETCH BUFFER CAN BE OVER WRITTEN
3559 013040 012701 013350 MOV #128,R1 ;SET UP R1 WITH ADDRESS OF ERROR
3560 ;ROUTINE
3561 013044 012717 MOV (PC)+,(PC) ;WRITE THE NOP OVER THE JMP INSTRUCTION
3562 013046 000240 .WORD NOP ;NOP INSTRUCTION
3563 013050 000111 64$: .WORD 111 ;JMP (R1)
3564 013052 012717 MOV (PC)+,(PC) ;WRITE THE NOP OVER THE JMP INSTRUCTION
3565 013054 000240 .WORD NOP ;NOP INSTRUCTION
3566 013056 000111 65$: .WORD 111 ;JMP (R1)
3567 013060 012717 MOV (PC)+,(PC) ;WRITE THE NOP OVER THE JMP INSTRUCTION
3568 013062 000240 .WORD NOP ;NOP INSTRUCTION
3569 013064 000111 66$: .WORD 111 ;JMP (R1)
3570 013066 012717 MOV (PC)+,(PC) ;WRITE THE NOP OVER THE JMP INSTRUCTION
3571 013070 000240 .WORD NOP ;NOP INSTRUCTION
3572 013072 000111 67$: .WORD 111 ;JMP (R1)
3573 013074 012717 MOV (PC)+,(PC) ;WRITE THE NOP OVER THE JMP INSTRUCTION
3574 013076 000240 .WORD NOP ;NOP INSTRUCTION
3575 013100 000111 68$: .WORD 111 ;JMP (R1)
3576 013102 012717 MOV (PC)+,(PC) ;WRITE THE NOP OVER THE JMP INSTRUCTION
3577 013104 000240 .WORD NOP ;NOP INSTRUCTION
3578 013106 000111 69$: .WORD 111 ;JMP (R1)
3579 013110 012717 MOV (PC)+,(PC) ;WRITE THE NOP OVER THE JMP INSTRUCTION
3580 013112 000240 .WORD NOP ;NOP INSTRUCTION
3581 013114 000111 70$: .WORD 111 ;JMP (R1)
3582 013116 012717 MOV (PC)+,(PC) ;WRITE THE NOP OVER THE JMP INSTRUCTION
3583 013120 000240 .WORD NOP ;NOP INSTRUCTION
3584 013122 000111 71$: .WORD 111 ;JMP (R1)
3585 013124 012717 MOV (PC)+,(PC) ;WRITE THE NOP OVER THE JMP INSTRUCTION
3586 013126 000240 .WORD NOP ;NOP INSTRUCTION
3587 013130 000111 72$: .WORD 111 ;JMP (R1)
3588 013132 012717 MOV (PC)+,(PC) ;WRITE THE NOP OVER THE JMP INSTRUCTION
3589 013134 000240 .WORD NOP ;NOP INSTRUCTION
3590 013136 000111 73$: .WORD 111 ;JMP (R1)
3591 013140 012717 MOV (PC)+,(PC) ;WRITE THE NOP OVER THE JMP INSTRUCTION
3592 013142 000240 .WORD NOP ;NOP INSTRUCTION
3593 013144 000111 74$: .WORD 111 ;JMP (R1)
3594 013146 012717 MOV (PC)+,(PC) ;WRITE THE NOP OVER THE JMP INSTRUCTION
3595 013150 000240 .WORD NOP ;NOP INSTRUCTION
3596 013152 000111 75$: .WORD 111 ;JMP (R1)
3597 013154 012717 MOV (PC)+,(PC) ;WRITE THE NOP OVER THE JMP INSTRUCTION
3598 013156 000240 .WORD NOP ;NOP INSTRUCTION
3599 013160 000111 76$: .WORD 111 ;JMP (R1)
```

1065

CORPABO RDJ11 B CLUSTER MACY11 30(1046)
COKDAB.P11 05-APR 84 16:45

05 APR 84 16:45 PAGE 69
***** DOUBLE OPERAND TESTS *****

SEQ 0068

3600	013162	012717	MOV	(PC)+,(PC)	;WRITE THE NOP OVER THE JMP INSTRUCTION
3601	013164	000240	.WORD	NOP	;NOP INSTRUCTION
3602	013166	000111	77\$: .WORD	111	;JMP (R1)
3603	013170	012717	MOV	(PC)+,(PC)	;WRITE THE NOP OVER THE JMP INSTRUCTION
3604	013172	000240	.WORD	NOP	;NOP INSTRUCTION
3605	013174	000111	78\$: .WORD	111	;JMP (R1)
3606	013176	012717	MOV	(PC)+,(PC)	;WRITE THE NOP OVER THE JMP INSTRUCTION
3607	013200	000240	.WORD	NOP	;NOP INSTRUCTION
3608	013202	000111	79\$: .WORD	111	;JMP (R1)
3609	013204	012717	MOV	(PC)+,(PC)	;WRITE THE NOP OVER THE JMP INSTRUCTION
3610	013206	000240	.WORD	NOP	;NOP INSTRUCTION
3611	013210	000111	80\$: .WORD	111	;JMP (R1)
3612	013212	012717	MOV	(PC)+,(PC)	;WRITE THE NOP OVER THE JMP INSTRUCTION
3613	013214	000240	.WORD	NOP	;NOP INSTRUCTION
3614	013216	000111	81\$: .WORD	111	;JMP (R1)
3615	013220	012717	MOV	(PC)+,(PC)	;WRITE THE NOP OVER THE JMP INSTRUCTION
3616	013222	000240	.WORD	NOP	;NOP INSTRUCTION
3617	013224	000111	82\$: .WORD	111	;JMP (R1)
3618	013226	012717	MOV	(PC)+,(PC)	;WRITE THE NOP OVER THE JMP INSTRUCTION
3619	013230	000240	.WORD	NOP	;NOP INSTRUCTION
3620	013232	000111	83\$: .WORD	111	;JMP (R1)
3621	013234	012717	MOV	(PC)+,(PC)	;WRITE THE NOP OVER THE JMP INSTRUCTION
3622	013236	000240	.WORD	NOP	;NOP INSTRUCTION
3623	013240	000111	84\$: .WORD	111	;JMP (R1)
3624	013242	012717	MOV	(PC)+,(PC)	;WRITE THE NOP OVER THE JMP INSTRUCTION
3625	013244	000240	.WORD	NOP	;NOP INSTRUCTION
3626	013246	000111	85\$: .WORD	111	;JMP (R1)
3627	013250	012717	MOV	(PC)+,(PC)	;WRITE THE NOP OVER THE JMP INSTRUCTION
3628	013252	000240	.WORD	NOP	;NOP INSTRUCTION
3629	013254	000111	86\$: .WORD	111	;JMP (R1)
3630	013256	012717	MOV	(PC)+,(PC)	;WRITE THE NOP OVER THE JMP INSTRUCTION
3631	013260	000240	.WORD	NOP	;NOP INSTRUCTION
3632	013262	000111	87\$: .WORD	111	;JMP (R1)
3633	013264	012717	MOV	(PC)+,(PC)	;WRITE THE NOP OVER THE JMP INSTRUCTION
3634	013266	000240	.WORD	NOP	;NOP INSTRUCTION
3635	013270	000111	88\$: .WORD	111	;JMP (R1)
3636	013272	012717	MOV	(PC)+,(PC)	;WRITE THE NOP OVER THE JMP INSTRUCTION
3637	013274	000240	.WORD	NOP	;NOP INSTRUCTION
3638	013276	000111	89\$: .WORD	111	;JMP (R1)
3639	013300	012717	MOV	(PC)+,(PC)	;WRITE THE NOP OVER THE JMP INSTRUCTION
3640	013302	000240	.WORD	NOP	;NOP INSTRUCTION
3641	013304	000111	90\$: .WORD	111	;JMP (R1)
3642	013306	012717	MOV	(PC)+,(PC)	;WRITE THE NOP OVER THE JMP INSTRUCTION
3643	013310	000240	.WORD	NOP	;NOP INSTRUCTION
3644	013312	000111	91\$: .WORD	111	;JMP (R1)
3645	013314	012717	MOV	(PC)+,(PC)	;WRITE THE NOP OVER THE JMP INSTRUCTION
3646	013316	000240	.WORD	NOP	;NOP INSTRUCTION
3647	013320	000111	92\$: .WORD	111	;JMP (R1)
3648	013322	012717	MOV	(PC)+,(PC)	;WRITE THE NOP OVER THE JMP INSTRUCTION
3649	013324	000240	.WORD	NOP	;NOP INSTRUCTION
3650	013326	000111	93\$: .WORD	111	;JMP (R1)
3651	013330	012717	MOV	(PC)+,(PC)	;WRITE THE NOP OVER THE JMP INSTRUCTION
3652	013332	000240	.WORD	NOP	;NOP INSTRUCTION
3653	013334	000111	94\$: .WORD	111	;JMP (R1)
3654	013336	012717	MOV	(PC)+,(PC)	;WRITE THE NOP OVER THE JMP INSTRUCTION
3655	013340	000240	.WORD	NOP	;NOP INSTRUCTION

3656	013342	000111		95\$:	.WORD	111		;JMP (R1)
3657	013344	000137	013352		JMP	@#129\$		;JUMP OVER ERROR CALL
3658	013350			128\$:				;ERROR! PRE-FETCH BUFFER WAS NOT
3659								;OVER WRITTEN
3660	013350	104001			ERROR	+1		;CPU ERROR
3661								
3662								
3663								NOW RESTORE THE OVER WRITTEN JMP INSTRUCTIONS FOR THE NEXT PASS.
3664	013352	012702	000040	129\$:	MOV	#32,R2		;SET UP R2 AS COUNTER
3665	013356	012703	013050		MOV	#64\$,R3		;SET UP R3 AS POINTER
3666	013362	012713	000111	130\$:	MOV	#111,(R3)		;RESTORE OVER WRITTEN JUMPS
3667	013366	062703	000006		ADD	#6,R3		;POINT TO NEXT OVER WRITTEN ADDR.
3668	013372	077205			SOB	R2,130\$		;DO UNTIL R2=0
3669								
3670								
3671	013374				MJP:			
3672								
3673								TEST JMP MODES 17,27,37,67,77
3674	013374	012737	000000	003032	MOV	#0,@#SEQ		;SETUP TEST SEQUENCER
3675	013402	000117			JMP	(R7)		;JUMP MODE 17(SHOULD BE IN-LINE)
3676								
3677	013404	005737	003032		MJP17:	TST	@#SEQ	;CHECK SEQUENCE
3678	013410	001401			BEQ	2\$		;BRANCH IF GOOD
3679								;ERROR! BAD JUMP
3680	013412	104001		1\$:	ERROR	+1		;CPU ERROR
3681	013414	005237	003032	2\$:	INC	@#SEQ		;UPDATE SEQUENCE NUMBER
3682	013420	000127	000401		JMP	#401		;JUMP MODE 27
3683								; (THE #401-BR UPDATED PC+2)
3684								
3685	013424	000000			HALT			
3686								
3687	013426	023727	003032	000001	MJP27:	CMP	@#SEQ,#1	;CHECK IF CORRECT SEQUENCE
3688	013434	001401			BEQ	MJP27A		;BRANCH IF IN SEQUENCE
3689	013436	104001		1\$:	ERROR	+1		;CPU ERROR
3690								;TEST OUT OF SEQUENCE
3691	013440	005237	003032		MJP27A:	INC	@#SEQ	;UPDATE SEQUENCER
3692	013444	000137	013512		JMP	@#MJP37		;JUMP MODE 37
3693	013450	023727	003032	000003	MJP67:	CMP	@#SEQ,#3	;CHECK FOR CORRECT SEQUENCE
3694	013456	001401			BEQ	MJP67A		;BRANCH IF IN SEQUENCE
3695	013460	104001		2\$:	ERROR	+1		;CPU ERROR
3696								;TEST OUT OF SEQUENCE
3697	013462	005237	003032		MJP67A:	INC	@#SEQ	;UPDATE SEQUENCER
3698	013466	000257			CCC			;INSURE ZBIT=0
3699	013470	000177	000002		JMP	@#MJP67B		;JUMP MODE 7
3700								
3701	013474	000000			HALT			
3702								
3703	013476	013500			MJP67B:	.WORD	MJP77	
3704								
3705	013500	023727	003032	000004	MJP77:	CMP	@#SEQ,#4	;TEST FOR CORRECT SEQUENCE
3706	013506	001412			BEQ	MJP77E		;BRANCH IF IN SEQUENCE
3707	013510	104001		3\$:	ERROR	+1		;CPU ERROR
3708								;TEST OUT OF SEQUENCE
3709	013512	023727	003032	000002	MJP37:	CMP	@#SEQ,#2	;TEST FOR CORRECT SEQUENCE
3710	013520	001401			BEQ	MJP37A		;BRANCH IF IN SEQUENCE
3711	013522	104001		4\$:	ERROR	+1		;CPU ERROR

```

3712
3713 013524 005237 003032 MJP37A: INC @0SEQ ; TEST OUT OF SEQUENCE
3714 013530 000167 177714 JMP MJP67 ; UPDATE SEQUENCER
3715 ; ; JUMP MODE 6
3716 ;
3717 013534 MJP77E:
3718 ;
3719 ;
3720 013534 MJSR:
3721 ;
3722 ; TEST JSR ALL MODES
3723 013534 010637 003034 MOV R6,@0SP5 ; SAVE STACK POINTER LOCATION
3724 013540 010637 003036 MOV R6,@0SP5J ;
3725 013544 162737 000002 003036 SUB @2,@0SP5J ; SPSJ = R6 AFTER DECRIMENT
3726 013552 012737 000001 003032 MOV @1,@0SEQ ; SETUP SEQUENCE COUNTER
3727 013560 012701 013654 MOV @MJSR1,R1 ; SETUP INITIAL JUMP IN MODE 1
3728 013564 005004 CLR R4 ;
3729 013566 005104 COM R4 ; R4 = -1 TO BE SAVED ON STACK
3730 013570 004411 JSR R4,(R1) ; JSR MODE 1
3731 ;
3732 013572 022737 000002 003032 MJSR2: CMP @2,@0SEQ ; TEST FOR CORRECT SEQUENCE
3733 013600 001401 BEQ MJSR2A ; BRANCH IF GOOD
3734 013602 104001 5$: ERROR +1 ; CPU ERROR
3735 ; MODE 2 JUMPED TO OUT OF SEQUENCE
3736 013604 023706 003036 MJSR2A: CMP @0SP5J,R6 ; VERIFY STACK DECRIMENT
3737 013610 001006 BNE 6$ ; BRANCH IF STACK INCORRECT
3738 013612 021627 125252 CMP (R6),@125252 ; VERIFY CONTENTS OF STACK
3739 013616 001003 BNE 6$ ; BRANCH IF DATA ON STACK INCORRECT
3740 013620 022704 013734 CMP @MJSR4,R4 ; SEE IF CORRECT RETURN ADDRESS
3741 013624 001401 BEQ MJSR2B ; BRANCH IF GOOD
3742 013626 104001 6$: ERROR +1 ; CPU ERROR
3743 ; JSR MODE 2 FAILED
3744 013630 005237 003032 MJSR2B: INC @0SEQ ; UPDATE SEQUENCE COUNTER
3745 013634 013706 003034 MOV @0SP5,R6 ; RELOAD STACK POINTER
3746 013640 012701 013652 MOV @MJSR4,R1 ; SETUP JSR MODE 3
3747 013644 005004 CLR R4 ; DIFFERENT DATA TO R4
3748 013646 004431 JSR R4,@(R1)+ ; JSR MODE 3
3749 013650 000000 HALT
3750 013652 014020 MJSR4: .WORD MJSR3 ; LITERAL FOR JUMP MODE 3
3751 ;
3752 013654 022737 000001 003032 MJSR1: CMP @1,@0SEQ ; TEST FOR CORRECT SEQUENCE
3753 013662 001401 BEQ MJSR1A ; BRANCH IF GOOD
3754 013664 104001 7$: ERROR +1 ; CPU ERROR
3755 ; MODE 1 JUMPED TO OUT OF SEQUENCE
3756 013666 023706 003036 MJSR1A: CMP @0SP5J,R6 ; VERIFY STACK DECRIMENT
3757 013672 001006 BNE 8$ ; BRANCH IF STACK INCORRECT
3758 013674 021627 177777 CMP (R6),@-1 ; VERIFY CONTENT OF STACK
3759 013700 001003 BNE 8$ ; BRANCH IF DATA ON STACK INCORRECT
3760 013702 022704 013572 CMP @MJSR2,R4 ; SEE IF CORRECT RETURN ADDRESS
3761 013706 001401 BEQ MJSR1B ; BRANCH IF GOOD
3762 013710 104001 8$: ERROR +1 ; CPU ERROR
3763 ; JSR MODE 2 FAILED
3764 013712 005237 003032 MJSR1B: INC @0SEQ ; UPDATE SEQUENCE COUNTER
3765 013716 013706 003034 MOV @0SP5,R6 ; RELOAD STACK POINTER
3766 013722 012704 125252 MOV @125252,R4 ; SETUP R4 DATA
3767 013726 012701 013572 MOV @MJSR2,R1 ; SETUP MODE 2 JUMP ADDRESS

```

```

3768 013732 004421          JSR      R4,(R1)+          ;*JUMP MODE 2
3769
3770
3771 013734 022737 000004 003032 MJSR4:  CMP      #4,#0SEQ          ; TEST FOR CORRECT SEQUENCE
3772 013742 001401          BEQ      MJSR4A          ;BRANCH IF GOOD
3773 013744 104001          9$:      ERROR      +1          ;CPU ERROR
3774
3775 013746 023706 003036          MJSR4A: CMP      @0SPSJ,R6          ;MODE 4 JUMPED TO OUT OF SEQUENCE
3776 013752 001006          BNE      10$          ;VERIFY STACK DECIMENT
3777 013754 021627 052525          CMP      (R6),#052525          ;BRANCH IF STACK INCORRECT
3778 013760 001003          BNE      10$          ;VERIFY CONTENTS OF STAACK
3779 013762 022704 014102          CMP      @MJSR6,R4          ;BRANCH IF DATA ON STACK INCORRECT
3780 013766 001401          BEQ      MJSR4B          ;SEE IF CORRECT RETURN ADDRESS
3781 013770 104001          10$:     ERROR      +1          ;BRANCH IF GOOD
3782
3783 013772 005237 003032          MJSR4B: INC      @0SEQ          ;CPU ERROR
3784 013776 013706 003034          MOV      @0SPS,R6          ;JSR MODE 4 FAILED
3785 014002 012704 000377          MOV      #377,R4          ;UPDATE SEQUENCE COUNTER
3786 014006 012701 014020          MOV      @MJSRB+2,R1        ;RELOAD STACK POINTER
3787 014012 004451          JSR      R4,@-(R1)          ;SETUP R4 DATA
3788 014014 000000          HALT
3789 014016 014166          MJSRB:  .WORD    MJSR5          ;SETUP JSR VECTOR
3790
3791
3792 014020 022737 000003 003032 MJSR3:  CMP      #3,#0SEQ          ; TEST FOR CORRECT SEQUENCE
3793 014026 001401          BEQ      MJSR3A          ;BRANCH IF GOOD
3794 014030 104001          11$:     ERROR      +1          ;CPU ERROR
3795
3796 014032 023706 003036          MJSR3A: CMP      @0SPSJ,R6          ;MODE 3 JUMPED TO OUT OF SEQUENCE
3797 014036 001006          BNE      12$          ;VERIFY STACK DECIMENT
3798 014040 021627 000000          CMP      (R6),#0          ;BRANCH IF STACK INCORRECT
3799 014044 001003          BNE      12$          ;VERIFY CONTENTS OF STAACK
3800 014046 022704 013650          CMP      @MJSRA-2,R4        ;BRANCH IF DATA ON STACK INCORRECT
3801 014052 001401          BEQ      MJSR3B          ;SEE IF CORRECT RETURN ADDRESS
3802 014054 104001          12$:     ERROR      +1          ;BRANCH IF GOOD
3803
3804 014056 005237 003032          MJSR3B: INC      @0SEQ          ;CPU ERROR
3805 014062 013706 003034          MOV      @0SPS,R6          ;JSR MODE 3 FAILED
3806 014066 012704 052525          MOV      #052525,R4        ;UPDATE SEQUENCE COUNTER
3807 014072 012701 013736          MOV      @MJSR4+2,R1        ;RELOAD STACK POINTER
3808 014076 000257          CCC
3809 014100 004441          JSR      R4,@-(R1)          ;SETUP R4 DATA
3810
3811
3812 014102 022737 000006 003032 MJSR6:  CMP      #6,#0SEQ          ; TEST FOR CORRECT SEQUENCE
3813 014110 001401          BEQ      MJSR6A          ;BRANCH IF GOOD
3814 014112 104001          13$:     ERROR      +1          ;CPU ERROR
3815
3816 014114 023706 003036          MJSR6A: CMP      @0SPSJ,R6          ;MODE 6 JUMPED TO OUT OF SEQUENCE
3817 014120 001006          BNE      14$          ;VERIFY STACK DECIMENT
3818 014122 021627 123456          CMP      (R6),#123456        ;BRANCH IF STACK INCORRECT
3819 014126 001003          BNE      14$          ;VERIFY CONTENTS OF STACK
3820 014130 022704 014250          CMP      @MJSR7,R4          ;BRANCH IF DATA ON STACK INCORRECT
3821 014134 001401          BEQ      MJSR6B          ;SEE IF CORRECT RETURN ADDRESS
3822 014136 104001          14$:     ERROR      +1          ;BRANCH IF GOOD
3823

```

;CPU ERROR
;JSR MODE 6 FAILED

H6

COKDABO KDJ11-B CLUSTER MACY11 30(1046)
COKDAB,P11 05-APR-84 16:4505-APR 84 16:45 PAGE 73
***** DOUBLE OPERAND TESTS *****

SEQ 0072

```

3824 014140 005237 003032 MJSR6B: INC @#SEQ ;UPDATE SEQUENCE COUNTER
3825 014144 013706 003034 MOV @#SPS,R6 ;RELOAD STACK POINTER
3826 014150 012704 177773 MOV #-5,R4 ;SETUP R4 DATA
3827 014154 012701 014174 MOV @#MJSRC+10,R1 ;SETUP JSR VECTOR
3828 014160 004471 177770 JSR R4,@-10(R1) ;JSR MODE 7
3829 014164 014250 MJSRC: .WORD MJSR7 ;JSR VECTOR
3830
3831 ;
3832 014166 022737 000005 003032 MJSR5: CMP #5,@#SEQ ;TEST FOR CORRECT SEQUENCE
3833 014174 001401 BEQ MJSR5A ;BRANCH IF GOOD
3834 014176 104001 15$: ERROR +1 ;CPU ERROR
3835 ;MODE 5 JUMPED TO OUT OF SEQUENCE
3836 014200 023706 003036 MJSR5A: CMP @#SPSJ,R6 ;VERIFY STACK DECIMENT
3837 014204 001006 BNE 16$ ;BRANCH IF STACK INCORRECT
3838 014206 021627 000377 CMP (R6),#377 ;VERIFY CONTENTS OF STACK
3839 014212 001003 BNE 16$ ;BRANCH IF DATA ON STACK INCORRECT
3840 014214 022704 014014 CMP @#MJSRB-2,R4 ;SEE IF CORRECT RETURN ADDRESS
3841 014220 001401 BEQ MJSR5B ;BRANCH IF GOOD
3842 014222 104001 16$: ERROR +1 ;CPU ERROR
3843 ;JSR MODE 5 FAILED
3844 014224 005237 003032 MJSR5B: INC @#SEQ ;UPDATE SEQUENCE COUNTER
3845 014230 013706 003034 MOV @#SPS,R6 ;RELOAD STACK POINTER
3846 014234 012704 123456 MOV #123456,R4 ;SETUP DATA IN R4
3847 014240 012701 014112 MOV @#MJSR6+10,R1 ;SETUP JSR VECTOR
3848 014244 004461 177770 JSR R4,-10(R1) ;JUMP MODE 6
3849
3850 ;
3851 014250 022737 000007 003032 MJSR7: CMP #7,@#SEQ ;TEST FOR CORRECT SEQUENCE
3852 014256 001401 BEQ MJSR7A ;BRANCH IF GOOD
3853 014260 104001 17$: ERROR +1 ;CPU ERROR
3854 ;MODE 7 JUMPED TO OUT OF SEQUENCE
3855 014262 023706 003036 MJSR7A: CMP @#SPSJ,R6 ;VERIFY STACK DECIMENT
3856 014266 001006 BNE 18$ ;BRANCH IF STACK INCORRECT
3857 014270 021627 177773 CMP (R6),#5 ;VERIFY CONTENTS OF STACK
3858 014274 001003 BNE 18$ ;BRANCH IF DATA ON STACK INCORRECT
3859 014276 022704 014164 CMP @#MJSR5-2,R4 ;SEE IF CORRECT RETURN ADDRESS
3860 014302 001401 BEQ MJSR7E ;BRANCH IF GOOD
3861 014304 104001 18$: ERROR +1 ;CPU ERROR
3862 ;JSR MODE 7 FAILED
3863 014306 MJSR7E: MOV @#SPS,R6 ;REPLACE STACK
3864 014306 013706 003034
3865
3866 ;
3867 014312 MJRA:
3868
3869 ; TEST JSR MODES 27, 37, 67, 77
3870 014312 012737 000001 003032 MOV #1,@#SEQ ;SETUP SEQUENCER
3871 014320 010637 003034 MOV R6,@#SPS ;SAVE STACK ADDRESS
3872 014324 010637 003036 MOV R6,@#SPSJ ;SAVE STACK DECIMENT ADDRESS
3873 014330 162737 000002 003036 SUB #2,@#SPSJ ;
3874 014336 012704 177777 MOV #-1,R4 ;SETUP R4 DATA
3875 014342 004427 000240 JSR R4,#240 ;EXECUTE A JSR MODE 27
3876
3877 014346 022737 000001 003032 MJR27: CMP #1,@#SEQ ;VERIFY CORRECT TEST SEQUENCE
3878 014354 001011 BNE 1$ ;INCORRECT TEST SEQUENCE
3879 014356 023706 003036 CMP @#SPSJ,R6 ;VERIFY STACK POINTER

```

3880	014362	001006			BNE	1\$		
3881	014364	021627	177777		CMP	(R6),#-1		;VERIFY R4 GOT LOADED ON THE STACK
3882	014370	001003			BNE	1\$		;BRANCH IF INCORRECT STACK CONTENTS
3883	014372	020427	014346		CMP	R4,#MJR27		;VERIFY CORRECT RETURN ADDRESS
3884	014376	001401			BEQ	MJR27A		;BRANCH IF GOOD RETURN ADDRESS ON STACK
3885	014400	104001		1\$:	ERROR	+1		;CPU ERROR
3886								;MODE 27 FAILED
3887	014402	005237	003032	MJR27A:	INC	@0SEQ		;UPDATE SEQUENCER
3888	014406	012704	152525		MOV	@152525,R4		;SETUP R4 TEST DATA
3889	014412	013706	003034		MOV	@0SPS,R6		;RESET STACK POINTER
3890	014416	004437	014504		JSR	R4,@0MJR37		;JSR MODE 37
3891	014422	000000		MJR27B:	HALT			
3892								
3893	014424	023727	003032	000003	MJR67:	CMP	@0SEQ,#3	;VERIFY TEST SEQUENCE
3894	014432	001011			BNE	2\$		;INCORRECT TEST SEQUENCE
3895	014434	023706	003036		CMP	@0SPS,R6		;VERIFY STACK DECREMENT
3896	014440	001006			BNE	2\$		;INCORRECT STACK DECREMENT
3897	014442	021627	000125		CMP	(R6),#125		;VERIFY STACK WAS LOADED
3898	014446	001003			BNE	2\$		
3899	014450	020427	014560		CMP	R4,#MJR77		;VERIFY RETURN ADDRESS
3900	014454	001401			BEQ	MJR67A		;BRANCH IF GOOD
3901	014456	104001		2\$:	ERROR	+1		;CPU ERROR
3902								;MODE 67 FAILED
3903	014460	005237	003032	MJR67A:	INC	@0SEQ		;UPDATE SEQUENCER
3904	014464	013706	003034		MOV	@0SPS,R6		;RESET STACK
3905	014470	012704	000001		MOV	#1,R4		;SETUP R4 DATA
3906	014474	004477	000002		JSR	R4,@MJR6B		;JSR MODE 77
3907	014500	000000		MJR6A:	HALT			
3908	014502	014560		MJR6B:	.WORD	MJR77		;DATA FOR MODE 77 JUMP
3909								
3910	014504	023727	003032	000002	MJR37:	CMP	@0SEQ,#2	;VERIFY TEST SEQUENCE
3911	014512	001011			BNE	2\$		;INCORRECT TEST SEQUENCE
3912	014514	023706	003036		CMP	@0SPS,R6		;VERIFY STACK DECREMENT
3913	014520	001006			BNE	2\$		;INCORRECT STACK DECREMENT
3914	014522	021627	152525		CMP	(R6),#152525		;VERIFY STACK WAS LOADED
3915	014526	001003			BNE	2\$		
3916	014530	020427	014422		CMP	R4,#MJR27B		;VERIFY RETURN ADDRESS
3917	014534	001401			BEQ	MJR37A		;BRANCH IF GOOD
3918	014536	104001		2\$:	ERROR	+1		;CPU ERROR
3919								;MODE 37 FAILED
3920	014540	005237	003032	MJR37A:	INC	@0SEQ		;UPDATE SEQUENCER
3921	014544	013706	003034		MOV	@0SPS,R6		;RELOAD STACK
3922	014550	012704	000125		MOV	#125,R4		;SETUP R4 TEST DATA
3923	014554	004467	177644		JSR	R4,MJR67		;JSR MODE 6
3924								
3925	014560	023727	003032	000004	MJR77:	CMP	@0SEQ,#4	;VERIFY TEST SEQUENCE
3926	014566	001011			BNE	2\$		;INCORRECT TEST SEQUENCE
3927	014570	023706	003036		CMP	@0SPS,R6		;VERIFY STACK DECREMENT
3928	014574	001006			BNE	2\$		;INCORRECT STACK DECREMENT
3929	014576	021627	000001		CMP	(R6),#1		;VERIFY STACK WAS LOADED
3930	014602	001003			BNE	2\$		
3931	014604	020427	014500		CMP	R4,#MJR6A		;VERIFY RETURN ADDRESS
3932	014610	001401			BEQ	MJR77A		;BRANCH IF GOOD
3933	014612	104001		2\$:	ERROR	+1		;CPU ERROR
3934								;MODE 77 FAILED
3935	014614			MJR77A:				

```

3936
3937 014614 013706 003034      ;      MOV      @SPS,R6          ;RESET STACK
3938
3939
3940 014620      ;MRTS:
3941
3942      ;      TEST RTS AND RTS R6
3943 014620 012706 001000      MOV      @STBOT,R6          ;INSURE VALID STACK
3944 014624 012746 123456      MOV      @123456,-(R6)      ;SETUP TEST REGISTER
3945 014630 012703 014640      MOV      @RTS1,R3          ;SETUP TEST PC
3946 014634 000203          RTS      R3          ;**TEST INSTRUCTION
3947 014636 104001          ERROR    +1          ;CPU ERROR
3948          ;INCORRECT PC ON RTS
3949 014640 022703 123456      RTS1:  CMP      @123456,R3
3950 014644 001401          BEQ      RTS6          ;BRANCH IF GOOD
3951 014646 104001          ERROR    +1          ;CPU ERROR
3952          ;REGISTER CONTENTS INCORRECT
3953
3954      ;THIS TEST CHECKS AN UN-TESTED PLA TERM
3955
3956 014650 010601          RTS6:  MOV      R6,R1          ;SAVE STACK IN R1
3957 014652 012705 014664      MOV      @1$,R5          ;MOVE EXPECTED RETURN ADDR TO R5
3958 014656 010506          MOV      R5,R6          ;MOVE RETURN ADDR TO R6
3959 014660 000206          RTS      R6
3960 014662 104001          ERROR    +1          ;CPU ERROR
3961          ;ERROR! RTS NOT EXECUTED
3962 014664 021506      1$:      CMP      (R5),R6          ;IS R6=31506?
3963 014666 001401          BEQ      RTSE          ;IF IT IS THEN GO TO END OF TEST
3964 014670 104001          ERROR    +1          ;CPU ERROR
3965          ;ERROR! WRONG ADDR IN R6
3966 014672 010106      RTSE:   MOV      R1,R6          ;RESTORE STACK
3967
3968 014674      TSMMUO:
3969      ;      SETUP AND TEST KERNEL, SUPERVISOR AND USER STACKS
3970 014674 012737 040000 177776      MOV      @40000,@177776      ;SET PS TO SUP MODE
3971 014702 012706 177777      MOV      @177777,R6          ;INIT SUP STACK TO ALL ONES
3972 014706 022706 177777      CMP      @177777,R6          ;ARE ALL BITS SET
3973 014712 001401          BEQ      1$          ;YES GO ON
3974          ;NO GO TO ERROR
3975 014714 104001          ERROR    +1          ;CPU ERROR
3976 014716 005006      1$:      CLR      R6          ;SET SUP STACK TO ALL ZEROES
3977 014720 022706 000000      CMP      @0,R6          ;ARE ALL BITS CLEARED
3978 014724 001401          BEQ      2$          ;YES GO ON
3979          ;NO GO TO ERROR
3980 014726 104001          ERROR    +1          ;CPU ERROR
3981 014730 012706 125252      2$:      MOV      @125252,R6          ;SET SUP STACK TO ALTERNATING PATTERN
3982 014734 022706 125252      CMP      @125252,R6          ;IS SUP SP CORRECT
3983 014740 001401          BEQ      3$          ;YES GO ON
3984          ;NO GO TO ERROR
3985 014742 104001          ERROR    +1          ;CPU ERROR
3986 014744 012706 052525      3$:      MOV      @52525,R6          ;SET SUP STACK TO ALTERNATING PATTERN
3987 014750 022706 052525      CMP      @52525,R6          ;IS SUP SP CORRECT
3988 014754 001401          BEQ      4$          ;YES GO ON
3989          ;NO GO TO ERROR
3990 014756 104001          ERROR    +1          ;CPU ERROR
3991 014760 012706 000700      4$:      MOV      @700,R6          ;SETUP SUP SP

```


CORDBO KDJ11 B CLUSTER MACY11 30(1046) 05 APR 84 16:45 PAGE 76
 CORDAB.P11 05-APR-84 16:45 ***** DOUBLE OPERAND TESTS *****

SEQ 0075

3992	014764	012737	140000	177776		MOV	#140000,#177776	;SET PS TO USER MODE
3993	014772	012706	177777			MOV	#177777,R6	;INIT USER STACK TO ALL ONES
3994	014776	022706	177777			CMP	#177777,R6	;ARE ALL BITS SET
3995	015002	001401				BEQ	5\$	;YES GO ON
3996								;NO GO TO ERROR
3997	015004	104001				ERROR	+1	;CPU ERROR
3998	015006	005006			5\$:	CLR	R6	;SET USER STACK TO ALL ZEROES
3999	015010	022706	000000			CMP	#0,R6	;ARE ALL BITS CLEARED
4000	015014	001401				BEQ	6\$	;YES GO ON
4001								;NO GO TO ERROR
4002	015016	104001				ERROR	+1	;CPU ERROR
4003	015020	012706	125252		6\$:	MOV	#125252,R6	;SET USER STACK TO ALTERNATING PATTERN
4004	015024	022706	125252			CMP	#125252,R6	;IS USER SP CORRECT
4005	015030	001401				BEQ	7\$	;YES GO ON
4006								;NO GO TO ERROR
4007	015032	104001				ERROR	+1	;CPU ERROR
4008	015034	012706	052525		7\$:	MOV	#52525,R6	;SET USER STACK TO ALTERNATING PATTERN
4009	015040	022706	052525			CMP	#52525,R6	;IS USER SP CORRECT
4010	015044	001401				BEQ	8\$	;YES GO ON
4011								;NO GO TO ERROR
4012	015046	104001				ERROR	+1	;CPU ERROR
4013	015050	012706	000600		8\$:	MOV	#600,R6	;SETUP USER SP
4014	015054	005037	177776			CLR	#177776	;SET PS TO KER MODE
4015	015060	012706	001000			MOV	#STBOT,R6	;SETUP KERNEL SP
4016	015064	005037	000700			CLR	#700	;CLEAR FIRST WORDS OF SUP, KER, AND USE STACKS
4017	015070	005037	000600			CLR	#600	
4018	015074	005037	001000			CLR	#STBOT	
4019	015100	004767	000054			JSR	PC,CHECK	; TEST KER, SUP, AND USE STACKS
4020	015104	012737	040000	177776	RET1:	MOV	#40000,#177776	;SET PSW TO SUP MODE
4021	015112	022706	000700			CMP	#700,R6	;IS SUP SP CORRECT
4022	015116	001401				BEQ	1\$	;YES GO ON
4023								;NO GO TO ERROR
4024	015120	104001				ERROR	+1	;CPU ERROR
4025	015122	012737	140000	177776	1\$:	MOV	#140000,#177776	;SET PSW TO USE MODE
4026	015130	022706	000600			CMP	#600,R6	;IS USE SP CORRECT
4027	015134	001401				BEQ	2\$	;YES GO ON
4028								;NO GO TO ERROR
4029	015136	104001				ERROR	+1	;CPU ERROR
4030	015140	005037	177776		2\$:	CLR	#177776	;SET PSW TO KER MODE
4031	015144	022706	001000			CMP	#STBOT,R6	;IS KER SP CORRECT
4032	015150	001401				BEQ	3\$	;YES GO ON
4033								;NO GO TO ERROR
4034	015152	104001				ERROR	+1	;CPU ERROR
4035	015154				3\$:			
4036	015154	000167	000206			JMP	M150	
4037								
4038								
4039								
4040	015160	012737	040000	177776		CHECK:	MOV	#40000,#177776
4041	015166	004767	000044			JSR	PC,CHECK1	; TEST SUP, KER, AND USE STACKS
4042	015172	022716	000000		RET2:	CMP	#0,(SP)	;IS SUP STACK CLEARED
4043	015176	001401				BEQ	1\$	;YES GO ON
4044								;NO GO TO ERROR
4045	015200	104001				ERROR	+1	;CPU ERROR
4046	015202	012737	140000	177776	1\$:	MOV	#140000,#177776	;SET PSW TO USE MODE
4047	015210	022716	000000			CMP	#0,(SP)	;IS USE STACK CLEARED

```

4048 015214 001401          BEQ      2$          ;YES GO ON
4049                                ;NO GO TO ERROR
4050 015216 104001          ERROR    +1          ;CPU ERROR
4051 015220 005037 177776  2$:      CLR      @#177776      ;SET PSW TO KER MODE
4052 015224 022716 015104      CMP      @RET1,(SP)      ;DOES KER STACK HAVE CORRECT DATA
4053 015230 001401          BEQ      3$          ;YES GO ON
4054                                ;NO GO TO ERROR
4055 015232 104001          ERROR    +1          ;CPU ERROR
4056 015234 000207 3$:      RTS      PC          ;RETURN
4057                                ;
4058                                ;ROUTINE TO CHECK STACKS AFTER ONE RTS
4059                                ;
4060 015236 012737 140000 177776 CHECK1: MOV      @140000,@#177776      ;SET PSW TO USE MODE
4061 015244 004767 000044      JSR      PC,CHECK2      ;TEST KER, SUP, AND USE STACKS
4062 015250 022716 000000 RET3:   CMP      @0,(SP)      ;IS USE STACK CLEARED
4063 015254 001401          BEQ      1$          ;YES GO ON
4064                                ;NO GO TO ERROR
4065 015256 104001          ERROR    +1          ;CPU ERROR
4066 015260 005037 177776  1$:      CLR      @#177776      ;SET PSW TO KER MODE
4067 015264 022716 015104      CMP      @RET1,(SP)      ;IS KER STACK CORRECT
4068 015270 001401          BEQ      2$          ;YES GO ON
4069                                ;NO GO TO ERROR
4070 015272 104001          ERROR    +1          ;CPU ERROR
4071 015274 012737 040000 177776 2$:      MOV      @40000,@#177776      ;SET PSW TO SUP MODE
4072 015302 022716 015172      CMP      @RET2,(SP)      ;IS SUP STACK CORRECT
4073 015306 001401          BEQ      3$          ;YES GO ON
4074                                ;NO GO TO ERROR
4075 015310 104001          ERROR    +1          ;CPU ERROR
4076 015312 000207 3$:      RTS      PC          ;RETURN
4077                                ;
4078                                ;ROUTINE TO CHECK STACKS AFTER ZERO RTS
4079                                ;
4080 015314 022716 015250 CHECK2: CMP      @RET3,(SP)      ;IS USE STACK CORRECT
4081 015320 001401          BEQ      1$          ;YES GO ON
4082                                ;NO GO TO ERROR
4083 015322 104001          ERROR    +1          ;CPU ERROR
4084 015324 012737 040000 177776 1$:      MOV      @40000,@#177776      ;SET PSW TO SUP MODE
4085 015332 022716 015172      CMP      @RET2,(SP)      ;IS SUP STACK CORRECT
4086 015336 001401          BEQ      2$          ;YES GO ON
4087                                ;NO GO TO ERROR
4088 015340 104001          ERROR    +1          ;CPU ERROR
4089 015342 005037 177776  2$:      CLR      @#177776      ;SET PSW TO KER MODE
4090 015346 022716 015104      CMP      @RET1,(SP)      ;IS KER STACK CORRECT
4091 015352 001401          BEQ      3$          ;YES GO ON
4092                                ;NO GO TO ERROR
4093 015354 104001          ERROR    +1          ;CPU ERROR
4094 015356 012737 140000 177776 3$:      MOV      @140000,@#177776      ;SET PSW TO USE MODE
4095 015364 000207          RTS      PC          ;RETURN
4096                                ;
4097 015366          MISO:
4098 015366          MMVCC:
4099                                ;
4100                                ;TEST MOV CONDITION CODES ***
4101 015366 000277          SCC
4102 015370 000244          CLR
4103 015372 012704 000000      MOV      @0,R4          ;CC=0101, R4=0

```


Address	Hex	Dec	Label	Op	OpC	OpD	OpE	OpF	OpG	OpH	OpI	OpJ	OpK	OpL	OpM	OpN	OpO	OpP	OpQ	OpR	OpS	OpT	OpU	OpV	OpW	OpX	OpY	OpZ	OpAA	OpAB	OpAC	OpAD	OpAE	OpAF	OpAG	OpAH	OpAI	OpAJ	OpAK	OpAL	OpAM	OpAN	OpAO	OpAP	OpAQ	OpAR	OpAS	OpAT	OpAU	OpAV	OpAW	OpAX	OpAY	OpAZ	OpBA	OpBB	OpBC	OpBD	OpBE	OpBF	OpBG	OpBH	OpBI	OpBJ	OpBK	OpBL	OpBM	OpBN	OpBO	OpBP	OpBQ	OpBR	OpBS	OpBT	OpBU	OpBV	OpBW	OpBX	OpBY	OpBZ	OpCA	OpCB	OpCC	OpCD	OpCE	OpCF	OpCG	OpCH	OpCI	OpCJ	OpCK	OpCL	OpCM	OpCN	OpCO	OpCP	OpCQ	OpCR	OpCS	OpCT	OpCU	OpCV	OpCW	OpCX	OpCY	OpCZ	OpDA	OpDB	OpDC	OpDD	OpDE	OpDF	OpDG	OpDH	OpDI	OpDJ	OpDK	OpDL	OpDM	OpDN	OpDO	OpDP	OpDQ	OpDR	OpDS	OpDT	OpDU	OpDV	OpDW	OpDX	OpDY	OpDZ	OpEA	OpEB	OpEC	OpED	OpEE	OpEF	OpEG	OpEH	OpEI	OpEJ	OpEK	OpEL	OpEM	OpEN	OpEO	OpEP	OpEQ	OpER	OpES	OpET	OpEU	OpEV	OpEW	OpEX	OpEY	OpEZ	OpFA	OpFB	OpFC	OpFD	OpFE	OpFF	OpFG	OpFH	OpFI	OpFJ	OpFK	OpFL	OpFM	OpFN	OpFO	OpFP	OpFQ	OpFR	OpFS	OpFT	OpFU	OpFV	OpFW	OpFX	OpFY	OpFZ	OpGA	OpGB	OpGC	OpGD	OpGE	OpGF	OpGG	OpGH	OpGI	OpGJ	OpGK	OpGL	OpGM	OpGN	OpGO	OpGP	OpGQ	OpGR	OpGS	OpGT	OpGU	OpGV	OpGW	OpGX	OpGY	OpGZ	OpHA	OpHB	OpHC	OpHD	OpHE	OpHF	OpHG	OpHH	OpHI	OpHJ	OpHK	OpHL	OpHM	OpHN	OpHO	OpHP	OpHQ	OpHR	OpHS	OpHT	OpHU	OpHV	OpHW	OpHX	OpHY	OpHZ	OpIA	OpIB	OpIC	OpID	OpIE	OpIF	OpIG	OpIH	OpII	OpIJ	OpIK	OpIL	OpIM	OpIN	OpIO	OpIP	OpIQ	OpIR	OpIS	OpIT	OpIU	OpIV	OpIW	OpIX	OpIY	OpIZ	OpJA	OpJB	OpJC	OpJD	OpJE	OpJF	OpJG	OpJH	OpJI	OpJJ	OpJK	OpJL	OpJM	OpJN	OpJO	OpJP	OpJQ	OpJR	OpJS	OpJT	OpJU	OpJV	OpJW	OpJX	OpJY	OpJZ	OpKA	OpKB	OpKC	OpKD	OpKE	OpKF	OpKG	OpKH	OpKI	OpKJ	OpKK	OpKL	OpKM	OpKN	OpKO	OpKP	OpKQ	OpKR	OpKS	OpKT	OpKU	OpKV	OpKW	OpKX	OpKY	OpKZ	OpLA	OpLB	OpLC	OpLD	OpLE	OpLF	OpLG	OpLH	OpLI	OpLJ	OpLK	OpLL	OpLM	OpLN	OpLO	OpLP	OpLQ	OpLR	OpLS	OpLT	OpLU	OpLV	OpLW	OpLX	OpLY	OpLZ	OpMA	OpMB	OpMC	OpMD	OpME	OpMF	OpMG	OpMH	OpMI	OpMJ	OpMK	OpML	OpMM	OpMN	OpMO	OpMP	OpMQ	OpMR	OpMS	OpMT	OpMU	OpMV	OpMW	OpMX	OpMY	OpMZ	OpNA	OpNB	OpNC	OpND	OpNE	OpNF	OpNG	OpNH	OpNI	OpNJ	OpNK	OpNL	OpNM	OpNN	OpNO	OpNP	OpNQ	OpNR	OpNS	OpNT	OpNU	OpNV	OpNW	OpNX	OpNY	OpNZ	OpOA	OpOB	OpOC	OpOD	OpOE	OpOF	OpOG	OpOH	OpOI	OpOJ	OpOK	OpOL	OpOM	OpON	OpOO	OpOP	OpOQ	OpOR	OpOS	OpOT	OpOU	OpOV	OpOW	OpOX	OpOY	OpOZ	OpPA	OpPB	OpPC	OpPD	OpPE	OpPF	OpPG	OpPH	OpPI	OpPJ	OpPK	OpPL	OpPM	OpPN	OpPO	OpPP	OpPQ	OpPR	OpPS	OpPT	OpPU	OpPV	OpPW	OpPX	OpPY	OpPZ	OpQA	OpQB	OpQC	OpQD	OpQE	OpQF	OpQG	OpQH	OpQI	OpQJ	OpQK	OpQL	OpQM	OpQN	OpQO	OpQP	OpQQ	OpQR	OpQS	OpQT	OpQU	OpQV	OpQW	OpQX	OpQY	OpQZ	OpRA	OpRB	OpRC	OpRD	OpRE	OpRF	OpRG	OpRH	OpRI	OpRJ	OpRK	OpRL	OpRM	OpRN	OpRO	OpRP	OpRQ	OpRR	OpRS	OpRT	OpRU	OpRV	OpRW	OpRX	OpRY	OpRZ	OpSA	OpSB	OpSC	OpSD	OpSE	OpSF	OpSG

N6

 CCKDABO KDJ11 B CLUSTER MAC111 30(1046)
 CCKDAB.P11 05-APR 84 16:45

 05-APR-84 16:45 PAGE 79
 ***** DOUBLE OPERAND TESTS *****

SEQ 0078

```

4160 015526 104001      1$:  ERROR  +1          ;CPU ERROR
4161                                     ;CC SHOULD=0101
4162 015530 005104      2$:  COM    R4          ;R4=-1
4163 015532 000277      SCC
4164 015534 000251      .WORD  251          ;CC=0110
4165 015536 042704 077777 BIC    077777,R4      ;CC=1000
4166 015542 001403      BEQ    3$          ;ERROR IF Z SET
4167 015544 102402      BVS    3$          ;ERROR IF V SET
4168 015546 103401      BCS    3$          ;ERROR IF C SET
4169 015550 100401      BMI    4$          ;EXIT IF N SET
4170 015552 104001      3$:  ERROR  +1          ;CPU ERROR
4171                                     ;CC SHOULD= 1000
4172 015554      4$:
4173
4174
4175 015554      MBSCC:
4176
4177      ;      TEST BIS CONDITION CODES
4178 015554 005004      CLR    R4          ;R4=0
4179 015556 000277      SCC
4180 015560 000246      .WORD  246          ;CC=1001
4181 015562 052704 000000 BIS    00,R4          ;R4=0, CC=0101
4182 015566 100403      BMI    1$          ;ERROR IF MINUS
4183 015570 102402      BVS    1$          ;ERROR IF V SET
4184 015572 103001      BCC    1$          ;ERROR IF C CLEAR
4185 015574 001401      BEQ    2$          ;BRANCH IF GOOD
4186 015576 104001      1$:  ERROR  +1          ;CPU ERROR
4187                                     ;BIS CC FAILED
4188 015600 000277      2$:  SCC
4189 015602 000241      CLC          ;CC=1110
4190 015604 052704 100076 BIS    0100076,R4      ;R4=100076, CC=1000
4191 015610 001403      BEQ    3$          ;ERROR IF Z SET
4192 015612 102402      BVS    3$          ;ERROR IF V SET
4193 015614 103401      BCS    3$          ;ERROR IF C SET
4194 015616 100401      BMI    4$          ;BRANCH IF GOOD
4195 015620 104001      3$:  ERROR  +1          ;CPU ERROR
4196                                     ;BAD BIS CC
4197 015622      4$:
4198
4199
4200 015622      MDCCC:
4201
4202      ;      TEST DEC, INC CONDITION CODES
4203 015622 012704 077777 MOV    077777,R4      ;R4=77777
4204 015626 000257      CCC
4205 015630 000261      SEC          ;CC=0001
4206 015632 005204      INC    R4          ;R4=100000, CC=0011
4207 015634 001403      BEQ    1$          ;ERROR IF ZERO
4208 015636 100002      BPL    1$          ;ERROR IF POSITIVE
4209 015640 102001      BVC    1$          ;ERROR IF V CLEAR
4210 015642 103401      BCS    2$          ;BRANCH IF GOOD
4211 015644 104001      1$:  ERROR  +1          ;CPU ERROR
4212                                     ;INC FAILED
4213 015646 000257      2$:  CCC
4214 015650 005204      INC    R4          ;R4=100001, CC=1000
4215 015652 103413      BCS    3$          ;ERROR IF C SET

```

CORDARO RDJ11 B CLUSTER MACY11 30(1046)
COKDAH,P11 05 APR 84 16:45

05 APR 84 16:45 PAGE 80
***** DOUBLE OPERAND TESTS *****

SEQ 0079

4216	015654	102412	BVS	3:		ERROR IF V SET
4217	015656	005304	DEC	R4		R4-100000, CC-1000
4218	015660	102410	BVS	3:		ERROR IF V SET
4219	015662	103407	BCS	3:		ERROR IF C SET
4220	015664	000277	SCC			
4221	015666	000257	WORD	25,2		CC-0101
4222	015670	005304	DEC	R4		R4-77777, CC-1011
4223	015672	001403	BEQ	3:		ERROR IF Z SET
4224	015674	102002	BVC	3:		ERROR IF V CLEAR
4225	015676	103001	BCC	3:		ERROR IF C CLEAR
4226	015700	100001	RPI	4:		BRANCH IF GOOD
4227	015702	104001	3:	ERROR	1	CPU ERROR
4228						BAD CC
4229	015704		4:			
4230						
4231						
4232	015704					
4233						
4234						
4235						
4236						
4237	015704	000277	SCC			
4238	015706	000244	CLR			CC-1011
4239	015710	005004	CLR	R4		R4-0, CC-0100
4240	015712	100403	BMI	1:		ERROR IF MINUS
4241	015714	102402	BVS	1:		ERROR IF V SET
4242	015716	103401	BCS	1:		ERROR IF C SET
4243	015720	001401	BEQ	2:		BRANCH IF GOOD
4244	015722	104001	1:	ERROR	1	CPU ERROR
4245						BAD CC ON CLR
4246	015724	005104	2:	COM	R4	R4-1
4247	015726	000277	SCC			CC-1111
4248	015730	005704	TST	R4		CC-1000
4249	015732	001403	BEQ	3:		ERROR IF Z SET
4250	015734	102402	BVS	3:		ERROR IF V SET
4251	015736	103401	BCS	3:		ERROR IF C SET
4252	015740	100401	BMI	4:		BRANCH IF GOOD
4253	015742	104001	3:	ERROR	1	CPU ERROR
4254						BAD TST CC
4255	015744	000277	4:	SCC		CC-1111
4256	015746	000304	SWAB	R4		CC-1000
4257	015750	102402	BVS	3:		ERROR IF V SET
4258	015752	103401	BCS	3:		ERROR IF C SET
4259	015754	100401	BMI	6:		BRANCH IF GOOD
4260	015756	104001	5:	ERROR	1	CPU ERROR
4261						BAD SWAB CC
4262	015760		6:			
4263						
4264						
4265	015760					
4266						
4267						
4268	015760	012704	012704			
4269	015764	012701	000001			
4270	015770	000257				
4271	015772	060401				

C7

CORDABO R0311 B CLUSTER MACY11 30(1046) 05 APR 84 16:45 PAGE 81
 CORDAB.P11 05 APR 84 16:45 ***** DOUBLE OPERAND TESTS *****

SEQ 0080

4272	015774	102003		BVC	1:		ERROR IF V CLEAR
4273	015776	103402		BCS	1:		ERROR IF CARRY
4274	016000	001401		BEQ	1:		ERROR IF Z SET
4275	016002	100401		BMI	2:		BRANCH IF GOOD
4276	016004	104001	1:	ERROR	+	1	CPU ERROR
4277							ICC SHOULD = 1010
4278	016006	005204	2:	INC	R4		R4 = 100000
4279	016010	060401		ADD	R4,R1		100000 + 100000 = 0 IN R1
4280	016012	102002		BVC	3:		ERROR IF V CLEAR
4281	016014	103001		BCC	3:		ERROR IF CARRY CLEAR
4282	016016	001401		BEQ	4:		BRANCH IF GOOD
4283	016020	104001	3:	ERROR	+	1	CPU ERROR
4284							ICC SHOULD = 0111
4285	016022	060401	4:	ADD	R4,R1		0 + 100000 = 100000
4286	016024	102402		BVS	5:		ERROR IF V SET
4287	016026	103401		BCS	5:		ERROR IF C SET
4288	016030	100401		BMI	6:		BRANCH IF GOOD
4289	016032	104001	5:	ERROR	+	1	CPU ERROR
4290							ICC SHOULD = 1000
4291	016034		6:				
4292							
4293							
4294	016034			MACCC:			
4295							
4296							
4297	016034	012704	177777				
4298	016040	000277		TEST ADC CONDITION CODES			
4299	016042	005504		MOV	#177777,R4		R4 = 177777
4300	016044	100403		SCC			ICC = 1111
4301	016046	102402		ADC	R4		R4 = 0 CC = 0101
4302	016050	103001		BMI	1:		ERROR IF MINUS
4303	016052	001401		BVS	1:		ERROR IF V SET
4304	016054	104001		BCC	1:		ERROR IF C SET
4305				BEQ	2:		BRANCH IF GOOD
4306	016056	012704	077777	1:	ERROR	+	1
4307	016062	000277					CPU ERROR
4308	016064	000242					BAD ADC
4309	016066	005504		2:	MOV	#077777,R4	R4 = 077777
4310	016070	100003		SCC			ICC = 1101
4311	016072	103402		CLV			R4 = 100000 CC = 1010
4312	016074	001401		ADC	R4		ERROR IF PLUS
4313	016076	102401		BPL	3:		ERROR IF C SET
4314	016100	104001		BCH	3:		ERROR IF ZERO
4315				BEQ	3:		BRANCH IF GOOD
4316	016102	000277		BVS	4:		CPU ERROR
4317	016104	005504		4:	SCC		BAD ADC
4318	016106	102402		ADC	R4		ICC = 1111
4319	016110	103401		BVS	5:		R4 = 100000 CC = 1000
4320	016112	100401		BC	5:		ERROR IF V SET
4321	016114	104001		BMI	6:		ERROR IF C SET
4322				5:	ERROR	+	1
4323	016116						CPU ERROR
4324							BAD ADC CC SHOULD = 1000
4325							
4326	016116						
4327							
4328							
4329							
4330							
4331							
4332							
4333							
4334							
4335							
4336							
4337							
4338							
4339							
4340							
4341							
4342							
4343							
4344							
4345							
4346							
4347							
4348							
4349							
4350							
4351							
4352							
4353							
4354							
4355							
4356							
4357							
4358							
4359							
4360							
4361							
4362							
4363							
4364							
4365							
4366							
4367							
4368							
4369							
4370							
4371							
4372							
4373							
4374							
4375							
4376							
4377							
4378							
4379							
4380							
4381							
4382							
4383							
4384							
4385							
4386							
4387							
4388							
4389							
4390							
4391							
4392							
4393							
4394							
4395							
4396							
4397							
4398							
4399							
4400							
4401							
4402							
4403							
4404							
4405							
4406							
4407							
4408							
4409							
4410							
4411							
4412							
4413							
4414							
4415							
4416							
4417							
4418							
4419							
4420							
4421							
4422							
4423							
4424							
4425							
4426							
4427							
4428							
4429							
4430							
4431							
4432							
4433							
4434							
4435							
4436							
4437							
4438							
4439							
4440							
4441							
4442							
4443							
4444							
4445							
4446							
4447							
4448							
4449							
4450							
4451							
4452							
4453							
4454							
4455							
4456							
4457							
4458							
4459							
4460							
4461							
4462							
4463							
4464							
4465							
4466							
4467							
4468							
4469							
4470							
4471							
4472							
4473							
4474							
4475							
4476							
4477							
4478							
4479							
4480							
4481							
4482							
4483							
4484							

CORDARO RDJ11 B CLUSTER MAC111 30(1046)
COKDAR.P11 05-APR-84 16:45

05 APR 84 16:45 PAGE 82
***** DOUBLE OPERAND TESTS *****

SEQ 0081

4328				:	TEST NEG, CMP, COM CONDITION CODES	
4329	016116	012704	077777		MOV 077777,R4	;R4=77777
4330	016122	000257			CCC	;CC=0000
4331	016124	005404			NEG R4	;R4=100001 CC=1001
4332	016126	102403			BVS 1\$	;ERROR IF V SET
4333	016130	103002			BCC 1\$	;ERROR IF C CLEAR
4334	016132	001401			BEQ 1\$	;ERROR IF Z SET
4335	016134	100401			BMI 2\$	;BRANCH IF GOOD
4336	016136	104001		1\$:	ERROR +1	;CPU ERROR
4337						;BAD NEGATE
4338	016140	005004		2\$:	CLR R4	;R4=0
4339	016142	000257			CCC	;CC=0000
4340	016144	005404			NEG R4	;CC=0101
4341	016146	100403			BMI 3\$	;ERROR IF N SET
4342	016150	103402			BCS 3\$	;ERROR IF C SET
4343	016152	102401			BVS 3\$	;ERROR IF V SET
4344	016154	001401			BEQ 4\$	;BRANCH IF GOOD
4345	016156	104001		3\$:	ERROR +1	;CPU ERROR
4346						;BAD NEG
4347	016160	012704	077777	4\$:	MOV 077777,R4	;R4=77777
4348	016164	012701	170000		MOV 0170000,R1	;R1=170000
4349	016170	000257			CCC	;CC=0000
4350	016172	020401			CMP R4,R1	;77777 - 170000 = 107777 CC= 1011
4351	016174	102003			BVC 5\$	;ERROR IF V CLEAR
4352	016176	103002			BCC 5\$	;ERROR IF C CLEAR
4353	016200	001401			BEQ 5\$	;ERROR IF ZERO
4354	016202	100401			BMI 6\$	;BRANCH IF GOOD
4355	016204	104001		5\$:	ERROR +1	;CPU ERROR
4356						;BAD CMP
4357	016206	000257		6\$:	CCC	
4358	016210	005101			COM R1	;R1=7777
4359	016212	100403			BMI 7\$	;ERROR IF MINUS
4360	016214	001402			BEQ 7\$	;ERROR IF ZERO
4361	016216	103001			BCC 7\$	;ERROR IF CARRY
4362	016220	102001			BVC 8\$	;BRANCH IF GOOD
4363	016222	104001		7\$:	ERROR +1	;CPU ERROR
4364						;BAD COM
4365	016224	000277		8\$:	SCC	
4366	016226	005101			COM R1	
4367	016230	100401			BMI 10\$	;BRANCH IF GOOD
4368	016232	104001		9\$:	ERROR +1	;CPU ERROR
4369						;BAD COM
4370	016234			10\$:		
4371						
4372						
4373	016234			MSBCC:		
4374						
4375				:	TEST SUB CONDITION CODES	
4376	016234	012704	077775		MOV 077775,R4	;R4=77775
4377	016240	000257			CCC	;CC=0000
4378	016242	162704	137757		SUB 0137757,R4	;77775 - 137757
4379						;TRY TO CAUSE AN ARITHMETIC OVERFLOW
4380	016246	102003			BVC 1\$	;ERROR IF V CLEAR
4381	016250	100002			BPL 1\$	;ERROR IF RESULT IS POSITIVE
4382	016252	001401			BEQ 1\$	;ERROR IF Z SET
4383	016254	103401			BCS 2\$	;BRANCH IF GOOD

```

4384 016256 104001 1$: ERROR +1 ;CPU ERROR
4385 ;BAD SUBTRACT
4386 016260 012704 000005 2$: MOV #5,R4 ;R4=5
4387 016264 000257 CCC ;CC=0000
4388 016266 162704 000012 SUB #12,R4 ;5-12=-5 AND SETS CARRY
4389 016272 103003 BCC 3$ ;ERROR IF CARRY CLEAR
4390 016274 102402 BVS 3$ ;ERROR IF OVERFLOW
4391 016276 001401 BEQ 3$ ;ERROR IF ZERO
4392 016300 100401 BMI 4$ ;BRANCH IF GOOD
4393 016302 104001 3$: ERROR +1 ;CPU ERROR
4394 ;SUBTRACT FAILED
4395 016304 4$:
4396
4397
4398 016304 MSBCCC:
4399
4400 ; TEST SBC CONDITION CODES
4401 016304 012704 100000 MOV #100000,R4 ;R4=100000
4402 016310 000257 CCC ;C=0000
4403 016312 005604 SBC R4 ;TRY TO SET V
4404 016314 100006 BPL 1$ ;ERROR IF N CLEAR
4405 016316 102405 BVS 1$ ;ERROR IF V SET (HAVENT SET C YET)
4406 016320 000261 SEC ;CC SHOULD = 1001
4407 016322 005604 SBC R4 ;TRY AGAIN TO SET V
4408 016324 102002 BVC 1$ ;ERROR IF V CLEAR
4409 016326 103401 BCS 1$ ;ERROR IF C SET
4410 016330 100001 BPL 2$ ;BRANCH IF GOOD
4411 016332 104001 1$: ERROR +1 ;CPU ERROR
4412 ;SBC FAILED
4413 016334 005004 2$: CLR R4 ;R4=0
4414 016336 000277 SCC
4415 016340 000241 CLC ;CC=1110
4416 016342 005604 SBC R4 ;TRY TO CAUSE C FLAG FAILURE
4417 016344 103410 BCS 3$ ;ERROR IF C SET
4418 016346 102407 BVS 3$ ;ERROR IF V SET
4419 016350 001006 BNE 3$ ;ERROR IF NOT ZERO
4420 016352 000261 SEC ;SET CARRY
4421 016354 005604 SBC R4 ;NOW, 0 - CARRY = 177777
4422 016356 103003 BCC 3$ ;ERROR IF CARRY CLEAR
4423 016360 102402 BVS 3$ ;ERROR IF V SET
4424 016362 001401 BEQ 3$ ;ERROR IF ZERO
4425 016364 100401 BMI 4$ ;BRANCH IF GOOD
4426 016366 104001 3$: ERROR +1 ;CPU ERROR
4427 ;SBC FAILED
4428 016370 4$:
4429
4430
4431 016370 MRLCC:
4432
4433 ; TEST ROL CONDITION CODES
4434 016370 012704 060000 MOV #60000,R4 ;R4= 0110000000000000
4435 016374 000257 CCC ;CC=0000
4436 016376 006104 ROL R4 ;R4= 1100000000000000
4437 016400 103402 BCS 1$ ;ERROR IF CARRY
4438 016402 102001 BVC 1$ ;ERROR IF V CLEAR
4439 016404 100401 BMI 2$ ;BRANCH IF GOOD

```



```

4440 016406 104001 1$: ERROR +1 ;CPU ERROR
4441 ;ROL FAILED
4442 016410 006104 2$: ROL R4 ;R4 = 1000000000000000
4443 016412 103002 BCC 3$ ;ERROR IF CARRY CLEAR
4444 016414 102401 BVS 3$ ;ERROR IF V SET
4445 016416 100401 BMI 4$ ;BRANCH IF GOOD
4446 016420 104001 3$: ERROR +1 ;CPU ERROR
4447 ;BAD ROL
4448 016422 006104 4$: ROL R4 ;R4 = 0000000000000001
4449 016424 102003 BVC 5$ ;ERROR IF V CLEAR
4450 016426 103002 BCC 5$ ;ERROR IF C CLEAR
4451 016430 100401 BMI 5$ ;ERROR IF MINUS
4452 016432 001001 BNE 6$ ;BRANCH IF GOOD
4453 016434 104001 5$: ERROR +1 ;CPU ERROR
4454 ;BAD ROL
4455 016436 006104 6$: ROL R4 ;R4 = 0000000000000011
4456 016440 102402 BVS 7$ ;ERROR IF V SET
4457 016442 103401 BCS 7$ ;ERROR IF C SET
4458 016444 100001 BPL 8$ ;BRANCH IF GOOD
4459 016446 104001 7$: ERROR +1 ;CPU ERROR
4460 ;BAD ROL
4461 016450 8$:
4462
4463
4464 016450 MRRCC:
4465
4466 : TEST ROR CONDITION CODES
4467 016450 012704 000003 MOV #3,R4 ;R4 = 0000000000000011
4468 016454 000257 CCC ;CC = 0000
4469 016456 006004 ROR R4 ;R4 = 0000000000000001
4470 016460 103002 BCC 1$ ;ERROR IF NO CARRY
4471 016462 102001 BVC 1$ ;ERROR IF V CLEAR
4472 016464 100001 BPL 2$ ;BRANCH IF GOOD
4473 016466 104001 1$: ERROR +1 ;CPU ERROR
4474 ;ROR FAILED
4475 016470 006004 2$: ROR R4 ;R4 = 1000000000000000
4476 016472 103002 BCC 3$ ;ERROR IF CARRY CLEAR
4477 016474 102401 BVS 3$ ;ERROR IF V SET
4478 016476 100401 BMI 4$ ;BRANCH IF GOOD
4479 016500 104001 3$: ERROR +1 ;CPU ERROR
4480 ;BAD ROR
4481 016502 006004 4$: ROR R4 ;R4 = 1100000000000000
4482 016504 102002 BVC 5$ ;ERROR IF V
4483 016506 103401 BCS 5$ ;ERROR IF C SET
4484 016510 100401 BMI 6$ ;BRANCH IF GOOD
4485 016512 104001 5$: ERROR +1 ;CPU ERROR
4486 ;BAD ROR
4487 016514 006004 6$: ROR R4 ;R4 = 0110000000000000
4488 016516 102402 BVS 7$ ;ERROR IF V SET
4489 016520 103401 BCS 7$ ;ERROR IF C SET
4490 016522 100001 BPL 8$ ;BRANCH IF GOOD
4491 016524 104001 7$: ERROR +1 ;CPU ERROR
4492 ;BAD ROR
4493 016526 8$:
4494
4495

```

COKDABO KDJ11-B CLUSTER MACY11 30(1046)
COKDAB.P11 05-APR-84 16:45

05-APR-84 16:45 PAGE 85
***** DOUBLE OPERAND TESTS *****

SEQ 0084

```

4496 016526      XCBIT:
4497
4498      ;      TEST C BIT WITH ROR/ROL
4499      ;*****
4500      ;THIS TEST IS TO CHECK FOR A SLOW C BIT PATH INTERNAL TO THE J11 DATA CHIP
4501      ;PROBLEM IS ONLY EXHIBITED ON EARLY MASK SETS (1590 AND 1593)
4502 016526 012701 052525      MOV      #52525,R1      ;INIT R1 WITH DATA
4503 016532 000241      CLC      ;CLEAR THE C BIT
4504 016534 006001      ROR      R1      ;R1=025252, C BIT =1
4505 016536 006001      ROR      R1      ;R1=112525, C BIT =0
4506 016540 006001      ROR      R1      ;R1=045252, C BIT =1
4507 016542 103401      BCS      1$      ;BRANCH IF CARRY BIT SET
4508 016544 104001      ERROR    +1      ;CPU ERROR
4509 016546 022701 045252      1$:    CMP      #45252,R1      ;IS DATA IN R1 = TO EXPECTED DATA?
4510 016552 001401      BEQ      2$      ;BRANCH IF YES
4511 016554 104001      ERROR    +1      ;CPU ERROR
4512 016556 012701 125252      2$:    MOV      #125252,R1      ;SET UP R1
4513 016562 000241      CLC      ;CLEAR THE CARRY BIT
4514 016564 006101      ROL      R1      ;R1=052524, C BIT =1
4515 016566 006101      ROL      R1      ;R1=125251, C BIT =0
4516 016570 006101      ROL      R1      ;R1=052522, C BIT =1
4517 016572 103401      BCS      3$      ;BRANCH IF CARRY SET
4518 016574 104001      ERROR    +1      ;CPU ERROR
4519 016576 022701 052522      3$:    CMP      #052522, R1      ;IS DATA IN R1 = TO EXPECTED DATA?
4520 016602 001401      BEQ      4$
4521 016604 104001      ERROR    +1      ;CPU ERROR
4522 016606
4523
4524 016606      MALCC:
4525
4526      ;      TEST ASL CONDITION CODES
4527 016606 012704 060000      MOV      #60000,R4      ;R4= 0110000000000000
4528 016612 000257      CCC      ;CC=0000
4529 016614 006304      ASL      R4      ;C=0 R4= 1100000000000000
4530 016616 103402      BCS      1$      ;ERROR IF CARRY
4531 016620 102001      BVC      1$      ;ERROR IF V CLEAR
4532 016622 100401      BMI      2$      ;BRANCH IF GOOD
4533 016624 104001      1$:    ERROR    +1      ;CPU ERROR
4534      ;ASL FAILED
4535 016626 006304      2$:    ASL      R4      ;C=1 R4= 1000000000000000
4536 016630 103002      BCC      3$      ;ERROR IF CARRY CLEAR
4537 016632 102401      BVS      3$      ;ERROR IF V SET
4538 016634 100401      BMI      4$      ;BRANCH IF GOOD
4539 016636 104001      3$:    ERROR    +1      ;CPU ERROR
4540      ;BAD ASL
4541 016640 006304      4$:    ASL      R4      ;C=1 R4= 0000000000000000
4542 016642 102003      BVC      5$      ;ERROR IF V CLEAR
4543 016644 103002      BCC      5$      ;ERROR IF C CLEAR
4544 016646 100401      BMI      5$      ;ERROR IF MINUS
4545 016650 001401      BEQ      6$      ;BRANCH IF GOOD
4546 016652 104001      5$:    ERROR    +1      ;CPU ERROR
4547      ;BAD ASL
4548 016654 006304      6$:    ASL      R4      ;C=0 R4= 0000000000000000
4549 016656 102402      BVS      7$      ;ERROR IF V SET
4550 016660 103401      BCS      7$      ;ERROR IF C SET
4551 016662 100001      BPL      8$      ;BRANCH IF GOOD

```


H7

COKDABO KDJ11 8 CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 86
 COKDAB.P11 05-APR-84 16:45 ***** DOUBLE OPERAND TESTS *****

SEQ 0085

```

4552 016664 104001      7$:      ERROR      +1          ;CPU ERROR
4553                                     ;BAD ASL
4554 016666      8$:
4555
4556 016666      MARCC:
4557
4558      ;      TEST ASR CONDITION CODES
4559 016666 012704 000341      MOV      #341,R4          ;R4= 0000000011100001
4560 016672 000257      CCC          ;CC=0000
4561 016674 006204      ASR      R4          ;R4= 0000000001110000
4562 016676 103002      BCC      1$          ;ERROR IF NO CARRY
4563 016700 102001      BVC      1$          ;ERROR IF V CLEAR
4564 016702 100001      BPL      2$          ;BRANCH IF GOOD
4565 016704 104001      1$:      ERROR      +1          ;CPU ERROR
4566                                     ;ASR FAILED
4567 016706 052704 100001      2$:      BIS      #100001,R4      ;R4= 1000000001110001
4568 016712 006204      ASR      R4          ;R4= 1100000000111000
4569 016714 103002      BCC      3$          ;ERROR IF CARRY CLEAR
4570 016716 102401      BVS      3$          ;ERROR IF V SET
4571 016720 100401      BMI      4$          ;BRANCH IF GOOD
4572 016722 104001      3$:      ERROR      +1          ;CPU ERROR
4573                                     ;BAD ASR
4574 016724 006204      4$:      ASR      R4          ;R4= 1110000000011100
4575 016726 102002      BVC      5$          ;ERROR IF V
4576 016730 103401      BCS      5$          ;ERROR IF C SET
4577 016732 100401      BMI      6$          ;BRANCH IF GOOD
4578 016734 104001      5$:      ERROR      +1          ;CPU ERROR
4579                                     ;BAD ASR
4580 016736 006204      6$:      ASR      R4          ;R4= 1111000000001110
4581 016740 102005      BVC      7$          ;ERROR IF V CLEAR
4582 016742 103404      BCS      7$          ;ERROR IF C SET
4583 016744 100003      BPL      7$          ;ERROR IF PLUS
4584 016746 022704 170016      CMP      #170016,R4      ;SEE IF EXPECTED RESULT
4585 016752 001401      BEQ      8$          ;BRANCH IF GOOD
4586 016754 104001      7$:      ERROR      +1          ;CPU ERROR
4587                                     ;BAD ASR
4588 016756      8$:
4589
4590
4591 016756      MSXICC:
4592
4593      ;      TEST SXT CONDITION CODES / -+0-
4594 016756 012704 123456      MOV      #123456,R4          ;R4=123456
4595 016762 010401      MOV      R4,R1          ;SAVE CONTENTS
4596 016764 000257      CCC          ;CC=0000
4597 016766 006704      SXT      R4          ;R4=0 CC=0100
4598 016770 103403      BCS      1$          ;ERROR IF CARRY
4599 016772 100402      BMI      1$          ;ERROR IF MINUS
4600 016774 102401      BVS      1$          ;ERROR IF OVERFLOW
4601 016776 001401      BEQ      2$          ;BRANCH IF GOOD
4602 017000 104001      1$:      ERROR      +1          ;CPU ERROR
4603                                     ;BAD SXT
4604 017002 010104      2$:      MOV      R1,R4          ;RESTORE R4
4605 017004 000277      SCC          ;CC=1111
4606 017006 006704      SXT      R4          ;R4= 1 CC=1001
4607 017010 001405      BEQ      3$          ;ERROR IF ZERO

```

```

4608 017012 100004          BPL      3$          ;ERROR IF PLUS
4609 017014 103003          BCC      3$          ;ERROR IF NO CARRY
4610 017016 102402          BVS      3$          ;ERROR IF OVERFLOW
4611 017020 005104          COM      R4          ;R4=0
4612 017022 001401          BEQ      4$          ;BRANCH IF GOOD
4613 017024 104001          3$:  ERROR      +1          ;CPU ERROR
4614                                ;BAD SXT
4615 017026          4$:
4616
4617
4618 017026          MXRCC:
4619
4620          ;      TEST XOR CONDITION CODES / **0-
4621 017026 012704 123456      MOV      0123456,R4          ;R4=123456
4622 017032 012701 052525      MOV      052525,R1          ;R1=52525
4623 017036 000257          CCC          ;CC=0000
4624 017040 074104          XOR      R1,R4          ;*TI* R4=171173
4625 017042 102403          BVS      1$          ;ERROR IF OVERFLOW
4626 017044 001402          BEQ      1$          ;ERROR IF ZERO
4627 017046 103401          BCS      1$          ;ERROR IF CARRY
4628 017050 100401          BMI      2$          ;BRANCH IF GOOD
4629 017052 104001          1$:  ERROR      +1          ;CPU ERROR
4630                                ;BAD XOR
4631 017054 012701 125252      2$:  MOV      0125252,R1          ;R1=125252
4632 017060 000277          SCC          ;CC=1111
4633 017062 074104          XOR      R1,R4          ;R4=054321
4634 017064 100403          BMI      3$          ;ERROR IF MINUS
4635 017066 001402          BEQ      3$          ;ERROR IF ZERO
4636 017070 103001          BCC      3$          ;ERROR IF CARRY CLEAR
4637 017072 102001          BVC      4$          ;BRANCH IF GOOD
4638 017074 104001          3$:  ERROR      +1          ;CPU ERROR
4639                                ;BAD XOR
4640 017076 074404          4$:  XOR      R4,R4          ;R4=0
4641 017100 102406          BVS      5$          ;ERROR IF OVERFLOW
4642 017102 100405          BMI      5$          ;ERROR IF MINUS
4643 017104 103004          BCC      5$          ;ERROR IF NO CARRY
4644 017106 001003          BNE      5$          ;ERROR IF NOT ZERO
4645 017110 022704 000000      CMP      00,R4          ;SEE IF EXPECTED RESULT
4646 017114 001401          BEQ      6$          ;BRANCH IF GOOD
4647 017116 104001          5$:  ERROR      +1          ;CPU ERROR
4648                                ;BAD XOR
4649 017120          6$:
4650
4651
4652          ;
4653 017120          MSXT:
4654
4655          ;      TEST SXT (SIGN EXTEND INSTRUCTION)
4656          ;*****
4657          ;AN ADDITIONAL TEST IS INCLUDED TO CHECK FOR A SLOW N BIT PATH
4658          ;ON A TRANSITION FROM ZERO TO ONE INTERNAL TO THE J11 DATA CHIP
4659          ;THE PROBLEM IS ONLY EXHIBITED ON EARLY MASK SETS (1590 OR 1593)
4660 017120 005004          CLR      R4          ;TRASH R4
4661 017122 000257          CCC          ;CC=0000
4662 017124 000271          .WORD      271          ;CC=1001
4663 017126 006704          SXT      R4          ;*TEST INSTRUCTION

```

```

4664 017130 102405      BVS      1$      ;BRANCH IF OVERFLOW IS NOT CLEARED
4665 017132 100004      BPL      1$      ;BRANCH IF N BIT EFFECTED
4666 017134 001403      BEQ      1$      ;BRANCH IF Z BIT EFFECTED
4667 017136 103002      BCC      1$      ;BRANCH IF C BIT EFFECTED
4668 017140 005204      INC      R4
4669 017142 001401      BEQ      2$      ;BRANCH IF R4 =0
4670 017144 104001      1$:      ERROR    +1      ;CPU ERROR
4671                                ;CC SHOULD HAVE = 1101
4672 017146 000277      2$:      SCC
4673 017150 000250      CLN                                ;CC=0111
4674 017152 005004      CLR      R4
4675 017154 012714 000055      MOV      #55,(R4)      ;TRASH R4
4676 017160 006714      SXT      (R4)      ;*TEST INSTRUCTION
4677 017162 001005      BNE      3$      ;BRANCH IF BIT EFFECTED
4678 017164 102404      BVS      3$      ;BRANCH IF OVERFLOW
4679 017166 103403      BCS      3$
4680 017170 100402      BMI      3$      ;BRANCH IF N IS SET
4681 017172 005714      TST      (R4)      ;VERIFY INSTRUCTION WORKED
4682 017174 001401      BEQ      4$      ;BRANCH IF R4=0
4683 017176 104001      3$:      ERROR    +1      ;CPU ERROR
4684                                ;SXT FAILED
4685
4686      ;      NOW TEST FOR SLOW N BIT IN J11 DATA CHIP
4687
4688 017200 012700 177777      4$:      MOV      # 1,R0      ;R0=177777, N BIT = 1
4689 017204 005004      CLR      R4      ;CLEAR THE N BIT
4690 017206 006700      SXT      R0      ;***TEST INSTRUCTION***
4691                                ;TEST N BIT TRANSITION 1 TO 0
4692 017210 005700      TST      R0      ;R0 SHOULD = 0
4693 017212 001401      BEQ      5$      ;BRANCH IF OK
4694 017214 104001      ERROR    +1      ;CPU ERROR
4695 017216 005000      5$:      CLR      R0      ;CLEAR R0, N BIT = 0
4696 017220 012704 177777      MOV      # 1,R4      ;SET N BIT
4697 017224 006700      SXT      R0      ;***TEST INSTRUCTION***
4698                                ;TEST N BIT TRANSITION 0 TO 1
4699 017226 022700 177777      CMP      # 1,R0      ;R0 SHOULD = 177777
4700 017232 001401      BEQ      6$      ;BRANCH IF OK
4701 017234 104001      ERROR    +1      ;CPU ERROR
4702 017236
4703      6$:
4704      ;      MXOR:
4705
4706      ;      TEST XOR
4707 017236 012701 007643      MOV      #7643,R1      ;SETUP DATA
4708 017242 012704 133333      MOV      #133333,R4      ;SETUP DATA
4709 017246 000277      SCC
4710 017250 074401      XOR      R4,R1      ;*TEST INSTRUCTION
4711 017252 100006      BPL      1$      ;BRANCH IF PLUS TO ERROR
4712 017254 001405      BEQ      1$      ;ERROR IF ZERO
4713 017256 103004      BCC      1$      ;ERROR IF CARRY CLEAR
4714 017260 102403      BVS      1$      ;ERROR IF V SET
4715 017262 020127 134570      CMP      R1,#134570      ;VERIFY CORRECT RESULT
4716 017266 001401      BEQ      2$      ;BRANCH IF GOOD
4717 017270 104001      1$:      ERROR    +1      ;CPU ERROR
4718                                ;BAD XOR
4719 017272 010102      2$:      MOV      R1,R2

```

CORDBO RDJ11 B CLUSTER MACY11 30(1046) 05 APR 84 16:45 PAGE 89
 CORDAB.P11 05-APR-84 16:45 ***** DOUBLE OPERAND TESTS *****

SEQ 0088

```

4720 017274 000257 CCC
4721 017276 074402 XOR R4,R2 ;*TEST INSTRUCTION
4722 017300 100405 BMI 3$ ;ERROR IF MINUS
4723 017302 102404 BVS 3$ ;ERROR IF OVERFLOW
4724 017304 103403 BCS 3$ ;ERROR IF CARRY
4725 017306 020227 007643 CMP R2,#7643
4726 017312 001401 BEQ 4$ ;BRANCH IF GOOD
4727 017314 104001 3$: ERROR +1 ;CPU ERROR
4728 ;BAD XOR
4729 017316 4$:
4730
4731
4732 017316 ;MSOB:
4733
4734 ;
4735 017316 012704 000555 ;TEST SOB
4736 017322 000277 MOV #555,R4 ;SETUP TEST COUNTER
4737 017324 103015 SCC ;CC=17
4738 017326 102014 1$: BCC 2$ ;ERROR IF CARRY CLEAR
4739 017330 100013 BVC 2$ ;ERROR IF NO OVERFLOW
4740 017332 001012 BPL 2$ ;ERROR IF PLUS
4741 017334 077405 BNE 2$ ;ERROR IF ZERO
4742 017336 103005 SOB R4,1$ ;*TEST INSTRUCTION
4743 017340 102004 BCC 3$ ;ERROR IF CARRY CLEAR
4744 017342 100003 BVC 3$ ;ERROR IF NO OVERFLOW
4745 017344 001002 BPL 3$ ;ERROR IF PLUS
4746 017346 000167 000010 BNE 3$ ;ERROR IF ZERO
4747 017352 104001 3$: JMP 4$ ;CPU ERROR
4748 ;CC EFFECTED DURING TEST
4749 017354 000167 000002 JMP 4$
4750 017360 104001 2$: ERROR +1 ;CPU ERROR
4751 ;CC EFFECTED AFTER TEST
4752 017362 020427 000000 4$: CMP R4,#0 ;IS R4 CORRECT
4753 017366 001401 BEQ 5$ ;YES GO ON
4754 017370 104001 ERROR +1 ;CPU ERROR
4755 ;NO GO TO ERROR
4756 017372 5$:
4757
4758
4759 017372 ;MMARK:
4760
4761 ;
4762 017372 012706 000700 MARK INSTRUCTION TEST
4763 017376 012737 125252 000776 MOV #STBOT-100,SP ;SETUP TEST STACK - 700
4764 017404 012705 017422 MOV #125252,#STBOT-2 ;SET UP NEW R5 VALUE ON STACK
4765 017410 012746 006437 MOV #1$,R5 ;PUT NEW PC IN R5
4766 017414 000277 MOV #MARK+37,(SP) ;INSERT MARK 37 INSTRUCTION ONTO STACK
4767 017416 000116 SCC
4768 JMP (SP) ;*TEST INSTRUCTION
4769 017420 104001 ERROR +1 ;MARK INSTRUCTION SHOULD HAVE GONE TO 1$
4770 ;CPU ERROR
4771 017422 101002 1$: BHI 2$ ;ERROR IF C OR Z BIT CLEAR
4772 017424 100001 BPL 2$ ;ERROR IF N BIT CLEAR
4773 017426 102401 BVS 3$ ;BRANCH IF V BIT SET
4774 ;BAD CONDITION CODES ON MARK
4775 017430 104001 2$: ERROR +1 ;CPU ERROR

```

```

4776 017432 022705 125252 3$: CMP #125252,R5 ;VERIFY R5
4777 017436 001401 BEQ 4$ ;BRANCH IF GOOD
4778 017440 104001 ERROR +1 ;CPU ERROR
4779
4780 017442 020627 001000 4$: CMP SP,#STBOT ;VERIFY THAT STACK IS CORRECT
4781 017446 001401 BEQ 15$ ;BRANCH IF OK
4782 ;ERROR! STACK WAS NOT CORRECT AFTER MARK
4783 017450 104001 ERROR +1 ;CPU ERROR
4784
4785 017452 012746 052525 15$: MOV #52525,-(SP) ;SETUP EXPECTED R5
4786 017456 012746 006400 MOV #6400,-(SP) ;MOVE MARK 0 INSTRUCTION ON STACK
4787 017452 010605 MOV SP,R5 ;R5=ADDRESS OF INSTRUCTION
4788 017464 004767 000004 JSR PC,5$ ;LEAVE 6$ ON STACK
4789 017470 000167 000006 6$: JMP 16$ ;MARK RETURNED CORRECTLY
4790
4791 017474 000257 5$: CCC ;CLEAR THE CONDITION CODES
4792 017476 000205 RTS R5 ;RETURN TO MARK INSTRUCTION
4793 ;NEXT INSTRUCTION ON STACK IS THE RETURN
4794 ;FROM THE JSR
4795 ;ERROR! BAD MARK SEQUENCE
4796 017500 104001 ERROR +1 ;CPU ERROR
4797
4798 017502 101400 16$: BLOS 7$ ;IS C OR Z BIT SET?
4799 017504 100401 BMI 7$ ;IS N BIT SET?
4800 017506 102001 BVC 8$ ;IS V BIT SET?
4801 ;ERROR! CONDITIONS CODES INCORRECT
4802 017510 104001 7$: ERROR +1 ;CPU ERROR
4803
4804 017512 020627 001000 8$: CMP R6,#STBOT ;IS THE STACK CORRECT?
4805 017516 001401 BEQ 9$ ;BRANCH IF YES
4806 ;ERROR! BAD STACK CLEANUP
4807 017520 104001 ERROR +1 ;CPU ERROR
4808 017522 022705 052525 9$: CMP #52525,R5 ;VERIFY THAT R5 WAS LOADED PROPERLY.
4809 017526 001401 BEQ 10$ ;IF OK, GO TO NEXT TEST.
4810 ;ERROR! R5 WAS NOT CORRECT AFTER MARK.
4811 017530 104001 ERROR +1 ;CPU ERROR
4812 017532 10$:
4813 017532 XME100:
4814
4815 ; TEST CLEAR CONDITION CODES INSTRUCTION
4816 017532 012737 030017 177776 MOV #30017,@#177776 ;SETUP PSW
4817 017540 000257 CCC ; TEST INSTRUCTION
4818 017542 022737 030000 177776 CMP #30000,@#177776 ;DID IT CLEAR ALL CONDITION CODE BITS
4819 017550 001401 BEQ 1$ ;YES GO ON
4820 017552 104001 ERROR +1 ;CPU ERROR
4821 ;NO GO TO ERROR
4822 017554 1$:
4823 ;
4824 017554 XME101:
4825
4826 ; TEST CLEAR C BIT INSTRUCTION
4827 017554 012737 030017 177776 MOV #30017,@#177776 ;SETUP PSW
4828 017562 000241 CLC ; TEST INSTRUCTION
4829 017564 022737 030016 177776 CMP #30016,@#177776 ;DID IT CLEAR CARRY BIT
4830 017572 001401 BEQ 1$ ;YES GO ON
4831 ;C BIT NOT CLEAR GO TO ERROR

```

Address	Offset	Instruction	Op-Code	Target Address	Comments
4832	017574	104001			
4833	017576				
4834					
4835	017576				
4836					
4837	017576	012737	030017	177776	
4838	017604	000250			
4839	017606	022737	030007	177776	
4840	017614	001401			
4841	017616	104001			
4842					
4843	017620				
4844					
4845	017620				
4846					
4847	017620	012737	030017	177776	
4848	017626	000242			
4849	017630	022737	030015	177776	
4850	017636	001401			
4851	017640	104001			
4852					
4853	017642				
4854					
4855	017642				
4856					
4857	017642	012737	030017	177776	
4858	017650	000244			
4859	017652	022737	030013	177776	
4860	017660	001401			
4861	017662	104001			
4862					
4863	017664				
4864					
4865	017664				
4866					
4867	017664	012737	030000	177776	
4868	017672	000277			
4869	017674	022737	030017	177776	
4870	017702	001401			
4871	017704	104001			
4872					
4873	017706				
4874					
4875	017706				
4876					
4877	017706	012737	030000	177776	
4878	017714	000261			
4879	017716	022737	030001	177776	
4880	017724	001401			
4881	017726	104001			
4882					
4883	017730				
4884					
4885	017730				
4886					
4887	017730	012737	030000	177776	

N7

COKDABO KDJ11-B CLUSTER MACY11 30(1046)
COKDAB.P11 05-APR-84 16:4505-APR-84 16:45 PAGE 92
***** DOUBLE OPERAND TESTS *****

SEQ 0091

4888	017736	000270			SEN		; TEST INSTRUCTION
4889	017740	022737	030010	177776	CMP	#30010, @#177776	; DID IT SET THE NEGATIVE BIT
4890	017746	001401			BEQ	1\$	; YES GO ON
4891	017750	104001			ERROR	+1	; CPU ERROR
4892							; NO GO TO ERROR
4893	017752				1\$:		
4894					:		
4895	017752				TE110:		
4896					:	TEST SET V BIT INST (SEV)	
4897	017752	012737	030000	177776	MOV	#30000, @#177776	; SETUP PSW
4898	017760	000262			SEV		; TEST INSTRUCTION
4899	017762	022737	030002	177776	CMP	#30002, @#177776	; DID IT SET THE OVERFLOW BIT
4900	017770	001401			BEQ	1\$	; YES GO ON
4901	017772	104001			ERROR	+1	; CPU ERROR
4902							; NO GO TO ERROR
4903	017774				1\$:		
4904					:		
4905	017774				TE111:		
4906					:	TEST SET Z BIT INST (SEZ)	
4907	017774	012737	030000	177776	MOV	#30000, @#177776	; SETUP PSW
4908	020002	000264			SEZ		; TEST INSTRUCTION
4909	020004	022737	030004	177776	CMP	#30004, @#177776	; DID IT SET THE ZERO BIT
4910	020012	001401			BEQ	1\$	; YES GO ON
4911	020014	104001			ERROR	+1	; CPU ERROR
4912							; NO GO TO ERROR
4913	020016				1\$:		
4914					:		
4915	020016				TE112:		
4916					:	TEST MULTIPLE CLEARS OF CC BITS	
4917	020016	012737	030000	177776	MOV	#30000, @#177776	; INIT PSW
4918	020024	000277			SCC		; SETUP PSW
4919	020026	000243			.WORD	243	; TEST CLC CLV
4920	020030	022737	030014	177776	CMP	#30014, @#177776	; PSW CORRECT?
4921	020036	001401			BEQ	1\$	; YES GO ON
4922	020040	104001			ERROR	+1	; CPU ERROR
4923							; NO GO TO ERROR
4924	020042	000277			1\$:	SCC	; SETUP PSW
4925	020044	000245			.WORD	245	; TEST CLC CLZ
4926	020046	022737	030012	177776	CMP	#30012, @#177776	; PSW CORRECT?
4927	020054	001401			BEQ	2\$	; YES GO ON
4928	020056	104001			ERROR	+1	; CPU ERROR
4929							; NO GO TO ERROR
4930	020060	000277			2\$:	SCC	; SETUP PSW
4931	020062	000246			.WORD	246	; TEST CLV CLZ
4932	020064	022737	030011	177776	CMP	#30011, @#177776	; PSW CORRECT?
4933	020072	001401			BEQ	3\$	; YES GO ON
4934	020074	104001			ERROR	+1	; CPU ERROR
4935							; NO GO TO ERROR
4936	020076	000277			3\$:	SCC	; SETUP PSW
4937	020100	000247			.WORD	247	; TEST CLC CLV CLZ
4938	020102	022737	030010	177776	CMP	#30010, @#177776	; PSW CORRECT?
4939	020110	001401			BEQ	4\$	; YES GO ON
4940	020112	104001			ERROR	+1	; CPU ERROR
4941							; NO GO TO ERROR
4942	020114	000277			4\$:	SCC	; SETUP PSW
4943	020116	000251			.WORD	251	; TEST CLN CLC

[illegible]

5000	020316	000257		3\$:	CCC			! SETUP PSW
5001	020320	000267			.WORD	267		! TEST SEC SEV SEZ
5002	020322	022737	030007 177776		CMP	030007,00177776		! PSW CORRECT?
5003	020330	001401			BEQ	4\$		! YES GO ON
5004	020332	104001			ERROR	+1		! CPU ERROR
5005								! NO GO TO ERROR
5006	020334	000257		4\$:	CCC			! SETUP PSW
5007	020336	000271			.WORD	271		! TEST SEN SEC
5008	020340	022737	030011 177776		CMP	030011,00177776		! PSW CORRECT?
5009	020346	001401			BEQ	5\$		! YES GO ON
5010	020350	104001			ERROR	+1		! CPU ERROR
5011								! NO GO TO ERROR
5012	020352	000257		5\$:	CCC			! SETUP PSW
5013	020354	000272			.WORD	272		! TEST SEN SEV
5014	020356	022737	030012 177776		CMP	030012,00177776		! PSW CORRECT?
5015	020364	001401			BEQ	6\$		! YES GO ON
5016	020366	104001			ERROR	+1		! CPU ERROR
5017								! NO GO TO ERROR
5018	020370	000257		6\$:	CCC			! SETUP PSW
5019	020372	000273			.WORD	273		! TEST SEN SEC SEV
5020	020374	022737	030013 177776		CMP	030013,00177776		! PSW CORRECT?
5021	020402	001401			BEQ	7\$		! YES GO ON
5022	020404	104001			ERROR	+1		! CPU ERROR
5023								! NO GO TO ERROR
5024	020406	000257		7\$:	CCC			! SETUP PSW
5025	020410	000274			.WORD	274		! TEST SEN SEZ
5026	020412	022737	030014 177776		CMP	030014,00177776		! PSW CORRECT?
5027	020420	001401			BEQ	8\$		! YES GO ON
5028	020422	104001			ERROR	+1		! CPU ERROR
5029								! NO GO TO ERROR
5030	020424	000257		8\$:	CCC			! SETUP PSW
5031	020426	000275			.WORD	275		! TEST SEN SEC SEZ
5032	020430	022737	030015 177776		CMP	030015,00177776		! PSW CORRECT?
5033	020436	001401			BEQ	9\$		! YES GO ON
5034	020440	104001			ERROR	+1		! CPU ERROR
5035								! NO GO TO ERROR
5036	020442	000257		9\$:	CCC			! SETUP PSW
5037	020444	000276			.WORD	276		! TEST SEN SEV SEZ
5038	020446	022737	030016 177776		CMP	030016,00177776		! PSW CORRECT?
5039	020454	001401			BEQ	10\$		! YES GO ON
5040	020456	104001			ERROR	+1		! CPU ERROR
5041								! NO GO TO ERROR
5042	020460			10\$:				
5043								
5044	020460			IF 113A:				
5045								
5046	020460	000257			CCC			! CLEAR ALL CC BITS IN PSW
5047	020462	002001			BGE	1\$		! BGE SHOULD BRANCH
5048	020464	104001			ERROR	+1		! CPU ERROR
5049								! ERROR, DIDN'T BRANCH
5050	020466	003001		1\$:	BGT	2\$		! BGT SHOULD BRANCH
5051	020470	104001			ERROR	+1		! CPU ERROR
5052								! ERROR, DIDN'T BRANCH
5053	020472	003401		2\$:	BLE	3\$		! BLE SHOULD NOT BRANCH
5054	020474	000401			BR	4\$		! BRANCH TO NEXT TEST
5055	020476	104001		3\$:	ERROR	+1		! CPU ERROR

5113	020624	002401		25\$:	BLT	26\$		;ERROR; BLE SHOULD NOT HAVE BRANCHED
5114	020626	000401			BR	27\$		;BLT SHOULD NOT BRANCH
5115	020630	104001		26\$:	ERROR	*1		;BRANCH TO NEXT TEST
5116								;CPU ERROR
5117	020632			27\$:				;ERROR; BLT SHOULD NOT HAVE BRANCHED
5118	020632			TB114:				
5119				:	TEST NOP INST			
5120	020632	012737	030000	177776	MOV	#30000,#177776		;INIT PSW
5121	020640	012700	000001		MOV	#1,R0		;INIT R0
5122	020644	012701	000002		MOV	#2,R1		;INIT R1
5123	020650	012702	000003		MOV	#3,R2		;INIT R2
5124	020654	012703	000004		MOV	#4,R3		;INIT R3
5125	020660	012704	000005		MOV	#5,R4		;INIT R4
5126	020664	012705	000006		MOV	#6,R5		;INIT R5
5127	020670	010637	002762		MOV	R6,#SLOC00		;SAVE SP
5128	020674	012737	030017	110502	MOV	#30017,#EXPDAT		;SETUP PSW
5129	020702	000277			SCC			;SET ALL CONDITION CODE BITS
5130	020704	000240			NOP			; TEST INSTRUCTION
5131	020706	000240			NOP			; TEST INSTRUCTION
5132	020710	000240			NOP			; TEST INSTRUCTION
5133	020712	004767	000046		JSR	PC,T114		;CHECK PSW, AND GPR'S
5134	020716	020637	002762		CMP	R6,#SLOC00		;CHECK SP
5135	020722	001401			BEQ	1\$		;OK GO ON
5136	020724	104001			ERROR	*1		;CPU ERROR
5137								;NO GO TO ERROR
5138	020726	012737	030000	110502	1\$:	MOV	#30000,#EXPDAT	;SETUP PSW
5139	020734	000257			CCC			;CLEAR ALL CONDITION CODE BITS
5140	020736	000260			.WORD	260		; TEST FOR NOP OPERATION
5141	020740	000260			.WORD	260		; TEST FOR NOP OPERATION
5142	020742	000260			.WORD	260		; TEST FOR NOP OPERATION
5143	020744	004767	000014		JSR	PC,T114		;CHECK PSW, AND GPR'S
5144	020750	020637	002762		CMP	R6,#SLOC00		;CHECK SP
5145	020754	001401			BEQ	2\$		;OK GO ON
5146	020756	104001			ERROR	*1		;CPU ERROR
5147								;NO GO TO ERROR
5148	020760			2\$:				
5149	020760	000167	000104		JMP	FINNOP		
5150				:				
5151	020764	023737	110502	177776	T114:	CMP	#EXPDAT,#177776	;CHECK PSW
5152	020772	001405			BEQ	TA114		;OK GO ON
5153	020774	010067	157400		MOV	RO,400		;SAVE RO
5154	021000	104001			ERROR	*1		;CPU ERROR
5155								;NO GO TO ERROR
5156	021002	016700	157372		MOV	400,RO		;RESTORE RO
5157	021006	022700	000001		TA114:	CMP	#1,RO	;CHECK RO
5158	021012	001401			BEQ	TB114		;OK GO ON
5159	021014	104001			ERROR	*1		;CPU ERROR
5160								;NO GO TO ERROR
5161	021016	022701	000002		TB114:	CMP	#2,R1	;CHECK R1
5162	021022	001401			BEQ	TC114		;OK GO ON
5163	021024	104001			ERROR	*1		;CPU ERROR
5164								;NO GO TO ERROR
5165	021026	022702	000003		TC114:	CMP	#3,R2	;CHECK R2
5166	021032	001401			BEQ	TD114		;OK GO ON
5167	021034	104001			ERROR	*1		;CPU ERROR

```

5168
5169 021036 022703 000004 TD114: CMP    #4,R3      ;NO GO TO ERROR
5170 021042 001401          BEQ    TF114      ;CHECK R3
5171 021044 104001          ERROR   +1        ;OK GO ON
5172                                     ;CPU ERROR
5173 021046 022704 000005 TF114: CMP    #5,R4      ;NO GO TO ERROR
5174 021052 001401          BEQ    TG114      ;CHECK R4
5175 021054 104001          ERROR   +1        ;OK GO ON
5176                                     ;CPU ERROR
5177 021056 022705 000006 TG114: CMP    #6,R5      ;NO GO TO ERROR
5178 021062 001401          BEQ    TH114      ;CHECK R5
5179 021064 104001          ERROR   +1        ;OK GO ON
5180                                     ;CPU ERROR
5181 021066 000207 TH114: RTS     PC          ;NO GO TO ERROR
5182                                     ;RETURN
5183 021070 000240 FINNOP: NOP
5184
5185
5186
5187 ;-----
5188 ;TEST ALTERNATE REGISTER SET
5189
5189 021072 005000          CLR     R0          ;-----CLEAR-----
5190 021074 005001          CLR     R1          ;-----PRIMARY-----
5191 021076 005002          CLR     R2          ;-----GENERAL-----
5192 021100 005003          CLR     R3          ;-----PURPOSE-----
5193 021102 005004          CLR     R4          ;-----REGISTER-----
5194 021104 005005          CLR     R5          ;-----SET-----
5195 021106 012767 004000 156662 MOV    #4000,PS ;SELECT ALTERNATE REGISTER SET
5196 021114 032767 004000 156654 BIT    #BIT11,PS ;TEST TO SEE THAT BIT IS SET IN PS
5197 021122 001001          BNE     1$        ;IF IT'S SET GO TESTS REGISTERS
5198 021124 104001          ERROR   +1        ;CPU ERROR
5199                                     ;ERROR! ALTERNATE REGISTER SELECT
5200                                     ;BIT IN PS IS NOT SET
5201 021126 012700 177777 1$: MOV    #177777,R0 ;WRITE ALL ONES TO ALTERNATE REG SET
5202 021132 010001          MOV    R0,R1
5203 021134 010102          MOV    R1,R2
5204 021136 010203          MOV    R2,R3
5205 021140 010304          MOV    R3,R4
5206 021142 010405          MOV    R4,R5
5207 021144 042767 004000 156624 BIC    #BIT11,PS ;CLEAR BIT 11 IN PS
5208 021152 032767 004000 156616 BIT    #BIT11,PS ;IS BIT 11 = 0?
5209 021160 001401          BEQ     2$        ;IF IT'S NOT ZERO
5210 021162 104001          ERROR   +1        ;CPU ERROR
5211                                     ;ERROR! BIT 11 DID NOT CLEAR
5212 021164 005700          2$: TST     R0      ;MAKE SURE THAT WE REALLY
5213 021166 001401          BEQ     3$        ;WERE TALKING TO ALTERNATE REGISTERS.
5214 021170 104001          ERROR   +1        ;CPU ERROR
5215                                     ;ERROR!
5216 021172 005701          3$: TST     R1
5217 021174 001401          BEQ     4$
5218 021176 104001          ERROR   +1        ;CPU ERROR
5219                                     ;ERROR!
5220 021200 005702          4$: TST     R2
5221 021202 001401          BEQ     5$
5222 021204 104001          ERROR   +1        ;CPU ERROR
5223                                     ;ERROR!

```

(68)

COKDABO KDJ11 B CLUSTER MACY11 30(1046) 05 APR 84 16:45 PAGE 98
 COKDAB.P11 05-APR-84 16:45

***** DOUBLE OPERAND TESTS *****

SEQ 0097

```

5224 021206 005703      5$:  TST      R3      ;
5225 021210 001401      BEQ      6$      ;
5226 021212 104001      ERROR    +1      ;CPU ERROR
5227                                ;
5228 021214 005704      6$:  TST      R4      ;
5229 021216 001401      BEQ      7$      ;
5230 021220 104001      ERROR    +1      ;CPU ERROR
5231                                ;
5232 021222 005705      7$:  TST      R5      ;
5233 021224 001401      BEQ      8$      ;
5234 021226 104001      ERROR    +1      ;CPU ERROR
5235                                ;ERROR!
5236 021230 012767 004000 156540 8$:  MOV      #BIT11,PS      ;ENABLE ALTERNATE REGISTER SET
5237 021236 005000      CLR      R0      ;-----CLEAR-----
5238 021240 005001      CLR      R1      ; - ALTERNATE -
5239 021242 005002      CLR      R2      ; ---REGISTER---
5240 021244 005003      CLR      R3      ;-----SET-----
5241 021246 005004      CLR      R4      ;
5242 021250 005005      CLR      R5      ;
5243 021252 042767 004000 156516 BIC      #BIT11,PS      ;BACK TO PRIMARY GPRS
5244 021260 012700 177777      MOV      #177777,R0      ;ENSURE THAT WRITES TO PRIMARY GPRS
5245 021264 010001      MOV      R0,R1      ;DO NOT EFFECT ALTERNATE GPRS
5246 021266 010102      MOV      R1,R2      ;
5247 021270 010203      MOV      R2,R3      ;
5248 021272 010304      MOV      R3,R4      ;
5249 021274 010405      MOV      R4,R5      ;
5250 021276 052767 004000 156472 BIS      #BIT11,PS      ;RETURN TO ALTERNATE REGISTERS
5251 021304 005700      TST      R0      ;SHOULD BE ALL 0 S
5252 021306 001401      BEQ      9$      ;
5253 021310 104001      ERROR    +1      ;CPU ERROR
5254                                ;ERROR!
5255 021312 005701      9$:  TST      R1      ;
5256 021314 001401      BEQ      10$     ;
5257 021316 104001      ERROR    +1      ;CPU ERROR
5258                                ;
5259 021320 005702      10$: TST      R2      ;
5260 021322 001401      BEQ      11$     ;
5261 021324 104001      ERROR    +1      ;CPU ERROR
5262                                ;
5263 021326 005703      11$: TST      R3      ;
5264 021330 001401      BEQ      12$     ;
5265 021332 104001      ERROR    +1      ;CPU ERROR
5266                                ;
5267 021334 005704      12$: TST      R4      ;
5268 021336 001401      BEQ      13$     ;
5269 021340 104001      ERROR    +1      ;CPU ERROR
5270                                ;
5271 021342 005705      13$: TST      R5      ;
5272 021344 001401      BEQ      14$     ;
5273 021346 104001      ERROR    +1      ;CPU ERROR
5274 021350      14$:
5275
5276 021350      ALROTS:
5277 ;
5278 021350 012700 177777      MOV      #177777,R0      ;RO=177777
5279 021354 020027 177777      CMP      R0,#177777      ;DOES RO=177777

```

COKDABO KDJ11-B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 99
 COKDAB.P11 05-APR-84 16:45 ***** DOUBLE OPERAND TESTS *****

SEQ 0098

```

5280 021360 001401      BEQ      1$      ;YES GO ON
5281 021362 104001      ERROR     +1      ;CPU ERROR
5282                                     ;NO GO TO ERROR
5283 021364 005000      1$:      CLR      R0      ;R0=0
5284 021366 020027 000000      CMP      R0,#0      ;DOES R0=0
5285 021372 001401      BEQ      2$      ;YES GO ON
5286 021374 104001      ERROR     +1      ;CPU ERROR
5287                                     ;NO GO TO ERROR
5288 021376 012700 125252      2$:      MOV      #125252,R0      ;R0=125252
5289 021402 020027 125252      CMP      R0,#125252      ;DOES R0=125252
5290 021406 001401      BEQ      3$      ;YES GO ON
5291 021410 104001      ERROR     +1      ;CPU ERROR
5292                                     ;NO GO TO ERROR
5293 021412 012700 052525      3$:      MOV      #52525,R0      ;R0=52525
5294 021416 020027 052525      CMP      R0,#52525      ;DOES R0=52525
5295 021422 001401      BEQ      4$      ;YES GO ON
5296 021424 104001      ERROR     +1      ;CPU ERROR
5297                                     ;NO GO TO ERROR
5298 021426      4$:
5299      ;
5300 021426      ALR1TS:
5301      ; ALTERNATE REGISTER SET R1 BIT TESTS
5302 021426 012701 177777      MOV      #177777,R1      ;R1=177777
5303 021432 020127 177777      CMP      R1,#177777      ;DOES R1=177777
5304 021436 001401      BEQ      1$      ;YES GO ON
5305 021440 104001      ERROR     +1      ;CPU ERROR
5306                                     ;NO GO TO ERROR
5307 021442 005001      1$:      CLR      R1      ;R1=0
5308 021444 020127 000000      CMP      R1,#0      ;DOES R1=0
5309 021450 001401      BEQ      2$      ;YES GO ON
5310 021452 104001      ERROR     +1      ;CPU ERROR
5311                                     ;NO GO TO ERROR
5312 021454 012701 125252      2$:      MOV      #125252,R1      ;R1=125252
5313 021460 020127 125252      CMP      R1,#125252      ;DOES R1=125252
5314 021464 001401      BEQ      3$      ;YES GO ON
5315 021466 104001      ERROR     +1      ;CPU ERROR
5316                                     ;NO GO TO ERROR
5317 021470 012701 052525      3$:      MOV      #52525,R1      ;R1=52525
5318 021474 020127 052525      CMP      R1,#52525      ;DOES R1=52525
5319 021500 001401      BEQ      4$      ;YES GO ON
5320 021502 104001      ERROR     +1      ;CPU ERROR
5321                                     ;NO GO TO ERROR
5322 021504      4$:
5323      ;
5324 021504      ALR2TS:
5325      ; ALTERNATE REGISTER SET R2 BIT TESTS
5326 021504 012702 177777      MOV      #177777,R2      ;R2=177777
5327 021510 020227 177777      CMP      R2,#177777      ;DOES R2=177777
5328 021514 001401      BEQ      1$      ;YES GO ON
5329 021516 104001      ERROR     +1      ;CPU ERROR
5330                                     ;NO GO TO ERROR
5331 021520 005002      1$:      CLR      R2      ;R2=0
5332 021522 020227 000000      CMP      R2,#0      ;DOES R2=0
5333 021526 001401      BEQ      2$      ;YES GO ON
5334 021530 104001      ERROR     +1      ;CPU ERROR
5335                                     ;NO GO TO ERROR

```


CO:KABO R0J11 B CLUSTER MACY11 30(1046) 05 APR 84 16:45 PAGE 100
 CO:KAB.P11 05-APR 84 16:45

***** DOUBLE OPERAND TESTS *****

SEQ 0099

5336	021532	012702	125252	2\$:	MOV	#125252,R2		;R2=125252
5337	021536	020227	125252		CMP	R2,#125252		;DOES R2=125252
5338	021542	001401			BEQ	3\$		;YES GO ON
5339	021544	104001			ERROR	+1		;CPU ERROR
5340								;NO GO TO ERROR
5341	021546	012702	052525	3\$:	MOV	#52525,R2		;R2=52525
5342	021552	020227	052525		CMP	R2,#52525		;DOES R2=52525
5343	021556	001401			BEQ	4\$		;YES GO ON
5344	021560	104001			ERROR	+1		;CPU ERROR
5345								;NO GO TO ERROR
5346	021562			4\$:				
5347								
5348	021562							
5349								
5350	021562	012703	177777					
5351	021566	020327	177777					
5352	021572	001401						
5353	021574	104001						
5354								
5355	021576	005003		1\$:	CLR	R3		;R3=0
5356	021600	020327	000000		CMP	R3,#0		;DOES R3=0
5357	021604	001401			BEQ	2\$		;YES GO ON
5358	021606	104001			ERROR	+1		;CPU ERROR
5359								;NO GO TO ERROR
5360	021610	012703	125252	2\$:	MOV	#125252,R3		;R3=125252
5361	021614	020327	125252		CMP	R3,#125252		;DOES R3=125252
5362	021620	001401			BEQ	3\$		;YES GO ON
5363	021622	104001			ERROR	+1		;CPU ERROR
5364								;NO GO TO ERROR
5365	021624	012703	052525	3\$:	MOV	#52525,R3		;R3=52525
5366	021630	020327	052525		CMP	R3,#52525		;DOES R3=52525
5367	021634	001401			BEQ	4\$		;YES GO ON
5368	021636	104001			ERROR	+1		;CPU ERROR
5369								;NO GO TO ERROR
5370	021640			4\$:				
5371								
5372	021640							
5373								
5374	021640	012704	177777					
5375	021644	020427	177777					
5376	021650	001401						
5377	021652	104001						
5378								
5379	021654	005004		1\$:	CLR	R4		;R4=0
5380	021656	020427	000000		CMP	R4,#0		;DOES R4=0
5381	021662	001401			BEQ	2\$		;YES GO ON
5382	021664	104001			ERROR	+1		;CPU ERROR
5383								;NO GO TO ERROR
5384	021666	012704	125252	2\$:	MOV	#125252,R4		;R4=125252
5385	021672	020427	125252		CMP	R4,#125252		;DOES R4=125252
5386	021676	001401			BEQ	3\$		;YES GO ON
5387	021700	104001			ERROR	+1		;CPU ERROR
5388								;NO GO TO ERROR
5389	021702	012704	052525	3\$:	MOV	#52525,R4		;R4=52525
5390	021706	020427	052525		CMP	R4,#52525		;DOES R4=52525
5391	021712	001401			BEQ	4\$		;YES GO ON

```

5392 021714 104001          ERROR +1          ;CPU ERROR
5393                                     ;NO GO TO ERROR
5394 021716          4$:
5395          ;
5396 021716          ALR5TS:
5397          ;
5398 021716 012705 177777          MOV #177777,R5          ;R5=177777
5399 021722 020527 177777          CMP R5,#177777          ;DOES R5=177777
5400 021726 001401          BEQ 1$          ;YES GO ON
5401 021730 104001          ERROR +1          ;CPU ERROR
5402                                     ;NO GO TO ERROR
5403 021732 005005          1$: CLR R5          ;R5=0
5404 021734 020527 000000          CMP R5,#0          ;DOES R5=0
5405 021740 001401          BEQ 2$          ;YES GO ON
5406 021742 104001          ERROR +1          ;CPU ERROR
5407                                     ;NO GO TO ERROR
5408 021744 012705 125252          2$: MOV #125252,R5          ;R5=125252
5409 021750 020527 125252          CMP R5,#125252          ;DOES R5=125252
5410 021754 001401          BEQ 3$          ;YES GO ON
5411 021756 104001          ERROR +1          ;CPU ERROR
5412                                     ;NO GO TO ERROR
5413 021760 012705 052525          3$: MOV #52525,R5          ;R5=52525
5414 021764 020527 052525          CMP R5,#52525          ;DOES R5=52525
5415 021770 001401          BEQ 4$          ;YES GO ON
5416 021772 104001          ERROR +1          ;CPU ERROR
5417                                     ;NO GO TO ERROR
5418 021774 042767 004000 155774 4$: BIC #BIT11,PS          ;RETURN TO PRIMARY GEN PURPOSE REGS
5419
5420          ;
5421 022002          TE115:
5422          ;
5423 022002 005004          ; TEST MOVE FROM PROCCESOR STATUS INST (MFPS)
5424          CLR R4          ;SETUP DESTINATION R4
5425 022004 012701 022212          MOV #TE115A,R1          ;SETUP POINTERS TO TABLES
5426 022010 012702 022222          MOV #TE115B,R2          ;
5427 022014 012703 022230          MOV #TE115C,R3          ;
5428 022020 012137 177776          1$: MOV (R1),@#177776          ;SETUP PSW
5429 022024 106704          MFPS R4          ; TEST INSTRUCTION
5430 022026 023722 177776          CMP @#177776,(R2)          ;CHECK PSW
5431 022032 001401          BEQ 2$          ;OK GO ON
5432 022034 104001          ERROR +1          ;CPU ERROR
5433                                     ;NO GO TO ERROR
5434 022036 020423          2$: CMP R4,(R3)          ;CHECK R4
5435 022040 001401          BEQ 3$          ;OK GO ON
5436 022042 104001          ERROR +1          ;CPU ERROR
5437                                     ;NO GO TO ERROR
5438 022044 021137 177777          3$: CMP (R1),#177777          ;ARE WE DONE
5439 022050 001363          BNE 1$          ;NO GO TO 1$
5440
5441          ;
5442 022052 012701 022212          MOV #TE115A,R1          ;SETUP POINTERS TO TABLES
5443 022056 012702 022222          MOV #TE115B,R2          ;
5444 022062 012703 022230          MOV #TE115C,R3          ;
5445 022066 010605          MOV R6,R5          ;SAVE STACK IN R5
5446 022070 011137 177776          101$: MOV (R1),@#177776          ;SETUP PSW
5447 022074 106706          MFPS R6          ; TEST INSTRUCTION

```


K8

CORDBO RDU11 B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 102
 CORDAB.P11 05-APR-84 16:45 ***** DOUBLE OPERAND TESTS *****

SEQ 0101

```

5448 022076 023712 177776      CMP      @#177776,(R2)      ;CHECK PSW
5449 022102 001401      BEQ      102$      ;OK GO ON
5450 022104 104001      ERROR    +1      ;CPU ERROR
5451      ;NO GO TO ERROR
5452 022106 020613      102$:  CMP      R6,(R3)      ;CHECK R6
5453 022110 001401      BEQ      103$      ;OK GO ON
5454 022112 104001      ERROR    +1      ;CPU ERROR
5455      ;NO GO TO ERROR
5456 022114 010506      103$:  MOV      R5,R6      ;RESTORE STACK
5457
5458
5459 022116 012701 022212      MOV      @TE115A,R1      ;SETUP POINTERS TO TABLES
5460 022122 012702 022222      MOV      @TE115B,R2      ;
5461 022126 012703 022236      MOV      @TE115D,R3      ;
5462 022132 005037 110502      CLR      @#EXPDAT      ;INIT EXPECTED DATA HOLDER.
5463 022136 012704 110502      4$:  MOV      @EXPDAT,R4      ;SETUP POINTER TO TEST LOCATION
5464 022142 012137 177776      MOV      (R1)+,@#177776      ;SETUP PSW
5465 022146 106724      MFPS      (R4)+      ;TEST INSTRUCTION
5466 022150 023722 177776      CMP      @#177776,(R2)+      ;CHECK PSW
5467 022154 001401      BEQ      5$      ;OK GO ON
5468 022156 104001      ERROR    +1      ;CPU ERROR
5469      ;NO GO TO ERROR
5470 022160 020427 110503      5$:  CMP      R4,@EXPDAT+1      ;CHECK R4
5471 022164 001401      BEQ      6$      ;OK GO ON
5472 022166 104001      ERROR    +1      ;CPU ERROR
5473      ;NO GO TO ERROR
5474 022170 023723 110502      6$:  CMP      @#EXPDAT,(R3)+      ;CHECK TEST LOCATION
5475 022174 001401      BEQ      7$      ;OK GO ON
5476 022176 104001      ERROR    +1      ;CPU ERROR
5477      ;NO GO TO ERROR
5478 022200 021127 177777      7$:  CMP      (R1),@#177777      ;ARE WE DONE
5479 022204 001354      BNE      4$      ;NO GO TO 4$
5480
5481 022206 000167 000032      JMP      TE115F
5482
5483      ;
5484      ;
5485 022212 030207      TE115A: .WORD    30207
5486 022214 030000      .WORD    30000
5487 022216 030057      .WORD    30057
5488 022220 177777      .WORD    177777
5489 022222 030211      TE115B: .WORD    30211
5490 022224 030004      .WORD    30004
5491 022226 030041      .WORD    30041
5492 022230 177607      TE115C: .WORD    177607
5493 022232 000000      .WORD    0
5494 022234 000057      .WORD    57
5495 022236 000207      TE115D: .WORD    207
5496 022240 000000      .WORD    0
5497 022242 000057      .WORD    57
5498 022244 000240      TE115F: NOP
5499      ;
5500 022246      TE116:
5501      ;
5502      ;
5503 022246 012737 030000 177776      MOV      #30000,@#177776      ;SET PSW TO KERNEL MODE

```

CORRADO KDJ11 B CLUSTER MACY11 30(1046) 05 APR 84 16:45 PAGE 103
 CORRADO.P11 05-APR-84 16:45

***** DOUBLE OPERAND TESTS *****

SEQ 0102

```

5504 022254 012701 022550      MOV      #TE116D,R1      ;SETUP POINTERS TO TABLES
5505 022260 012702 022526      MOV      #TE116B,R2      ;
5506 022264 010103              MOV      R1,R3          ;
5507 022266 004767 000136      JSR      PC,T116         ; TEST INSTRUCTION AND CHECK PSW
5508                                ;AND SOURCE OPERAND
5509
5510
5511 022272 012737 140000 177776  MOV      #140000,#177776      ;SET PSW TO USER MODE
5512 022300 012706 000600      MOV      #600,R6          ;SETUP USER STACK
5513 022304 012701 022550      MOV      #TE116D,R1      ;SETUP POINTERS TO TABLES
5514 022310 012702 022540      MOV      #TE116C,R2      ;
5515 022314 010103              MOV      R1,R3          ;
5516 022316 004767 000106      JSR      PC,T116         ; TEST INSTRUCTION AND CHECK PSW
5517                                ;AND SOURCE OPERAND
5518
5519
5520 022322 012737 140000 177776  MOV      #140000,#177776      ;SET PSW TO USER MODE AND CLEAR CC BITS
5521 022330 012701 022522      MOV      #TE116A,R1      ;SETUP POINTERS TO TABLES
5522 022334 012702 022540      MOV      #TE116C,R2      ;
5523 022340 012703 022550      MOV      #TE116D,R3      ;
5524 022344 010104              MOV      R1,R4          ;SAVE A COPY OF R1 INTO R4
5525 022346 004767 000110      JSR      PC,T116         ; TEST INSTRUCTION AND CHECK PSW
5526                                ;AND SOURCE OPERAND
5527
5528
5529 022352 012737 030000 177776  MOV      #30000,#177776      ;SET PSW TO KERNEL MODE
5530 022360 012701 022522      MOV      #TE116A,R1      ;SETUP POINTERS TO TABLES
5531 022364 012702 022526      MOV      #TE116B,R2      ;
5532 022370 012703 022550      MOV      #TE116D,R3      ;
5533 022374 010104              MOV      R1,R4          ;SAVE A COPY OF R1 INTO R4
5534 022376 004767 000060      JSR      PC,T116         ; TEST INSTRUCTION AND CHECK PSW
5535                                ;AND SOURCE OPERAND
5536
5537 022402 005037 177776          CLR      #177776          ;SET PSW TO KERNEL MODE
5538 022406 106427 177412          MTPS     #177412          ;TEST INSTRUCTION
5539 022412 022737 000012 177776  CMP      #12,#177776      ;IS PSW CORRECT
5540 022420 001401              BEQ      100$          ;YES GO ON
5541 022422 104001              ERROR    +1            ;CPU ERROR
5542                                ;NO GO TO ERROR
5543
5544 022424 000167 000130          JMP      FIN116
5545
5546 022430 012105              ;
5547 022432 106405              T116: MOV      (R1)+,R5          ;MOVE TEST DATA TO R5
5548 022434 023722 177776      MTPS     R5              ; TEST INSTRUCTION
5549 022440 001401              CMP      #177776,(R2)+      ;IS PSW CORRECT
5550 022442 104001              BEQ      1$              ;YES GO ON
5551                                ;CPU ERROR
5552                                ;NO GO TO ERROR
5553 022444 022305              1$: CMP      (R3)+,R5          ;IS R5 CORRECT
5554 022446 001401              BEQ      2$              ;YES GO ON
5555 022450 104001              ERROR    +1            ;CPU ERROR
5556                                ;NO GO TO ERROR
5557 022452 021227 177777      2$: CMP      (R2),#177777      ;ARE WE DONE
5558 022456 001364              BNE     T116          ;NO GO TO T116
5559 022460 000207              RTS      PC              ;RETURN

```

COKDABO KDJ11 B CLUSTER MACY11 30(1046) 05 APR-84 16:45 PAGE 104
 COKDAB.P11 05-APR-84 16:45 ***** DOUBLE OPERAND TESTS *****

SEQ 0103

5560	022462	106421		TA116:	MTPS	(R1)+		; TEST INSTRUCTION
5561	022464	023722	177776		CMP	#177775,(R2)+		; IS PSW CORRECT
5562	022470	001401			BEQ	1\$		; YES GO ON
5563	022472	104001			ERROR	+1		; CPU ERROR
5564								; NO GO TO ERROR
5565	022474	122423		1\$:	CMPB	(R4)+,(R3)+		; CHECK TEST LOCATION
5566	022476	001401			BEQ	2\$		; OR GO ON
5567	022500	104001			ERROR	+1		; CPU ERROR
5568								; NO GO TO ERROR
5569	022502	020104		2\$:	CMP	R1,R4		; IS SOURCE OPERAND CORRECT
5570	022504	001401			BEQ	3\$		; YES GO ON
5571	022506	104001			ERROR	+1		; CPU ERROR
5572								; NO GO TO ERROR
5573	022510	005203		3\$:	INC	R3		; POINT TO NEXT WORD
5574	022512	021227	177777		CMP	(R2),#177777		; ARE WE DONE
5575	022516	001361			BNE	TA116		; NO GO TO TA116
5576	022520	000207			RTS	PC		; RETURN
5577								
5578	022522	377		TE116A:	.BYTE	377		
5579	022523	000			.BYTE	0		
5580	022524	252			.BYTE	252		
5581	022525	125			.BYTE	125		
5582	022526	030357		TE116B:	.WORD	30357		
5583	022530	030000			.WORD	30000		
5584	022532	030252			.WORD	30252		
5585	022534	030105			.WORD	30105		
5586	022536	177777			.WORD	177777		
5587	022540	140017		TE116C:	.WORD	140017		
5588	022542	140000			.WORD	140000		
5589	022544	140012			.WORD	140012		
5590	022546	140005			.WORD	140005		
5591	022550	177777		TE116D:	.WORD	177777		
5592	022552	177400			.WORD	177400		
5593	022554	177652			.WORD	177652		
5594	022556	177525			.WORD	177525		
5595								
5596	022560			FIN116:				
5597								
5598	022560			TE117:				
5599								
5600	022560	013746	000010		TEST MFPT (MOVE FROM PROCESSOR TYPE)			
5601	022564	012737	022672	000010	MOV	#10,-(SP)		; SAVE VECTOR
5602	022572	012700	177777		MOV	#TE117A,#10		; SETUP VECTOR TO HANDLE POSSIBLE ILLEGAL INST TR
5603	022576	012737	030000	177776	MOV	#177777,R0		; INIT R0
5604	022604	000007			MOV	#30000,#177776		; SETUP PSW
5605	022606	022737	030000	177776	.WORD	7		; TEST INSTRUCTION
5606	022614	001401			CMP	#30000,#177776		; IS PSW CORRECT
5607	022616	104001			BEQ	1\$		; YES GO ON
5608					ERROR	+1		; CPU ERROR
5609	022620	020027	000005					; NO GO TO ERROR
5610	022624	001401		1\$:	CMP	R0,#5		; IS R0 CORRECT
5611	022626	104001			BEQ	2\$		; YES GO ON
5612					ERROR	+1		; CPU ERROR
5613	022630	012700	177777					; NO GO TO ERROR
5614	022634	000277		2\$:	MOV	#177777,R0		; INIT R0
5615	022636	000007			SCC			; SET ALL CC BITS
					.WORD	7		; TEST INSTRUCTION

COKDABO KDJ11-B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 105
COKDAB.P11 05-APR-84 16:45 ***** DOUBLE OPERAND TESTS *****

SEQ 0104

5616	022640	022737	030017	177776		CMP	#30017,#177776	;IS PSW CORRECT
5617	022646	001401				BEQ	3\$	;YES GO ON
5618	022650	104001				ERROR	+1	;CPU ERROR
5619								;NO GO TO ERROR
5620	022652	020027	000005		3\$:	CMP	R0,#5	;IS R0 CORRECT
5621	022656	001401				BEQ	4\$	;YES GO ON
5622	022660	104001				ERROR	+1	;CPU ERROR
5623								;NO GO TO ERROR
5624	022662	012637	000010		4\$:	MOV	(SP)+,#10	;RESTORE VECTOR
5625								
5626	022666	000167	000002			JMP	FIN117	
5627					:			
5628	022672	104001			TE117A:	ERROR	+1	;CPU ERROR
5629								;GO TO ERROR IF TRAP TAKES PLACE
5630					:			
5631	022674	000240			FIN117:	NOP		
5632					:			
5633	022676				TE120:			
5634					:			
5635	022676	005037	177766			TEST HALT (NOT KERNEL MODE)		
5636	022702	005037	177776			CLR	#177766	;INIT CPU ERROR REG
5637	022706	013746	000004			CLR	#177776	;INIT PSW-SET KERNEL MODE
5638	022712	013746	000006			MOV	#4,-(SP)	;SAVE VECTOR
5639	022716	012737	022752	000004		MOV	#6,-(SP)	;SAVE VECTOR
5640	022724	005037	000006			MOV	TE120A,#4	;SET UP VECTOR TO HANDLE ILLEGAL HALT
5641	022730	012767	140000	155040		CLR	#6	;SET UP VECTOR TO COME BACK IN KERNL MODE
5642	022736	012706	000600			MOV	#140000,PS	;SET IN USER MODE
5643	022742	000000				MOV	#600,R6	;INITIALIZE THE USER STACK POINTER
5644	022744	104001				HALT		; TEST INSTRUCTION
5645					PROCNT:	ERROR	+1	;CPU ERROR
5646	022746	000167	000044			JMP	FIN120	;IF NOTHING HAPPENED GO TO ERROR
5647					:			
5648					:			
5649	022752	022737	030000	177776	TE120A:	CMP	#30000,#177776	;IS PSW CORRECT/PREVIOUS MODE = USER?
5650	022760	001401				BEQ	1\$	;YES GO ON
5651	022762	104001				ERROR	+1	;CPU ERROR
5652								;NO GO TO ERROR
5653	022764	022737	000200	177766	1\$:	CMP	#200,#177766	; TEST CPU ERROR REGISTER
5654	022772	001401				BEQ	2\$	;YES GO ON
5655	022774	104001				ERROR	+1	;CPU ERROR
5656								;NO GO TO ERROR
5657	022776	022627	022744		2\$:	CMP	(SP)+,#PROCNT	;DOES STACK CONTAIN CORRECT PC
5658	023002	001401				BEQ	3\$	;YES GO ON
5659	023004	104001				ERROR	+1	;CPU ERROR
5660								;NO GO TO ERROR
5661	023006	022627	140000		3\$:	CMP	(SP)+,#140000	;DOES STACK CONTAIN CORRECT PSW
5662	023012	001401				BEQ	FIN120	;YES GO ON
5663	023014	104001				ERROR	+1	;CPU ERROR
5664								;NO GO TO ERROR
5665	023016	012637	000006		FIN120:	MOV	(SP)+,#6	;RESTORE VECTOR
5666	023022	012637	000004			MOV	(SP)-,#4	;RESTORE VECTOR
5667					:			
5668					TE121:			
5669	023026				:			
5670					:	TEST RESET		
5671	023026	122767	000001	156164		CMPB	#APTENV,\$ENV	;ARE WE IN APT MODE?

CORDARO RDJ11 B CLUSTER MACY11 30(1046)
CORDAR,P11 05-APR-84 16:45

05 APR 84 16:45 PAGE 106
***** DOUBLE OPERAND TESTS *****

SEQ 0105

```
5672 023034 001002      BNE      1$
5673 023036 000167 000622      JMP      FIN122
5674
5675 023042 012737 030340 177776 1$:      MOV      030340,00177776      ;IF NOT: DO THIS TEST
5676 023050 012737 160000 177572      MOV      0160000,00177572      ;ELSE SKIP THIS TEST BECAUSE RESETS
5677 023056 012737 000077 172516      MOV      077,00172516      ;SCREW UP THE APT MONITOR.
5678 023064 005037 177772      CLR      00177772      ;SETUP PSW TO KERNEL MODE
5679 023070 023727 177772 000000      CMP      00177772,00      ;SETUP MMRO
5680 023076 001401      BEQ      C121A      ;SETUP MMR3
5681 023100 104001      ERROR    +1      ;CLEAR PIRQ
5682      ;IS PIRQ CORRECT
5683 023102 012737 025000 177772 C121A: MOV      025000,00177772      ;YES GO ON
5684 023110 022737 025252 177772      CMP      025252,00177772      ;CPU ERROR
5685 023116 001401      BEQ      C121B      ;NO GO TO ERROR
5686 023120 104001      ERROR    +1      ;MOVE AN ALTERNATING PATTERN TO PIRQ
5687
5688 023122 012737 077000 177772 C121B: MOV      077000,00177772      ;IS PIRQ CORRECT
5689 023130 022737 077314 177772      CMP      077314,00177772      ;YES GO ON
5690 023136 001401      BEQ      C121C      ;CPU ERROR
5691 023140 104001      ERROR    +1      ;NO GO TO ERROR
5692
5693 023142 000277      SCC      C121C:      ;SET ALL CC BITS
5694 023144 000005      RESET      ;TEST INSTRUCTION
5695 023146 022737 030357 177776      CMP      030357,00177776      ;IS PSW CORRECT
5696 023154 001401      BEQ      1$      ;YES GO ON
5697 023156 104001      ERROR    +1      ;CPU ERROR
5698      ;NO GO TO ERROR
5699 023160 013701 177572 1$:      MOV      005R0,R1      ;SAVE SRO IN R1.
5700 023164 042701 000176      BIC      0176,R1      ;STRIP OFF UNDEFINED BITS 1-6 FROM MMRO
5701 023170 022701 000000      CMP      00,R1      ;IS MMRO CORRECT
5702 023174 001401      BEQ      2$      ;YES GO ON
5703 023176 104001      ERROR    +1      ;CPU ERROR
5704      ;NO GO TO ERROR
5705 023200 022737 000000 172516 2$:      CMP      00,00172516      ;IS MMR3 CORRECT
5706 023206 001401      BEQ      3$      ;YES GO ON
5707 023210 104001      ERROR    +1      ;CPU ERROR
5708      ;NO GO TO ERROR
5709 023212 022737 000000 177772 3$:      CMP      00,00177772      ;IS PIRQ CORRECT
5710 023220 001401      BEQ      4$      ;YES GO ON
5711 023222 104001      ERROR    +1      ;CPU ERROR
5712      ;NO GO TO ERROR
5713
5714 023224 013702 000004 4$:      MOV      004,R2      ;SAVE LOC 4 IN R2
5715 023230 013703 000006      MOV      006,R3      ;SAVE LOC 6 IN R3
5716 023234 012737 023372 000004      MOV      081,004      ;SETUP VECTORS IN CASE AN ERROR CAUSE
5717      ;A HALT TO OCCUR IN USER MODE.
5718 023242 012737 000340 000006      MOV      0340,006      ;THIS SETS KERNEL MODE, AND PREVENTS PIRQ
5719      ;INTERRUPTS FROM TRASHING THE STACK IF A
5720      ;USER MODE HALT OCCURS.
5721 023250 012737 140340 177776      MOV      0140340,00177776      ;SETUP PSW TO USER MODE
5722 023256 012737 160000 177572      MOV      0160000,00177572      ;SETUP MMRO
5723 023264 012737 000077 172516      MOV      077,00172516      ;SETUP MMR3
5724 023272 012737 077000 177772      MOV      077000,00177772      ;SETUP PIRQ
5725 023300 000277      SCC      ;SET ALL CC BITS
5726 023302 000005      RESET      ;TEST INSTRUCTION
5727 023304 022737 140357 177776      CMP      0140357,00177776      ;IS PSW CORRECT
```

```

5728 023312 001402      BEQ      5$      ;YES GO ON
5729 023314 000000      HALT      ;USER MODE HALT; WILL TRAP TO LOC 4
5730 023316 104001      ERROR    +1      ;CPU ERROR
5731                                ;NO GO TO ERROR
5732 023320 013701 177572 5$:      MOV      @0177572,R1      ;SAVE MMRO IN R1
5733 023324 042701 000176      BIC      @176,R1      ;CLEAR BITS 1-6 FROM MMRO
5734 023330 022701 160000      CMP      @160000,R1      ;IS MMRO CORRECT
5735 023334 001402      BEQ      6$      ;YES GO ON
5736 023336 000000      HALT      ;USER MODE HALT; WILL TRAP TO LOC 4
5737 023340 104001      ERROR    +1      ;CPU ERROR
5738                                ;NO GO TO ERROR
5739 023342 022737 000077 172516 6$:      CMP      @77,@0172516      ;IS MMRO CORRECT
5740 023350 001402      BEQ      7$      ;YES GO ON
5741 023352 000000      HALT      ;USER MODE HALT; WILL TRAP TO LOC 4
5742 023354 104001      ERROR    +1      ;CPU ERROR
5743                                ;NO GO TO ERROR
5744 023356 022737 077314 177772 7$:      CMP      @77314,@0177772      ;IS PIRQ CORRECT
5745 023364 001403      BEQ      9$      ;YES GO ON
5746 023366 000000      HALT      ;USER MODE HALT; WILL TRAP TO LOC 4
5747 023370 104001      ERROR    +1      ;CPU ERROR
5748                                ;NO GO TO ERROR
5749 023372 000002      RTI      ;USER MODE HALT OCCURRED; GO TO ERROR.
5750
5751 023374 005037 177772 9$:      CLR      @0177772      ;CLEAR PIRQ
5752 023400 005037 177776      CLR      @0177776      ;CLEAR PSW
5753 023404 010237 000004      MOV      R2,@04      ;RESTORE VECTORS TO PREVIOUS STATE
5754 023410 010337 000006      MOV      R3,@06      ;
5755 023414      FIN121:
5756      ;
5757 023414      ;E122:
5758      ;
5759      ;      TEST SPL (SET PRIORITY LEVEL)
5760
5761 023414 012705 000010      MOV      @8,R5      ;INIT COUNTER
5762 023420 012701 023644      MOV      @T122B,R1      ;SETUP POINTER TO DATA
5763 023424 012737 030000 177776 1$:      MOV      @30000,@0177776      ;INIT PSW
5764 023432 004767 000054      JSR      PC,T122A      ;TEST INSTRUCTION
5765 023436 022137 177776      CMP      (R1),@0177776      ;IS PSW CORRECT
5766 023442 001401      BEQ      2$      ;YES GO ON
5767 023444 104001      ERROR    +1      ;CPU ERROR
5768                                ;NO GO TO ERROR
5769 023446 077512      SOB      R5,1$      ;REPEAT UNTIL ALL CASES ARE TESTED
5770
5771 023450 012705 000010      MOV      @8,R5      ;INIT COUNTER
5772 023454 012737 140000 177776 3$:      MOV      @140000,@0177776      ;SETUP PSW TO USER MODE
5773 023462 012706 000600      MOV      @600,R6      ;SETUP USER STACK
5774 023466 004767 000020      JSR      PC,T122A      ;TEST INSTRUCTION
5775 023472 022137 140017 177776      CMP      @140017,@0177776      ;IS PSW CORRECT
5776 023500 001401      BEQ      4$      ;YES GO ON
5777 023502 104001      ERROR    +1      ;CPU ERROR
5778                                ;NO GO TO ERROR
5779 023504 077515      SOB      R5,3$      ;REPEAT UNTIL ALL CASES ARE TESTED
5780
5781
5782 023506 000167 000152      JMP      FIN122
5783
  
```


CORDABO KDJ11 B CLUSTER MACY11 3001046) 05 APR 84 16:45 PAGE 108
 CORDAB.P11 05-APR-84 16:45

***** DOUBLE OPERAND TESTS *****

SEQ 0107

5784	023512	020527	000010	T122A:	CMP	R5,#8.	;FIND OUT WHAT COUNTER IS
5785	023516	001003			BNE	1\$	;IF NOT PRIORITY 0 GO TO 1\$
5786	023520	000277			SCC		;SET ALL CC BITS
5787	023522	000230			SPL	0	;SET PRIORITY TO 0
5788	023524	000446			BR	8\$	;RETURN
5789	023526	020527	000007	1\$:	CMP	R5,#7	;FIND OUT WHAT COUNTER IS
5790	023532	001003			BNE	2\$	;IF NOT PRIORITY 1 GO TO 2\$
5791	023534	000277			SCC		;SET ALL CC BITS
5792	023536	000231			SPL	1	;SET PRIORITY TO 1
5793	023540	000440			BR	8\$	;RETURN
5794	023542	020527	000006	2\$:	CMP	R5,#6	;FIND OUT WHAT COUNTER IS
5795	023546	001003			BNE	3\$	;IF NOT PRIORITY 2 GO TO 3\$
5796	023550	000277			SCC		;SET ALL CC BITS
5797	023552	000232			SPL	2	;SET PRIORITY TO 2
5798	023554	000432			BR	8\$	;RETURN
5799	023556	020527	000005	3\$:	CMP	R5,#5	;FIND OUT WHAT COUNTER IS
5800	023562	001003			BNE	4\$	;IF NOT PRIORITY 3 GO TO 4\$
5801	023564	000277			SCC		;SET ALL CC BITS
5802	023566	000233			SPL	3	;SET PRIORITY TO 3
5803	023570	000424			BR	8\$	;RETURN
5804	023572	020527	000004	4\$:	CMP	R5,#4	;FIND OUT WHAT COUNTER IS
5805	023576	001003			BNE	5\$	;IF NOT PRIORITY 4 GO TO 5\$
5806	023600	000277			SCC		;SET ALL CC BITS
5807	023602	000234			SPL	4	;SET PRIORITY TO 4
5808	023604	000416			BR	8\$	;RETURN
5809	023606	020527	000003	5\$:	CMP	R5,#3	;FIND OUT WHAT COUNTER IS
5810	023612	001003			BNE	6\$	;IF NOT PRIORITY 5 GO TO 6\$
5811	023614	000277			SCC		;SET ALL CC BITS
5812	023616	000235			SPL	5	;SET PRIORITY TO 5
5813	023620	000410			BR	8\$	;RETURN
5814	023622	020527	000002	6\$:	CMP	R5,#2	;FIND OUT WHAT COUNTER IS
5815	023626	001003			BNE	7\$	;IF NOT PRIORITY 6 GO TO 7\$
5816	023630	000277			SCC		;SET ALL CC BITS
5817	023632	000236			SPL	6	;SET PRIORITY TO 6
5818	023634	000402			BR	8\$	;RETURN
5819	023636	000277		7\$:	SCC		;SET ALL CC BITS
5820	023640	000237			SPL	7	;SET PRIORITY TO 7
5821	023642	000207		8\$:	RTS	PC	;RETURN
5822				:			
5823	023644	030017		T122B:	.WORD	30017	
5824	023646	030057			.WORD	30057	
5825	023650	030117			.WORD	30117	
5826	023652	030157			.WORD	30157	
5827	023654	030217			.WORD	30217	
5828	023656	030257			.WORD	30257	
5829	023660	030317			.WORD	30317	
5830	023662	030357			.WORD	30357	
5831	023664	000240		FIN122:	NOP		
5832				:			
5833	023666			T123:			
5834				:			
5835	023666	005037	177776		TEST TESTSET INSTRUCTION (MULTI PROCESSING INST)		
5836	023672	012703	000012		CLR	#0177776	;INIT PSW
5837	023676	012701	000400		MOV	#10.,R5	;INIT COUNTER
5838	023702	012700	024050		MOV	#400,R1	;SETUP DESTINATION
5839	023706	012021		100\$:	MOV	#T123A,R0	;SETUP SOURCE
						(R0)+,(R1)+	;RELOCATE TABLES

COKDABO KDJ11 8 CLUSTER MACY11 30(1046) 05 APR 84 16:45 PAGE 109
 COKDAB.P11 05-APR-84 16:45 ***** DOUBLE OPERAND TESTS *****

SEQ 0108

5840	023710	077302			SOB	R3,100\$		;ARE WE DONE
5841	023712	013746	000010		MOV	@10,-(SP)		;SAVE VECTOR
5842	023716	012737	024074	000010	MOV	@T123D,@10		;SETUP NEW VECTOR
5843	023724	005000			CLR	R0		;INIT R0
5844	023726	012701	000400		MOV	@400,R1		;SETUP POINTERS TO TABLES
5845	023732	012702	000410		MOV	@410,R2		
5846	023736	012703	000416		MOV	@416,R3		
5847	023742	010104			MOV	R1,R4		
5848	023744	012737	030000	177776	1\$:	MOV	@30000,@177776	;SETUP PSW
5849	023752	000262			SEV			;SET V BIT
5850	023754	007221			.WORD	7221		;TEST INSTRUCTION
5851	023756	022237	177776		CMP	(R2)+,@177776		;IS PSW CORRECT
5852	023762	001401			BEQ	2\$		;YES GO ON
5853	023764	104001			ERROR	+1		;CPU ERROR
5854								;NO GO TO ERROR
5855	023766	020013			2\$:	CMP	R0,(R3)	;IS R0 CORRECT
5856	023770	001401			BEQ	3\$		;YES GO ON
5857	023772	104001			ERROR	+1		;CPU ERROR
5858								;NO GO TO ERROR
5859	023774	005204			3\$:	INC	R4	;SETUP EXPECTED DATA
5860	023776	005204			INC	R4		
5861	024000	020401			CMP	R4,R1		;IS R1 CORRECT
5862	024002	001401			BEQ	4\$		;YES GO ON
5863	024004	104001			ERROR	+1		;CPU ERROR
5864								;NO GO TO ERROR
5865	024006	052713	000001		4\$:	BIS	@1,(R3)	;SETUP EXPECTED DATA
5866	024012	022341			CMP	(R3)+,-(R1)		;IS TEST LOCATION CORRECT
5867	024014	001401			BEQ	5\$		;YES GO ON
5868	024016	104001			ERROR	+1		;CPU ERROR
5869								;NO GO TO ERROR
5870	024020	005201			5\$:	INC	R1	;POINT TO NEXT TEST LOCATION
5871	024022	005201			INC	R1		
5872	024024	021127	177777		CMP	(R1),@177777		;ARE WE DONE
5873	024030	001345			BNE	1\$		;NO GO TO 1\$
5874	024032	012737	024076	000010	MOV	@T123E,@10		;SETUP NEW VECTOR
5875	024040	007201			.WORD	7201		;TEST INSTRUCTION ILLEGAL MODE
5876	024042	104001			ERROR	+1		;CPU ERROR
5877								;GO TO ERROR IF DIDN'T TRAP
5878	024044	000167	000032		JMP	T123F		
5879								
5880								
5881	024050	167604			T123A:	.WORD	167604	
5882	024052	000000				.WORD	0	
5883	024054	000001				.WORD	1	
5884	024056	177777				.WORD	177777	
5885	024060	030010			T123B:	.WORD	30010	
5886	024062	030004				.WORD	30004	
5887	024064	030001				.WORD	30001	
5888	024066	167604			T123C:	.WORD	167604	
5889	024070	000000				.WORD	0	
5890	024072	000001				.WORD	1	
5891	024074	104001			T123D:	ERROR	+1	;CPU ERROR
5892								;GO TO ERROR IF TRAPPED
5893	024076	005726			T123E:	ISI	(SP)+	;CLEAN UP STACK
5894	024100	005726				ISI	(SP)+	
5895	024102	012637	000010		T123F:	MOV	(SP)+,@10	;RESTORE VECTOR

CORRADO KDJ11 B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 110
 CORRAD.P11 05-APR 84 16:45 ***** DOUBLE OPERAND TESTS *****

SEQ 0109

```

5896
5897
5898 024106      ;E124:
5899      ;
5900 024106 005037 177776      ; TEST WRTLOCK (WRITE LOCK MULTI PROCESSING INST)
5901 024112 012703 000012      CLR      @#177776      ; INIT PSW
5902 024116 012701 000400      MOV      @#10.,R3      ; INIT COUNTER
5903 024122 012700 024302      MOV      @400,R1      ; SETUP DESTINATION
5904 024126 012021      100$: MOV      @T124A,R0      ; SETUP SOURCE
5905 024130 077302      SOB      R3,100$      ; RELOCATE TABLES
5906 024132 013746 000010      MOV      @#10,-(SP)      ; ARE WE DONE
5907 024136 012737 024326 000010      MOV      @T124D,@#10      ; SAVE VECTOR
5908 024144 012701 000400      MOV      @400,R1      ; SETUP NEW VECTOR
5909 024150 012702 000410      MOV      @410,R2      ; SETUP POINTERS TO TABLES
5910 024154 012703 000416      MOV      @416,R3
5911 024160 010204      MOV      R2,R4
5912 024162 012737 030000 177776 1$: MOV      @30000,@#177776      ; SETUP PSW
5913 024170 011100      MOV      (R1),R0      ; SETUP R0
5914 024172 020327 000416      CMP      R3,@416      ; IS THIS THE FIRST TEST CASE
5915 024176 001401      BEQ      2$      ; YES GO TO 2$
5916 024200 000402      BR      3$      ; NO GO TO 3$
5917 024202 000261      2$: SEC      ; SET C BIT
5918 024204 000401      BR      4$
5919 024206 000241      3$: CLC      ; CLEAR C BIT
5920 024210 000262      4$: SEV      ; SET V BIT
5921 024212 007322      .WORD      7322      ; TEST INSTRUCTION
5922 024214 022337 177776      CMP      (R3)+,@#177776      ; IS PSW CORRECT
5923 024220 001401      BEQ      5$      ; YES GO ON
5924 024222 104001      ERROR      +1      ; CPU ERROR
5925      ; NO GO TO ERROR
5926 024224 021100      5$: CMP      (R1),R0      ; IS R0 CORRECT
5927 024226 001401      BEQ      6$      ; YES GO ON
5928 024230 104001      ERROR      +1      ; CPU ERROR
5929      ; NO GO TO ERROR
5930 024232 005204      6$: INC      R4      ; SETUP EXPECTED DATA
5931 024234 005204      INC      R4
5932 024236 020204      CMP      R2,R4      ; IS R2 CORRECT
5933 024240 001401      BEQ      7$      ; YES GO ON
5934 024242 104001      ERROR      +1      ; CPU ERROR
5935      ; NO GO TO ERROR
5936 024244 022142      7$: CMP      (R1)+,-(R2)      ; IS TEST LOCATION CORRECT
5937 024246 001401      BEQ      8$      ; YES GO ON
5938 024250 104001      ERROR      +1      ; CPU ERROR
5939      ; NO GO TO ERROR
5940 024252 005202      8$: INC      R2      ; POINT TO NEXT TEST LOCATION
5941 024254 005202      INC      R2
5942 024256 021127 177777      CMP      (R1),@#177777      ; ARE WE DONE
5943 024262 001337      BNE      1$      ; NO GO TO 1$
5944 024264 012737 024330 000010      MOV      @T124E,@#10      ; SETUP NEW VECTOR
5945 024272 007302      .WORD      7302      ; TEST INSTRUCTION ILLEGAL MODE
5946 024274 104001      ERROR      +1      ; CPU ERROR
5947      ; GO TO ERROR IF DIDN'T TRAP
5948 024276 000167 000032      JMP      T124F
5949
5950
5951 024302 167604      ;T124A: .WORD      167604

```

COKDABO KDJ11 B CLUSTER MAC11 30(1046) 05-APR-84 16:45 PAGE 111
 COKDAB.P11 05-APR-84 16:45 ***** DOUBLE OPERAND TESTS *****

SEQ 0110

```

5952 024304 000000      .WORD 0
5953 024306 000001      .WORD 1
5954 024310 177777      .WORD 177777
5955 024312 177777      T124B: .WORD 177777
5956 024314 177777      .WORD 177777
5957 024316 177777      .WORD 177777
5958 024320 030011      T124C: .WORD 30011
5959 024322 030004      .WORD 30004
5960 024324 030000      .WORD 30000
5961 024326 104001      T124D: ERROR +1      ;CPU ERROR
5962                                ;GO TO ERROR IF TRAPPED
5963 024330 005726      T124E: TST (SP)+      ;CLEAN UP STACK
5964 024332 005726      TST (SP)+      ;
5965 024334 012637 000010 T124F: MOV (SP)+,R10      ;RESTORE VECTOR
5966
5967      ;
5968 024340      TE125:
5969      ;      TEST MUL (MULTIPLY INST)
5970 024340 005037 177776      CLR R1,R177776      ;INIT PS
5971 024344 012701 024600      MOV R1,TE125A,R1      ;SETUP POINTERS TO TABLES
5972
5973 024350 010137 110502      1$: MOV R1,R1,EXPDAT      ;
5974 024354 062737 000002 110502 ADD R2,R1,EXPDAT      ;POINT TO SOURCE
5975 024362 012703 122222      MOV R3,R1,22222,R3      ;INIT R3 TO A KNOWN STATE
5976 024366 011102      MOV R2,R1,R2      ;INIT DESTINATION REG
5977 024370 000277      SCC      ;SET ALL CC BITS
5978 024372 070261 000002      MUL R2,R1,R2      ;TEST INSTRUCTION
5979 024376 026137 000004 177776 CMP R4,R1,R1,177776      ;IS PS CORRECT
5980 024404 001401      BEQ R4,R1,2$      ;YES GO ON
5981 024406 104001      ERROR +1      ;CPU ERROR
5982                                ;NO GO TO ERROR
5983 024410 026103 000006      2$: CMP R3,R1,R3      ;IS R3 CORRECT
5984 024414 001401      BEQ R3,R1,3$      ;YES GO ON
5985 024416 104001      ERROR +1      ;CPU ERROR
5986                                ;NO GO TO ERROR
5987 024420 026102 000010      3$: CMP R2,R1,R2      ;IS R2 CORRECT
5988 024424 001401      BEQ R2,R1,4$      ;YES GO ON
5989 024426 104001      ERROR +1      ;CPU ERROR
5990                                ;NO GO TO ERROR
5991 024430 026177 000002 064044 4$: CMP R5,R1,EXPDAT      ;IS SOURCE LOCATION OK
5992 024436 001401      BEQ R5,R1,5$      ;YES GO ON
5993 024440 104001      ERROR +1      ;CPU ERROR
5994                                ;NO GO TO ERROR
5995 024442 062701 000012      5$: ADD R1,R1,12,R1      ;GO TO NEXT TEST
5996 024446 020127 025026      CMP R1,R1,FIN125      ;ARE WE FINISHED
5997 024452 001336      BNE R1,R1,1$      ;NO GO TO 1$
5998
5999      ;
6000      ;SECOND PART
6001      ;USING ODD REGISTER
6002      ;
6003 024454 012701 024600      6$: MOV R1,TE125A,R1      ;SETUP POINTERS TO TABLES
6004 024460      7$:
6005 024460 010102      MOV R1,R2
6006 024462 012706 001000      MOV R6,R1,STBOT,R6      ;INIT R6 TO A KNOWN STATE
6007 024466 012704 000004      MOV R4,R4      ;SETUP R4 VALUE

```

```

6008 024472 011105      MOV      (R1),R5      ;INIT DESTINATION REG
6009 024474 000277      SCC              ;SET ALL CC BITS
6010 024476 070561 000002 MUL      2(R1),R5      ; TEST INSTRUCTION
6011 024502 026137 000004 177776 CMP      4(R1),#0177776 ;IS PS CORRECT
6012 024510 001401      BEQ      8$          ;YES GO ON
6013 024512 104001      ERROR    +1          ;CPU ERROR
6014                                ;NO GO TO ERROR
6015 024514 026105 000006 8$:      CMP      6(R1),R5      ;IS R5 CORRECT
6016 024520 001401      BEQ      9$          ;YES GO ON
6017 024522 104001      ERROR    +1          ;CPU ERROR
6018                                ;NO GO TO ERROR
6019 024524 020627 001000 9$:      CMP      R6,#STBOT    ;IS R6 CORRECT
6020 024530 001403      BEQ      10$         ;YES GO ON
6021 024532 012706 001000      MOV      #STBOT,R6      ;RESTORE SP
6022 024536 104001      ERROR    +1          ;CPU ERROR
6023                                ;NO GO TO ERROR
6024 024540 005722      10$:      TST      (R2),#        ;POINT TO SOURCE OPERAND
6025 024542 021261 000002      CMP      (R2),2(R1)      ;IS SOURCE LOCATION OK
6026 024546 001401      BEQ      11$         ;YES GO ON
6027 024550 104001      ERROR    +1          ;CPU ERROR
6028                                ;NO GO TO ERROR
6029 024552 020427 000004 11$:      CMP      R4,#4        ;IS R4 CORRECT
6030 024556 001401      BEQ      12$         ;YES GO ON
6031 024560 104001      ERROR    +1          ;CPU ERROR
6032                                ;NO GO TO ERROR
6033 024562 062701 000012 12$:      ADD      #12,R1        ;
6034 024566 020127 025026      CMP      R1,#FIN125      ;ARE WE FINISHED
6035 024572 001332      BNE      7$          ;NO GO TO 7$
6036
6037
6038 024574 000167 000226      JMP      FIN125
6039
6040
6041 024600 177777      ;
6042 024602 177777      ;TE125A: .WORD 177777      ;MULTIPLICAND
6043 024604 000000      .WORD 177777      ;MULTIPLIER
6044 024606 000001      .WORD 0
6045 024610 000000      .WORD 1
6046                                .WORD 0
6047 024612 006772      .WORD 6772      ;MULTIPLICAND
6048 024614 100000      .WORD 100000      ;MULTIPLIER
6049 024616 000011      .WORD 11
6050 024620 000000      .WORD 0
6051 024622 174403      .WORD 174403
6052
6053 024624 177777      .WORD 177777      ;MULTIPLICAND
6054 024626 077777      .WORD 77777      ;MULTIPLIER
6055 024630 000010      .WORD 10
6056 024632 100001      .WORD 100001
6057 024634 177777      .WORD 177777
6058
6059 024636 077777      .WORD 77777      ;MULTIPLICAND
6060 024640 000456      .WORD 456      ;MULTIPLIER
6061 024642 000001      .WORD 1
6062 024644 177322      .WORD 177322
6063 024646 000226      .WORD 226

```

CORDBO RDJ11 H CLUSTER MACY11 30(1046) 05 APR 84 16:45 PAGE 113
 CORDAB.P11 05-APR-84 16:45 ***** DOUBLE OPERAND TESTS *****

SEQ 0112

6064					
6065	024650	173210	.WORD	173210	;MULTIPLICAND
6066	024652	000000	.WORD	0	;MULTIPLIER
6067	024654	000004	.WORD	4	
6068	024656	000000	.WORD	0	
6069	024660	000000	.WORD	0	
6070					
6071	024662	000000	.WORD	0	;MULTIPLICAND
6072	024664	003251	.WORD	3251	;MULTIPLIER
6073	024666	000004	.WORD	4	
6074	024670	000000	.WORD	0	
6075	024672	000000	.WORD	0	
6076					
6077	024674	000000	.WORD	0	;MULTIPLICAND
6078	024676	000000	.WORD	0	;MULTIPLIER
6079	024700	000004	.WORD	4	
6080	024702	000000	.WORD	0	
6081	024704	000000	.WORD	0	
6082					
6083	024706	100000	.WORD	100000	;MULTIPLICAND
6084	024710	000001	.WORD	1	;MULTIPLIER
6085	024712	000010	.WORD	10	
6086	024714	100000	.WORD	100000	
6087	024716	177777	.WORD	177777	
6088					
6089	024720	077777	.WORD	77777	;MULTIPLICAND
6090	024722	000001	.WORD	1	;MULTIPLIER
6091	024724	000000	.WORD	0	
6092	024726	077777	.WORD	77777	
6093	024730	000000	.WORD	0	
6094					
6095	024732	000010	.WORD	10	;MULTIPLICAND
6096	024734	010000	.WORD	10000	;MULTIPLIER
6097	024736	000001	.WORD	1	
6098	024740	100000	.WORD	100000	
6099	024742	000000	.WORD	0	
6100					
6101	024744	001452	.WORD	1452	;MULTIPLICAND
6102	024746	034527	.WORD	34527	;MULTIPLIER
6103	024750	000001	.WORD	1	
6104	024752	066506	.WORD	66506	
6105	024754	000265	.WORD	265	
6106					
6107	024756	000007	.WORD	7	;MULTIPLICAND
6108	024760	000400	.WORD	400	;MULTIPLIER
6109	024762	000000	.WORD	0	
6110	024764	003400	.WORD	3400	
6111	024766	000000	.WORD	0	
6112					
6113	024770	000002	.WORD	2	;MULTIPLICAND
6114	024772	100000	.WORD	100000	;MULTIPLIER
6115	024774	000011	.WORD	11	
6116	024776	000000	.WORD	0	
6117	025000	177777	.WORD	177777	
6118					
6119	025002	100000	.WORD	100000	;MULTIPLICAND

6120	025004	077777			.WORD	77777		;MULTIPLIER
6121	025006	000011			.WORD	11		
6122	025010	100000			.WORD	100000		
6123	025012	140000			.WORD	140000		
6124								
6125	025014	000001			.WORD	1		;MULTIPLICAND
6126	025016	177777			.WORD	177777		;MULTIPLIER
6127	025020	000010			.WORD	10		
6128	025022	177777			.WORD	177777		
6129	025024	177777			.WORD	177777		
6130	025026							
6131								
6132	025026							
6133								
6134	025026	005037	177776		TEST DIV (DIVIDE INST)			
6135	025032	005006			CLR	00177776		;INIT PSW
6136	025034	013705	000000		CLR	R6		;INIT SP
6137	025040	013701	000002		MOV	000,R5		;SAVE VECTORS
6138	025044	012737	000137	000000	MOV	002,R1		
6139	025052	012737	025074	000002	MOV	0137,000		;SETUP NEW VECTORS
6140	025060	000277			MOV	0TE126A,002		
6141	025062	071627	000032		SCC			;SET ALL CC BITS
6142	025066	012706	001000		DIV	02,R6		;TEST INSTRUCTION
6143	025072	104001			A126: MOV	0STBOT,R6		;RESTORE SP BEFORE GOING TO ERROR
6144					ERROR	+1		;CPU ERROR
6145	025074	022737	000000	177776	TE126A: CMP	00,00177776		;IF R7 ISN'T CORRECT GO TO ERROR
6146	025102	001403			BEQ	1\$		;IS PS CORRECT
6147	025104	012706	001000		MOV	0STBOT,R6		;YES GO ON
6148	025110	104001			ERROR	+1		;RESTORE SP BEFORE GOING TO ERROR
6149								;CPU ERROR
6150	025112	012704	025066		1\$: MOV	0A126,R4		;NO GO TO ERROR
6151	025116	006204			ASR	R4		;SETUP EXPECTED DATA
6152	025120	020406			CMP	R4,R6		
6153	025122	001403			BEQ	2\$		;IS R6 CORRECT
6154	025124	012706	001000		MOV	0STBOT,R6		;YES GO ON
6155	025130	104001			ERROR	+1		;RESTORE SP BEFORE GOING TO ERROR
6156								;CPU ERROR
6157	025132	010537	000000		2\$: MOV	R5,000		;NO GO TO ERROR
6158	025136	010137	000002		MOV	R1,002		;RESTORE VECTORS
6159	025142	012706	001000		MOV	0STBOT,R6		
6160	025146	012702	000006		MOV	06,R2		;INIT SP
6161	025152	012703	000047		MOV	047,R3		;INIT GPR 2
6162	025156	000277			SCC			;INIT GPR 3
6163	025160	071302			DIV	R2,R3		;SET ALL CC BITS
6164	025162	022737	000002	177776	CMP	02,00177776		;TEST INSTRUCTION
6165	025170	001401			BEQ	3\$		;IS PS CORRECT
6166	025172	104001			ERROR	+1		;YES GO ON
6167								;CPU ERROR
6168	025174	022702	000006		3\$: CMP	06,R2		;NO GO TO ERROR
6169	025200	001401			BEQ	4\$		;IS R2 CORRECT
6170	025202	104001			ERROR	+1		;YES GO ON
6171								;CPU ERROR
6172	025204	022703	000047		4\$: CMP	047,R3		;NO GO TO ERROR
6173	025210	001401			BEQ	5\$		;IS R3 CORRECT
6174	025212	104001			ERROR	+1		;YES GO ON
6175								;CPU ERROR
								;NO GO TO ERROR

Address	Hex Address	Hex Instruction	Hex Comment	Hex Label	Hex Instruction	Hex Comment
6176	025214	005004		5\$:	CLR R4	; INIT R4
6177	025216	012705	000004		MOV #4,R5	; INIT R5
6178	025222	000277			SCC	; SET ALL CC BITS
6179	025224	071427	000000		DIV #0,R4	; TEST INSTRUCTION
6180	025230	022737	000007	177776	CMP #7,#177776	; IS PS CORRECT
6181	025236	001401			BEQ 6\$	; YES GO ON
6182	025240	104001			ERROR +1	; CPU ERROR
6183						; NO GO TO ERROR
6184	025242	022704	000000	6\$:	CMP #0,R4	; IS R4 CORRECT
6185	025246	001401			BEQ 7\$	; YES GO ON
6186	025250	104001			ERROR +1	; CPU ERROR
6187						; NO GO TO ERROR
6188	025252	022705	000004	7\$:	CMP #4,R5	; IS R5 CORRECT
6189	025256	001401			BEQ 8\$	; YES GO ON
6190	025260	104001			ERROR +1	; CPU ERROR
6191						; NO GO TO ERROR
6192	025262	012700	000004	8\$:	MOV #4,R0	; INIT R0
6193	025266	012705	000010		MOV #10,R5	; INIT R5
6194	025272	005004			CLR R4	; INIT R4
6195	025274	000277			SCC	; SET ALL CC BITS
6196	025276	071400			DIV R0,R4	; TEST INSTRUCTION
6197	025300	022737	000000	177776	CMP #0,#177776	; IS PS CORRECT
6198	025306	001405			BEQ 9\$	; YES GO ON
6199	025310	010067	153064		MOV R0,400	; SAVE R0
6200	025314	104001			ERROR +1	; CPU ERROR
6201						; NO GO TO ERROR
6202	025316	016700	153056		MOV 400,R0	; RESTORE R0
6203	025322	022700	000004	9\$:	CMP #4,R0	; IS R0 CORRECT
6204	025326	001401			BEQ 10\$	; YES GO ON
6205	025330	104001			ERROR +1	; CPU ERROR
6206						; NO GO TO ERROR
6207	025332	022704	000002	10\$:	CMP #2,R4	; IS R4 CORRECT
6208	025336	001401			BEQ 11\$	; YES GO ON
6209	025340	104001			ERROR +1	; CPU ERROR
6210						; NO GO TO ERROR
6211	025342	022705	000000	11\$:	CMP #0,R5	; IS R5 CORRECT
6212	025346	001401			BEQ 12\$	; YES GO ON
6213	025350	104001			ERROR +1	; CPU ERROR
6214						; NO GO TO ERROR
6215	025352	012705	000010	12\$:	MOV #10,R5	; INIT R5
6216	025356	005004			CLR R4	; INIT R4
6217	025360	000277			SCC	; SET ALL CC BITS
6218	025362	071427	000003		DIV #3,R4	; TEST INSTRUCTION
6219	025366	022737	000000	177776	CMP #0,#177776	; IS PS CORRECT
6220	025374	001401			BEQ 13\$	; YES GO ON
6221	025376	104001			ERROR +1	; CPU ERROR
6222						; NO GO TO ERROR
6223	025400	022704	000002	13\$:	CMP #2,R4	; IS R4 CORRECT
6224	025404	001401			BEQ 14\$	; YES GO ON
6225	025406	104001			ERROR +1	; CPU ERROR
6226						; NO GO TO ERROR
6227	025410	022705	000002	14\$:	CMP #2,R5	; IS R5 CORRECT
6228	025414	001401			BEQ 15\$	; YES GO ON
6229	025416	104001			ERROR +1	; CPU ERROR
6230						; NO GO TO ERROR
6231						

COKDABO KDJ11-B CLUSTER MACY11 30(1046) 05 APR-84 16:45 PAGE 116
 COKDAB.P11 05-APR-84 16:45 ***** DOUBLE OPERAND TESTS *****

SEQ 0115

```

6232      ;
6233      ;
6234      025420 012701 025550      15$:  MOV    #TE126B,R1          ;SETUP POINTERS TO TABLES
6235      ;
6236      025424 010137 110502      16$:  MOV    R1,@#EXPDAT      ;SAVE A COPY OF R1
6237      025430 011104              MOV    (R1),R4          ;INIT R4
6238      025432 016103 000004      MOV    4(R1),R3          ;SAVE SOURCE
6239      ;
6240      025436 016105 000002      MOV    2(R1),R5          ;INIT R5
6241      025442 000277              SCC                     ;SET ALL CC BITS
6242      025444 071461 000004      DIV    4(R1),R4          ;TEST INSTRUCTION
6243      025450 026137 000006 177776 CMP    6(R1),@#177776    ;IS PS CORRECT
6244      025456 001401              BEQ     17$              ;YES GO ON
6245      025460 104001              ERROR   +1              ;CPU ERROR
6246      ;                          ;NO GO TO ERROR
6247      025462 026105 000010      17$:  CMP    10(R1),R5         ;IS R5 CORRECT
6248      025466 001401              BEQ     18$              ;YES GO ON
6249      025470 104001              ERROR   +1              ;CPU ERROR
6250      ;                          ;NO GO TO ERROR
6251      025472 026104 000012      18$:  CMP    12(R1),R4         ;IS R4 CORRECT
6252      025476 001401              BEQ     19$              ;YES GO ON
6253      025500 104001              ERROR   +1              ;CPU ERROR
6254      ;                          ;NO GO TO ERROR
6255      025502 023701 110502      19$:  CMP    @#EXPDAT,R1      ;IS R1 CORRECT
6256      025506 001403              BEQ     20$              ;YES GO ON
6257      025510 104001              ERROR   +1              ;CPU ERROR
6258      ;                          ;NO GO TO ERROR
6259      025512 013701 110502      MOV    @#EXPDAT,R1      ;RESTORE CORRECT VALUE
6260      025516 026103 000004      20$:  CMP    4(R1),R3          ;IS SOURCE CORRECT
6261      025522 001403              BEQ     21$              ;YES GO ON
6262      025524 104001              ERROR   +1              ;CPU ERROR
6263      ;                          ;NO GO TO ERROR
6264      025526 010361 000004      MOV    R3,4(R1)          ;TRY TO RESTORE CODE
6265      025532 062701 000014      21$:  ADD     #14,R1          ;POINT TO NEXT LOCATION
6266      025536 021127 000333      CMP     (R1),#333      ;ARE WE DONE
6267      025542 001330              BNE     16$              ;NO GO TO 16$
6268      ;
6269      ;
6270      025544 000167 000316      JMP     FIN126
6271      ;
6272      ;
6273      025550 177777      TE126B: .WORD 177777 ;DIVIDEND
6274      025552 177777      .WORD 177777 ;INIT R5
6275      025554 177777      .WORD 177777 ;DIVISOR
6276      025556 000000      .WORD 0      ;PSW
6277      025560 000000      .WORD 0      ;R5 RESULT
6278      025562 000001      .WORD 1      ;R4 RESULT
6279      ;
6280      025564 000000      .WORD 0      ;DIVIDEND
6281      025566 177777      .WORD 177777 ;INIT R5
6282      025570 177777      .WORD 177777 ;DIVISOR
6283      025572 000012      .WORD 12     ;PSW
6284      025574 177777      .WORD 177777 ;R5 RESULT
6285      025576 000000      .WORD 0      ;R4 RESULT
6286      ;
6287      025600 177777      .WORD 177777 ;DIVIDEND

```

COKDABO KDJ11-B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 117
 COKDAB.P11 05-APR-84 16:45 ***** DOUBLE OPERAND TESTS *****

SEQ 0116

6288	025602	000000	.WORD	0	; INIT R5
6289	025604	177777	.WORD	177777	; DIVISOR
6290	025606	000002	.WORD	2	; PSW
6291	025610	000000	.WORD	0	; R5 RESULT
6292	025612	177777	.WORD	177777	; R4 RESULT
6293					
6294	025614	000000	.WORD	0	; DIVIDEND
6295	025616	007642	.WORD	7642	; INIT R5
6296	025620	007643	.WORD	7643	; DIVISOR
6297	025622	000004	.WORD	4	; PSW
6298	025624	007642	.WORD	7642	; R5 RESULT
6299	025626	000000	.WORD	0	; R4 RESULT
6300					
6301	025630	000000	.WORD	0	; DIVIDEND
6302	025632	000137	.WORD	137	; INIT R5
6303	025634	177543	.WORD	177543	; DIVISOR
6304	025636	000004	.WORD	4	; PSW
6305	025640	000137	.WORD	137	; R5 RESULT
6306	025642	000000	.WORD	0	; R4 RESULT
6307					
6308	025644	000000	.WORD	0	; DIVIDEND
6309	025646	007643	.WORD	7643	; INIT R5
6310	025650	007643	.WORD	7643	; DIVISOR
6311	025652	000000	.WORD	0	; PSW
6312	025654	000000	.WORD	0	; R5 RESULT
6313	025656	000001	.WORD	1	; R4 RESULT
6314					
6315	025660	100000	.WORD	100000	; DIVIDEND
6316	025662	004376	.WORD	4376	; INIT R5
6317	025664	010021	.WORD	10021	; DIVISOR
6318	025666	000012	.WORD	12	; PSW
6319	025670	004376	.WORD	4376	; R5 RESULT
6320	025672	100000	.WORD	100000	; R4 RESULT
6321					
6322	025674	177700	.WORD	177700	; DIVIDEND
6323	025676	170033	.WORD	170033	; INIT R5
6324	025700	010021	.WORD	10021	; DIVISOR
6325	025702	000010	.WORD	10	; PSW
6326	025704	171307	.WORD	171307	; R5 RESULT
6327	025706	176024	.WORD	176024	; R4 RESULT
6328					
6329	025710	177700	.WORD	177700	; DIVIDEND
6330	025712	170033	.WORD	170033	; INIT R5
6331	025714	167757	.WORD	167757	; DIVISOR
6332	025716	000000	.WORD	0	; PSW
6333	025720	171307	.WORD	171307	; R5 RESULT
6334	025722	001754	.WORD	1754	; R4 RESULT
6335					
6336	025724	000000	.WORD	0	; DIVIDEND
6337	025726	177777	.WORD	177777	; INIT R5
6338	025730	000001	.WORD	1	; DIVISOR
6339	025732	000002	.WORD	2	; PSW
6340	025734	177777	.WORD	177777	; R5 RESULT
6341	025736	000000	.WORD	0	; R4 RESULT
6342					
6343	025740	177777	.WORD	177777	; DIVIDEND

COKDABO KDJ11-B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 118
 COKDAB.P11 05-APR-84 16:45 ***** DOUBLE OPERAND TESTS *****

SEQ 0117

6344	025742	045716	.WORD	45716	; INIT R5
6345	025744	000001	.WORD	1	; DIVISOR
6346	025746	000012	.WORD	12	; PSW
6347	025750	045716	.WORD	45716	; R5 RESULT
6348	025752	177777	.WORD	177777	; R4 RESULT
6349					
6350	025754	000000	.WORD	0	; DIVIDEND
6351	025756	000002	.WORD	2	; INIT R5
6352	025760	177770	.WORD	177770	; DIVISOR
6353	025762	000004	.WORD	4	; PSW
6354	025764	000002	.WORD	2	; R5 RESULT
6355	025766	000000	.WORD	0	; R4 RESULT
6356					
6357	025770	177777	.WORD	177777	; DIVIDEND
6358	025772	177776	.WORD	177776	; INIT R5
6359	025774	000010	.WORD	10	; DIVISOR
6360	025776	000004	.WORD	4	; PSW
6361	026000	177776	.WORD	177776	; R5 RESULT
6362	026002	000000	.WORD	0	; R4 RESULT
6363					
6364	026004	000001	.WORD	1	; DIVIDEND
6365	026006	177777	.WORD	177777	; INIT R5
6366	026010	000001	.WORD	1	; DIVISOR
6367	026012	000002	.WORD	2	; PSW
6368	026014	177777	.WORD	177777	; R5 RESULT
6369	026016	000001	.WORD	1	; R4 RESULT
6370					
6371	026020	000001	.WORD	1	; DIVIDEND
6372	026022	000000	.WORD	0	; INIT R5
6373	026024	000002	.WORD	2	; DIVISOR
6374	026026	000002	.WORD	2	; PSW
6375	026030	000000	.WORD	0	; R5 RESULT
6376	026032	000001	.WORD	1	; R4 RESULT
6377					
6378	026034	000001	.WORD	1	; DIVIDEND
6379	026036	000000	.WORD	0	; INIT R5
6380	026040	000003	.WORD	3	; DIVISOR
6381	026042	000000	.WORD	0	; PSW
6382	026044	000001	.WORD	1	; R5 RESULT
6383	026046	052525	.WORD	52525	; R4 RESULT
6384					
6385	026050	000023	.WORD	23	; DIVIDEND
6386	026052	016054	.WORD	16054	; INIT R5
6387	026054	016537	.WORD	16537	; DIVISOR
6388	026056	000000	.WORD	0	; PSW
6389	026060	010222	.WORD	10222	; R5 RESULT
6390	026062	000246	.WORD	246	; R4 RESULT
6391					
6392	026064	000333	.WORD	333	
6393	026066				
6394					
6395	026066				
6396					
6397	026066	005037			
6398	026072	012702			
6399	026076	000277			

FIN126;

;

TE127;

;

TEST ASH (ARITHMETIC SHIFT)

CLR #0177776

MOV #1,R2

SCC

; INIT PSW

; SETUP OPERAND

; SET ALL CC BITS

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

C10

CORDBO RDJ11 B CLUSTER MACY11 30(1046) 05 APR 84 16:45 PAGE 120
 CORDAB.P11 05-APR 84 16:45 ***** DOUBLE OPERAND TESTS *****

SFQ 0119

6456					
6457	026266	177761	TE127A: .WORD	177761	.SOURCE
6458	026270	077777	.WORD	77777	.DEST
6459	026272	000005	.WORD	5	
6460	026274	000000	.WORD	0	
6461					
6462	026276	177700	.WORD	177700	.SOURCE
6463	026300	017777	.WORD	17777	.DEST
6464	026302	000000	.WORD	0	
6465	026304	017777	.WORD	17777	
6466					
6467	026306	177700	.WORD	177700	.SOURCE
6468	026310	100000	.WORD	100000	.DEST
6469	026312	000010	.WORD	10	
6470	026314	100000	.WORD	100000	
6471					
6472	026316	177777	.WORD	177777	.SOURCE
6473	026320	100000	.WORD	100000	.DEST
6474	026322	000010	.WORD	10	
6475	026324	140000	.WORD	140000	
6476					
6477	026326	177737	.WORD	177737	.SOURCE
6478	026330	177777	.WORD	177777	.DEST
6479	026332	000011	.WORD	11	
6480	026334	177777	.WORD	177777	
6481					
6482	026336	177706	.WORD	177706	.SOURCE
6483	026340	102000	.WORD	102000	.DEST
6484	026342	000007	.WORD	7	
6485	026344	000000	.WORD	0	
6486					
6487	026346	177710	.WORD	177710	.SOURCE
6488	026350	017777	.WORD	17777	.DEST
6489	026352	000013	.WORD	13	
6490	026354	177400	.WORD	177400	
6491					
6492	026356	177713	.WORD	177713	.SOURCE
6493	026360	000012	.WORD	12	.DEST
6494	026362	000000	.WORD	0	
6495	026364	050000	.WORD	50000	
6496					
6497	026366	177707	.WORD	177707	.SOURCE
6498	026370	170001	.WORD	170001	.DEST
6499	026372	000002	.WORD	2	
6500	026374	000200	.WORD	200	
6501					
6502	026376	177717	.WORD	177717	.SOURCE
6503	026400	000001	.WORD	1	.DEST
6504	026402	000012	.WORD	12	
6505	026404	100000	.WORD	100000	
6506					
6507	026406	177740	.WORD	177740	.SOURCE
6508	026410	017777	.WORD	17777	.DEST
6509	026412	000004	.WORD	4	
6510	026414	000000	.WORD	0	
6511					

D10

COKDABO RDJ11 B CLUSTER MAC111 30(1046) 05 APR 84 16:45 PAGE 121
COKDAB.P11 05-APR 84 16:45 ***** DOUBLE OPERAND TESTS *****

SEQ 0120

6512	026416	177771	.WORD	177771	;SOURCE
6513	026420	150000	.WORD	150000	;DEST
6514	026422	000010	.WORD	10	
6515	026424	177640	.WORD	177640	
6516					
6517	026426	177742	.WORD	177742	;SOURCE
6518	026430	100000	.WORD	100000	;DEST
6519	026432	000011	.WORD	11	
6520	026434	177777	.WORD	177777	
6521					
6522	026436	177764	.WORD	177764	;SOURCE
6523	026440	100000	.WORD	100000	;DEST
6524	026442	000010	.WORD	10	
6525	026444	177770	.WORD	177770	
6526					
6527	026446	177750	.WORD	177750	;SOURCE
6528	026450	052525	.WORD	52525	;DEST
6529	026452	000004	.WORD	4	
6530	026454	000000	.WORD	0	
6531					
6532	026456	177760	.WORD	177760	;SOURCE
6533	026460	100000	.WORD	100000	;DEST
6534	026462	000011	.WORD	11	
6535	026464	177777	.WORD	177777	
6536					
6537	026466	177770	.WORD	177770	;SOURCE
6538	026470	100000	.WORD	100000	;DEST
6539	026472	000010	.WORD	10	
6540	026474	177600	.WORD	177600	
6541					
6542	026476	177712	.WORD	177712	;SOURCE
6543	026500	004367	.WORD	4367	;DEST
6544	026502	000013	.WORD	13	
6545	026504	156000	.WORD	156000	
6546					
6547	026506	177764	.WORD	177764	;SOURCE
6548	026510	017777	.WORD	17777	;DEST
6549	026512	000001	.WORD	1	
6550	026514	000001	.WORD	1	
6551					
6552	026516	177701	.WORD	177701	;SOURCE
6553	026520	110000	.WORD	110000	;DEST
6554	026522	000003	.WORD	3	
6555	026524	020000	.WORD	20000	
6556					
6557	026526	000240			
6558					
6559	026530				
6560					
6561	026530	005037	177776		
6562	026534	012701	000023		
6563	026540	012705	052525		
6564	026544	005004			
6565	026546	000277			
6566	026550	073401			
6567	026552	023727	177776	000012	

FIN127:	NOP
IE130:	
	TEST ASHC (ARITHMETIC SHIFT COMBINED)
	CLR @0177776 ;INIT PSW
	MOV @23,R1 ;SETUP R1
	MOV @52525,R5 ;SETUP R5
	CLR R4 ;SETUP R4
	SCC ;SET ALL CC BITS
	ASHC R1,R4 ;TEST INSTRUCTION
	CMP @0177776,@12 ;IS PS CORRECT

6568	026560	001401			BEG	1\$			;YES GO ON
6569	026562	104001			ERROR	+1			;CPU ERROR
6570									;NO GO TO ERROR
6571	026564	020127	000023	1\$:	CMP	R1,#23			;IS R1 CORRECT
6572	026570	001401			BEG	2\$			;YES GO ON
6573	026572	104001			ERROR	+1			;CPU ERROR
6574									;NO GO TO ERROR
6575	026574	020427	125250	2\$:	CMP	R4,#125250			;IS R4 CORRECT
6576	026600	001401			BEG	3\$			;YES GO ON
6577	026602	104001			ERROR	+1			;CPU ERROR
6578									;NO GO TO ERROR
6579	026604	020527	000000	3\$:	CMP	R5,#0			;IS R5 CORRECT
6580	026610	001401			BEG	4\$			;YES GO ON
6581	026612	104001			ERROR	+1			;CPU ERROR
6582									;NO GO TO ERROR
6583	026614	012703	052525	4\$:	MOV	#52525,R3			;SETUP R3
6584	026620	005002			CLR	R2			;SETUP R2
6585	026622	012704	164731		MOV	#164731,R4			;SETUP R4
6586	026626	000277			SCC				;SET ALL CC BITS
6587	026630	073327	000023		ASHC	#23,R3			;TEST INSTRUCTION
6588	026634	023727	177776	000012	CMP	#0177776,#12			;IS PS CORRECT
6589	026642	001401			BEG	5\$			;YES GO ON
6590	026644	104001			ERROR	+1			;CPU ERROR
6591									;NO GO TO ERROR
6592	026646	020227	000000	5\$:	CMP	R2,#0			;IS R2 CORRECT
6593	026652	001401			BEG	6\$			;YES GO ON
6594	026654	104001			ERROR	+1			;CPU ERROR
6595									;NO GO TO ERROR
6596	026656	020327	000000	6\$:	CMP	R3,#0			;IS R3 CORRECT
6597	026662	001401			BEG	7\$			;YES GO ON
6598	026664	104001			ERROR	+1			;CPU ERROR
6599									;NO GO TO ERROR
6600	026666	020427	164731	7\$:	CMP	R4,#164731			;IS R4 CORRECT
6601	026672	001401			BEG	8\$			;YES GO ON
6602	026674	104001			ERROR	+1			;CPU ERROR
6603									;NO GO TO ERROR
6604									
6605									
6606	026676	012701	027006	8\$:	MOV	#TE130A,R1			;SETUP POINTERS TO TABLES
6607									
6608	026702	010104		9\$:	MOV	R1,R4			;MAKE A COPY OF R1
6609	026704	016102	000002		MOV	2(R1),R2			;COPY R2
6610	026710	016103	000004		MOV	4(R1),R3			;COPY R3
6611	026714	000277			SCC				;SET ALL CC BITS
6612	026716	073211			ASHC	(R1),R2			;TEST INSTRUCTION
6613	026720	023761	177776	000006	CMP	#0177776,6(R1)			;IS PS CORRECT
6614	026726	001401			BEG	10\$			;YES GO ON
6615	026730	104001			ERROR	+1			;CPU ERROR
6616									;NO GO TO ERROR
6617	026732	026102	000010	10\$:	CMP	10(R1),R2			;IS R2 CORRECT
6618	026736	001401			BEG	11\$			;YES GO ON
6619	026740	104001			ERROR	+1			;CPU ERROR
6620									;NO GO TO ERROR
6621	026742	026103	000012	11\$:	CMP	12(R1),R3			;IS R3 CORRECT
6622	026746	001401			BEG	12\$			;YES GO ON
6623	026750	104001			ERROR	+1			;CPU ERROR

COKDABO KDJ11 B CLUSTER MACY11 30(1046) 05 APR-84 16:45 PAGE 123
COKDAB.P11 05-APR-84 16:45 ***** DOUBLE OPERAND

***** DOUBLE OPERAND TESTS *****

SEQ 0122

6624						;NO GO TO ERROR
6625	026752	020401		12\$:	CMP R4,R1	;IS R1 CORRECT
6626	026754	001402			BEG 13\$	;YES GO ON
6627	026756	104001			ERROR +1	;CPU ERROR
6628						;NO GO TO ERROR
6629	026760	010401			MOV R4,R1	
6630	026762	021114		13\$:	CMP (R1),(R4)	;IS SOURCE CORRECT
6631	026764	001401			BEG 14\$	;YES GO ON
6632	026766	104001			ERROR +1	;CPU ERROR
6633						;NO GO TO ERROR
6634						;POSSIBLE SOURCE CODE CORRUPTION
6635	026770	062701	000014	14\$:	ADD #14,R1	;GO TO NEXT TEST
6636	026774	020127	027416		CMP R1,#FIN150	;ARE WE DONE
6637	027000	001340			BNE 9\$	;NO GO TO 9\$
6638						
6639						
6640	027002	000167	000410		JMP FIN130	
6641				:		
6642				:TE130A:	.WORD 177700	;SOURCE
6643	027006	177700			.WORD 100125	;DESTINATION WORD 1
6644	027010	100125			.WORD 177777	;DESTINATION WORD 2
6645	027012	177777			.WORD 10	;TEST PSW
6646	027014	000010			.WORD 100125	;RESULT WORD 1
6647	027016	100125			.WORD 177777	;RESULT WORD 2
6648	027020	177777				
6649						
6650	027022	177777			.WORD 177777	;SOURCE
6651	027024	000001			.WORD 1	;DESTINATION WORD 1
6652	027026	000000			.WORD 0	;DESTINATION WORD 2
6653	027030	000000			.WORD 0	;TEST PSW
6654	027032	000000			.WORD 0	;RESULT WORD 1
6655	027034	100000			.WORD 100000	;RESULT WORD 2
6656						
6657	027036	177701			.WORD 177701	;SOURCE
6658	027040	047777			.WORD 47777	;DESTINATION WORD 1
6659	027042	100000			.WORD 100000	;DESTINATION WORD 2
6660	027044	000012			.WORD 12	;TEST PSW
6661	027046	117777			.WORD 117777	;RESULT WORD 1
6662	027050	000000			.WORD 0	;RESULT WORD 2
6663						
6664	027052	177706			.WORD 177706	;SOURCE
6665	027054	004256			.WORD 4256	;DESTINATION WORD 1
6666	027056	177700			.WORD 177700	;DESTINATION WORD 2
6667	027060	000002			.WORD 2	;TEST PSW
6668	027062	025677			.WORD 25677	;RESULT WORD 1
6669	027064	170000			.WORD 170000	;RESULT WORD 2
6670						
6671	027066	177711			.WORD 177711	;SOURCE
6672	027070	065700			.WORD 65700	;DESTINATION WORD 1
6673	027072	000012			.WORD 12	;DESTINATION WORD 2
6674	027074	000013			.WORD 13	;TEST PSW
6675	027076	100000			.WORD 100000	;RESULT WORD 1
6676	027100	012000			.WORD 12000	;RESULT WORD 2
6677						
6678	027102	177737			.WORD 177737	;SOURCE
6679	027104	000000			.WORD 0	;DESTINATION WORD 1

G10

COKDABO KDJ11 B CLUSTER MACY11 30(1046) 05 APR 84 16:45 PAGE 124
 COKDAB.P11 05 APR 84 16:45 ***** DOUBLE OPERAND TESTS *****

SEQ 0123

6680	027106	000001	.WORD	1	; DESTINATION WORD 2
6681	027110	000004	.WORD	4	; TEST PSW
6682	027112	000000	.WORD	0	; RESULT WORD 1
6683	027114	000000	.WORD	0	; RESULT WORD 2
6684					
6685	027116	177736	.WORD	177736	; SOURCE
6686	027120	000000	.WORD	0	; DESTINATION WORD 1
6687	027122	000001	.WORD	1	; DESTINATION WORD 2
6688	027124	000000	.WORD	0	; TEST PSW
6689	027126	040000	.WORD	40000	; RESULT WORD 1
6690	027130	000000	.WORD	0	; RESULT WORD 2
6691					
6692	027132	177740	.WORD	177740	; SOURCE
6693	027134	100000	.WORD	100000	; DESTINATION WORD 1
6694	027136	000000	.WORD	0	; DESTINATION WORD 2
6695	027140	000011	.WORD	11	; TEST PSW
6696	027142	177777	.WORD	177777	; RESULT WORD 1
6697	027144	177777	.WORD	177777	; RESULT WORD 2
6698					
6699	027146	177725	.WORD	177725	; SOURCE
6700	027150	177777	.WORD	177777	; DESTINATION WORD 1
6701	027152	174000	.WORD	174000	; DESTINATION WORD 2
6702	027154	000007	.WORD	7	; TEST PSW
6703	027156	000000	.WORD	0	; RESULT WORD 1
6704	027160	000000	.WORD	0	; RESULT WORD 2
6705					
6706	027162	177724	.WORD	177724	; SOURCE
6707	027164	177777	.WORD	177777	; DESTINATION WORD 1
6708	027166	174000	.WORD	174000	; DESTINATION WORD 2
6709	027170	000011	.WORD	11	; TEST PSW
6710	027172	100000	.WORD	100000	; RESULT WORD 1
6711	027174	000000	.WORD	0	; RESULT WORD 2
6712					
6713	027176	177733	.WORD	177733	; SOURCE
6714	027200	177777	.WORD	177777	; DESTINATION WORD 1
6715	027202	157023	.WORD	157023	; DESTINATION WORD 2
6716	027204	000012	.WORD	12	; TEST PSW
6717	027206	114000	.WORD	114000	; RESULT WORD 1
6718	027210	000000	.WORD	0	; RESULT WORD 2
6719					
6720	027212	177727	.WORD	177727	; SOURCE
6721	027214	000000	.WORD	0	; DESTINATION WORD 1
6722	027216	177777	.WORD	177777	; DESTINATION WORD 2
6723	027220	000013	.WORD	13	; TEST PSW
6724	027222	177600	.WORD	177600	; RESULT WORD 1
6725	027224	000000	.WORD	0	; RESULT WORD 2
6726					
6727	027226	177717	.WORD	177717	; SOURCE
6728	027230	177777	.WORD	177777	; DESTINATION WORD 1
6729	027232	000001	.WORD	1	; DESTINATION WORD 2
6730	027234	000011	.WORD	11	; TEST PSW
6731	027236	100000	.WORD	100000	; RESULT WORD 1
6732	027240	100000	.WORD	100000	; RESULT WORD 2
6733					
6734	027242	177741	.WORD	177741	; SOURCE
6735	027244	100000	.WORD	100000	; DESTINATION WORD 1

H10

COKDABO KDJ11 B CLUSTER MACY11 30(1046) 05 APR-84 16:45 PAGE 125
 COKDAB.P11 05-APR-84 16:45 ***** DOUBLE OPERAND TESTS *****

SEQ 0124

6736	027246	000000	.WORD	0	; DESTINATION WORD 2
6737	027250	000010	.WORD	10	; TEST PSW
6738	027252	177777	.WORD	177777	; RESULT WORD 1
6739	027254	177777	.WORD	177777	; RESULT WORD 2
6740					
6741	027256	177742	.WORD	177742	; SOURCE
6742	027260	037777	.WORD	37777	; DESTINATION WORD 1
6743	027262	177777	.WORD	177777	; DESTINATION WORD 2
6744	027264	000005	.WORD	5	; TEST PSW
6745	027266	000000	.WORD	0	; RESULT WORD 1
6746	027270	000000	.WORD	0	; RESULT WORD 2
6747					
6748	027272	177742	.WORD	177742	; SOURCE
6749	027274	077777	.WORD	77777	; DESTINATION WORD 1
6750	027276	177777	.WORD	177777	; DESTINATION WORD 2
6751	027300	000001	.WORD	1	; TEST PSW
6752	027302	000000	.WORD	0	; RESULT WORD 1
6753	027304	000001	.WORD	1	; RESULT WORD 2
6754					
6755	027306	177711	.WORD	177711	; SOURCE
6756	027310	065600	.WORD	65600	; DESTINATION WORD 1
6757	027312	000012	.WORD	12	; DESTINATION WORD 2
6758	027314	000003	.WORD	3	; TEST PSW
6759	027316	000000	.WORD	0	; RESULT WORD 1
6760	027320	012000	.WORD	12000	; RESULT WORD 2
6761					
6762	027322	177740	.WORD	177740	; SOURCE
6763	027324	077777	.WORD	77777	; DESTINATION WORD 1
6764	027326	177777	.WORD	177777	; DESTINATION WORD 2
6765	027330	000004	.WORD	4	; TEST PSW
6766	027332	000000	.WORD	0	; RESULT WORD 1
6767	027334	000000	.WORD	0	; RESULT WORD 2
6768					
6769	027336	177737	.WORD	177737	; SOURCE
6770	027340	177777	.WORD	177777	; DESTINATION WORD 1
6771	027342	177774	.WORD	177774	; DESTINATION WORD 2
6772	027344	000011	.WORD	11	; TEST PSW
6773	027346	177777	.WORD	177777	; RESULT WORD 1
6774	027350	177777	.WORD	177777	; RESULT WORD 2
6775					
6776	027352	177747	.WORD	177747	; SOURCE
6777	027354	100000	.WORD	100000	; DESTINATION WORD 1
6778	027356	174000	.WORD	174000	; DESTINATION WORD 2
6779	027360	000010	.WORD	10	; TEST PSW
6780	027362	177777	.WORD	177777	; RESULT WORD 1
6781	027364	177700	.WORD	177700	; RESULT WORD 2
6782					
6783	027366	177753	.WORD	177753	; SOURCE
6784	027370	006324	.WORD	6324	; DESTINATION WORD 1
6785	027372	071002	.WORD	71002	; DESTINATION WORD 2
6786	027374	000001	.WORD	1	; TEST PSW
6787	027376	000000	.WORD	0	; RESULT WORD 1
6788	027400	000146	.WORD	146	; RESULT WORD 2
6789					
6790	027402	177765	.WORD	177765	; SOURCE
6791	027404	102351	.WORD	102351	; DESTINATION WORD 1

CORRADO KDJ11 B CLUSTER MACY11 30(1046) 05 APR 84 16:45 PAGE 126
 CORRADO P11 05 APR 84 16:45

***** DOUBLE OPERAND TESTS *****

SEQ 0125

```

6792 027406 177231      .WORD 177231      ;DESTINATION WORD 2
6793 027410 000011      .WORD 11      ;TEST PSW
6794 027412 177760      .WORD 177760     ;RESULT WORD 1
6795 027414 116477      .WORD 116477     ;RESULT WORD 2
6796 027416
6797 027416      FIN130:
6798      MSPAU:
6799      ;
6800 027416 005006      ; TEST THAT AUTO DEC/INC OPERATIONS USING SP ARE ON WORD BOUNDRIES
6801 027420 112667 153434      CLR R6      ;CLEAR SP
6802 027424 022706 000002      MOVB (R6)+,COUNT ;TRY AUTOINC ON R6
6803 027430 001401      CMP #2,R6      ;VERIFY AUTO INC BY 2
6804      BEQ SPAU1      ;BRANCH IF GOOD
6805 027432 104001      ;BAD AUTO-INC
6806 027434 005006      SPAU1:  CLR R6      ;CPU ERROR
6807 027436 112667 153416      MOVB (R6)+,COUNT ;CLEAR R6
6808 027442 112667 153412      MOVB (R6)+,COUNT ;DOUBLE BYTE AUTO-INC
6809 027446 022706 000004      CMP #4,R6      ;VERIFY RESULT
6810 027452 001401      BEQ SPAU2      ;BRANCH IF GOOD
6811 027454 104001      ERROR +1      ;CPU ERROR
6812      ;BAD DOUBLE AUTO-INC
6813 027456 012706 001000      SPAU2:  MOV #STBOT,R6      ;LOAD R6
6814 027462 114667 153372      MOVB -(R6),COUNT ;TEST AUTO-DEC
6815 027466 022706 000776      CMP #776,R6      ;VERIFY RESULT
6816 027472 001401      BEQ SPAU3      ;BRANCH IF GOOD
6817 027474 104001      ERROR +1      ;CPU ERROR
6818
6819 027476 012706 001000      SPAU3:  MOV #STBOT,R6      ;LOAD R6
6820 027502 114667 153352      MOVB -(R6),COUNT ;TEST AUTO-DEC
6821 027506 114667 153346      MOVB -(R6),COUNT ;TEST AUTO-DEC
6822 027512 022706 000774      CMP #774,R6      ;VERIFY RESULT
6823 027516 001401      BEQ SPAU4      ;BRANCH IF GOOD
6824 027520 104001      ERROR +1      ;CPU ERROR
6825
6826 027522 005006      SPAU4:  CLR R6      ;TEST AUTO-INC ON SOP
6827 027524 105726      TSTB (R6)+      ;TEST AUTO-INC
6828 027526 020627 000002      CMP R6,#2
6829 027532 001401      BEQ SPAU5      ;BRANCH IF GOOD
6830 027534 104001      ERROR +1      ;CPU ERROR
6831
6832 027536 012706 001000      SPAU5:  MOV #STBOT,R6      ;LOAD R6
6833 027542 105746      TSTB -(R6)      ;TEST AUTO-DEC
6834 027544 022706 000776      CMP #776,R6      ;VERIFY RESULT
6835 027550 001401      BEQ SPAU6      ;BRANCH IF GOOD
6836 027552 104001      ERROR +1      ;CPU ERROR
6837
6838 027554 012706 001000      SPAU6:  MOV #STBOT,R6
6839
6840      ;
6841 027560      MTRY:
6842
6843      ;
6844 027560 005067 150202      ; VERIFY YELLOW ZONE TRAP ON AUTO DEC OF R6
6845 027564 012706 000150      CLR CPEREG      ;INIT CPU ERROR REGISTER
6846      MOV #150,R6      ;LOAD R6 WITH A VALUE THAT WILL
6847 027570 016767 150210 153164      MOV 4,SI 0C00      ;CAUSE A YELLOW STACK TRAP(IE, +400)
                          ;SAVE VECTOR

```

```

6848 027576 012767 027634 150200      MOV      #MTRYA,4      ;SETUP THE STACK OVERFLOW TRAP POINTER
6849 027604 016701 150336      MOV      146,R1      ;SAVE VECTOR
6850 027610 016702 150330      MOV      144,R2      ;SAVE VECTOR
6851 027614 016703 150322      MOV      142,R3      ;SAVE VECTOR
6852 027620 005067 150322      CLR      146      ;JUST AS A PRECAUTION
6853 027624 005046      CLR      -(R6)      ;CAUSE A STACK OVERFLOW TRAP
6854 027626 012706 001000      MOV      #STBOT,R6    ;RESTORE R6 FOR ERROR CALL
6855 027632 104001      ERROR    +1      ;CPU ERROR
6856                                ;OVERFLOW TRAP FAILED
6857 027634      MTRYA:
6858 027634 022767 000010 150124      CMP      #BIT03,CPEREG ;WAS CPU ERROR REG SET PROPERLY?
6859 027642 001003      BNE      1$      ;GO TO ERROR IF NOT
6860 027644 020627 000142      CMP      R6,#142      ;VERIFY CORRECT DECREMENT OF R6
6861 027650 001401      BEQ      MTRYB      ;BRANCH IF GOOD
6862 027652 104001      1$:      ERROR    +1      ;CPU ERROR
6863                                ;R6 IMPROPERLY DECREMENTED
6864                                ;OR CPU ERROR REGISTER NOT CORRECT
6865 027654      MTRYB:
6866 027654 005067 150106      CLR      CPEREG      ;CLEAR THE CPU ERROR REGISTER
6867 027660 016767 153076 150116      MOV      SLOC00,4      ;RESTORE VECTOR
6868 027666 010167 150254      MOV      R1,146      ;RESTORE VECTORS
6869 027672 010267 150246      MOV      R2,144      ;
6870 027676 010367 150240      MOV      R3,142      ;
6871 027702 012706 001000      MOV      #STBOT,R6
6872
6873
6874
6875 027706      MTRYM:
6876
6877      ;      TEST STACK OVERFLOW TRAPS IN VARIOUS MODES
6878 027706 005067 150054      CLR      CPEREG      ;CLEAR CPU ERROR REGISTER
6879 027712 012706 000400      MOV      #400,R6      ;SETUP OVERFLOW R6 DATA
6880 027716 016767 150062 153036      MOV      4,SLOC00      ;SAVE VECTOR
6881 027724 012767 027746 150052      MOV      #TRYMA,4
6882 027732 005067 150440      CLR      376      ;JUST AS A PRECAUTION
6883 027736 005046      CLR      -(R6)      ;CAUSE OVERFLOW TRAP
6884 027740 012706 001000      MOV      #STBOT,R6    ;RESTORE R6 FOR ERROR CALL
6885 027744 104001      ERROR    +1      ;CPU ERROR
6886                                ;NO OVERFLOW TRAP
6887 027746      TRYMA:
6888 027746 005067 150014      CLR      CPEREG      ;CLEAR CPU ERROR REGISTER
6889 027752 012705 001000      MOV      #1000,R5      ;SETUP R5 DATA
6890 027756 012706 000400      MOV      #400,R6      ;SETUP OVERFLOW R6 DATA
6891 027762 012767 030000 150014      MOV      #TRYMB,4
6892 027770 064645      ADD      (R6),-(R5)      ;CAUSE OVERFLOW TRAP
6893 027772 012706 001000      MOV      #STBOT,R6    ;RESTORE R6 FOR ERROR CALL
6894 027776 104001      ERROR    +1      ;CPU ERROR
6895                                ;NO OVERFLOW TRAP
6896      TRYMB:
6897 030000 005067 147762      CLR      CPEREG      ;CLEAR CPU ERROR REGISTER
6898 030004 012706 000150      MOV      #150,R6      ;SETUP OVERFLOW R6 DATA
6899 030010 012767 030026 147766      MOV      #TRYMC,4
6900 030016 044546      BIC      (R5),-(R6)      ;CAUSE OVERFLOW TRAP
6901 030020 012706 001000      MOV      #STBOT,R6    ;RESTORE R6 FOR ERROR CALL
6902 030024 104001      ERROR    +1      ;CPU ERROR
6903                                ;NO OVERFLOW TRAP

```

K10

COKDABO KDJ11-B CLUSTER MACY11 30(1046)
COKDAB.P11 05-APR-84 16:4505 APR-84 16:45 PAGE 128
***** DOUBLE OPERAND TESTS *****

SEQ 0127

```

6904 030026 005067 147734 TRYMC: CLR CPEREG ;CLEAR CPU ERROR REGISTER
6905 030032 016767 152724 147744 MOV SLOC00,4 ;RESTORE VECTOR
6906 030040 012706 001000 MOV #STBOT,R6
6907
6908
6909
6910 030044 ; MILLO:
6911
6912 ; TEST STACK OVERFLOW ON ILLEGAL INST TRAP
6913 030044 005067 147716 CLR CPEREG ;CLEAR CPU ERROR REGISTER
6914 030050 012706 000400 MOV #400,R6 ;SETUP FOR OVERFLOW TRAP
6915 030054 016767 147730 152700 MOV 10,SLOC00 ;SAVE VECTOR
6916 030062 012767 030110 147720 MOV #MILLOA,10 ;SETUP ILLEGAL TRAP VECTOR
6917 030070 016767 147710 152666 MOV 4,SLOC01 ;SAVE VECTOR
6918 030076 012767 030116 147700 MOV #MILLOB,4 ;SETUP OVERFLOW TRAP VECTOR
6919 030104 000077 77 ;UNUSED INSTRUCTION TRAP
6920 030106 000240 NOP
6921 030110 012706 001000 MILLOA: MOV #STBOT,R6 ;RESTORE R6 FOR ERROR CALL
6922 030114 104001 ERROR +1 ;CPU ERROR
6923 ;UNUSED INSTRUCTION TRAP
6924 030116
6925 030116 016767 152642 147660 MILLOB: MOV SLOC01,4 ;RESTORE VECTOR
6926 030124 016767 152632 147656 MOV SLOC00,10 ;RESTORE VECTOR
6927 030132 005067 147630 CLR CPEREG ;CLEAR CPU ERROR REGISTER
6928 030136 012706 001000 MOV #STBOT,R6 ;RESTORE R6
6929
6930
6931
6932 ; MIOTO:
6933 030142
6934
6935 ; TEST STACK OVERFLOW ON IOT TRAP
6936 030142 005067 147620 CLR CPEREG ;CLEAR CPU ERROR REGISTER
6937 030146 012706 000400 MOV #400,R6 ;SETUP STACK FOR OVERFLOW
6938 030152 016767 147642 152602 MOV 20,SLOC00 ;SAVE OLD IOT VECTOR
6939 030160 012767 030206 147632 MOV #IOTGA,20 ;SETUP ERROR ACTION ON IOT
6940 030166 016767 147612 152570 MOV 4,SLOC01 ;SAVE VECTOR
6941 030174 012767 030214 147602 MOV #IOTOB,4 ;SETUP CORRECT TRAP VECTOR FOR
6942 ;OVERFLOW
6943 030202 000004 IOT ;TEST INSTRUCTION
6944 030204 000240 NOP
6945 030206 012706 001000 IOTGA: MOV #STBOT,R6 ;RESTORE R6 FOR ERROR CALL
6946 030212 104001 ERROR +1 ;CPU ERROR
6947 ;FAILURE OF STACK OVERFLOW
6948 030214
6949 030214 005067 147546 IOTOB: CLR CPEREG ;CLEAR CPU ERROR REGISTER
6950 030220 012706 001000 MOV #STBOT,R6
6951 030224 016767 152534 147552 MOV SLOC01,4 ;RESTORE VECTOR
6952 030232 016767 152524 147560 MOV SLOC00,20 ;RESTORE TRAP VECTOR
6953
6954
6955 ;
6956 030240 ; MEMTO:
6957
6958 ; TEST STACK OVERFLOW ON LMT TRAP
6959 030240 005067 147522 CLR CPEREG ;CLEAR CPU ERROR REGISTER

```

COKDABO KDJ11-B CLUSTER MACY11 30(1046)
COKDAB.P11 05-APR-84 16:45

05-APR-84 16:45 PAGE 129
***** DOUBLE OPERAND TESTS *****

SEQ 0128

6960	030244	012706	000400		MOV	#400,R6		; SETUP STACK FOR OVERFLOW
6961	030250	016767	147554	152504	MOV	30,SLOC00		; SAVE OLD EMT VECTOR
6962	030256	012767	030304	147544	MOV	#EMTOA,30		; SETUP ERROR ACTION ON EMT
6963	030264	016767	147514	152472	MOV	4,SLOC01		; SAVE VECTOR
6964	030272	012767	030312	147504	MOV	#EMTOB,4		; SETUP CORRECT TRAP VECTOR FOR
6965								; OVERFLOW
6966	030300	104000			EMT			; TEST INSTRUCTION
6967	030302	000240			NOP			
6968	030304	012706	001000		EMTOA:	MOV	#STBOT,R6	; RESTORE R6 FOR ERROR CALL
6969	030310	104001			ERROR	+1		; CPU ERROR
6970								; FAILURE OF STACK OVERFLOW
6971	030312				EMTOB:			
6972	030312	016767	152444	147510	MOV	SLOC00,30		; RESTORE TRAP VECTOR
6973	030320	016767	152440	147456	MOV	SLOC01,4		; RESTORE VECTOR
6974	030326	005067	147434		CLR	CPEREG		; CLEAR CPU ERROR REGISTER
6975	030332	012706	001000		MOV	#STBOT,R6		
6976								
6977	030336				MTRPO:			
6978								
6979								
6980	030336	005067	147424					
6981	030342	012706	000400					
6982	030346	016767	147462	152406	CLR	CPEREG		; CLEAR CPU ERROR REGISTER
6983	030354	012767	030402	147452	MOV	#400,R6		; SETUP STACK FOR OVERFLOW
6984	030362	016767	147416	152374	MOV	34,SLOC00		; SAVE OLD TRP VECTOR
6985	030370	012767	030410	147406	MOV	#TRPOA,34		; SETUP ERROR ACTION ON TRP
6986					MOV	4,SLOC01		; SAVE VECTOR
6987	030376	104400			MOV	#TRPOB,4		; SETUP CORRECT TRAP VECTOR FOR
6988	030400	000240						; OVERFLOW
6989	030402	012706	001000		TRAP			; TEST INSTRUCTION
6990	030406	104001			NOP			
6991					TRPOA:	MOV	#STBOT,R6	; RESTORE R6 FOR ERROR CALL
6992	030410				ERROR	+1		; CPU ERROR
6993	030410	016767	152346	147416				; FAILURE OF STACK OVERFLOW
6994	030416	016767	152342	147360	TRPOB:			
6995	030424	005067	147336		MOV	SLOC00,34		; RESTORE TRAP VECTOR
6996	030430	012706	001000		MOV	SLOC01,4		; RESTORE VECTOR
6997					CLR	CPEREG		; CLEAR CPU ERROR REGISTER
6998					MOV	#STBOT,R6		
6999								
7000	030434							
7001					MBPTO:			
7002								
7003	030434	005067	147326					
7004	030440	012706	000400					
7005	030444	016767	147344	152310				
7006	030452	012767	030500	147334	CLR	CPEREG		; CLEAR CPU ERROR REGISTER
7007	030460	016767	147320	152276	MOV	#400,R6		; SETUP STACK FOR OVERFLOW
7008	030466	012767	030506	147310	MOV	14,SLOC00		; SAVE OLD BPT VECTOR
7009					MOV	#BPTOA,14		; SETUP ERROR ACTION ON BPT
7010	030474	000003			MOV	4,SLOC01		; SAVE VECTOR
7011	030476	000240			MOV	#BPTOB,4		; SETUP CORRECT TRAP VECTOR FOR
7012	030500	012706	001000					; OVERFLOW
7013	030504	104001			BPT			; TEST INSTRUCTION
7014					NOP			
7015	030506				BPTOA:	MOV	#STBOT,R6	; RESTORE R6 FOR ERROR CALL
					ERROR	+1		; CPU ERROR
								; FAILURE OF STACK OVERFLOW
					BPTOB:			

M10

COKDABO KDJ11 B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 130
 COKDAB.P11 05-APR-84 16:45 ***** DOUBLE OPERAND TESTS *****

SEQ 0129

```

7016 030506 005067 147254          CLR      CPEREG          ;CLEAR CPU ERROR REGISTER
7017 030512 016767 152244 147274    MOV      SLOC00,14        ;RESTORE TRAP VECTOR
7018 030520 016767 152240 147256    MOV      SLOC01,4        ;RESTORE VECTOR
7019 030526 012706 001000          MOV      #STBOT,R6
7020
7021
7022
7023 030532          ;
7024          ; MILAO:
7025          ;
7026 030532 005067 147230          ; TEST STACK OVERFLOW AND ILLEGAL JMP INSTRUCTION
7027 030536 012706 000400          CLR      CPEREG          ;CLEAR CPU ERROR REGISTER
7028 030542 016767 147242 152212    MOV      #400,R6        ;SETUP STACK FOR OVERFLOW
7029 030550 012767 030600 147232    MOV      10,SLOC00      ;SAVE OLD ILLEGAL INST. VECTOR
7030 030556 016767 147222 152200    MOV      #ILAOA,10     ;SETUP ERROR ACTION ILLEGAL OPCODE
7031 030564 012767 030606 147212    MOV      4,SLOC01      ;SAVE VECTOR
7032          MOV      #ILBOB,4    ;SETUP CORRECT TRAP VECTOR FOR
7033          ;OVERFLOW
7034 030572 005001          CLR      R1
7035 030574 000101          JMP      R1          ; TEST INSTRUCTION
7036 030576 000240          NOP
7037 030600 012706 001000  ILAOA: MOV      #STBOT,R6        ;RESTORE R6 FOR ERROR CALL
7038 030604 104001          ERROR    +1          ;CPU ERROR
7039          ;FAILURE OF STACK OVERFLOW
7040 030606          ILBOB:
7041 030614 016767 152152 147170    MOV      SLOC01,4        ;RESTORE VECTOR
7042 030614 016767 152142 147166    MOV      SLOC00,10     ;RESTORE TRAP VECTOR
7043 030622 005067 147140          CLR      CPEREG          ;CLEAR CPU ERROR REGISTER
7044 030626 012706 001000          MOV      #STBOT,R6
7045
7046
7047 030632          ;
7048          ; MILLBO:
7049          ;
7050 030632 012706 000400          ; TEST STACK OVERFLOW ON ILLEGAL JSR INST
7051 030636 016767 147146 152116    MOV      #400,R6        ;SETUP STACK FOR OVERFLOW
7052 030644 012767 030674 147136    MOV      10,SLOC00      ;SAVE OLD VECTOR
7053 030652 016767 147126 152104    MOV      #ILLBOA,10     ;SETUP ERROR ACTION ON ILL. OPCODE
7054 030660 012767 030702 147116    MOV      4,SLOC01      ;SAVE VECTOR
7055          MOV      #ILLBOB,4    ;SETUP CORRECT TRAP VECTOR FOR
7056          ;OVERFLOW
7057 030666 005001          CLR      R1
7058 030670 004501          JSR      R5,R1          ; TEST INSTRUCTION
7059 030672 000240          NOP
7060 030674 012706 001000  ILLBOA: MOV      #STBOT,R6        ;RESTORE R6 FOR ERROR CALL
7061 030700 104001          ERROR    +1          ;CPU ERROR
7062          ;FAILURE OF STACK OVERFLOW
7063 030702          ILLBOB:
7064 030702 016767 152056 147074    MOV      SLOC01,4        ;RESTORE VECTOR
7065 030710 016767 152046 147072    MOV      SLOC00,10     ;RESTORE TRAP VECTOR
7066 030716 012706 001000          MOV      #STBOT,R6
7067
7068
7069 030722          ;
7070          ; MSTO:
7071          ;
          ; TEST FOR FALSE STACK OVERFLOW

```

N10

COKDABO KDJ11-B CLUSTER MAC:11 30(1046) 05-APR-84 16:45 PAGE 131
 COKDAB.P11 05-APR-84 16:45 ***** DOUBLE OPERAND TESTS *****

SEQ 0130

7072	030722	016767	147056	152032	MOV	4, SLOC00	;SAVE VECTOR
7073	030730	012767	030776	147046	MOV	#MSTOE, 4	;ANTICIPATE OVERFLOW ERROR
7074	030736	012706	001002		MOV	#1002, R6	;SETUP LEGAL R6
7075	030742	005746			TST	-(R6)	;TRY TO CAUSE STACK OVERFLOW
7076	030744	012706	002002		MOV	#2002, R6	;SETUP LEGAL R6
7077	030750	005746			TST	-(R6)	;TRY TO CAUSE STACK OVERFLOW
7078	030752	012706	004002		MOV	#4002, R6	;SETUP LEGAL R6
7079	030756	005746			TST	-(R6)	;TRY TO CAUSE STACK OVERFLOW
7080	030760	012706	010002		MOV	#10002, R6	;SETUP LEGAL R6
7081	030764	005746			TST	-(R6)	;TRY TO CAUSE STACK OVERFLOW
7082	030766	012706	100402		MOV	#100402, R6	;SETUP LEGAL R6
7083	030772	005746			TST	-(R6)	;TRY TO CAUSE STACK OVERFLOW
7084	030774	000403			RR	MSTOE	;EXIT MODULE
7085	030776	012706	001000		MSTOE: MOV	#STBOT, R6	;RESTORE R6 FOR ERROR CALL
7086	031002	104001			ERROR	+1	;CPU ERROR
7087							;STACK OVERFLOW ERROR
7088	031004	016767	151752	146772	MSTOE: MOV	SLOC00, 4	;RESTORE VECTOR
7089	031012	012706	001000		MOV	#STBOT, R6	
7090							
7091							
7092							
7093	031016				MTT:		
7094							
7095						TEST T-BIT TRAPS	
7096	031016	012706	001000		MOV	#STBOT, R6	;SETUP STACK
7097	031022	016767	146766	151732	MOV	14, SLOC00	;SAVE OLD T-BIT VECTOR
7098	031030	012746	000020		MOV	#20, -(R6)	;PUSH T-BIT
7099	031034	012746	031052		MOV	#MTTA, -(R6)	;SETUP ERROR TRAP VECTOR
7100	031040	012767	031054	146746	MOV	#MTTB, 14	;SETUP NEW T-BIT VECTOR
7101	031046	000002			RTI		;CAUSE A T BIT SET IN PSW
7102	031050	104001			ERROR	+1	;CPU ERROR
7103							;SHOULD NEVER BE EXECUTED
7104	031052	104001			MTTA: ERROR	+1	;CPU ERROR
7105							;DIDNT TAKE CORRECT TRAP
7106	031054	022706	000774		MTTB: CMP	#STBOT-4, R6	;VERIFY SP DECIMENT
7107	031060	001401			BEQ	MTTD	;BRANCH IF GOOD
7108	031062	104001			ERROR	+1	;CPU ERROR
7109							;BAD SP
7110	031064	021627	031052		MTTD: CMP	(R6), #MTTA	;VERIFY PC SAVED ON STACK
7111	031070	001401			BEQ	MTTE	;BRANCH IF GOOD
7112	031072	104001			ERROR	+1	;CPU ERROR
7113							;INCORRECT PC ON STACK
7114	031074				MTTE:		
7115	031074	016767	151662	146712	MOV	SLOC00, 14	;RESTORE VECTOR 14
7116							
7117	031102	012706	001000		MOV	#STBOT, R6	
7118							
7119							
7120	031106				MTTS:		
7121							
7122						TEST T-BIT TRAPS WITH RTI	
7123	031106	012706	001000		MOV	#STBOT, R6	;SETUP STACK
7124	031112	016767	146676	151642	MOV	14, SLOC00	;SAVE OLD T-BIT VECTOR
7125	031120	012746	000020		MOV	#20, -(R6)	;PUSH T-BIT
7126	031124	012746	031142		MOV	#MTTSA, -(R6)	;SETUP ERROR TRAP VECTOR
7127	031130	012767	031146	146656	MOV	#MTTSB, 14	;SETUP NEW T-BIT VECTOR

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


```

7240 031540 MTP:
7241
7242
7243 031540 012706 001000 ; TEST TRAP INST
7244 031544 016767 146264 151210 MOV #STBOT,R6 ; SETUP STACK
7245 031552 012767 031572 146254 MOV 34,SLOC00 ; SAVE OLD VECTOR
7246 031560 005067 146212 MOV #MTPB,34 ; SETUP NEW TRAP VECTOR
7247 031564 000257 CLR PS ; CLEAR PRIORITY ABND COND C
7248 031566 104400 CCC
7249 031570 104001 TRAP
7250 MTPR: ERROR +1 ; CPU ERROR
7251 031572 022706 000774 ; DIDNT TAKE CORRECT TRAP
7252 031576 001401 MTPB: CMP #STBOT-4,R6 ; VERIFY SP DECRIMENT
7253 031600 104001 BEQ MTPQ ; BRANCH IF GOOD
7254 ERROR +1 ; CPU ERROR
7255 031602 021627 031570 MTPQ: CMP (R6),#MTPR ; BAD PC ON STACK
7256 031606 001401 BEQ MTPF ; VERIFY PROPER PC ON STACK
7257 031610 104001 ERROR +1 ; BRANCH IF GOOD
7258 ; CPU ERROR
7259 031612 MTPF: ; INCORRECT PC ON STACK
7260 031612 016767 151144 146214 MOV SLOC00,34 ; RESTORE VECTOR
7261 031620 012706 001000 MOV #STBOT,R6
7262
7263
7264
7265 031624 MTPD:
7266
7267
7268 ; TEST OLD STATUS SAVED ON TRAP
7269 031630 012706 001000 MOV #STBOT,R6 ; SETUP STACK
7270 031636 012767 031656 146170 MOV 34,SLOC00 ; SAVE OLD VECTOR
7271 031644 005067 146126 MOV #MTPOB,34 ; SETUP NEW TRAP VECTOR
7272 031650 000257 CLR PS ; CLEAR PRIORITY AND COND C
7273 031652 104400 CCC
7274 031654 104001 TRAP
7275 MTPOA: ERROR +1 ; CPU ERROR
7276 031656 026727 147114 000000 MTPOB: CMP STBOT-2,#0 ; DIDNT TAKE CORRECT TRAP
7277 031664 001401 BEQ MTPOC ; VERIFY PSW ON STACK
7278 031666 104001 ERROR +1 ; BRANCH IF CORRECT STATUS
7279 ; CPU ERROR
7280 031670 012706 001000 MTPOC: MOV #STBOT,R6 ; BAD STATUS ON STACK
7281 031674 012767 031716 146132 MOV #MTPOE,34 ; SETUP STACK
7282 031702 012767 000357 146066 MOV #357,PS ; SET UP TRAP VECTOR
7283 031710 000277 SCC ; SET PRIORITY
7284 031712 104400 TRAP ; SET CONDITION CODES
7285 031714 104001 MTPOD: ERROR +1 ; ISSUE TRAP
7286 ; CPU ERROR
7287 031716 026727 147054 000357 MTPOE: CMP STBOT-2,#357 ; DIDNT TAKE CORRECT TRAP
7288 031724 001401 BEQ MTPOF ; VERIFY OLD PSW ON STACK
7289 031726 104001 ERROR +1 ; BRANCH IF GOOD
7290 ; CPU ERROR
7291 031730 MTPOF: ; OLD PSW INCORRECT
7292 031730 016767 151026 146076 MOV SLOC00,34 ; RESTORE TRAP VECTOR
7293 031736 012706 001000 MOV #STBOT,R6
7294
7295

```

```

7296
7297 031742      ; MTPA:
7298
7299      ; TEST ALL TRAP OPCODES - SELF MODIFYING
7300 031742 005003      CLR      R3              ;SETUP REGISTER TO INDICATE OPCODE
7301 031744 012706 001000      MOV      @STBOT,R6      ;SETUP STACK
7302 031750 016767 146060 151004      MOV      34,SLOC00      ;SAVE OLD VECTOR
7303 031756 016767 146022 151000      MOV      4,SLOC01      ;SAVE IN CASE OF HALT
7304 031764 012767 032014 146012      MOV      @MTPAH,4      ;SETUP HALT TRAP
7305 031772 012767 032016 146034      MOV      @MTPAA,34      ;SETUP NEW TRAP VECTOR
7306 032000 000167 000012      JMP      MTPAA      ;GO INTO LOOPING CODE
7307
7308 032004 000000      ; MTPAL: HALT              ;SET TO A ZERO
7309 032006 104001      ERROR      +1      ;CPU ERROR
7310
7311      ; TRAP INSTRUCTION FAILED TO TRAP
7312 032010 000167 000002      JMP      MTPAA      ;EXAMINE OPCCODE AT LOCATION MTPAL:
7313      ; ATTEMPT TO GO ON
7314 032014 104001      ; MTPAH: ERROR      +1      ;CPU ERROR
7315      ; ERROR, EITHER CANT MODIFY LOCATION MTPAL
7316      ; OR TRAP INSTRUCTION FAILED
7317 032016
7318 032016 005203      MTPAA:      INC      R3              ;GET NEXT OPCODE
7319
7320 032020 012706 001000      MOV      @STBOT,R6      ;RESTORE STACK
7321 032024 020327 000400      CMP      R3,#400      ;SEE IF LAST OPCODE
7322 032030 001406      BEQ      MTPAE      ;BRANCH IF DONE
7323 032032 012767 104400 177744      MOV      @104400,MTPAL      ;TRAP OPCODE INTO LOCATION
7324 032040 060367 177740      ADD      R3,MTPAL      ;FORM TEST OPCODE
7325 032044 000757      BR      MTPAL      ;EXECUTE TEST
7326 032046      MTPAE:
7327
7328 032046 016767 150710 145760      MOV      SLOC00,34
7329 032054 016767 150704 145722      MOV      SLOC01,4      ;RESTORE VECTORS
7330
7331 032062 012706 001000      MOV      @STBOT,R6
7332
7333      ;
7334      ; MIOT:
7335 032066
7336
7337      ; TEST IOT TRAP
7338 032066 012706 001000      MOV      @STBOT,R6      ;SETUP STACK
7339 032072 016767 145722 150662      MOV      20,SLOC00      ;SAVE OLD VECTOR
7340 032100 012767 032112 145712      MOV      @MIOTB,20      ;SETUP NEW IOT VECTOR
7341 032106 000004      IOT
7342 032110 104001      MIOTA: ERROR      +1      ;CPU ERROR
7343      ; DIDNT TAKE CORRECT TRAP
7344 032112 022706 000774      MIOTB: CMP      @STBOT-4,R6      ;VERIFY SP DECRIMENT
7345 032116 001401      BEQ      MIOTD      ;BRANCH IF GOOD
7346 032120 104001      ERROR      +1      ;CPU ERROR
7347      ; BAD PC ON STACK
7348 032122 021627 032110      MIOTD: CMP      (R6),@MIOTA      ;VERIFY PROPER PC ON STACK
7349 032126 001401      BEQ      MIOTF      ;BRANCH IF GOOD
7350 032130 104001      ERROR      +1      ;CPU ERROR
7351      ; INCORRECT PC ON STACK

```

COKDABO KDJ11 B CLUSTER MACY11 30(1046)
COKDAB.P11 05-APR 84 16:45

05-APR-84 16:45 PAGE 136
***** DOUBLE OPERAND TESTS *****

SEQ 0135

```

7352 032132 016767 150624 145660 MIOTF: MOV SLOC00,20 ;RESTORE VECTOR
7353 032140 012706 001000 MOV #STBOT,R6
7354
7355 032144 MITO:
7356
7357 ; TEST OLD STATUS ON IOT TRAP
7358 032144 012706 001000 MOV #STBOT,R6 ;SETUP STACK
7359 032150 016767 145644 150604 MOV 20,SLOC00 ;SAVE OLD VECTOR
7360 032156 012767 032176 145634 MOV #MITOB,20 ;SETUP NEW IOT VECTOR
7361 032164 005067 145606 CLR PS ;CLEAR PRIORITY AND COND C
7362 032170 000257 CCC
7363 032172 000004 IOT
7364 032174 104001 MITOA: ERROR +1 ;CPU ERROR
7365 ;DIDNT TAKE CORRECT TRAP
7366 032176 026727 146574 000000 MITOB: CMP STBOT-2,#0 ;VERIFY PSW ON STACK
7367 032204 001401 BEQ MITOC ;BRANCH IF CORRECT STATUS
7368 032206 104001 ERROR +1 ;CPU ERROR
7369 ;BAD STATUS ON STACK
7370 032210 012706 001000 MITOC: MOV #STBOT,R6 ;SETUP STACK
7371 032214 012767 032236 145576 MOV #MITOF,20 ;SET UP TRAP VECTOR
7372 032222 012767 000357 145546 MOV #357,PS ;SET PRIORITY
7373 032230 000277 SCC ;SET CONDITION CODES
7374 032232 000004 IOT
7375 032234 104001 MITOD: ERROR +1 ;CPU ERROR
7376 ;DIDNT TAKE CORRECT TRAP
7377 032236 026727 146534 000357 MITOE: CMP STBOT-2,#357 ;VERIFY OLD PSW ON STACK
7378 032244 001401 BEQ MITOF ;BRANCH IF GOOD
7379 032246 104001 ERROR +1 ;CPU ERROR
7380 ;OLD PSW INCORRECT
7381 032250 MITOF:
7382 032250 016767 150506 145542 MOV SLOC00,20 ;RESTORE VECTOR
7383 032256 012706 001000 MOV #STBOT,R6
7384
7385 ;
7386 032262 MET:
7387
7388 ; TEST EMULATOR TRAP INSTRUCTION (EMT)
7389 032262 012706 001000 MOV #STBOT,R6 ;SETUP STACK
7390 032266 016767 145536 150466 MOV 30,SLOC00 ;SAVE OLD VECTOR
7391 032274 012767 032330 145526 MOV #METB,30 ;SETUP NEW EMT VECTOR
7392 032302 016767 145526 150454 MOV 34,SLOC01 ;SAVE TRAP VECTOR
7393 032310 012767 134132 145516 MOV #ERROR,34 ;SET UP TO HANDLE EMT ERROR
7394 032316 104000 EMT
7395 032320 104400 META: TRAP ;TRAP ON ERROR
7396 032322 001057 .WORD 559,
7397 032324 000001 .WORD 1 ;CPUERR
7398 032326 000001 .WORD 1 ;ERRTN
7399 ;DIDNT TAKE CORRECT TRAP
7400 032330 022706 000774 METB: CMP #STBOT-4,R6 ;VERIFY SP DECRIMENT
7401 032334 001401 BEQ METD ;BRANCH IF GOOD
7402 032336 104001 ERROR +1 ;CPU ERROR
7403 ;BAD PC ON STACK
7404 032340 021627 032320 METD: CMP (R6),#META ;VERIFY PROPER PC ON STACK
7405 032344 001401 BEQ METF ;BRANCH IF GOOD
7406 032346 104001 ERROR +1 ;CPU ERROR
7407 ;INCORRECT PC ON STACK

```

COKDABO KDJ11-B CLUSTER MAC111 30(1046) 05-APR-84 16:45 PAGE 137
 COKDAB.P11 05-APR-84 16:45 ***** DOUBLE OPERAND TESTS *****

SEQ 0136

7408	032350	016767	150410	145456	METF:	MOV	SLOC01,34		;RESTORE VECTOR
7409	032356	016767	150400	145444		MOV	SLOC00,30		;RESTORE VECTOR
7410	032364	012706	001000			MOV	#STBOT,R6		
7411									
7412									
7413	032370								
7414									
7415									
7416	032370	012706	001000						
7417	032374	016767	145430	150360		MOV	#STBOT,R6		;SETUP STACK
7418	032402	012767	032444	145420		MOV	30,SLOC00		;SAVE OLD VECTOR
7419	032410	016767	145420	150346		MOV	#METOB,30		;SETUP NEW EMT VECTOR
7420	032416	012767	134132	145410		MOV	34,SLOC01		;SAVE TRAP VECTOR
7421	032424	005067	145346			MOV	#\$ERROR,34		;SET UP TRAP VECTOR
7422	032430	000257				CLR	PS		;CLEAR PRIORITY AND COND C
7423	032432	104000				CCC			
7424	032434	104400				EMT			
7425	032436	001062				TRAP			
7426	032440	000001				.WORD	562.		
7427	032442	000001				.WORD	1		;CPUERR
7428						.WORD	1		;ERRTN
7429	032444	026727	146326	000000	METOB:	CMP	STBOT-2,#0		;DIDNT TAKE CORRECT TRAP
7430	032452	001401				BEQ	METOC		;VERIFY PSW ON STACK
7431	032454	104001				ERROR	+1		;BRANCH IF CORRECT STATUS
7432									;CPU ERROR
7433	032456	012706	001000		METOC:	MOV	#STBOT,R6		;BAD STATUS ON STACK
7434	032462	012767	032512	145340		MOV	#METOE,30		;SETUP STACK
7435	032470	012767	000357	145300		MOV	#357,PS		;SET UP TRAP VECTOR
7436	032476	000277				SCC			;SET PRIORITY
7437	032500	104000				EMT			;SET CONDITION CODES
7438	032502	104400			METOD:	TRAP			
7439	032504	001064				.WORD	564.		
7440	032506	000001				.WORD	1		;CPUERR
7441	032510	000001				.WORD	1		;ERRTN
7442									;DIDNT TAKE CORRECT TRAP
7443	032512	026727	146260	000357	METOE:	CMP	STBOT-2,#357		;VERIFY OLD PSW ON STACK
7444	032520	001401				BEQ	METOF		;BRANCH IF GOOD
7445	032522	104001				ERROR	+1		;CPU ERROR
7446									;OLD PSW INCORRECT
7447	032524				METOF:				
7448	032524	016767	150234	145302		MOV	SLOC01,34		;RESTORE VECTOR
7449	032532	016767	150224	145270		MOV	SLOC00,30		;RESTORE VECTOR
7450	032540	012706	001000			MOV	#STBOT,R6		
7451									
7452									
7453									
7454	032544								
7455									
7456									
7457	032544	012706	001000						
7458	032550	016767	145240	150204		MOV	#STBOT,R6		;SETUP STACK
7459	032556	012767	032570	145230		MOV	14,SLOC00		;SAVE OLD VECTOR
7460	032564	000003				MOV	#MBTB,14		;SETUP NEW BPT VECTOR
7461	032566	104001				BPT			
7462					MBTA:	ERROR	+1		;CPU ERROR
7463	032570	022706	000774		MBTB:	CMP	#STBOT-4,R6		;DIDNT TAKE CORRECT TRAP
									;VERIFY SP DECREMENT

7464	032574	001401			BEQ	MBTD		;BRANCH IF GOOD
7465	032576	104001			ERROR	+1		;CPU ERROR
7466								;BAD PC ON STACK
7467	032600	021627	032568		MBTD:	CMP	(R6),#MBTA	;VERIFY PROPER PC ON STACK
7468	032604	001401			BEQ	MBTF		;BRANCH IF GOOD
7469	032606	104001			ERROR	+1		;CPU ERROR
7470								;INCORRECT PC ON STACK
7471	032610	016767	150146	145176	MBTF:	MOV	SLOC00,14	;RESTORE VECTOR
7472	032616	012706	001000			MOV	#STBOT,R6	
7473								
7474					:			
7475					:			
7476	032622				MBTO:			
7477								
7478					:		TEST OLD STATUS ON BPT TRAP	
7479	032622	012706	001000			MOV	#STBOT,R6	;SETUP STACK
7480	032626	016767	145162	150126		MOV	14,SLOC00	;SAVE OLD VECTOR
7481	032634	012767	032654	145152		MOV	#MBTOB,14	;SETUP NEW BPT VECTOR
7482	032642	005067	145130			CLR	PS	;CLEAR PRIORITY AND COND C
7483	032646	000257				CCC		
7484	032650	000003				BPT		
7485	032652	104001			MBTOA:	ERROR	+1	;CPU ERROR
7486								;DIDNT TAKE CORRECT TRAP
7487	032654	026727	146116	000000	MBTOB:	CMP	STBOT-2,#0	;VERIFY PSW ON STACK
7488	032662	001401				BEQ	MBTOC	;BRANCH IF CORRECT STATUS
7489	032664	104001				ERROR	+1	;CPU ERROR
7490								;BAD STATUS ON STACK
7491	032666	012706	001000		MBTOC:	MOV	#STBOT,R6	;SETUP STACK
7492	032672	012767	032714	145114		MOV	#MBTOE,14	;SET UP TRAP VECTOR
7493	032700	012767	000357	145070		MOV	#357,PS	;SET PRIORITY
7494	032706	000277				SCC		;SET CONDITION CODES
7495	032710	000003				BPT		
7496	032712	104001			MBTOD:	ERROR	+1	;CPU ERROR
7497								;DIDNT TAKE CORRECT TRAP
7498	032714	026727	146056	000357	MBTOE:	CMP	STBOT-2,#357	;VERIFY OLD PSW ON STACK
7499	032722	001401				BEQ	MBTOF	;BRANCH IF GOOD
7500	032724	104001				ERROR	+1	;CPU ERROR
7501								;OLD PSW INCORRECT
7502	032726				MBTOF:			
7503	032726	016767	150030	145060		MOV	SLOC00,14	;RESTORE VECTOR
7504	032734	012706	001000			MOV	#STBOT,R6	
7505								
7506					:			
7507					:			
7508	032740				MIL:			
7509								
7510					:		TEST ILLEGAL JUMP INSTRUCTION TRAP	
7511	032740	012706	001000			MOV	#STBOT,R6	;SETUP STACK
7512	032744	016767	145040	150010		MOV	10,SLOC00	;SAVE OLD VECTOR
7513	032752	012767	032766	145030		MOV	#MILB,10	;SETUP NEW ILLEGAL VECTOR
7514	032760	005001				CLR	R1	
7515	032762	000101				JMP	R1	**TEST INSTRUCTIO
7516	032764	104001			MILA:	ERROR	+1	;CPU ERROR
7517								;DIDNT TAKE CORRECT TRAP
7518	032766	022706	000774		MILB:	CMP	#STBOT-4,R6	;VERIFY SP DECRIMENT
7519	032772	001401				BEQ	MILD	;BRANCH IF GOOD

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

CORDBA0 KDJ11 B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 140
 CORDAB.P11 05-APR-84 16:45 ***** DOUBLE OPERAND TESTS *****

SEQ 0139

7576	033176	021627	033164		MIALLD: CMP	(R6),#MIALLA		;VERIFY PROPER PC ON STACK
7577	033202	001401			BEQ	MIALLF		;BRANCH IF GOOD
7578	033204	104001			ERROR	+1		;CPU ERROR
7579								;INCORRECT PC ON STACK
7580	033206	016767	147550	144574	MIALLF: MOV	SLOC00,10		;RESTORE VECTOR
7581	033214	012706	001000		MOV	#STBOT,R6		
7582								
7583								
7584								
7585	033220				MJSI:			
7586								
7587								
7588	033220	012706	001000			TEST OLD STATUS ON ILLEGAL JSR TRAP		
7589	033224	016767	144560	147530	MOV	#STBOT,R6		;SETUP STACK
7590	033232	012767	033254	144550	MOV	10,SLOC00		;SAVE OLD VECTOR
7591	033240	005067	144532		MOV	#MJSIB,10		;SETUP NEW VECTOR
7592	033244	000257			CLR	PS		;CLEAR PRIORITY AND COND C
7593	033246	005003			CCC			
7594	033250	004303			CLR	R3		
7595	033252	104001			JSR	R3,R3		
7596					MJSIA: ERROR	+1		;CPU ERROR
7597	033254	026727	145516	000004				;DIDNT TAKE CORRECT TRAP
7598	033262	001401			MJSIB: CMP	STBOT-2,#4		;VERIFY PSW ON STACK
7599	033264	104001			BEQ	MJSIC		;BRANCH IF CORRECT STATUS
7600					ERROR	+1		;CPU ERROR
7601	033266	012706	001000					;BAD STATUS ON STACK
7602	033272	012767	033314	144510	MJSIC: MOV	#STBOT,R6		;SETUP STACK
7603	033300	012767	000357	144470	MOV	#MJSIE,10		;SET UP TRAP VECTOR
7604	033306	000277			MOV	#357,PS		;SET PRIORITY
7605	033310	004303			SCC			;SET CONDITION CODES
7606	033312	104001			JSR	R3,R3		
7607					MJSID: ERROR	+1		;CPU ERROR
7608	033314	026727	145456	000357				;DIDNT TAKE CORRECT TRAP
7609	033322	001401			MJSIE: CMP	STBOT-2,#357		;VERIFY OLD PSW ON STACK
7610	033324	104001			BEQ	MJSIF		;BRANCH IF GOOD
7611					ERROR	+1		;CPU ERROR
7612	033326							;OLD PSW INCORRECT
7613	033326	016767	147430	144454	MJSIF: MOV	SLOC00,10		;RESTORE VECTOR
7614	033334	012706	001000		MOV	#STBOT,R6		
7615								
7616								
7617								
7618	033340							
7619								
7620	033340	005067	144422			I/O TIME OUT TEST		
7621	033344	016767	144434	147410	CLR	CPEREG		;CLEAR CPU ERROR REGISTER
7622	033352	012767	033374	144424	MOV	4,SLOC00		;SAVE VECTOR
7623	033360	012767	030000	144410	MOV	#PS,4		;SET UP VECTOR TO HANDLE NXM
7624	033366	005737	177700		MOV	#30000,PS		;INIT THE PSW TO A KNOWN STATE
7625					TST	#177700		;TRY TO ACCESS HARDWARE ADDRESS
7626								;FOR GENERAL PURPOSE REG 0. THIS
7627								;IS NOT IMPLEMENTED ON KDJ11
7628	033372	104001						;SHOULD CAUSE TIME OUT.
7629	033374	022767	000020	144364	1\$: ERROR	+1		;CPU ERROR
7630	033402	001401			2\$: CMP	#BIT04,CPEREG		;IS CPU ERROR REGISTER CORRECT?
7631	033404	104001			BEQ	3\$		
					ERROR	+1		;CPU ERROR

K11

COKDABO KDJ11-B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 141
COKDAB.P11 05-APR-84 16:45

***** DOUBLE OPERAND TESTS *****

SEQ 0140

```

7632 033406 022627 033372 3$: CMP (SP)+, #1$ ;CHECK THAT STACK CONTAINS CORRECT ADDR.
7633 033412 001401 BEQ 4$ ;
7634 033414 104001 ERROR +1 ;CPU ERROR
7635 033416 022627 030000 4$: CMP (SP)+, #30000 ;IS THE PSW OK?
7636 033422 001401 BEQ 5$ ;
7637 033424 104001 ERROR +1 ;CPU ERROR
7638 033426 005067 144334 5$: CLR CPREG ;CLEAR THE CPU ERROR REGISTER
7639 033432 016767 147324 144344 MOV SLOC00,4 ;RESTORE VECTOR
7640
7641 033440 ODDXX:
7642
7643 ; ODD ADDRESS/ILLEGAL INST FETCH TRAP TEST
7644 ;*****
7645 ;THIS PROGRAM GENERATES AN ODD ADDRESS IN THE PC. THE KDJ11 SHOULD
7646 ;TRAP THROUGH ADDR 4
7647 033440 005067 144322 CLR CPREG ;INIT THE CPU ERROR REG
7648 033444 016767 144334 147310 MOV 4, SLOC00 ;SAVE VECTOR
7649 033452 012767 033506 144324 MOV #2$, 4 ;SET UP VECTOR TO HANDLE ODD ADDR TRAP
7650 033460 016767 144322 147276 MOV 6, SLOC01 ;SAVE VECTOR
7651 033466 005067 144314 CLR 6 ;INIT VECTOR
7652 033472 012746 030000 MOV #30000, -(SP) ;PUSH A KNOWN PSW ON THE STACK
7653 033476 012746 033505 MOV #1$+1, -(SP) ;PUSH AN ODD NUMBER ON THE STACK
7654 033502 000002 RTI ;POP ODD ADDRESS OFF STACK INTO PC
7655 ;SHOULD TRAP HERE
7656 033504 104001 1$: ERROR +1 ;CPU ERROR
7657 033506 022767 000100 144252 2$: CMP #BIT06, CPREG ;IS CPU ERROR REGISTER CORRECT?
7658 033514 001401 BEQ 3$ ;
7659 033516 104001 ERROR +1 ;CPU ERROR
7660 033520 022726 033505 3$: CMP #1$+1, (SP)+ ;IS STACK CONTENTS CORRECT?
7661 033524 001401 BEQ 4$ ;
7662 033526 104001 ERROR +1 ;CPU ERROR
7663 033530 022726 030000 4$: CMP #30000, (SP)+ ;IS PSW CORRECT ON STACK?
7664 033534 001401 BEQ 5$ ;
7665 033536 104001 ERROR +1 ;CPU ERROR
7666 033540 005067 144222 5$: CLR CPREG ;CLEAR CPU ERROR REG
7667
7668 ;NOW WE'LL TRY TO FETCH AN INSTRUCTION FROM AN INTERNAL REGISTER.
7669 ;THIS SHOULD CAUSE A TRAP TO ADDR 4 AND SET BIT 6 IN THE CPU
7670 ;ERROR REGISTER.
7671 ;
7672 033544 012767 033572 144232 MOV #7$, 4 ;LOAD VECTOR WITH TRAP HANDLER ADDR
7673 033552 012767 030340 144226 MOV #30340, 6 ;LOAD VEC WITH PSW VALUE ON TRAP
7674 033560 005067 144212 CLR PS ;CLEAR THE PSW
7675 033564 000167 144206 JMP PS ;*****TEST INSTRUCTION*****
7676 ;TRY INSTRUCTION FETCH FROM INTERNAL
7677 ;REGISTER-- SHOULD TRAP VIA ADDR 4
7678 033570 104001 6$: ERROR +1 ;CPU ERROR
7679 033572 016701 144200 7$: MOV PS, R1 ;SAVE CONTENTS OF PSW IN R1
7680 033576 022767 000100 144162 CMP #BIT06, CPREG ;IS CPU ERROR REGISTER CORRECT?
7681 033604 001401 BEQ 8$ ;
7682 033606 104001 ERROR +1 ;CPU ERROR
7683 033610 022726 177776 8$: CMP #PS, (SP)+ ;IS STACK CONTENTS CORRECT?
7684 033614 001401 BEQ 9$ ;
7685 033616 104001 ERROR +1 ;CPU ERROR
7686 033620 022726 000000 9$: CMP #0, (SP)+ ;IS STACK CONTENTS CORRECT?
7687 033624 001401 BEQ 10$ ;

```



```

7688 033626 104001          ERROR +1          ;CPU ERROR
7689 033630 022701 000340 10$: CMP #340,R1          ;WAS PSW LOADED PROPERLY ON TRAP?
7690 033634 001401          BEQ 11$          ;
7691 033636 104001          ERROR +1          ;CPU ERROR
7692 033640 005067 144122 11$: CLR CPREG          ;CLEAR CPU ERROR REGISTER
7693 033644 016767 147112 144132 MOV SLOC00,4          ;RESTORE VECTOR
7694 033652 016767 147106 144126 MOV SLOC01,6          ;
7695
7696 033660          RXXX:
7697
7698          ; RED ZONE TRAP TEST
7699 033660 013767 000004 147074 MOV @04,SLOC00          ;SAVE VECTOR
7700 033666 012737 033716 000004 MOV @2$,@04          ;SET UP VECTOR
7701 033674 012706 000777 MOV @777,R6          ;SET UP THE STACK WITH ODD ADDRESS.
7702 033700 005067 144062 CLR CPREG          ;CLEAR THE CPU ERROR REGISTER
7703 033704 005067 144066 CLR PSW          ;CLEAR THE PSW
7704 033710 005737 177700 TST @0177700          ;ACCESS NON-EXISTANT I/O ADDRESS
7705 033714 104001          ERROR +1          ;CPU ERROR
7706 033716 022706 000000 2$: CMP #0,SP          ;IS R6 CORRECT?
7707 033722 001401          BEQ 3$          ;BRANCH IF YES
7708 033724 104001          ERROR +1          ;CPU ERROR
7709 033726 022726 033714 3$: CMP @1$, (SP)+          ;IS DATA AT ADDR 0 CORRECT?
7710 033732 001401          BEQ 4$          ;BRANCH IF YES
7711 033734 104001          ERROR +1          ;CPU ERROR
7712 033736 022716 000000 4$: CMP @0, (SP)          ;IS PSW DATA IN ADDR 2 CORRECT?
7713 033742 001401          BEQ 5$          ;BRANCH IF YES
7714 033744 104001          ERROR +1          ;CPU ERROR
7715 033746 022767 000124 144012 5$: CMP @124,CPREG          ;IS CPU ERROR REGISTER CORRECT?
7716 033754 001401          BEQ 6$          ;BRANCH IF YES
7717 033756 104001          ERROR +1          ;CPU ERROR
7718 033760 005067 144002 6$: CLR CPREG          ;CLEAR CPU ERROR REGISTER
7719 033764 012706 001000 MOV @STBOT,SP          ;RESTORE STACK
7720 033770 016737 146766 000004 MOV SLOC00,@04          ;RESTORE VECTOR
7721 033776 005037 000000 CLR @00          ;RESTORE ADDR 0
7722 034002 005037 000002 CLR @02          ;RESTORE ADDR 2
7723
7724
7725
7726 034006          PIRXXX:
7727          ; TEST PIRQ REGISTER RESPONSE
7728 034006 016767 143772 146746 MOV 4,SLOC00          ;SAVE CONTENTS OF VECTORS
7729 034014 012767 034042 143762 MOV @PIRQNX,4          ;SETUP INTERRUPT VECTOR
7730 034022 005067 143740 CLR CPREG          ;CLEAR CPU ERROR REGISTER
7731 034026 005737 177772 TST @0PIRQ          ;LOOK FOR REPLY FROM PIRQ
7732 034032 016767 146724 143744 MOV SLOC00,4          ;RESTORE VECTORS
7733 034040 000401 BR PIRTEX          ;IF IT RESPONDS, CONTINUE TESTING.
7734          ;ERROR! NO RESPONSE FROM PIRQ REG
7735 034042 104001          PIRQNX: ERROR +1          ;CPU ERROR
7736 034044          PIRTEX:
7737
7738 034044          PIR:
7739          ; TEST PIRQ REGISTER DATA
7740 034044 013767 000240 146710 MOV @0PIRQVEC,SLOC00          ;SAVE PIRQ VECTORS
7741 034052 013767 000242 146704 MOV @PIRQVEC+2,SLOC01          ;
7742 034060 012737 034270 000240 MOV @UNXPIR,@0PIRQVEC          ;SET UP PIRQ VECTOR FOR UNEXPECTED INTERRUPT
7743 034066 012737 000340 000242 MOV @PR7,@0PIRQVEC+2          ;

```

M11

COKDABO R0J11 B CLUSTER MACY11 30(1046) 05 APR 84 16:45 PAGE 143
 COKDAB.P11 05-APR 84 16:45 ***** DOUBLE OPERAND TESTS *****

SEQ 0142

```

7744 034074 012703 001000      MOV      #1000,R3      ;PUT 1000 IN R3; START TESTING
7745                                ;BITS IN PIRQ REG BY FLOATING
7746                                ;A BIT THROUGH BITS 9-15.
7747 034100 012704 034144      MOV      #PIRTBL,R4      ;SET UP R4 AS A POINTER TO EXPECTED
7748                                ;ENCODED PRIORITY LEVELS IN PIRTL
7749 034104 000237      1$:      SPL      7      ;DON'T ALLOW INTERRUPTS.
7750 034106 005037 177772      CLR      @PIRQ      ;CLEAR OUT THE PIRQ
7751 034112 023724 177772      CMP      @PIRQ,(R4)+    ;IS PIRQ OK??
7752 034116 001401      BEQ      2$      ;BRANCH IF OK
7753                                ;ERROR; PIRQ REG WAS NOT CLEAR
7754 034120 104001      ERROR    +1      ;CPU ERROR
7755 034122 010337 177772      2$:      MOV      R3,@PIRQ      ;SET A BIT IN PIRQ REGISTER
7756 034126 023724 177772      CMP      @PIRQ,(R4)+    ;COMPARE THE ENCODED PRIORITY BITS
7757                                ;WITH DATA IN THE TABLE (PIRTBL)
7758 034132 001401      BEQ      3$      ;BRANCH IF ITS OK
7759                                ;ERROR; ENCODED PRIORITY LEVELS ARE
7760                                ;NOT CORRECT
7761 034134 104001      ERROR    +1      ;CPU ERROR
7762 034136 006303      3$:      ASL      R3      ;FLOAT A "1" THRU BITS 9-15
7763 034140 103370      BCC      2$      ;IF CARRY BIT ISN'T SET, DO AGAIN.
7764 034142 000410      BR       EXPIR1      ;GO TO EXIT TEST.
7765
7766 034144 000000      PIRTL: .WORD    0
7767 034146 001042      .WORD    1042
7768 034150 002104      .WORD    2104
7769 034152 004146      .WORD    4146
7770 034154 010210      .WORD    10210
7771 034156 020252      .WORD    20252
7772 034160 040314      .WORD    40314
7773 034162 100356      .WORD    100356
7774
7775 034164      EXPIR1:
7776
7777
7778 034164      PIR2:
7779      ;      TEST PIRQ REGISTER LEVEL ENCODING
7780      ;*****
7781      ;THIS TEST IS TO CHECK THAT THE HIGHEST PRIORITY LEVEL SET IN THE
7782      ;PROGRAM INTERRUPT REQUEST BITS IS REFLECTED IN THE ENCODED PROGRAM
7783      ;INTERRUPT ACTIVE BITS.
7784 034164 000237      SPL      7      ;SHUT OFF INTERRUPTS
7785 034166 005037 177772      CLR      @PIRQ      ;CLEAR PIRQ REGISTER
7786 034172 012767 000001 146576      MOV      #1,FLAG      ;SETUP END OF LOOP SIGNAL
7787 034200 012703 177000      MOV      #177000,R3      ;SET UP DESIRED PATTERN IN R3
7788 034204 012704 034250      MOV      #PIRTBL,R4      ;SETUP R4 AS A TABLE POINTER
7789 034210 010337 177772      1$:      MOV      R3,@PIRQ      ;SET PATTERN IN PIRQ REGISTER
7790
7791 034214 023724 177772      CMP      @PIRQ,(R4)+    ;COMPARE PATTERN IN PIRQ WITH PATTERN IN
7792                                ;EXPECTED PATTERN TABLE.
7793 034220 001012      BNE      2$      ;GO TO ERROR IF NOT THE SAME
7794 034222 005737 002776      IST      @FLAG      ;IS THIS THE LAST TIME THROUGH??
7795 034226 001421      BEQ      PIR2EX      ;YES, IF FLAG IS ZERO
7796 034230 006003      ROR      R3      ;SHIFT PATTERN TO RIGHT FOR NEXT TEST
7797 034232 042703 000777      BIC      #777,R3      ;STRIP OFF BITS 8-0
7798 034236 001364      BNE      1$      ;IF R3 IS NOT = 0 GO DO THE NEXT PATTERN
7799 034240 005037 002776      CLR      @FLAG      ;CLR THE FLAG TO INDICATE LAST

```

N11

COKDABO KDJ11-B CLUSTER MACY11 30(1046)
COKDAB.P11 05-APR-84 16:4505-APR-84 16:45 PAGE 144
***** DOUBLE OPERAND TESTS *****

SEQ 0143

```

7800                                     ;TIME THROUGH THE LOOP
7801 034244 000761                     BR      1$      ;GO THROUGH LOOP ONCE MORE
7802                                     ;ERROR; PATTERNS WERE NOT THE SAME
7803 034246 104001                     2$:  ERROR  +1      ;CPU ERROR
7804
7805 034250 177356 077314 037252 PITBL1: .WORD 177356,77314,37252,17210,7146,3104,1042,0
7806 034256 017210 007146 003104
7807 034264 001042 000000
7808
7809 034270                                     UNXPIR:
7810 034270 104001                     ERROR  +1      ;CPU ERROR
7811
7812 034272                                     PIR2EX:
7813
7814 034272                                     PIR3:
7815                                     ;      TEST PIRQ INTERRUPTS
7816                                     ;*****
7817                                     ;THIS TEST CHECKS THAT EACH PROGRAM INTERRUPT OCCURS PROPERLY
7818 034272 012703 001000                     MOV     #1000,R3      ;SETUP R3 AS A WORKING REGISTER
7819 034276 012737 034342 000240                     MOV     @PIRRTN,@PIRQVEC      ;SET UP INTERRUPT ROUTINE AT PIR VECTOR
7820 034304 012737 000340 000242                     MOV     #340,@PIRQVEC+2      ;
7821 034312 005004                                     CLR     R4      ;INITIALIZE R4 AS EXPECTED DATA HOLDER.
7822 034314 005724                                     TST     (R4)+    ;INCREMENT R4 BY 2
7823 034316 050337 177772                     PI1:  BIS     R3,@PIRQ      ;SET PIRQ TO INTERRUPT
7824 034322 000230                                     SPL     0      ;ENABLE INTERRUPT TO OCCUR
7825                                     ;ERROR! INTERRUPT DID NOT OCCUR
7826 034324 104001                     ERROR  +1      ;CPU ERROR
7827                                     ;ERROR! INTERRUPT OCCURRED IN WRONG SEQUENCE
7828 034326 104001                     PI2:  ERROR  +1      ;CPU ERROR
7829 034330 062706 000004                     PI3:  ADD     #4,SP      ;CLEAN UP THE STACK
7830 034334 006303                                     ASL     R3      ;SHIFT R3 TO LEFT FOR NEXT INTERRUPT
7831 034336 103366                                     BCC     PI1     ;END IF CARRY BIT IS SET
7832 034340 000412                                     BR      PIR3EX  ;ON TO THE NEXT TEST
7833
7834 034342 013705 177772                     PIRRTN: MOV     @PIRQ,R5      ;MOVE THE CONTENTS OF PIRQ REG TO R5
7835 034346 005037 177772                     CLR     @PIRQ      ;KILL PIRQ INTERRUPT.
7836 034352 042705 177761                     BIC     #177761,R5      ;MASK OFF ALL BITS EXCEPT 1-4.
7837 034356 020405                                     CMP     R4,R5      ;DID THE CORRECT LEVEL INTERRUPT OCCUR?
7838 034360 001362                                     BNE     PI2      ;IF NOT; GO TO ERROR.
7839 034362 000167 177742                     JMP     PI3      ;RETURN FROM SUCCESSFUL INTERRUPT,GET
7840                                     ;READY FOR THE NEXT ONE.
7841
7842 034366                                     PIR3EX:
7843
7844
7845 034366                                     PIR4:
7846                                     ;      TEST PIRQ VS PSW INTERRUPT LEVEL
7847                                     ;*****
7848                                     ;THIS TEST IS TO ENSURE THAT PIRQ CANNOT INTERRUPT WHEN PSW INTERRUPT
7849                                     ;LEVEL IS SET TO BLOCK THEM OUT.
7850 034366 005037 177772                     CLR     @PIRQ      ;CLEAR THE PIRQ
7851 034372 005037 177776                     CLR     @PSW      ;CLEAR THE PSW.
7852 034376 000237                                     SPL     7      ;BLOCK INTERRUPTS FROM BEING SERVICED.
7853 034400 012703 001000                     MOV     #1000,R3      ;USE R3 AS A WORKING REGISTER.
7854 034404 012737 034436 000240                     MOV     #2$,@PIRQVEC      ;SETUP INTERRUPT VECTORS
7855 034412 012737 000340 000242                     MOV     #340,@PIRQVEC+2

```

7856	034420	010337	177772	13:	MOV	R3,00PIRQ	SET INTERRUPT LEVEL IN PIRQ.
7857	034424	006303			AND	R3	SHIFT A "1" LEFT TO INCREASE INTERRUPT
7858							PRIORITY LEVEL.
7859	034426	103374			BCC	13	IF C BIT NOT SET THEN DO IT AGAIN.
7860	034430	005037	177772		CLR	00PIRQ	ELSE CLEAR THE PIRQ
7861	034434	000403			BR	33	EXIT TEST.
7862							
7863	034436	005037	177772	23:	CLR	00PIRQ	STOP PIRQ FROM INTERRUPTING.
7864							ERROR! NO INTERRUPTS SHOULD OCCUR.
7865	034440	104001			ERROR	+1	CPU ERROR
7866	034444			33:			
7867							
7868							
7869	034444						
7870							
7871							
7872							
7873							
7874							
7875							
7876							
7877							
7878							
7879	034444	005037	177772		CLR	00PIRQ	CLR THE PIRQ
7880	034450	012737	034562	000240	MOV	0101,00PIRQVEC	SET UP INTERRUPT VECTORS
7881	034456	012737	000340	000242	MOV	0PR7,00PIRQVEC+2	
7882	034458	012705	034640		MOV	0PITBL2,R5	SETUP R5 AS A TABLE POINTER.
7883	034470	012737	000006	110500	MOV	06,00DCOUNT	INIT LOCATION DCOUNT.
7884	034476	005004			MOV	04,R4	INITIALIZE R4 AS A COUNTER.
7885	034500	012703	001000		MOV	01000,R3	USE R3 AS A WORKING REGISTER.
7886	034504	012567	143266		MOV	(R5),R3	MOVE PRIORITY LEVEL TO PSW
7887	034510	005724			MOV	(R4),R3	INCREMENT R4 BY 1
7888	034512	010337	177772		MOV	R3,00PIRQ	SET INTERRUPT LEVEL IN PIRQ.
7889							
7890	034516	013701	177776		MOV	00PS,R1	SAVE PS IN R1
7891	034522	013702	177772		MOV	00PIRQ,R2	SAVE PIRQ IN R2
7892	034526	042701	177437		BIC	0177437,R1	CLEAR EXTRANEOUS BITS
7893	034532	042702	177437		BIC	0177437,R2	CLEAR EXTRANEOUS BITS
7894	034536	020102			CMP	R1,R2	COMPARE R1 AND R2
7895	034540	002001			BGE	43	IF R1 >= R2, GO ON
7896							ERROR! SHOULD HAVE INTERRUPTED.
7897	034542	104001			ERROR	+1	CPU ERROR
7898	034544	006303		43:	AND	R3	SHIFT A "1" LEFT UNTIL INTERRUPT LEVEL
7899							IS REACHED.
7900	034546	103360			BCC	33	BRANCH BACK IF CARRY IS NOT SET.
7901	034550	005337	110500		DEC	00DCOUNT	LOWER INTERRUPT PRIORITY LEVEL
7902	034554	001350			BNE	13	IF DCOUNT > 0, WE'RE DONE.
7903	034556	000167	000072		JMP	PIR53	JUMP TO END OF THIS TEST.
7904							
7905							
7906							
7907							
7908	034562	015746	177772	103:	MOV	00PIRQ,(SP)	SAVE PIRQ DATA ON STACK
7909	034566	005037	177772		CLR	00PIRQ	RESET OF PIRQ INTERRUPT
7910	034570	016601	000004		MOV	4(SP),R1	GET OLD PSW, SAVE IN R1.
7911	034576	011602			MOV	(SP),R2	GET PIRQ FROM STACK.

PIRQ INTERRUPT SERVICE ROUTINE

7912	034600	042702	177437		BIC	0177437,R2		;CLEAR UNWANTED BITS FROM R2
7913	034604	042701	177437		BIC	0177437,R1		;CLEAR UNWANTED BITS FROM R1
7914	034610	020102			CMP	R1,R2		;R2 SHOULD BE > R1.
7915	034612	100401			BMI	20\$		;GO CHECK SEQUENCE OF INTERRUPT.
7916								;ERROR! PRIORITY OF INTERRUPT WAS NOT
7917								;HIGH ENOUGH, SHOULDN'T HAVE OCCURRED.
7918	034614	104001			ERROR	+1		;CPU ERROR
7919	034616	012602		20\$:	MOV	(SP)+,R2		;POP OLD PIRQ OFF THE STACK.
7920	034620	042702	177761		BIC	0177761,R2		;CLEAR OFF EXTRANEOUS BITS.
7921	034624	020402			CMP	R4,R2		;SHOULD BE EQUAL.
7922	034626	001401			BEQ	21\$		;IF THEY ARE EQUAL, CLEAN UP STACK AND
7923								;GET READY FOR THE NEXT INTERRUPT
7924								;ELSE
7925								;ERROR! INTERRUPT OCCURRED OUT OF SEQUENCE.
7926	034630	104001			ERROR	+1		;CPU ERROR
7927	034632	012716	034544	21\$:	MOV	04\$, (SP)		;PUT RETURN ADDRESS ON THE STACK.
7928	034636	000002			RTI			;RETURN FROM INTERRUPT, RESTORE PSW.
7929								
7930	034640	000300		PITBL2:	.WORD	300		;PRIORITY LEVEL 6
7931	034642	000240			.WORD	240		;PRIORITY LEVEL 5
7932	034644	000200			.WORD	200		;PRIORITY LEVEL 4
7933	034646	000140			.WORD	140		;PRIORITY LEVEL 3
7934	034650	000100			.WORD	100		;PRIORITY LEVEL 2
7935	034652	000040			.WORD	40		;PRIORITY LEVEL 1
7936								
7937	034654			PIR5EX:				
7938								
7939								
7940	034654			PIR6:				
7941								;TEST THAT PIRQS ARE SERVICED IN CORRECT ORDER
7942								;*****
7943								;THIS TEST CHECKS THAT ALL PIRQ INTERRUPTS ARE SERVICED
7944								;IN THE CORRECT DECENDING ORDER, I.E. IRQ6 IS NOT SERVICED
7945								;BEFORE IRQ7 ETC. THE PIRQ IS LOADED WITH ALL INTERRUPTS SIMULTANEOUSLY
7946								;TURNED ON. THE PSW PRIORITY LEVEL IS THEN LOWERED TO 0. EACH INTERRUPT
7947								;SHOULD BE SERVICED IN DECENDING ORDER. EACH TIME AN INTERRUPT OCCURS, THE PSW
7948								;IS LOADED WITH PRIORITY LEVEL 7 WHICH STOPS THE PIRQ FROM FURTHER INTERRUPTS.
7949								;AFTER A CORRECT INTERRUPT, AN RTI IS EXECUTED, AND THE PSW PRIORITY LEVEL
7950								;RETRUNS TO ZERO ALLOWING THE NEXT LOWER INTERRUPT TO OCCUR.
7951	034654	005037	177772		CLR	00PIRQ		;CLEAR OUT THE PIRQ
7952	034660	000237			SPL	7		;NO INTERRUPTS WILL BE SERVICED.
7953	034662	012737	034722	000240	MOV	010\$, 00PIRQVEC		;SET UP VECTORS
7954	034670	012737	000340	000242	MOV	0PR7, 00PIRQVEC+2		
7955	034676	012703	177000		MOV	0177000, R3		;SETUP DESIRED PATTERN IN R3
7956	034702	010337	177772		MOV	R3, 00PIRQ		;MOVE PATTERN IN R3 TO PIRQ.
7957	033706	012704	000016		MOV	016, R4		;PRELOAD R4 AS A DECREMENTING COUNTER.
7958	034712	000230			SPL	0		;LET THE PIRQ INTERRUPT
7959								;***** INTERRUPTS OCCUR HERE!!! *****
7960	034714	005704			IST	R4		;IS R4 ZERO? IT SHOULD BE. THIS
7961								;INSTRUCTION SHOULDN'T BE EXECUTED
7962								;UNTIL ALL INTERRUPTS HAVE OCCURRED.
7963	034716	001417			BEQ	PIR6EX		;GO TO NEXT TEST.
7964								;ERROR! ALL INTERRUPTS DID NOT OCCUR.
7965	034720	104001			ERROR	+1		;CPU ERROR
7966	034722	013702	177772	10\$:	MOV	00PIRQ, R2		;SAVE THE PIRQ IN R2
7967	034726	042702	177761		BIC	0177761, R2		;STRIP OFF EXTRANEOUS BITS.

CORRADO RDJ11 B CLUSTER MACY11 30(1046)
COKDAB,P11 05-APR 84 16:45.

05 APR 84 16:45 PAGE 147
***** DOUBLE OPERAND TESTS *****

SEQ 0146

```

7968 034732 020402      CMP      R4,R2      ;SHOULD BE EQUAL.
7969 034734 001401      BFC      15$      ;SETUP FOR NEXT INTERRUPT.
7970                                     ;ERROR! INTERRUPT WAS SERVICED OUT OF ORDER
7971 034736 104001      ERROR    +1      ;CPU ERROR
7972 034740 005744      15$:      IST      (R4)      ;DECREMENT R4 BY 2
7973 034742 006003      ROR      R3      ;CHANGE PATTERN IN R3; LOWER INTERRUPT PRIORITY
7974 034744 042703 000777 BIC      0777,R3      ;STRIP OFF UNNEEDED BITS
7975 034750 010337 177772 MOV      R3,00PIRQ      ;LOWER INTERRUPT PRIORITY LEVEL.
7976 034754 000002      RTI
7977
7978 034756 016737 146000 000240 PIR6EX: MOV      SLOC00,00PIRQVEC      ;RESTORE VECTORS
7979 034764 016737 145774 000242      MOV      SLOC01,00PIRQVEC+2      ;
7980
7981
7982
7983
7984
7985
7986
7987
7988 034772
7989
7990 034772 005067 142770      ;SBTTL MEMORY MANAGEMENT TESTS
7991 034776 005037 177572      ;*****
7992 035002 005037 002776      ;*****
7993 035006 013746 000004      ;*****
7994 035012 012737 133302 000004      ;*****
7995 035020 005005      ;*****
7996 035022 013701 177572      ;*****
7997 035026 013701 177574      ;*****
7998 035032 013701 177576      ;*****
7999 035036 013701 172516      ;*****
8000 035042 012637 000004      ;*****
8001 035046 020527 000000      ;*****
8002 035052 001401      ;*****
8003 035054 104002      ;*****
8004
8005 035056
8006 035056
8007
8008 035056 005067 142704      ;*****
8009 035062 005037 177572      ;*****
8010 035066 005037 002776      ;*****
8011 035072 013746 000244      ;*****
8012 035076 013746 000246      ;*****
8013 035102 013746 000004      ;*****
8014 035106 012737 000246 000244      ;*****
8015 035114 012737 000002 000246      ;*****
8016 035122 012737 133302 000004      ;*****
8017 035130 005005      ;*****
8018 035132 012700 172200      ;*****
8019 035136 005020      ;*****
8020 035140 020027 172400      ;*****
8021 035144 001374      ;*****
8022 035146 012700 177600      ;*****
8023 035152 005020      ;*****

```


8024	035154	020027	177700		CMP	R0,#177700	:	
8025	035160	001374			BNE	21	:	
8026	035162	170127	000200		LDEPS	#200	:	
8027	035166	012700	003012		MOV	#FLOAT,R0	:	LOAD AC0-AC5 WITH 0
8028	035172	005020			CLR	(R0)+	:	
8029	035174	005020			CLR	(R0)+	:	
8030	035176	005020			CLR	(R0)+	:	
8031	035200	005020			CLR	(R0)+	:	
8032	035202	012700	003012		MOV	#FLOAT,R0	:	
8033	035206	172410			LDD	(R0),AC0	:	
8034	035210	172510			LDD	(R0),AC1	:	
8035	035212	172610			LDD	(R0),AC2	:	
8036	035214	172710			LDD	(R0),AC3	:	
8037	035216	174004			STD	AC0,AC4	:	
8038	035220	174005			STD	AC0,AC5	:	
8039	035222	174500		3\$:	DIVD	AC0,AC1	:	LOAD FEC WITH 4 AND FEA WITH #3\$
8040	035224	170337	003022		STST	#FLO	:	CHECK FEC FOR 4 AND FEA FOR #3\$
8041	035230	012704	003022		MOV	#FLO,R4	:	
8042	035234	022427	000004		CMP	(R4)+,#4	:	
8043	035240	001401			BEQ	21\$	:	
8044	035242	104002			ERROR	+2	:	MMU ERROR
8045							:	
8046	035244	021427	035222	21\$:	CMP	(R4),#3\$	:	
8047	035250	001401			BEQ	22\$	:	
8048	035252	104002			ERROR	+2	:	MMU ERROR
8049							:	
8050	035254	012704	172200	22\$:	MOV	#172200,R4	:	CHECK EACH PAR. PDR FOR 0 THEN
8051	035260	012701	000001		MOV	#1,R1	:	WRITE A UNIQUE NUMBER TO IT
8052	035264	010102		4\$:	MOV	R1,R2	:	
8053	035266	072227	000010		ASH	#10,R2	:	
8054	035272	021427	000000		CMP	(R4),#0	:	
8055	035276	001401			BEQ	5\$	:	
8056	035300	104002			ERROR	+2	:	MMU ERROR
8057							:	
8058	035302	010224		5\$:	MOV	R2,(R4)+	:	
8059	035304	005201			INC	R1	:	
8060	035306	020427	172400		CMP	R4,#172400	:	
8061	035312	001364			BNE	4\$	:	
8062	035314	012704	177600		MOV	#177600,R4	:	
8063	035320	010102		6\$:	MOV	R1,R2	:	
8064	035322	072227	000010		ASH	#10,R2	:	
8065	035326	021427	000000		CMP	(R4),#0	:	
8066	035332	001401			BEQ	7\$	:	
8067	035334	104002			ERROR	+2	:	MMU ERROR
8068							:	
8069	035336	010224		7\$:	MOV	R2,(R4)+	:	
8070	035340	005201			INC	R1	:	
8071	035342	020427	177700		CMP	R4,#177700	:	
8072	035346	001364			BNE	6\$	:	
8073	035350	012704	003022		MOV	#FLO,R4	:	CHECK AC5 FOR ALL ZEROES THEN LOAD A 6
8074	035354	012703	003012		MOV	#FLOAT,R3	:	
8075	035360	174014			STD	AC0,(R4)	:	
8076	035362	172405			LDD	AC5,AC0	:	
8077	035364	174013			STD	AC0,(R3)	:	
8078	035366	012702	000004		MOV	#4,R2	:	
8079	035372	022327	000000	8\$:	CMP	(R3)+,#0	:	

8080	035376	001401		BEQ	9\$			
8081	035400	104002		ERROR	+2		;MMU ERROR	
8082								
8083	035402	005302	9\$:	DEC	R2			
8084	035404	001372		BNE	8\$			
8085	035406	012703	003012	MOV	#FLOAT,R3			
8086	035412	012713	000006	MOV	#6,(R3)			
8087	035416	172413		LDD	(R3),AC0			
8088	035420	174005		STD	AC0,AC5			
8089	035422	172404		LDD	AC4,AC0		CHECK AC4 FOR ALL ZEROES THEN LOAD A 5	
8090	035424	174013		STD	AC0,(R3)			
8091	035426	012702	000004	MOV	#4,R2			
8092	035432	022327	000000	10\$:	CMP	(R3)+,#0		
8093	035436	001401		BEQ	11\$			
8094	035440	104002		ERROR	+2		;MMU ERROR	
8095								
8096	035442	005302	11\$:	DEC	R2			
8097	035444	001372		BNE	10\$			
8098	035446	012703	003012	MOV	#FLOAT,R3			
8099	035452	012713	000005	MOV	#5,(R3)			
8100	035456	172413		LDD	(R3),AC0			
8101	035460	174004		STD	AC0,AC4			
8102	035462	012702	000004	MOV	#4,R2		CHECK AC0 FOR ALL ZEROES THEN LOAD A 1	
8103	035466	022427	000000	12\$:	CMP	(R4)+,#0		
8104	035472	001401		BEQ	13\$			
8105	035474	104002		ERROR	+2		;MMU ERROR	
8106								
8107	035476	005302	13\$:	DEC	R2			
8108	035500	001372		BNE	12\$			
8109	035502	012713	000001	MOV	#1,(R3)			
8110	035506	172413		LDD	(R3),AC0			
8111	035510	012704	003022	MOV	#FLO,R4		CHECK AC1 FOR ALL ZEROES THEN LOAD A 2	
8112	035514	012702	000004	MOV	#4,R2			
8113	035520	174114		STD	AC1,(R4)			
8114	035522	022427	000000	14\$:	CMP	(R4)+,#0		
8115	035526	001401		BEQ	15\$			
8116	035530	104002		ERROR	+2		;MMU ERROR	
8117								
8118	035532	005302	15\$:	DEC	R2			
8119	035534	001372		BNE	14\$			
8120	035536	012713	000002	MOV	#2,(R3)			
8121	035542	172513		LDD	(R3),AC1			
8122	035544	012704	003022	MOV	#FLO,R4		CHECK AC2 FOR ALL ZEROES THEN LOAD A 3	
8123	035550	012702	000004	MOV	#4,R2			
8124	035554	174214		STD	AC2,(R4)			
8125	035556	022427	000000	16\$:	CMP	(R4)+,#0		
8126	035562	001401		BEQ	17\$			
8127	035564	104002		ERROR	+2		;MMU ERROR	
8128								
8129	035566	005302	17\$:	DEC	R2			
8130	035570	001372		BNE	16\$			
8131	035572	012713	000003	MOV	#3,(R3)			
8132	035576	172513		LDD	(R3),AC2			
8133	035600	012704	003022	MOV	#FLO,R4		CHECK AC3 FOR ALL ZEROES THEN LOAD A 4	
8134	035604	012702	000004	MOV	#4,R2			
8135	035610	174314		STD	AC3,(R4)			

8136	035612	022427	000000	18\$:	CMP	(R4)+, #0	:
8137	035616	001401			BEQ	19\$	:
8138	035620	104002			ERROR	+2	;MMU ERROR
8139							:
8140	035622	005302		19\$:	DEC	R2	:
8141	035624	001372			BNE	18\$	:
8142	035626	012713	000004		MOV	#4, (R3)	:
8143	035632	172713			LDD	(R3), AC3	:
8144	035634	012704	003022		MOV	#FLO, R4	;CHECK FPS FOR 100204 THEN LOAD IT WITH 200
8145	035640	170214			STFPS	(R4)	:
8146	035642	022714	100204		CMP	#100204, (R4)	:
8147	035646	001401			BEQ	20\$	:
8148	035650	104002			ERROR	+2	;MMU ERROR
8149							:
8150	035652	170127	000200	20\$:	LDFPS	#200	:
8151	035656	012704	172200		MOV	#172200, R4	;CHECK PDR, PAR FOR UNIQUE NUMBERS
8152	035662	012701	000001		MOV	#1, R1	:
8153	035666	010102		23\$:	MOV	R1, R2	:
8154	035670	072227	000010		ASH	#10, R2	:
8155	035674	022402			CMP	(R4)+, R2	:
8156	035676	001401			BEQ	24\$	:
8157	035700	104002			ERROR	+2	;MMU ERROR
8158							:
8159	035702	005201		24\$:	INC	R1	:
8160	035704	020427	172400		CMP	R4, #172400	:
8161	035710	001366			BNE	23\$	:
8162	035712	012704	177600		MOV	#177600, R4	:
8163	035716	010102		25\$:	MOV	R1, R2	:
8164	035720	072227	000010		ASH	#10, R2	:
8165	035724	022402			CMP	(R4)+, R2	:
8166	035726	001401			BEQ	26\$	:
8167	035730	104002			ERROR	+2	;MMU ERROR
8168							:
8169	035732	005201		26\$:	INC	R1	:
8170	035734	020427	177700		CMP	R4, #177700	:
8171	035740	001366			BNE	25\$	:
8172	035742	012701	003012		MOV	#FLOAT, R1	;CHECK AC5 FOR #6
8173	035746	012704	003022		MOV	#FLO, R4	:
8174	035752	174014			STD	AC0, (R4)	:
8175	035754	172405			LDD	AC5, AC0	:
8176	035756	174011			STD	AC0, (R1)	:
8177	035760	022127	000006		CMP	(R1)+, #6	:
8178	035764	001401			BEQ	27\$	:
8179	035766	104002			ERROR	+2	;MMU ERROR
8180							:
8181	035770	012703	000003	27\$:	MOV	#4, R3	:
8182	035774	022127	000000	28\$:	CMP	(R1)+, #0	:
8183	036000	001401			BEQ	29\$	:
8184	036002	104002			ERROR	+2	;MMU ERROR
8185							:
8186	036004	005303		29\$:	DEC	R3	:
8187	036006	001372			BNE	28\$	:
8188	036010	012701	003012		MOV	#FLOAT, R1	;CHECK AC4 FOR #5
8189	036014	172404			LDD	AC4, AC0	:
8190	036016	174011			STD	AC0, (R1)	:
8191	036020	022127	000005		CMP	(R1)+, #5	:

H12

CONDABO RDJ11 R CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 151
 COKDAB.P11 05-APR-84 16:45 MEMORY MANAGEMENT TESTS

SEQ 0150

8192	036024	001401			BEQ	30\$			
8193	036026	104002			ERROR	+2		; MMU ERROR	
8194									
8195	036030	012703	000003	30\$:	MOV	#3,R3			
8196	036034	022127	000000	31\$:	CMP	(R1)+,#0			
8197	036040	001401			BEQ	32\$			
8198	036042	104002			ERROR	+2		; MMU ERROR	
8199									
8200	036044	005303		32\$:	DEC	R3			
8201	036046	001372			BNE	31\$			
8202	036050	022427	000001		CMP	(R4)+,#1		; CHECK AC0 FOR #1	
8203	036054	001401			BEQ	33\$			
8204	036056	104002			ERROR	+2		; MMU ERROR	
8205									
8206	036060	012703	000003	33\$:	MOV	#3,R3			
8207	036064	022427	000000	34\$:	CMP	(R4)+,#0			
8208	036070	001401			BEQ	35\$			
8209	036072	104002			ERROR	+2		; MMU ERROR	
8210									
8211	036074	005303		35\$:	DEC	R3			
8212	036076	001372			BNE	34\$			
8213	036100	012701	003012		MOV	#FLOAT,R1		; CHECK AC1 FOR #2	
8214	036104	174111			STD	AC1,(R1)			
8215	036106	022127	000002		CMP	(R1)+,#2			
8216	036112	001401			BEQ	36\$			
8217	036114	104002			ERROR	+2		; MMU ERROR	
8218									
8219	036116	012703	000003	36\$:	MOV	#3,R3			
8220	036122	022127	000000	37\$:	CMP	(R1)+,#0			
8221	036126	001401			BEQ	38\$			
8222	036130	104002			ERROR	+2		; MMU ERROR	
8223									
8224	036132	005303		38\$:	DEC	R3			
8225	036134	001372			BNE	37\$			
8226	036136	012701	003012		MOV	#FLOAT,R1		; CHECK AC2 FOR #3	
8227	036142	174211			STD	AC2,(R1)			
8228	036144	022127	000003		CMP	(R1)+,#3			
8229	036150	001401			BEQ	39\$			
8230	036152	104002			ERROR	+2		; MMU ERROR	
8231									
8232	036154	012703	000003	39\$:	MOV	#3,R3			
8233	036160	022127	000000	40\$:	CMP	(R1)+,#0			
8234	036164	001401			BEQ	41\$			
8235	036166	104002			ERROR	+2		; MMU ERROR	
8236									
8237	036170	005303		41\$:	DEC	R3			
8238	036172	001372			BNE	40\$			
8239	036174	012701	003012		MOV	#FLOAT,R1		; CHECK AC3 FOR #4	
8240	036200	174311			STD	AC3,(R1)			
8241	036202	022127	000004		CMP	(R1)+,#4			
8242	036206	001401			BEQ	42\$			
8243	036210	104002			ERROR	+2		; MMU ERROR	
8244									
8245	036212	012703	000003	42\$:	MOV	#3,R3			
8246	036216	022127	000000	43\$:	CMP	(R1)+,#0			
8247	036222	001401			BEQ	44\$			

8248	036224	104002		ERROR	+2		;MMU ERROR
8249							
8250	036226	005303		44\$:	DEC	R3	
8251	036230	001372			BNE	43\$	
8252	036232	020527	000000		CMP	R5,#0	;IS TIME OUT FLAG 0
8253	036236	001401			BEQ	45\$	;YES GO ON
8254	036240	104002			ERROR	+2	;MMU ERROR
8255							;NO GO TO ERROR
8256	036242	012637	000004	45\$:	MOV	(SP)+,#04	;RESTORE TIME OUT VECTOR
8257	036246	012637	000246		MOV	(SP)+,#0246	;RESTORE FP VECTOR
8258	036252	012637	000244		MOV	(SP)+,#0244	
8259							
8260	036256			TSMHU3:			
8261							
8262	036256	005037	177572		CLR	#0177572	;MMU OFF
8263	036262	005037	002776		CLR	#0FLAG	;CLEAR MMU ABORT FLAG
8264	036266	012703	172200		MOV	#172200,R3	;LOAD ALL PARS AND PDRS WITH ONES
8265	036272	012723	177777	1\$:	MOV	#177777,(R3)+	
8266	036276	020327	172400		CMP	R3,#172400	
8267	036302	001373			BNE	1\$	
8268	036304	012703	177600		MOV	#177600,R3	
8269	036310	012723	177777	2\$:	MOV	#177777,(R3)+	
8270	036314	020327	177700		CMP	R3,#177700	
8271	036320	001373			BNE	2\$	
8272	036322	012703	172200		MOV	#172200,R3	;CHECK SPDRS FOR ONES
8273	036326	022327	177416	3\$:	CMP	(R3)+,#177416	
8274	036332	001401			BEQ	4\$	
8275	036334	104002			ERROR	+2	;MMU ERROR
8276							
8277	036336	020327	172240	4\$:	CMP	R3,#172240	
8278	036342	001371			BNE	3\$	
8279	036344	022327	177777	5\$:	CMP	(R3)+,#177777	;CHECK SPARS FOR ONES
8280	036350	001401			BEQ	6\$	
8281	036352	104002			ERROR	+2	;MMU ERROR
8282							
8283	036354	020327	172300	6\$:	CMP	R3,#172300	
8284	036360	001371			BNE	5\$	
8285	036362	022327	177416	7\$:	CMP	(R3)+,#177416	;CHECK KPDRS FOR ONES
8286	036366	001401			BEQ	8\$	
8287	036370	104002			ERROR	+2	;MMU ERROR
8288							
8289	036372	020327	172340	8\$:	CMP	R3,#172340	
8290	036376	001371			BNE	7\$	
8291	036400	022327	177777	9\$:	CMP	(R3)+,#177777	;CHECK KPARS FOR ONES
8292	036404	001401			BEQ	10\$	
8293	036406	104002			ERROR	+2	;MMU ERROR
8294							
8295	036410	020327	172400	10\$:	CMP	R3,#172400	
8296	036414	001371			BNE	9\$	
8297	036416	012703	177600		MOV	#177600,R3	;CHECK UPDRS FOR ONES
8298	036422	022327	177416	11\$:	CMP	(R3)+,#177416	
8299	036426	001401			BEQ	12\$	
8300	036430	104002			ERROR	+2	;MMU ERROR
8301							
8302	036432	020327	177640	12\$:	CMP	R3,#177640	
8303	036436	001371			BNE	11\$	

8304	036440	022327	177777	13\$:	CMP	(R3)+,#177777	;CHECK UPARS FOR ONES
8305	036444	001401			BEQ	14\$	
8306	036446	104002			ERROR	+2	;MMU ERROR
8307							
8308	036450	020327	177700	14\$:	CMP	R3,#177700	
8309	036454	001371			BNE	13\$	
8310	036456	012703	172200		MOV	#172200,R3	;LOAD ALL PARS AND PDRS WITH ZEROES
8311	036462	012723	000000	15\$:	MOV	#0,(R3)+	
8312	036466	020327	172400		CMP	R3,#172400	
8313	036472	001373			BNE	15\$	
8314	036474	012703	177600		MOV	#177600,R3	
8315	036500	012723	000000	16\$:	MOV	#0,(R3)+	
8316	036504	020327	177700		CMP	R3,#177700	
8317	036510	001373			BNE	16\$	
8318	036512	012703	172200		MOV	#172200,R3	;CHECK ALL PARS AND PDRS FOR ZEROES
8319	036516	022327	000000	17\$:	CMP	(R3)+,#0	
8320	036522	001401			BEQ	18\$	
8321	036524	104002			ERROR	+2	;MMU ERROR
8322							
8323	036526	020327	172400	18\$:	CMP	R3,#172400	
8324	036532	001371			BNE	17\$	
8325	036534	012703	177600		MOV	#177600,R3	
8326	036540	022327	000000	19\$:	CMP	(R3)+,#0	
8327	036544	001401			BEQ	20\$	
8328	036546	104002			ERROR	+2	;MMU ERROR
8329							
8330	036550	020327	177700	20\$:	CMP	R3,#177700	
8331	036554	001371			BNE	19\$	
8332							
8333	036556			TSMU4:			
8334							
8335	036556	005037	177572		CLR	#177572	;MMU OFF
8336	036562	005067	144210		CLR	FLAG	;CLEAR MMU ABORT FLAG
8337	036566	012700	172200		MOV	#172200,R0	;LOAD SPDRS WITH ALTERNATING PATTERN
8338	036572	012720	052404	1\$:	MOV	#52404,(R0)+	
8339	036576	012720	125012		MOV	#125012,(R0)+	
8340	036602	020027	172240		CMP	R0,#172240	
8341	036606	001371			BNE	1\$	
8342	036610	012720	125252	2\$:	MOV	#125252,(R0)+	;LOAD SPARS WITH ALTERNATING PATTERN
8343	036614	012720	052525		MOV	#52525,(R0)+	
8344	036620	020027	172300		CMP	R0,#172300	
8345	036624	001371			BNE	2\$	
8346	036626	012720	052404	3\$:	MOV	#52404,(R0)+	;LOAD KPDRS WITH ALTERNATING PATTERN
8347	036632	012720	125012		MOV	#125012,(R0)+	
8348	036636	020027	172340		CMP	R0,#172340	
8349	036642	001371			BNE	3\$	
8350	036644	012720	125252	4\$:	MOV	#125252,(R0)+	;LOAD KPARS WITH ALTERNATING PATTERN
8351	036650	012720	052525		MOV	#52525,(R0)+	
8352	036654	020027	172400		CMP	R0,#172400	
8353	036660	001371			BNE	4\$	
8354	036662	012700	177600		MOV	#177600,R0	;LOAD UPDRS WITH ALTERNATING PATTERN
8355	036666	012720	052404	5\$:	MOV	#52404,(R0)+	
8356	036672	012720	125012		MOV	#125012,(R0)+	
8357	036676	020027	177640		CMP	R0,#177640	
8358	036702	001371			BNE	5\$	
8359	036704	012720	125252	6\$:	MOV	#125252,(R0)+	;LOAD UPARS WITH ALTERNATING PATTERN

K12

COKDABO KDJ11 8 CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 154
 COKDAB.P11 05-APR-84 16:45 MEMORY MANAGEMENT TESTS

SEQ 0153

8360	036710	012720	052525		MOV	#52525,(R0)+	:
8361	036714	020027	177700		CMP	R0,#177700	:
8362	036720	001371			BNE	6\$	:
8363				:			
8364	036722	012703	172200		MOV	#172200,R3	:CHECK SPDRS
8365	036726	022327	052404	7\$:	CMP	(R3)+,#52404	:
8366	036732	001401			BEQ	8\$	:
8367	036734	104002			ERROR	+2	:MMU ERROR
8368							:
8369	036736	022327	125012	8\$:	CMP	(R3)+,#125012	:
8370	036742	001401			BEQ	9\$	:
8371	036744	104002			ERROR	+2	:MMU ERROR
8372							:
8373	036746	020327	172240	9\$:	CMP	R3,#172240	:
8374	036752	001365			BNE	7\$	:
8375	036754	022327	125252	10\$:	CMP	(R3)+,#125252	:CHECK SPARS
8376	036760	001401			BEQ	11\$	:
8377	036762	104002			ERROR	+2	:MMU ERROR
8378							:
8379	036764	022327	052525	11\$:	CMP	(R3)+,#52525	:
8380	036770	001401			BEQ	12\$	:
8381	036772	104002			ERROR	+2	:MMU ERROR
8382							:
8383	036774	020327	172300	12\$:	CMP	R3,#172300	:
8384	037000	001365			BNE	10\$	:
8385	037002	022327	052404	13\$:	CMP	(R3)+,#52404	:CHECK KPDRS
8386	037006	001401			BEQ	14\$	:
8387	037010	104002			ERROR	+2	:MMU ERROR
8388							:
8389	037012	022327	125012	14\$:	CMP	(R3)+,#125012	:
8390	037016	001401			BEQ	15\$	:
8391	037020	104002			ERROR	+2	:MMU ERROR
8392							:
8393	037022	020327	172340	15\$:	CMP	R3,#172340	:
8394	037026	001365			BNE	13\$	:
8395	037030	022327	125252	16\$:	CMP	(R3)+,#125252	:CHECK KPARS
8396	037034	001401			BEQ	17\$	:
8397	037036	104002			ERROR	+2	:MMU ERROR
8398							:
8399	037040	022327	052525	17\$:	CMP	(R3)+,#52525	:
8400	037044	001401			BEQ	18\$	:
8401	037046	104002			ERROR	+2	:MMU ERROR
8402							:
8403	037050	020327	172400	18\$:	CMP	R3,#172400	:
8404	037054	001365			BNE	16\$	:
8405	037056	012703	177600		MOV	#177600,R3	:CHECK UPDRS
8406	037062	022327	052404	19\$:	CMP	(R3)+,#52404	:
8407	037066	001401			BEQ	20\$	:
8408	037070	104002			ERROR	+2	:MMU ERROR
8409							:
8410	037072	022327	125012	20\$:	CMP	(R3)+,#125012	:
8411	037076	001401			BEQ	21\$	:
8412	037100	104002			ERROR	+2	:MMU ERROR
8413							:
8414	037102	020327	177640	21\$:	CMP	R3,#177640	:
8415	037106	001365			BNE	19\$	:

CORDBO KDJ11 B CLUSTER MAC11 30(1046) 05-APR-84 16:45 PAGE 155
 CORDAB.P11 05-APR-84 16:45 MEMORY MANAGEMENT TESTS

SEQ 0154

```

8416 037110 022327 125252 22$: CMP (R3)+,#125252 ;CHECK UPARS
8417 037114 001401 BEQ 23$ ;
8418 037116 104002 ERROR +2 ;MMU ERROR
8419 ;
8420 037120 022327 052525 23$: CMP (R3)+,#52525 ;
8421 037124 001401 BEQ 24$ ;
8422 037126 104002 ERROR +2 ;MMU ERROR
8423 ;
8424 037130 020327 177700 24$: CMP R3,#177700 ;
8425 037134 001365 BNE 22$ ;
8426 ;
8427 ;REVERSE ALTERNATING PATTERN
8428 ;
8429 037136 012700 172200 MOV #172200,R0 ;LOAD SPDRS WITH REVERSE PATTERN
8430 037142 012720 125012 25$: MOV #125012,(R0)+ ;
8431 037146 012720 052404 MOV #52404,(R0)+ ;
8432 037152 020027 172240 CMP R0,#172240 ;
8433 037156 001371 BNE 25$ ;
8434 037160 012720 052525 26$: MOV #52525,(R0)+ ;LOAD SPARS WITH REVERSE PATTERN
8435 037164 012720 125252 MOV #125252,(R0)+ ;
8436 037170 020027 172300 CMP R0,#172300 ;
8437 037174 001371 BNE 26$ ;
8438 037176 012720 125012 27$: MOV #125012,(R0)+ ;LOAD KPDRS WITH REVERSE PATTERN
8439 037202 012720 052404 MOV #52404,(R0)+ ;
8440 037206 020027 172340 CMP R0,#172340 ;
8441 037212 001371 BNE 27$ ;
8442 037214 012720 052525 28$: MOV #52525,(R0)+ ;LOAD KPARS WITH REVERSE PATTERN
8443 037220 012720 125252 MOV #125252,(R0)+ ;
8444 037224 020027 172400 CMP R0,#172400 ;
8445 037230 001371 BNE 28$ ;
8446 037232 012700 177600 MOV #177600,R0 ;LOAD UPDRS WITH REVERSE PATTERN
8447 037236 012720 125012 29$: MOV #125012,(R0)+ ;
8448 037242 012720 052404 MOV #52404,(R0)+ ;
8449 037246 020027 177640 CMP R0,#177640 ;
8450 037252 001371 BNE 29$ ;
8451 037254 012720 052525 30$: MOV #52525,(R0)+ ;LOAD UPARS WITH REVERSE PATTERN
8452 037260 012720 125252 MOV #125252,(R0)+ ;
8453 037264 020027 177700 CMP R0,#177700 ;
8454 037270 001371 BNE 30$ ;
8455 ;
8456 037272 012703 172200 MOV #172200,R3 ;CHECK SPDRS
8457 037276 022327 125012 31$: CMP (R3)+,#125012 ;
8458 037302 001401 BEQ 32$ ;
8459 037304 104002 ERROR +2 ;MMU ERROR
8460 ;
8461 037306 022327 052404 32$: CMP (R3)+,#52404 ;
8462 037312 001401 BEQ 33$ ;
8463 037314 104002 ERROR +2 ;MMU ERROR
8464 ;
8465 037316 020327 172240 33$: CMP R3,#172240 ;
8466 037322 001365 BNE 31$ ;
8467 037324 022327 052525 34$: CMP (R3)+,#52525 ;CHECK SPARS
8468 037330 001401 BEQ 35$ ;
8469 037332 104002 ERROR +2 ;MMU ERROR
8470 ;
8471 037334 022327 125252 35$: CMP (R3)+,#125252 ;

```



```

8472 037340 001401      BEQ      36$      ;
8473 037342 104002      ERROR    +2      ;MMU ERROR
8474                                     ;
8475 037344 020327 172300 36$:  CMP      R3,#172300      ;
8476 037350 001365      BNE      34$      ;
8477 037352 022327 125012 37$:  CMP      (R3)+,#125012    ;CHECK KPDRS
8478 037356 001401      BEQ      38$      ;
8479 037360 104002      ERROR    +2      ;MMU ERROR
8480                                     ;
8481 037362 022327 052404 38$:  CMP      (R3)+,#52404    ;
8482 037366 001401      BEQ      39$      ;
8483 037370 104002      ERROR    +2      ;MMU ERROR
8484                                     ;
8485 037372 020327 172340 39$:  CMP      R3,#172340      ;
8486 037376 001365      BNE      37$      ;
8487 037400 022327 052525 40$:  CMP      (R3)+,#52525    ;CHECK KPARS
8488 037404 001401      BEQ      41$      ;
8489 037406 104002      ERROR    +2      ;MMU ERROR
8490                                     ;
8491 037410 022327 125252 41$:  CMP      (R3)+,#125252    ;
8492 037414 001401      BEQ      42$      ;
8493 037416 104002      ERROR    +2      ;MMU ERROR
8494                                     ;
8495 037420 020327 172400 42$:  CMP      R3,#172400      ;
8496 037424 001365      BNE      40$      ;
8497 037426 012703 177600  MOV      #177600,R3      ;CHECK UPDRS
8498 037432 022327 125012 43$:  CMP      (R3)+,#125012    ;
8499 037436 001401      BEQ      44$      ;
8500 037440 104002      ERROR    +2      ;MMU ERROR
8501                                     ;
8502 037442 022327 052404 44$:  CMP      (R3)+,#52404    ;
8503 037446 001401      BEQ      45$      ;
8504 037450 104002      ERROR    +2      ;MMU ERROR
8505                                     ;
8506 037452 020327 177640 45$:  CMP      R3,#177640      ;
8507 037456 001365      BNE      43$      ;
8508 037460 022327 052525 46$:  CMP      (R3)+,#52525    ;CHECK UPARS
8509 037464 001401      BEQ      47$      ;
8510 037466 104002      ERROR    +2      ;MMU ERROR
8511                                     ;
8512 037470 022327 125252 47$:  CMP      (R3)+,#125252    ;
8513 037474 001401      BEQ      48$      ;
8514 037476 104002      ERROR    +2      ;MMU ERROR
8515                                     ;
8516 037500 020327 177700 48$:  CMP      R3,#177700      ;
8517 037504 001365      BNE      46$      ;
8518                                     ;
8519 037506      TSHMUS:
8520      ;
8521 037506 012737 160000 177572  MOV      #160000,#0177572    ;LOAD MMRO<15:13>-111
8522 037514 005067 143256      CLR      FLAG      ;CLEAR MMU ABORT FLAG
8523 037520 013700 177572      MOV      @SRO,R0      ;SAVE SRO IN R0
8524 037524 042700 000176      BIC      #176,R0      ;CLEAR UNDEFINED BITS FROM SRO
8525 037530 020027 160000      CMP      R0,#160000    ;CHECK MMRO
8526 037534 001401      BEQ      1$      ;
8527 037536 104002      ERROR    +2      ;MMU ERROR

```

N12

COKDABO KDJ11-H CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 157
 COKDAB.P11 05-APR-84 16:45 MEMORY MANAGEMENT TESTS

SEQ 0156

```

8528
8529 037540 005037 177572 1$: CLR @#177572 ; LOAD MMRO=0
8530 037544 013700 177572 MOV @#SRO,R0 ; SAVE SRO IN R0
8531 037550 042700 000176 BIC #176,R0 ; CLEAR UNDEFINED BITS FROM SRO
8532 037554 020027 000000 CMP R0,#0 ; CHECK MMRO
8533 037560 001401 BEQ 2$ ;
8534 037562 104002 ERROR +2 ;MMU ERROR
8535
8536 037564 012737 120000 177572 2$: MOV #120000,@#177572 ; LOAD MMRO<15:13>=101
8537 037572 013700 177572 MOV @#SRO,R0 ; SAVE SRO IN R0
8538 037576 042700 000176 BIC #176,R0 ; CLEAR UNDEFINED BITS FROM SRO.
8539 037602 020027 120000 CMP R0,#120000 ; CHECK MMRO
8540 037606 001401 BEQ 3$ ;
8541 037610 104002 ERROR +2 ;MMU ERROR
8542
8543 037612 012737 040000 177572 3$: MOV #40000,@#177572 ; LOAD MMRO<15:13>=010
8544 037620 013700 177572 MOV @#SRO,R0 ; SAVE SRO IN R0
8545 037624 042700 000176 BIC #176,R0 ; CLEAR UNDEFINED BITS FROM SRO.
8546 037630 020027 040000 CMP R0,#40000 ; CHECK MMRO
8547 037634 001401 BEQ 4$ ;
8548 037636 104002 ERROR +2 ;MMU ERROR
8549 037640 4$:
8550 037640 TSMU6:
8551 ; TEST MMR3 BITS 5-0
8552 037640 005037 177572 CLR @#177572 ;MMU OFF
8553 037644 005067 143126 CLR FLAG ;CLEAR MMU ABORT FLAG
8554 037650 012737 000077 172516 MOV #77,@#172516 ;LOAD MMR3<5:0>=77
8555 037656 023727 172516 000077 CMP @#172516,#77 ;CHECK MMR3
8556 037664 001401 BEQ 1$ ;
8557 037666 104002 ERROR +2 ;MMU ERROR
8558 037670 005037 172516 1$: CLR @#172516 ;LOAD MMR3<5:0>=0
8559 037674 023727 172516 000000 CMP @#172516,#0 ;CHECK MMR3
8560 037702 001401 BEQ 2$ ;
8561 037704 104002 ERROR +2 ;MMU ERROR
8562 037706 012737 000052 172516 2$: MOV #52,@#172516 ;LOAD MMR3<5:0>=52
8563 037714 023727 172516 000052 CMP @#172516,#52 ;CHECK MMR3
8564 037722 001401 BEQ 3$ ;
8565 037724 104002 ERROR +2 ;MMU ERROR
8566 037726 012737 000025 172516 3$: MOV #25,@#172516 ;LOAD MMR3<5:0>=25
8567 037734 023727 172516 000025 CMP @#172516,#25 ;CHECK MMR3
8568 037742 001401 BEQ 4$ ;
8569 037744 104002 ERROR +2 ;MMU ERROR
8570 037746 4$:
8571 037746 TSMU6A:
8572 ; TEST MFPI (MOVE FROM PREVIOUS INST SPACE)
8573 037746 005037 177572 CLR @#177572 ;MMU OFF
8574 037752 005037 002776 CLR @#FLAG ;CLEAR MMU ABORT FLAG
8575 037756 012737 140000 177776 MOV #140000,@#177776 ;POINT TO USER SPACE
8576 037764 012706 001000 MOV #STB01,SP ;INIT THE USER STACK POINTER
8577 037770 010637 003002 MOV R6,@#SAVUSE ;SAVE USER SP
8578 037774 012737 040000 177776 MOV #40000,@#177776 ;POINT TO SUPERVISOR SPACE
8579 040002 012706 001000 MOV #STB01,SP ;INIT THE SUPERVISOR STACK POINTER
8580 040006 010637 003000 MOV R6,@#SAVSUP ;SAVE SUPERVISOR SP
8581 040012 012737 030000 177776 MOV #30000,@#177776 ;SETUP PSW
8582 040020 004767 073110 JSR PC,MMU ;INIT MMU
8583 040024 012737 000027 172516 MOV #27,@#172516 ;SETUP MMR3

```


CORDBO RD111 B CLUSTER MAC111 3001046) 05 APR 84 16:45 PAGE 158
 CORDAB, P11 05 APR 84 16:45 MEMORY MANAGEMENT TESTS

SEQ 0157

8584	040032	013746	000244			MOV	00244, (SP)	SAVE DATA AT TEST LOCATION
8585	040036	012746	177777			MOV	0177777, (SP)	PUT KNOWN DATA ON TOP OF STACK
8586	040042	012757	135072	000244		MOV	0135072, 00244	SETUP DATA AT TEST LOCATION
8587	040050	012767	077400	137547		MOV	077400, UOPDRO	SETUP UOPDRO TO ABORT
8588	040056	012703	000244			MOV	0244, R3	SETUP POINTER TO TEST LOCATION
8589	040062	005237	177572			INC	00177572	TURN MMU ON
8590	040066	006523				BE PT	(R3)	TEST INSTRUCTION
8591	040070	021737	030010	177776		CMP	030010, 00177776	IS PSW CORRECT
8592	040076	001401				BEQ	13	YES GO ON
8593	040100	104002				ERROR	12	MMU ERROR
8594								NO GO TO ERROR
8595	040102	005037	177572		13:	CLR	00177572	TURN MMU OFF
8596	040106	012737	140000	177776		MOV	0140000, 00177776	POINT TO USER SPACE
8597	040114	020637	003002			CMP	R6, 00SAVUSE	IS USER SP CORRECT
8598	040120	001401				BEQ	1003	YES GO ON
8599	040122	104002				ERROR	12	MMU ERROR
8600								NO GO TO ERROR
8601	040124	012737	040000	177776	1003:	MOV	040000, 00177776	POINT TO SUPERVISOR SPACE
8602	040132	020637	003000			CMP	R6, 00SAVSUP	IS SUPERVISOR SP CORRECT
8603	040136	001401				BEQ	2003	YES GO ON
8604	040140	104002				ERROR	12	MMU ERROR
8605								NO GO TO ERROR
8606	040142	023727	000244	135072	2003:	CMP	00244, 0135072	IS TEST DATA OK
8607	040150	001401				BEQ	23	YES GO ON
8608	040152	104002				ERROR	12	MMU ERROR
8609								NO GO TO ERROR
8610	040154	020327	000246		23:	CMP	R3, 0246	IS R3 CORRECT
8611	040160	001401				BEQ	33	YES GO ON
8612	040162	104002				ERROR	12	MMU ERROR
8613								NO GO TO ERROR
8614	040164	005037	177776		33:	CLR	00177776	SET PSW TO KERNEL MODE
8615	040170	022627	135072			CMP	(SP), 0135072	IS KERNEL STACK CORRECT
8616	040174	001401				BEQ	43	YES GO ON
8617	040176	104002				ERROR	12	MMU ERROR
8618								NO GO TO ERROR
8619	040200	021627	177777		43:	CMP	(SP), 0177777	IS STACK CORRECT
8620	040204	001401				BEQ	53	YES GO ON
8621	040206	104002				ERROR	12	MMU ERROR
8622								NO GO TO ERROR
8623	040210	012737	030017	177776	53:	MOV	030017, 00177776	SETUP PSW
8624	040216	012737	173621	000244		MOV	0173621, 00244	SETUP TEST LOCATION
8625	040224	012701	000244			MOV	0244, R1	SETUP R1
8626	040230	005237	177572			INC	00177572	TURN MMU ON
8627	040234	006511				BE PT	(R1)	TEST INSTRUCTION
8628	040236	022737	030011	177776		CMP	030011, 00177776	IS PSW CORRECT
8629	040244	001401				BEQ	1033	YES GO ON
8630	040246	104002				ERROR	12	MMU ERROR
8631								NO GO TO ERROR
8632	040250	005037	177572		3001:	CLR	00177572	TURN MMU OFF
8633	040254	023727	000244	173621		CMP	00244, 0173621	IS TEST LOCATION CORRECT
8634	040262	001401				BEQ	3013	YES GO ON
8635	040264	104002				ERROR	12	MMU ERROR
8636								NO GO TO ERROR
8637	040266	020127	000244		3013:	CMP	R1, 0244	IS R1 CORRECT
8638	040272	001401				BEQ	4023	YES GO ON
8639	040274	104002				ERROR	12	MMU ERROR

8640							END GO TO ERROR
8641	040376	005037	177776	3073:	CIR	@0177776	;SET PSW TO KERNEL MODE
8642	040309	022627	175621		CMP	(SP)+,@175621	;IS STACK CORRECT
8643	040306	001401			BEG	3033	;YES GO ON
8644	040310	104002			ERROR	*	;MMU ERROR
8645							END GO TO ERROR
8646	040312	021627	177777	3033:	CMP	(SP),@177777	;IS STACK CORRECT
8647	040316	001401			BEG	3043	;YES GO ON
8648	040320	104002			ERROR	*	;MMU ERROR
8649							END GO TO ERROR
8650	040322	005003		3043:	CIR	R3	;SETUP SOURCE FOR NEXT TEST
8651	040324	005237	177572		INC	@177572	;TURN MMU ON
8652	040330	006503			MFPD	R3	;TEST INSTRUCTION
8653	040332	022737	000004	177776	CMP	@4,@0177776	;IS PSW CORRECT
8654	040340	001401			BEG	63	;YES GO ON
8655	040342	104002			ERROR	*	;MMU ERROR
8656							END GO TO ERROR
8657	040344	005037	177572	63:	CIR	@177572	;TURN MMU OFF
8658	040350	020327	000000		CMP	R3,#0	;IS R3 CORRECT
8659	040354	001401			BEG	73	;YES GO ON
8660	040356	104002			ERROR	*	;MMU ERROR
8661							END GO TO ERROR
8662	040360	022627	000000	73:	CMP	(SP)+,#0	;IS STACK CORRECT
8663	040364	001401			BEG	83	;YES GO ON
8664	040366	104002			ERROR	*	;MMU ERROR
8665							END GO TO ERROR
8666	040370	022627	177777	83:	CMP	(SP)+,@177777	;IS STACK CORRECT
8667	040374	001401			BEG	93	;YES GO ON
8668	040376	104002			ERROR	*	;MMU ERROR
8669							END GO TO ERROR
8670	040400	012637	000244	93:	MOV	(SP)+,@0244	;RESTORE TEST LOCATION
8671							
8672							
8673	040404			I			
8674				TSMH6B:			
8675	040404	005037	177572	I	TEST MFPD (MOVE FROM PREVIOUS DATA SPACE)		
8676	040410	005037	002776		CIR	@177572	;MMU OFF
8677	040414	012737	140000	177776	CIR	@0FLAG	;CLEAR MMU ABORT FLAG
8678	040422	010637	003002		MOV	@140000,@0177776	;POINT TO USER SPACE
8679	040426	012737	040000	177776	MOV	R6,@0SAVEUSE	;SAVE USER SP
8680	040434	010637	003000		MOV	@40000,@0177776	;POINT TO SUPERVISOR SPACE
8681	040440	012737	030000	177776	MOV	R6,@0SAVSUP	;SAVE SUPERVISOR SP
8682	040446	004767	072462		MOV	@30000,@0177776	;SETUP PSW
8683	040452	012737	000027	172516	JSR	PC,MMU	;INIT MMU
8684	040460	013746	000244		MOV	@27,@017516	;SETUP MHRS
8685	040464	012746	177777		MOV	@0244,(SP)	;SAVE DATA AT TEST LOCATION
8686	040470	012737	157002	000244	MOV	@177777,(SP)	;PUT KNOWN DATA ON TOP OF FLAGC
8687	040476	012767	077400	137074	MOV	@157002,@0244	;SETUP DATA AT TEST LOCATION
8688	040504	012703	000244		MOV	@7400,DIPDR0	;SETUP DIPDR0 TO ABORT
8689	040510	005237	177572		MOV	@44,R3	;SETUP POINTER TO TEST LOCATION
8690	040514	106523			INC	@0177776	;TURN MMU ON
8691	040516	022737	030010	177776	MFPD	(R3)	;TEST INSTRUCTION
8692	040524	001401			CMP	@30010,@0177776	;IS PSW CORRECT
8693	040526	104002			BEG	13	;YES GO ON
8694					ERROR	*	;MMU ERROR
8695	040530	005037	177572	13:	CIR	@0177776	;TURN MMU OFF

CORDBO RDJ11 B CLESTER MACY11 30(1046) 05 APR 84 16:45 PAGE 160
 CORDAB,P11 05 APR 84 16:45 MEMORY MANAGEMENT TESTS

SEQ 0159

8696	040534	012737	140000	177776		MOV	0140000,00177776	;POINT TO USER SPACE
8697	040542	020657	003002			CMP	R6,005AVUSE	;IS USER SP CORRECT
8698	040546	001401				BEQ	100\$	;YES GO ON
8699	040550	104002				ERROR	02	;MMU ERROR
8700								;NO GO TO ERROR
8701	040552	012737	040000	177776	100\$:	MOV	040000,00177776	;POINT TO SUPERVISOR SPACE
8702	040560	020657	003000			CMP	R6,005AVSUP	;IS SUPERVISOR SP CORRECT
8703	040564	001401				BEQ	200\$	;YES GO ON
8704	040566	104002				ERROR	02	;MMU ERROR
8705								;NO GO TO ERROR
8706	040570	023127	000244	157002	200\$:	CMP	00244,0157002	;IS TEST DATA OK
8707	040576	001401				BEQ	2\$	;YES GO ON
8708	040600	104002				ERROR	02	;MMU ERROR
8709								;NO GO TO ERROR
8710	040602	020327	000246		2\$:	CMP	R3,0246	;IS R3 CORRECT
8711	040606	001401				BEQ	3\$	;YES GO ON
8712	040610	104002				ERROR	02	;MMU ERROR
8713								;NO GO TO ERROR
8714	040612	005037	177776		3\$:	CLR	00177776	;SET PSW TO KERNEL MODE
8715	040616	022627	157002			CMP	(SP),0157002	;IS KERNEL STACK CORRECT
8716	040622	001401				BEQ	4\$	;YES GO ON
8717	040624	104002				ERROR	02	;MMU ERROR
8718								;NO GO TO ERROR
8719	040626	021627	177777		4\$:	CMP	(SP),0177777	;IS STACK CORRECT
8720	040632	001401				BEQ	5\$	;YES GO ON
8721	040634	104002				ERROR	02	;MMU ERROR
8722								;NO GO TO ERROR
8723	040636	012737	030017	177776	5\$:	MOV	030017,00177776	;SETUP PSW
8724	040644	012737	103456	000244		MOV	0103456,00244	;SETUP TEST LOCATION
8725	040652	012701	000244			MOV	0244,R1	;SETUP R1
8726	040656	005237	177572			INC	00177572	;TURN MMU ON
8727	040662	106511				MF PD	(R1)	;TEST INSTRUCTION
8728	040664	022737	030011	177776		CMP	030011,00177776	;IS PSW CORRECT
8729	040672	001401				BEQ	300\$	;YES GO ON
8730	040674	104002				ERROR	02	;MMU ERROR
8731								;NO GO TO ERROR
8732	040676	005037	177572		300\$:	CLR	00177572	;TURN MMU OFF
8733	040702	023727	000244	103456		CMP	00244,0103456	;IS TEST LOCATION CORRECT
8734	040710	001401				BEQ	301\$	;YES GO ON
8735	040712	104002				ERROR	02	;MMU ERROR
8736								;NO GO TO ERROR
8737	040714	020127	000244		301\$:	CMP	R1,0244	;IS R1 CORRECT
8738	040720	001401				BEQ	302\$	;YES GO ON
8739	040722	104002				ERROR	02	;MMU ERROR
8740								;NO GO TO ERROR
8741	040724	005037	177776		302\$:	CLR	00177776	;SET PSW TO KERNEL MODE
8742	040730	022627	103456			CMP	(SP),0103456	;IS STACK CORRECT
8743	040734	001401				BEQ	303\$	;YES GO ON
8744	040736	104002				ERROR	02	;MMU ERROR
8745								;NO GO TO ERROR
8746	040740	021627	177777		303\$:	CMP	(SP),0177777	;IS STACK CORRECT
8747	040744	001401				BEQ	304\$	;YES GO ON
8748	040746	104002				ERROR	02	;MMU ERROR
8749								;NO GO TO ERROR
8750	040750	012737	030017	177776	304\$:	MOV	030017,00177776	;SETUP PSW
8751	040756	012737	113672	000244		MOV	0113672,00244	;SETUP TEST LOCATION

CORDBO RDJ11 B CLUSTER MACY11 30(1046) 05 APR 84 16:45 PAGE 161
COKDAB,P11 05-APR 84 16:45 MEMORY MANAGEMENT TESTS

SEQ 0160

8752	040764	012701	000246		MOV	R1,0246	; SETUP R1
8753	040770	005237	177572		INC	00177572	; TURN MMU ON
8754	040774	106541			MFPD	(R1)	; TEST INSTRUCTION
8755	040776	022737	030011	177776	CMP	030011,00177776	; IS PSW CORRECT
8756	041004	001401			BEQ	400\$	; YES GO ON
8757	041006	104002			ERROR	+2	; MMU ERROR
8758							; NO GO TO ERROR
8759	041010	005037	177572		CLR	00177572	; TURN MMU OFF
8760	041014	023727	000244	113672	CMP	00244,0113672	; IS TEST LOCATION CORRECT
8761	041022	001401			BEQ	401\$	; YES GO ON
8762	041024	104002			ERROR	+2	; MMU ERROR
8763							; NO GO TO ERROR
8764	041026	020127	000244		CMP	R1,0244	; IS R1 CORRECT
8765	041032	001401			BEQ	402\$	; YES GO ON
8766	041034	104002			ERROR	+2	; MMU ERROR
8767							; NO GO TO ERROR
8768	041036	005037	177776		CLR	00177776	; SET PSW TO KERNEL MODE
8769	041042	022627	113672		CMP	(SP)+,0113672	; IS STACK CORRECT
8770	041046	001401			BEQ	403\$	; YES GO ON
8771	041050	104002			ERROR	+2	; MMU ERROR
8772							; NO GO TO ERROR
8773	041052	021627	177777		CMP	(SP),0177777	; IS STACK CORRECT
8774	041056	001401			BEQ	404\$	; YES GO ON
8775	041060	104002			ERROR	+2	; MMU ERROR
8776							; NO GO TO ERROR
8777	041062	005003			CLR	R3	; SETUP SOURCE FOR NEXT TEST
8778	041064	005237	177572		INC	00177572	; TURN MMU ON
8779	041070	106503			MFPD	R3	; TEST INSTRUCTION
8780	041072	022737	000004	177776	CMP	04,00177776	; IS PSW CORRECT
8781	041100	001401			BEQ	6\$	; YES GO ON
8782	041102	104002			ERROR	+2	; MMU ERROR
8783							; NO GO TO ERROR
8784	041104	005037	177572		CLR	00177572	; TURN MMU OFF
8785	041110	020327	000000		CMP	R3,00	; IS R3 CORRECT
8786	041114	001401			BEQ	7\$	; YES GO ON
8787	041116	104002			ERROR	+2	; MMU ERROR
8788							; NO GO TO ERROR
8789	041120	022627	000000		CMP	(SP)+,00	; IS STACK CORRECT
8790	041124	001401			BEQ	8\$	; YES GO ON
8791	041126	104002			ERROR	+2	; MMU ERROR
8792							; NO GO TO ERROR
8793	041130	022627	177777		CMP	(SP)+,0177777	; IS STACK CORRECT
8794	041134	001401			BEQ	9\$	; YES GO ON
8795	041136	104002			ERROR	+2	; MMU ERROR
8796							; NO GO TO ERROR
8797	041140	012637	000244		MOV	(SP)+,00244	; RESTORE TEST LOCATION
8798							
8799							
8800	041144				; TSM6C:		
8801					; TEST MPI (MOVE TO PREVIOUS INSTRUCTION SPACE)		
8802	041144	005037	177572		CLR	00177572	; MMU OFF
8803	041150	005037	002776		CLR	0001AG	; CLEAR MMU ABORT FLAG
8804	041154	012737	140000	177776	MOV	0140000,00177776	; POINT TO USER SPACE
8805	041162	010637	003002		MOV	R6,00SAVUSE	; SAVE USER SP
8806	041166	012737	040000	177776	MOV	040000,00177776	; POINT TO SUPERVISOR SPACE
8807	041174	010637	003000		MOV	R6,00SAVSUP	; SAVE SUPERVISOR SP

CORDBO KDJ11 B CLUSTER MAC11 30(1046) 05-APR-84 16:45 PAGE 162
 CORDAB,P11 05-APR-84 16:45 MEMORY MANAGEMENT TESTS

SEQ 0161

8808	041200	012737	030000	177776		MOV	#30000,#0177776	;SETUP PSW
8809	041206	004767	071722			JSR	PC,MMU	;INIT MMU
8810	041212	012737	000027	172516		MOV	#27,#0172516	;SETUP MMR3
8811	041220	013746	000244			MOV	#0244, (SP)	;SAVE DATA AT TEST LOCATION
8812	041224	012746	177777			MOV	#177777, -(SP)	;PUT KNOWN DATA ON STACK
8813	041230	012746	120413			MOV	#120413, -(SP)	;PUT TEST DATA ON STACK
8814	041234	012737	177777	000244		MOV	#177777,#0244	;PUT KNOWN DATA AT TEST LOCATION
8815	041242	012767	077400	136350		MOV	#77400,UDPDRO	;SETUP UDPDRO TO ABORT
8816	041250	012703	000244			MOV	#244,R3	;SETUP POINTER TO TEST LOCATION
8817	041254	005237	177572			INC	#0177572	;TURN MMU ON
8818	041260	006623				MTP1	(R3),	;TEST INSTRUCTION
8819	041262	022737	030010	177776		CMP	#30010,#0177776	;IS PSW CORRECT
8820	041270	001401				BEQ	1\$	;YES GO ON
8821	041272	104002				ERROR	+2	;MMU ERROR
8822								;NO GO TO ERROR
8823	041274	005037	177572		1\$:	CLR	#0177572	;TURN MMU OFF
8824	041300	012737	140000	177776		MOV	#140000,#0177776	;POINT TO USER SPACE
8825	041306	020637	003002			CMP	R6,#0SAVUSE	;IS USER SP CORRECT
8826	041312	001401				BEQ	100\$	;YES GO ON
8827	041314	104002				ERROR	+2	;MMU ERROR
8828								;NO GO TO ERROR
8829	041316	012737	040000	177776	100\$:	MOV	#40000,#0177776	;POINT TO SUPERVISOR SPACE
8830	041324	020637	003000			CMP	R6,#0SAVSUP	;IS SUPERVISOR SP CORRECT
8831	041330	001401				BEQ	200\$	;YES GO ON
8832	041332	104002				ERROR	+2	;MMU ERROR
8833								;NO GO TO ERROR
8834	041334	023727	000244	120413	200\$:	CMP	#0244,#120413	;IS TEST LOCATION CORRECT
8835	041342	001401				BEQ	2\$	;YES GO ON
8836	041344	104002				ERROR	+2	;MMU ERROR
8837								;NO GO TO ERROR
8838	041346	020327	000246		2\$:	CMP	R3,#0246	;IS R3 CORRECT
8839	041352	001401				BEQ	3\$	;YES GO ON
8840	041354	104002				ERROR	+2	;MMU ERROR
8841								;NO GO TO ERROR
8842	041356	005037	177776		3\$:	CLR	#0177776	;SET PSW TO KERNEL MODE
8843	041362	021627	177777			CMP	(SP),#0177777	;IS KERNEL STACK CORRECT
8844	041366	001401				BEQ	4\$	;YES GO ON
8845	041370	104002				ERROR	+2	;MMU ERROR
8846								;NO GO TO ERROR
8847	041372	012737	030017	177776	4\$:	MOV	#30017,#0177776	;SETUP PSW
8848	041400	012746	145121			MOV	#145121, (SP)	;SETUP TEST DATA
8849	041404	012701	000244			MOV	#244,R1	;SETUP R1
8850	041410	005237	177572			INC	#0177572	;TURN MMU ON
8851	041414	006611				MTP1	(R1)	;TEST INSTRUCTION
8852	041416	022737	030011	177776		CMP	#30011,#0177776	;IS PSW CORRECT
8853	041424	001401				BEQ	300\$	;YES GO ON
8854	041426	104002				ERROR	+2	;MMU ERROR
8855								;NO GO TO ERROR
8856	041430	005037	177572		300\$:	CLR	#0177572	;TURN MMU OFF
8857	041434	023727	000244	145121		CMP	#0244,#145121	;IS TEST LOCATION CORRECT
8858	041442	001401				BEQ	301\$	;YES GO ON
8859	041444	104002				ERROR	+2	;MMU ERROR
8860								;NO GO TO ERROR
8861	041446	020127	000244		301\$:	CMP	R1,#0244	;IS R1 CORRECT
8862	041452	001401				BEQ	302\$	;YES GO ON
8863	041454	104002				ERROR	+2	;MMU ERROR

```

8864
8865 041456 005037 177776 302$: CLR @0177776 ;NO GO TO ERROR
8866 041462 021627 177777 CMP (SP),@177777 ;SET PSW TO KERNEL MODE
8867 041466 001401 BEQ 304$ ;IS STACK CORRECT
8868 041470 104002 ERROR +2 ;YES GO ON
8869 ;MMU ERROR
8870 041472 012737 030017 177776 304$: MOV @30017,@0177776 ;NO GO TO ERROR
8871 041500 012746 122347 MOV @122347,-(SP) ;SETUP PSW
8872 041504 012701 000246 MOV @246,R1 ;SETUP TEST DATA
8873 041510 005237 177572 INC @0177572 ;SETUP R1
8874 041514 006641 MTPI -(R1) ;TURN MMU ON
8875 041516 022737 030011 177776 CMP @30011,@0177776 ;TEST INSTRUCTION
8876 041524 001401 BEQ 400$ ;IS PSW CORRECT
8877 041526 104002 ERROR +2 ;YES GO ON
8878 ;MMU ERROR
8879 041530 005037 177572 400$: CLR @0177572 ;NO GO TO ERROR
8880 041534 023727 000244 122347 CMP @0244,@122347 ;TURN MMU OFF
8881 041542 001401 BEQ 401$ ;IS TEST LOCATION CORRECT
8882 041544 104002 ERROR +2 ;YES GO ON
8883 ;MMU ERROR
8884 041546 020127 000244 401$: CMP R1,@244 ;NO GO TO ERROR
8885 041552 001401 BEQ 402$ ;IS R1 CORRECT
8886 041554 104002 ERROR +2 ;YES GO ON
8887 ;MMU ERROR
8888 041556 005037 177776 402$: CLR @0177776 ;NO GO TO ERROR
8889 041562 021627 177777 CMP (SP),@177777 ;SET PSW TO KERNEL MODE
8890 041566 001401 BEQ 404$ ;IS STACK CORRECT
8891 041570 104002 ERROR +2 ;YES GO ON
8892 ;MMU ERROR
8893 041572 005046 404$: CLR -(SP) ;NO GO TO ERROR
8894 041574 005237 177572 INC @0177572 ;SETUP STACK FOR NEXT TEST
8895 041600 006603 MTPI R3 ;TURN MMU ON
8896 041602 022737 000004 177776 CMP @4,@0177776 ;TEST INSTRUCTION
8897 041610 001401 BEQ 5$ ;IS PSW CORRECT
8898 041612 104002 ERROR +2 ;YES GO ON
8899 ;MMU ERROR
8900 041614 005037 177572 5$: CLR @0177572 ;NO GO TO ERROR
8901 041620 020327 000000 CMP R3,@0 ;TURN MMU OFF
8902 041624 001401 BEQ 6$ ;IS R3 CORRECT
8903 041626 104002 ERROR +2 ;YES GO ON
8904 ;MMU ERROR
8905 041630 022627 177777 6$: CMP (SP)+,@177777 ;NO GO TO ERROR
8906 041634 001401 BEQ 7$ ;IS STACK CORRECT
8907 041636 104002 ERROR +2 ;YES GO ON
8908 ;MMU ERROR
8909 041640 012637 000244 7$: MOV (SP)+,@0244 ;NO GO TO ERROR
8910 ;RESTORE TEST LOCATION
8911 ;
8912 041644 ;TSMMD:
8913 ;
8914 041644 005037 177572 ;TEST MTPD (MOVE TO PREVIOUS DATA SPACE)
8915 041650 005037 002776 CLR @0177572 ;MMU OFF
8916 041654 012737 140000 177776 CLR @0FLAG ;CLEAR MMU ABORT FLAG
8917 041662 010637 003002 MOV @140000,@0177776 ;POINT TO USER SPACE
8918 041666 012737 040000 177776 MOV R6,@0SAVE ;SAVE USER SP
8919 041674 010637 003000 MOV R6,@0SAVEUP ;POINT TO SUPERVISOR SPACE
8920 ;SAVE SUPERVISOR SP

```


CORDBO RDJ11 B CLUSTER MACY11 30(1046) 05 APR 84 16:45 PAGE 164
 CORDAB.P11 05-APR 84 16:45 MEMORY MANAGEMENT TESTS

SEQ 0163

8920	041700	012737	030000	177776		MOV	#30000,#177776	;SETUP PSW
8921	041706	004767	071222			JSR	PC,MMU	;INIT MMU
8922	041712	012737	000027	172516		MOV	#27,#172516	;SETUP MMR3
8923	041720	013746	000244			MOV	#244,(SP)	;SAVE DATA AT TEST LOCATION
8924	041724	012746	177777			MOV	#177777,-(SP)	;PUT KNOWN DATA ON STACK
8925	041730	012746	100004			MOV	#100004,-(SP)	;PUT TEST DATA ON STACK
8926	041734	012737	177777	000244		MOV	#177777,#244	;PUT KNOWN DATA AT TEST LOCATION
8927	041742	012767	077400	135630		MOV	#77400,UIPDRO	;SETUP UIPDRO TO ABORT
8928	041750	012703	000244			MOV	#244,R3	;SETUP POINTER TO TEST LOCATION
8929	041754	005237	177572			INC	#177572	;TURN MMU ON
8930	041760	106623				MTPD	(R3),	;TEST INSTRUCTION
8931	041762	022737	030010	177776		CMP	#30010,#177776	;IS PSW CORRECT
8932	041770	001401				BEQ	1\$	;YES GO ON
8933	041772	104002				ERROR	*2	;MMU ERROR
8934								;NO GO TO ERROR
8935	041774	005037	177572		1\$:	CLR	#177572	;TURN MMU OFF
8936	042000	012737	140000	177776		MOV	#140000,#177776	;POINT TO USER SPACE
8937	042006	020637	003002			CMP	R6,#SAVUSE	;IS USER SP CORRECT
8938	042012	001401				BEQ	100\$	;YES GO ON
8939	042014	104002				ERROR	*2	;MMU ERROR
8940								;NO GO TO ERROR
8941	042016	012737	040000	177776	100\$:	MOV	#40000,#177776	;POINT TO SUPERVISOR SPACE
8942	042024	020637	003000			CMP	R6,#SAVSUP	;IS SUPERVISOR SP CORRECT
8943	042030	001401				BEQ	200\$	;YES GO ON
8944	042032	104002				ERROR	*2	;MMU ERROR
8945								;NO GO TO ERROR
8946	042034	023727	000244	100004	200\$:	CMP	#244,#100004	;IS TEST LOCATION CORRECT
8947	042042	001401				BEQ	2\$	;YES GO ON
8948	042044	104002				ERROR	*2	;MMU ERROR
8949								;NO GO TO ERROR
8950	042046	020327	000246		2\$:	CMP	R3,#246	;IS R3 CORRECT
8951	042052	001401				BEQ	3\$	;YES GO ON
8952	042054	104002				ERROR	*2	;MMU ERROR
8953								;NO GO TO ERROR
8954	042056	005037	177776		3\$:	CLR	#177776	;SET PSW TO KERNEL MODE
8955	042062	021627	177777			CMP	(SP),#177777	;IS KERNEL STACK CORRECT
8956	042068	001401				BEQ	4\$	;YES GO ON
8957	042070	104002				ERROR	*2	;MMU ERROR
8958								;NO GO TO ERROR
8959	042072	012737	030017	177776	4\$:	MOV	#30017,#177776	;SETUP PSW
8960	042100	012746	100737			MOV	#100737,-(SP)	;SETUP TEST DATA
8961	042104	012701	000244			MOV	#244,R1	;SETUP R1
8962	042110	005237	177572			INC	#177572	;TURN MMU ON
8963	042114	106611				MTPD	(R1)	;TEST INSTRUCTION
8964	042116	022737	030011	177776		CMP	#30011,#177776	;IS PSW CORRECT
8965	042124	001401				BEQ	300\$	;YES GO ON
8966	042126	104002				ERROR	*2	;MMU ERROR
8967								;NO GO TO ERROR
8968	042130	005037	177572		300\$:	CLR	#177572	;TURN MMU OFF
8969	042134	023727	000244	100737		CMP	#244,#100737	;IS TEST LOCATION CORRECT
8970	042142	001401				BEQ	301\$	;YES GO ON
8971	042144	104002				ERROR	*2	;MMU ERROR
8972								;NO GO TO ERROR
8973	042146	020127	000244		301\$:	CMP	R1,#244	;IS R1 CORRECT
8974	042152	001401				BEQ	302\$	;YES GO ON
8975	042154	104002				ERROR	*2	;MMU ERROR

PC	Op	Op2	Op3	Op4	Op5	Op6	Op7	Op8	Op9	Op10	Op11	Op12	Op13	Op14	Op15	Op16	Op17	Op18	Op19	Op20	Op21	Op22	Op23	Op24	Op25	Op26	Op27	Op28	Op29	Op30	Op31	Op32	Op33	Op34	Op35	Op36	Op37	Op38	Op39	Op40	Op41	Op42	Op43	Op44	Op45	Op46	Op47	Op48	Op49	Op50	Op51	Op52	Op53	Op54	Op55	Op56	Op57	Op58	Op59	Op60	Op61	Op62	Op63	Op64	Op65	Op66	Op67	Op68	Op69	Op70	Op71	Op72	Op73	Op74	Op75	Op76	Op77	Op78	Op79	Op80	Op81	Op82	Op83	Op84	Op85	Op86	Op87	Op88	Op89	Op90	Op91	Op92	Op93	Op94	Op95	Op96	Op97	Op98	Op99	Op100	Op101	Op102	Op103	Op104	Op105	Op106	Op107	Op108	Op109	Op110	Op111	Op112	Op113	Op114	Op115	Op116	Op117	Op118	Op119	Op120	Op121	Op122	Op123	Op124	Op125	Op126	Op127	Op128	Op129	Op130	Op131	Op132	Op133	Op134	Op135	Op136	Op137	Op138	Op139	Op140	Op141	Op142	Op143	Op144	Op145	Op146	Op147	Op148	Op149	Op150	Op151	Op152	Op153	Op154	Op155	Op156	Op157	Op158	Op159	Op160	Op161	Op162	Op163	Op164	Op165	Op166	Op167	Op168	Op169	Op170	Op171	Op172	Op173	Op174	Op175	Op176	Op177	Op178	Op179	Op180	Op181	Op182	Op183	Op184	Op185	Op186	Op187	Op188	Op189	Op190	Op191	Op192	Op193	Op194	Op195	Op196	Op197	Op198	Op199	Op200	Op201	Op202	Op203	Op204	Op205	Op206	Op207	Op208	Op209	Op210	Op211	Op212	Op213	Op214	Op215	Op216	Op217	Op218	Op219	Op220	Op221	Op222	Op223	Op224	Op225	Op226	Op227	Op228	Op229	Op230	Op231	Op232	Op233	Op234	Op235	Op236	Op237	Op238	Op239	Op240	Op241	Op242	Op243	Op244	Op245	Op246	Op247	Op248	Op249	Op250	Op251	Op252	Op253	Op254	Op255	Op256	Op257	Op258	Op259	Op260	Op261	Op262	Op263	Op264	Op265	Op266	Op267	Op268	Op269	Op270	Op271	Op272	Op273	Op274	Op275	Op276	Op277	Op278	Op279	Op280	Op281	Op282	Op283	Op284	Op285	Op286	Op287	Op288	Op289	Op290	Op291	Op292	Op293	Op294	Op295	Op296	Op297	Op298	Op299	Op300	Op301	Op302	Op303	Op304	Op305	Op306	Op307	Op308	Op309	Op310	Op311	Op312	Op313	Op314	Op315	Op316	Op317	Op318	Op319	Op320	Op321	Op322	Op323	Op324	Op325	Op326	Op327	Op328	Op329	Op330	Op331	Op332	Op333	Op334	Op335	Op336	Op337	Op338	Op339	Op340	Op341	Op342	Op343	Op344	Op345	Op346	Op347	Op348	Op349	Op350	Op351	Op352	Op353	Op354	Op355	Op356	Op357	Op358	Op359	Op360	Op361	Op362	Op363	Op364	Op365	Op366	Op367	Op368	Op369	Op370	Op371	Op372	Op373	Op374	Op375	Op376	Op377	Op378	Op379	Op380	Op381	Op382	Op383	Op384	Op385	Op386	Op387	Op388	Op389	Op390	Op391	Op392	Op393	Op394	Op395	Op396	Op397	Op398	Op399	Op400	Op401	Op402	Op403	Op404	Op405	Op406	Op407	Op408	Op409	Op410	Op411	Op412	Op413	Op414	Op415	Op416	Op417	Op418	Op419	Op420	Op421	Op422	Op423	Op424	Op425	Op426	Op427	Op428	Op429	Op430	Op431	Op432	Op433	Op434	Op435	Op436	Op437	Op438	Op439	Op440	Op441	Op442	Op443	Op444	Op445	Op446	Op447	Op448	Op449	Op450	Op451	Op452	Op453	Op454	Op455	Op456	Op457	Op458	Op459	Op460	Op461	Op462	Op463	Op464	Op465	

COKDABO KDJ11-B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 166
 COKDAB.P11 05-APR-84 16:45 MEMORY MANAGEMENT TESTS

SEQ 0165

9032	042374	005067	140410		CLR	SAVMR2	:
9033	042400	004767	070530		JSR	PC,MMU	;INIT MMU
9034	042404	012737	030000	177776	MOV	#30000,#177776	;SETUP PSW
9035	042412	012702	000200		MOV	#200,R3	:
9036	042416	012737	077400	177600	MOV	#77400,#177600	;SETUP FOR AN ABORT
9037	042424	004767	000164		JSR	PC,TS7	;CAUSE AN ABORT TO OCCUR AND
9038							;THEN CHECK IF ABORT FLAG REGISTERED
9039							;THIS EVENT AND CHECK IF STATUS REGS
9040							;CONTAINED EXPECTED VALUES.
9041							;IF NO ABORT OCCURRED THEN GO TO ERROR
9042							;OTHERWISE CONTINUE.
9043	042430	012737	077404	177600	MOV	#77404,#177600	;SETUP FOR AN ABORT
9044	042436	004767	000152		JSR	PC,TS7	;CAUSE AN ABORT TO OCCUR AND
9045							;THEN CHECK IF ABORT FLAG REGISTERED
9046							;THIS EVENT AND CHECK IF STATUS REGS
9047							;CONTAINED EXPECTED VALUES.
9048							;IF NO ABORT OCCURRED THEN GO TO ERROR
9049							;OTHERWISE CONTINUE.
9050	042442	012701	000220		MOV	#220,R1	:
9051	042446	004767	070462		JSR	PC,MMU	;INIT MMU
9052	042452	005003			CLR	R3	;SETUP MMR1 EXPECTED DATA
9053	042454	012767	000001	140314	MOV	#1,FLAG	;SETUP FLAG FOR AN ABORT
9054	042462	012737	000001	177572	MOV	#1,#177572	;TURN MMU ON
9055	042470	012737	100000	177776	MOV	#100000,#177776	;SETUP PSW FOR AN ABORT (ILLEGAL MODE)
9056	042476	012241			MOV	(R2)+,-(R1)	;CAUSE AN ABORT
9057	042500	004767	000220		JSR	PC,TSM7	;CHECK IF AN ABORT OCCURRED BY
9058							;CHECKING ABORT FLAG AND STATUS REGS
9059							;IF NO ABORT OCCURRED THEN GO TO ERROR
9060							;OTHERWISE CONTINUE.
9061	042504	005067	140274		CLR	SAVMR0	;CLEAR STATUS REGS SAVE AREAS
9062	042510	005067	140272		CLR	SAVMR1	:
9063	042514	005067	140270		CLR	SAVMR2	:
9064	042520	012703	000022		MOV	#22,R3	;SETUP MMR1 EXPECTED DATA
9065	042524	012767	000001	140244	MOV	#1,FLAG	;SETUP FLAG FOR AN ABORT
9066	042532	012737	000001	177572	MOV	#1,#177572	;TURN MMU ON
9067	042540	012737	020000	177776	MOV	#20000,#177776	;SETUP PSW FOR AN ABORT (ILLEGAL MODE)
9068	042546	006522			MFPI	(R2)+	;CAUSE AN ABORT
9069	042550	004767	000150		JSR	PC,TSM7	;CHECK IF AN ABORT OCCURRED BY
9070							;CHECKING ABORT FLAG AND STATUS REGS
9071							;IF NO ABORT OCCURRED THEN GO TO ERROR
9072							;OTHERWISE CONTINUE.
9073	042554	012737	030000	177776	MOV	#30000,#177776	;SETUP PSW
9074	042562	012737	077400	177600	MOV	#77400,#177600	;SETUP FOR AN ABORT
9075	042570	005037	177572		CLR	#177572	;MMU OFF
9076	042574	006522			MFPI	(R2)+	;TRY TO CAUSE AN ABORT
9077	042576	012603			MOV	(SP)+,R3	;POP THE STACK
9078	042600	012637	000216		MOV	(SP)+,#216	;RESTORE DATA AT TEST LOCATIONS
9079	042604	012637	000214		MOV	(SP)+,#214	:
9080							
9081	042610	000167	000154		JMP	TS7 IN	
9082							
9083							
9084							
9085	042614	012767	000001	140154	TS7:	MOV	#1,FLAG
9086	042622	012737	000001	177572		MOV	#1,#177572
9087	042630	010701				MOV	R7,R1

;ROUTINE TO CAUSE AND CHECK NONRESIDENT ABORTS

K13

COKDABO KDJ11 B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 167
 COKDAB.P11 05-APR-84 16:45 MEMORY MANAGEMENT TESTS

SEQ 0166

```

9088 042632 006522 MFPI (R2)+ ;CAUSE AN ABORT
9089 042634 022767 000000 140134 CMP #0,FLAG ;DID AN ABORT OCCUR
9090 042642 001401 BEQ OK7 ;IF YES GO ON
9091 042644 104002 ERROR +2 ;MMU ERROR
9092 ;IF NO GO TO ERROR
9093 042646 105067 140132 OK7: CLRB SAVMRO ;SETUP EXPECTED DATA
9094 042652 022767 100000 140124 CMP #100000,SAVMRO ;TEST MMRO FOR EXPECTED VALUE
9095 042660 001401 BEQ OKA7 ;IF OK THEN CONTINUE
9096 042662 104002 ERROR +2 ;MMU ERROR
9097 ;NOT OK THEN GO TO ERROR
9098 042664 026727 140116 000022 OKA7: CMP SAVMR1,#22 ;TEST MMR1 FOR EXPECTED VALUE
9099 042672 001401 BEQ OKAY7 ;IF OK THEN CONTINUE
9100 042674 104002 ERROR +2 ;MMU ERROR
9101 ;NOT OK THEN GO TO ERROR
9102 042676 026701 140106 OKAY7: CMP SAVMR2,R1 ;TEST MMR2 FOR EXPECTED VALUE
9103 042702 001401 BEQ OKAY7A ;IF OK THEN CONTINUE
9104 042704 104002 ERROR +2 ;MMU ERROR
9105 ;NOT OK THEN GO TO ERROR
9106 042706 005067 140072 OKAY7A: CLR SAVMRO ;CLEAR STATUS REGS SAVE AREAS
9107 042712 005067 140070 CLR SAVMR1 ;
9108 042716 005067 140066 CLR SAVMR2 ;
9109 042722 000207 RTS PC ;RETURN
9110 ;
9111 ;ROUTINE TO CHECK IF A NONRESIDENT ABORT OCCURRED
9112 ;
9113 042724 022767 000000 140044 TSM7: CMP #0,FLAG ;DID AN ABORT OCCUR
9114 042732 001401 BEQ TSMA ;IF YES GO ON
9115 042734 104002 ERROR +2 ;MMU ERROR
9116 ;IF NO THEN GO TO ERROR
9117 042736 042737 040377 003004 TSMA: BIC #40377,#SAVMRO ;SETUP EXPECTED DATA
9118 042744 022767 100000 140032 CMP #100000,SAVMRO ;TEST MMRO FOR EXPECTED VALUE
9119 042752 001401 BEQ TSMB ;IF OK THEN CONTINUE
9120 042754 104002 ERROR +2 ;MMU ERROR
9121 ;IF NO THEN GO TO ERROR
9122 042756 020367 140024 TSMB: CMP R3,SAVMR1 ;TEST MMR1 FOR EXPECTED VALUE
9123 042762 001401 BEQ TSMC ;IF OK THEN CONTINUE
9124 042764 104002 ERROR +2 ;MMU ERROR
9125 ;IF NOT OK THEN GO TO ERROR
9126 042766 000207 TSMC: RTS PC ;RETURN
9127 ;
9128 042770 000240 TS7FIN: NOP
9129 042772 TSMU8:
9130 ;
9131 042772 005037 177572 ;MMU OFF
9132 042776 005067 137774 CLR FLAG ;CLEAR MMU ABORT FLAG
9133 043002 013746 000244 MOV #244,-(SP) ;SAVE DATA AT TEST LOCATIONS
9134 043006 013746 000246 MOV #246,-(SP) ;
9135 043012 005067 137766 CLR SAVMRO ;CLEAR STATUS REGS SAVE AREAS
9136 043016 005067 137764 CLR SAVMR1 ;
9137 043022 005067 137762 CLR SAVMR2 ;
9138 043026 004767 070102 JSR PC,MMU ;INIT MMU
9139 043032 012737 030000 177776 MOV #30000,#177776 ;SETUP PSW
9140 043040 012702 000244 MOV #244,R2 ;
9141 043044 012737 077402 177600 MOV #77402,#177600 ;SETUP FOR AN ABORT
9142 043052 012746 000246 MOV #246,-(SP) ;PUSH DATA ONTO THE STACK
9143 043056 012767 000001 137772 MOV #1,FLAG ;SETUP FLAG FOR AN ABORT

```

COKDABO KDJ11 B CLUSTER MACY11 30(1046) 05 APR 84 16:45 PAGE 168
 COKDAB.P11 05-APR-84 16:45 MEMORY MANAGEMENT TESTS

SEQ 0167

```

9144 043064 012737 000001 177572 MOV #1,0#177572 ;TURN MMU ON
9145 043072 010701 MOV R7,R1 ;SAVE PC
9146 043074 006622 MTPI (R2)+ ;CAUSE ABORT
9147 043076 022767 000000 137672 CMP #0,FLAG ;DID ABORT OCCUR
9148 043104 001401 BEQ 1$ ;IF YES THEN GO ON
9149 043106 104002 ERROR +2 ;MMU ERROR
9150 ;IF NO THEN GO TO ERROR
9151 043110 105067 137670 1$: CLRB SAVMR0 ;SETUP EXPECTED DATA
9152 043114 022767 020000 137662 CMP #20000,SAVMR0 ;TEST MMRO FOR EXPECTED VALUE
9153 043122 001401 BEQ 2$ ;IF OK THEN CONTINUE
9154 043124 104002 ERROR +2 ;MMU ERROR
9155 ;OTHERWISE GO TO ERROR
9156 043126 022767 011026 137652 2$: CMP #11026,SAVMR1 ;TEST MMR1 FOR EXPECTED VALUE
9157 043134 001401 BEQ 3$ ;IF OK THEN CONTINUE
9158 043136 104002 ERROR +2 ;MMU ERROR
9159 ;OTHERWISE GO TO ERROR
9160 043140 020167 137644 3$: CMP R1,SAVMR2 ;TEST MMR2 FOR EXPECTED VALUE
9161 043144 001401 BEQ 4$ ;IF OK THEN CONTINUE
9162 043146 104002 ERROR +2 ;MMU ERROR
9163 ;OTHERWISE GO TO ERROR
9164 043150 012737 030000 177776 4$: MOV #30000,0#177776 ;SETUP PSW
9165 043156 012746 000002 MOV #2, -(SP) ;PUSH DATA ONTO STACK
9166 043162 006622 MTPI (R2)+ ;TRY TO CAUSE ABORT
9167 043164 012637 000246 MOV (SP)+,0#246 ;RESTORE DATA AT TEST LOCATIONS
9168 043170 012637 000244 MOV (SP)+,0#244 ;
9169
9170 043174 TSMMU9:
9171 ; TEST PAGE LENGTH ERROR ABORTS
9172 043174 005037 177572 CLR 0#177572 ;MMU OFF
9173 043200 005067 137572 CLR FLAG ;CLEAR MMU ABORT FLAG
9174 043204 005067 137574 CLR SAVMR0 ;CLEAR STATUS REGS SAVE AREAS
9175 043210 005067 137572 CLR SAVMR1 ;
9176 043214 005067 137570 CLR SAVMR2 ;
9177 043220 012737 030000 177776 MOV #30000,0#177776 ;SETUP PSW
9178 043226 004767 067702 JSR PC,MMU ;INIT MMU
9179 043232 012703 043514 MOV #PLE0,R3 ;LET R3, R1, AND R2 POINT TO THE
9180 043236 012701 043566 MOV #BNO,R1 ;UPWARD EXPANSION TABLES
9181 043242 012702 043636 MOV #ABORT0,R2 ;
9182 043246 012737 000026 172516 MOV #26,0#172516 ;DISABLE USER DATA SPACE
9183 043254 004767 000050 JSR PC,TSM9 ;TURN MMU ON
9184 ;DO RELOCATIONS FOR THE DIFFERENT
9185 ;VALUES OF THE PAGE LENGTH FIELD AND
9186 ;BLOCK NUMBER. IF AN ABORT OCCURS
9187 ;CHECK TO SEE IF IT WAS SUPPOSED TO,
9188 ;AND IF YES CHECK ABORT FLAG AND
9189 ;STATUS REGISTERS
9190 043260 012703 043706 MOV #1,R3 ;LET R3, R1, AND R2 POINT TO THE
9191 043264 012701 043756 MOV #BNO,R1 ;DOWNWARD EXPANSION TABLES
9192 043270 012702 044024 MOV #ABORT1,R2 ;
9193 043274 004767 000030 JSR PC,TSM9 ;TURN MMU ON
9194 ;DO RELOCATIONS FOR THE DIFFERENT
9195 ;VALUES OF THE PAGE LENGTH FIELD AND
9196 ;BLOCK NUMBER. IF AN ABORT OCCURS
9197 ;CHECK TO SEE IF IT WAS SUPPOSED TO,
9198 ;AND IF YES CHECK ABORT FLAG AND
9199 ;STATUS REGISTERS.

```

COKDABO KDJ11 B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 169
 COKDAB.P11 05-APR-84 16:45 MEMORY MANAGEMENT TESTS

SEQ 0168

```

9200 043300 005037 177572      CLR      @#177572      ;MMU OFF
9201 043304 012703 043716      MOV      @PLF1+10,R3    ;POINT TO A VALUE WHICH SHOULD CAUSE
9202 043310 012701 043766      MOV      @BN1+10,R1    ;AN ABORT IF MMU IS ON.
9203 043314 011337 177600      MOV      (R3),@#177600    ;SETUP UIPDRO
9204 043320 006521              MFPI      (R1)+      ;DO A RELOCATION
9205 043322 012605              MOV      (SP)+,R5      ;POP THE STACK
9206
9207 043324 000167 000542      JMP      TSM9FIN
9208
9209      ;ROUTINE TO CAUSE AND CHECK PAGE LENGTH ERROR ABORTS
9210      ;
9211 043330 012337 177600      TSM9:  MOV      (R3)+,@#177600    ;SETUP UIPDRO
9212 043334 010100              MOV      R1,R0      ;SAVE A COPY OF R1
9213 043336 012767 000001 137432  MOV      @1,FLAG    ;SETUP FOR AN ABORT
9214 043344 012737 000001 177572  MOV      @1,@#177572 ;TURN MMU ON
9215 043352 010704              MOV      R7,R4      ;SAVE PC
9216 043354 006530              MFPI      @R0+      ;DO A RELOCATION OPERATION
9217 043356 021227 000000      CMP      (R2),#0      ;WAS AN ABORT SUPPOSED TO OCCUR
9218 043362 001007              BNE      2$          ;IF YES GO TO 2$
9219 043364 012605              MOV      (SP)+,R5      ;POP THE STACK
9220 043366 022767 000001 137402  CMP      @1,FLAG    ;DID AN ABORT OCCUR
9221 043374 001401              BEQ      1$          ;NO GO ON
9222 043376 104002              ERROR    +2          ;MMU ERROR
9223                                     ;YES GO TO ERROR
9224 043400 000425              1$:  BR      6$          ;
9225 043402 022767 000000 137366  2$:  CMP      @0,FLAG    ;DID AN ABORT OCCUR
9226 043410 001401              BEQ      3$          ;YES GO ON
9227 043412 104002              ERROR    +2          ;MMU ERROR
9228                                     ;NO GO TO ERROR
9229 043414 105067 137364      3$:  CLRB      SAVMRO      ;SETUP EXPECTED DATA
9230 043420 022767 040000 137356  CMP      @40000,SAVMRO ;TEST MMRO FOR EXPECTED VALUE
9231 043426 001401              BEQ      4$          ;IF OK THEN CONTINUE
9232 043430 104002              ERROR    +2          ;MMU ERROR
9233                                     ;NOT OK THEN GO TO ERROR
9234 043432 022767 000020 137346  4$:  CMP      @20,SAVMR1  ;TEST MMR1 FOR EXPECTED VALUE
9235 043440 001401              BEQ      5$          ;IF OK THEN CONTINUE
9236 043442 104002              ERROR    +2          ;MMU ERROR
9237                                     ;NOT OK THEN GO TO ERROR
9238 043444 020467 137340      5$:  CMP      R4,SAVMR2  ;TEST MMR2 FOR EXPECTED VALUE
9239 043450 001401              BEQ      6$          ;IF OK THEN CONTINUE
9240 043452 104002              ERROR    +2          ;MMU ERROR
9241                                     ;NOT OK THEN GO TO ERROR
9242 043454 005067 137316      6$:  CLR      FLAG      ;CLEAR MMU ABORT FLAG
9243 043460 005067 137320      CLR      SAVMRO      ;CLEAR STATUS REGS SAVE AREAS
9244 043464 005067 137316      CLR      SAVMR1
9245 043470 005067 137314      CLR      SAVMR2
9246 043474 005201              INC      R1
9247 043476 005201              INC      R1
9248 043500 005202              INC      R2
9249 043502 005202              INC      R2
9250 043504 021327 000777      CMP      (R3),#777    ;HAVE ALL ENTRIES BEEN TRIED
9251 043510 001307              BNE      TSM9
9252 043512 000207              RTS      PC
9253                                     ;YES RETURN
9254
9255      ;UPWARD EXPANSION TABLES
  
```

N13

CORRADO RDJ11 B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 170
 CORRAB.P11 05-APR-84 16:45 MEMORY MANAGEMENT TESTS

SEQ 0169

9256	043514	070006	PLFO:	.WORD	70006
9257	043516	070006		.WORD	70006
9258	043520	070006		.WORD	70006
9259	043522	013406		.WORD	13406
9260	043524	020006		.WORD	20006
9261	043526	004006		.WORD	04006
9262	043530	040006		.WORD	40006
9263	043532	070006		.WORD	70006
9264	043534	024006		.WORD	24006
9265	043536	004006		.WORD	04006
9266	043540	014006		.WORD	14006
9267	043542	012006		.WORD	12006
9268	043544	002006		.WORD	02006
9269	043546	001406		.WORD	01406
9270	043550	004006		.WORD	04006
9271	043552	002006		.WORD	02006
9272	043554	000406		.WORD	00406
9273	043556	007406		.WORD	07406
9274	043560	001006		.WORD	01006
9275	043562	003406		.WORD	03406
9276	043564	000777		.WORD	777
9277	043566	013000	BNO:	.WORD	013000
9278	043570	016000		.WORD	016000
9279	043572	017000		.WORD	017000
9280	043574	002700		.WORD	002700
9281	043576	014000		.WORD	014000
9282	043600	002000		.WORD	002000
9283	043602	004000		.WORD	004000
9284	043604	007000		.WORD	007000
9285	043606	002000		.WORD	002000
9286	043610	000700		.WORD	000700
9287	043612	004000		.WORD	004000
9288	043614	001000		.WORD	001000
9289	043616	000300		.WORD	000300
9290	043620	000400		.WORD	000400
9291	043622	001400		.WORD	001400
9292	043624	000600		.WORD	000600
9293	043626	000200		.WORD	000200
9294	043630	001700		.WORD	001700
9295	043632	000300		.WORD	000300
9296	043634	000700		.WORD	000700
9297	043636	000000	ABORTO:	.WORD	0
9298	043640	000000		.WORD	0
9299	043642	000001		.WORD	1
9300	043644	000000		.WORD	0
9301	043646	000001		.WORD	1
9302	043650	000001		.WORD	1
9303	043652	000000		.WORD	0
9304	043654	000000		.WORD	0
9305	043656	000000		.WORD	0
9306	043660	000000		.WORD	0
9307	043662	000001		.WORD	1
9308	043664	000000		.WORD	0
9309	043666	000000		.WORD	0
9310	043670	000001		.WORD	1
9311	043672	000001		.WORD	1

9312	043674	000001	.WORD	1
9313	043676	000001	.WORD	1
9314	043700	000000	.WORD	0
9315	043702	000001	.WORD	1
9316	043704	000000	.WORD	0
9317				
9318				
9319				
9320	043706	000416	.WORD	00416
9321	043710	020016	.WORD	20016
9322	043712	022016	.WORD	24016
9323	043714	034016	.WORD	34016
9324	043716	074016	.WORD	74016
9325	043720	040016	.WORD	40016
9326	043722	020016	.WORD	20016
9327	043724	000016	.WORD	00016
9328	043726	030016	.WORD	30016
9329	043730	010016	.WORD	10016
9330	043732	014016	.WORD	14016
9331	043734	004016	.WORD	04016
9332	043736	002016	.WORD	02016
9333	043740	000016	.WORD	00016
9334	043742	000016	.WORD	00016
9335	043744	005416	.WORD	05416
9336	043746	001016	.WORD	01016
9337	043750	001416	.WORD	01416
9338	043752	000416	.WORD	00416
9339	043754	000777	.WORD	777
9340	043756	000100	.WORD	000100
9341	043760	010000	.WORD	010000
9342	043762	006000	.WORD	006000
9343	043764	016000	.WORD	016000
9344	043766	016000	.WORD	016000
9345	043770	004000	.WORD	004000
9346	043772	000000	.WORD	000000
9347	043774	000000	.WORD	000000
9348	043776	004000	.WORD	004000
9349	044000	004000	.WORD	004000
9350	044002	004000	.WORD	004000
9351	044004	000000	.WORD	000000
9352	044006	000500	.WORD	000500
9353	044010	000000	.WORD	000000
9354	044012	000400	.WORD	000400
9355	044014	001000	.WORD	001000
9356	044016	000100	.WORD	000100
9357	044020	000400	.WORD	000400
9358	044022	000200	.WORD	000200
9359	044024	000000	.WORD	0
9360	044026	000000	.WORD	0
9361	044030	000000	.WORD	0
9362	044032	000000	.WORD	0
9363	044034	000001	.WORD	1
9364	044036	000001	.WORD	1
9365	044040	000001	.WORD	1
9366	044042	000000	.WORD	0
9367	044044	000001	.WORD	1

DOWNWARD EXPANSION TABLES

PLT 1:

HN1:

ABORT 2:

9368	044046	000000		.WORD	0	
9369	044050	000000		.WORD	0	
9370	044052	000001		.WORD	1	
9371	044054	000001		.WORD	1	
9372	044056	000001		.WORD	1	
9373	044060	000000		.WORD	0	
9374	044062	000000		.WORD	0	
9375	044064	000001		.WORD	1	
9376	044066	000000		.WORD	0	
9377	044070	000000		.WORD	0	
9378						
9379	044072	000240				
9380	044074					
9381						
9382	044074	005037	177572	CLR	00177572	MMU OFF
9383	044100	005067	136672	CLR	FLAG	CLEAR MMU ABORT FLAG
9384	044104	005067	136674	CLR	SAVMRO	CLEAR STATUS REGS SAVE AREAS
9385	044110	005067	136672	CLR	SAVMR1	
9386	044114	005067	136670	CLR	SAVMR2	
9387	044120	004767	067010	JSR	PC,MMU	INIT MMU
9388	044124	005037	177776	CLR	00177776	INIT PSW: PREVIOUS MODE = KERNAL
9389	044130	012702	020200	MOV	020200,R1	
9390	044134	012737	077400 177502	MOV	077400,00177502	SETUP KIPDR1 TO ABORT
9391	044142	012767	000001 136676	MOV	01,FLAG	SETUP FLAG FOR AN ABORT
9392	044150	012737	000001 177572	MOV	01,00177572	TURN MMU ON
9393	044156	010701		MOV	R7,R1	SAVE PC
9394	044160	006522		MFP1	(R7)	DO A RELOCATION VIA KIPDR1
9395	044162	012704	100003	MOV	0100003,R4	SETUP EXPECTED DATA
9396	044166	004767	000202	JSR	PC,1510	CHECK IF AN ABORT OCCURRED AND
9397						IF YES CHECK BITS <6:1> OF MMRO.
9398	044172	012737	030000 177776	MOV	030000,00177776	INIT PSW: PREVIOUS MODE = USER
9399	044200	004767	066730	JSR	PC,MMU	INIT MMU
9400	044204	012737	077400 177636	MOV	077400,00177636	SETUP KIPDR2 TO ABORT
9401	044212	012702	160000	MOV	0160000,R2	
9402	044216	012767	000001 136652	MOV	01,FLAG	SETUP FLAG FOR AN ABORT
9403	044224	012737	000001 177572	MOV	01,00177572	TURN MMU ON
9404	044232	010701		MOV	R7,R1	SAVE PC
9405	044234	106522		MFPD	(R7)	DO A RELOCATION VIA KIPDR2
9406	044236	012704	100177	MOV	0100177,R4	SETUP EXPECTED DATA
9407	044242	004767	000176	JSR	PC,1510	CHECK IF AN ABORT OCCURRED AND
9408						IF YES CHECK BITS <6:1> OF MMRO.
9409	044246	012737	010000 177776	MOV	010000,00177776	INIT PSW: PREVIOUS MODE = SUPERVISOR
9410	044254	004767	066654	JSR	PC,MMU	INIT MMU
9411	044260	012737	077400 177212	MOV	077400,00177212	SETUP SIPDR5 TO ABORT
9412	044266	012702	120000	MOV	0120000,R2	ACCESS PAGE 05
9413	044272	012767	000001 136476	MOV	01,FLAG	SETUP FLAG FOR AN ABORT
9414	044300	012737	000001 177572	MOV	01,00177572	TURN MMU ON
9415	044306	010701		MOV	R7,R1	SAVE PC
9416	044310	006522		MFP1	(R7)	DO A RELOCATION VIA SIPDR5
9417	044312	012704	100052	MOV	0100052,R4	SETUP EXPECTED DATA; ABORT, PAGE 05
9418	044316	004767	000052	JSR	PC,1510	CHECK IF AN ABORT OCCURRED AND
9419						IF YES CHECK BITS <6:1> OF MMRO.
9420						
9421						
9422						
9423	044322	012737	020000 177776	MOV	020000,00177776	INIT PSW: SET ILLEGAL PREVIOUS MODE

0174

SEQ 0172

9434	044350	004767	066600		JSR	PC,MMU	INIT MMU
9435	044354	012702	040000		MOV	040000,R2	SET UP ACCESS TO PAGE
9436	044340	012767	000001	136450	MOV	01,FLAG	SETUP FLAG FOR AN ABORT
9437	044346	012757	000001	177572	MOV	01,00177572	TURN MMU ON
9438	044354	010701			MOV	R7,R1	SAVE PC
9439	044356	106522			MOV	(R2),	DO A RELOCATION
9450	044360	012704	100105		MOV	0100105,R4	SETUP EXPECTED DATA:ABORT, ILLEGAL
9451							PROCESSOR MODE, PAGE 02
9452	044364	004767	000004		JSR	PC,T510	CHECK IF AN ABORT OCCURRED AND
9453							IF YES CHECK BITS <6:1> OF MMRO.
9454							
9455	044370	000167	000062		JMP	T10F IN	
9456							
9457							
9458							
9459							
9460							
9461	044374	022767	000000	136374	T510:	CMP	00,FLAG
9462	044402	001401				BEQ	1:
9463	044404	104002				ERROR	2:
9464							MMU ERROR
9465	044406	020467	136372		1:	CMP	R4,SAVMRO
9466	044412	001401				BEQ	2:
9467	044414	104002				ERROR	2:
9468							MMU ERROR
9469	044416	022767	000022	136362	2:	CMP	022,SAVMR1
9470	044424	001401				BEQ	3:
9471	044426	104002				ERROR	2:
9472							MMU ERROR
9473	044430	020167	136354		3:	CMP	R1,SAVMR2
9474	044434	001401				BEQ	4:
9475	044436	104002				ERROR	2:
9476							MMU ERROR
9477	044440	005067	136340		4:	CLR	SAVMRO
9478	044444	005067	136336			CLR	SAVMR1
9479	044450	005067	136334			CLR	SAVMR2
9480	044454	000207				RTS	PC
9481							RETURN
9482	044456						
9483	044456						
9484							
9485	044456	005037	177572				
9486	044462	005067	136310				
9487	044466	012737	030000	177776			
9488	044474	012701	000026				
9489	044500	012703	177610				
9490	044504	012704	000021				
9491	044510	004767	000060				
9492	044514	012737	000000	177776			
9493	044522	012701	000023				
9494	044526	012703	177310				
9495	044532	012704	000024				
9496	044536	004767	000032				
9497	044542	012737	010000	177776			
9498	044550	012701	000025				
9499	044554	012703	177310				
9500	044560	012704	000022				

CORDBO EDJ11 B CLUSTER MACY11 50(1046) 05 APR 84 16:45 PAGE 174
 CORDAB,P11 05 APR 84 16:45 MEMORY MANAGEMENT TESTS

SEQ 0173

```

9480 044564 004767 000004      JSR      PC,1511      ; TEST ENABLE SUPERVISOR DATA SPACE BIT
9481
9482 044570 000167 000120      JMP      T11FIN
9483
9484      ; ROUTINE TO TEST ENABLE DATA SPACE BITS OF MMRS
9485
9486 044574 004767 066554      T111:  JSR      PC,MMU      ; INIT MMU
9487 044600 010137 172516      MOV      R1,@0172516      ; DISABLE DATA SPACE OF MODE UNDER TEST
9488 044604 012713 077400      MOV      @77400,(R3)      ; SETUP IPDR TO ABORT
9489 044610 012702 100000      MOV      @100000,R2      ;
9490 044614 012767 000001 136154  MOV      @1,FLAG      ; SETUP FLAG FOR AN ABORT
9491 044622 012737 000001 177572  MOV      @1,@0177572      ; MMU ON
9492 044630 106522      MFPD      (R2),      ; DO A RELOCATION
9493 044632 022767 000000 136136  CMP      @0,FLAG      ; DID AN ABORT OCCUR
9494 044640 001401      BEQ      1$      ; YES GO ON
9495 044642 104002      ERROR      2$      ; MMU ERROR
9496
9497 044644 010437 172516      1$:  MOV      R4,@0172516      ; NO GO TO ERROR
9498 044650 012702 100000      MOV      @100000,R2      ; ENABLE DATA SPACE OF MODE UNDER TEST
9499 044654 012767 000001 136114  MOV      @1,FLAG      ;
9500 044662 012737 000001 177572  MOV      @1,@0177572      ; SETUP FLAG FOR AN ABORT
9501 044670 106522      MFPD      (R2),      ; MMU ON
9502 044672 005726      TST      (SP),      ; DO A RELOCATION
9503 044674 022767 000001 136074  CMP      @1,FLAG      ; POP THE STACK
9504 044702 001401      BEQ      2$      ; DID AN ABORT OCCUR
9505 044704 104002      ERROR      2$      ; NO GO ON
9506
9507 044706 005067 136064      2$:  CLR      FLAG      ; MMU ERROR
9508 044712 000207      RTS      PC      ; YES GO TO ERROR
9509
9510 044714      ; T11FIN:      ; CLEAR MMU ABORT FLAG
9511 044714      ; TSM12:      ; RETURN
9512
9513 044714 005037 177572      ; MMRI FUNCTIONAL TEST
9514 044720 005067 136052      CLR      @0177572      ; MMU OFF
9515 044724 005067 136056      CLR      FLAG      ; CLEAR MMU ABORT FLAG
9516 044730 004767 066200      CLR      SAVMR1      ; CLEAR STATUS REG SAVE AREA
9517 044734 012737 030000 177776  JSR      PC,MMU      ; INIT MMU
9518 044742 012704 100200      MOV      @30000,@0177776  ; INIT PSW
9519 044746 010401      MOV      @100200,R4      ; SETUP TEST LOCATIONS
9520 044750 012705 100101      MOV      R4,R1      ;
9521 044754 010502      MOV      @100101,R5      ;
9522 044756 012737 000020 172516  MOV      @20,@0172516      ; INIT MMRS
9523 044764 012737 077402 172310  MOV      @77402,@0172310  ; SETUP KIPDR4 TO ABORT
9524 044772 012703 006414      MOV      @6414,R3      ; SETUP EXPECTED DATA FOR MMRI
9525 044776 012767 000001 135772  MOV      @1,FLAG      ; SETUP FLAG FOR AN ABORT
9526 045004 012737 000001 177572  MOV      @1,@0177572      ; TURN MMU ON
9527 045012 010767 135744      MOV      R2,5U0C00      ; SAVE PC
9528 045016 112425      MOVH     (R4), (R5),      ; DO A RELOCATION
9529 045020 004767 000206      JSR      PC,1512      ; CHECK IF AN ABORT OCCURRED AND IF
9530
9531 045024 012703 175011      MOV      @175011,R3      ; YES IF MMRI EQUALS EXPECTED DATA
9532 045030 012767 000001 135740  MOV      @1,FLAG      ; SETUP EXPECTED DATA FOR MMRI
9533 045036 012737 000001 177572  MOV      @1,@0177572      ; SETUP FLAG FOR AN ABORT
9534 045044 010767 135712      MOV      R2,5U0C00      ; TURN MMU ON
9535 045050 112142      MOVH     (R1), (R2),      ; SAVE PC
                                ; DO A RELOCATION

```

```

9536 045052 004767 000154 JSR PC,TS12 ;CHECK IF AN ABORT OCCURRED AND IF
9537 ;YES IF MMRI EQUALS EXPECTED DATA
9538 045056 012703 006771 MOV #6771,R3 ;SETUP EXPECTED DATA FOR MMRI
9539 045062 012767 000001 135706 MOV #1,FLAG ;SETUP FLAG FOR AN ABORT
9540 045070 012737 000001 177572 MOV #1,#0177572 ;TURN MMU ON
9541 045076 010767 135660 MOV R7,SLOC00 ;SAVE PC
9542 045102 114125 MOV# (R1),(R5)+ ;DO A RELOCATION
9543 045104 004767 000122 JSR PC,TS12 ;CHECK IF AN ABORT OCCURRED AND IF
9544 ;YES IF MMRI EQUALS EXPECTED DATA
9545 045110 012703 006411 MOV #6411,R3 ;SETUP EXPECTED DATA FOR MMRI
9546 045114 012767 000001 135654 MOV #1,FLAG ;SETUP FLAG FOR AN ABORT
9547 045122 012737 000001 177572 MOV #1,#0177572 ;TURN MMU ON
9548 045130 010767 135626 MOV R7,SLOC00 ;SAVE PC
9549 045134 112125 MOV# (R1),(R5)+ ;DO A RELOCATION
9550 045136 004767 000070 JSR PC,TS12 ;CHECK IF AN ABORT OCCURRED AND IF
9551 ;YES IF MMRI EQUALS EXPECTED DATA
9552 045142 012703 171025 MOV #171025,R3 ;SETUP EXPECTED DATA FOR MMRI
9553 045146 012767 000001 135622 MOV #1,FLAG ;SETUP FLAG FOR AN ABORT
9554 045154 012737 000001 177572 MOV #1,#0177572 ;TURN MMU ON
9555 045162 010767 135574 MOV R7,SLOC00 ;SAVE PC
9556 045166 012542 MOV (R5)+,(R2)+ ;DO A RELOCATION
9557 045170 004767 000036 JSR PC,TS12 ;CHECK IF AN ABORT OCCURRED AND IF
9558 ;YES IF MMRI EQUALS EXPECTED DATA
9559 045174 012703 012762 MOV #12762,R3 ;SETUP EXPECTED DATA FOR MMRI
9560 045200 012767 000001 135570 MOV #1,FLAG ;SETUP FLAG FOR AN ABORT
9561 045206 012737 000001 177572 MOV #1,#0177572 ;TURN MMU ON
9562 045214 010767 135542 MOV R7,SLOC00 ;SAVE PC
9563 045220 014225 MOV (R2),(R5)+ ;DO A RELOCATION
9564 045222 004767 000004 JSR PC,TS12 ;CHECK IF AN ABORT OCCURRED AND IF
9565 ;YES IF MMRI EQUALS EXPECTED DATA
9566
9567 045226 000167 000046 JMP T12FIN
9568
9569 ;ROUTINE TO CHECK IF AN ABORT OCCURRED AND IF MMRI EQUALS EXPECTED DATA
9570
9571 045232 022767 000000 135536 TS12: CMP #0,FLAG ;DID AN ABORT OCCUR
9572 045240 001401 BEQ 1$ ;YES GO ON
9573 045242 104002 ERROR 1$ ;MMU ERROR
9574 ;NO GO TO ERROR
9575 045244 020367 135536 1$: CMP R3,SAVMR1 ;TEST MMRI FOR EXPECTED DATA
9576 045250 001401 BEQ 2$ ;OK GO ON
9577 045252 104002 ERROR 2$ ;MMU ERROR
9578 ;NO GO TO ERROR
9579 045254 026767 135502 135526 2$: CMP SLOC00,SAVMR2 ;TEST MMRI2 FOR EXPECTED DATA
9580 045262 001401 BEQ 3$ ;OK GO ON
9581 045264 104002 ERROR 3$ ;MMU ERROR
9582 ;NO GO TO ERROR
9583 045266 005067 135514 3$: CLR SAVMR1 ;CLEAR STATUS REG SAVE AREA
9584 045272 005067 135512 CLR SAVMR2
9585 045276 000207 RTS PC ;RETURN
9586
9587 045300 000240 T12FIN: NOP
9588 045302 T12FIN:
9589 ;
9590 ;*****
9591 ;(NEED 16 BITS OF MEMORY ADDRESSING)

```

CORRADO RDU11 B CLUSTER MACY11 30(1046) 05 APR 84 16:45 PAGE 176
 CORDAB.P11 05-APR-84 16:45 MEMORY MANAGEMENT TESTS

SEQ 0175

```

9592 045302 005037 177572      CLR      @0177572      ;MMU OFF
9593 045306 005067 135464      CLR      FLAG        ;CLEAR MMU ABORT FLAG
9594 045312 005037 177776      CLR      @0177776      ;INIT PSW
9595 045316 004767 065612      JSR      PC,MMU      ;INIT MMU
9596 045322 012737 000020 172516 MOV      @20,@0172516  ;INIT MMR3
9597 045330 012703 045504      MOV      @PARAD1,R3    ;SETUP PARS WITH TEST VALUES
9598 045334 012701 045536      MOV      @PARVA1,R1    ;
9599 045340 012133      1$: MOV      (R1)+,@(R3)+    ;
9600 045342 021127 000333      CMP      (R1),@333    ;
9601 045346 001374      BNE      1$          ;
9602 045350 012703 045622      MOV      @PHY1,R3     ;SET POINTERS TO ADDER PART A
9603 045354 012701 045570      MOV      @VIR1,R1     ;TEST TABLES.
9604 045360 012702 045654      MOV      @MODE1,R2    ;
9605 045364 012237 177776      2$: MOV      (R2)+,@0177776  ;INIT PSW
9606 045370 013305      MOV      @R3)+,R5     ;SAVE DATA AT PHYSICAL ADDRESS
9607 045372 012737 000001 177572 MOV      @1,@0177572  ;TURN MMU ON
9608 045400 006531      MFPD      @R1)+      ;SAVE DATA AT RELOCATED VIRTUAL ADDRESS
9609 045402 012604      MOV      (SP)+,R4    ;
9610 045404 005037 177572      CLR      @0177572    ;TURN MMU OFF
9611 045410 020504      CMP      R5,R4      ;IS DATA EQUAL TO EXPECTED
9612 045412 001401      BEQ      3$          ;YES GO ON
9613 045414 104002      ERROR    *2      ;MMU ERROR
9614                                ;NO IT IS AN ADDER ERROR
9615 045416 021327 000111      3$: CMP      (R3),@111    ;ARE WE READY TO TEST DATA SPACE
9616 045422 001360      BNE      2$          ;NO GO TO 2$
9617 045424 005203      INC      R3          ;POINT TO DATA SPACE VALUES
9618 045426 005203      INC      R3          ;
9619 045430 005201      INC      R1          ;
9620 045432 005201      INC      R1          ;
9621 045434 005202      INC      R2          ;
9622 045436 005202      INC      R2          ;
9623 045440 012237 177776      MOV      (R2)+,@0177776  ;INIT PSW
9624 045444 013305      MOV      @R3)+,R5     ;SAVE DATA AT PHYSICAL ADDRESS
9625 045446 012737 000027 172516 MOV      @27,@0172516  ;INIT MMR3
9626 045454 012737 000001 177572 MOV      @1,@0177572  ;TURN MMU ON
9627 045462 106531      MFPD      @R1)+      ;SAVE DATA AT RELOCATED VIRTUAL ADDRESS
9628 045464 012604      MOV      (SP)+,R4    ;POP THE STACK
9629 045466 005037 177572      CLR      @0177572    ;TURN MMU OFF
9630 045472 020504      CMP      R5,R4      ;IS DATA EQUAL TO EXPECTED
9631 045474 001401      BEQ      4$          ;YES GO ON
9632 045476 104002      ERROR    *2      ;MMU ERROR
9633                                ;NO IT IS AN ADDER ERROR
9634 045500      4$:      JMP      T1$FIN
9635 045500 000167 000202      ;
9636                                ;ADDER TEST PART A TABLES
9637                                ;
9638                                ;
9639 045504 172240      PARAD1: .WORD    172240
9640 045506 177642      .WORD    177642
9641 045510 172252      .WORD    172252
9642 045512 177640      .WORD    177640
9643 045514 172242      .WORD    172242
9644 045516 172254      .WORD    172254
9645 045520 177652      .WORD    177652
9646 045522 177644      .WORD    177644
9647 045524 172246      .WORD    172246

```

CORRADO R0J11 B CLUSTER MAC111 30(1046) 05-APR-84 16:45 PAGE 177
 CORRADO.P11 05-APR-84 16:45 MEMORY MANAGEMENT TESTS

SEQ 0176

9648	045526	177654		.WORD	177654
9649	045530	172250		.WORD	172250
9650	045532	177660		.WORD	177660
9651	045534	000333		.WORD	333
9652	045536	000000	PARVA1:	.WORD	000000
9653	045540	000010		.WORD	000010
9654	045542	177777		.WORD	177777
9655	045544	177601		.WORD	177601
9656	045546	000010		.WORD	000010
9657	045550	000052		.WORD	000052
9658	045552	000070		.WORD	000070
9659	045554	000010		.WORD	000010
9660	045556	000010		.WORD	000010
9661	045560	000060		.WORD	000060
9662	045562	000000		.WORD	000000
9663	045564	000010		.WORD	000010
9664	045566	000333		.WORD	333
9665	045570	000000	VIR1:	.WORD	000000
9666	045572	025000		.WORD	025000
9667	045574	135224		.WORD	135224
9668	045576	017700		.WORD	017700
9669	045600	033000		.WORD	033000
9670	045602	145252		.WORD	145252
9671	045604	121000		.WORD	121000
9672	045606	043000		.WORD	043000
9673	045610	075000		.WORD	075000
9674	045612	142000		.WORD	142000
9675	045614	117700		.WORD	117700
9676	045616	000111		.WORD	111
9677	045618	007000		.WORD	007000
9678	045622	000000	PHY1:	.WORD	000000
9679	045624	006000		.WORD	006000
9680	045626	015124		.WORD	015124
9681	045630	000000		.WORD	000000
9682	045632	014000		.WORD	014000
9683	045634	012452		.WORD	012452
9684	045636	010000		.WORD	010000
9685	045640	004000		.WORD	004000
9686	045642	016000		.WORD	016000
9687	045644	010000		.WORD	010000
9688	045646	017700		.WORD	017700
9689	045650	000111		.WORD	111
9690	045652	010000		.WORD	010000
9691	045654	010000	MODE1:	.WORD	010000
9692	045658	030000		.WORD	030000
9693	045660	010000		.WORD	010000
9694	045662	030000		.WORD	030000
9695	045664	010000		.WORD	010000
9696	045666	010000		.WORD	010000
9697	045670	030000		.WORD	030000
9698	045672	030000		.WORD	030000
9699	045674	010000		.WORD	010000
9700	045676	030000		.WORD	030000
9701	045700	010000		.WORD	010000
9702	045702	000111		.WORD	111
9703	045704	030000		.WORD	030000

COKDABO KDJ11 R CLUSTER MAC111 30(1046) 05 APR-84 16:45 PAGE 178
 COKDAB.P11 05-APR-84 16:45 MEMORY MANAGEMENT TESTS

SEQ 0177

```

9704      ;
9705      045706      T13FIN:
9706      045706      T51822:
9707      ;          TEST 22/18 BIT ADDRESS OPTION
9708      ;*****
9709      ;CHECK THE SOFTWARE SWITCH REGISTER TO DETERMINE IF THIS IS A 22 BIT OR AN
9710      ;18 BIT ADDRESS SYSTEM. BIT 08 IN THE SWR+1 INDICATES AN 18 BIT SYSTEM.
9711      ;IF WE'RE IN A 22 BIT SYSTEM WE CAN PERFORM SOME EXTRA TESTS.
9712      045706      032777      000400      133224      BIT      0BIT08,SWR      ;IS BIT 08 SET?
9713      045714      001405      BEQ      100$      ;BRANCH IF ITS NOT
9714      045716      062737      000001      001204      ADD      01,00$TESTN      ;KEEP TEST NUMBERS IN ORDER
9715      ;ADD 1 FOR THE TESTS WE'RE SKIPPING
9716      045724      000167      001422      JMP      T14FIN      ;SKIP OVER THESE TESTS IF IT IS
9717      ;
9718      ;IF THIS IS A 22 BIT SYSTEM CHECK THE 22/18 BIT ADDRESS OPTION
9719      ;
9720      ;TO TEST 22 BIT ADDRESSING WE DO THE FOLLOWING:
9721      ;      A. ENABLE 22 BIT ADDRESSING MODE
9722      ;      B. CLEAR ADDRESS 0
9723      ;      C. WRITE PHYSICAL ADDRESS 17000000 WITH ALL ONES
9724      ;      D. CHECK ADDRESS 0
9725      ;IF ADDRESS 0 IS UNCHANGED (-0) OR A TIME OUT OCCURRED, IT INDICATES
9726      ;22 BIT MODE IS FUNCTIONING.
9727      ;IF ADDRESS 0 =177777 IT INDICATES THAT 22 BIT MODE IS NOT FUNCTIONING.
9728      ;
9729      045730      013767      000004      135024      100$:      MOV      004,SLOC00      ;SAVE VECTOR
9730      045736      013767      000006      135020      MOV      006,SLOC01      ;SAVE VECTOR
9731      045744      012737      046024      000004      MOV      013,004      ;SET VECTOR FOR NXM TRAP
9732      045752      012737      000340      000006      MOV      0340,006      ;
9733      045760      012767      000020      124530      MOV      020,SR3      ;ENABLE 22 BIT MODE ADDRESSING
9734      045766      012767      170000      124360      MOV      0170000,KIPAR6      ;SET KIPAR6 FOR 1920-1924KW ADDR RANGE
9735      045774      012767      000001      131570      MOV      01,SRO      ;ENABLE MMU
9736      046002      005067      131772      CLR      0      ;CLEAR ADDR 0
9737      046006      012737      177777      140000      MOV      0177777,00140000      ;MOVE ALL ONES TO ADDR 17000000 VIA KIPAR6
9738      ;A TIME OUT ERROR OR
9739      046014      005767      131760      TST      0      ;ADDR 0 REMAINING CLEAR INDICATES
9740      ;THAT 22 BIT ADDRESS MODE IS WORKING AND
9741      ;THAT SOME FURTHER TESTS SHOULD BE PERFORMED
9742      046020      001405      BEQ      2$      ;IF ADDR 0 =177777
9743      ;ERROR! 22 BIT ADDRESS MODE BAD
9744      046022      104002      ERROR      02      ;MMU ERROR
9745      046024      012706      001000      1$:      MOV      051801,SP      ;GOT HERE AS A RESULT OF NXM TRAP--
9746      ;CLEAN UP THE STACK
9747      046030      005037      177766      CLR      00177766      ;CLEAR CPU ERROR REGISTER
9748      046034      012737      046074      000004      2$:      MOV      033,004      ;SET UP VECTOR FOR NXM TRAP
9749      046042      042767      000020      124446      BIC      001104,SR3      ;SET 18 BIT ADDRESSING MODE IN SR3
9750      046050      012767      170000      124276      MOV      0170000,KIPAR6      ;SET KIPAR6 SO THAT BITS 18-21 SHOULD
9751      ;BE ASSERTED IF 22 BIT ADDR WAS ENABLED
9752      046056      012737      177777      140000      MOV      0177777,00140000      ;TRY TO WRITE ADDR 17000000 VIA KIPAR6
9753      ;ADDR 0 SHOULD = 177777. A TIME OUT
9754      046064      022737      177777      000000      CMP      0177777,000      ;OR ADDR 0 = ZERO INDICATES AN ERROR
9755      046072      001403      BEQ      4$      ;GO TO NEXT TEST IF ADDR 0=177777
9756      046074      005067      131472      3$:      CLR      SRO      ;DISABLE MMU BEFORE ERROR.
9757      ;ERROR! 18 BIT ADDR OPTION IS N.G.
9758      046100      104002      ERROR      02      ;MMU ERROR
9759      ;

```

```

9760 ;TEST ADDRESS BITS 18 THRU 21
9761 ;
9762
9763 046102 052767 000020 124406 4$: BIS #BIT04,SR3 ;ENABLE 22 BIT ADDRESSING MODE
9764 046110 012767 046152 131666 MOV #5$,4 ;SET UP FOR NXM TRAP
9765 046116 005067 131656 CLR 0 ;CLEAR ADDRESS 0
9766 046122 012767 010000 124124 MOV #10000,KIPAR6 ;TEST ADDRESS BIT 18
9767 046130 012737 177777 140000 MOV #177777,#0140000 ;WRITE ALL ONES TO ADDR 1000000
9768 046136 005767 131636 TST 0 ;TEST ADDRESS 0. SHOULD = ZERO
9769 046142 001403 BEQ 5$ ;BRANCH IF ADDRESS 0=0
9770 046144 005067 131422 CLR SR0 ;DISABLE MMU BEFORE ERROR
9771 046150 104002 ERROR +2 ;MMU ERROR
9772 ;ERROR! BIT 18 DID NOT ASSERT
9773 046152 012737 046214 000004 5$: MOV #6$,#04 ;SET UP FOR NXM TRAP
9774 046160 005067 131614 CLR 0 ;CLEAR ADDR 0
9775 046164 012767 020000 124162 MOV #20000,KIPAR6 ;TEST ADDRESS BIT 19
9776 046172 012737 177777 140000 MOV #177777,#0140000 ;WRITE ALL ONES TO ADDR 2000000
9777 046200 005767 131574 TST 0 ;TEST ADDR 0. SHOULD = 0
9778 046204 001403 BEQ 6$ ;BRANCH IF ADDRESS 0=0
9779 046206 005067 131360 CLR SR0 ;DISABLE MMU BEFORE ERROR
9780 046212 104002 ERROR +2 ;MMU ERROR
9781 ;ERROR! BIT 19 DID NOT ASSERT
9782 046214 012737 046256 000004 6$: MOV #7$,#04 ;SET UP FOR NXM TRAP
9783 046222 005067 131552 CLR 0 ;CLEAR ADDR 0
9784 046226 012767 040000 124120 MOV #40000,KIPAR6 ;TEST ADDRESS BIT 20
9785 046234 012737 177777 140000 MOV #177777,#0140000 ;WRITE ALL ONES TO ADDR 4000000
9786 046242 005767 131532 TST 0 ;TEST ADDR 0. SHOULD = 0
9787 046246 001403 BEQ 7$ ;BRANCH IF ADDRESS 0 = 0
9788 046250 005067 131316 CLR SR0 ;DISABLE MMU BEFORE ERROR
9789 046254 104002 ERROR +2 ;MMU ERROR
9790 ;ERROR! BIT 20 DID NOT ASSERT
9791 046256 012737 046320 000004 7$: MOV #8$,#04 ;SET UP FOR NXM
9792 046264 005067 131510 CLR 0 ;CLEAR ADDRESS 0
9793 046270 012767 100000 124056 MOV #100000,KIPAR6 ;TEST ADDRESS BIT 21
9794 046276 012737 177777 140000 MOV #177777,#0140000 ;WRITE ALL ONES AT ADDR 1000000
9795 046304 005767 131470 TST 0 ;CHECK ADDRESS 0. SHOULD = 0
9796 046310 001403 BEQ 8$ ;BRANCH IF ADDR 0 = 0
9797 046312 005067 131254 CLR SR0 ;DISABLE MMU BEFORE ERROR
9798 046316 104002 ERROR +2 ;MMU ERROR
9799 ;ERROR! ADDR BIT 21 DID NOT ASSERT
9800 046320 005067 131246 8$: CLR SR0 ;DISABLE MMU
9801 046324 005037 177766 CLR #0177766 ;CLEAR CPU ERROR REGISTER
9802 046330 012706 001000 MOV #51801,R6 ;RESET STACK POINTER
9803 046334 013737 002762 000004 MOV #05LOC00,#04 ;RESTORE VECTORS
9804 046342 013737 002764 000006 MOV #05LOC01,#06 ;
9805
9806 046350
9807
9808
9809
9810 046350 005037 177572
9811 046354 005067 131406
9812 046360 013737 000004 002762
9813 046366 013737 000006 002764
9814 046374 012737 047064 000004
9815 046402 012737 000340 000006
  
```

```

15MM14:
; ADDER RELOCATION TEST PART B
;*****
;(NEED 22 BITS OF MEMORY ADDRESSING)
  
```


COKDABO KDJ11 B CLUSTER MACY11 30(1046) 05-APR 84 16:45 PAGE 180
 COKDAB.P11 05-APR-84 16:45 MEMORY MANAGEMENT TESTS

SEQ 0179

```

9816 046410 005037 172340          CLR      @#KIPAR0          ;SET KER PAR0 FOR 1ST 4KW OF MEMORY.
9817 046414 012767 077406 123656    MOV      @77406,KIPDR0      ;SET KER PDR FOR 4KW R/W ACCESS.
9818 046422 012737 177500 172354    MOV      @177500,@#KIPAR6    ;SET UP KERNEL PAGE ADDR REG 6
9819                                     ;FOR HIGHEST 4K WORDS OF NON-I/O
9820                                     ;FOR 2 MEG WORDS OF MEMORY.
9821 046430 012767 077406 123656    MOV      @77406,KIPDR6    ;SET KER PDR6 FOR 4KW R/W ACCESS.
9822 046436 012737 000020 172516    MOV      @20,@#SR3        ;ENABLE 22 BIT ADDRESSING.
9823 046444 012737 000001 177572    MOV      @1,@#SR0        ;TURN ON THE MMU.
9824 046452 005737 157776          TST      @#157776          ;ATTEMPT TO ADDRESS LAST MEMORY ADDR.
9825                                     ;*****WILL TRAP TO 4 IF 2 MEG WORDS OF MEMORY NOT AVAILABLE*****
9826 046456 013737 002762 000004    MOV      @#SLOC00,@#4      ;RESTORE LOC 4
9827 046464 013737 002764 000006    MOV      @#SLOC01,@#6      ;RESTORE LOC 6
9828 046472 005037 177572          CLR      @#177572          ;MMU OFF
9829 046476 005037 002776          CLR      @#FLAG          ;CLEAR MMU ABORT FLAG
9830 046502 004767 064426          JSR      PC,MMU          ;INIT MMU
9831 046506 012737 010000 177776    MOV      @10000,@#177776  ;INIT PSW
9832 046511 012737 000020 172516    MOV      @20,@#172516    ;INIT MMR3
9833 046522 052737 001000 177746    BIS      @1000,@#177746  ;TURN CACHE TEST FEATURE ON
9834 046530 012704 047266          MOV      @PARVA3,R4        ;SET POINTERS TO INIT. TABLES
9835 046534 012701 047320          MOV      @VIR3,R1
9836 046540 012437 172246 1$:      MOV      (R4)+,@#172246    ;INIT SIPAR3
9837 046544 012737 000001 177572    MOV      @1,@#177572    ;TURN MMU ON
9838 046552 012746 125252          MOV      @125252,-(SP)    ;PUSH BACKGROUND DATA ON TO THE STACK
9839 046556 006671 000000          MTPI      @0(R1)        ;WRITE DATA TO PHYSICAL ADDRESS
9840 046562 006531 125252          MFPI      @0(R1)        ;WRITE DATA AT PHYSICAL ADDRESS TO STACK
9841 046564 022726 125252          CMP      @125252,(SP)+    ;IS DATA EQUAL TO EXPECTED
9842 046570 001403 177572          BEQ      @#177572    ;YES GO ON
9843 046572 005037 177572          CLR      @#177572    ;TURN MMU OFF
9844 046576 104002 177572          ERROR    +2          ;MMU ERROR
9845                                     ;NOT EQUAL GO TO ERROR
9846 046600 005037 177572 2$:      CLR      @#177572    ;TURN MMU OFF
9847 046604 021427 000333          CMP      (R4),@#333    ;ARE WE DONE
9848 046610 001353 177572          BNE     1$          ;NO GO TO 1$
9849 046612 012704 047150          MOV      @PARVA2,R4    ;SET POINTERS TO PAR INIT TABLES
9850 046616 012701 047116          MOV      @PARAD2,R1
9851 046622 012431 177572 3$:      MOV      (R4)+,@(R1)+    ;INIT PAR3
9852 046624 021127 000333          CMP      (R1),@#333    ;ARE WE DONE
9853 046630 001374 177572          BNE     3$          ;NO, GO TO 3$
9854 046632 012704 047234          MOV      @MODE2,R4    ;SET POINTERS TO ADDR PART B TABLES
9855 046636 012701 047202          MOV      @VIR2,R1
9856 046642 012702 047266          MOV      @PARVA3,R2
9857 046646 012703 047320          MOV      @VIR3,R3
9858 046652 004767 000076 4$:      JSR      PC,TS14    ;WRITE DATA TO PHYSICAL ADDRESS AND THEN
9859                                     ;CHECK IF DATA AT PHYSICAL ADDRESS IS
9860                                     ;EQUAL TO EXPECTED AND IF NOT DETERMINE
9861                                     ;IF IT IS AN ADDR ERROR OR A MEMORY ERROR
9862 046656 021127 000111          CMP      (R1),@#111    ;HAVE WE DONE ALL THE 22 BIT MODE I SPACE
9863                                     ;CASES
9864 046662 001373 177572          BNE     4$          ;NO GO TO 4$
9865 046664 005201 177572          INC      R1          ;POINT TO 22 BIT MODE D SPACE CASE
9866 046666 005201 177572          INC      R1
9867 046670 005204 177572          INC      R4
9868 046672 005204 177572          INC      R4
9869 046674 012737 000027 172516    MOV      @27,@#172516    ;INIT MMR3
9870 046702 012437 177776          MOV      (R4)+,@#177776  ;INIT PSW
9871 046706 012746 052525          MOV      @52525,(SP)    ;PUSH DATA ONTO STACK

```

CORDBO KDJ11 B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 181
 CORDAB.P11 05-APR-84 16:45 MEMORY MANAGEMENT TESTS

SEQ 0180

```

9872 046712 012737 000001 177572      MOV      @1,@0177572      ;TURN MMU ON
9873 046720 106631                MTPD      @0(R1)+      ;WRITE DATA TO PHYSICAL ADDRESS
9874 046722 005037 177572          CLR      @0177572      ;TURN MMU OFF
9875 046726 012737 000020 172516      MOV      @20,@0172516      ;INIT MMR3
9876 046734 004767 000040          JSR      PC,T14      ;CHECK IF DATA AT PHYSICAL ADDRESS IS EQUAL
9877                                ;TO EXPECTED AND IF NOT DETERMINE IF IT
9878                                ;IS AN ADDER ERROR OR A MEMORY ERROR
9879 046740 005037 172516          CLR      @0172516      ;INIT MMR3 FOR 18 BIT MODE
9880 046744 004767 000004          JSR      PC,T514      ;WRITE DATA TO PHYSICAL ADDRESS AND THEN
9881                                ;CHECK IF DATA AT PHYSICAL ADDRESS IS
9882                                ;EQUAL TO EXPECTED AND IF NOT DETERMINE IF
9883                                ;IT IS AN ADDER ERROR OR A MEMORY ERROR
9884
9885 046750 000167 000376          JMP      T14FIN
9886
9887                                ;
9888                                ;ROUTINE TO WRITE DATA TO PHYSICAL ADDRESS AND TO CHECK IF DATA AT
9889                                ;PHYSICAL ADDRESS IS EQUAL TO EXPECTED AND IF NOT DETERMINE IF IT IS
9890                                ;AN ADDER ERROR OR A MEMORY ERROR
9891                                ;
9891 046754 012437 177776      T514:  MOV      (R4)+,@0177776      ;INIT PSW
9892 046760 012737 000001 177572      MOV      @1,@0177572      ;TURN MMU ON
9893 046766 012746 052525      MOV      @52525,-(SP)      ;WRITE DATA ONTO STACK
9894 046772 006631                MTPD      @0(R1)+      ;WRITE DATA TO PHYSICAL ADDRESS VIA STACK
9895 046774 005037 177572          CLR      @0177572      ;TURN MMU OFF
9896 047000 012737 010000 177776      T14:  MOV      @10000,@0177776      ;INIT PSW
9897 047006 012237 172246      MOV      (R2)+,@0172246      ;INIT SIPAR3
9898 047012 012737 000001 177572      MOV      @1,@0177572      ;TURN MMU ON
9899 047020 006573 000000      MEPT      @0(R3)      ;DO RELOCATION
9900 047024 022726 052525      CMP      @52525,(SP)+      ;IS DATA EQUAL TO EXPECTED
9901 047030 001410                BEQ      2$      ;YES GO ON
9902 047032 006573 000000      MEPT      @0(R3)      ;WHAT TYPE OF ERROR IS IT
9903 047036 022726 125252      CMP      @125252,(SP)+      ;
9904 047042 001402                BEQ      1$      ;
9905 047044 104002                ERROR      *2      ;MMU ERROR
9906                                ;IT IS A MEMORY ERROR
9907 047046 000401                BR      2$      ;
9908 047050 104002      1$:  ERROR      *2      ;MMU ERROR
9909                                ;IT IS AN ADDER ERROR
9910 047052 005037 177572      2$:  CLR      @0177572      ;TURN MMU OFF
9911 047056 005203                INC      R3      ;
9912 047060 005203                INC      R3      ;
9913 047062 000207                RTS      PC      ;RETURN
9914
9915                                ;
9916                                ;NON-EXISTANT MEMORY TRAP ROUTINE
9917                                ;
9917 047064 005037 177572      NXMTTP: CLR      @54RO      ;TURN OFF MMU
9918 047070 012716 047552      MOV      @T14FIN,(SP)      ;SET UP STACK WITH RETURN ADDR
9919 047074 013737 002762 000004      MOV      @0510C00,@04      ;RESTORE LOC 4
9920 047102 013737 002764 000006      MOV      @0513C01,@06      ;RESTORE LOC 6
9921 047110 005037 177766      CLR      @0177766      ;CLEAR TIME OUT INDICATION FROM
9922                                ;CPU ERROR REGISTER
9923 047114 000006      RET      ;RETURN FROM TRAP; GO TO NEXT TEST
9924
9925                                ;
9926                                ;ADDER TEST PART B TABLES
9927 047116 177646      PARADP: .WORD      177646

```


CORRADO KDJ11 8 CLUSTER MAC11 30(1046) 05 APR 84 16:45 PAGE 182
 CORRAD.P11 05-APR-84 15:45 MEMORY MANAGEMENT TESTS

SEQ 0181

9928	047120	177650	.WORD	177650
9929	047122	177652	.WORD	177652
9930	047124	172240	.WORD	172240
9931	047126	177640	.WORD	177640
9932	047130	177642	.WORD	177642
9933	047132	172244	.WORD	172244
9934	047134	177644	.WORD	177644
9935	047136	172252	.WORD	172252
9936	047140	172352	.WORD	172352
9937	047142	177662	.WORD	177662
9938	047144	172242	.WORD	172242
9939	047146	000333	.WORD	333
9940	047150	157700	PARVA2: .WORD	157700
9941	047152	137700	.WORD	137700
9942	047154	077700	.WORD	077700
9943	047156	176777	.WORD	176777
9944	047160	007600	.WORD	007600
9945	047162	167700	.WORD	167700
9946	047164	175700	.WORD	175700
9947	047166	177425	.WORD	177425
9948	047170	177220	.WORD	177220
9949	047172	173700	.WORD	173700
9950	047174	176700	.WORD	176700
9951	047176	077400	.WORD	077400
9952	047200	000333	.WORD	333
9953	047202	070000	VIR2: .WORD	070000
9954	047204	110000	.WORD	110000
9955	047206	130000	.WORD	130000
9956	047210	000000	.WORD	000000
9957	047212	000000	.WORD	000000
9958	047214	030000	.WORD	030000
9959	047216	050000	.WORD	050000
9960	047220	052524	.WORD	052524
9961	047222	136000	.WORD	136000
9962	047224	130000	.WORD	130000
9963	047226	000111	.WORD	111
9964	047230	030000	.WORD	030000
9965	047232	030000	.WORD	030000
9966	047234	030000	MODE2: .WORD	030000
9967	047236	030000	.WORD	030000
9968	047240	030000	.WORD	030000
9969	047242	010000	.WORD	010000
9970	047244	030000	.WORD	030000
9971	047246	030000	.WORD	030000
9972	047250	010000	.WORD	010000
9973	047252	030000	.WORD	030000
9974	047254	010000	.WORD	010000
9975	047256	000000	.WORD	000000
9976	047260	000111	.WORD	111
9977	047262	030000	.WORD	030000
9978	047264	010000	.WORD	010000
9979	047266	160000	PARVA3: .WORD	160000
9980	047270	140000	.WORD	140000
9981	047272	100000	.WORD	100000
9982	047274	176770	.WORD	176770
9983	047276	007600	.WORD	007600

```

9984 047300 170000 .WORD 170000
9985 047302 176000 .WORD 176000
9986 047304 177552 .WORD 177552
9987 047306 177400 .WORD 177400
9988 047310 174000 .WORD 174000
9989 047312 177000 .WORD 177000
9990 047314 007500 .WORD 007500
9991 047316 000333 .WORD 333
9992 047320 060000 VIR3: .WORD 060000
9993 047322 060000 .WORD 060000
9994 047324 060000 .WORD 060000
9995 047326 060700 .WORD 060700
9996 047330 060000 .WORD 060000
9997 047332 060000 .WORD 060000
9998 047334 060000 .WORD 060000
9999 047336 060024 .WORD 060024
10000 047340 060000 .WORD 060000
10001 047342 060000 .WORD 060000
10002 047344 060000 .WORD 060000
10003 047346 060000 .WORD 060000
10004 047350 000333 .WORD 333
10005
10006
10007
10008 047352 ;
10009 ;T14FIN:
10010 ;
10011 ; TEST NON-EXISTANT MEMORY TRAP
10012 ;*****
10013 ;WE ARE ASSUMING THAT THE NON EXISTANT MEMORY TIME OUT
10014 ;FEATURE IS WORKING SINCE WE CAN'T GUARANTEE THAT
10015 ;THE SYSTEM BEING TESTED HAS A NON EXISTANT MEMORY LOCATION.
10016 ;AT THIS TIME WE WILL ATTEMPT TO TEST THE NXM FUNCTION
10017 047352 004767 063556 JSR PC,MMU ;INIT THE MMU
10018 047356 012737 177400 172354 MOV #177400,#KIPAR6 ;SET KIPAR6 TO RELOCATE TO HIGHEST MEMORY
10019 047364 016767 130414 133370 MOV 4,SLOC00 ;SAVE VECTOR
10020 047372 016767 000026 130404 MOV 2$,4 ;LOAD VEC WITH ADDR OF TRAP HANDLER
10021 047400 052767 000001 130164 BIS #BIT00,SRO ;TURN ON THE MMU
10022 047406 005067 130354 CLR CPEREG ;CLEAR THE CPU ERROR REGISTER
10023 047412 005067 130360 CLR PS ;CLEAR THE PSW
10024 047416 005737 157776 TST #157776 ;ACCESS PHYSICAL ADDR 17757776
10025 047422 000415 1$: BR NXMFIN ;IF IT DOESN'T TRAP WE'LL ASSUME
10026 ;THAT THIS IS A 4 MEGABYTE SYSTEM
10027 047424 022767 000040 130334 2$: CMP #BIT05,CPEREG ;IS CPU ERROR REGISTER CORRECT?
10028 047432 001401 BEQ 3$ ;
10029 047434 104002 ERROR +2 ;MMU ERROR
10030 047436 022726 047422 3$: CMP #1$(SP), ;IS CONTENTS OF STACK CORRECT?
10031 047442 001401 BEQ 4$ ;
10032 047444 104002 ERROR +2 ;MMU ERROR
10033 047446 022726 000000 4$: CMP #0,(SP), ;IS CONTENTS OF STACK CORRECT?
10034 047452 001401 BEQ NXMFIN ;
10035 047454 104002 ERROR +2 ;MMU ERROR
10036 047456 005067 130110 NXMFIN: CLR SRO ;TURN OFF THE MMU
10037 047462 005067 130300 CLR CPEREG ;CLEAR THE CPU ERROR REGISTER
10038 047466 016767 133270 130310 MOV SLOC00,4 ;RESTORE THE VECTOR
10039

```

10040									
10041	04 74 74								
10042									
10043	04 74 74	005037	177572						
10044	04 7500	005067	155272						
10045	04 7504	004767	063474						
10046	04 7510	005037	177576						
10047	04 7514	017704	177500						
10048	04 7520	004767	000114						
10049									
10050									
10051	04 7524	004767	000160						
10052									
10053	04 7530	017737	050000	177776					
10054	04 7536	017704	177700						
10055	04 7540	004767	000072						
10056									
10057									
10058	04 7546	004767	000136						
10059									
10060									
10061									
10062	04 7552	005037	177776						
10063	04 7558	017737	170000	177776					
10064	04 7564	017704	177600						
10065	04 7570	004767	000044						
10066									
10067									
10068	04 7574	004767	000110						
10069									
10070									
10071	04 7600	005037	177776						
10072	04 7604	017704	177500						
10073	04 7610	004767	000152						
10074									
10075									
10076	04 7614	017704	177700						
10077	04 7620	004767	000142						
10078									
10079									
10080	04 7624	017704	177600						
10081	04 7630	004767	000132						
10082									
10083									
10084									
10085	04 7634	000167	000154						
10086									
10087									
10088									
10089									
10090	04 7640	005001							
10091	04 7642	017737	000020	177572					
10092	04 7650	017737	000001	177572					
10093	04 7656	011111							
10094	04 7660	005037	177572						
10095	04 7664	022427	077506						

```

1:MM15:
PAGE WRITTEN HIT TEST
CLR 00177572
CLR FLAG
JSR PC,MMU
CLR 00177776
MOV 0177500,R4
JSR PC,7515

```

```

;MMU OFF
;CLEAR MMU ABORT FLAG
;INIT MMU
;INIT PSW
;SET POINTER TO KPDR
;DO RELOCATIONS AND TEST KPDR FOR
;PAGE WRITTEN HIT BEING SET AND IF
;NOT SET GO TO ERROR
;DO RELOCATIONS AND TEST KPDR FOR
;PAGE WRITTEN HIT BEING SET AND IF NOT
;SET GO TO ERROR
;INIT PSW
;SET POINTER TO SPDR
;DO RELOCATIONS AND TEST SPDR FOR
;PAGE WRITTEN HIT BEING SET AND IF NOT
;SET GO TO ERROR
;DO RELOCATIONS AND TEST SPDR FOR
;PAGE WRITTEN HIT BEING SET AND IF NOT
;SET GO TO ERROR
;INIT PSW TO A KNOWN STATE
;INIT PSW
;SET POINTER TO UPDR
;DO RELOCATIONS AND TEST UPDR FOR
;PAGE WRITTEN HIT BEING SET AND IF
;NOT SET GO TO ERROR
;DO RELOCATIONS AND TEST UPDR FOR
;PAGE WRITTEN HIT BEING SET AND IF NOT
;SET GO TO ERROR
;INIT PSW TO A KNOWN STATE
;SET POINTER TO KPDR
;EXPLICITLY WRITE TO KPDR AND TEST
;FOR PAGE WRITTEN HIT BEING CLEARED
;AND IF NOT CLEARED GO TO ERROR
;SET POINTER TO SPDR
;EXPLICITLY WRITE TO SPDR AND TEST
;FOR PAGE WRITTEN HIT BEING CLEARED
;AND IF NOT CLEARED GO TO ERROR
;SET POINTER TO UPDR
;EXPLICITLY WRITE TO UPDR AND TEST
;FOR PAGE WRITTEN HIT BEING CLEARED
;AND IF NOT CLEARED GO TO ERROR

```

```

;ROUTINE TO DO RELOCATIONS AND TEST KPDR FOR PAGE WRITTEN HIT BEING
;SET AND IF NOT SET REPORT AN ERROR

```

```

1:MM15:
CLR R1
MOV 070,00177516
MOV 01,00177517
MOV CR0,CR1
CLR 00177517
CMP (R4),077506

```

```

;SET POINTER TO VIRTUAL ADDRESS
;INIT MMU
;TURN MMU ON
;DO A RELOCATION
;TURN MMU OFF
;IS DATA EQUAL TO EXPECTED

```

```

10096 047670 001401      BEQ      2$      ;OK GO ON
10097 047672 104002      ERROR    1$      ;MMU ERROR
10098                                ;NO GO TO ERROR
10099 047674 062701 000000 1$: ADD    020000,R1      ;POINT TO NEXT VIRTUAL ADDRESS
10100 047700 020127 160000    CMP    R1,0160000      ;ARE WE DONE
10101 047704 001561      BNE     1$      ;NO GO TO 1$
10102 047706 000207      RTS      PC      ;RETURN
10103
10104 ;ROUTINE TO DO RELOCATIONS AND TEST DPDFS FOR PAGE WRITTEN BIT BEING SET
10105 ;AND IF NOT SET REPORT AN ERROR
10106
10107 047710 005001      T15:  CLR     R1      ;SET POINTER TO VIRTUAL ADDRESS
10108 047712 062704 000002      ADD    02,R4      ;POINT TO FIRST DPDF
10109 047716 012757 000027 172516    MOV    027,00172516      ;INIT MMU3
10110 047724 012757 000001 177572 1$: MOV    01,00177572      ;TURN MMU ON
10111 047732 011146      MOV    (R1), (SP)      ;PUSH DATA ONTO THE STACK
10112 047734 106611      MTPD    (R1)      ;DO A RELOCATION
10113 047736 005037 177572      CLR    00177572      ;TURN MMU OFF
10114 047742 027427 077506      CMP    (R4)+,077506      ;IS DATA EQUAL TO EXPECTED
10115 047746 001401      BEQ      2$      ;OK GO ON
10116 047750 104002      ERROR    1$      ;MMU ERROR
10117                                ;NO GO TO ERROR
10118 047752 062701 000000 2$: ADD    020000,R1      ;POINT TO NEXT VIRTUAL ADDRESS
10119 047756 020127 160000    CMP    R1,0160000      ;ARE WE DONE
10120 047762 001561      BNE     1$      ;NO GO TO 1$
10121 047764 000207      RTS      PC      ;RETURN
10122
10123 ;ROUTINE TO EXPLICITLY WRITE TO PDRS AND TEST PAGE WRITTEN BIT FOR BEING
10124 ;CLEARED AND IF NOT CLEARED REPORT AN ERROR
10125
10126 047766 005002      T15A:  CLR     R2      ;CLEAR COUNTER
10127 047770 011414 1$: MOV    (R4),(R4)      ;DO AN EXPLICIT WRITE TO PDR
10128 047772 022427 077406      CMP    (R4)+,077406      ;IS DATA EQUAL TO EXPECTED
10129 047776 001401      BEQ      2$      ;OK GO ON
10130 050000 104002      ERROR    1$      ;MMU ERROR
10131                                ;NO GO TO ERROR
10132 050002 005202 2$: INC     R2      ;INCREMENT POINTER
10133 050004 020227 000020      CMP    R2,020      ;ARE WE DONE
10134 050010 001561      BNE     1$      ;NO GO TO 1$
10135 050012 000207      RTS      PC      ;RETURN
10136 050014
10137 T15E IN:
10138 050014
10139 TMM16:
10140 050014 005037 177572 ;TEST CSM (CALL SUPERVISOR MODE)
10141 050020 005037 002776    CLR    00177572      ;MMU OFF
10142 050024 012704 050544    CLR    001776      ;CLEAR MMU ABORT FLAG
10143 050030 004767 065100    MOV    0TMM16,R4      ;INIT R4
10144 050034 012757 000057 172516    LSR    PC,MMU      ;INIT MMU
10145 050042 005037 177776    MOV    027,00172516      ;ENABLE CSM INSTRUCTION
10146 050046 015746 000010      CLR    00177776      ;SET PS TO KER MODE
10147 050052 015746 000014      MOV    0010, (SP)      ;SAVE VECTORS
10148 050056 015746 000016      MOV    0014, (SP)
10149 050062 012757 050254 000010      MOV    0016, (SP)
10150 050070 012757 000157 000014      MOV    0TMM16,0010
10151 050076 012737 050354 000016      MOV    0157,0014
10152                                MOV    0TMM16A,0016

```

10152	050104	007014			.WORD	7014		: TEST INSTRUCTION
10153	050106	104002			ERROR	.		;MMU ERROR
10154								;GO TO ERROR IF NOT TRAPPED
10155	050110	012737	050264	000010	TSM16A:	MOV	@TMM16C,@010	;SETUP NEW VECTOR
10156	050116	012737	000027	172516		MOV	@27,@0172516	;DISABLE CSM INSTRUCTION
10157	050124	012737	140000	177776		MOV	@140000,@0177776	;SET PS TO USER MODE
10158	050132	007014			.WORD	7014		: TEST INSTRUCTION
10159	050134	104002			ERROR	.		;MMU ERROR
10160								;GO TO ERROR IF NOT TRAPPED
10161	050136	012737	050314	000010	TSM16B:	MOV	@TMM16D,@010	;SETUP NEW VECTOR
10162	050144	012737	000027	172516		MOV	@27,@0172516	;DISABLE CSM INSTRUCTION
10163	050152	005037	177776			CLR	@0177776	;SET PS TO KER MODE
10164	050156	007014			.WORD	7014		: TEST INSTRUCTION
10165	050160	104002			ERROR	.		;MMU ERROR
10166								;GO TO ERROR IF NOT TRAPPED
10167	050162	012737	000037	172516	TSM16C:	MOV	@37,@0172516	;ENABLE CSM INSTRUCTION
10168	050170	012737	040000	177776		MOV	@40000,@0177776	;SET PS TO SUP MODE
10169	050176	012706	000700			MOV	@700,R6	;INIT SUP SP
10170	050202	012737	140000	177776		MOV	@140000,@0177776	;SET PS TO USER MODE
10171	050210	012706	000600			MOV	@600,R6	;INIT USER SP
10172	050214	012737	000014	000010		MOV	@14,@010	;SETUP NEW VECTOR
10173	050222	000277				SCC		;SET ALL CC BITS
10174	050224	007024			.WORD	7024		: TEST INSTRUCTION
10175	050226	104002			TSM16D:	ERROR	.	;MMU ERROR
10176								;GO TO ERROR IF NOT TRAPPED
10177	050230	000167	000504		JMP	TM16A		
10178					:			
10179					:			
10180	050234	042737	007777	177776	TMM16B:	BIC	@777,@0177776	;CLEAR UNWANTED BITS
10181	050242	022737	000000	177776		CMP	@0,@0177776	;IS PS CORRECT
10182	050250	001401				REQ	13	;YES GO ON
10183	050252	104002			ERROR	.		;MMU ERROR
10184								;NO GO TO ERROR
10185	050254	005726			13:	ISI	(SP).	;CLEAN UP STACK
10186	050256	005726				ISI	(SP).	:
10187	050260	000167	177624			JMP	TM16A	;CONTINUE TESTING
10188	050264	042737	007777	177776	TMM16C:	BIC	@777,@0177776	;CLEAR UNWANTED BITS
10189	050272	022737	030000	177776		CMP	@30000,@0177776	;IS PS CORRECT
10190	050300	001401				REQ	13	;YES GO ON
10191	050302	104002			ERROR	.		;MMU ERROR
10192								;NO GO TO ERROR
10193	050304	005726			13:	ISI	(SP).	;CLEAN UP STACK
10194	050306	005726				ISI	(SP).	:
10195	050310	000167	177624			JMP	TM16B	;CONTINUE TESTING
10196	050314	042737	007777	177776	TMM16D:	BIC	@777,@0177776	;CLEAR UNWANTED BITS
10197	050322	022737	000000	177776		CMP	@0,@0177776	;IS PS CORRECT
10198	050330	001401				REQ	13	;YES GO ON
10199	050332	104002			ERROR	.		;MMU ERROR
10200								;NO GO TO ERROR
10201	050334	005726			13:	ISI	(SP).	;CLEAN UP STACK
10202	050336	005726				ISI	(SP).	:
10203	050340	000167	177624			JMP	TM16C	;CONTINUE TESTING
10204	050344	156430			TMM16E:	.WORD	156430	: TEST LOCATION
10205	050346	104002			TMM16F:	ERROR	.	;MMU ERROR
10206								

10208	050354	022737	070017	177776	TMM16A: CMP	070017,00177776	;IS P5 CORRECT
10209	050362	001401			REQ	15	;YES GO ON
10210	050364	104002			ERROR	02	;MMU ERROR
10211							;NO GO TO ERROR
10212	050366	020627	000572		15: CMP	R6,0572	;IS SP CORRECT
10213	050372	001401			REQ	25	;YES GO ON
10214	050374	104002			ERROR	02	;MMU ERROR
10215							;NO GO TO ERROR
10216	050376	020427	050346		25: CMP	R4,0TMM16E+2	;IS R4 CORRECT
10217	050402	001401			REQ	35	;YES GO ON
10218	050404	104002			ERROR	02	;MMU ERROR
10219							;NO GO TO ERROR
10220	050406	023727	050344	156430	35: CMP	00TMM16E,0156430	;IS TEST LOCATION OK
10221	050414	001401			REQ	45	;YES GO ON
10222	050416	104002			ERROR	02	;MMU ERROR
10223							;NO GO TO ERROR
10224	050420	022627	156430		45: CMP	(SP)+,0156430	;IS STACK CORRECT
10225	050424	001401			REQ	55	;YES GO ON
10226	050426	104002			ERROR	02	;MMU ERROR
10227							;NO GO TO ERROR
10228	050430	022627	050226		55: CMP	(SP)+,0TSM16D	;IS STACK CORRECT
10229	050434	001401			REQ	65	;YES GO ON
10230	050436	104002			ERROR	02	;MMU ERROR
10231							;NO GO TO ERROR
10232	050440	022627	140000		65: CMP	(SP)+,0140000	;IS STACK CORRECT
10233	050444	001401			REQ	75	;YES GO ON
10234	050446	104002			ERROR	02	;MMU ERROR
10235							;NO GO TO ERROR
10236	050450	012706	000700		75: MOV	0700,R6	;RESTORE SUP SP
10237	050454	012737	140000	177776	MOV	0140000,00177776	;SET P5 TO USER MODE
10238	050462	020627	000600		CMP	R6,0600	;IS USER SP CORRECT
10239	050466	001401			REQ	85	;YES GO ON
10240	050470	104002			ERROR	02	;MMU ERROR
10241							;NO GO TO ERROR
10242	050472	012767	077400	121500	85: MOV	077400,SIPDRO	;SETUP SIPDRO TO ABORT
10243	050500	012737	050346	000016	MOV	0TMM16E,0016	;SETUP VECTOR
10244	050506	012737	000001	002776	MOV	01,00FLAG	;SETUP FLAG FOR AN ABORT
10245	050514	012737	000001	177572	MOV	01,00177572	;TURN MMU ON
10246	050522	010701			MOV	R7,R1	;SAVE OLD PC
10247	050524	007014			WORD	7014	;TEST INSTRUCTION
10248	050526	022737	000000	002776	CMP	00,00FLAG	;DID AN ABORT OCCUR
10249	050534	001401			REQ	95	;YES GO ON
10250	050536	104002			ERROR	02	;MMU ERROR
10251							;NO GO TO ERROR
10252	050540	023701	003010		95: CMP	00AVMRO,R1	;IS MMRO CORRECT
10253	050544	001401			REQ	105	;YES GO ON
10254	050546	104002			ERROR	02	;MMU ERROR
10255							;NO GO TO ERROR
10256	050550	023727	003004	100041	105: CMP	00AVMRO,0100041	;IS MMRO CORRECT
10257	050556	001401			REQ	115	;YES GO ON
10258	050560	104002			ERROR	02	;MMU ERROR
10259							;NO GO TO ERROR
10260	050562	012737	000057	177516	115: MOV	057,00177516	;ENABLE CSM
10261	050570	012737	040000	177776	MOV	040000,00177776	;SET PSW TO SUP
10262	050576	012706	000700		MOV	0700,R6	;SETUP SUP SP
10263	050602	012737	140000	177776	MOV	0140000,00177776	;SET PSW TO USE


```

10264 050610 012706 000600      MOV      #600,R6          ;SETUP USE SP
10265 050614 012737 000014 000010  MOV      #14,#010        ;SETUP NEW VECTOR
10266 050622 012737 050644 000016  MOV      #TS16,#016      ;SETUP NEW VECTOR
10267 050630 000777              SCC              ;SET ALL CC BITS
10268 050632 007027              .WORD    7027              ;TEST INSTRUCTION
10269 050634 045712              .WORD    45712
10270 050636 104002      TS16A:  ERROR      *2          ;MMU ERROR
10271                          ;GO TO ERROR IF DIDN'T TRAP
10272 050640 000167 000074      JMP      TM16A
10273 050644 022737 070017 177776  TS16:  CMP      #70017,#0177776      ;IS PSW CORRECT
10274 050652 001401              BEQ      200$              ;YES GO ON
10275 050654 104002              ERROR      *2          ;MMU ERROR
10276                          ;NO GO TO ERROR
10277 050656 020627 000572      200$:  CMP      R6,#572              ;IS SP CORRECT
10278 050662 001401              BEQ      201$              ;YES GO ON
10279 050664 104002              ERROR      *2          ;MMU ERROR
10280                          ;NO GO TO ERROR
10281 050666 022627 045712      201$:  CMP      (SP)+,#045712      ;IS STACK CORRECT
10282 050672 001401              BEQ      202$              ;YES GO ON
10283 050674 104002              ERROR      *2          ;MMU ERROR
10284                          ;NO GO TO ERROR
10285 050676 022627 050636      202$:  CMP      (SP)+,#TS16A      ;IS STACK CORRECT
10286 050702 001401              BEQ      203$              ;YES GO ON
10287 050704 104002              ERROR      *2          ;MMU ERROR
10288                          ;NO GO TO ERROR
10289 050706 022627 140000      203$:  CMP      (SP)+,#140000      ;IS STACK CORRECT
10290 050712 001401              BEQ      204$              ;YES GO ON
10291 050714 104002              ERROR      *2          ;MMU ERROR
10292                          ;NO GO TO ERROR
10293 050716 012706 000700      204$:  MOV      #700,R6          ;RESTORE SUP SP
10294 050722 012737 140000 177776  MOV      #140000,#0177776 ;SET PSW TO USER MODE
10295 050730 020627 000600      CMP      R6,#600          ;IS USER SP CORRECT
10296 050734 001401              BEQ      TM16A          ;YES GO ON
10297 050736 104002              ERROR      *2          ;MMU ERROR
10298                          ;NO GO TO ERROR
10299 050740 005037 177776      TM16A:  CLR      #0177776      ;SET PSW TO KER MODE
10300 050744 012637 000016      MOV      (SP)+,#016      ;RESTORE VECTORS
10301 050750 012637 000014      MOV      (SP)+,#014      ;
10302 050754 012637 000010      MOV      (SP)+,#010      ;
10303
10304 ;*****
10305 ;FRTU FLOATING POINT TESTS
10306 ;*****
10307 ;
10308 ;      BEGIN FLOATING POINT TESTING
10309 ;*****
10310 ;*****
10311 MBI1:
10312 ;
10313 ;      FPP REGISTER BIT TESTS
10314 ;*****
10315 ;R5=FPP POINTER
10316 ;R1=TEMPORARY COUNTER
10317 ;R2=POINTER TO EXPECTED DATA
10318 ;R3=POINTER TO RECEIVED DATA
10319 ;R4=ODD/EVEN COUNTER
10319 050760 170011      SETD

```

CORDBA0 RDJ11 B CLUSTER MACY11 30(1046) 05 APR 84 16:45 PAGE 189
 CORDAB.P11 05-APR 84 16:45 FLOATING POINT TESTS

SEQ 0188

10320	050762	005005		MBT2:	CLR	R5		;SETUP FPP ACC POINTER
10321	050764	012702	003040		MOV	0BTEXP,R2		;POINT TO TEST DATA
10322	050770	012703	003050		MOV	0BTRES,R3		;POINT TO RECEIVED DATA
10323	050774	170400		MBT2A:	CLRD	AC0		;SETUP FPP REGISTER VALUES
10324	050776	174012			STD	AC0,(R2)		;CLEAR EXPECTED VALUE
10325	051000	005004			CLR	R4		
10326	051002	170400		BTG0:	CLRD	AC0		;SETUP FPP REGISTER VALUES
10327	051004	170401			CLRD	AC1		
10328	051006	170402			CLRD	AC2		
10329	051010	170403			CLRD	AC3		
10330	051012	170404			CLRD	AC4		
10331	051014	170405			CLRD	AC5		
10332								
10333	051016	010501			MOV	R5,R1		;GET FPP AC NUMBER INTO R1
10334	051020	070127	000014		MUL	014,R1		;ALLOW 10 LOCATIONS FOR OPERATION
10335	051024	062701	051032		ADD	0MAC0,R1		;SETUP JMP LOCATION
10336	051030	000111			JMP	(R1)		
10337	051032	172467	132002	MAC0:	LDD	BTEXP,AC0		;LOAD TEST DATA INTO TEST REGISTER
10338	051036	174067	132006	MAC0A:	STD	AC0,BTRES		;SAVE TEST RESULT
10339	051042	000167	000074		JMP	MACE		;GET OUT
10340	051046	172567	131766	MAC1:	LDD	BTEXP,AC1		;LOAD TEST DATA INTO TEST REGISTER
10341	051052	174167	131772		STD	AC1,BTRES		;SAVE TEST RESULT
10342	051056	000167	000060		JMP	MACE		;GET OUT
10343	051062	172667	131752	MAC2:	LDD	BTEXP,AC2		;LOAD TEST DATA INTO TEST REGISTER
10344	051066	174267	131756		STD	AC2,BTRES		;SAVE TEST RESULT
10345	051072	000167	000044		JMP	MACE		;GET OUT
10346	051076	172767	131736	MAC3:	LDD	BTEXP,AC3		;LOAD TEST DATA INTO TEST REGISTER
10347	051102	174367	131742		STD	AC3,BTRES		;SAVE TEST RESULT
10348	051106	000167	000030		JMP	MACE		;GET OUT
10349	051112	172467	131722	MAC4:	LDD	BTEXP,AC0		
10350	051116	174004			STD	AC0,AC4		;LOAD TEST DATA INTO TEST REGISTER
10351	051120	172404			LDD	AC4,AC0		;SAVE TEST RESULT
10352	051122	000167	177710		JMP	MAC0A		;GET OUT
10353	051126	172467	131706	MAC5:	LDD	BTEXP,AC0		;LOAD TEST DATA INTO TEST REGISTER
10354	051132	174005			STD	AC0,AC5		;LOAD TEST REGISTER
10355	051134	172405			LDD	AC5,AC0		;STORE RESULT INTO XFER FPP REGISTER
10356	051136	000167	177674		JMP	MAC0A		;GET OUT
10357	051142	026767	131672	131700	MACE:	CMP	BTEXP,BTRES	
10358	051150	001014			BNE	BTER		;BRANCH IF REGISTER ERROR
10359	051152	026767	131664	131672	CMP	BTEXP+2,BTRES+2		
10360	051160	001010			BNE	BTER		
10361	051162	026767	131656	131664	CMP	BTEXP+4,BTRES+4		
10362	051170	001004			BNE	BTER		
10363	051172	026767	131650	131656	CMP	BTEXP+6,BTRES+6		
10364	051200	001401			BEQ	MBT8		;GOOD RESULT
10365	051202	104003		BTER:	ERROR	...		;FPP ERROR
10366								;FPP AC LOADED INCORRECTLY
10367								
10368	051204							
10369	051204	005001		MBT8:	CLR	R1		;CLEAR TEMPORARY COUNTER
10370	051206	005705			TEST	R5		;SEE IF R0 UNDER TEST
10371	051210	001411			BEQ	MBT8A		;BRANCH IF TESTING R0
10372	051212	020527	000004		CMP	R5,#4		;SEE IF TESTING FPP REGISTER #RA
10373	051216	100006			BPL	MBT8A		;SKIP R0 TESTING
10374	051220	174067	131654		STD	AC0,BTRES		;SAVE AC TEST RESULT
10375	051224	004767	000216		JSR	R7,BTST		;VERIFY THAT CONTENTS REMAINED ZERO

1115

SEQ 0189

10376	051230	001401		BEQ	MBT8A				
10377	051237	104003		ERROR	+3				
10378									
10379	051234	020527	000001	MBT8A:	CMP	R5,#1			
10380	051240	001406			BEQ	MBT8B			
10381	051242	174167	131602		STD	AC1,BTRES			
10382	051246	004767	000174		JSR	R7,BTIST			
10383	051252	001401			BEQ	MBT8B			
10384	051254	104003			ERROR	+3			
10385									
10386	051256	020527	000002	MBT8B:	CMP	R5,#2			
10387	051262	001406			BEQ	MBT8C			
10388	051264	174267	131560		STD	AC2,BTRES			
10389	051270	004767	000152		JSR	R7,BTIST			
10390	051274	001401			BEQ	MBT8C			
10391	051276	104003			ERROR	+3			
10392									
10393	051300	020527	000003	MBT8C:	CMP	R5,#3			
10394	051304	001406			BEQ	MBT8D			
10395	051306	174367	131536		STD	AC3,BTRES			
10396	051312	004767	000130		JSR	R7,BTIST			
10397	051316	001401			BEQ	MBT8D			
10398	051320	104003			ERROR	+3			
10399									
10400	051322	020527	000004	MBT8D:	CMP	R5,#4			
10401	051326	001407			BEQ	MBT8E			
10402	051330	172404			LDD	AC4,ACO			
10403	051332	174067	131512		STD	AC0,BTRES			
10404	051336	004767	000104		JSR	R7,BTIST			
10405	051342	001401			BEQ	MBT8E			
10406	051344	104003			ERROR	+3			
10407									
10408	051346	020527	000005	MBT8E:	CMP	R5,#5			
10409	051352	001407			BEQ	MBT8F			
10410	051354	172405			LDD	AC5,ACO			
10411	051356	174067	131466		STD	AC0,BTRES			
10412	051362	004767	000060		JSR	R7,BTIST			
10413	051366	001401			BEQ	MBT8F			
10414	051370	104003			ERROR	+3			
10415									
10416	051372	005204		MBT8F:	INC	R4			
10417	051374	000241			CLC				
10418	051376	042704	177776		BIC	#17776,R4			
10419	051407	001401			BEQ	MBT8FG			
10420	051404	000261			SEC				
10421	051406	006112		MBT8FG:	ROL	(R2)			
10422	051410	006162	000002		ROL	(R2)			
10423	051414	006162	000004		ROL	(R2)			
10424	051420	006162	000006		ROL	(R2)			
10425	051424	103402			BCH	MBT8I			
10426	051426	000167	177350		JMP	BTGO			
10427									
10428	051432	005205		MBT8I:	INC	R5			
10429	051434	020527	000006		CMP	R5,#6			
10430	051440	100016			BPL	MBT8I			
10431	051442	000167	177326		JMP	MBT8A			

```

;BRANCH IF EXPECTED RESULT
;FPP ERROR
;BAD AC0
;SEE IF R1 UNDER TEST
;BRANCH IF R1 UNDER TEST
;SAVE AC TEST RESULT
;VERIFY THAT CONTENTS REMAINED ZERO
;BRANCH IF GOOD
;FPP ERROR
;BAD AC1
;SEE IF TESTING FPP REGISTER AC2
;BRANCH IF R2 UNDER TEST
;SAVE AC TEST RESULT
;VERIFY THAT CONTENTS REMAINED ZERO
;BRANCH IF GOOD
;FPP ERROR
;BAD AC2
;SEE IF R3 UNDER TEST
;BRANCH IF R3 UNDER TEST
;SAVE AC TEST RESULT
;VERIFY THAT CONTENTS REMAINED ZERO
;BRANCH IF GOOD
;FPP ERROR
;BAD AC3
;SEE IF R4 UNDER TEST
;BRANCH IF R4 UNDER TEST
;MOVE REGISTER CONTENT
;SAVE AC TEST RESULT
;VERIFY THAT CONTENTS REMAINED ZERO
;BRANCH IF GOOD
;FPP ERROR
;BAD AC4
;SEE IF R0 UNDER TEST
;BRANCH IF R0 UNDER TEST
;MOVE REGISTER CONTENTS
;SAVE AC TEST RESULT
;VERIFY THAT CONTENTS REMAINED ZERO
;BRANCH IF GOOD
;FPP ERROR
;BAD AC5
;INCREMENT PATTERN COUNTER
;TEST FOR ODD/EVEN
;BRANCH IF EVEN
;SET CARRY FOR TEST PATTERN SHIFT
;ROTATE 1 WORD OF TEST PATTERN
;ROTATE 2 WORD OF TEST PATTERN
;ROTATE 3 WORD OF TEST PATTERN
;ROTATE 4 WORD OF TEST PATTERN
;JUMP IF THROUGH WITH TEST PATTERN
;CONTINUE WITH NEW TEST PATTERN
;GO TO NEXT REGISTER TEST
;SEE IF THROUGH TESTING
;JUMP IF THROUGH
;CONTINUE TESTING WITH NEW PATTERN

```

```

10432
10433
10434 051446 005767 131376 BITST: TST BTRES ;VERIFY CONTENTS AS ZERO
10435 051452 001010 BNE BTSTIE ;EXIT IF NOT ZERO
10436 051454 005767 131372 TST BTRES+2 ;VERIFY CONTENTS AS ZERO
10437 051460 001005 BNE BTSTIE ;EXIT IF NOT ZERO
10438 051462 005767 131366 TST BTRES+4 ;VERIFY CONTENTS AS ZERO
10439 051466 001002 BNE BTSTIE ;EXIT IF NOT ZERO
10440 051470 005767 131362 TST BTRES+6 ;VERIFY CONTENTS AS ZERO
10441 051474 000207 BITSTIE: RTS R7 ;GO BACK TO CALLING ROUTINE
10442
10443
10444
10445
10446 051476 MBTE:
10447
10448
10449 051476 MFACU:
10450
10451 ; TEST UNIQUENESS OF FPP ACCUMULATORS
10452 ;*****
10453 ;THIS TEST LOADS UNIQUE PATTERNS INTO EACH ACCUMULATOR SIMULTANEOUSLY.
10454 ;R2=POINTER TO EXPECTED DATA
10455 ;R3=POINTER TO RECEIVED DATA
10456
10457
10458
10459 051476 170011 MFA: SETD
10460 051500 005000 CLR R0 ;SETUP FPP ACC POINTER
10461 051502 005004 CLR R4
10462 051504 012702 003040 MOV @BTEXP,R2 ;POINT TO TEST DATA
10463 051510 012703 003050 MOV @BTRES,R3 ;POINT TO RECEIVED DATA
10464 051514 012767 000051 131316 MOV @51,BTEXP ;SETUP EXPECTED DATA
10465 051522 012767 000052 131312 MOV @52,BTEXP+2
10466 051530 012767 000053 131308 MOV @53,BTEXP+4
10467 051536 012767 000054 131302 MOV @54,BTEXP+6
10468 051544 172467 131270 LDD BTEXP,ACO ;MOVE DATA TEMPORARILY
10469 051550 174005 STD ACO,AC5 ;PUT DATA INTO TEST REGISTER
10470 051552 004567 000210 JSR R5,SUBT ;SUBTRACT TEN FROM EACH EXPECTED DATA
10471 051556 172467 131256 LDD BTEXP,ACO ;MOVE DATA TEMPORARILY
10472 051562 174004 STD ACO,AC4 ;MOVE DATA INTO TEST REGISTER
10473 051564 004567 000176 JSR R5,SUBT ;SUBTRACT 10 FROM TEST DATA WORD
10474 051570 172767 131244 LDD BTEXP,AC3 ;STORE INTO TEST REGISTER
10475 051574 004567 000166 JSR R5,SUBT ;GET NEXT SET OF UNIQUE DATA WORDS
10476 051600 172667 131234 LDD BTEXP,AC2 ;STORE INTO TEST REGISTER
10477 051604 004567 000156 JSR R5,SUBT ;GET NEXT SET OF TEST DATA
10478 051610 172567 131224 LDD BTEXP,AC1 ;LOAD TEST REGISTER
10479 051614 004567 000146 JSR R5,SUBT ;GET NEXT SET OF TEST WORDS
10480 051620 172467 131214 LDD BTEXP,ACO ;LOAD FINAL TEST REGISTER
10481
10482 051624 174067 131200 STD ACO,BTRES ;ALL REGISTER CONTAIN UNIQUE TEST WORDS
10483 051630 004567 000216 JSR R5,BFA ;STORE ACO,RESULT
10484 051634 001401 BREQ BFAC1 ;CHECK RESULT
10485 051636 104003 ERROR ;BRANCH IF GOOD
10486
10487 051640 004567 000154 BFAC1: JSR R5,ADDT ;FPP ERROR
;HAD ACO ;UPDATE EXPECTED RESULT

```

Address	Offset	Label	Op	Op2	Op3	Op4	Op5	Op6	Op7	Op8	Op9	Op10	Op11	Op12	Op13	Op14	Op15	Op16	Op17	Op18	Op19	Op20	Op21	Op22	Op23	Op24	Op25	Op26	Op27	Op28	Op29	Op30	Op31	Op32	Op33	Op34	Op35	Op36	Op37	Op38	Op39	Op40	Op41	Op42	Op43	Op44	Op45	Op46	Op47	Op48	Op49	Op50	Op51	Op52	Op53	Op54	Op55	Op56	Op57	Op58	Op59	Op60	Op61	Op62	Op63	Op64	Op65	Op66	Op67	Op68	Op69	Op70	Op71	Op72	Op73	Op74	Op75	Op76	Op77	Op78	Op79	Op80	Op81	Op82	Op83	Op84	Op85	Op86	Op87	Op88	Op89	Op90	Op91	Op92	Op93	Op94	Op95	Op96	Op97	Op98	Op99	Op100	Op101	Op102	Op103	Op104	Op105	Op106	Op107	Op108	Op109	Op110	Op111	Op112	Op113	Op114	Op115	Op116	Op117	Op118	Op119	Op120	Op121	Op122	Op123	Op124	Op125	Op126	Op127	Op128	Op129	Op130	Op131	Op132	Op133	Op134	Op135	Op136	Op137	Op138	Op139	Op140	Op141	Op142	Op143	Op144	Op145	Op146	Op147	Op148	Op149	Op150	Op151	Op152	Op153	Op154	Op155	Op156	Op157	Op158	Op159	Op160	Op161	Op162	Op163	Op164	Op165	Op166	Op167	Op168	Op169	Op170	Op171	Op172	Op173	Op174	Op175	Op176	Op177	Op178	Op179	Op180	Op181	Op182	Op183	Op184	Op185	Op186	Op187	Op188	Op189	Op190	Op191	Op192	Op193	Op194	Op195	Op196	Op197	Op198	Op199	Op200	Op201	Op202	Op203	Op204	Op205	Op206	Op207	Op208	Op209	Op210	Op211	Op212	Op213	Op214	Op215	Op216	Op217	Op218	Op219	Op220	Op221	Op222	Op223	Op224	Op225	Op226	Op227	Op228	Op229	Op230	Op231	Op232	Op233	Op234	Op235	Op236	Op237	Op238	Op239	Op240	Op241	Op242	Op243	Op244	Op245	Op246	Op247	Op248	Op249	Op250	Op251	Op252	Op253	Op254	Op255	Op256	Op257	Op258	Op259	Op260	Op261	Op262	Op263	Op264	Op265	Op266	Op267	Op268	Op269	Op270	Op271	Op272	Op273	Op274	Op275	Op276	Op277	Op278	Op279	Op280	Op281	Op282	Op283	Op284	Op285	Op286	Op287	Op288	Op289	Op290	Op291	Op292	Op293	Op294	Op295	Op296	Op297	Op298	Op299	Op300	Op301	Op302	Op303	Op304	Op305	Op306	Op307	Op308	Op309	Op310	Op311	Op312	Op313	Op314	Op315	Op316	Op317	Op318	Op319	Op320	Op321	Op322	Op323	Op324	Op325	Op326	Op327	Op328	Op329	Op330	Op331	Op332	Op333	Op334	Op335	Op336	Op337	Op338	Op339	Op340	Op341	Op342	Op343	Op344	Op345	Op346	Op347	Op348	Op349	Op350	Op351	Op352	Op353	Op354	Op355	Op356	Op357	Op358	Op359	Op360	Op361	Op362	Op363	Op364	Op365	Op366	Op367	Op368	Op369	Op370	Op371	Op372	Op373	Op374	Op375	Op376	Op377	Op378	Op379	Op380	Op381	Op382	Op383	Op384	Op385	Op386	Op387	Op388	Op389	Op390	Op391	Op392	Op393	Op394	Op395	Op396	Op397	Op398	Op399	Op400	Op401	Op402	Op403	Op404	Op405	Op406	Op407	Op408	Op409	Op410	Op411	Op412	Op413	Op414	Op415	Op416	Op417	Op418	Op419	Op420	Op421	Op422	Op423	Op424	Op425	Op426	Op427	Op428	Op429	Op430	Op431	Op432	Op433	Op434	Op435	Op436	Op437	Op438	Op439	Op440	Op441	Op442	Op443	Op444	Op445	Op446	Op447	Op448	Op449	Op450	Op451	Op452	Op453	Op454	Op455	Op456	Op457	Op458	Op459	Op460	Op461	Op462	Op463	Op464

K15

CORRADO RDJ11 B CLUSTER MACY11 30(1046) 05 APR 84 16:45 PAGE 193
 CORRADO.P11 05-APR-84 16:45 FLOATING POINT TESTS

SEQ 0192

10544						
10545						
10546	052112			TSFP1:		
10547				:	TEST LDPS AND STEPS MODE 0	
10548	052112	005037	133436	CLR	@TRPFLG	;CLEAR TRAP FLAG
10549	052116	012704	147757	MOV	#147757,R4	;SETUP DATA TO BE LOADED
10550	052122	004767	000032	JSR	PC,LOST	;LOAD AND STORE FPS WITH DATA
10551	052126	012704	105252	MOV	#105252,R4	;SETUP DATA TO BE LOADED
10552	052132	004767	000032	JSR	PC,LOST	;LOAD AND STORE FPS WITH DATA
10553	052136	012704	042505	MOV	#42505,R4	;SETUP DATA TO BE LOADED
10554	052142	004767	000012	JSR	PC,LOST	;LOAD AND STORE FPS WITH DATA
10555	052146	005004		CLR	R4	;SETUP DATA TO BE LOADED
10556	052150	004767	000004	JSR	PC,LOST	;LOAD AND STORE FPS WITH DATA
10557						
10558				:		
10559	052154	000167	000014	JMP	FIN1	
10560				:		
10561	052160	170104		LOST:	LDPS R4	;LOAD FPS WITH DATA
10562	052162	170201			STPS R1	;LOAD R1 WITH (FPS)
10563	052164	020401			CMP R4,R1	;DID THE INSTRUCTIONS WORK
10564	052166	001401			BEQ 1\$	;YES GO ON
10565	052170	104003			ERROR +3	;FPP ERROR
10566						;NO GO TO ERROR
10567	052172	000207		1\$:	RTS PC	;RETURN
10568	052174	000240		FIN1:	NOP	
10569				:		
10570	052176			TSFP2:		
10571				:	TEST CFCC	
10572	052176	005037	133436	CLR	@TRPFLG	;CLEAR TRAP FLAG
10573	052202	012704	000017	MOV	#17,R4	;SETUP DATA TO BE LOADED
10574	052206	004767	000032	JSR	PC,TSF2	;LOAD FPS AND COPY CONDITION CODES TO PS
10575	052212	012704	000012	MOV	#12,R4	;SETUP DATA TO BE LOADED
10576	052216	004767	000022	JSR	PC,TSF2	;LOAD FPS AND COPY CONDITION CODES TO PS
10577	052222	012704	000005	MOV	#5,R4	;SETUP DATA TO BE LOADED
10578	052226	004767	000012	JSR	PC,TSF2	;LOAD FPS AND COPY CONDITION CODES TO PS
10579	052232	005004		CLR	R4	;SETUP DATA TO BE LOADED
10580	052234	004767	000004	JSR	PC,TSF2	;LOAD FPS AND COPY CONDITION CODES TO PS
10581						
10582				:		
10583	052240	000167	000024	JMP	FIN2	
10584				:		
10585	052244	170104		TSF2:	LDPS R4	;LOAD FPS
10586	052246	170000			CFCC	;COPY CONDITION CODES TO PS
10587	052250	013701	177776	MOV	@177776,R1	;SAVE PS TO R1
10588	052254	042701	177760	RIC	#177760,R1	;MASK OUT UNWANTED BITS
10589	052260	020401		CMP	R4,R1	;WAS CONDITION CODE BITS TRANSFERRED
10590						;CORRECTLY
10591	052262	001401			BEQ 1\$	;YES GO ON
10592	052264	104003			ERROR +3	;FPP ERROR
10593						;NO GO TO ERROR
10594	052266	000207		1\$:	RTS PC	;RETURN
10595	052270			FIN2:		
10596				:		
10597	052270			TSFP3:		
10598				:	TEST SEFL, SEFD, SEFI, SEFI	
10599	052270	005037	133436	CLR	@TRPFLG	;CLEAR TRAP FLAG

10600	052274	012704	000200		MOV	#200,R4		;SETUP DATA TO BE LOADED
10601	052300	170104			LDFPS	R4		;LOAD FPS
10602	052302	170001			SETF			;MAKE FD=0
10603	052304	170201			STFPS	R1		;STORE FPS
10604	052306	020127	000000		CMP	R1,#0		;IS FD=0
10605	052312	001401			BEQ	1\$		;YES GO ON
10606	052314	104003			ERROR	+3		;FPP ERROR
10607								;NO GO TO ERROR
10608	052316	170011		1\$:	SETD			;MAKE FD=1
10609	052320	170201			STFPS	R1		;STORE FPS
10610	052322	020104			CMP	R1,R4		;IS FD=1
10611	052324	001401			BEQ	2\$		;YES GO ON
10612	052326	104003			ERROR	+3		;FPP ERROR
10613								;NO GO TO ERROR
10614	052330	012704	000100	2\$:	MOV	#100,R4		;SETUP DATA TO BE LOADED
10615	052334	170104			LDFPS	R4		;LOAD FPS
10616	052336	170002			SETI			;MAKE FL=0
10617	052340	170201			STFPS	R1		;STORE FPS
10618	052342	020127	000000		CMP	R1,#0		;IS FL=0
10619	052346	001401			BEQ	3\$		;YES GO ON
10620	052350	104003			ERROR	+3		;FPP ERROR
10621								;NO GO TO ERROR
10622	052352	170012		3\$:	SETL			;MAKE FL=1
10623	052354	170201			STFPS	R1		;STORE FPS
10624	052356	020104			CMP	R1,R4		;IS FL=1
10625	052360	001401			BEQ	4\$		;YES GO ON
10626	052362	104003			ERROR	+3		;FPP ERROR
10627								;NO GO TO ERROR
10628	052364			4\$:				
10629								
10630	052364							
10631								
10632	052364	005037	133436					
10633	052370	012705	170003		CLR	#TRPFLG		;CLEAR TRAP FLAG
10634	052374	013746	000244		MOV	#170003,R5		;INIT OP CODE
10635	052400	012737	052530	000244	MOV	#0244,(SP)		;SAVE FP VECTOR
10636	052406	013746	000004		MOV	#ILLTOP1,#0244		;SETUP NEW VECTOR
10637	052412	012737	052600	000004	MOV	#04,(SP)		;SAVE TIME OUT VECTOR
10638	052420	013746	000010		MOV	#TIMEOUT,#04		;SETUP NEW VECTOR
10639	052424	012737	052604	000010	MOV	#010,(SP)		;SAVE ILLEGAL VECTOR
10640	052432	005003			MOV	#ILLTOP2,#010		;SETUP NEW VECTOR
10641	052434	170103		D1:	CLR	R3		
10642	052436	005002			LDFPS	R3		;CLEAR FPS
10643	052440	010537	052444		CLR	R2		
10644	052444	000000			MOV	R5,#0D2		;SETUP THE ILLEGAL INST
10645	052446	170000		D2:	WORD	0		
10646	052450	005202		D3:	CHCC			;MEMORY WORDS TO BE USED WITH
10647	052452	005202			INC	R2		;EXECUTION OF ILLEGAL OP CODE
10648	052454	170201			INC	R2		
10649	052456	104003			STFPS	R1		;SAVE FPS
10650					ERROR	+3		;FPP ERROR
10651	052460	022705	170010					;GO TO ERROR
10652	052464	001003		D4:	CMP	#170010,R5		;COMPUTE NEXT OP CODE
10653	052466	012705	170013		BNE	D5		
10654	052472	000757			MOV	#170013,R5		
10655	052474	022705	170077		BR	D1		
				D5:	CMP	#170077,R5		

M15

COKDABO RDJ11 B CLUSTER MACY11 30(1046) 05 APR 84 16:45 PAGE 195
 COKDAB.P11 05-APR-84 16:45 FLOATING POINT TESTS

SEQ 0194

10656	052500	001001			BNE	D6		
10657	052502	000402			BR	D7		
10658	052504	005205		D6:	INC	R5		
10659	052506	000751			BR	D1		
10660	052510	012637	000010	D7:	MOV	(SP)+, @#10		;RESTORE VECTORS
10661	052514	012637	000004		MOV	(SP)+, @#4		
10662	052520	012637	000244		MOV	(SP)+, @#244		
10663								
10664								
10665	052524	000167	000060		JMP	FIN4		
10666								
10667	052530	022716	052446	ILLOP1:	CMP	@D3, (SP)		;DID TRAP OCCUR ON TEST INST
10668	052534	001401			BEQ	1\$		;YES GO ON
10669	052536	104003			ERROR	+3		;FPP ERROR
10670								;NO GO TO ERROR
10671	052540	022626		1\$:	CMP	(SP)+, (SP)+		;CLEAN UP STACK
10672	052542	170201			STEPS	R1		;STORE FPS
10673	052544	022701	100000		CMP	@100000, R1		;IS FPS CORRECT
10674	052550	001401			BEQ	2\$		;YES GO ON
10675	052552	104003			ERROR	+3		;FPP ERROR
10676								;NO GO TO ERROR
10677	052554	005004		2\$:	CLR	R4		;INT R4 TO A KNOWN STATE
10678	052556	170304			STST	R4		;STORE FEC AT R4
10679								;IF THE DESTINATION MODE IS IMPROPERLY
10680								;DECODED AN ODD ADDRESS TRAP TO 4
10681								;SHOULD OCCUR
10682	052560	022704	000002		CMP	@2, R4		;IS FEC CORRECT
10683	052564	001002			BNE	3\$		;NO GO TO ERROR
10684	052566	000167	177666		JMP	D4		;YES GO ON
10685	052572	104003		3\$:	ERROR	+3		;FPP ERROR
10686								;GO TO ERROR
10687	052574	000167	177660		JMP	D4		;THEN GO ON
10688								
10689	052600	104003		TIMEOUT:	ERROR	+3		;FPP ERROR
10690								;ERROR BECAUSE OF TRAP TO 4
10691	052602	000006			RTT			;RETURN
10692								
10693	052604	104003		ILLOP2:	ERROR	+3		;FPP ERROR
10694								;ERROR BECAUSE OF TRAP TO 10
10695	052606	000006			RTT			;RETURN
10696	052610			FIN4:				
10697								
10698	052610			TEST FPS:				
10699								
10700	052610	005037	133436		TEST FID (INTERRUPT DISABLE BIT)			
10701	052614	013746	000244		CLR	@TRPFLG		;CLEAR TRAP FLAG
10702	052620	012737	052674		MOV	@#244, (SP)		;SAVE FP VECTOR
10703	052626	012703	040000		MOV	@11, @#244		;SETUP NEW VECTOR
10704	052632	170103			MOV	@40000, R3		;SETUP DATA TO BE LOADED
10705	052634	170020			LD FPS	R3		;LOAD FPS, FID+1
10706	052636	170000			WORD	170020		;ILLEGAL FP INSTRUCTION
10707	052640	170201			CECC			
10708	052642	022701	140000		STEPS	R1		;SEE IF ERROR WAS RECORDED IN FPS
10709	052646	001401			CMP	@140000, R1		
10710	052650	104003			BEQ	1\$		;YES GO ON
10711					ERROR	+3		;FPP ERROR
								;NO GO TO ERROR

N15

CORDBO RDJ11 B CLUSTER MACY11 30(1046) 05 APR 84 16:45 PAGE 196
 CORDAB,P11 05-APR-84 16:45 FLOATING POINT TESTS

SEQ 0195

10712	052652	170304		1\$:	STST	R4		;SEE IF FEC=2
10713	052654	022704	000002		CMP	#2,R4		
10714	052660	001401			BEQ	2\$		;YES GO ON
10715	052662	104003			ERROR	+3		;FPP ERROR
10716								;NO GO TO ERROR
10717	052664	012637	000244	2\$:	MOV	(SP)+, @#244		;RESTORE VECTOR
10718								
10719								
10720	052670	000167	000004		JMP	FIN5		
10721								
10722	052674	104003		ILL:	ERROR	+3		;FPP ERROR
10723								;FID ERROR
10724	052676	000006			RTT			;RETURN
10725	052700			FIN5:				
10726								
10727	052700			TSFP6:				
10728								
10729	052700	005037	133436		TEST LDD, STD F SRC AND FDST MODE 1			
10730	052704	005004			CLR	@#TRPFLG		;CLEAR TRAP FLAG
10731	052706	170104			CLR	R4		;SETUP TO LOAD DATA
10732	052710	170011			LDFPS	R4		;CLEAR FPS
10733	052712	013746	000004		SETD			;SET FD TO 1
10734	052716	012737	053050		MOV	@#4, -(SP)		;SAVE TIMEOUT VECTOR
10735	052724	012704	053040	000004	MOV	@TSF6, @#4		;SETUP NEW VECTOR
10736	052730	172414			MOV	@TS6DAT, R4		;SETUP POINTER TO DATA
10737	052732	020427	053040		LDD	(R4), ACO		;TEST INSTRUCTION
10738	052736	001401			CMP	R4, @TS6DAT		;IS R4 CORRECT
10739	052740	104003			BEQ	1\$		;YES GO ON
10740					ERROR	+3		;FPP ERROR
10741	052742	012701	053030	1\$:	MOV	@TS6DA, R1		;NO GO TO ERROR
10742	052746	012703	000004		MOV	@4, R3		;SETUP POINTER TO DATA
10743	052752	022421						;INIT COUNTER
10744	052754	001401		2\$:	CMP	(R4)+, (R1)+		;WAS SOURCE DATA ALTERED
10745	052756	104003			BEQ	3\$		;NO GO ON
10746					ERROR	+3		;FPP ERROR
10747	052760	077304						;YES GO TO ERROR
10748	052762	012704	003122	3\$:	SUB	R3, 2\$		;ARE WE DONE
10749	052766	174014			MOV	@TSTLOC, R4		;SETUP POINTER FOR DATA
10750	052770	020427	003122		STD	ACO, (R4)		;TEST INSTRUCTION
10751	052774	001401			CMP	R4, @TSTLOC		;IS R4 CORRECT
10752	052776	104003			BEQ	4\$		;YES GO ON
10753					ERROR	+3		;FPP ERROR
10754	053000	012701	053030	4\$:	MOV	@TS6DA, R1		;NO GO TO ERROR
10755	053004	012703	000004		MOV	@4, R3		;SETUP POINTER TO DATA
10756	053010	022421						;INIT COUNTER
10757	053012	001401		5\$:	CMP	(R4)+, (R1)+		;IS DESTINATION DATA CORRECT
10758	053014	104003			BEQ	6\$		;YES GO ON
10759					ERROR	+3		;FPP ERROR
10760	053016	077304						;NO GO TO ERROR
10761	053020	012637	000004	6\$:	SUB	R3, 5\$		;ARE WE DONE
10762					MOV	(SP)+, @#4		;RESTORE VECTOR
10763								
10764	053024	000167	000024		JMP	FIN6		
10765								
10766	053030	177777		TS6DA:	.WORD	177777		
10767	053032	000000			.WORD	000000		

[illegible]

10824	053240	104003		TS7:	ERROR	3			
10825									
10826	053242	000006			RTI			DEP ERROR	
10827								ODD ADDRESS TRAP	
10828	053244	000000						RETURN	
10829	053246	000000		TS7DA1:	.WORD	0			
10830	053250	000000			.WORD	0			
10831	053252	000000			.WORD	0			
10832	053254	037641		TS7DA2:	.WORD	37641			
10833	053256	065121			.WORD	65121			
10834	053260	037373			.WORD	37373			
10835	053262	022265			.WORD	22265			
10836	053264	000000		TS7DA4:	.WORD	0			
10837	053266	000000			.WORD	0			
10838	053270	037373			.WORD	37373			
10839	053272	022265			.WORD	22265			
10840	053274			FIN7:					
10841									
10842	053274			TSFP10:					
10843					TEST STD. STE. FOST MODE 0				
10844	053274	005037	133436		CLR	20TRP16			
10845	053300	012704	000200		MOV	200,R4			
10846	053304	170104			LDAPS	R4			
10847	053306	013746	000004		MOV	204,(R4)			
10848	053312	012737	053460	000004	MOV	2TSF10,204			
10849	053320	012704	053464		MOV	2TS10D1,R4			
10850	053324	172414			LD	(R4),AC0			
10851	053326	012701	053474		MOV	2TS10D2,R1			
10852	053332	172511			LD	(R1),AC1			
10853	053334	174100			STD	AC1,AC0			
10854	053336	012704	003122		MOV	2TS10C,R4			
10855	053342	174114			STD	AC1,(R4)			
10856	053344	004767	000022		JSR	PC,CHEC10			
10857	053350	012704	003122		MOV	2TS10C,R4			
10858	053354	012701	053474		MOV	2TS10D2,R1			
10859	053360	174014			STD	AC0,(R4)			
10860	053362	004767	000054		JSR	PC,CHEC10			
10861	053366	012701	053464		MOV	2TS10D1,R1			
10862	053372	172511			LD	(R1),AC1			
10863	053374	170001			SET				
10864	053376	174100			SET	AC1,AC0			
10865	053400	170011			SET				
10866	053402	012704	003122		MOV	2TS10C,R4			
10867	053406	174114			STD	AC1,(R4)			
10868	053410	004767	000026		JSR	PC,CHEC10			
10869	053414	012704	053504		MOV	2TS10D4,R4			
10870	053420	012701	003122		MOV	2TS10C,R1			
10871	053424	174011			STD	AC0,(R1)			
10872	053426	004767	000010		JSR	PC,CHEC10			
10873	053432	012637	000004		MOV	2SP,204			
10874									
10875									
10876	053436	000167	000052		IMP	FIN10			
10877									
10878	053442	012707	000004		CHEC10:	MOV	24,R4		
10879	053446	022421			CH10:	CMP	(R4), (R1)		

Address	Op-Code	Op-Code Hex	Op-Code Dec	Instruction	Comments
10880	053450	001401		BEQ	CH10
10881	053452	104003		ERROR	IF YES GO ON
10882					IFPP ERROR
10883	053454	077304		CH10:	NO GO TO ERROR
10884	053456	000707		RTS	ARE WE DONE
10885					IF YES RETURN
10886	053460	104003		IF 10:	IFPP ERROR
10887					ODD ADDRESS TRAP
10888	053462	000006		RTT	
10889					
10890	053464	000000		TS10D1:	.WORD 0
10891	053466	000000			.WORD 0
10892	053470	000000			.WORD 0
10893	053472	000000			.WORD 0
10894	053474	177777		TS10D2:	.WORD 177777
10895	053476	111236			.WORD 111236
10896	053500	100045			.WORD 100045
10897	053502	003651			.WORD 3651
10898	053504	000000		TS10D4:	.WORD 0
10899	053506	000000			.WORD 0
10900	053510	100045			.WORD 100045
10901	053512	003651			.WORD 3651
10902	053514			FIN10:	
10903					
10904	053514			IFPP11:	
10905					TEST FDS1 SINGLE OPERAND MODE 0
10906	053514	005037	133436	CLR	00TRPFLG
10907	053520	012704	000200	MOV	0200,R4
10908	053524	170104		LDPPS	R4
10909	053526	012704	053600	MOV	0TS11D1,R4
10910	053532	172414		LDD	(R4),AC0
10911	053534	170400		CLRD	AC0
10912	053536	170203		STEPS	R3
10913	053540	012704	003122	MOV	0TS1LOC,R4
10914	053544	174014		STD	AC0,(R4)
10915	053546	012701	000004	MOV	04,R1
10916	053552	022427	000000	1\$:	CMP (R4),00
10917	053556	001401			BEQ 2\$
10918	053560	104003			ERROR 3\$
10919					IFPP ERROR
10920	053562	077105		2\$:	NO GO TO ERROR
10921	053564	020327	000204		ARE WE DONE
10922	053570	001401			CHECK FPS
10923	053572	104003			IFPP ERROR
10924					NO GO TO ERROR
10925	053574			3\$:	
10926					
10927	053574	000167	000010		JMP FIN11
10928					
10929	053600	177777		TS11D1:	.WORD 177777
10930	053602	177777			.WORD 177777
10931	053604	177777			.WORD 177777
10932	053606	177777			.WORD 177777
10933	053610			FIN11:	
10934					
10935	053610			IFPP11:	

```

10936      TEST FOST SOP MODE 0 WITH ILLEGAL AC7
10937 053610 005037 133436      CLR      @TRPFLG      ;CLEAR TRAP FLAG,
10938 053614 012703 040200      MOV      @40200,R3      ;SETUP TO LOAD FPS,
10939 053620 170103              LD FPS      R3      ;SET FID=1, AND FD=1
10940 053622 170407              CLRD      AC7      ;TEST INSTRUCTION
10941 053624 170204              STEPS     R4      ;GET FPS
10942 053626 170305              STST      R5      ;GET FEC
10943 053630 022704 140200      CMP      @140200,R4      ;IS FPS CORRECT
10944 053634 001401              BEQ       1$      ;YES GO ON
10945 053636 104003              ERROR     *3      ;FPP ERROR
10946                                ;NO GO TO ERROR
10947 053640 022705 000002      1$:      CMP      @2,R5      ;IS FEC CORRECT
10948 053644 001401              BEQ       2$      ;YES GO ON
10949 053646 104003              ERROR     *3      ;FPP ERROR
10950                                ;NO GO TO ERROR
10951 053650      2$:
10952      ;
10953 053650      ;SEF13:
10954      ;
10955 053650 013746 000004      ;TEST FOST SOP MODE 1
10956 053654 012737 053770 000004      MOV      @@4,-(SP)      ;SAVE TIMEOUT VECTOR
10957 053662 005037 133436      MOV      @TSE13,@@4      ;SETUP NEW VECTOR
10958 053666 012702 000200      CLR      @TRPFLG      ;CLEAR TRAP FLAG
10959 053672 170103              MOV      @200,R2      ;SETUP TO LOAD FPS
10960 053674 012705 000004      LD FPS      R2      ;SET FD=1
10961 053700 012704 003122      MOV      @4,R5      ;INIT COUNTER
10962 053704 012724 177777      MOV      @TSTLOC,R4      ;SETUP POINTER TO TEST LOCATION
10963 053710 077503 100$:      MOV      @177777,(R4)      ;MOVE ALL ONES TO TEST LOCATION
10964 053712 012702 003122      SOB      R5,100$      ;ARE WE DONE
10965 053716 170412              MOV      @TSTLOC,R2      ;SETUP POINTER TO DATA
10966 053720 170203              CLRD      (R2)      ;TEST INSTRUCTION
10967 053722 020227 003122      STEPS     R3      ;GET FPS
10968 053726 001401              CMP      R2,@TSTLOC      ;WAS R2 ALTERED
10969 053730 104003              BEQ       1$      ;NO GO ON
10970                                ;FPP ERROR
10971 053732 012701 000004      1$:      MOV      @4,R1      ;YES GO TO ERROR
10972 053736 022227 000000      2$:      CMP      (R2)+,@0      ;INIT COUNTER
10973 053742 001401              ;CHECK LOCATION FOR 0
10974 053744 104003              BEQ       3$      ;OK GO ON
10975                                ;FPP ERROR
10976                                ;NO GO TO ERROR
10977 053746 077105      3$:      SOB      R1,3$      ;ARE WE DONE
10978 053750 020327 000204      CMP      R3,@204      ;CHECK FPS
10979 053754 001401              BEQ       4$      ;OK GO ON
10980                                ;FPP ERROR
10981 053760 012637 000004      4$:      MOV      (SP)+,@@4      ;RESTORE VECTOR
10982                                ;
10983      ;
10984 053764 000167 000004      JMP      FIN13
10985      ;
10986 053770 104003      5$:      ERROR     *3      ;FPP ERROR
10987                                ;ODD ADDRESS TRAP
10988 053772 000006      RTT      ;RETURN
10989      ;
10990 053774      FIN13:
10991      ;

```

10992	053774				TSEP14:		
10993					:	TEST FDS1 SOP MODE 2	
10994	053774	013746	000004		MOV	004, (SP)	;SAVE TIMEOUT VECTOR
10995	054006	012737	054120	000004	MOV	0TSEP14,004	;SETUP NEW VECTOR
10996	054006	005037	133436		CLR	00TRPFLG	;CLEAR TRAP FLAG
10997	054012	012702	000200		MOV	0200,R2	;SETUP TO LOAD FPS
10998	054016	170102			LDFPS	R2	;SET FD+1
10999	054020	012705	000004		MOV	04,R5	;INIT COUNTER
11000	054024	012704	003122		MOV	0TSTLOC,R4	;SETUP POINTER TO TEST LOCATION
11001	054030	012704	177777		100\$:	MOV	0177777,(R4);
11002	054034	077503			SOB	R5,100\$	;ARE WE DONE
11003	054036	012702	003122		MOV	0TSTLOC,R2	;SETUP POINTER TO DATA
11004	054042	170422			CLRD	(R2);	;TEST INSTRUCTION
11005	054044	170203			STEPS	R3	;GET FPS
11006	054046	020227	003132		CMP	R2,0TSTLOC+10	;IS R2 CORRECT
11007	054052	001401			BEQ	1\$	;YES GO ON
11008	054054	104003			ERROR	+3	;FPP ERROR
11009							;NO GO TO ERROR
11010	054056	012702	003122		1\$:	MOV	0TSTLOC,R2
11011	054062	012701	000004		MOV	04,R1	;SETUP POINTER TO DATA
11012	054066	022227	000000		2\$:	CMP	(R2)+,00
11013	054072	001401			BEQ	3\$	;CHECK LOCATION FOR 0
11014	054074	104003			ERROR	+3	;YES GO ON
11015							;FPP ERROR
11016	054076	077105			3\$:	SOB	R1,2\$
11017	054100	020327	000204		CMP	R3,0204	;ARE WE DONE
11018	054104	001401			BEQ	4\$	;CHECK FPS
11019	054106	104003			ERROR	+3	;OK GO ON
11020							;FPP ERROR
11021	054110	012637	000004		4\$:	MOV	(SP)+,004
11022							;NO GO TO ERROR
11023							;RESTORE VECTOR
11024	054114	000167	000004		:	JMP	FIN14
11025					:		
11026	054120	104003			TSEP14:	ERROR	+3
11027							;FPP ERROR
11028	054122	000006				RTI	;ODD ADDRESS TRAP
11029							;RETURN
11030	054124	000240			FIN14:	NOP	
11031					:		
11032					:		
11033					:		
11034					:	TEST FDS1 SOP MODE 3	
11035					:		
11036					:		
11037	054126				TSEP15:		
11038					:		
11039	054126	013746	000004		MOV	004, (SP)	;SAVE TIMEOUT VECTOR
11040	054132	012737	054306	000004	MOV	0TSEP15,004	;SETUP NEW VECTOR
11041	054140	005037	133436		CLR	00TRPFLG	;CLEAR TRAP FLAG
11042	054144	012702	000200		MOV	0200,R2	;SETUP TO LOAD FPS
11043	054150	170102			LDFPS	R2	;SET FD+1
11044	054152	012705	000011		MOV	09,R5	;INIT COUNTER
11045	054156	012704	003122		MOV	0TSTLOC,R4	;SETUP POINTER TO TEST LOCATION
11046	054162	012704	177777		100\$:	MOV	0177777,(R4);
11047	054166	077503			SOB	R5,100\$	;ARE WE DONE

CORDBO RDJ11 B CLUSTER MACY11 3001046) 05 APR 84 16:45 PAGE 202
COKDAB,P11 05-APR 84 16:45 FLOATING POINT TESTS

SEQ 0201

```

11048 054170 012737 003134 003122 MOV #TSTLOC+12,#0TSTLOC ;INIT TEST LOCATION
11049 054176 012702 003122 MOV #TSTLOC,R2 ;SETUP POINTER TO DATA
11050 054202 170432 CLRD @R2) ; TEST INSTRUCTION
11051 054204 170203 STEPS R3 ;GET FPS
11052 054206 020227 003124 CMP R2,#TSTLOC+2 ;IS R2 CORRECT
11053 054212 001401 BEQ 1$ ;YES GO ON
11054 054214 104003 ERROR +3 ;FPP ERROR
11055 ;NO GO TO ERROR
11056 054216 012702 003122 1$: MOV #TSTLOC,R2 ;SETUP POINTER TO DATA
11057 054222 022227 003134 CMP (R2)+,#TSTLOC+12 ;IS DATA CORRECT
11058 054226 001401 BEQ 2$ ;YES GO ON
11059 054230 104003 ERROR +3 ;FPP ERROR
11060 ;NO GO TO ERROR
11061 054232 012701 000004 2$: MOV #4,R1 ;INIT COUNTER
11062 054236 022227 177777 3$: CMP (R2)+,#177777 ;IS LOCATION ALL ONES
11063 054242 001401 BEQ 4$ ;YES GO ON
11064 054244 104003 ERROR +3 ;FPP ERROR
11065 ;NO GO TO ERROR
11066 054246 077105 4$: SOB R1,3$ ;ARE WE DONE
11067 054250 012701 000004 MOV #4,R1 ;INIT COUNTER
11068 054254 022227 000000 5$: CMP (R2)+,#0 ;IS LOCATION 0
11069 054260 001401 BEQ 6$ ;YES GO ON
11070 054262 104003 ERROR +3 ;FPP ERROR
11071 ;NO GO TO ERROR
11072 054264 077105 6$: SOB R1,5$ ;ARE WE DONE
11073 054266 020327 000204 CMP R3,#204 ;CHECK FPS
11074 054272 001401 BEQ 7$ ;OK GO ON
11075 054274 104003 ERROR +3 ;FPP ERROR
11076 ;NO GO TO ERROR
11077 054276 012637 000004 7$: MOV (SP)+,#04 ;RESTORE VECTOR
11078
11079
11080 054302 000167 000004 JMP FIN15
11081
11082 054306 104003 15F15: ERROR +3 ;FPP ERROR
11083 ;ODD ADDRESS TRAP
11084 054310 000006 RTI ;RETURN
11085
11086 054312 000240 FIN15: NOP
11087
11088
11089
11090 ;TEST FOST SOP MODE 4
11091
11092
11093 054314 15F16:
11094
11095 054314 013746 000004 MOV #04,(SP) ;SAVE TIMEOUT VECTOR
11096 054320 012737 054452 000004 MOV #TSTLOC+16,#04 ;SETUP NEW VECTOR
11097 054326 005037 133436 CLRD @TSTLOC ;CLEAR TRAP FLAG
11098 054332 012702 000200 MOV #200,R2 ;SETUP TO LOAD FPS
11099 054336 170102 STEPS R2 ;GET FPS
11100 054340 012705 000010 MOV #8,R5 ;INIT COUNTER
11101 054344 012704 003122 MOV #TSTLOC,R4 ;SETUP POINTER TO TEST LOCATION
11102 054350 012704 177777 100$: MOV #177777,(R4) ;INIT TEST LOCATION
11103 054354 077503 SOB R5,100$ ;ARE WE DONE

```

11104	054356	012702	003132		MOV	#15TLOC+10,R2		;SETUP POINTER TO DATA
11105	054362	170442			CLRD	(R2)		;TEST INSTRUCTION
11106	054364	170203			STPS	R3		;GET FPS
11107	054366	020227	003122		CMP	R2,#15TLOC		;IS R2 CORRECT
11108	054372	001401			BEQ	1\$		;YES GO ON
11109	054374	104003			ERROR	+3		;FPP ERROR
11110								;NO GO TO ERROR
11111	054376	012701	000004	1\$:	MOV	#4,R1		;INIT COUNTER
11112	054402	022227	000000	2\$:	CMP	(R2)+,#0		;IS LOCATION 0
11113	054406	001401			BEQ	3\$		;YES GO ON
11114	054410	104003			ERROR	+3		;FPP ERROR
11115								;NO GO TO ERROR
11116	054412	077105		3\$:	SOB	R1,2\$		;ARE WE DONE
11117	054414	012701	000004		MOV	#4,R1		;INIT COUNTER
11118	054420	022227	177777	4\$:	CMP	(R2)+,#177777		;IS LOCATION UNCHANGED
11119	054424	001401			BEQ	5\$		;YES GO ON
11120	054426	104003			ERROR	+3		;FPP ERROR
11121								;NO GO TO ERROR
11122	054430	077105		5\$:	SOB	R1,4\$		;ARE WE DONE
11123	054432	020327	000204		CMP	R3,#204		;CHECK FPS
11124	054436	001401			BEQ	6\$		;OK GO ON
11125	054440	104003			ERROR	+3		;FPP ERROR
11126								;NO GO TO ERROR
11127	054442	012637	000004	6\$:	MOV	(SP)+,#04		;RESTORE VECTOR
11128								
11129								
11130	054446	000167	000004		JMP	FIN16		
11131								
11132	054452	104003		TSF16:	ERROR	+3		;FPP ERROR
11133								;ADD ADDRESS TRAP
11134	054454	000006			RTT			;RETURN
11135								
11136	054456	000240		FIN16:	NOP			
11137								
11138								
11139								
11140								
11141								
11142								
11143	054460			TSFP17:				
11144								
11145	054460	013746	000004		MOV	#24,(SP)		;SAVE TIMEOUT VECTOR
11146	054464	012737	054634	000004	MOV	#17,#24		;SETUP NEW VECTOR
11147	054472	005037	133436		CLR	#TRPFLG		;CLEAR TRAP FLAG
11148	054476	012702	000200		MOV	#200,R2		;SETUP TO LOAD FPS
11149	054502	170102			LD FPS	R2		;SET FPS
11150	054504	012705	000011		MOV	#9,R5		;INIT COUNTER
11151	054510	012704	003122		MOV	#15TLOC,R4		;SETUP POINTER TO TEST LOCATION
11152	054514	012724	177777	100\$:	MOV	#177777,(R4)+		;INIT TEST LOCATION
11153	054520	077503			SOB	R5,100\$		;ARE WE DONE
11154	054522	012737	003134	003122	MOV	#15TLOC+12,#15TLOC		;INIT TEST LOCATION
11155	054530	012702	003124		MOV	#15TLOC+2,R2		;SETUP POINTER TO DATA
11156	054534	170452			CLRD	#(R2)		;TEST INSTRUCTION
11157	054536	170203			STPS	R3		;GET FPS
11158	054540	020227	003122		CMP	R2,#15TLOC		;IS R2 CORRECT
11159	054544	001401			BEQ	1\$		;YES GO ON


```

11160 054546 104003          ERROR      +3          ;FPP ERROR
11161                                     ;NO GO TO ERROR
11162 054550 022227 003134    1$:      CMP      (R2)+,01STLOC+12          ;IS DATA CORRECT
11163 054554 001401                                     ;YES GO ON
11164 054556 104003          ERROR      +3          ;FPP ERROR
11165                                     ;NO GO TO ERROR
11166 054560 012701 000004    2$:      MOV      #4,R1          ;INIT COUNTER
11167 054564 022227 177777    3$:      CMP      (R2)+,0177777          ;IS LOCATION ALL ONES
11168 054570 001401                                     ;YES GO ON
11169 054572 104003          ERROR      +3          ;FPP ERROR
11170                                     ;NO GO TO ERROR
11171 054574 077105    4$:      SOB      R1,5$          ;ARE WE DONE
11172 054576 012701 000004    MOV      #4,R1          ;INIT COUNTER
11173 054602 022227 000000    5$:      CMP      (R2)+,00          ;IS LOCATION 0
11174 054606 001401                                     ;YES GO ON
11175 054610 104003          ERROR      +3          ;FPP ERROR
11176                                     ;NO GO TO ERROR
11177 054612 077105    6$:      SOB      R1,5$          ;ARE WE DONE
11178 054614 020327 000204    CMP      R5,0204          ;CHECK EPS
11179 054620 001401                                     ;OK GO ON
11180 054622 104003          ERROR      +3          ;FPP ERROR
11181                                     ;NO GO TO ERROR
11182 054624 012637 000004    7$:      MOV      (SP)+,004          ;RESTORE VECTOR
11183                                     ;
11184                                     ;
11185 054630 000167 000004          JMP      FIN17
11186                                     ;
11187 054634 104003    TSF17:  ERROR      +3          ;FPP ERROR
11188                                     ;ODD ADDRESS TRAP
11189                                     ;RETURN
11189 054636 000006          RTT
11190                                     ;
11191 054640 000240    FIN17:  NOP
11192                                     ;
11193                                     ;
11194                                     ;
11195                                     ;TEST FOST SOP MODE 6
11196                                     ;
11197                                     ;
11198 054642    TSF20:
11199                                     ;
11200 054642 005037 153436          CLR      @TRPFLG          ;CLEAR TRAP FLAG
11201 054646 013746 000004          MOV      #04, (SP)          ;SAVE TIMEOUT VECTOR
11202 054652 012737 055006 000004    MOV      #TSF20,004          ;SETUP NEW VECTOR
11203 054660 012702 000200          MOV      #200,R2          ;SETUP TO LOAD EPS
11204 054664 170102          LD EPS R2          ;SET ED-1
11205 054666 012705 000010          MOV      #8,R5          ;INIT COUNTER
11206 054672 012704 003122          MOV      #1STLOC,R4          ;SETUP POINTER TO TEST LOCATION
11207 054676 012724 177777    100$:  MOV      #177777,(R4)+          ;INIT TEST LOCATION
11208 054702 077505          SOB      R5,100$          ;ARE WE DONE
11209 054704 012702 003123          MOV      #1STLOC+1,R2          ;SETUP POINTER TO DATA
11210 054710 170462 000007          C/RD      7(R2)          ;TEST INSTRUCTION
11211 054714 170703          ST EPS R5          ;GET EPS
11212 054716 020327 003123          CMP      R2,01STLOC+1          ;IS R2 CORRECT
11213 054722 001401                                     ;YES GO ON
11214 054724 104003          ERROR      +3          ;FPP ERROR
11215                                     ;NO GO TO ERROR

```

CORDBAR RD.111 B CLUSTER MACY11 3001046) 05 APR 84 16:45 PAGE 205
 CORDAB.F11 05-APR 84 16:45 FLOATING POINT TESTS

SEQ 0204

11216	054726	012702	003122	1\$:	MOV	@TSTLOC,R2		;SETUP POINTER TO DATA
11217	054732	012701	000004		MOV	@4,R1		;INIT COUNTER
11218	054736	022227	177777	2\$:	CMP	(R2)+,@177777		;IS DATA CORRECT
11219	054742	001401			BEQ	3\$		;YES GO ON
11220	054744	104003			ERROR	+3		;FPP ERROR
11221								;NO GO TO ERROR
11222	054746	077105		3\$:	SOB	R1,2\$		;ARE WE DONE
11223	054750	012701	000004		MOV	@4,R1		;INIT COUNTER
11224	054754	022227	000000	4\$:	CMP	(R2)+,@0		;IS DATA CORRECT
11225	054760	001401			BEQ	5\$		;YES GO ON
11226	054762	104003			ERROR	+3		;FPP ERROR
11227								;NO GO TO ERROR
11228	054764	077105		5\$:	SOB	R1,4\$		;ARE WE DONE
11229	054766	022227	000204		CMP	R3,@204		;IS FPS CORRECT
11230	054772	001401			BEQ	6\$		;YES GO ON
11231	054774	104003			ERROR	+3		;FPP ERROR
11232								;NO GO TO ERROR
11233	054776	012657	000004	6\$:	MOV	(SP)+,@04		;RESTORE VECTOR
11234								
11235								
11236	055002	000167	000004		JMP	FIN20		
11237								
11238	055006	104003		TSF20:	ERROR	+3		;FPP ERROR
11239								;ODD ADDRESS TRAP
11240	055010	000006			RTT			;RETURN
11241								
11242	055012	000240		FIN20:	NOP			
11243								
11244								
11245								
11246								
11247								
11248								
11249	055014			TSFP21:				
11250								
11251	055014	005037	133436		CLR	@TRPFLG		;CLEAR TRAP FLAG
11252	055020	013746	000004		MOV	@04,(SP)		;SAVE TIMEOUT VECTOR
11253	055024	012737	055176	000004	MOV	@TSFP1,@04		;SETUP NEW VECTOR
11254	055032	012702	000200		MOV	@200,R2		;SETUP TO LOAD FPS
11255	055036	170102			LOFPS	R2		;SET FD=1
11256	055040	012705	000010		MOV	@8,R5		;INIT COUNTER
11257	055044	012704	003122		MOV	@TSTLOC,R4		;SETUP POINTER TO TEST LOCATION
11258	055050	012724	177777	100\$:	MOV	@177777,(R4)+		;INIT TEST LOCATION
11259	055054	077503			SOB	R5,100\$		;ARE WE DONE
11260	055056	012737	003122	003132	MOV	@TSTLOC,@TSTLOC+10		;INIT TEST LOCATION
11261	055064	012702	003125		MOV	@TSTLOC+3,R2		;SETUP POINTER TO DATA
11262	055070	170422	000005		CLRD	@(R2)		;TEST INSTRUCTION
11263	055074	170203			STFPS	R3		;GET FPS
11264	055076	020227	003125		CMP	R2,@TSTLOC+3		;IS R2 CORRECT
11265	055102	001401			BEQ	1\$		;YES GO ON
11266	055104	104003			ERROR	+3		;FPP ERROR
11267								;NO GO TO ERROR
11268	055106	012702	003127	1\$:	MOV	@TSTLOC,R2		;SETUP POINTER TO DATA
11269	055112	012701	000004		MOV	@4,R1		;INIT COUNTER
11270	055116	022227	000000	2\$:	CMP	(R2)+,@0		;IS DATA CORRECT
11271	055122	001401			BEQ	3\$		;YES GO ON

K16

SEQ 0205

11272	055124	104003		ERROR	+3		;FPP ERROR
11273							;NO GO TO ERROR
11274	055126	077105		5\$:	SOB	R1,2\$	;ARE WE DONE
11275	055130	022227	003122		CMP	(R2)+,4TSTLOC	;IS DATA CORRECT
11276	055134	001401			BEQ	4\$	;YES GO ON
11277	055136	104003		ERROR	+3		;FPP ERROR
11278							;NO GO TO ERROR
11279	055140	012701	000003	4\$:	MOV	#3,R1	;INIT COUNTER
11280	055144	022227	177777	5\$:	CMP	(R2)+,0177777	;IS DATA CORRECT
11281	055150	001401			BEQ	6\$	;YES GO ON
11282	055152	104003		ERROR	+3		;FPP ERROR
11283							;NO GO TO ERROR
11284	055154	077105		6\$:	SOB	R1,5\$	;ARE WE DONE
11285	055156	020327	000204		CMP	R3,0204	;CHECK FPS
11286	055162	001401			BEQ	7\$	;OK GO ON
11287	055164	104003		ERROR	+3		;FPP ERROR
11288							;NO GO TO ERROR
11289	055166	012637	000004	7\$:	MOV	(SP)+,004	;RESTORE VECTOR
11290							
11291							
11292	055172	000167	000004		JMP	FIN21	
11293							
11294	055176	104003		TSF21:	ERROR	+3	;FPP ERROR
11295							;ODD ADDRESS TRAP
11296	055200	000006			RTT		;RETURN
11297							
11298	055202	000240		FIN21:	NOP		
11299							
11300							
11301							
11302							
11303							
11304							
11305	055204			TSFP22:			
11306							
11307	055204	005037	133436		CLR	00TRPFLG	;CLEAR TRAP FLAG
11308	055210	013746	000004		MOV	004, (SP)	;SAVE TIME OUT VECTOR
11309	055214	012737	055342	000004	MOV	0TSP22,004	;SETUP NEW VECTOR
11310	055222	012702	000200		MOV	0200,R2	;SETUP TO LOAD FPS
11311	055226	170102			LD FPS	R2	;SET FD=1
11312	055230	012705	000010		MOV	08,,R5	;INIT COUNTER
11313	055234	012704	003122		MOV	0TSTLOC,R4	;SETUP POINTER TO TEST LOCATION
11314	055240	012724	177777	100\$:	MOV	0177777,(R4)+	;INIT TEST LOCATION
11315	055244	077503			SOB	R5,100\$	;ARE WE DONE
11316	055246	012737	003132	003122	MOV	0TSTLOC+10,00TSTLOC	;INIT TEST LOCATION
11317	055254	170437	003122		CLRD	00TSTLOC	;TEST INSTRUCTION
11318	055260	170203			STEPS	R5	;GET FPS
11319	055262	012702	003122		MOV	0TSTLOC,R2	;SETUP POINTER TO DATA
11320	055266	012701	000004		MOV	04,R1	;INIT COUNTER
11321	055272	022227	000000	1\$:	CMP	(R1)+,00	;IS DATA CORRECT
11322	055276	001401			BEQ	2\$	;YES GO ON
11323	055300	104003		ERROR	+3		;FPP ERROR
11324							;NO GO TO ERROR
11325	055302	077105		2\$:	SOB	R1,1\$	;ARE WE DONE
11326	055304	012701	000004		MOV	04,R1	;INIT COUNTER
11327	055310	022227	177777	4\$:	CMP	(R2)+,0177777	;IS DATA CORRECT

```

11328 055314 001401      BEQ      4$      ;YES GO ON
11329 055316 104003      ERROR    +3      ;FPP ERROR
11330                                     ;NO GO TO ERROR
11331 055320 077105      SOB      R1,3$      ;ARE WE DONE
11332 055322 020327 000204  CMP      R3,#204      ;CHECK FPS
11333 055326 001401      BEQ      5$      ;OK GO ON
11334 055330 104003      ERROR    +3      ;FPP ERROR
11335                                     ;NO GO TO ERROR
11336 055332 012637 000004 5$:      MOV      (SP)+,#04      ;RESTORE VECTOR
11337                                     ;
11338                                     ;
11339 055336 000167 000004      JMP      FIN22
11340                                     ;
11341 055342 104003      TSF22:  ERROR    +3      ;FPP ERROR
11342                                     ;ODD ADDRESS TRAP
11343 055344 000006      RTT                                     ;RETURN
11344                                     ;
11345 055346 000240      FIN22:  NOP
11346                                     ;
11347                                     ;
11348                                     ;
11349                                     ;TEST FOST SOP MODE 6 GR7
11350                                     ;
11351                                     ;
11352 055350      TSFP23:
11353                                     ;
11354 055350 005037 133436      CLR      @TRPFLG      ;CLEAR TRAP FLAG
11355 055354 013746 000004      MOV      @04, (SP)      ;SAVE TIMEOUT VECTOR
11356 055360 012737 055462 000004  MOV      @TSF23,@04      ;SETUP NEW VECTOR
11357 055366 012702 000200      MOV      @200,R2      ;SETUP TO LOAD FPS
11358 055372 170102      LD FPS      R2      ;SET FD+1
11359 055374 012705 000004      MOV      @4,R5      ;INIT COUNTER
11360 055400 012704 003122      MOV      @ISTLOC,R4      ;SETUP POINTER TO TEST LOCATION
11361 055404 012724 177777 100$:  MOV      @177777,(R4)+      ;INIT TEST LOCATION
11362 055410 077503      SOB      R5,100$      ;ARE WE DONE
11363 055412 170467 125504      CLRD     ISTLOC      ;TEST INSTRUCTION
11364 055416 170203      STEPS     R3      ;GET FPS
11365 055420 012701 000004      MOV      @4,R1      ;INIT COUNTER
11366 055424 012702 003122      MOV      @ISTLOC,R2      ;SETUP POINTER TO DATA
11367 055430 022227 000000 1$:      CMP      (R2)+,#0      ;IS DATA CORRECT
11368 055434 001401      BEQ      2$      ;YES GO ON
11369 055436 104003      ERROR    +3      ;FPP ERROR
11370                                     ;NO GO TO ERROR
11371 055440 077105      SOB      R1,1$      ;ARE WE DONE
11372 055442 020327 000204 2$:      CMP      R3,#204      ;CHECK FPS
11373 055446 001401      BEQ      3$      ;OK GO ON
11374 055450 104003      ERROR    +3      ;FPP ERROR
11375                                     ;NO GO TO ERROR
11376 055452 012637 000004 3$:      MOV      (SP)+,#04      ;RESTORE VECTOR
11377                                     ;
11378                                     ;
11379 055456 000167 000004      JMP      FIN23
11380                                     ;
11381 055462 104003      TSF23:  ERROR    +3      ;FPP ERROR
11382                                     ;ODD ADDRESS TRAP
11383 055464 000006      RTT                                     ;

```

CORDABO RDJ11-B CLUSTER MACY11 30(1046) 05 APR 84 16:45 PAGE 208
 CORDAB.P11 05-APR 84 16:45 FLOATING POINT TESTS

SEQ 0207

```

11384 055466 000240 FIN23: NOP
11385
11386
11387
11388
11389
11390
11391 055470 TSFP24:
11392
11393 055470 005037 133456 CLR      @TRPFLG      ;CLEAR TRAP FLAG
11394 055474 013746 000004 MOV      @#4, -(SP)    ;SAVE TIMEOUT VECTOR
11395 055500 012737 055636 000004 MOV      @TSFP24, @#4    ;SETUP NEW VECTOR
11396 055506 012702 000700 MOV      @200, R2      ;SETUP TO LOAD FPS
11397 055512 170102 LDDEPS  R2          ;SET FD=1
11398 055514 012705 000010 MOV      @8, R5          ;INIT COUNTER
11399 055520 012704 003122 MOV      @TSTLOC, R4      ;SETUP TEST LOCATION POINTER
11400 055524 012724 177777 100$: MOV      @177777, (R4),    ;INIT TEST LOCATION
11401 055530 077503 SOB      R5, 100$      ;ARE WE DONE
11402 055532 012737 003122 003132 MOV      @TSTLOC, @@TSTLOC+10 ;INIT TEST LOCATION
11403 055540 170477 125366 CLRD     @TSTLOC+10    ; TEST INSTRUCTION
11404 055544 170203 STEPS   R3          ;GET FPS
11405 055546 012702 003122 MOV      @TSTLOC, R2      ;SETUP POINTER TO DATA
11406 055552 012701 000004 MOV      @4, R1          ;INIT COUNTER
11407 055556 022227 000000 1$: CMP      (R2)+, @0      ;IS DATA CORRECT
11408 055562 001401 BEQ      2$          ;YES GO ON
11409 055564 104003 ERROR    +3          ;FPP ERROR
11410
11411 055566 077105 2$: SOB      R1, 1$          ;NO GO TO ERROR
11412 055570 022227 003122 CMP      (R2)+, @TSTLOC    ;ARE WE DONE
11413 055574 001401 BEQ      3$          ;IS DATA CORRECT
11414 055576 104003 ERROR    +3          ;YES GO ON
11415
11416 055600 012701 000003 3$: MOV      @3, R1          ;FPP ERROR
11417 055604 022227 177777 4$: CMP      (R2)+, @177777    ;NO GO TO ERROR
11418 055610 001401 BEQ      5$          ;INIT COUNTER
11419 055612 104003 ERROR    +3          ;IS DATA CORRECT
11420
11421 055614 077105 5$: SOB      R1, 4$          ;YES GO ON
11422 055616 020327 000204 CMP      R3, @204      ;FPP ERROR
11423 055622 001401 BEQ      6$          ;ARE WE DONE
11424 055624 104003 ERROR    +3          ;CHECK FPS
11425
11426 055626 012637 000004 6$: MOV      (SP)+, @#4    ;OK GO ON
11427
11428
11429 055632 000167 000004 JMP      FIN24      ;FPP ERROR
11430
11431 055636 104003 TSFP24: ERROR    +3          ;NO GO TO ERROR
11432
11433 055640 000006 RIT          ;RESTORE VECTOR
11434
11435 055642 000240 FIN24: NOP
11436
11437
11438
11439
;TEST CLRF

```



```

11440
11441
11442 055644
11443
11444 055644 005037 133436
11445 055650 005002
11446 055652 170102
11447 055654 012705 000004
11448 055660 012704 003122
11449 055664 012724 177777
11450 055670 077503
11451 055672 012702 003122
11452 055676 170422
11453 055700 170203
11454 055702 020227 003126
11455 055706 001401
11456 055710 104003
11457
11458 055712 012702 003122
11459 055716 012701 000002
11460 055722 022227 000000
11461 055726 001401
11462 055730 104003
11463
11464 055732 077105
11465 055734 012701 000002
11466 055740 022227 177777
11467 055744 001401
11468 055746 104003
11469
11470 055750 077105
11471 055752 020327 000004
11472 055756 001401
11473 055760 104003
11474
11475 055762
11476
11477
11478
11479
11480
11481
11482 055762
11483
11484 055762 005037 133436
11485 055766 005004
11486 055770 170104
11487 055772 170567 000244
11488 055776 170203
11489 056000 020327 000004
11490 056004 001401
11491 056006 104003
11492
11493 056010 012704 056244
11494 056014 012702 056244
11495 056020 004767 000200

;
;TSFP25:
;
; CLR B*TRPFLG ;CLEAR TRAP FLAG
; CLR R2 ;SETUP TO LOAD FPS
; LDFPS R2 ;SET FD=0
; MOV #4,R5 ;INIT COUNTER
; MOV #TSTLOC,R4 ;SETUP POINTER TO TEST LOCATION
1001: MOV #177777,(R4); ;INIT TEST LOCATION
; SOB R5,1001 ;ARE WE DONE
; MOV #TSTLOC,R2 ;SETUP POINTER TO DATA
; CLRF (R2); ;TEST INSTRUCTION
; STFPS R3 ;GET FPS
; CMP R2,#TSTLOC+4 ;IS R2 CORRECT
; BEQ 11 ;YES GO ON
; ERROR +3 ;FPP ERROR
; ;NO GO TO ERROR
; ;SETUP POINTER TO DATA
; ;INIT COUNTER
; ;IS DATA CORRECT
21: CMP (R2)+,#0 ;YES GO ON
; BEQ 31 ;FPP ERROR
; ERROR +3 ;NO GO TO ERROR
; ;ARE WE DONE
; ;INIT COUNTER
; ;IS DATA CORRECT
41: CMP (R2)+,#177777 ;YES GO ON
; BEQ 51 ;FPP ERROR
; ERROR +3 ;NO GO TO ERROR
; ;ARE WE DONE
; ;CHECK FPS
51: SOB R1,41 ;OK GO ON
; CMP R3,#4 ;FPP ERROR
; BEQ 61 ;NO GO TO ERROR
; ERROR +3
; ;TEST INSTE AND TESTD
; ;TSFP26:
;
; CLR B*TRPFLG ;CLEAR TRAP FLAG
; CLR R4 ;SETUP TO LOAD FPS
; LDFPS R4 ;SET FD=0
; TEST 152600 ;TEST INSTRUCTION
; STFPS R3 ;GET FPS
; CMP R3,#4 ;CHECK FPS
; BEQ 11 ;OK GO ON
; ERROR +3 ;FPP ERROR
; ;NO GO TO ERROR
; ;SETUP POINTERS TO DATA
; ;CHECK IF DATA IS CORRECT
11: MOV #152600,R4
; MOV #152603,R3
; JSR PC,CMLCP6

```

11496	056024	170537	056252		1STF	00TS26D1		1 TEST INSTRUCTION
11497	056030	170203			STEPS	R3		1GET FPS
11498	056032	020327	000010		CMP	R3,010		1CHECK FPS
11499	056036	001401			BEQ	03		1OK GO ON
11500	056040	104003			ERROR	13		1PPP ERROR
11501								1NO GO TO ERROR
11502	056042	012704	056252	23:	MOV	0TS26D1,R4		1SETUP POINTERS TO DATA
11503	056046	012702	056302		MOV	0TS26D4,R2		1
11504	056052	004767	000146		JSR	PC,CHEC26		1CHECK IF DATA IS CORRECT
11505	056056	170567	000200		1STF	TS26D2		1 TEST INSTRUCTION
11506	056062	170203			STEPS	R3		1GET FPS
11507	056064	020327	000000		CMP	R3,00		1CHECK FPS
11508	056070	001401			BEQ	33		1OK GO ON
11509	056072	104003			ERROR	13		1PPP ERROR
11510								1NO GO TO ERROR
11511	056074	012704	056262	33:	MOV	0TS26D2,R4		1SETUP POINTERS TO DATA
11512	056100	012702	056312		MOV	0TS26D5,R2		1
11513	056104	004767	000114		JSR	PC,CHEC26		1CHECK IF DATA IS CORRECT
11514	056110	012704	000200		MOV	0200,R4		1SETUP TO LOAD FPS
11515	056114	170104			LD FPS	R4		1SET FD=1
11516	056116	170537	056242		1STD	00TS26D0		1 TEST INSTRUCTION
11517	056122	170203			STEPS	R3		1GET FPS
11518	056124	020327	000204		CMP	R3,0204		1CHECK FPS
11519	056130	001401			BEQ	43		1OK GO ON
11520	056132	104003			ERROR	13		1PPP ERROR
11521								1NO GO TO ERROR
11522	056134	012704	056242	43:	MOV	0TS26D0,R4		1SETUP POINTERS TO DATA
11523	056140	012702	056272		MOV	0TS26D3,R2		1
11524	056144	004767	000054		JSR	PC,CHEC26		1CHECK IF DATA IS CORRECT
11525	056150	170567	000076		1STD	TS26D1		1 TEST INSTRUCTION
11526	056154	170203			STEPS	R3		1GET FPS
11527	056156	020327	000210		CMP	R3,0210		1CHECK FPS
11528	056162	001401			BEQ	53		1OK GO ON
11529	056164	104003			ERROR	13		1PPP ERROR
11530								1NO GO TO ERROR
11531	056166	012704	056252	53:	MOV	0TS26D1,R4		1SETUP POINTERS TO DATA
11532	056172	012702	056302		MOV	0TS26D4,R2		1
11533	056176	004767	000022		JSR	PC,CHEC26		1CHECK IF DATA IS CORRECT
11534	056202	170567	000054		1STD	TS26D2		1 TEST INSTRUCTION
11535	056206	170203			STEPS	R3		1GET FPS
11536	056210	020327	000200		CMP	R3,0200		1CHECK FPS
11537	056214	001401			BEQ	63		1OK GO ON
11538	056216	104003			ERROR	13		1PPP ERROR
11539								1NO GO TO ERROR
11540	056220			63:				
11541				1				
11542	056220	000167	000076		CMP	000076		
11543				1				
11544	056224	012701	000004	CHEC26:	MOV	04,R1		1INIT COUNTER
11545	056230	022427		13:	CMP	(R4), (R2)+		1IS DATA CORRECT
11546	056232	001401			BEQ	13		1YES GO ON
11547	056234	104003			ERROR	13		1PPP ERROR
11548								1NO GO TO ERROR
11549	056236	077104		13:	NOB	R1,13		1ARE WE DONE
11550	056240	000207			RTS	PC		1RETURN
11551				1				

11552	056242	000177		TS26D0:	.WORD	177	
11553	056244	177777			.WORD	177777	
11554	056246	177777			.WORD	177777	
11555	056250	177777			.WORD	177777	
11556	056252	177777		TS26D1:	.WORD	177777	
11557	056254	000000			.WORD	0	
11558	056256	000000			.WORD	0	
11559	056260	000000			.WORD	0	
11560	056262	077777		TS26D2:	.WORD	77777	
11561	056264	000000			.WORD	0	
11562	056266	000000			.WORD	0	
11563	056270	000000			.WORD	0	
11564	056272	000177		TS26D3:	.WORD	177	
11565	056274	177777			.WORD	177777	
11566	056276	177777			.WORD	177777	
11567	056300	177777			.WORD	177777	
11568	056302	177777		TS26D4:	.WORD	177777	
11569	056304	000000			.WORD	0	
11570	056306	000000			.WORD	0	
11571	056310	000000			.WORD	0	
11572	056712	077777		TS26D5:	.WORD	77777	
11573	056314	000000			.WORD	0	
11574	056316	000000			.WORD	0	
11575	056320	000000			.WORD	0	
11576	056322	000240		FIN26:	NOP		
11577				:			
11578				:			
11579				:			
11580				:TEST ABSF			
11581				:			
11582				:			
11583	056324			TSFP27:			
11584				:			
11585	056324	005037	133436		CLR	@TRAPFLAG	:CLEAR TRAP FLAG
11586	056330	005005			CLR	R5	:SETUP TO LOAD FPS
11587	056332	170105			LD FPS	R5	:SET FD=0
11588	056334	012701	000014		MOV	#12,R1	:INIT COUNTER
11589	056340	012704	003122		MOV	#TESTLOC,R4	:SETUP POINTER TO TEST LOCATION
11590	056344	012705	056550		MOV	#TEST700,R5	:SETUP POINTER TO TEST VALUE
11591	056350	012324		100%:	MOV	(R5), (R4)	:INIT TEST LOCATION
11592	056352	077102			SOB	R1,100%	:ARE WE DONE
11593	056354	012705	003122		MOV	#TESTLOC,R5	:SETUP POINTER TO DATA
11594	056360	170615			ABSF	(R5)	:TEST INSTRUCTION
11595	056362	170203			STEPS	R5	:GET FPS
11596	056364	020527	003122		CMP	R5,#TESTLOC	:IS R5 CORRECT
11597	056370	001401			BEQ	1%	:YES GO ON
11598	056372	104003			ERROR	1%	:PP ERROR
11599							:NO GO TO ERROR
11600	056374	012702	056600	1%:	MOV	#TEST700,R2	:SETUP POINTER TO DATA
11601	056376	004767	000177		JSR	PC,CHECK7	:CHECK IF DATA IS CORRECT
11602	056404	000327	000000		CMP	R5,00	:CHECK FPS
11603	056410	001401			BEQ	2%	:OK GO ON
11604	056412	104003			ERROR	1%	:PP ERROR
11605							:NO GO TO ERROR
11606	056414	012705	003132	2%:	MOV	#TESTLOC+10,R5	:SETUP POINTER TO DATA
11607	056420	170625			ABSF	(R5)	:TEST INSTRUCTION

11608	056422	170203		STEPS	R3		
11609	056424	020527	003136	CMP	R5, #TSTLOC+14		;GET FPS
11610	056430	001401		BEQ	33		;IS R5 CORRECT
11611	056432	104003		ERROR	33		;YES GO ON
11612							;FPP ERROR
11613	056434	012705	003132	33:	MOV	#TSTLOC+10,R5	;NO GO TO ERROR
11614	056440	012702	056610		MOV	#TS27D4,R2	;SETUP POINTER TO DATA
11615	056444	004767	000062		JSR	PC,CHEC27	
11616	056450	020327	000000		CMP	R3,#0	;CHECK IF DATA IS CORRECT
11617	056454	001401			BEQ	43	;CHECK FPS
11618	056456	104003			ERROR	33	;OK GO ON
11619							;FPP ERROR
11620	056460	012705	003122	43:	MOV	#TSTLOC,R5	;NO GO TO ERROR
11621	056464	170665	000020		ARSH	PC(R5)	;SETUP POINTER TO DATA
11622	056470	170203			STEPS	R3	;TEST INSTRUCTION
11623	056472	020527	003122		CMP	R5,#TSTLOC	;GET FPS
11624	056476	001401			BEQ	53	;IS R5 CORRECT
11625	056500	104003			ERROR	33	;YES GO ON
11626							;FPP ERROR
11627	056502	012705	003142	53:	MOV	#TSTLOC+20,R5	;NO GO TO ERROR
11628	056506	012702	056620		MOV	#TS27D5,R2	;SETUP POINTERS TO DATA
11629	056512	004767	000014		JSR	PC,CHEC27	
11630	056516	020327	000004		CMP	R3,#4	;CHECK IF DATA IS CORRECT
11631	056522	001401			BEQ	63	;CHECK FPS
11632	056524	104003			ERROR	33	;OK GO ON
11633							;FPP ERROR
11634	056526			63:			;NO GO TO ERROR
11635							
11636	056526	000167	000076		JMP	FIN27	
11637							
11638	056532	012701	000004	CHEC27:	MOV	#4,R1	;INIT COUNTER
11639	056536	022527		13:	CMP	(R5)+,(R2)+	;IS DATA CORRECT
11640	056540	001401			BEQ	23	;YES GO ON
11641	056542	104003			ERROR	33	;FPP ERROR
11642							;NO GO TO ERROR
11643	056544	077104		23:	SUB	R1,13	;ARE WE DONE
11644	056546	000207			RTS	PC	;RETURN
11645							
11646	056550	177777		TS27D0:	.WORD	177777	
11647	056552	177777			.WORD	177777	
11648	056554	177777			.WORD	177777	
11649	056556	177777			.WORD	177777	
11650	056560	000377		TS27D1:	.WORD	377	
11651	056562	175436			.WORD	175436	
11652	056564	136477			.WORD	136477	
11653	056566	000001			.WORD	1	
11654	056570	000177		TS27D2:	.WORD	177	
11655	056572	175436			.WORD	175436	
11656	056574	136477			.WORD	136477	
11657	056576	000001			.WORD	1	
11658	056600	077777		TS27D3:	.WORD	77777	
11659	056602	177777			.WORD	177777	
11660	056604	177777			.WORD	177777	
11661	056606	177777			.WORD	177777	
11662	056610	000377		TS27D4:	.WORD	377	
11663	056612	175436			.WORD	175436	

11664	056614	136477		.WORD	136477	
11665	056616	000001		.WORD	1	
11666	056620	000000	TS27D5:	.WORD	0	
11667	056622	000000		.WORD	0	
11668	056624	136477		.WORD	136477	
11669	056626	000001		.WORD	1	
11670	056630	000240	FIN27:	NOP		
11671						
11672						
11673						
11674						
11675						
11676						
11677	056632		TSFP30:			
11678						
11679	056632	005037	133436	CLR	@TRAPFLG	;CLEAR TRAP FLAG
11680	056636	012705	000200	MOV	@200,R5	;SETUP TO LOAD FPS
11681	056642	170105		LD FPS	R5	;SET FD=1
11682	056644	012701	000014	MOV	@12,R1	;INIT COUNTER
11683	056650	012704	003122	MOV	@TSTLOC,R4	;SETUP POINTER TO TEST LOCATION
11684	056654	012703	057060	MOV	@TS30D0,R3	;SETUP POINTER TO TEST VALUE
11685	056660	012324		MOV	(R3), (R4)	;INIT TEST LOCATION
11686	056662	077102		SUB	R1,100	;ARE WE DONE
11687	056664	012705	003122	MOV	@TSTLOC,R5	;SETUP POINTER TO DATA
11688	056670	170615		ABSD	(R5)	;TEST INSTRUCTION
11689	056672	170203		STEPS	R3	;GET FPS
11690	056674	020527	003122	CMP	R5,@TSTLOC	;IS R5 CORRECT
11691	056700	001401		BEQ	1\$	;YES GO ON
11692	056702	104003		ERROR	1\$	;FPP ERROR
11693						;NO GO TO ERROR
11694	056704	012702	057110	MOV	@TS30D3,R2	;SETUP POINTER TO DATA
11695	056710	004767	000126	JSR	PC,CHEC30	;CHECK IF DATA IS CORRECT
11696	056714	020327	000200	CMP	R3,@200	;CHECK FPS
11697	056720	001401		BEQ	2\$	;OK GO ON
11698	056722	104003		ERROR	2\$	;FPP ERROR
11699						;NO GO TO ERROR
11700	056724	012705	003132	MOV	@TSTLOC+10,R5	;SETUP POINTER TO DATA
11701	056730	170625		ABSD	(R5)	;TEST INSTRUCTION
11702	056732	170203		STEPS	R3	;GET FPS
11703	056734	020527	003142	CMP	R5,@TSTLOC+20	;IS R5 CORRECT
11704	056740	001401		BEQ	3\$	;YES GO ON
11705	056742	104003		ERROR	3\$	;FPP ERROR
11706						;NO GO TO ERROR
11707	056744	012705	003152	MOV	@TSTLOC+10,R5	;SETUP POINTERS TO DATA
11708	056750	012702	057120	MOV	@TS30D4,R2	
11709	056754	004767	000002	JSR	PC,CHEC30	;CHECK IF DATA IS CORRECT
11710	056760	020327	000200	CMP	R3,@200	;CHECK FPS
11711	056764	001401		BEQ	4\$	;OK GO ON
11712	056766	104003		ERROR	4\$	;FPP ERROR
11713						;NO GO TO ERROR
11714	056770	012705	003122	MOV	@TSTLOC,R5	;SETUP POINTER TO DATA
11715	056774	170625	000020	ABSD	(R5)	;TEST INSTRUCTION
11716	057000	170203		STEPS	R3	;GET FPS
11717	057002	020527	003122	CMP	R5,@TSTLOC	;IS R5 CORRECT
11718	057004	001401		BEQ	5\$	;YES GO ON
11719	057010	104003		ERROR	5\$	;FPP ERROR

```

11720 057012 012705 003142 5$: MOV #TS1LOC+20,R5 ;NO GO TO ERROR
11721 057016 012702 057130 MOV #TS30D5,R2 ;SETUP POINTERS TO DATA
11722 057022 004767 000014 JSR PC,CHEC30 ;CHECK IF DATA IS CORRECT
11723 057026 020327 000204 CMP R3,#204 ;CHECK EPS
11724 057032 001401 BEQ 6$ ;OK GO ON
11725 057034 104003 ERROR +3 ;FPP ERROR
11726 057036 6$: ;NO GO TO ERROR
11727 057038 000167 000016 JMP FIN30
11728 057042 012701 000004 ;CHEC30: MOV #4,R1 ;INIT COUNTER
11729 057046 020527 1$: CMP (R5)+,(R2)+ ;IS DATA CORRECT
11730 057050 001401 BEQ 2$ ;YES GO ON
11731 057052 104003 ERROR +3 ;FPP ERROR
11732 057054 077104 2$: SOB R1,1$ ;NO GO TO ERROR
11733 057056 000207 RTS PC ;ARE WE DONE
11734 057060 177777 ; ;RETURN
11735 057062 177777 TS30D0: .WORD 177777
11736 057064 177777 .WORD 177777
11737 057066 177777 .WORD 177777
11738 057070 000377 TS30D1: .WORD 377
11739 057072 175436 .WORD 175436
11740 057074 136477 .WORD 136477
11741 057076 000001 .WORD 1
11742 057100 000177 TS30D2: .WORD 177
11743 057102 175436 .WORD 175436
11744 057104 136477 .WORD 136477
11745 057106 000001 .WORD 1
11746 057110 077777 TS30D3: .WORD 77777
11747 057112 177777 .WORD 177777
11748 057114 177777 .WORD 177777
11749 057116 177777 .WORD 177777
11750 057120 000377 TS30D4: .WORD 377
11751 057122 175436 .WORD 175436
11752 057124 136477 .WORD 136477
11753 057126 000001 .WORD 1
11754 057130 000000 TS30D5: .WORD 0
11755 057132 000000 .WORD 0
11756 057134 000000 .WORD 0
11757 057136 000000 .WORD 0
11758 057140 000240 FIN30: NOP
11759 057142 005037 133436 ;TEST FPP31:
11760 057144 013742 000004 ; ;
11761 057152 012737 057254 000004 CLR #TRPFLG ;CLEAR TRAP FLAG
11762 057154 013742 000004 MOV #4,CMP ;SAVE TIMEOUT VECTOR
11763 057156 012737 057254 000004 MOV #TSF31,CMP ;SETUP NEW VECTOR

```

11776	057160	012702	000200		MOV	#200,R2		;SETUP TO LOAD FPS
11777	057161	170102			LD FPS	R2		;SET FD=1
11778	057166	170527	000005	ISD31:	ISID	#5		;TEST INSTRUCTION
11779	057172	000240			NOP			
11780	057174	000240			NOP			
11781	057176	000240			NOP			
11782	057200	170203			STEPS	R3		;GET FPS
11783	057202	020327	000204		CMP	R3,#204		;CHECK FPS
11784	057206	001401			BEQ	1\$		;OK GO ON
11785	057210	104003			ERROR	*3		;FPP ERROR
11786								;NO GO TO ERROR
11787	057212	012702	057170	1\$:	MOV	#ISD31,R2		;SETUP POINTER TO DATA
11788	057216	022227	000005		CMP	(R2),#5		;IS DATA CORRECT
11789	057222	001401			BEQ	2\$		;YES GO ON
11790	057224	104003			ERROR	*3		;FPP ERROR
11791								;NO GO TO ERROR
11792	057226	012701	000003	3\$:	MOV	#3,R1		;INIT COUNTER
11793	057232	022227	000240	4\$:	CMP	(R2),#240		;IS DATA CORRECT
11794	057236	001401			BEQ	4\$		;YES GO ON
11795	057240	104003			ERROR	*3		;FPP ERROR
11796								;NO GO TO ERROR
11797	057242	077105		4\$:	SUB	R1,3\$		;ARE WE DONE
11798	057244	012637	000004		MOV	(SP),#04		;RESTORE VECTOR
11799								
11800								
11801	057250	000167	000004		JMP	FIN31		
11802								
11803	057254	104003		ISF31:	ERROR	*3		;FPP ERROR
11804								;ADD ADDRESS TRAP
11805	057256	000006			R1			;RETURN
11806								
11807	057260	000240		FIN31:	NOP			
11808								
11809								
11810								
11811								
11812								
11813								
11814	057262			ISFP32:				
11815								
11816	057262	005037	133436		CLR	#TRAP16		;CLEAR TRAP FLAG
11817	057266	005005			CLR	R5		;SETUP TO LOAD FPS
11818	057270	170105			LD FPS	R5		;SET FD=0
11819	057272	012701	000014		MOV	#12,R1		;INIT COUNTER
11820	057276	012704	003122		MOV	#TESTLOC,R4		;SETUP POINTER TO TEST LOCATION
11821	057302	012703	057452		MOV	#TESTVAL,R3		;SETUP POINTER TO TEST VALUE
11822	057306	012324		1003:	MOV	(R3),(R4)		;INIT TEST LOCATION
11823	057310	077105			SUB	R1,100\$		;ARE WE DONE
11824	057312	170267	123604		NEG	TESTLOC		;TEST INSTRUCTION
11825	057316	170203			STEPS	R5		;GET FPS
11826	057320	012705	003122		MOV	#TESTLOC,R5		;SETUP POINTERS TO DATA
11827	057324	012702	057502		MOV	#TESTLOC,R2		
11828	057330	004767	000100		CLR	PC,CHECK32		;CHECK IF DATA IS CORRECT
11829	057334	000401	000000		CMP	R5,00		;CHECK FPS
11830	057340	001401			BEQ	1\$		;YES GO ON
11831	057342	104003			ERROR	*3		;FPP ERROR

11832									
11833	057344	170767	123562	1\$:	NEG	TSTLOC+10			;NO GO TO ERROR
11834	057350	170203			STEPS	R3			; TEST INSTRUCTION
11835	057352	012705	003142		MOV	#TSTLOC+10,R5			;GET FPS
11836	057356	012702	057512		MOV	#T532D4,R2			;SETUP POINTERS TO DATA
11837	057362	004767	000046		JSR	PC,CHEC32			
11838	057366	020327	000010		CMP	R3,#10			;CHECK IF DATA IS CORRECT
11839	057372	001401			BEQ	2\$			;CHECK FPS
11840	057374	104003			ERROR	*3			;OK GO ON
11841									;FPP ERROR
11842	057376	170767	123540	2\$:	NEG	TSTLOC+20			;NO GO TO ERROR
11843	057402	170203			STEPS	R3			; TEST INSTRUCTION
11844	057404	012705	003142		MOV	#TSTLOC+20,R5			;GET FPS
11845	057410	012702	057522		MOV	#T532D5,R2			;SETUP POINTERS TO DATA
11846	057414	004767	000014		JSR	PC,CHEC32			
11847	057420	020327	000004		CMP	R3,#4			;CHECK IF DATA IS CORRECT
11848	057424	001401			BEQ	3\$			;CHECK FPS
11849	057426	104003			ERROR	*3			;OK GO ON
11850									;FPP ERROR
11851	057430			3\$:					;NO GO TO ERROR
11852									
11853	057430	000167	000076		JMP	FIN52			
11854									
11855	057434	012701	000004		CHEC32:	MOV	#4,R1		;INIT COUNTER
11856	057440	020527		1\$:	CMP	(R5),,(R2),			;IS DATA CORRECT
11857	057442	001401			BEQ	2\$			;YES GO ON
11858	057444	104003			ERROR	*3			;FPP ERROR
11859									;NO GO TO ERROR
11860	057446	077104		2\$:	SOB	R1,1\$			;ARE WE DONE
11861	057450	000207			RTS	PC			;RETURN
11862									
11863	057452	170000			T532D0:	.WORD	170000		
11864	057454	003541				.WORD	3541		
11865	057456	177777				.WORD	177777		
11866	057460	172710				.WORD	172710		
11867	057462	070000			T532D1:	.WORD	70000		
11868	057464	003541				.WORD	3541		
11869	057466	177777				.WORD	177777		
11870	057470	172710				.WORD	172710		
11871	057472	000177				.WORD	177		
11872	057474	100000				.WORD	100000		
11873	057476	177777				.WORD	177777		
11874	057500	177007				.WORD	177007		
11875	057502	070000			T532D3:	.WORD	70000		
11876	057504	003541				.WORD	3541		
11877	057506	177777				.WORD	177777		
11878	057510	172710				.WORD	172710		
11879	057512	170000			T532D4:	.WORD	170000		
11880	057514	003541				.WORD	3541		
11881	057516	177777				.WORD	177777		
11882	057520	172710				.WORD	172710		
11883	057522	000000			T532D5:	.WORD	0		
11884	057524	000000				.WORD	0		
11885	057526	177777				.WORD	177777		
11886	057530	177007				.WORD	177007		
11887	057532	000240			FIN52:	NOP			

Address	Hex	Dec	Label	Op	Op1	Op2	Op3	Op4	Op5	Op6	Op7	Op8	Op9	Op10	Op11	Op12	Op13	Op14	Op15	Op16	Op17	Op18	Op19	Op20	Op21	Op22	Op23	Op24	Op25	Op26	Op27	Op28	Op29	Op30	Op31	Op32	Op33	Op34	Op35	Op36	Op37	Op38	Op39	Op40	Op41	Op42	Op43	Op44	Op45	Op46	Op47	Op48	Op49	Op50	Op51	Op52	Op53	Op54	Op55	Op56	Op57	Op58	Op59	Op60	Op61	Op62	Op63	Op64	Op65	Op66	Op67	Op68	Op69	Op70	Op71	Op72	Op73	Op74	Op75	Op76	Op77	Op78	Op79	Op80	Op81	Op82	Op83	Op84	Op85	Op86	Op87	Op88	Op89	Op90	Op91	Op92	Op93	Op94	Op95	Op96	Op97	Op98	Op99	Op100	Op101	Op102	Op103	Op104	Op105	Op106	Op107	Op108	Op109	Op110	Op111	Op112	Op113	Op114	Op115	Op116	Op117	Op118	Op119	Op120	Op121	Op122	Op123	Op124	Op125	Op126	Op127	Op128	Op129	Op130	Op131	Op132	Op133	Op134	Op135	Op136	Op137	Op138	Op139	Op140	Op141	Op142	Op143	Op144	Op145	Op146	Op147	Op148	Op149	Op150	Op151	Op152	Op153	Op154	Op155	Op156	Op157	Op158	Op159	Op160	Op161	Op162	Op163	Op164	Op165	Op166	Op167	Op168	Op169	Op170	Op171	Op172	Op173	Op174	Op175	Op176	Op177	Op178	Op179	Op180	Op181	Op182	Op183	Op184	Op185	Op186	Op187	Op188	Op189	Op190	Op191	Op192	Op193	Op194	Op195	Op196	Op197	Op198	Op199	Op200	Op201	Op202	Op203	Op204	Op205	Op206	Op207	Op208	Op209	Op210	Op211	Op212	Op213	Op214	Op215	Op216	Op217	Op218	Op219	Op220	Op221	Op222	Op223	Op224	Op225	Op226	Op227	Op228	Op229	Op230	Op231	Op232	Op233	Op234	Op235	Op236	Op237	Op238	Op239	Op240	Op241	Op242	Op243	Op244	Op245	Op246	Op247	Op248	Op249	Op250	Op251	Op252	Op253	Op254	Op255	Op256	Op257	Op258	Op259	Op260	Op261	Op262	Op263	Op264	Op265	Op266	Op267	Op268	Op269	Op270	Op271	Op272	Op273	Op274	Op275	Op276	Op277	Op278	Op279	Op280	Op281	Op282	Op283	Op284	Op285	Op286	Op287	Op288	Op289	Op290	Op291	Op292	Op293	Op294	Op295	Op296	Op297	Op298	Op299	Op300	Op301	Op302	Op303	Op304	Op305	Op306	Op307	Op308	Op309	Op310	Op311	Op312	Op313	Op314	Op315	Op316	Op317	Op318	Op319	Op320	Op321	Op322	Op323	Op324	Op325	Op326	Op327	Op328	Op329	Op330	Op331	Op332	Op333	Op334	Op335	Op336	Op337	Op338	Op339	Op340	Op341	Op342	Op343	Op344	Op345	Op346	Op347	Op348	Op349	Op350	Op351	Op352	Op353	Op354	Op355	Op356	Op357	Op358	Op359	Op360	Op361	Op362	Op363	Op364	Op365	Op366	Op367	Op368	Op369	Op370	Op371	Op372	Op373	Op374	Op375	Op376	Op377	Op378	Op379	Op380	Op381	Op382	Op383	Op384	Op385	Op386	Op387	Op388	Op389	Op390	Op391	Op392	Op393	Op394	Op395	Op396	Op397	Op398	Op399	Op400	Op401	Op402	Op403	Op404	Op405	Op406	Op407	Op408	Op409	Op410	Op411	Op412	Op413	Op414	Op415	Op416	Op417	Op418	Op419	Op420	Op421	Op422	Op423	Op424	Op425	Op426	Op427	Op428	Op429	Op430	Op431	Op432	Op433	Op434	Op435	Op436	Op437	Op438	Op439	Op440	Op441	Op442	Op443	Op444	Op445	Op446	Op447	Op448	Op449	Op450	Op451	Op452	Op453	Op454	Op455	Op456	Op457	Op458	Op459	Op460	Op461	Op462	Op4

11944	057730	003541		.WORD	3541	
11945	057732	177777		.WORD	177777	
11946	057734	172710		.WORD	172710	
11947	057736	070000	TS33D1:	.WORD	70000	
11948	057740	003541		.WORD	3541	
11949	057742	177777		.WORD	177777	
11950	057744	172710		.WORD	172710	
11951	057746	000177	TS33D2:	.WORD	177	
11952	057750	100000		.WORD	100000	
11953	057752	177777		.WORD	177777	
11954	057754	177007		.WORD	177007	
11955	057756	070000	TS33D3:	.WORD	70000	
11956	057760	003541		.WORD	3541	
11957	057762	177777		.WORD	177777	
11958	057764	172710		.WORD	172710	
11959	057766	170000	TS33D4:	.WORD	170000	
11960	057770	003541		.WORD	3541	
11961	057772	177777		.WORD	177777	
11962	057774	172710		.WORD	172710	
11963	057776	000000	TS33D5:	.WORD	0	
11964	060000	000000		.WORD	0	
11965	060002	000000		.WORD	0	
11966	060004	000000		.WORD	0	
11967	060006	000240	FIN33:	NOP		
11968			:			
11969			:			
11970			:			
11971			:			
11972			:			
11973			:	TEST LDD MODE 0, ILLEGAL AC7		
11974			:			
11975			:			
11976			:			
11977	060010		MF SRCMO:			
11978			:			
11979			:			
11980	060010	012704		MOV	#47600,R4	;SETUP FPP STATUS
11981	060014	170104		LD R5	R4	;LOAD FPP STATUS
11982	060016	012702		MOV	#RECEC,R5	;POINT TO RECEIVED FEC MEMORY
11983	060022	172407	1\$:	LDD	R7,AC0	;*TEST INSTRUCTION
11984						;LOAD AC0 FROM ILLEGAL AC7
11985	060024	170201		STPS	R1	;SAVE FPP STATUS
11986	060026	022701		CMP	#147600,R1	;VERIFY FER BIT SET
11987	060032	001401		BEQ	2\$	;BRANCH IF GOOD ERROR CONDITION
11988	060034	104003		ERROR	2\$	;FPP ERROR
11989						;THE FER BIT DIDNT SET
11990	060036	170312	2\$:	STST	(R1)	;SAVE FEC AND FEA
11991	060040	022722		CMP	#1,(R2)	;VERIFY FEC CONTENTS
11992	060044	001401		BEQ	3\$	;BRANCH IF GOOD
11993	060046	104003		ERROR	3\$	;FPP ERROR
11994						;FEC NOT CORRECT ERROR ADDRESS
11995	060050	012722	3\$:	CMP	#13,(R1)	;VERIFY FEA CONTENTS
11996	060054	001401		BEQ	4\$	;BRANCH IF GOOD
11997	060056	104003		ERROR	4\$	;FPP ERROR
11998						;FEA NOT CORRECT ERROR ADDRESS
11999	060060		4\$:			

Address	Hex	Op	Op1	Op2	Op3	Op4	Op5	Op6	Op7	Op8	Op9	Op10	Op11	Op12	Op13	Op14	Op15	Op16	Op17	Op18	Op19	Op20	Op21	Op22	Op23	Op24	Op25	Op26	Op27	Op28	Op29	Op30	Op31	Op32	Op33	Op34	Op35	Op36	Op37	Op38	Op39	Op40	Op41	Op42	Op43	Op44	Op45	Op46	Op47	Op48	Op49	Op50	Op51	Op52	Op53	Op54	Op55	Op56	Op57	Op58	Op59	Op60	Op61	Op62	Op63	Op64	Op65	Op66	Op67	Op68	Op69	Op70	Op71	Op72	Op73	Op74	Op75	Op76	Op77	Op78	Op79	Op80	Op81	Op82	Op83	Op84	Op85	Op86	Op87	Op88	Op89	Op90	Op91	Op92	Op93	Op94	Op95	Op96	Op97	Op98	Op99	Op100	Op101	Op102	Op103	Op104	Op105	Op106	Op107	Op108	Op109	Op110	Op111	Op112	Op113	Op114	Op115	Op116	Op117	Op118	Op119	Op120	Op121	Op122	Op123	Op124	Op125	Op126	Op127	Op128	Op129	Op130	Op131	Op132	Op133	Op134	Op135	Op136	Op137	Op138	Op139	Op140	Op141	Op142	Op143	Op144	Op145	Op146	Op147	Op148	Op149	Op150	Op151	Op152	Op153	Op154	Op155	Op156	Op157	Op158	Op159	Op160	Op161	Op162	Op163	Op164	Op165	Op166	Op167	Op168	Op169	Op170	Op171	Op172	Op173	Op174	Op175	Op176	Op177	Op178	Op179	Op180	Op181	Op182	Op183	Op184	Op185	Op186	Op187	Op188	Op189	Op190	Op191	Op192	Op193	Op194	Op195	Op196	Op197	Op198	Op199	Op200	Op201	Op202	Op203	Op204	Op205	Op206	Op207	Op208	Op209	Op210	Op211	Op212	Op213	Op214	Op215	Op216	Op217	Op218	Op219	Op220	Op221	Op222	Op223	Op224	Op225	Op226	Op227	Op228	Op229	Op230	Op231	Op232	Op233	Op234	Op235	Op236	Op237	Op238	Op239	Op240	Op241	Op242	Op243	Op244	Op245	Op246	Op247	Op248	Op249	Op250	Op251	Op252	Op253	Op254	Op255	Op256	Op257	Op258	Op259	Op260	Op261	Op262	Op263	Op264	Op265	Op266	Op267	Op268	Op269	Op270	Op271	Op272	Op273	Op274	Op275	Op276	Op277	Op278	Op279	Op280	Op281	Op282	Op283	Op284	Op285	Op286	Op287	Op288	Op289	Op290	Op291	Op292	Op293	Op294	Op295	Op296	Op297	Op298	Op299	Op300	Op301	Op302	Op303	Op304	Op305	Op306	Op307	Op308	Op309	Op310	Op311	Op312	Op313	Op314	Op315	Op316	Op317	Op318	Op319	Op320	Op321	Op322	Op323	Op324	Op325	Op326	Op327	Op328	Op329	Op330	Op331	Op332	Op333	Op334	Op335	Op336	Op337	Op338	Op339	Op340	Op341	Op342	Op343	Op344	Op345	Op346	Op347	Op348	Op349	Op350	Op351	Op352	Op353	Op354	Op355	Op356	Op357	Op358	Op359	Op360	Op361	Op362	Op363	Op364	Op365	Op366	Op367	Op368	Op369	Op370	Op371	Op372	Op373	Op374	Op375	Op376	Op377	Op378	Op379	Op380	Op381	Op382	Op383	Op384	Op385	Op386	Op387	Op388	Op389	Op390	Op391	Op392	Op393	Op394	Op395	Op396	Op397	Op398	Op399	Op400	Op401	Op402	Op403	Op404	Op405	Op406	Op407	Op408	Op409	Op410	Op411	Op412	Op413	Op414	Op415	Op416	Op417	Op418	Op419	Op420	Op421	Op422	Op423	Op424	Op425	Op426	Op427	Op428	Op429	Op430	Op431	Op432	Op433	Op434	Op435	Op436	Op437	Op438	Op439	Op440	Op441	Op442	Op443	Op444	Op445	Op446	Op447	Op448	Op449	Op450	Op451	Op452	Op453	Op454	Op455	Op456	Op457	Op458	Op459	Op460	Op461	Op462	Op463	Op464</

12056	060200	172434		LDD	@(R4),AC0		;*TEST INSTRUCTION - MODE 2
12057	060202	170203		STPS	R3		;SAVE TEST FPP STATUS
12058	060204	174031		STL	AC0,@(R1),		;SAVE TEST RESULT IN MODE 3
12059	060205	022703	047740	CMP	#47740,R3		;VERIFY FPP STATUS
12060	060212	001401		BEQ	1\$		;BRANCH IF GOOD
12061	060214	104003		ERROR	+3		;FPP ERROR
12062							;BAD FPP STATUS
12063	060216	022704	003124	1\$: CMP	#TSTLOC+2,R4		;VERIFY AUTO-INCR
12064	060222	001401		BEQ	2\$		;BAD AUTO-DEC ON LDD
12065	060224	104003		ERROR	+3		;FPP ERROR
12066							;BAD AUTO-INC
12067	060226	022701	003126	2\$: CMP	#TSTLOC+4,R1		;TEST STD AUTO-INC
12068	060232	001401		BEQ	3\$		;BRANCH IF GOOD
12069	060234	104003		ERROR	+3		;FPP ERROR
12070							;BAD AUTO-INCR
12071	060236	012704	003204	3\$: MOV	#TAB2,R4		;POINT TO RECEIVED DATA
12072	060242	012701	003107	MOV	#RECDST,R1		;POINT TO RECEIVED DATA
12073	060246	004767	053210	JSR	R7,DATVER		;VERIFY DATA FROM FPP
12074	060252	005767	122602	TST	COUNT		;SEE IF COUNTER=0
12075	060256	001401		BEQ	4\$		;BRANCH IF GOOD COMPARE
12076	060260	104003		ERROR	+3		;FPP ERROR
12077							;BAD DATA FROM FPP
12078	060262			4\$:			
12079							
12080				:			
12081				:			
12082				:			
12083				:			
12084				:			
12085				:			
12086				:			
12087				:			
12088				:			
12089				:			
12090	060262			MI DDM4:			
12091				:			
12092				:			
12093	060262	012701	003106	:			
12094	060266	012704	003220	MOV	#RECDST+4,R1		;POINT TO RECEIVED DATA LOCATION
12095	060272	012705	003254	MOV	#TAB3+4,R4		;POINT TO GOOD DATA
12096	060276	170127	000200	MOV	#TAB2,R5		;CLEAR OUT AC0
12097	060302	172415		LDFPS	#200		;SET TO DOUBLE
12098	060304	012702	047550	LDD	(R5),AC0		;AC0=0
12099	060310	170102		MOV	#47550,R2		;LOAD GOOD STATUS FLOATING
12100	060312	172444		LDFPS	R2		;LOAD FPP STATUS - DOUBLE, ID
12101	060314	170203		LDF	(R4),AC0		;*TEST INSTRUCTION - MODE 4
12102	060316	012702	047750	STPS	R4		;SAVE TEST FPP STATUS
12103	060322	170102		MOV	#47750,R2		;SET TO DOUBLE MODE
12104	060324	174041		LDFPS	R2		;SET FPP TO DOUBLE
12105	060326	022703	047540	STD	AC0,(R1)		;SAVE TEST RESULT
12106	060332	001401		CMP	#47540,R3		;VERIFY FPP STATUS
12107	060334	104003		BEQ	1\$		;BRANCH IF GOOD
12108				ERROR	+3		;FPP ERROR
12109	060336	022704	003214	1\$: CMP	#TAB3,R4		;VERIFY AUTO-DEC
12110	060342	001401		BEQ	2\$		;BRANCH IF GOOD
12111	060344	104003		ERROR	+3		;FPP ERROR

N1

CORDBO KDJ11-B CLUSTER MACY11 30(1046) 05 APR 84 16:45 PAGE 221
 CORDAB.P11 05-APR-84 16:45 FLOATING POINT TESTS

SEQ 0220

```

12112
12113 060346 012704 003214      2$:  MOV    #TAB3,R4      ;BAD AUTO INCR
12114 060352 004767 053104      JSR    R7,DATVER      ;POINT TO RECEIVED DATA
12115 060356 005767 122476      IST    COUNT      ;VERIFY DATA FROM FPP
12116 060362 001401              BEQ    3$      ;SEE IF COUNTER=0
12117 060364 104003              ERROR   +3      ;BRANCH IF GOOD COMPARE
12118
12119 060366      3$:          ;FPP ERROR
12120
12121      ;          ;BAD DATA FROM FPP
12122
12123
12124
12125
12126
12127      ;-----
12128      ;TEST LDD MODE 5
12129
12130
12131 060366      ;
12132      ;
12133      ;
12134 060366 012701 003102      ;
12135 060372 012704 003124      ;      MOV    #RECDST,R1      ;POINT TO RECEIVED DATA LOCATION
12136 060376 012737 003174      MOV    #ISTLOC+2,R4      ;POINT TO GOOD DATA
12137 060404 012702 047750      MOV    #TAB1,@ISTLOC      ;SET UP MODE 5 POINTER TO DATA
12138 060410 170102              MOV    #47750,R2      ;LOAD GOOD STATUS
12139 060412 172454              LDFPS  R2      ;LOAD FPP STATUS - DOUBLE,10
12140 060414 170203              LDD    @-(R4),AC0      ;*TEST INSTRUCTION MODE 5
12141 060416 174011              STEPS  R3      ;SAVE TEST FPP STATUS
12142 060420 020203              STD    AC0,(R1)      ;SAVE TEST RESULT
12143 060422 001401              CMP    R2,R3      ;VERIFY FPP STATUS
12144 060424 104003              BEQ    1$      ;BRANCH IF GOOD
12145
12146 060426 022704 003122      1$:  CMP    #ISTLOC,R4      ;FPP ERROR
12147 060432 001401              BEQ    2$      ;BAD FPP STATUS
12148 060434 104003              ERROR   +3      ;VERIFY AUTO-DEC
12149
12150 060436 012704 003174      2$:  MOV    #TAB1,R4      ;BRANCH IF GOOD
12151 060442 004767 053014      JSR    R7,DATVER      ;FPP ERROR
12152 060446 005767 122406      IST    COUNT      ;BAD AUTO DEC
12153 060452 001401              BEQ    3$      ;POINT TO EXPECTED DATA
12154 060454 104003              ERROR   +3      ;VERIFY DATA FROM FPP
12155
12156 060456      3$:          ;SEE IF COUNTER=0
12157
12158
12159
12160
12161      ;-----
12162      ;TEST LDD MODE 6
12163
12164
12165
12166 060456      ;
12167      ;

```

```

12168
12169 060456 012701 003302      ;      MOV      @RECDST+200,R1      ;POINT TO RECEIVED DATA LOCATION
12170 060462 012704 003004      MOV      @TAB2+200,R4      ;SETUP R4 FOR MODE 6
12171 060466 012702 047750      MOV      @47750,R2      ;LOAD GOOD STATUS
12172 060472 170102              LDFPS    R2      ;LOAD FPP STATUS - DOUBLE, ID
12173 060474 172464 000200      LDD      200(R4),ACO      ;LDD MODE 6
12174 060500 170203              STFPS    R3      ;SAVE TEST FPP STATUS
12175 060502 174061 177600      STD      ACO, -200(R1)      ;SAVE TEST RESULT
12176 060506 022703 047740      CMP      @47740,R3      ;VERIFY FPP STATUS
12177 060512 001401              BEQ      1$      ;BRANCH IF GOOD
12178 060514 104003              ERROR    +3      ;FPP ERROR
12179                                ;BAD FPP STATUS
12180 060516 162701 000200      1$:      SUB      @200,R1      ;R1=RECDST
12181 060522 062704 000200      2$:      ADD      @200,R4      ;POINT TO EXPECTED DATA
12182 060526 004767 052730      JSR      R7,DATVER      ;VERIFY DATA FROM FPP
12183 060532 005767 122322      TST      COUNT      ;SEE IF COUNTER=0
12184 060536 001401              BEQ      3$      ;BRANCH IF GOOD COMPARE
12185 060540 104003              ERROR    +3      ;FPP ERROR
12186                                ;BAD DATA FROM FPP
12187 060542
12188
12189
12190
12191
12192      ;TEST LDD MODE 7
12193
12194
12195
12196
12197 060542      MLDDM7:
12198
12199
12200 060542 012701 003102      ;      MOV      @RECDST,R1      ;POINT TO RECEIVED DATA LOCATION
12201 060546 005004              CLR      R4      ;R4=0
12202 060550 012727 003174 003122      MOV      @TAB1,@TSTLOC      ;POINTER FOR MODE 7 GOOD DATA
12203 060556 012702 047750      MOV      @47750,R2      ;LOAD GOOD STATUS
12204 060562 170102              LDFPS    R2      ;LOAD FPP STATUS - DOUBLE, ID
12205 060564 172474 003122      LDD      @TSTLOC(R4),ACO      ;*TEST INSTRUCTION - MODE 7
12206 060570 170203              STFPS    R3      ;SAVE TEST FPP STATUS
12207 060572 174011              STD      ACO,(R1)      ;SAVE TEST RESULT
12208 060574 020203              CMP      R2,R3      ;VERIFY FPP STATUS
12209 060576 001401              BEQ      1$      ;BRANCH IF GOOD
12210 060600 104003              ERROR    +3      ;FPP ERROR
12211                                ;BAD FPP STATUS
12212 060602 005704      1$:      TST      R4      ;VERIFY CONTENTS OF R4
12213 060604 001401              BEQ      1$      ;BRANCH IF GOOD
12214 060606 104003              ERROR    +3      ;FPP ERROR
12215                                ;BAD R4
12216 060610 012704 003174      2$:      MOV      @TAB1,R4      ;POINT TO RECEIVED DATA
12217 060614 004767 052642      JSR      R7,DATVER      ;VERIFY DATA FROM FPP
12218 060620 005767 122234      TST      COUNT      ;SEE IF COUNTER=0
12219 060624 001401              BEQ      3$      ;BRANCH IF GOOD COMPARE
12220 060626 104003              ERROR    +3      ;FPP ERROR
12221                                ;BAD DATA FROM FPP
12222 060630
12223      3$:

```

```

12234
12235
12236
12237
12238
12239
12230
12231
12232 060630
12233
12234
12235 060630 012701 003102
12236 060634 012704 003234
12237 060640 012702 047750
12238 060644 005005
12239 060646 170102
12240 060650 172427 043243
12241 060654 005205
12242 060656 005205
12243 060660 005205
12244 060662 022705 000003
12245 060666 001401
12246 060670 104003
12247
12248 060672 170203
12249 060674 174011
12250 060676 022703 047740
12251 060702 001401
12252 060704 104003
12253
12254 060706 004767 052550
12255 060712 005767 122142
12256 060716 001401
12257 060720 104003
12258
12259 060722
12260
12261
12262
12263
12264
12265
12266
12267
12268 060722
12269
12270
12271 060722 012704 003254
12272 060726 005067 122154
12273 060732 005067 122152
12274 060736 012702 040000
12275 060742 170102
12276 060744 172414
12277 060746 172014
12278 060750 170207
12279 060752 022703 040004

;
;
;
; TEST LDD MODE 27 - ONLY 16 BITS ARE LOADED OR STORED
;
;
;
; MLD27:
;
; MOV 0RECDST,R1 ;POINT TO RECEIVED DATA LOCATION
; MOV 0TAB5,R4 ;POINT TO GOOD DATA
; MOV 047750,R2 ;LOAD GOOD STATUS
; CLR R5 ;R5=0
; LDEPS R2 ;LOAD FPP STATUS DOUBLE ID
; LDD 05205,AC0 ;*TEST INSTRUCTION - MODE 27
; INC R5
; INC R5
; INC R5
; CMP 03,R5 ;TEST PROPER PC PATH
; BEQ 1$ ;VERIFY ONLY 3 PC INCREMENT
; ERROR 1$ ;BRANCH IF PROPER PC ACTION
; FPP ERROR
; BAD MODE 27 LOAD
; 1$: STEPS R3 ;SAVE TEST FPP STATUS
; STD AC0,(R1) ;SAVE TEST RESULT
; CMP 047740,R3 ;VERIFY FPP STATUS
; BEQ 2$ ;BRANCH IF GOOD
; ERROR 1$ ;FPP ERROR
; BAD FPP STATUS
; 2$: JSR R7,DATVER ;VERIFY DATA FROM FPP
; IST COUNT ;SEE IF COUNTER=0
; BEQ 3$ ;BRANCH IF GOOD COMPARE
; ERROR 1$ ;FPP ERROR
; BAD DATA FROM FPP
; 3$:
;
;
;
; TEST ADDF, ADDD, SUBF, SUBD AC0=0 FSRC=0;
;
;
; MNNRM1:
;
; MOV 0TAB6,R4 ;POINT TO FSRC TEST DATA
; CLR RECDST,4 ;CLEAR OUT RECEIVED DATA TABLE
; CLR RECDST,6
; MOV 040000,R3
; LDEPS R2
; LDE (R4),AC0 ;SET UP GOOD STATUS
; ADDF (R4),AC0 ;LOAD FPP STATUS, FLOATING
; STEPS R3 ;LOAD AC0 WITH 0
; CMP 040004,R3 ;SAVE STATUS
; ;VERIFY STATUS

```

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

PC	Op	Op1	Op2	Op3	Op4	Op5	Op6	Op7	Op8	Op9	Op10	Op11	Op12	Op13	Op14	Op15	Op16	Op17	Op18	Op19	Op20	Op21	Op22	Op23	Op24	Op25	Op26	Op27	Op28	Op29	Op30	Op31	Op32	Op33	Op34	Op35	Op36	Op37	Op38	Op39	Op40	Op41	Op42	Op43	Op44	Op45	Op46	Op47	Op48	Op49	Op50	Op51	Op52	Op53	Op54	Op55	Op56	Op57	Op58	Op59	Op60	Op61	Op62	Op63	Op64	Op65	Op66	Op67	Op68	Op69	Op70	Op71	Op72	Op73	Op74	Op75	Op76	Op77	Op78	Op79	Op80	Op81	Op82	Op83	Op84	Op85	Op86	Op87	Op88	Op89	Op90	Op91	Op92	Op93	Op94	Op95	Op96	Op97	Op98	Op99	Op100	Op101	Op102	Op103	Op104	Op105	Op106	Op107	Op108	Op109	Op110	Op111	Op112	Op113	Op114	Op115	Op116	Op117	Op118	Op119	Op120	Op121	Op122	Op123	Op124	Op125	Op126	Op127	Op128	Op129	Op130	Op131	Op132	Op133	Op134	Op135	Op136	Op137	Op138	Op139	Op140	Op141	Op142	Op143	Op144	Op145	Op146	Op147	Op148	Op149	Op150	Op151	Op152	Op153	Op154	Op155	Op156	Op157	Op158	Op159	Op160	Op161	Op162	Op163	Op164	Op165	Op166	Op167	Op168	Op169	Op170	Op171	Op172	Op173	Op174	Op175	Op176	Op177	Op178	Op179	Op180	Op181	Op182	Op183	Op184	Op185	Op186	Op187	Op188	Op189	Op190	Op191	Op192	Op193	Op194	Op195	Op196	Op197	Op198	Op199	Op200	Op201	Op202	Op203	Op204	Op205	Op206	Op207	Op208	Op209	Op210	Op211	Op212	Op213	Op214	Op215	Op216	Op217	Op218	Op219	Op220	Op221	Op222	Op223	Op224	Op225	Op226	Op227	Op228	Op229	Op230	Op231	Op232	Op233	Op234	Op235	Op236	Op237	Op238	Op239	Op240	Op241	Op242	Op243	Op244	Op245	Op246	Op247	Op248	Op249	Op250	Op251	Op252	Op253	Op254	Op255	Op256	Op257	Op258	Op259	Op260	Op261	Op262	Op263	Op264	Op265	Op266	Op267	Op268	Op269	Op270	Op271	Op272	Op273	Op274	Op275	Op276	Op277	Op278	Op279	Op280	Op281	Op282	Op283	Op284	Op285	Op286	Op287	Op288	Op289	Op290	Op291	Op292	Op293	Op294	Op295	Op296	Op297	Op298	Op299	Op300	Op301	Op302	Op303	Op304	Op305	Op306	Op307	Op308	Op309	Op310	Op311	Op312	Op313	Op314	Op315	Op316	Op317	Op318	Op319	Op320	Op321	Op322	Op323	Op324	Op325	Op326	Op327	Op328	Op329	Op330	Op331	Op332	Op333	Op334	Op335	Op336	Op337	Op338	Op339	Op340	Op341	Op342	Op343	Op344	Op345	Op346	Op347	Op348	Op349	Op350	Op351	Op352	Op353	Op354	Op355	Op356	Op357	Op358	Op359	Op360	Op361	Op362	Op363	Op364	Op365	Op366	Op367	Op368	Op369	Op370	Op371	Op372	Op373	Op374	Op375	Op376	Op377	Op378	Op379	Op380	Op381	Op382	Op383	Op384	Op385	Op386	Op387	Op388	Op389	Op390	Op391	Op392	Op393	Op394	Op395	Op396	Op397	Op398	Op399	Op400	Op401	Op402	Op403	Op404	Op405	Op406	Op407	Op408	Op409	Op410	Op411	Op412	Op413	Op414	Op415	Op416	Op417	Op418	Op419	Op420	Op421	Op422	Op423	Op424	Op425	Op426	Op427	Op428	Op429	Op430	Op431	Op432	Op433	Op434	Op435	Op436	Op437	Op438	Op439	Op440	Op441	Op442	Op443	Op444	Op445	Op446	Op447	Op448	Op449	Op450	Op451	Op452	Op453	Op454	Op455	Op456	Op457	Op458	Op459	Op460	Op461	Op462	Op463	Op464	Op4

```

12392
12393 061274 MNNRM3:
12394
12395
12396 061274 012701 003102      MOV    #RECDST,R1      ;POINT TO RECEIVED DATA TABLE
12397 061300 012705 003254      MOV    #TAB6,R5      ;POINT TO ACO DATA TABLE
12398 061304 012704 003174      MOV    #TAB1,R4      ;POINT TO FSRC DATA
12399 061310 012702 000200      MOV    #200,R2      ;SETUP FPP STATUS
12400 061314 170102              LDGPS  R2      ;LOAD FPS
12401 061316 172415              LDD     (R5),ACO      ;LOAD ACO
12402 061320 172014              ADDD   (R4),ACO      ;*TEST INSTRUCTION
12403 061322 170203              STEPS  R3      ;SAVE STATUS
12404 061324 174011              STD     ACO,(R1)      ;SAVE ACO
12405 061326 022703 000210      CMP     #210,R3      ;VERIFY NEGATIVE RESULT
12406 061332 001401              BEQ     1$      ;BRANCH IF GOOD
12407 061334 104003              ERROR   +3      ;FPP ERROR
12408                                ;BAD FPS
12409 061336 004767 052100      1$:    JSR     R7,DATVER      ;VERIFY ACO
12410 061342 005767 121512      TST     COUNT      ;CHECK RESULT
12411 061346 001401              BEQ     2$      ;BRANCH IF GOOD
12412 061350 104003              ERROR   +3      ;FPP ERROR
12413                                ;BAD ACO
12414 061352 170127 000200      2$:    LDGPS  #200      ;SET STATUS TO DOUBLE NODE
12415 061356 172415              LDF     (R5),ACO      ;LOAD ACO WITH A VALUE
12416 061360 173014              SUBD   (R4),ACO      ;*TEST INSTRUCTION
12417 061362 170203              STEPS  R3      ;SAVE FPP STATUS
12418 061364 174011              STF     ACO,(R1)      ;SAVE ACO
12419 061366 022703 000200      CMP     #200,R3      ;VERIFY RESULT
12420 061372 001401              BEQ     3$      ;BRANCH IF GOOD
12421 061374 104003              ERROR   +3      ;FPP ERROR
12422                                ;BAD SUBD
12423 061376 012704 003444      3$:    MOV     #TAB18,R4      ;POINT TO EXPECTED DATA
12424 061402 004767 052054      JSR     R7,DATVER      ;VERIFY ACO
12425 061406 005767 121446      TST     COUNT      ;
12426 061412 001401              BEQ     4$      ;BRANCH IF GOOD ACO
12427 061414 104003              ERROR   +3      ;FPP ERROR
12428                                ;BAD ACO
12429 061416      4$:
12430
12431
12432
12433
12434
12435      ;TEST ADDE, SUBD - EXP(ACO) = EXP(FSRC)
12436
12437
12438
12439
12440 061416 MNNRM4:
12441
12442
12443 061416 012702 003240      MOV     #3240,R1      ;SET FID,FD,FT
12444 061422 170102              LDGPS  R1
12445 061424 012704 003274      MOV     #TAB7,R4      ;SET FSRC
12446 061430 012705 003304      MOV     #TAB8,R5      ;SETUP ACO
12447 061434 012701 003102      MOV     #RECDST,R1      ;POINT TO RECEIVED DATA

```


12448	061440	172415		LDD	(R5),ACO	;LOAD ACO
12449	061442	172014		ADDD	(R4),ACO	;+TEST INSTRUCTION
12450	061444	174011		STD	ACO,(R1)	;SAVE TEST RESULT
12451	061446	012704	003314	MOV	#TAB9,R4	;POINT TO EXPECTED DATA
12452	061452	004767	052004	JSR	R7,DATVER	;VERIFY ACO DATA
12453	061456	005767	121376	TST	COUNT	
12454	061462	001401		BEQ	1\$	;BRANCH IF GOOD
12455	061464	104003		ERROR	+3	;FPP ERROR
12456						;BAD ADD
12457	061466	012704	003304	1\$: MOV	#TAB8,R4	
12458	061472	012703	003304	MOV	#TAB8,R3	;SETUP SAME ACO
12459	061476	012702	003200	MOV	#3200,R2	;ROUND MODE
12460	061502	170102		LDFPS	R2	
12461	061504	172415		LDD	(R3),ACO	;LOAD ACO
12462	061506	061400		ADD	(R4),ACO	;+TEST INSTRUCTION
12463	061510	174011		STD	ACO,(R1)	;SAVE DATA
12464	061512	004767	051744	JSR	R7,DATVER	;VERIFY ACO
12465	061516	005767	121336	TST	COUNT	
12466	061522	001401		BEQ	2\$	;BRANCH IF GOOD
12467	061524	104003		ERROR	+3	;FPP ERROR
12468						;BAD ROUND RESULT
12469	061526			2\$:		
12470				:		
12471				:		
12472				:		
12473				:		
12474				:		
12475				:		
12476				:		
12477				:		
12478				:		
12479				:		
12480	061526			MXDF1:		
12481				:		
12482				:		
12483	061526	012702	003200	:		
12484	061532	170102		MOV	#3200,R2	;R2=FPP STATUS
12485	061534	012704	003334	LDFPS	R2	;LOAD FPS STATUS
12486	061540	012701	003102	MOV	#TAB11,R4	;POINT TO FSRC DATA
12487	061544	012705	003324	MOV	#RECDST,R1	;POINT TO ACO RESULT
12488	061550	172415		MOV	#TAB10,R5	;POINT TO ACO DATA
12489	061552	172014		LDD	(R5),ACO	;LOAD ACO DATA
12490	061554	170203		ADDD	(R4),ACO	;+TEST INSTRUCTIONS
12491	061556	174011		STEPS	R3	;SAVE FPP STATUS
12492	061560	022703	003200	STD	ACO,(R1)	;SAVE ACO DATA
12493	061564	001401		CMPL	#3200,R2	;VERIFY FPP STATUS
12494	061566	104003		BEQ	1\$	;BRANCH IF GOOD
12495				ERROR	+3	;FPP ERROR
12496	061570	012704	003344	1\$: MOV	#TAB11A,R4	;POINT TO EXPECTED DATA
12497	061574	004767	051662	JSR	R7,DATVER	;VERIFY CONTENTS OF ACO
12498	061500	005767	121254	TST	COUNT	
12499	061604	001401		BEQ	2\$	;BRANCH IF GOOD ACO
12500	061606	104003		ERROR	+3	;FPP ERROR
12501						;BAD ACO, SHOULD = FSRC
12502	061610	012704	003354	2\$: MOV	#TAB11,R4	;POINT TO FSRC DATA
12503	061614	172415		LDD	(R5),ACO	;ACO

H2

COKDABO KDJ11 B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 228
 COKDAB.P11 05-APR-84 16:45 FLOATING POINT TESTS

SEQ 0227

12504	061616	172014		ADD	(R4),ACO	;*TEST INSTRUCTION
12505	061620	012704	003374	MOV	#TAB13B,R4	;POINT TO EXPECTED RESULT
12506	061624	174011		STD	ACO,(R1)	;SAVE ACO DATA INTO RECDAT
12507	061626	004767	051630	JSR	R7,DATVER	;VERIFY DATA
12508	061632	005767	121222	TST	COUNT	
12509	061636	001401		BEQ	3\$	;BRANCH IF GOOD DATA
12510	061640	104003		ERROR	+3	;FPP ERROR
12511						;BAD ACO DATA
12512	061642	012702	003000	3\$: MOV	#3000,R2	;GET FPP STATUS DATA
12513	061646	012704	003404	MOV	#TAB14,R4	;POINT TO FSR DATA
12514	061652	012705	003414	MOV	#TAB15,R5	;POINT TO ACO DATA
12515	061656	172415		LDD	(R5),ACO	;LOAD ACO
12516	061660	170102		LDFPS	R2	;FPP STATUS = FLOAT, INTERRUPTS ENABLE
12517	061662	172014		ADDF	(R4),ACO	;*TEST INSTRUCTION
12518	061664	170127	000200	LDFPS	#200	;RESET TO DOUBLE
12519	061670	174011		STD	ACO,(R1)	;RECDST=ACO
12520	061672	012704	003324	MOV	#TAB10,R4	;POINT TO GOOD DATA
12521	061676	004767	051560	JSR	R7,DATVER	;VERIFY CONTENTS OF ACO
12522	061702	005767	121152	TST	COUNT	
12523	061706	001401		BEQ	4\$	;BRANCH IF GOOD
12524	061710	104003		ERROR	+3	;FPP ERROR
12525						;BAD FLOATING ADD
12526	061712	012705	003424	4\$: MOV	#TAB16,R5	;POINT TO ACO DATA
12527	061716	170102		LDFPS	R2	;FPP STATUS = FLOAT
12528	061720	172415		LDF	(R5),ACO	;LOAD ACO
12529	061722	172014		ADDF	(R4),ACO	;*TEST INSTRUCTION
12530	061724	174011		STD	ACO,(R1)	;SAVE ACO DATA
12531	061726	012704	003434	MOV	#TAB17,R4	;POINT TO GOOD DATA
12532	061732	004767	051574	JSR	R7,DATVER	
12533	061736	005767	121116	TST	COUNT	
12534	061742	001401		BEQ	5\$	;BRANCH IF GOOD
12535	061744	104003		ERROR	+3	;FPP ERROR
12536						;BAD FLOATING ADD
12537	061746			5\$:		
12538						
12539						
12540						
12541						
12542						
12543						
12544						
12545						
12546						
12547						
12548						
12549	061746			MNGUP:		
12550						
12551						
12552	061746	012702	003200	MOV	#3200,R2	;LOAD FPS VALUE
12553	061752	170102		LDFPS	R2	
12554	061754	012704	003454	MOV	#TAB21,R4	;DATA ADDRESS FOR ACO AND FSR
12555	061760	172414		LDD	(R4),ACO	;ACO=100,000 0 0 0
12556	061762	172014		ADDF	(R4),ACO	;*TEST INSTRUCTION
12557	061764	170203		STEPS	R3	;SAVE STATUS
12558	061766	012701	003102	MOV	#RECDST,R1	;POINT TO RECEIVED DATA TABLE
12559	061772	174011		STD	ACO,(R1)	;SAVE ACO DATA

12560	061774	022703	003210		CMP	#3210,R3		;VERIFY STATUS
12561	062000	001401			BEQ	1\$		;BRANCH IF GOOD
12562	062002	104003			ERROR	+3		;FPP ERROR
12563								
12564	062004	012704	003464	1\$:	MOV	#TAB22,R4		;POINT TO EXPECTED DATA
12565	062010	004767	051446		JSR	R7,DATVER		
12566	062014	005767	121040		TST	COUNT		;VERIFY DATA
12567	062020	001401			BEQ	2\$		;BRANCH IF GOOD
12568	062022	104003			ERROR	+3		;FPP ERROR
12569								
12570								
12571	062024	012704	003454	2\$:		!-FSRC! - !ACO!		
12572	062030	012701	003474		MOV	#TAB21,R4		;POINT TO FSRC DATA
12573	062034	012737	062052	000244	MOV	#TAB23,R1		;POINT TO ACO DATA
12574	062042	172411			MOV	#101\$,#0#FVEC		;SETUP FP VECTOR
12575	062044	172014			LDD	(R1),ACO		;LOAD ACO
12576	062046	170000			ADD	(R4),ACO		;+TEST INSTRUCTION
12577	062050	104003		100\$:	CFCC			;COPY FPP CC
12578					ERROR	+3		;FPP ERROR
12579	062052	170203						;GO TO ERROR
12580	062054	012701	003102	101\$:	STEPS	R3		;SAVE FPP STATUS
12581	062060	174011			MOV	#RECDST,R1		;POINT TO RECEIVED DATA TABLE
12582	062062	022703	103200		STD	ACO,(R1)		;SAVE ACO DATA
12583	062066	001401			CMP	#103200,R3		;VERIFY STATUS
12584	062070	104003			BEQ	3\$		;BRANCH IF GOOD
12585					ERROR	+3		;FPP ERROR
12586	062072	012605		3\$:	MOV	(SP)+,R5		;BAD STATUS
12587	062074	020527	062046		CMP	R5,#100\$		;GET ERROR PC
12588	062100	001401			BEQ	102\$		;VERIFY ERROR ADDRESS ON STACK
12589	062102	104003			ERROR	+3		;BRANCH IF GOOD
12590								;FPP ERROR
12591	062104	005726		102\$:	TST	(SP)+		;BAD ERROR RETURN ON STACK
12592	062106	012704	003504		MOV	#TAB24,R4		;RESTORE STACK
12593	062112	004767	051344		JSR	R7,DATVER		;POINT TO EXPECTED DATA TABLE
12594	062116	005767	120736		TST	COUNT		;VERIFY DATA
12595	062122	001401			BEQ	4\$		;BRANCH IF GOOD
12596	062124	104003			ERROR	+3		;FPP ERROR
12597								;BAD ACO DATA
12598								
12599	062126	012704	003474	4\$:		!-ACO! - !FSRC!		
12600	062132	012701	003454		MOV	#TAB23,R4		;POINT TO FSRC DATA
12601	062136	012737	062162	000244	MOV	#TAB21,R1		;POINT TO ACO DATA
12602	062144	012702	003200		MOV	#104\$,#0#FVEC		;SETUP FP VECTOR
12603	062150	170102			MOV	#3200,R2		;LOAD FPS VALUE
12604	062152	172411			LDFPS	R2		
12605	062154	172014			LDD	(R1),ACO		;LOAD ACO DATA
12606	062156	170000			ADD	(R4),ACO		;+TEST INSTRUCTION
12607	062160	104003		103\$:	CFCC			;COPY FPP CC
12608					ERROR	+3		;FPP ERROR
12609	062162	170203						;GO TO ERROR
12610	062164	012701	003102	104\$:	STEPS	R3		;SAVE FPS
12611	062170	174011			MOV	#RECDST,R1		;SAVE ACO
12612	062172	022703	103200		STD	ACO,(R1)		
12613	062174	001401			CMP	#103200,R3		;VERIFY STATUS
12614	062176	104003			BEQ	5\$		;BRANCH IF GOOD
12615	062178	104003			ERROR	+3		;FPP ERROR
								;BAD FPS STATUS

Address	Hex	Dec	Label	Op	OpC	OpD	OpE	OpF	OpG	OpH	OpI	OpJ	OpK	OpL	OpM	OpN	OpO	OpP	OpQ	OpR	OpS	OpT	OpU	OpV	OpW	OpX	OpY	OpZ	OpAA	OpAB	OpAC	OpAD	OpAE	OpAF	OpAG	OpAH	OpAI	OpAJ	OpAK	OpAL	OpAM	OpAN	OpAO	OpAP	OpAQ	OpAR	OpAS	OpAT	OpAU	OpAV	OpAW	OpAX	OpAY	OpAZ	OpBA	OpBB	OpBC	OpBD	OpBE	OpBF	OpBG	OpBH	OpBI	OpBJ	OpBK	OpBL	OpBM	OpBN	OpBO	OpBP	OpBQ	OpBR	OpBS	OpBT	OpBU	OpBV	OpBW	OpBX	OpBY	OpBZ	OpCA	OpCB	OpCC	OpCD	OpCE	OpCF	OpCG	OpCH	OpCI	OpCJ	OpCK	OpCL	OpCM	OpCN	OpCO	OpCP	OpCQ	OpCR	OpCS	OpCT	OpCU	OpCV	OpCW	OpCX	OpCY	OpCZ	OpDA	OpDB	OpDC	OpDD	OpDE	OpDF	OpDG	OpDH	OpDI	OpDJ	OpDK	OpDL	OpDM	OpDN	OpDO	OpDP	OpDQ	OpDR	OpDS	OpDT	OpDU	OpDV	OpDW	OpDX	OpDY	OpDZ	OpEA	OpEB	OpEC	OpED	OpEE	OpEF	OpEG	OpEH	OpEI	OpEJ	OpEK	OpEL	OpEM	OpEN	OpEO	OpEP	OpEQ	OpER	OpES	OpET	OpEU	OpEV	OpEW	OpEX	OpEY	OpEZ	OpFA	OpFB	OpFC	OpFD	OpFE	OpFF	OpFG	OpFH	OpFI	OpFJ	OpFK	OpFL	OpFM	OpFN	OpFO	OpFP	OpFQ	OpFR	OpFS	OpFT	OpFU	OpFV	OpFW	OpFX	OpFY	OpFZ	OpGA	OpGB	OpGC	OpGD	OpGE	OpGF	OpGG	OpGH	OpGI	OpGJ	OpGK	OpGL	OpGM	OpGN	OpGO	OpGP	OpGQ	OpGR	OpGS	OpGT	OpGU	OpGV	OpGW	OpGX	OpGY	OpGZ	OpHA	OpHB	OpHC	OpHD	OpHE	OpHF	OpHG	OpHH	OpHI	OpHJ	OpHK	OpHL	OpHM	OpHN	OpHO	OpHP	OpHQ	OpHR	OpHS	OpHT	OpHU	OpHV	OpHW	OpHX	OpHY	OpHZ	OpIA	OpIB	OpIC	OpID	OpIE	OpIF	OpIG	OpIH	OpII	OpIJ	OpIK	OpIL	OpIM	OpIN	OpIO	OpIP	OpIQ	OpIR	OpIS	OpIT	OpIU	OpIV	OpIW	OpIX	OpIY	OpIZ	OpJA	OpJB	OpJC	OpJD	OpJE	OpJF	OpJG	OpJH	OpJI	OpJJ	OpJK	OpJL	OpJM	OpJN	OpJO	OpJP	OpJQ	OpJR	OpJS	OpJT	OpJU	OpJV	OpJW	OpJX	OpJY	OpJZ	OpKA	OpKB	OpKC	OpKD	OpKE	OpKF	OpKG	OpKH	OpKI	OpKJ	OpKK	OpKL	OpKM	OpKN	OpKO	OpKP	OpKQ	OpKR	OpKS	OpKT	OpKU	OpKV	OpKW	OpKX	OpKY	OpKZ	OpLA	OpLB	OpLC	OpLD	OpLE	OpLF	OpLG	OpLH	OpLI	OpLJ	OpLK	OpLL	OpLM	OpLN	OpLO	OpLP	OpLQ	OpLR	OpLS	OpLT	OpLU	OpLV	OpLW	OpLX	OpLY	OpLZ	OpMA	OpMB	OpMC	OpMD	OpME	OpMF	OpMG	OpMH	OpMI	OpMJ	OpMK	OpML	OpMM	OpMN	OpMO	OpMP	OpMQ	OpMR	OpMS	OpMT	OpMU	OpMV	OpMW	OpMX	OpMY	OpMZ	OpNA	OpNB	OpNC	OpND	OpNE	OpNF	OpNG	OpNH	OpNI	OpNJ	OpNK	OpNL	OpNM	OpNN	OpNO	OpNP	OpNQ	OpNR	OpNS	OpNT	OpNU	OpNV	OpNW	OpNX	OpNY	OpNZ	OpOA	OpOB	OpOC	OpOD	OpOE	OpOF	OpOG	OpOH	OpOI	OpOJ	OpOK	OpOL	OpOM	OpON	OpOO	OpOP	OpOQ	OpOR	OpOS	OpOT	OpOU	OpOV	OpOW	OpOX	OpOY	OpOZ	OpPA	OpPB	OpPC	OpPD	OpPE	OpPF	OpPG	OpPH	OpPI	OpPJ	OpPK	OpPL	OpPM	OpPN	OpPO	OpPP	OpPQ	OpPR	OpPS	OpPT	OpPU	OpPV	OpPW	OpPX	OpPY	OpPZ	OpQA	OpQB	OpQC	OpQD	OpQE	OpQF	OpQG	OpQH	OpQI	OpQJ	OpQK	OpQL	OpQM	OpQN	OpQO	OpQP	OpQQ	OpQR	OpQS	OpQT	OpQU	OpQV	OpQW	OpQX	OpQY	OpQZ	OpRA	OpRB	OpRC	OpRD	OpRE	OpRF	OpRG	OpRH	OpRI	OpRJ	OpRK	OpRL	OpRM	OpRN	OpRO	OpRP	OpRQ	OpRR	OpRS	OpRT	OpRU	OpRV	OpRW	OpRX	OpRY	OpRZ	OpSA	OpSB	OpSC	OpSD	OpSE	OpSF	OpSG

12672	062416	170203		STEPS	R3		;SAVE STATUS
12673	062420	012701	003102	MOV	#RECDST,R1		;POINT TO RECEIVED DATA TABLE
12674	062424	174011		STD	ACO,(R1)		;SAVE ACO
12675	062426	020327	003200	CMP	R3,#3200		;VERIFY STATUS
12676	062432	001401		BEQ	11\$		;BRANCH IF GOOD
12677	062434	104003		ERROR	+3		;FPP ERROR
12678							;BAD FPS
12679	062436	012704	003564	11\$:	MOV	#TAB29A,R4	;POINT TO EXPECTED DATA
12680	062442	004767	051014		JSR	R7,DATVER	;VERIFY DATA
12681	062446	005767	120406		IST	COUNT	
12682	062452	001401			BEQ	12\$	;BRANCH IF GOOD
12683	062454	104003			ERROR	+3	;FPP ERROR
12684							
12685	062456			12\$:			
12686							
12687							
12688							
12689							
12690							
12691							
12692							
12693							
12694							
12695							
12696							
12697	062456			MSB:			
12698							
12699							
12700	062456	012702	003200		MOV	#3200,R2	;LOAD FPS DATA
12701	062462	170102			LD FPS	R2	;LOAD FPS
12702	062464	012704	003454		MOV	#TAB21,R4	;POINT TO FSRC DATA
12703	062470	012701	003102		MOV	#RECDST,R1	;POINT TO ACO RECEIVED DATA TABLE
12704	062474	172414			LDD	(R4),ACO	;LOAD ACO
12705	062476	173014			SUBD	(R4),ACO	;*TEST INSTRUCTION
12706	062500	170203			STEPS	R3	;SAVE STATUS
12707	062502	174011			STD	ACO,(R1)	;SAVE ACO INTO RECDST
12708	062504	022703	003204		CMP	#3204,R3	;VERIFY STATUS
12709	062510	001401			BEQ	1\$	;BRANCH IF GOOD
12710	062512	104003			ERROR	+3	;FPP ERROR
12711							;BAD FPS STATUS
12712	062514	012704	003254	1\$:	MOV	#TAB6,R4	;POINT TO EXPECTED DATA
12713	062520	004767	050736		JSR	R7,DATVER	;VERIFY ACO
12714	062524	005767	120330		IST	COUNT	
12715	062530	001401			BEQ	2\$	;BRANCH IF GOOD
12716	062532	104003			ERROR	+3	;FPP ERROR
12717							;BAD ACO
12718	062534	012704	003404	2\$:	MOV	#TAB14,R4	;POINT TO FSRC AND ACO DATA
12719	062540	172414			LDD	(R4),ACO	;LOAD ACO DATA
12720	062542	173014			SUBD	(R4),ACO	;*TEST INSTRUCTION
12721	062544	170203			STEPS	R3	;SAVE FPS
12722	062546	174011			STD	ACO,(R1)	;SAVE ACO INTO RECDST
12723	062550	022703	003204		CMP	#3204,R3	;VERIFY FPS
12724	062554	001401			BEQ	3\$	;BRANCH IF GOOD
12725	062556	104003			ERROR	+3	;FPP ERROR
12726							;BAD ACO
12727	062560	012704	003254	3\$:	MOV	#TAB6,R4	;POINT TO EXPECTED DATA

```

12728 062564 004767 050677 JSR R7,DATVER ;VERIFY ACO
12729 062570 005767 120264 TST COUNT
12730 062574 001401 BEQ 4$ ;BRANCH IF GOOD
12731 062576 104003 ERROR +3 ;FPP ERROR
12732 ;BAD ACO
12733 062600 4$:
12734 ;
12735 ;
12736 ;
12737 ;
12738 ;
12739 ;TEST NORMALIZE
12740 ;
12741 ;
12742 ;
12743 ;
12744 062600 MNRM:
12745 ;
12746 ;
12747 062600 012702 003200 MOV #3200,R2 ;LOAD FPS
12748 062604 170102 LDFPS R2 ;
12749 062606 012705 003604 MOV #TAB31,R5 ;POINT TO FSRC DATA
12750 062612 012701 003574 MOV #TAB30,R1 ;POINT TO ACO DATA
12751 062616 172411 LDD (R1),ACO ;LOAD ACO
12752 062620 173015 SUBD (R5),ACO ;*TEST INSTRUCTION
12753 ;1 LEFT SHIFT
12754 062622 170203 STEPS R3 ;SAVE STATUS
12755 062624 012704 003102 MOV #RECDST,R4 ;POINT TO RECDATA
12756 062630 174014 STD ACO,(R4) ;SAVE ACO
12757 062632 012701 003634 MOV #TAB34,R1 ;POINT TO EXPECTED DATA
12758 062636 004767 050620 JSR R7,DATVER ;VERIFY DATA
12759 062642 005767 120212 TST COUNT
12760 062646 001401 BEQ 1$ ;BRANCH IF GOOD
12761 062650 104003 ERROR +3 ;FPP ERROR
12762 ;
12763 062652 012701 003614 1$: MOV #TAB32,R1 ;ACO DATA
12764 062656 012705 003624 MOV #TAB33,R5 ;FSRC DATA
12765 062662 172411 LDD (R1),ACO ;LOAD ACO
12766 062664 173015 SUBD (R5),ACO ;*TEST INSTRUCTION
12767 ;56 LEFT SHIFTS
12768 062666 012701 003102 MOV #RECDST,R1 ;SAVE DATA
12769 062672 174011 STD ACO,(R1) ;
12770 062674 004767 050562 JSR R7,DATVER
12771 062700 005767 120154 TST COUNT
12772 062704 001401 BEQ 2$ ;
12773 062706 104003 ERROR +3 ;FPP ERROR
12774 ;
12775 062710 2$:
12776 ;
12777 ;
12778 ;
12779 ;
12780 ;TEST ADDD WITH OVERFLOW AND UNDERFLOW
12781 ;
12782 ;
12783 ;

```

```

12784
12785 062710
12786
12787
12788 062710 012702 000200
12789 062714 170102
12790 062716 012704 003644
12791 062722 012701 003644
12792 062726 172411
12793 062730 172014
12794 062732 170203
12795 062734 012701 003102
12796 062740 174011
12797 062742 022703 000206
12798 062746 001401
12799 062750 104003
12800
12801 062752 012704 003254
12802 062756 004767 050500
12803 062762 005767 120072
12804 062766 001401
12805 062770 104003
12806
12807
12808 062772 012702 001200
12809 062776 170102
12810 063000 012704 003644
12811 063004 012701 003644
12812 063010 172411
12813 063012 012737 063026 000244
12814 063020 172014
12815 063022 170000
12816 063024 104003
12817
12818 063026 170203
12819 063030 012701 003102
12820 063034 174011
12821 063036 022703 101206
12822 063042 001401
12823 063044 104003
12824
12825 063046 012600
12826 063050 022700 063022
12827 063054 001401
12828 063056 104003
12829
12830 063060 012600
12831 063062 012704 003254
12832 063066 004767 050370
12833 063072 005767 117762
12834 063076 001401
12835 063100 104003
12836
12837
12838 063102 012702 000200
12839 063106 170102

;MOVAD:
;
;
;MOV #200,R2
;LDFPS R2
;POINT TO FSRC DATA
;POINT TO ACO DATA
;LOAD ACO WITH TEST DATA
;*TEST INSTRUCTION
;SAVE FPS
;POINT TO RECEIVED DATA TABLE
;SAVE ACO RESULT
;VERIFY STATUS
;BRANCH IF GOOD
;FPP ERROR
;BAD FPS
;POINT TO EXPECTED DATA
;VERIFY DATA
;BRANCH IF GOOD
;FPP ERROR
;BAD ACO
;OVERFLOW TRAPS ENABLED
2$: MOV #1200,R2
;LDFPS R2
;POINT TO FSRC DATA
;POINT TO ACO DATA
;LOAD ACO WITH TEST DATA
;CHANGE TRAP VECTOR
;*TEST INSTRUCTION
;FPP ERROR
;FAILED TO TRAP ON OVERFLOW
;SAVE FPS
;POINT TO RECEIVED DATA TABLE
;SAVE ACO RESULT
;VERIFY STATUS
;BRANCH IF GOOD
;FPP ERROR
;BAD FPS
;CHECK STORED PC
;BRANCH IF RETURN ADDRESS IS GOOD
;FPP ERROR
;BAD RETURN ADDRESS
;CLEAN UP STACK
;POINT TO EXPECTED DATA
;VERIFY DATA
;BRANCH IF GOOD
;FPP ERROR
;BAD ACO
;UNDERFLOW TRAPS DISABLED
7$: MOV #200,R2
;LDFPS R2
;SETUP FLOATING POINT STATUS
;LOAD FPS

```

N2

SEQ 0233

12840	063110	012737	133426	000244	MOV	#WLDTRP,#FPVEC	;REPLACE WILD TRAP VECTOR
12841	063116	012704	003274		MOV	#TAB7,R4	;POINT TO FSRC DATA
12842	063122	012701	003654		MOV	#TAB41,R1	;POINT TO ACO DATA
12843	063126	172411			LDD	(R1),ACO	;LOAD ACO WITH TEST DATA
12844	063130	172014			ADDD	(R4),ACO	;*TEST INSTRUCTION
12845	063132	170203			STFPS	R3	;SAVE FPS
12846	063134	012701	003102		MOV	#RECDST,R1	;POINT TO RECEIVED DATA TABLE
12847	063140	174011			STD	ACO,(R1)	;SAVE ACO RESULT
12848	063142	022703	000204		CMP	#204,R3	;VERIFY STATUS
12849	063146	001401			BEQ	8\$	;BRANCH IF GOOD
12850	063150	104003			ERROR	+3	;FPP ERROR
12851							;BAD FPS
12852	063152	012704	003254		8\$: MOV	#TAB6,R4	;POINT TO EXPECTED DATA
12853	063156	004767	050300		JSR	R7,DATVER	;VERIFY DATA
12854	063162	005767	117672		TST	COUNT	
12855	063166	001401			BEQ	9\$	;BRANCH IF GOOD
12856	063170	104003			ERROR	+7	;FPP ERROR
12857							;BAD ACO
12858					;UNDERFLOW TRAPS ENABLED		
12859	063172	012702	002200		9\$: MOV	#220,R2	;SETUP FLOATING POINT STATUS
12860	063176	170102			LDFPS	R2	;LOAD FPS
12861	063200	012737	063226	000244	MOV	#11,#FPVEC	;REPOSITION TRAP VECTOR
12862	063206	012704	003274		MOV	#TAB7,R4	;POINT TO FSRC DATA
12863	063212	012701	003654		MOV	#TAB41,R1	;POINT TO ACO DATA
12864	063216	172411			LDD	(R1),ACO	;LOAD ACO WITH TEST DATA
12865	063220	172014			ADDD	(R4),ACO	;*TEST INSTRUCTION
12866	063222	170000			CFCC		;COPY FPP CC
12867	063224	104003			ERROR	+3	;FPP ERROR
12868							;FAILED TO TRAP ON UNDERFLOW
12869	063226	170203			11\$: STFPS	R3	;SAVE FPS
12870	063230	012701	003102		MOV	#RECDST,R1	;POINT TO RECEIVED DATA TABLE
12871	063234	174011			STD	ACO,(R1)	;SAVE ACO RESULT
12872	063236	022703	102210		CMP	#102210,R3	;VERIFY STATUS
12873	063242	001401			BEQ	12\$	;BRANCH IF GOOD
12874	063244	104003			ERROR	+3	;FPP ERROR
12875							;BAD FPS
12876	063246	012605			12\$: MOV	(SP)+,R5	;GET ERROR PC
12877	063250	020527	063222		CMP	R5,#10\$	;VERIFY ERROR ADDRESS ON STACK
12878	063254	001401			BEQ	13\$	;BRANCH IF GOOD
12879	063256	104003			ERROR	+3	;FPP ERROR
12880							;BAD ERROR RETURN ON STACK
12881	063260	005726			13\$: TST	(SP)+	;RESTORE STACK
12882	063262	012704	003244		MOV	#TAB5A,R4	;POINT TO EXPECTED DATA
12883	063266	004767	050170		JSR	R7,DATVER	;VERIFY DATA
12884	063272	005767	117562		TST	COUNT	
12885	063276	001401			BEQ	14\$	;BRANCH IF GOOD
12886	063300	104003			ERROR	+4	;FPP ERROR
12887							;BAD ACO
12888					;UNDERFLOW WITH TRAPS DISABLED - NON-ZERO RESULT		
12889	063302	012702	000200		14\$: MOV	#200,R2	;SETUP FLOATING POINT STATUS
12890	063306	170102			LDFPS	R2	;LOAD FPS
12891	063310	012737	133426	000244	MOV	#WLDTRP,#FPVEC	;RESTORE TRAP VECTOR
12892	063316	012704	003654		MOV	#TAB41,R4	;POINT TO FSRC DATA
12893	063322	012701	003664		MOV	#TAB41,R1	;POINT TO ACO DATA
12894	063326	172411			LDD	(R1),ACO	;LOAD ACO WITH TEST DATA
12895	063330	172014			ADDD	(R4),ACO	;*TEST INSTRUCTION

13624	066610	004767	000424	JSR	R7,MLFSUB		IDO TEST
13625	066614	040177	177777	.WORD	40177,-1	!ACO	
13626	066620	040200	000000	.WORD	40200,0	!FSRC	
13627	066624	040177	177777	.WORD	40177,1		!RESULT
13628	066630	000040		.WORD	40	!TEST FPS	
13629	066632	000040		.WORD	40	!RESULTANT FPS	
13630				!6/NORMALIZE			
13631	066634	005037	002776	CLR	!0FLAG	!NO INTERRUPT	
13632	066640	004767	000374	JSR	R7,MLFSUB	!DO TEST	
13633	066644	040100	000000	.WORD	40100,0	!ACO	
13634	066650	040100	000000	.WORD	40100,0	!FSRC	
13635	066654	040020	000000	.WORD	40020,0	!RESULT	
13636	066660	000012		.WORD	12	!TEST FPS	
13637	066662	000000		.WORD	0	!RESULTANT FPS	
13638				!7/ROUND			
13639	066664	005037	002776	CLR	!0FLAG	!NO INTERRUPT	
13640	066670	004767	000344	JSR	R7,MLFSUB	!DO TEST	
13641	066674	017500	000000	.WORD	17500,0	!ACO	
13642	066700	023652	125252	.WORD	23652,125252	!FSRC	
13643	066704	003177	177777	.WORD	3177,-1	!RESULT	
13644	066710	007417		.WORD	7417	!TEST FPS	
13645	066712	007400		.WORD	7400	!RESULTANT FPS	
13646				!8/AC>0>FSRC ROUND			
13647	066714	005037	002776	CLR	!0FLAG	!NO INTERRUPT	
13648	066720	004767	000314	JSR	R7,MLFSUB	!DO TEST	
13649	066724	040342	177777	.WORD	40342,-1	!ACO	
13650	066730	176543	025252	.WORD	176543,025252	!FSRC	
13651	066734	176711	067324	.WORD	176711,67324	!RESULT	
13652	066740	007500		.WORD	7500	!TEST FPS	
13653	066742	007510		.WORD	7510	!RESULTANT FPS	
13654				!9/!AC<FSRC<0, ROUND			
13655	066744	005037	002776	CLR	!0FLAG	!NO INTERRUPT	
13656	066750	004767	000264	JSR	R7,MLFSUB	!DO TEST	
13657	066754	144600	000000	.WORD	144600,0	!ACO	
13658	066760	154000	000000	.WORD	154000,0	!FSRC	
13659	066764	060400	000000	.WORD	60400,0	!RESULT	
13660	066770	000017		.WORD	17	!TEST FPS	
13661	066772	000000		.WORD	0	!RESULT FPS	
13662				!10/AC<FSRC, ROUND			
13663	066774	005037	002776	CLR	!0FLAG	!NO INTERRUPT	
13664	067000	004767	000234	JSR	R7,MLFSUB	!DO TEST	
13665	067004	060000	000000	.WORD	60000,0	!ACO	
13666	067010	140377	177776	.WORD	140377,177776	!FSRC	
13667	067014	160177	177776	.WORD	160177,177776	!RESULT	
13668	067020	000017		.WORD	17	!TEST FPS	
13669	067022	000010		.WORD	10	!RESULT FPS	
13670				!11/AC>0>FSRC, TRUNCATE			
13671	067024	005037	002776	CLR	!0FLAG	!NO INTERRUPT	
13672	067030	004767	000204	JSR	R7,MLFSUB	!DO TEST	
13673	067034	060000	000000	.WORD	60000,0	!ACO	
13674	067040	140377	177776	.WORD	140377,177776	!FSRC	
13675	067044	160177	177776	.WORD	160177,177776	!RESULT	
13676	067050	007547		.WORD	7547	!TEST FPS	
13677	067052	007550		.WORD	7550	!RESULT FPS	
13678				!12/UNDERFLOW, NO INTERRUPTS			
13679	067054	012737	000002 002776	MOV	!0, !0FLAG	!NO INTERRUPT	

Address	Op-Code	Address	Op-Code	Comments
12953				
12954	063502	012702	000300	
12955	063506	170102		
12956	063510	012704	003704	
12957	063514	012701	003254	
12958	063520	172411		
12959	063522	177424		
12960	063524	012701	003102	
12961	063530	174011		
12962	063532	022704	003710	
12963	063536	001401		
12964	063540	104003		
12965				
12966	063542	012704	003714	
12967	063546	004767	047710	
12968	063552	005767	117302	
12969	063556	001401		
12970	063560	104003		
12971				
12972				
12973	063562	005002		
12974	063564	170102		
12975	063566	012704	003704	
12976	063572	012701	003424	
12977	063576	172411		
12978	063600	177424		
12979	063602	020427	003714	
12980	063606	001401		
12981	063610	104003		
12982				
12983	063612	170203		
12984	063614	012701	003102	
12985	063620	174011		
12986	063622	022703	000000	
12987	063626	001401		
12988	063630	104003		
12989				
12990	063632	012704	003754	
12991	063636	004767	047620	
12992	063642	005767	117212	
12993	063646	001401		
12994	063650	104003		
12995				
12996				
12997	063652	012702	000200	
12998	063656	170102		
12999	063660	005003		
13000	063662	177427	043243	
13001	063666	005203		
13002	063670	005203		
13003	063672	005203		
13004	063674	022703	000003	
13005	063700	001401		
13006	063702	104003		
13007				

103

SEQ 0236

13008						
13009	063704	012702	000200	6\$:	MOV	*200,R2
13010	063710	170102			LDFPS	R2
13011	063712	012704	003724		MOV	*TAB47,R4
13012	063716	012701	003704		MOV	*TAB45,R1
13013	063722	172411			LDD	(R1),ACO
13014	063724	177414			LDCFD	(R4),ACO
13015	063726	170203			STEPS	R3
13016	063730	012701	003102		MOV	*RECDST,R1
13017	063734	174011			STD	ACO,(R1)
13018	063736	022703	000210		CMP	*210,R3
13019	063742	001401			BEQ	7\$
13020	063744	104003			ERROR	*3
13021						
13022	063746	012704	003744	7\$:	MOV	*TAB48,R4
13023	063752	004767	047504		JSR	R7,DATVER
13024	063756	005767	117076		TST	COUNT
13025	063762	001401			BEQ	8\$
13026	063764	104003			ERROR	*3
13027						
13028						
13029	063766	012702	000200	8\$:	MOV	*200,R2
13030	063772	170102			LDFPS	R2
13031	063774	012704	003254		MOV	*TAB6,R4
13032	064000	012701	003744		MOV	*TAB48,R1
13033	064004	172411			LDD	(R1),ACO
13034	064006	177414			LDCFD	(R4),ACO
13035	064010	170203			STEPS	R3
13036	064012	012701	003102		MOV	*RECDST,R1
13037	064016	174011			STD	ACO,(R1)
13038	064020	022703	000204		CMP	*204,R3
13039	064024	001401			BEQ	9\$
13040	064026	104003			ERROR	*3
13041						
13042	064030	012704	003254	9\$:	MOV	*TAB6,R4
13043	064034	004767	047422		JSR	R7,DATVER
13044	064040	005767	117014		TST	COUNT
13045	064044	001401			BEQ	10\$
13046	064046	104003			ERROR	*3
13047						
13048	064050			10\$:		
13049						
13050						
13051						
13052						
13053						
13054						
13055						
13056						
13057						
13058						
13059	064050					
13060						
13061						
13062						
13063	064050	005037	002776			

```

;NEGATIVE OPERANDS
;SETUP FLOATING POINT STATUS
;LOAD FPS
;POINT TO FSRC DATA
;POINT TO ACO DATA
;LOAD ACO WITH TEST DATA
;*TEST INSTRUCTION
;SAVE FPS
;POINT TO RECEIVED DATA TABLE
;SAVE ACO RESULT
;VERIFY STATUS
;BRANCH IF GOOD
;FPP ERROR
;BAD FPS
;POINT TO EXPECTED DATA
;VERIFY DATA
;BRANCH IF GOOD
;FPP ERROR
;BAD ACO
;SETUP FLOATING POINT STATUS
;LOAD FPS
;POINT TO FSRC DATA
;POINT TO ACO DATA
;LOAD ACO WITH TEST DATA
;*TEST INSTRUCTION
;SAVE FPS
;POINT TO RECEIVED DATA TABLE
;SAVE ACO RESULT
;VERIFY STATUS
;BRANCH IF GOOD
;FPP ERROR
;BAD FPS
;POINT TO EXPECTED DATA
;VERIFY DATA
;BRANCH IF GOOD
;FPP ERROR
;BAD ACO
;TEST CMPD
;CMPD WITH FSRC-ACO-0
;CLR *201,AC
;SIGNAL THAT ACO REMAINS CONSTANT

```

```

13064 064054 004767 000152      JSR    R7,CMPRTN      ;ROUTINE TO TEST DATA
13065 064060 000000 000000 000000 .WORD    0,0,0,0      ;ACO AT START
13066 064066 000000
13067 064070 000000 000000 000000 .WORD    0,0,0,0      ;FSRC AT START
13068 064076 000000
13069 064100 000200      .WORD    200      ;FPS AT START (D)
13070 064102 000204      .WORD    204      ;FPS AT END
13071
13072 064104 012737 000001 002776 ; CMPD WITH EXP(FSRC)=0, EXP(ACO)=0
13073 064112 004767 000114      MOV     01,00FLAG      ; SIGNAL THAT ACO WILL = 0
13074 064116 000000 000000 000000 JSR     R7,CMPRTN      ;ROUTINE TO TEST DATA
13075 064124 125252      .WORD    0,0,0,125252 ;ACO AT START
13076 064126 000100 000022 000123 .WORD    100,22,123,123 ;FSRC AT START
13077 064134 000123
13078 064136 000200      .WORD    200      ;FPS AT START (D)
13079 064140 000204      .WORD    204      ;FPS AT END
13080
13081 064142 005037 002776 ; CMPD FSRC>EXP(ACO)=0
13082 064146 004767 000060      CLR     00FLAG      ;ACO REMAINS UNCHANGED
13083 064152 000400 012346 012346 JSR     R7,CMPRTN      ;ROUTINE TO TEST DATA
13084 064160 000023      .WORD    400,12346,12346,23 ;ACO AT START
13085 064162 000200 000000 000000 .WORD    200,0,0,0      ;FSRC AT START
13086 064170 000000
13087 064172 000200      .WORD    200      ;FPS AT START (D)
13088 064174 000210      .WORD    210      ;FPS AT END
13089
13090 064176 004767 000030 ; CMPD FSRC=ACO>0
13091 064202 077777 177777 177777 JSR     R7,CMPRTN      ;ROUTINE TO TEST DATA
13092 064210 177777      .WORD    77777,-1,-1,-1 ;ACO AT START
13093 064212 077777 177777 177777 .WORD    77777,-1,-1,-1 ;FSRC AT START
13094 064220 177777
13095 064222 000200      .WORD    200      ;FPS AT START (D)
13096 064224 000204      .WORD    204      ;FPS AT END
13097 064226 000167 000116      JMP     HOP44         ;HOP OVER SUBROUTINE
13098
13099
13100 ;*****
13101 ;*****
13102 ;COMPARE ROUTINE DATA TABLES
13103 ;
13104 ;          ACO
13105 ;          FSRC
13106 ;          FPS BEFORE EXECUTION
13107 ;          FPS AFTER EXECUTION
13108 ;          (FEC)
13109 ;*****
13110 ;*****
13111 ;
13112 064232 012605      CMPRTN: MOV     CSD,R5      ; RETURN ADDRESS TO USE AS POINTER
13113 064234 012702 000200      MOV     0200,R2      ;SET TO DOUBLE MODE FOR LOAD
13114 064240 170102      LDEFS   R2          ;LOAD FPS
13115 064242 010504      MOV     R5,R4        ;POINT TO FSRC DATA
13116 064244 062704 000010      ADD     010,R4        ;
13117 064250 010501      MOV     R5,R1        ;POINT TO ACO DATA
13118 064252 172411      LOD     (R1),ACO      ;LOAD ACO WITH TEST DATA
13119 064254 016502 000020      MOV     20(R5),R2      ;GET TEST FPS

```

```

13120 064250 170102          LDFPS R2          ;LOAD TEST FPS
13121 064262 173414          1$: CMPD (R4),ACO    ;*TEST INSTRUCTION
13122 064264 170203          STFPS R3          ;SAVE FPS
13123 064266 012702 000200    MOV #200,R2      ;SET FPP TO DOUBLE
13124 064272 170102          LDFPS R2
13125 064274 012701 003102    MOV #RECDST,R1    ;POINT TO RECEIVED DATA TABLE
13126 064300 174011          STD ACO,(R1)       ;SAVE ACO RESULT
13127 064302 026503 000022    CMP 22(R5),R3    ;VERIFY STATUS
13128 064306 001401          BEQ 2$            ;BRANCH IF GOOD
13129 064310 104003          ERROR +3          ;FPP ERROR
13130                                ;BAD FPS
13131 064312 005737 002776    2$: TST #0FLAG    ;SEE IF ACO REMAINS UNCHANGED
13132 064316 001403          BEQ 3$            ;BRANCH IF ACO STAYS THE SAME
13133 064320 012704 003254    MOV #TAB6,R4      ;ACO=0
13134 064324 000401          BR 4$            ;GO VERIFY DATA
13135 064326 010504          3$: MOV R5,R4      ;POINT TO EXPECTED DATA
13136 064330 004767 047126    4$: JSR R7,DATVER ;VERIFY DATA
13137 064334 005767 116520    TST COUNT
13138 064340 001401          BEQ 5$            ;BRANCH IF GOOD
13139 064342 104003          ERROR +3          ;FPP ERROR
13140                                ;BAD ACO
13141 064344 000165 000024    5$: JMP 24(R5)    ;RETURN
13142 064350          HOP44:
13143
13144
13145
13146
13147
13148
13149
13150
13151
13152
13153
13154 064350          MDIVF:
13155
13156
13157
13158 064350 012737 000002 002776    ;1/EXP[AC]=FSRC=0
13159 064356 004767 000706          MOV #0,#0FLAG ;NO INTERRUPT, BUT FEC
13160 064362 000100 000027          JSR R7,DVFSUB ;DO TEST
13161 064366 000000 000000          .WORD 100,27  ;ACO
13162 064372 000100 000027          .WORD 0,0     ;FSRC
13163 064376 040000          .WORD 100,27       ;RESULT
13164 064400 140000          .WORD 40000        ;TEST FPS
13165 064402 000004          .WORD 140000       ;RESULT FPS
13166                                .WORD 4      ;FEC
13167
13168 064404 012737 000001 002776    ;2/AC=EXP[FSRC]=0
13169 064412 004767 000652          ;TRAPS ENABLED
13170 064416 000000 000000          MOV #1,#0FLAG ;INTERUPT
13171 064422 000100 000000          JSR R7,DVFSUB ;DO TEST
13172 064426 000000 000000          .WORD 0,0     ;ACO
13173 064432 000000 000000          .WORD 100,0    ;FSRC
13174 064434 100000          .WORD 0,0           ;RESULT
13175 064436 000004          .WORD 0              ;TEST FPS
13176                                .WORD 100000   ;RESULT FPS
13177                                .WORD 4         ;FEC

```

13176					;3/FSRC>ACO=0	
13177	064440	005037	002776		CLR 00FLAG	;NO INTERRUPT
13178	064444	004767	000620		JSR R7,DVFSUB	;DO TEST
13179	064450	000177	000234		.WORD 177,234	;ACO
13180	064454	004100	000000		.WORD 4100,0	;FSRC
13181	064460	000000	000000		.WORD 0,0	;RESULT
13182	064464	007400			.WORD 7400	;TEST FPS
13183	064466	007404			.WORD 7404	;RESULT FPS
13184						
13185	064470	012737	000001	002776	;4/ACO>EXP(FSRC)=0	
13186	064476	004767	000566		MOV 01,00FLAG	; INTERRUPT
13187	064502	040200	104210		JSR R7,DVFSUB	;DO TEST
13188	064506	000125	025252		.WORD 40200,104210	;ACO
13189	064512	040200	104210		.WORD 125,25252	;FSRC
13190	064516	007557			.WORD 40200,104210	;RESULT
13191	064520	107557			.WORD 7557	;TEST FPS
13192	064522	000004			.WORD 107557	;RESULT FPS
13193					.WORD 4	;FEC
13194	064524	005037	002776		;5/EXP(AC)=EXP(FSRC)	
13195	064530	004767	000534		CLR 00FLAG	;NO INTERRUPT
13196	064534	077760	177777		JSR R7,DVFSUB	;DO TEST
13197	064540	077760	000000		.WORD 77760,-1	;ACO
13198	064544	040200	104210		.WORD 77760,0	;FSRC
13199	064550	007414			.WORD 40200,104210	;RESULT
13200	064552	007400			.WORD 7414	;TEST FPS
13201					.WORD 7400	;RESULT FPS
13202	064554	005037	002776		;6/AC=FSRC	
13203	064560	004767	000504		CLR 00FLAG	;NO INTERRUPT
13204	064564	052525	052525		JSR R7,DVFSUB	;DO TEST
13205	064570	052525	052525		.WORD 52525,52525	;ACO
13206	064574	040200	000000		.WORD 52525,52525	;FSRC
13207	064600	007400			.WORD 40200,0	;RESULT
13208	064602	007400			.WORD 7400	;TEST FPS
13209					.WORD 7400	;RESULT FPS
13210	064604	005037	002776		;7/FSRC>0<ACO, ROUND	
13211	064610	004767	000454		CLR 00FLAG	;NO INTERRUPT
13212	064614	077777	125252		JSR R7,DVFSUB	;DO TEST
13213	064620	040300	000000		.WORD 77777,125252	;ACO
13214	064624	077652	070707		.WORD 40300,0	;FSRC
13215	064630	007400			.WORD 77652,070707	;RESULT
13216	064632	007400			.WORD 7400	;TEST FPS
13217					.WORD 7400	;RESULT FPS
13218	064634	005037	002776		;8/AC>0<FSRC	
13219	064640	004767	000424		CLR 00FLAG	;NO INTERRUPT
13220	064644	055377	177777		JSR R7,DVFSUB	;DO TEST
13221	064650	055300	000000		.WORD 55377,-1	;ACO
13222	064654	040252	125252		.WORD 55300,0	;FSRC
13223	064660	000000			.WORD 40252,125252	;RESULT
13224	064662	000000			.WORD 0	;TEST FPS
13225					.WORD 0	;RESULT FPS
13226	064664	005037	002776		;9/FSRC>AC>0	
13227	064670	004767	000374		CLR 00FLAG	;NO INTERRUPT
13228	064674	064600	000001		JSR R7,DVFSUB	;DO TEST
13229	064700	066600	000000		.WORD 64600,1	;ACO
13230	064704	036200	000001		.WORD 66600,0	;FSRC
13231	064710	000000			.WORD 36200,1	;RESULT
					.WORD 0	;TEST FPS

H3

COKDABO RDJ11 B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 241
 COKDAB.P11 05-APR-84 16:45 FLOATING POINT TESTS

SEQ 0240

13232	064712	000000		.WORD	0		;RESULT FPS
13233				;10/AC>FSRC>0			
13234	064714	005037	002776	CLR	@FLAG		;NO INTERRUPT
13235	064720	004767	000344	JSR	R7,DVFSUB		;DO TEST
13236	064724	012345	156024	.WORD	12345,156024		;ACO
13237	064730	005600	000000	.WORD	05600,0		;FSRC
13238	064734	044745	156024	.WORD	44745,156024		;RESULT
13239	064740	000017		.WORD	17		;TEST FPS
13240	064742	000000		.WORD	0		;RESULT FPS
13241				;11/FSRC<0			
13242	064744	005037	002776	CLR	@FLAG		;NO INTERRUPT
13243	064750	004767	000314	JSR	R7,DVFSUB		;DO TEST
13244	064754	040422	101010	.WORD	40422,101010		;ACO
13245	064760	140511	101010	.WORD	140511,101010		;FSRC
13246	064764	140072	020167	.WORD	140072,20167		;RESULT
13247	064770	000057		.WORD	57		;TEST FPS
13248	064772	000050		.WORD	50		;RESULT FPS
13249				;12/AC<0			
13250	064774	005037	002776	CLR	@FLAG		;NO INTERRUPT
13251	065000	004767	000264	JSR	R7,DVFSUB		;DO TEST
13252	065004	160077	000101	.WORD	160077,101		;ACO
13253	065010	040417	177777	.WORD	40417,-1		;FSRC
13254	065014	157651	143527	.WORD	157651,143527		;RESULT
13255	065020	000007		.WORD	7		;TEST FPS
13256	065022	000010		.WORD	10		;RESULT FPS
13257				;13/TRUNCATE TEST			
13258	065024	005037	002776	CLR	@FLAG		;NO INTERRUPT
13259	065030	004767	000234	JSR	R7,DVFSUB		;DO TEST
13260	065034	060100	000177	.WORD	60100,177		;ACO
13261	065040	040300	000000	.WORD	40300,0		;FSRC
13262	065044	060000	000124	.WORD	60000,124		;RESULT
13263	065050	000040		.WORD	40		;TEST FPS
13264	065052	000040		.WORD	40		;RESULT FPS
13265				;14/ROUND TEST			
13266							
13267	065054	005037	002776	CLR	@FLAG		;NO INTERRUPT
13268	065060	004767	000204	JSR	R7,DVFSUB		;DO TEST
13269	065064	060100	000177	.WORD	60100,177		;ACO
13270	065070	040300	000000	.WORD	40300,0		;FSRC
13271	065074	060000	000125	.WORD	60000,125		;RESULT
13272	065100	000000		.WORD	0		;TEST FPS
13273	065102	000000		.WORD	0		;RESULT FPS
13274				;15/OVERFLOW, INTERRUPTS ENABLED			
13275	065104	012737	000001	MOV	#1,@FLAG		;INTERRUPT
13276	065112	004767	000152	JSR	R7,DVFSUB		;DO TEST
13277	065116	177700	000000	.WORD	177700,0		;ACO
13278	065122	000200	000000	.WORD	200,0		;FSRC
13279	065126	137700	000000	.WORD	137700,0		;RESULT
13280	065132	001100		.WORD	1100		;TEST FPS
13281	065134	101112		.WORD	101112		;RESULT FPS
13282	065136	000010		.WORD	10		;FEC
13283				;16/OVERFLOW, TRAPS DISABLED			
13284	065140	012737	000002	MOV	#2,@FLAG		;NO INTERRUPT
13285	065146	004767	000116	JSR	R7,DVFSUB		;DO TEST
13286	065152	000200	000000	.WORD	200,0		;ACO
13287	065156	177700	000000	.WORD	177700,0		;FSRC

COKDABO KDJ11 B CLUSTER MAC111 30(1046) 05 APR 84 16:45 PAGE 242
COKDAB.P11 05-APR-84 16:45 FLOATING POINT TESTS

SEQ 0241

```

13288 065162 000000 000000 .WORD 0,0 ;RESULT
13289 065166 041100 .WORD 41100 ;TEST FPS
13290 065170 041104 .WORD 41104 ;RESULT FPS
13291 065172 000010 .WORD 10 ;FEC OVERFLOW
13292 ;17/UNDERFLOW, TRAPS ENABLED, UV RESULT
13293 065174 012737 000001 002776 MOV #1,0#FLAG ;INTERRUPT
13294 065202 004767 000062 JSR R7,DVFSUB ;DO TEST
13295 065206 100200 000000 .WORD 100200,0 ;ACO
13296 065212 040377 177777 .WORD 40377,-1 ;FSRC
13297 065216 100000 000001 .WORD 100000,1 ;RESULT
13298 065222 002000 .WORD 2000 ;TEST FPS
13299 065224 102014 .WORD 102014 ;RESULT FPS
13300 065226 000012 .WORD 12 ;FEC
13301 ;18/UNDERFLOW, TRAPS ENABLED, ROUND
13302 065230 012737 000001 002776 MOV #1,0#FLAG ;INTERRUPT
13303 065236 004767 000026 JSR R7,DVFSUB ;DO TEST
13304 065242 030325 025252 .WORD 30325,25252 ;ACO
13305 065246 076777 023456 .WORD 76777,23456 ;FSRC
13306 065252 071525 157716 .WORD 71525,157716 ;RESULT
13307 065256 002537 .WORD 2537 ;TEST FPS
13308 065260 102500 .WORD 102500 ;RESULT FPS
13309 065262 000012 .WORD 12 ;FEC
13310 ;
13311 ;
13312 065264 000167 000212 JMP HOP10 ;GO TO N. TEST
13313 ;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
13314 ;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
13315 ;DIVE SUBROUTINE:
13316 ; ACO
13317 ; FSRC
13318 ; FPS BEFORE EXECUTION
13319 ; FPS AFTER EXECUTION
13320 ; (FEC)
13321 ;
13322 ;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
13323 ;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
13324 ;
13325 065270 012605 DVFSUB: MOV (SP)+,R5 ; RETURN ADDRESS TO USE AS POINTER
13326 065272 012737 065346 000244 MOV #503,0#FPVEC ;REDIRECT TRAP VECTOR
13327 065300 012702 000200 MOV #200,R2 ;SET TO DOUBLE MODE FOR LOAD
13328 065304 170102 LDFPS R2 ;LOAD FPS
13329 065306 010504 MOV R5,R4 ;POINT TO FSRC DATA
13330 065310 062704 000004 ADD #4,R4
13331 065314 172415 LDD (R5),ACO ;LOAD ACO WITH TEST DATA
13332 065316 016502 000014 MOV 14(R5),R2 ;GET TEST FPS
13333 065322 170102 LDFPS R2 ;LOAD TEST FPS
13334 ;
13335 065324 174414 DIVE (R4),ACO ;*TEST INSTRUCTION
13336 065326 170001 1$: SETF ;WAIT FOR POSSIBLE FPA TRAP.
13337 ;
13338 ;INSTRUCTION DIDNT TRAP
13339 065330 032737 000001 002776 BIT #1,0#FLAG ;VERIFY A NO TRAP CONDITION
13340 065336 001420 BEQ 2$ ;BRANCH IF GOOD
13341 065340 104003 ERROR 1$ ;FPP ERROR
13342 065342 000167 000032 JMP 2$ ;INSTRUCTION SHOULD HAVE TRAPPED
13343 ;REJOIN CODE

```



```

13344
13345 ; INSTRUCTION TRAPPED
13346 065346 032737 000001 002776 50$: BIF #1, @FLAG ;SEE IF EXPECTING A TRAP
13347 065354 001003 BNE 51$ ;BRANCH IF EXPECTING A TRAP
13348 065356 104003 ERROR +3 ;FPP ERROR
13349 ;INSTRUCTION WASNT SUPPOSE TO TRAP
13350 065360 000167 000014 JMP 2$ ;REJOIN CODE
13351 065364 012604 51$: MOV (SP)+, R4 ;SEE IF PC = INSTRUCTION
13352 065366 005726 TST (SP), ;CLEAN UP STACK
13353 065370 022704 065326 CMP #1$, R4 ;
13354 065374 001401 BEQ 2$ ;BRANCH IF GOOD COMPARE
13355 065376 104003 ERROR +3 ;FPP ERROR
13356 ;PC WAS INCORRECT
13357
13358 ;COMMON CODE FOR TRAP AND NO TRAP
13359 065400 170203 2$: STEPS R3 ;SAVE FPS
13360 065402 012702 000200 MOV #200, R2 ;SET FPP TO DOUBLE
13361 065406 170102 LDFPS R2
13362 065410 012701 003102 MOV #RECDST, R1 ;POINT TO RECEIVED DATA TABLE
13363 065414 174011 STD ACO, (R1) ;SAVE ACO RESULT
13364 065416 026503 000016 CMP 16(R5), R3 ;VERIFY STATUS
13365 065422 001401 BEQ 3$ ;BRANCH IF GOOD
13366 065424 104003 ERROR +3 ;FPP ERROR
13367 ;BAD FPS
13368 065426 010504 3$: MOV R5, R4 ;POINT TO EXPECTED DATA
13369 065430 062704 000010 ADD #10, R4
13370 065434 004767 046004 4$: JSR R7, DATVER ;VERIFY DATA
13371 065440 005767 115414 TST COUNT
13372 065444 001401 BEQ 5$ ;BRANCH IF GOOD
13373 065446 104003 ERROR +3 ;FPP ERROR
13374 ;BAD ACO
13375 065450 005737 002776 5$: TST @FLAG ;SEE IF NEED TO CHECK FEC
13376 065454 001002 BNE 6$ ;BRANCH IF NEED TO CHECK
13377 065456 000165 000020 JMP 20(R5) ;RETURN FROM TEST
13378 065462 170301 6$: STST R1 ;SAVE FEC
13379 065464 016504 000020 MOV 20(R5), R4 ;GET FEC
13380 065470 020401 CMP R4, R1 ;VERIFY FEC
13381 065472 001401 BEQ 7$ ;BRANCH IF GOOD
13382 065474 104003 ERROR +3 ;FPP ERROR
13383 ;BAD FEC
13384 065476 000165 000022 7$: JMP 22(R5) ;RETURN FROM TEST
13385
13386 065502 ;HOP10:
13387 ;
13388 ;
13389 ;
13390 ;TEST DIVD
13391 ;
13392 ;
13393 ;
13394 ;
13395 065502 ;MOIVD:
13396 ;
13397 ;
13398 ;
13399 065502 012737 000002 002776 ;1/AC +SRC 0 TRAPS DISABLED
MOV #2, @FLAG ;NO INTERRUPT

```

Address	Hex Data	Hex Data	Hex Data	Hex Data	Assembly	Comments
13400	065510	004767	000516		JSR R7,DVDSUB	;DO TEST
13401	065514	000000	000000	000000	.WORD 0,0,0,1	;AC0
13402	065522	000001				
13403	065524	000100	000000	000000	.WORD 100,0,0,0	;FSRC
13404	065532	000000				
13405	065534	000000	000000	000000	.WORD 0,0,0,1	;RESULT
13406	065542	000001				
13407	065544	040000			.WORD 40000	; TEST FPS
13408	065546	140000			.WORD 140000	;RESULT FPS
13409	065550	000004			.WORD 4	;FEC
13410					;2/FSRC=0, TRAPS ENABLED	
13411	065552	012737	000001	002776	MOV #1,#FLAG	; INTERRUPT
13412	065560	004767	000446		JSR R7,DVDSUB	;DO TEST
13413	065564	000402	000000	000000	.WORD 402,0,0,0	;AC0
13414	065572	000000				
13415	065574	000000	000000	000000	.WORD 0,0,0,0	;FSRC
13416	065602	000000				
13417	065604	000402	000000	000000	.WORD 402,0,0,0	;RESULT
13418	065612	000000				
13419	065614	000200			.WORD 200	; TEST FPS
13420	065616	100200			.WORD 100200	;RESULT FPS
13421	065620	000004			.WORD 4	;FEC
13422					;3/ROUND	
13423	065622	005037	002776		CLR #FLAG	;NO INTERRUPT
13424	065626	004767	000400		JSR R7,DVDSUB	;DO TEST
13425	065632	034300	000000	000000	.WORD 34300,0,0,1	;AC0
13426	065640	000001				
13427	065642	140300	000000	000000	.WORD 140300,0,0,0	;FSRC
13428	065650	000000				
13429	065652	134200	000000	000000	.WORD 134200,0,0,1	;RESULT
13430	065660	000001				
13431	065662	000200			.WORD 200	; TEST FPS
13432	065664	000210			.WORD 210	;RESULT FPS
13433					;4/TRUNCATE	
13434	065666	005037	002776		CLR #FLAG	;NO INTERRUPT
13435	065672	004767	000354		JSR R7,DVDSUB	;DO TEST
13436	065676	034300	000000	000000	.WORD 34300,0,0,1	;AC0
13437	065704	000001				
13438	065706	140300	000000	000000	.WORD 140300,0,0,0	;FSRC
13439	065714	000000				
13440	065716	134200	000000	000000	.WORD 134200,0,0,0	;RESULT
13441	065724	000000				
13442	065726	000240			.WORD 240	; TEST FPS
13443	065730	000250			.WORD 250	;RESULT FPS
13444					;5/ROUND NEGATIVE AC, FSRC	
13445	065732	005037	002776		CLR #FLAG	;NO INTERRUPT
13446	065736	004767	000270		JSR R7,DVDSUB	;DO TEST
13447	065742	177642	000000	000000	.WORD 177642,0,0,151	;AC0
13448	065750	000151				
13449	065752	166600	000000	000000	.WORD 166600,0,0,123	;FSRC
13450	065760	000123				
13451	065762	051242	000000	000000	.WORD 51242,0,0,0	;RESULT
13452	065770	000000				
13453	065772	000200			.WORD 20	


```

13456 065776 005037 002776          CLR      @#FLAG          ;NO INTERRUPT
13457 066002 004767 000224          JSR      R7,DVDSUB          ;DO TEST
13458 066006 177642 000000 000000    .WORD    177642,0,0,151    ;ACO
13459 066014 000151                .WORD    166600,0,0,123    ;FSRC
13460 066016 166600 000000 000000    .WORD    51241,-1,-1,-1    ;RESULT
13461 066024 000123                .WORD    240              ; TEST FPS
13462 066026 051241 177777 177777    .WORD    240              ;RESULT FPS
13463 066034 177777                ;7/AC=FSRC
13464 066036 000240                CLR      @#FLAG          ;NO INTERRUPT
13465 066040 000240                JSR      R7,DVDSUB          ;DO TEST
13466                                .WORD    55521,47621,100333,-1  ;ACO
13467 066042 005037 002776          .WORD    55521,47621,100333,-1  ;FSRC
13468 066046 004767 000160          .WORD    40200,0,0,0      ;RESULT
13469 066052 055521 047621 100333    .WORD    7717            ; TEST FPS
13470 066060 177777                .WORD    7700            ;RESULT FPS
13471 066062 055521 047621 100333    ;8/UNDERFLOW TRAPS ENABLED, UV RESULT
13472 066070 177777                MOV      @1,@#FLAG          ;INTERRUPT
13473 066072 040200 000000 000000    JSR      R7,DVDSUB          ;DO TEST
13474 066100 000000                .WORD    100200,0,0,0      ;ACO
13475 066102 007717                .WORD    77777,0,0,0      ;FSRC
13476 066104 007700                .WORD    140400,100200,100200,100201  ;RESULT
13477                                .WORD    2200              ; TEST FPS
13478 066106 012737 000001 002776    .WORD    102210          ;RESULT FPS
13479 066114 004767 000112          .WORD    12              ;FEC
13480 066120 100200 000000 000000    ;9/OVERFLOW TRAPS ENABLED
13481 066126 000000                MOV      @1,@#FLAG          ;INTERRUPT
13482 066130 077777 000000 000000    JSR      R7,DVDSUB          ;DO TEST
13483 066136 000000                .WORD    77000,123465,12346,525  ;ACO
13484 066140 140400 100200 100200    .WORD    303,1,140000,140001  ;FSRC
13485 066146 100201                .WORD    36650,163002,103645,64003  ;RESULT
13486 066150 002200                .WORD    1700            ; TEST FPS
13487 066152 102210                .WORD    101702          ;RESULT FPS
13488 066154 000012                .WORD    10              ;FEC
13489                                ;
13490 066156 012737 000001 002776    JMP      HOP11              ;POP OVER SUBROUTINE
13491 066164 004767 000042          ;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
13492 066170 077000 123465 012346    ;XXYYXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
13493 066176 000525                ;
13494 066200 000303 000001 140000    ;DIVD SUBROUTINE:
13495 066206 140001                ;
13496 066210 036650 163002 103645    ; ACO
13497 066216 064003                ;
13498 066220 001700                ; FSRC
13499 066222 101702                ;
13500 066224 000010                ; FPS BEFORE EXECUTION
13501                                ; FPS AFTER EXECUTION
13502                                ;
13503 066226 000167 000212          ;
13504                                ;
13505                                ;
13506                                ;
13507                                ;
13508                                ;
13509                                ;
13510                                ;
13511                                ;

```

M3

CORDARO RDJ11-B CLUSTER MACY11 30(1046) 05 APR-84 16:45 PAGE 246
 CORDAR.P11 05-APR-84 16:45 FLOATING POINT TESTS

SEQ 0245

```

13512                                     (FEC)
13513                                     ;
13514                                     ;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
13515                                     ;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
13516
13517 066232 012605 DVDSUB: MOV      (SP)+,R5          ; RETURN ADDRESS TO USE AS POINTER
13518 066234 012737 066310 000244 MOV      #50$,@#FPVEC      ; REDIRECT TRAP VECTOR
13519 066242 012702 000200 MOV      #200,R2          ; SET TO DOUBLE MODE FOR LOAD
13520 066246 170102 LDFPS      R2              ; LOAD FPS
13521 066250 010504 MOV      R5,R4              ; POINT TO FSRC DATA
13522 066252 062704 000010 ADD      #10,R4
13523 066256 172415 LDD      (R5),AC0          ; LOAD ACO WITH TEST DATA
13524 066260 016502 000030 MOV      30(R5),R2      ; GET TEST FPS
13525 066264 170102 LDFPS      R2              ; LOAD TEST FPS
13526
13527 066266 174414 DIVD      (R4),AC0          ; *TEST INSTRUCTION
13528 066270 170000 1$:      CFCC              ; WAIT FOR POSSIBLE FPA TRAP.
13529
13530                                     ; INSTRUCTION DIDNT TRAP
13531 066272 032737 000001 002776 BIT      #1,@#FLAG          ; VERIFY A NO TRAP CONDITION
13532 066300 001420 BEQ      2$              ; BRANCH IF GOOD
13533 066302 104003 ERROR    +3              ; FPP ERROR
13534                                     ; INSTRUCTION SHOULD HAVE TRAPPED
13535 066304 000167 000032 JMP      2$              ; REJOIN CODE
13536
13537                                     ; INSTRUCTION TRAPPED
13538 066310 032737 000001 002776 50$:      BIT      #1,@#FLAG          ; SEE IF EXPECTING A TRAP
13539 066316 001003 BNE      51$              ; BRANCH IF EXPECTING A TRAP
13540 066320 104003 ERROR    +3              ; FPP ERROR
13541                                     ; INSTRUCTION WASNT SUPPOSE TO TRAP
13542 066322 000167 000014 JMP      2$              ; REJOIN CODE
13543 066326 012604 51$:      MOV      (SP)+,R4          ; SEE IF PC = INSTRUCTION
13544 066330 005726 TST      (SP)+          ; CLEAN UP STACK
13545 066332 022704 066270 CMP      #1$,R4
13546 066336 001401 BEQ      2$              ; BRANCH IF GOOD COMPARE
13547 066340 104003 ERROR    +3              ; FPP ERROR
13548                                     ; PC WAS INCORRECT
13549
13550                                     ; COMMON CODE FOR TRAP AND NO TRAP
13551 066342 170203 2$:      STEPS      R3              ; SAVE FPS
13552 066344 012702 000200 MOV      #200,R2          ; SET FPP TO DOUBLE
13553 066350 170102 LDFPS      R2
13554 066352 012701 003102 MOV      #RECDST,R1          ; POINT TO RECEIVED DATA TABLE
13555 066356 174011 STD      ACO,(R1)          ; SAVE ACO RESULT
13556 066360 026503 000032 CMP      30(R5),R3      ; VERIFY STATUS
13557 066364 001401 BEQ      3$              ; BRANCH IF GOOD
13558 066366 104003 ERROR    +3              ; FPP ERROR
13559                                     ; BAD FPS
13560 066370 010504 3$:      MOV      R5,R4          ; POINT TO EXPECTED DATA
13561 066372 062704 ADD      #20,R4
13562 066376 004767 045060 4$:      JSR      R7,DATVER          ; VERIFY DATA
13563 066402 005767 114452 TST      COUNT
13564 066406 001401 BEQ      5$              ; BRANCH IF GOOD
13565 066410 104003 ERROR    +3              ; FPP ERROR
13566                                     ; BAD ACO
13567 066412 005737 002776 5$:      TST      @#FLAG          ; SEE IF NEED TO CHECK FEC

```

N3

COKDABO KDJ11-B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 247
 COKDAB.P11 05-APR-84 16:45 FLOATING POINT TESTS

SEQ 0246

```

13568 066416 001002          BNE      6$              ;BRANCH IF NEED TO CHECK
13569 066420 000165 000034    JMP      34(R5)          ;RETURN FROM TEST
13570 066424 170301          STST     R1              ;SAVE FEC
13571 066426 016504 000034    MOV      34(R5),R4        ;GET FEC
13572 066432 020401          CMP      R4,R1           ;VERIFY FEC
13573 066434 001401          BEQ      7$              ;BRANCH IF GOOD
13574 066436 104003          ERROR    +3              ;FPP ERROR
13575                                ;BAD FEC
13576 066440 000165 000036    JMP      36(R5)          ;RETURN FROM TEST
13577                                ;
13578 066444                                ;HOP11:
13579                                ;
13580                                ;
13581                                ;-----
13582                                ;TEST MULF
13583                                ;
13584                                ;
13585                                ;
13586                                ;
13587 066444                                ;MMULF:
13588                                ;
13589                                ;
13590                                ;1/AC0=FSRC=0 - INTERRUPTS DISABLED
13591 066444 005037 002776    CLR      @#FLAG          ;NO INTERRUPT
13592 066450 004767 000564    JSR      R7,MLFSUB        ;DO TEST
13593 066454 000000 000000    .WORD    0,0           ;ACO
13594 066460 000000 000000    .WORD    0,0           ;FSRC
13595 066464 000000 000000    .WORD    0,0           ;RESULT
13596 066470 007517          .WORD    7517          ;TEST FPS
13597 066472 007504          .WORD    7504          ;RESULTANT FPS
13598                                ;2/AC>FSRC=0 - INTERRUPTS ON
13599 066474 005037 002776    CLR      @#FLAG          ;NO INTERRUPT
13600 066500 004767 000534    JSR      R7,MLFSUB        ;DO TEST
13601 066504 000200 000000    .WORD    200,0       ;ACO
13602 066510 000000 000000    .WORD    0,0           ;FSRC
13603 066514 000000 000000    .WORD    0,0           ;RESULT
13604 066520 000013          .WORD    13           ;TEST FPS
13605 066522 000004          .WORD    4            ;RESULTANT FPS
13606                                ;3/AC=0 FSRC>0 -
13607 066524 005037 002776    CLR      @#FLAG          ;NO INTERRUPT
13608 066530 004767 000504    JSR      R7,MLFSUB        ;DO TEST
13609 066534 000100 000000    .WORD    100,0       ;ACO
13610 066540 000300 000000    .WORD    300,0       ;FSRC
13611 066544 000000 000000    .WORD    0,0           ;RESULT
13612 066550 007500          .WORD    7500          ;TEST FPS
13613 066552 007504          .WORD    7504          ;RESULTANT FPS
13614                                ;4/AC=1 FSRC - ROUND
13615 066554 005037 002776    CLR      @#FLAG          ;NO INTERRUPT
13616 066560 004767 000454    JSR      R7,MLFSUB        ;DO TEST
13617 066564 040200 000000    .WORD    40200,0     ;ACO
13618 066570 040177 177777    .WORD    40177,-1    ;FSRC
13619 066574 040177 177777    .WORD    40177,-1    ;RESULT
13620 066600 000000          .WORD    0            ;TEST FPS
13621 066602 000000          .WORD    0            ;RESULTANT FPS
13622                                ;5/TRUNCATE
13623 066604 005037 002776    CLR      @#FLAG          ;NO INTERRUPT

```

13624	066610	004767	000424	JSR	R7,M,F SUB	NO INTERRUPT	DO TEST
13625	066614	040177	177777	.WORD	40177,-1	ACO	
13626	066620	040200	000000	.WORD	40200,0	FSRC	
13627	066624	040177	177777	.WORD	40177,1	RESULT	
13628	066630	000040		.WORD	40	TEST FPS	
13629	066632	000040		.WORD	40	RESULTANT FPS	
13630				16/NORMALIZE			
13631	066634	005037	002776	CLR	00FLAG	NO INTERRUPT	
13632	066640	004767	000374	JSR	R7,M,F SUB	DO TEST	
13633	066644	040100	000000	.WORD	40100,0	ACO	
13634	066650	040100	000000	.WORD	40100,0	FSRC	
13635	066654	040020	000000	.WORD	40020,0	RESULT	
13636	066660	000012		.WORD	12	TEST FPS	
13637	066662	000000		.WORD	0	RESULTANT FPS	
13638				17/ROUND			
13639	066664	005037	002776	CLR	00FLAG	NO INTERRUPT	
13640	066670	004767	000344	JSR	R7,M,F SUB	DO TEST	
13641	066674	017500	000000	.WORD	17500,0	ACO	
13642	066700	023652	125252	.WORD	23652,125252	FSRC	
13643	066704	003177	177777	.WORD	3177,-1	RESULT	
13644	066710	007417		.WORD	7417	TEST FPS	
13645	066712	007400		.WORD	7400	RESULTANT FPS	
13646				18/AC>0>FSRC ROUND			
13647	066714	005037	002776	CLR	00FLAG	NO INTERRUPT	
13648	066720	004767	000314	JSR	R7,M,F SUB	DO TEST	
13649	066724	040342	177777	.WORD	40342,-1	ACO	
13650	066730	176543	025252	.WORD	176543,025252	FSRC	
13651	066734	176711	067324	.WORD	176711,67324	RESULT	
13652	066740	007500		.WORD	7500	TEST FPS	
13653	066742	007510		.WORD	7510	RESULTANT FPS	
13654				19/IAC<FSRC<0, ROUND			
13655	066744	005037	002776	CLR	00FLAG	NO INTERRUPT	
13656	066750	004767	000264	JSR	R7,M,F SUB	DO TEST	
13657	066754	144600	000000	.WORD	144600,0	ACO	
13658	066760	154000	000000	.WORD	154000,0	FSRC	
13659	066764	060400	000000	.WORD	60400,0	RESULT	
13660	066770	000017		.WORD	17	TEST FPS	
13661	066772	000000		.WORD	0	RESULTANT FPS	
13662				110/AC<FSRC, ROUND			
13663	066774	005037	002776	CLR	00FLAG	NO INTERRUPT	
13664	067000	004767	000234	JSR	R7,M,F SUB	DO TEST	
13665	067004	060000	000000	.WORD	60000,0	ACO	
13666	067010	140377	177776	.WORD	140377,177776	FSRC	
13667	067014	160177	177776	.WORD	160177,177776	RESULT	
13668	067020	000017		.WORD	17	TEST FPS	
13669	067022	000010		.WORD	10	RESULTANT FPS	
13670				111/AC>0>FSRC, TRUNCATE			
13671	067024	005037	002776	CLR	00FLAG	NO INTERRUPT	
13672	067030	004767	000204	JSR	R7,M,F SUB	DO TEST	
13673	067034	060000	000000	.WORD	60000,0	ACO	
13674	067040	140377	177776	.WORD	140377,177776	FSRC	
13675	067044	160177	177776	.WORD	160177,177776	RESULT	
13676	067050	007547		.WORD	7547	TEST FPS	
13677	067052	007550		.WORD	7550	RESULTANT FPS	
13678				112/UNDERFLOW, NO INTERRUPTS			
13679	067054	012737	000002 002776	MOV	02,00FLAG	NO INTERRUPT	

C4

SEQ 0248

13680	067062	004767	000152		JSR	R7,MLFSUB		IDO TEST
13681	067066	000200	000001		.WORD	200,1		ACO
13682	067072	000200	000001		.WORD	200,1		FSRC
13683	067076	040200	000002		.WORD	40200,0		RESULT
13684	067102	042117			.WORD	42117		TEST FPS
13685	067104	142100			.WORD	142100		RESULT FPS
13686	067106	000012			.WORD	12		FEC
13687								
13688	067110	012737	000001	002776	;13/OVERFLOW, TRAP			
13689	067116	004767	000116		MOV	01,00FLAG		INTERRUPT
13690	067122	177777	177777		JSR	R7,MLFSUB		IDO TEST
13691	067126	040300	000000		.WORD	177777,-1		ACO
13692	067132	100077	177777		.WORD	40300,0		FSRC
13693	067136	001117			.WORD	100077,-1		RESULT
13694	067140	101116			.WORD	1117		TEST FPS
13695	067142	000010			.WORD	101116		RESULT FPS
13696					.WORD	10		FEC
13697	067144	012737	000002	002776	;14/OVERFLOW NO TRAP			
13698	067152	004767	000062		MOV	02,00FLAG		NO INTERRUPT
13699	067156	077700	000000		JSR	R7,MLFSUB		IDO TEST
13700	067162	077700	000000		.WORD	77700,0		ACO
13701	067166	000000	000000		.WORD	77700,0		FSRC
13702	067172	040117			.WORD	0,0		RESULT
13703	067174	040106			.WORD	40117		TEST FPS
13704	067176	000010			.WORD	40106		RESULT FPS
13705					.WORD	10		FEC
13706	067200	012737	000001	002776	;15/UNDEFINED VARIABLE IN FSRC, TRAP ENABLED			
13707	067206	004767	000026		MOV	01,00FLAG		INTERRUPT
13708	067212	123465	000000		JSR	R7,MLFSUB		IDO TEST
13709	067216	100022	000000		.WORD	123465,0		ACO
13710	067222	123465	000000		.WORD	100022,0		FSRC
13711	067226	004000			.WORD	123465,0		RESULT
13712	067230	104000			.WORD	4000		TEST FPS
13713	067232	000014			.WORD	104000		RESULT FPS
13714					.WORD	14		FEC
13715								
13716	067234	000167	000212		JMP	HOP12		
13717					;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX			
13718					;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX			
13719					;			
13720					ACO			
13721					FSRC			
13722					FPS BEFORE EXECUTION			
13723					FPS AFTER EXECUTION			
13724					(FEC)			
13725					;			
13726					;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX			
13727					;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX			
13728					;			
13729	067240	012605			MLFSUB:	MOV	(SP),R5	RETURN ADDRESS TO USE AS POINTER
13730	067242	012737	067316	000244		MOV	0501,00FVEC	REDIRECT TRAP VECTOR
13731	067250	012702	000200			MOV	0200,R2	SET TO DOUBLE MODE FOR LOAD
13732	067254	170102				LDPS	R2	LOAD FPS
13733	067256	172415				LDD	(R5),ACO	LOAD ACO WITH TEST DATA
13734	067260	010504				MOV	R5,R4	POINT TO FSRC DATA
13735	067262	062704	000004			ADD	04,R4	

13736	067266	016502	000014		MOV	14(R5),R2		;GET TEST FPS
13737	067272	170102			LDFPS	R2		;LOAD TEST FPS
13738								
13739	067274	171014			MULF	(R4),ACO		;*TEST INSTRUCTION
13740	067276	170001			SETF			;WAIT FOR POSSIBLE FPA TRAP.
13741								
13742								
13743	067300	032737	000001	002776				
13744	067306	001420			BIT	01,00FLAG		;VERIFY A NO TRAP CONDITION
13745	067310	104003			BEQ	2		;BRANCH IF GOOD
13746					ERROR	+3		;FPP ERROR
13747	067312	000167	000032		JMP	2		;INSTRUCTION SHOULD HAVE TRAPPED
13748								;REJOIN CODE
13749								
13750	067316	032737	000001	002776				
13751	067324	001003			50:	BIT	01,00FLAG	;SEE IF EXPECTING A TRAP
13752	067326	104003				BNE	51	;BRANCH IF EXPECTING A TRAP
13753						ERROR	+3	;FPP ERROR
13754	067330	000167	000014			JMP	2	;INSTRUCTION WASNT SUPPOSE TO TRAP
13755	067334	012604			51:	MOV	(SP)+,R4	;REJOIN CODE
13756	067336	005726				TST	(SP)+	;SEE IF PC = INSTRUCTION
13757	067340	022704	067276			CMP	01,R4	;CLEAN UP STACK
13758	067344	001401				BEQ	2	
13759	067346	104003				ERROR	+3	;BRANCH IF GOOD COMPARE
13760								;FPP ERROR
13761								;PC WAS INCORRECT
13762								
13763	067350	170203						
13764	067352	012702	000200		1:	STFPS	R3	;SAVE FPS
13765	067356	170102				MOV	0200,R2	;SET FPP TO DOUBLE
13766	067360	012701	003102			LDFPS	R2	
13767	067364	174011				MOV	0RECDST,R1	;POINT TO RECEIVED DATA TABLE
13768	067366	026503	000016			STD	ACO,(R1)	;SAVE ACO RESULT
13769	067372	001401				CMP	16(R5),R3	;VERIFY STATUS
13770	067374	104003				BEQ	3	;BRANCH IF GOOD
13771						ERROR	+3	;FPP ERROR
13772	067376	010504						;BAD FPS
13773	067400	062704	000010		3:	MOV	R5,R4	;POINT TO EXPECTED DATA
13774	067404	004767	044034			ADD	010,R4	
13775	067410	005767	113444		4:	JSR	R7,DATVER	;VERIFY DATA
13776	067414	001401				TST	COUNT	
13777	067416	104003				BEQ	5	;BRANCH IF GOOD
13778						ERROR	+3	;FPP ERROR
13779	067420	005737	002776					;BAD ACO
13780	067424	001002			5:	TST	00FLAG	;SEE IF NEED TO CHECK FEC
13781	067426	000165	000020			BNE	6	;BRANCH IF NEED TO CHECK
13782						JMP	20(R5)	;RETURN FROM TEST
13783	067432	170301						
13784	067434	016504	000020		6:	STST	R1	;SAVE FEC
13785	067440	020401				MOV	20(R5),R4	;GET FEC
13786	067442	001401				CMP	R4,R1	;VERIFY FEC
13787	067444	104003				BEQ	7	;BRANCH IF GOOD
13788						ERROR	+3	;FPP ERROR
13789	067446	000165	000022					;BAD FEC
13790	067452				7:	JMP	22(R5)	;RETURN FROM TEST
13791					HOP1,7:			


```

13792
13793
13794
13795
13796
13797
13798
13799 067452
13800
13801
13802
13803 067452 005037 002776
13804 067456 004767 000554
13805 067462 000100 000300 000000
13806 067470 000000
13807 067472 000411 177777 000000
13808 067500 000001
13809 067502 000000 000000 000000
13810 067510 000000
13811 067512 000200
13812 067514 000204
13813
13814 067516 005037 002776
13815 067522 004767 000510
13816 067526 077777 000000 000000
13817 067534 000000
13818 067536 000000 000000 000000
13819 067544 000000
13820 067546 000000 000000 000000
13821 067554 000000
13822 067556 007700
13823 067560 007704
13824
13825 067562 005037 002776
13826 067566 004767 000444
13827 067572 040200 000000 000000
13828 067600 000000
13829 067602 000277 177777 177777
13830 067610 177777
13831 067612 000277 177777 177777
13832 067620 177777
13833 067622 007717
13834 067624 007700
13835
13836 067626 005037 002776
13837 067632 004767 000400
13838 067636 065500 000000 000000
13839 067644 000001
13840 067646 037577 177777 177777
13841 067654 177776
13842 067656 065077 177777 177777
13843 067664 177777
13844 067666 007717
13845 067670 007700
13846
13847 067672 005037 002776

```

;
 ; TEST MUL0
 ;
 ;
 ;
 ;
 MMULD;
 ;
 ; 1/AC=0
 CLR 00FLAG ;NO INTERRUPT
 JSR R7,MULDSUB ;DO TEST
 .WORD 100,0,0,0 ;ACO
 .WORD 411,-1,0,1 ;FSRC
 .WORD 0,0,0,0 ;RESULT
 .WORD 200 ; TEST FPS
 .WORD 204 ;RESULTANT FPS
 ; 2/FSRC=0
 CLR 00FLAG ;NO INTERRUPT
 JSR R7,MULDSUB ;DO TEST
 .WORD 77777,0,0,0 ;ACO
 .WORD 0,0,0,0 ;FSRC
 .WORD 0,0,0,0 ;RESULT
 .WORD 7700 ; TEST FPS
 .WORD 7704 ;RESULTANT FPS
 ; 3/AC=1
 CLR 00FLAG ;NO INTERRUPT
 JSR R7,MULDSUB ;DO TEST
 .WORD 40200,0,0,0 ;ACO
 .WORD 277,-1,-1,-1 ;FSRC
 .WORD 277,-1,-1,-1 ;RESULT
 .WORD 7717 ; TEST FPS
 .WORD 7700 ;RESULTANT FPS
 ; 4/AC>FSRC>0, TRUNCATE
 CLR 00FLAG ;NO INTERRUPT
 JSR R7,MULDSUB ;DO TEST
 .WORD 65500,0,0,1 ;ACO
 .WORD 37577,1,1,2 ;FSRC
 .WORD 65077,1,-1,-1 ;RESULT
 .WORD 7717 ; TEST FPS
 .WORD 7700 ;RESULTANT FPS
 ; 5/AC<FSRC<0
 CLR 00FLAG ;NO INTERRUPT

CORRADO RDJ11 B CLUSTER MAC111 50(1046) 05 APR 84 16:45 PAGE 252
 CORRADO.P11 05-APR 84 16:45 FLOATING POINT TESTS

SEQ 0251

```

13848 067676 004767 000334 JSR R7,MLDSUB ;DO TEST
13849 067702 137577 177777 .WORD 137577,-1,-1,-2 ;ACO
13850 067710 177776
13851 067712 165400 000000 000000 .WORD 165400,0,0,1 ;FSRC
13852 067720 000001
13853 067722 065000 000000 000000 .WORD 65000,0,0,0 ;RESULT
13854 067730 000000
13855 067732 007717 .WORD 7717 ; TEST FPS
13856 067734 007700 .WORD 7700 ;RESULTANT FPS
13857
;6/AC>FSRC>0
13858 067736 005037 002776 CLR @FLAG ;NO INTERRUPT
13859 067742 004767 000270 JSR R7,MLDSUB ;DO TEST
13860 067746 017500 000000 000000 .WORD 17500,0,0,0 ;ACO
13861 067754 000000
13862 067756 123652 125252 125252 .WORD 123652,125252,125252,125252 ;FSRC
13863 067764 125252
13864 067766 103177 177777 177777 .WORD 103177,-1,-1,-1 ;RESULT
13865 067774 177777
13866 067776 000200 .WORD 200 ; TEST FPS
13867 070000 000210 .WORD 210 ;RESULTANT FPS
13868
;7/UNDERFLOW, TRAPS DISABLED
13869 070002 005037 002776 CLR @FLAG ;NO INTERRUPT
13870 070006 004767 000224 JSR R7,MLDSUB ;DO TEST
13871 070012 000300 000000 000000 .WORD 300,0,0,252 ;ACO
13872 070020 000252
13873 070022 000377 000001 000002 .WORD 377,1,2,3 ;FSRC
13874 070030 000003
13875 070032 000000 000000 000000 .WORD 0,0,0,0 ;RESULT
13876 070040 000000
13877 070042 005740 .WORD 5740 ; TEST FPS
13878 070044 005744 .WORD 5744 ;RESULT FPS
13879
;8/UNDERFLOW, TRAP ENABLED
13880 070046 012737 000001 002776 MOV @1,@FLAG ; INTERRUPT
13881 070054 004767 000156 JSR R7,MLDSUB ;DO TEST
13882 070060 100277 000001 000002 .WORD 100277,1,2,-1 ;ACO
13883 070066 177777
13884 070070 100300 000001 000001 .WORD 100300,1,1,1 ;FSRC
13885 070076 000001
13886 070100 040417 040001 077403 .WORD 40417,40001,77403,0 ;RESULT
13887 070106 000000
13888 070110 002217 .WORD 2217 ; TEST FPS
13889 070112 102200 .WORD 102200 ;RESULT FPS
13890 070114 000012 .WORD 12 ;FEC
13891
;9/OVERFLOW, TRAPS DISABLED
13892 070116 005037 002776 CLR @FLAG ;NO INTERRUPT
13893 070122 004767 000110 JSR R7,MLDSUB ;DO TEST
13894 070126 177777 177777 177777 .WORD 1,1,1,1 ;ACO
13895 070134 177777
13896 070136 040200 177777 177777 .WORD 40200,1,-1,1 ;FSRC
13897 070144 177777
13898 070146 000000 000000 000000 .WORD 0,0,0,0 ;RESULT
13899 070154 000000
13900 070156 006740 .WORD 6740 ; TEST FPS
13901 070160 006746 .WORD 6746 ;RESULT FPS
13902
;10/OVERFLOW, TRAPS ENABLED
13903 070162 012737 000001 002776 MOV @1,@FLAG ; INTERRUPT

```


CORRABO K0J11 B CLUSTER MACY11 30(1046) 05 APR 84 16:45 PAGE 253
 CORRAB.P11 05-APR-84 16:45 FLOATING POINT TESTS

SEQ 0252

```

13904 070170 004767 000042      JSR      R7,MLDSUB      ;DO TEST
13905 070174 157700 025252 025252 .WORD    157700,25252,25252,25252      ;ACO
13906 070202 025252
13907 070204 167700 000000 000000 .WORD    167700,0,0,0      ;FSRC
13908 070212 000000
13909 070214 007420 017777 117777 .WORD    7420,017777,117777,117777      ;RESULT
13910 070222 117777
13911 070224 001240      .WORD    1240      ; TEST FPS
13912 070226 101242      .WORD    101242      ;RESULT FPS
13913 070230 000010      .WORD    10      ;FEC
13914
13915
13916 070232 000167 000212      JMP      HOP13
13917
13918      ;*****
13919      ;*****
13920      ;
13921      ;          ACO
13922      ;          FSRC
13923      ;          FPS BEFORE EXECUTION
13924      ;          FPS AFTER EXECUTION
13925      ;          (FEC)
13926      ;
13927      ;*****
13928      ;*****
13929
13930 070236 012605      MLDSUB: MOV      (SP)+,R5      ; RETURN ADDRESS TO USE AS POINTER
13931 070240 012737 070314 000244      MOV      @50$,@0FPVEC      ;REDIRECT TRAP VECTOR
13932 070246 012702 000200      MOV      @200,R2      ;SET TO DOUBLE MODE FOR LOAD
13933 070252 170102      LDFPS   R2      ;LOAD FPS
13934 070254 172415      LDD      (R5),ACO      ;LOAD ACO WITH TEST DATA
13935 070256 010501      MOV      R5,R1      ;POINT TO FSRC DATA
13936 070260 062701 000010      ADD      @10,R1
13937 070264 016502 000030      MOV      30(R5),R2      ;GET TEST FPS
13938 070270 170102      LDFPS   R2      ;LOAD TEST FPS
13939
13940 070272 171011      MULD      (R1),ACO      ;*TEST INSTRUCTION
13941 070274 170011      SETD      ;WAIT FOR POSSIBLE FPA TRAP.
13942
13943      ; INSTRUCTION DIDNT TRAP
13944 070276 032737 000001 002776      BIT      @1,@0FLAG      ;VERIFY A NO TRAP CONDITION
13945 070304 001420      BEQ      @3      ;BRANCH IF GOOD
13946 070306 104003      ERROR      ;FPP ERROR
13947      ; INSTRUCTION SHOULD HAVE TRAPPED
13948 070310 000167 000032      JMP      @3      ;REJOIN CODE
13949
13950      ; INSTRUCTION TRAPPED
13951 070314 032737 000001 002776      BIT      @1,@0FLAG      ;SEE IF EXPECTING A TRAP
13952 070322 001003      BNE      @1$      ;BRANCH IF EXPECTING A TRAP
13953 070324 104003      ERROR      ;FPP ERROR
13954      ; INSTRUCTION WASNT SUPPOSE TO TRAP
13955 070326 000167 000014      JMP      @3      ;REJOIN CODE
13956 070332 012604      SET:   MOV      (SP)+,R4      ;SEE IF PC = INSTRUCTION
13957 070334 005726      LST      (SP)+      ;CLEAN UP STACK
13958 070336 022704 070274      CMP      @1$,R4
13959 070342 001401      BEQ      @3      ;BRANCH IF GOOD COMPARE

```

```

13960 070344 104003          ERROR      +3          ;FPP ERROR
13961                                     ;PC WAS INCORRECT
13962
13963      ;
13964      ;COMMON CODE FOR TRAP AND NO TRAP
13965      23:      STFPS      R3          ;SAVE FPS
13966      MOV      #200,R2          ;SET FPP TO DOUBLE
13967      LDFPS     R2
13968      MOV      #RECDST,R1        ;POINT TO RECEIVED DATA TABLE
13969      STD      ACO,(R1)          ;SAVE ACO RESULT
13970      CMP      32(R5),R3        ;VERIFY STATUS
13971      BEQ      3$              ;BRANCH IF GOOD
13972      ERROR     +3              ;FPP ERROR
13973                                     ;BAD FPS
13974      3$:      MOV      R5,R4          ;POINT TO EXPECTED DATA
13975      ADD      #20,R4
13976      JSR      R7,DATVER          ;VERIFY DATA
13977      TST      COUNT
13978      BEQ      5$              ;BRANCH IF GOOD
13979      ERROR     +3              ;FPP ERROR
13980                                     ;BAD ACO
13981      5$:      TST      #FLAG        ;SEE IF NEED TO CHECK FEC
13982      BNE      6$              ;BRANCH IF NEED TO CHECK
13983      JMP      34(R5)            ;RETURN FROM TEST
13984      ;VERIFY ERROR STATUS
13985      6$:      STST      R1          ;SAVE FEC
13986      MOV      34(R5),R4        ;GET FEC
13987      CMP      R4,R1            ;VERIFY FEC
13988      BEQ      7$              ;BRANCH IF GOOD
13989      ERROR     +3              ;FPP ERROR
13990                                     ;BAD FEC
13991      7$:      JMP      36(R5)        ;RETURN FROM TEST
13992      HOP13:
13993      ;
13994      ;
13995      ;TEST MODE
13996      ;
13997      ;
13998      ;
13999      ;
14000      070450      MODE:
14001      ;
14002      ;
14003      ;1/AC=0 F SRC=0
14004      CLR      #FLAG            ;NO INTERRUPT
14005      JSR      R7,MODE SUB      ;DO TEST
14006      .WORD    100,0          ;ACO
14007      .WORD    12346, 1
14008      .WORD    0,0            ;F SRC
14009      .WORD    0,0            ;FRACTIONAL RESULT
14010      .WORD    0,0            ;INTEGER RESULT
14011      .WORD    13            ;TEST FPS
14012      .WORD    4              ;RESULTANT FPS
14013      ;2/ F SRC=0
14014      CLR      #FLAG            ;NO INTERRUPT
14015      JSR      R7,MODE SUB      ;DO TEST
14016      .WORD    12356, 1          ;ACO

```

CORDBO RDJ11 B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 255
 CORDAB,P11 05-APR-84 16:45 FLOATING POINT TESTS

SEQ 0254

14016	070520	000000	000000	.WORD	0,0	;FSRC	
14017	070524	000000	000000	.WORD	0,0		;FRACTIONAL RESULT
14018	070530	000000	000000	.WORD	0,0		;INTEGER RESULT
14019	070534	000003		.WORD	3		;TEST FPS
14020	070536	000004		.WORD	4		;RESULTANT FPS
14021				;5/AC=0			
14022	070540	005037	002776	CLR	00FLAG		;NO INTERRUPT
14023	070544	004767	000464	JSR	R7,MDFSUB		;DO TEST
14024	070550	000000	000000	.WORD	0,0		;ACO
14025	070554	177777	177777	.WORD	1,-1		;FSRC
14026	070560	000000	000000	.WORD	0,0		;FRACTIONAL RESULT
14027	070564	000000	000000	.WORD	0,0		;INTEGER RESULT
14028	070570	007500		.WORD	7500		;TEST FPS
14029	070572	007504		.WORD	7504		;RESULT FPS
14030				;4/AC>FSRC>0			
14031	070574	005037	002776	CLR	00FLAG		;NO INTERRUPT
14032	070600	004767	000430	JSR	R7,MDFSUB		;DO TEST
14033	070604	046252	125252	.WORD	46252,125252		;ACO
14034	070610	040300	000000	.WORD	40300,0	;FSRC	
14035	070614	000000	000000	.WORD	0,0		;FRACTIONAL RESULT
14036	070620	046377	177777	.WORD	46377,-1		;INTEGER RESULT
14037	070624	000013		.WORD	13		;TEST FPS
14038	070626	000004		.WORD	4		;RESULTANT FPS
14039				;5/AC>FSRC>0			
14040	070630	005037	002776	CLR	00FLAG		;NO INTERRUPT
14041	070634	004767	000374	JSR	R7,MDFSUB		;DO TEST
14042	070640	077652	125252	.WORD	77652,125252		;ACO
14043	070644	040300	000000	.WORD	40300,0	;FSRC	
14044	070650	000000	000000	.WORD	0,0		;FRACTIONAL RESULT
14045	070654	077777	177777	.WORD	77777,1		;INTEGER RESULT
14046	070660	000000		.WORD	0		;TEST FPS
14047	070662	000004		.WORD	4		;RESULTANT FPS
14048				;6/AC>0<FSRC, INTEGERS			
14050	070664	005037	002776	CLR	00FLAG		;NO INTERRUPT
14051	070670	004767	000340	JSR	R7,MDFSUB		;DO TEST
14052	070674	060600	000000	.WORD	60600,0		;ACO
14053	070700	147400	025700	.WORD	147400,25700		;FSRC
14054	070704	000000	000000	.WORD	0,0		;FRACTIONAL RESULT
14055	070710	170000	025700	.WORD	170000,25700		;INTEGER RESULT
14056	070714	007400		.WORD	7400		;TEST FPS
14057	070716	007404		.WORD	7404		;RESULT FPS
14058				;7/AC<0<FSRC, FRACTIONAL			
14059	070720	005037	002776	CLR	00FLAG		;NO INTERRUPT
14060	070724	004767	000304	JSR	R7,MDFSUB		;DO TEST
14061	070730	100227	177777	.WORD	100227,1		;ACO
14062	070734	044025	025252	.WORD	44025,25252		;FSRC
14063	070740	104061	021251	.WORD	104061,21251		;FRACTIONAL RESULT
14064	070744	000000	000000	.WORD	0,0		;INTEGER RESULT
14065	070750	000000		.WORD	0		;TEST FPS
14066	070752	000010		.WORD	10		;RESULT FPS
14067				;8/AC<0>FSRC, TRUNCATE			
14068	070754	005037	002776	CLR	00FLAG		;NO INTERRUPT
14069	070760	004767	000252	JSR	R7,MDFSUB		;DO TEST
14070	070764	046252	125252	.WORD	46252,125252		;ACO
14071	070770	040300	000000	.WORD	40300,0	;FSRC	

14072	070774	000000	000000	.WORD	0,0	;FRACTIONAL RESULT
14073	071000	046377	177777	.WORD	46377,-1	;INTEGER RESULT
14074	071004	000053		.WORD	53	;TEST FPS
14075	071006	000044		.WORD	44	;RESULT FPS
14076				;9/ROUND INTEGER		
14077	071010	005037	002776	CLR	@FLAG	;NO INTERRUPT
14078	071014	004767	000214	JSR	R7,MDFSUB	;DO TEST
14079	071020	046252	125252	.WORD	46252,125252	;ACO
14080	071024	040300	000000	.WORD	40300,0	;FSRC
14081	071030	000000	000000	.WORD	0,0	;FRACTIONAL RESULT
14082	071034	046377	177777	.WORD	46377,-1	;INTEGER RESULT
14083	071040	000013		.WORD	13	;TEST FPS
14084	071042	000004		.WORD	4	;RESULT FPS
14085				;10/TRUNCATE FRACTION		
14086	071044	005037	002776	CLR	@FLAG	;NO INTERRUPT
14087	071050	004767	000160	JSR	R7,MDFSUB	;DO TEST
14088	071054	040777	177777	.WORD	40777,1	;ACO
14089	071060	040200	000000	.WORD	40200,0	;FSRC
14090	071064	040177	177770	.WORD	40177,177770	;FRACTIONAL RESULT
14091	071070	040740	000000	.WORD	40740,0	;INTEGER RESULT
14092	071074	000000		.WORD	0	;TEST FPS
14093	071076	000000		.WORD	0	;RESULT FPS
14094				;11/ROUND INTEGER		
14095	071100	005037	002776	CLR	@FLAG	;NO INTERRUPT
14096	071104	004767	000124	JSR	R7,MDFSUB	;DO TEST
14097	071110	000000	000000	.WORD	0,0	
14098	071114	000000	000000	.WORD	0,0	;FSRC
14099	071120	000000	000000	.WORD	0,0	;FRACTIONAL RESULT
14100	071124	000000	000000	.WORD	0,0	;INTEGER RESULT
14101	071130	000000		.WORD	0	;TEST FPS
14102	071132	000004		.WORD	4	;RESULT FPS
14103				;12/ROUND FRACTION		
14104	071134	005037	002776	CLR	@FLAG	;NO INTERRUPT
14105	071140	004767	000070	JSR	R7,MDFSUB	;DO TEST
14106	071144	040225	125252	.WORD	40225,125252	;ACO
14107	071150	066652	052525	.WORD	66652,52525	;FSRC
14108	071154	000000	000000	.WORD	0,0	;FRACTIONAL RESULT
14109	071160	066707	025160	.WORD	66707,25160	;INTEGER RESULT
14110	071164	007027		.WORD	7027	;TEST FPS
14111	071166	007004		.WORD	7004	;RESULT FPS
14112				;7/OVERFLOW		
14113	071170	012737	000001	MOV	#1,@FLAG	;INTERRUPT
14114	071176	004767	000032	JSR	R7,MDFSUB	;DO TEST
14115	071202	076000	000000	.WORD	76000,0	;ACO
14116	071206	076000	000000	.WORD	76000,0	;FSRC
14117	071212	000000	000000	.WORD	0,0	;FRACTIONAL RESULT
14118	071216	033600	000000	.WORD	33600,0	;INTEGER RESULT
14119	071222	001000		.WORD	1000	;TEST FPS
14120	071224	101006		.WORD	101006	;RESULT FPS
14121	071226	000010		.WORD	10	;FEC
14122						
14123	071230	000167	000254	JMP	HOP14	
14124						
14125						
14126						
14127						

```

14138      ;          ACO
14139      ;          FSRC
14140      ;          FRACTIONAL RESULT
14141      ;          INTEGER RESULT
14142      ;          FPS BEFORE EXECUTION
14143      ;          FPS AFTER EXECUTION
14144      ;          (FEC)
14145      ;
14146      ;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
14147      ;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
14148      ;
14149      ;
14150      ;
14151      ;
14152      ;
14153      ;
14154      ;
14155      ;
14156      ;
14157      ;
14158      ;
14159      ;
14160      ;
14161      ;
14162      ;
14163      ;
14164      ;
14165      ;
14166      ;
14167      ;
14168      ;
14169      ;
14170      ;
14171      ;
14172      ;
14173      ;
14174      ;
14175      ;
14176      ;
14177      ;
14178      ;
14179      ;
14180      ;
14181      ;
14182      ;
14183      ;

```

Address	Instruction	Comments
14138	MOV (SP)+,R5	RETURN ADDRESS TO USE AS POINTER
14139	MOV #50,\$,R0,FPVEC	REDIRECT TRAP VECTOR
14140	MOV #200,R2	SET TO DOUBLE MODE FOR LOAD
14141	LD FPS R2	LOAD FPS
14142	LDD (R5),ACO	LOAD ACO WITH TEST DATA
14143	MOV #MODGAR,R1	LOAD KNOWN INTO AC1
14144	LDD (R1),AC1	
14145	MOV R5,R1	POINT TO FSRC DATA
14146	ADD #4,R1	
14147	MOV 20(R5),R2	GET TEST FPS
14148	LD FPS R2	LOAD TEST FPS
14149		
14150		
14151	MODE (R1),ACO	TEST INSTRUCTION
14152	1\$: SETF	WAIT FOR POSSIBLE FPA TRAP.
14153		
14154		
14155	INSTRUCTION DIDNT TRAP	
14156	BIT #1,\$,FLAG	VERIFY A NO TRAP CONDITION
14157	BEQ 2\$	BRANCH IF GOOD
14158	ERROR 3\$	FPP ERROR
14159	JMP 2\$	INSTRUCTION SHOULD HAVE TRAPPED
14160		REJOIN CODE
14161		
14162	INSTRUCTION TRAPPED	
14163	50\$: BIT #1,\$,FLAG	SEE IF EXPECTING A TRAP
14164	BNE 51\$	BRANCH IF EXPECTING A TRAP
14165	ERROR 3\$	FPP ERROR
14166	JMP 2\$	INSTRUCTION WASNT SUPPOSE TO TRAP
14167	51\$: MOV (SP)+,R4	REJOIN CODE
14168	TST (SP)+	SEE IF PC - INSTRUCTION
14169	CMP #1\$,R4	CLEAN UP STACK
14170	BEQ 3\$	BRANCH IF GOOD COMPARE
14171	ERROR 3\$	FPP ERROR
14172		PC WAS INCORRECT
14173		
14174		
14175	COMMON CODE FOR TRAP AND NO TRAP	
14176	2\$: STFPS R5	SAVE FPS
14177	MOV #200,R2	SET FPP TO DOUBLE
14178	LD FPS R2	
14179	MOV #RECD,1,R1	POINT TO RECEIVED DATA TABLE
14180	SAVE FRACTIONAL RESULT	
14181	STD ACO,(R1)	SAVE ACO RESULT
14182	CMP 20(R5),R3	VERIFY STATUS
14183	BEQ 3\$	BRANCH IF GOOD
14184	ERROR 3\$	FPP ERROR


```

14240
14241 071564 005037 002776 ;2/AC=0<FSRC
14242 071570 004767 001110 CLR @FLAG ;NO INTERRUPT
14243 071574 000000 000000 000000 JSR R7,MDDSUB ;DO TEST
14244 071602 000000 .WORD 0,0,0,0 ;ACO
14245 071604 001234 177777 000000 .WORD 1234,-1,0,0 ;FSRC
14246 071612 000000 .WORD 0,0,0,0 ;FRACTIONAL RESULT
14247 071614 000000 000000 000000 .WORD 0,0,0,0 ;FRACTIONAL RESULT
14248 071622 000000 .WORD 0,0,0,0 ;INTEGER RESULT
14249 071624 000000 000000 000000 .WORD 0,0,0,0 ;INTEGER RESULT
14250 071632 000000 .WORD 7717 ;TEST FPS
14251 071634 007717 .WORD 7704 ;RESULTANT FPS
14252 071636 007704
14253 ;3/AC>FSRC>0
14254 071640 005037 002776 CLR @FLAG ;NO INTERRUPT
14255 071644 004767 001034 JSR R7,MDDSUB ;DO TEST
14256 071650 056252 125252 125252 .WORD 56252,125252,125252,125250 ;ACO
14257 071656 125250 .WORD 40300,0,0,0 ;FSRC
14258 071660 040300 000000 000000 .WORD 0,0,0,0 ;FRACTIONAL RESULT
14259 071666 000000 .WORD 0,0,0,0 ;FRACTIONAL RESULT
14260 071670 000000 000000 000000 .WORD 0,0,0,0 ;FRACTIONAL RESULT
14261 071676 000000 .WORD 56377,-1,-1,-4 ;INTEGER RESULT
14262 071700 056377 177777 177777 .WORD 56377,-1,-1,-4 ;INTEGER RESULT
14263 071706 177774 .WORD 213 ;TEST FPS
14264 071710 000213 .WORD 204 ;RESULTANT FPS
14265 071712 000204
14266 ;4/AC<0>FSRC
14267 071714 005037 002776 CLR @FLAG ;NO INTERRUPT
14268 071720 004767 000760 JSR R7,MDDSUB ;DO TEST
14269 071724 140240 000000 000000 .WORD 140240,0,0,0 ;ACO
14270 071732 000000 .WORD 63714,146314,133572,167737 ;FSRC
14271 071734 063714 146314 133572 .WORD 63714,146314,133572,167737 ;FSRC
14272 071742 167737 .WORD 0,0,0,0 ;FRACTIONAL RESULT
14273 071744 000000 000000 000000 .WORD 0,0,0,0 ;FRACTIONAL RESULT
14274 071752 000000 .WORD 163777,-1,162531,125726 ;INTEGER RESULT
14275 071754 163777 177777 162531 .WORD 163777,-1,162531,125726 ;INTEGER RESULT
14276 071762 125726 .WORD 210 ;TEST FPS
14277 071764 000210 .WORD 204 ;RESULTANT FPS
14278 071766 000204
14279 ;5/AC>FSRC>0
14280 071770 005037 002776 CLR @FLAG ;NO INTERRUPT
14281 071774 004767 000704 JSR R7,MDDSUB ;DO TEST
14282 072000 056200 000000 000000 .WORD 56200,0,0,1 ;ACO
14283 072006 000001 .WORD 40340,0,0,0 ;FSRC
14284 072010 040340 000000 000000 .WORD 40340,0,0,0 ;FSRC
14285 072016 000000 .WORD 0,0,0,0 ;FRACTIONAL RESULT
14286 072020 000000 000000 000000 .WORD 0,0,0,0 ;FRACTIONAL RESULT
14287 072026 000000 .WORD 56340,0,0,1 ;INTEGER RESULT
14288 072030 056340 000000 000000 .WORD 56340,0,0,1 ;INTEGER RESULT
14289 072036 000001 .WORD 213 ;TEST FPS
14290 072040 000213 .WORD 204 ;RESULTANT FPS
14291 072042 000204
14292 ;6/TRUNCATE
14293 072044 005037 002776 CLR @FLAG ;NO INTERRUPT
14294 072050 004767 000630 JSR R7,MDDSUB ;DO TEST
14295 072054 056252 125252 125252 .WORD 56252,125252,125252,125250 ;ACO

```

N4

COKDABO KDJ11-B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 260
 COKDAB.P11 05-APR-84 16:45 FLOATING POINT TESTS

SEQ 0259

```

14296 072062 125252
14297 072064 040300 000000 000000 .WORD 40300,0,0,0 ;FSRC
14298 072072 000000
14299 072074 000000 000000 000000 .WORD 0,0,0,0 ;FRACTIONAL RESULT
14300 072102 000000
14301 072104 056377 177777 177777 .WORD 56377,-1,-1,-1 ;INTEGER RESULT
14302 072112 177777
14303 072114 000253 .WORD 253 ;TEST FPS
14304 072116 000244 .WORD 244 ;RESULT FPS
14305 ;7/TRUNCATE FRACTION
14306 072120 005037 002776 CLR @#FLAG ;NO INTERRUPT
14307 072124 004767 000554 JSR R7,MDDSUB ;DO TEST
14308 072130 023252 125252 125252 .WORD 23252,125252,125252,125252 ;ACO
14309 072136 125252
14310 072140 040300 000000 000000 .WORD 40300,0,0,0 ;FSRC
14311 072146 000000
14312 072150 023377 177777 177777 .WORD 23377,-1,-1,-1 ;FRACTIONAL RESULT
14313 072156 177777
14314 072160 000000 000000 000000 .WORD 0,0,0,0 ;INTEGER RESULT
14315 072166 000000
14316 072170 000253 .WORD 253 ;TEST FPS
14317 072172 000240 .WORD 240 ;RESULT FPS
14318 ;8/ROUND INTEGER
14319 072174 005037 002776 CLR @#FLAG ;NO INTERRUPT
14320 072200 004767 000500 JSR R7,MDDSUB ;DO TEST
14321 072204 076600 000000 000000 .WORD 76600,0,0,125252 ;ACO
14322 072212 125252
14323 072214 040300 000000 000000 .WORD 40300,0,0,0 ;FSRC
14324 072222 000000
14325 072224 000000 000000 000000 .WORD 0,0,0,0 ;FRACTIONAL RESULT
14326 072232 000000
14327 072234 076700 000000 000000 .WORD 76700,0,0,-1 ;INTEGER RESULT
14328 072242 177777
14329 072244 000200 .WORD 200 ;TEST FPS
14330 072246 000204 .WORD 204 ;RESULT FPS
14331 ;9/ROUND THROUGH FRACTION
14332 072250 005037 002776 CLR @#FLAG ;NO INTERRUPT
14333 072254 004767 000424 JSR R7,MDDSUB ;DO TEST
14334 072260 041525 052525 052525 .WORD 41525,052525,52525,52525 ;ACO
14335 072266 052525
14336 072270 040300 000000 000000 .WORD 40300,0,0,0 ;FSRC
14337 072276 000000
14338 072300 040177 177777 177777 .WORD 40177,-1,-1,177740 ;FRACTIONAL RESULT
14339 072306 177740
14340 072310 041636 000000 000000 .WORD 41636,0,0,0 ;INTEGER RESULT
14341 072316 000000
14342 072320 007700 .WORD 7700 ;TEST FPS
14343 072322 007700 .WORD 7700 ;RESULT FPS
14344 ;10/OVERFLOW, TRAPS ENABLED
14345 072324 012737 000001 002776 MOV #1,@#FLAG ;INTERUPT
14346 072332 004767 000346 JSR R7,MDDSUB ;DO TEST
14347 072336 177777 177777 177777 .WORD -1,1,-1,-1 ;ACO
14348 072344 177777
14349 072346 040400 000000 000000 .WORD 40400,0,0,0 ;FSRC
14350 072354 000000
14351 072356 000000 000000 000000 .WORD 0,0,0,0 ;FRACTIONAL RESULT

```



```

14352 072364 000000
14353 072366 100177 177777 177777 .WORD 100177,-1,-1,-1 ; INTEGER RESULT
14354 072374 177777
14355 072376 007700 .WORD 7700 ; TEST FPS
14356 072400 107706 .WORD 107706 ; RESULT FPS
14357 072402 000010 .WORD 10 ; FEC
14358 ; /INTEGER CHOPPED TO 56 BITS
14359 072404 005037 002776 CLR 00FLAG ; NO INTERRUPT
14360 072410 004767 000270 JSR R7,MDDSUB ; DO TEST
14361 072414 056700 000000 000000 .WORD 56700,0,0,-1 ; ACO
14362 072422 177777
14363 072424 044440 177777 177777 .WORD 44440,1,-1,-1 ; FSRC
14364 072432 177777
14365 072434 000000 000000 000000 .WORD 0,0,0,0 ; FRACTIONAL RESULT
14366 072442 000000
14367 072444 063161 100000 000001 .WORD 63161,100000,1,40775 ; INTEGER RESULT
14368 072452 040775
14369 072454 000200 .WORD 200 ; TEST FPS
14370 072456 000204 .WORD 204 ; RESULT FPS
14371 ; /OVERFLOW, TRAPS DISABLED
14372 072460 012737 000002 002776 MOV 02,00FLAG ; NO INTERRUPT
14373 072466 004767 000212 JSR R7,MDDSUB ; DO TEST
14374 072472 066600 000000 000000 .WORD 66600,0,0,0 ; ACO
14375 072500 000000
14376 072502 066600 000000 000000 .WORD 66600,0,0,0 ; FSRC
14377 072510 000000
14378 072512 000000 000000 000000 .WORD 0,0,0,0 ; FRACTIONAL RESULT
14379 072520 000000
14380 072522 015200 000000 000000 .WORD 15200,0,0,0 ; INTEGER RESULT
14381 072530 000000
14382 072532 047700 .WORD 47700 ; TEST FPS
14383 072534 147706 .WORD 147706 ; RESULT FPS
14384 072536 000010 .WORD 10 ; FEC
14385 ; /UNDERFLOW, TRAPS DISABLED
14386 072540 012737 000002 002776 MOV 02,00FLAG ; NO INTERRUPT
14387 072546 004767 000132 JSR R7,MDDSUB ; DO TEST
14388 072552 100277 000001 000002 .WORD 100277,1,2,1 ; ACO
14389 072560 177777
14390 072562 100300 000001 000001 .WORD 100300,1,1,1 ; FSRC
14391 072570 000001
14392 072572 000000 000000 000000 .WORD 0,0,0,0 ; FRACTIONAL RESULT
14393 072600 000000
14394 072602 000000 000000 000000 .WORD 0,0,0,0 ; INTEGER RESULT
14395 072610 000000
14396 072612 005200 .WORD 5200 ; TEST FPS
14397 072614 005204 .WORD 5204 ; RESULT FPS
14398 072616 000010 .WORD 10 ; FEC
14399 ; /UNDERFLOW TRAPS ENABLED, UV AS RESULT
14400 072620 012737 000001 002776 MOV 01,00FLAG ; INTERRUPT
14401 072626 004767 000052 JSR R7,MDDSUB ; DO TEST
14402 072632 100277 000001 000002 .WORD 100277,1,2,1 ; ACO
14403 072640 177777
14404 072642 100300 000001 000001 .WORD 100300,1,1,1 ; FSRC
14405 072650 000001
14406 072652 040417 040001 077403 .WORD 40417,40001,77403,0 ; FRACTIONAL RESULT
14407 072660 000000
  
```

```

14408 072662 000000 000000 000000 .WORD 0,0,0,0 ;INTEGER RESULT
14409 072670 000000
14410 072672 002200 .WORD 2200 ; TEST FPS
14411 072674 102200 .WORD 102200 ;RESULT FPS
14412 072676 000012 .WORD 12 ;FEC
14413
14414
14415 072700 000167 000244 JMP HOP15 ;JUMP OVER SUBROUTINE
14416 ;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
14417 ;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
14418
14419 ;
14420 ; ACO
14421 ; FSRC
14422 ; FRACTIONAL RESULT
14423 ; INTEGER RESULT
14424 ; FPS BEFORE EXECUTION
14425 ; FPS AFTER EXECUTION
14426 ; (FEC)
14427 ;
14428 ;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
14429 ;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
14430
14431 072704 012605 MDDSUB: MOV (SP)+,R5 ; RETURN ADDRESS TO USE AS POINTER
14432 072706 012737 072770 000244 MOV #50$,R5 ;REDIRECT TRAP VECTOR
14433 072714 012702 000200 MOV #200,R2 ;SET TO DOUBLE MODE FOR LOAD
14434 072722 170102 LDFPS R2 ;LOAD FPS
14435 072724 012701 071500 LDD (R5),ACO ;LOAD ACO WITH TEST DATA
14436 072730 172511 MOV #MODGAR,R1 ;LOAD KNOWN INTO AC1
14437 072732 010501 LDD (R1),AC1
14438 072734 062701 000010 MOV R5,R1 ;POINT TO FSRC DATA
14439 072740 016502 000040 ADD #10,R1
14440 072744 170102 MOV 40(R5),R2 ;GET TEST FPS
14441 ; LDFPS R2 ;LOAD TEST FPS
14442 072746 171411
14443 072750 170011 ; MODD (R1),ACO ;TEST INSTRUCTION
14444 ; SETD ;WAIT FOR POSSIBLE FPA TRAP.
14445 ;
14446 072752 032737 000001 002776 ;INSTRUCTION DIDNT TRAP
14447 072760 001420 BIT #1,00FLAG ;VERIFY A NO TRAP CONDITION
14448 072762 104003 BEQ #3 ;BRANCH IF GOOD
14449 ; ERROR #3 ;FPP ERROR
14450 072764 000167 000032 JMP #3 ;INSTRUCTION SHOULD HAVE TRAPPED
14451 ; ;REJOIN CODE
14452 ;
14453 072770 032737 000001 002776 ;INSTRUCTION TRAPPED
14454 072776 001003 50$: BIT #1,00FLAG ;SEE IF EXPECTING A TRAP
14455 073000 104003 BNE #1$ ;BRANCH IF EXPECTING A TRAP
14456 ; ERROR #3 ;FPP ERROR
14457 073002 000167 000014 ;INSTRUCTION WASNT SUPPOSE TO TRAP
14458 073006 012604 51$: MOV (SP)+,R4 ;REJOIN CODE
14459 073010 005726 TST (SP) ;SEE IF PC + INSTRUCTION
14460 073012 022704 072750 CMP #1$,R4 ;CLEAN UP STACK
14461 073016 001401 BEQ #3 ;
14462 073020 104003 ;BRANCH IF GOOD COMPARE
14463 ; ;FPP ERROR
; ;PC WAS INCORRECT

```

```

14464
14465      ;COMMON CODE FOR TRAP AND NO TRAP
14466      2$:      STEPS      R3          ;SAVE FPS
14467      073022  012702  000200      MOV      0200,R2      ;SET FPS TO DOUBLE
14468      073030  170102      LDFPS     R2
14469      073032  012701  003102      MOV      0RECDST,R1      ;POINT TO RECEIVED DATA TABLE
14470      ;SAVE FRACTIONAL RESULT
14471      073036  174011      STD        AC0,(R1)      ;SAVE AC0 RESULT
14472      073040  026503  000042      CMP      42(R5),R3      ;VERIFY STATUS
14473      073044  001401      BEQ        3$          ;BRANCH IF GOOD
14474      073046  104003      ERROR      3$          ;FPP ERROR
14475      ;BAD FPS
14476      073050  010504      3$:      MOV      R5,R4          ;POINT TO EXPECTED DATA
14477      073052  062704  000020      ADD      020,R4
14478      073056  004767  040400      4$:      JSR      R7,DATVER      ;VERIFY DATA
14479      073062  005767  107772      TST      COUNT
14480      073066  001401      BEQ        5$          ;BRANCH IF GOOD
14481      073070  104003      ERROR      3$          ;FPP ERROR
14482      ;BAD AC0
14483      ;SAVE INTEGER RESULT
14484      073072  174111      5$:      STD        AC1,(R1)      ;SAVE AC1 RESULT
14485      073074  010504      MOV      R5,R1          ;POINT TO EXPECTED
14486      073076  062704  000030      ADD      030,R4
14487      073102  004767  040354      JSR      R7,DATVER      ;VERIFY DATA
14488      073106  005767  107746      TST      COUNT
14489      073112  001401      BEQ        6$          ;BRANCH IF GOOD
14490      073114  104003      ERROR      3$          ;FPP ERROR
14491      ;BAD AC1
14492      073116  005737  002776      6$:      TST      00FLAG      ;SEE IF NEED TO CHECK FEC
14493      073122  001002      BNE        7$          ;BRANCH IF NEED TO CHECK
14494      073124  000165  000044      JMP      44(R5)      ;RETURN FROM TEST
14495      073130  170301      7$:      STST     R1          ;SAVE FEC
14496      073132  016504  000044      MOV      44(R5),R4      ;GET FEC
14497      073136  020401      CMP      R4,R1          ;VERIFY FEC
14498      073140  001401      BEQ        8$          ;BRANCH IF GOOD
14499      073142  104003      ERROR      3$          ;FPP ERROR
14500      ;BAD FEC
14501      073144  000165  000046      8$:      JMP      46(R5)      ;RETURN FROM TEST
14502      ;
14503      073150      HOP15:
14504      ;
14505      ;
14506      ;
14507      ;TEST STCFD
14508      ;
14509      ;
14510      ;
14511      ;
14512      073150      MSFD:
14513      ;
14514      ;
14515      ;
14516      073150  004767  000170      ;1/AC=0
14517      073154  000177  000000  000000      JSR      R7,SEDSUB      ;DO TEST
14518      073162  000001  000000  000000      .WORD    0177,0,0,1      ;AC0
14519      073164  000000  000000  000000      .WORD    0,0,0,0      ;RESULT
  
```

```

14520 073172 000000
14521 073174 047557
14522 073176 047544
14523
14524 073200 004767 000140
14525 073204 077577 177777 177777
14526 073212 177777
14527 073214 077577 177777 000000
14528 073222 000000
14529 073224 007540
14530 073226 007540
14531
14532 073230 004767 000110
14533 073234 100377 177777 100000
14534 073242 000000
14535 073244 100377 177777 000000
14536 073252 000000
14537 073254 007517
14538 073256 007510
14539
14540 073260 004767 000060
14541 073264 100000 000000 000000
14542 073272 000000
14543 073274 000000 000000 000000
14544 073302 000000
14545 073304 007757
14546 073306 007744
14547
14548 073310 004767 000030
14549 073314 125252 125252 125252
14550 073322 125252
14551 073324 125252 125252 000000
14552 073332 000000
14553 073334 000000
14554 073336 000010
14555
14556
14557 073340 000167 000104
14558
14559
14560
14561
14562
14563
14564
14565
14566
14567
14568
14569
14570 073344 012605
14571 073346 012737 073442 000244
14572 073354 012702 000200
14573 073360 170102
14574 073362 172415
14575 073364 012701 003102

;2/AC>0, TRUNCATE
JSR R7,SFDSUB ;DO TEST
;WORD 77577,-1,-1,-1 ;ACO
;WORD 77577,-1,0,0 ;RESULT
;WORD 7540 ;TEST FPS
;WORD 7540 ;RESULT FPS
;3/AC<0, ROUND
JSR R7,SFDSUB ;DO TEST
;WORD 100377,-1,100000,0 ;ACO
;WORD 100377,-1,0,0 ;RESULT
;WORD 7517 ;TEST FPS
;WORD 7510 ;RESULT FPS
;4/AC=-0
JSR R7,SFDSUB ;DO TEST
;WORD 100000,0,0,0 ;ACO
;WORD 0,0,0,0 ;RESULT
;WORD 7757 ;TEST FPS
;WORD 7744 ;RESULT FPS
;5/AC<0
JSR R7,SFDSUB ;DO TEST
;WORD 125252,125252,125252,125252 ;ACO
;WORD 125252,125252,0,0 ;RESULT
;WORD 0 ;TEST FPS
;WORD 10 ;RESULT FPS
;
; JMP HOP16 ;GET OVER SUBROUTINE
;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
;STCFD
; ACO
; RESULT
; FPS BEFORE EXECUTION
; FPS AFTER EXECUTION
;
; THERE CAN BE NO TRAPS WITH THE STCFD INSTRUCTION
;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
;
SFDSUB: MOV (SP)+,R5 ; RETURN ADDRESS TO USE AS POINTER
MOV #50$,R0FPVEC ;REDIRECT TRAP VECTOR
MOV #200,R1 ;SET TO DOUBLE MODE FOR LOAD
LDPS R2 ;LOAD FPS
LDD (R5),ACO ;LOAD ACO WITH TEST DATA
MOV @RECDST,R1 ;POINT TO RESULT AREA

```

```

14576 073370 016502 000020      MOV      20(R5),R2      ;GET TEST FPS
14577 073374 170102      LDFPS     R2              ;LOAD TEST FPS
14578
14579 073376 176011      40$: STCFD     ACO,(R1)      ;*TEST INSTRUCTION
14580
14581      ;
14582      ;INSTRUCTION DIDNT TRAP
14583      ;VERIFY STATUS
14584 073400 170203      2$: STFPS     R3              ;SAVE FPS
14585 073402 016502 000022      MOV      22(R5),R2      ;GET EXPECTED STATUS
14586 073406 020203      CMP       R2,R3          ;VERIFY STATUS
14587 073410 001401      BEQ       3$              ;BRANCH IF GOOD
14588 073412 104003      ERROR     *3          ;FPP ERROR
14589      ;BAD FPS
14590 073414 010504      3$: MOV      R5,R4          ;POINT TO EXPECTED DATA
14591 073416 062704 000010      ADD      #10,R4
14592 073422 004767 040034      4$: JSR      R7,DATVER      ;VERIFY DATA
14593 073426 005767 107426      TST      COUNT
14594 073432 001401      BEQ       5$              ;BRANCH IF GOOD
14595 073434 104003      ERROR     *3          ;FPP ERROR
14596 073436 000165 000024      5$: JMP      24(R5)          ;BAD ACO
14597      ;INSTRUCTION TRAPPED
14598 073442 104003      50$: ERROR    *3          ;RETURN FROM TEST
14599      ;FPP ERROR
14600 073444 000165 000024      JMP      24(R5)          ;INSTRUCTION WASNT SUPPOSE TO TRAP
14601      ;RETURN FROM TEST
14602 073450
14603      HOP16:
14604
14605      ;
14606      ;
14607      ;-----
14608      ;TEST STCDF
14609      ;
14610      ;
14611      ;
14612 073450      MSDF:
14613
14614      ;
14615      ;1/AC=0
14616 073450 005037 002776      CLR      @#FLAG      ;NO INTERRUPT
14617 073454 004767 000220      JSR      R7,SDFSUB      ;DO TEST
14618 073460 000177 000000 000000      .WORD    177,0,0,0      ;ACO
14619 073466 000000
14620 073470 000000 000000      .WORD    0,0          ;RESULT
14621 073474 000200      .WORD    200          ;TEST FPS
14622 073476 000204      .WORD    204          ;RESULT FPS
14623
14624 073500 005037 002776      ;2/AC=-0
14625 073504 004767 000170      CLR      @#FLAG      ;NO INTERRUPT
14626 073510 100000 000300 000200      JSR      R7,SDFSUB      ;DO TEST
14627 073516 000100      .WORD    100000,300,200,100 ;ACO
14628 073520 000000 000000      .WORD    0,0          ;RESULT
14629 073524 007777      .WORD    7777          ;TEST FPS
14630 073526 007744      .WORD    7744          ;RESULT FPS
14631      ;3/AC>0, TRUNCATE

```

Address	Hex	Dec	Hex	Dec	Hex	Dec	Instruction	Comments
14632	073530	005037	002776				CLR	00FLAG ;NO INTERRUPT
14633	073534	004767	000140				JSR	R7,SDFSUB ;DO TEST
14634	073540	055555	055555	177777			.WORD	55555,55555,-1,-1 ;ACO
14635	073546	177777						
14636	073550	055555	055555				.WORD	55555,55555 ;RESULT
14637	073554	000240					.WORD	240 ; TEST FPS
14638	073556	000240					.WORD	240 ;RESULT FPS
14639								
14640	073560	012737	000001	002776			;4/AC<0, ROUND	TO UNDEFINED VARIABLE
14641	073566	004767	000106				MOV	01,00FLAG ; INTERRUPT
14642	073572	077777	177777	100000			JSR	R7,SDFSUB ;DO TEST
14643	073600	000000					.WORD	77777,-1,100000,0 ;ACO
14644	073602	000000	000000				.WORD	0,0 ;RESULT
14645	073606	001200					.WORD	1200 ; TEST FPS
14646	073610	101206					.WORD	101206 ;RESULT FPS
14647								
14648	073612	005037	002776				;5/AC<0, ROUND	
14649	073616	004767	000056				CLR	00FLAG ;NO INTERRUPT
14650	073622	125252	125252	125252			JSR	R7,SDFSUB ;DO TEST
14651	073630	125252					.WORD	125252,125252,125252,125252 ;ACO
14652	073632	125252	125253				.WORD	125252,125253 ;RESULT
14653	073636	007700					.WORD	7700 ; TEST FPS
14654	073640	007710					.WORD	7710 ;RESULT FPS
14655								
14656	073642	012737	000002	002776			;6/ROUND TO UV, TRAPS DISABLED	
14657	073650	004767	000024				MOV	02,00FLAG ; INTERRUPT
14658	073654	077777	177777	177777			JSR	R7,SDFSUB ;DO TEST
14659	073652	000000					.WORD	77777,-1,-1,0 ;ACO
14660	073664	000000	000000				.WORD	0,0 ;RESULT
14661	073670	006700					.WORD	6700 ; TEST FPS
14662	073672	006706					.WORD	6706 ;RESULT FPS
14663								
14664								
14665	073674	000167	000202				JMP	HOP17 ;GET OVER SUBROUTINE
14666								
14667								
14668								
14669								
14670								
14671								
14672								
14673								
14674								
14675								
14676								
14677								
14678								
14679								
14680	073700	012605					SDFSUB: MOV	(SP),R5 ; RETURN ADDRESS TO USE AS POINTER
14681	073702	012737	073756	000244			MOV	0503,00FPVEC ;REDIRECT TRAP VECTOR
14682	073710	012702	000200				MOV	0200,R2 ;SET TO DOUBLE MODE FOR LOAD
14683	07							


```

14688
14689 073732 176011
14690 073734 170327
14691 073736 000000
14692
14693
14694 073740 032737 0000C1 002776
14695 073746 001420
14696 073750 104003
14697
14698 073752 000167 000032
14699
14700
14701 073756 032737 000001 002776
14702 073764 001003
14703 073766 104003
14704
14705 073770 000167 000014
14706 073774 012604
14707 073776 005726
14708 074000 022704 073734
14709 074004 001401
14710 074006 104003
14711
14712
14713
14714
14715 074010 170203
14716 074012 016502 000016
14717 074016 020203
14718 074020 001401
14719 074022 104003
14720
14721 074024 010504
14722 074026 062704 000010
14723 074032 004767 037406
14724 074036 005767 107016
14725 074042 001401
14726 074044 104003
14727
14728 074046 005737 002776
14729 074052 001002
14730 074054 000165 000020
14731
14732 074060 012704 003062
14733 074064 170314
14734 074066 021427 000010
14735 074072 001401
14736 074074 104003
14737
14738 074076 000165 000020
14739
14740 074102
14741
14742
14743

;
40$: STCDF ACO,(R1)
1$: STST (PC),
   .WORD 0
; *TEST INSTRUCTION
; WAIT FOR POSSIBLE FPA TRAP.
; STORE STATUS HERE.
;
; INSTRUCTION DIDNT TRAP
   BIT #1,00FLAG
   BEQ 2$
   ERROR +3
; VERIFY A NO TRAP CONDITION
; BRANCH IF GOOD
; FPP ERROR
; INSTRUCTION SHOULD HAVE TRAPPED
; REJOIN CODE
;
; INSTRUCTION TRAPPED
50$: BIT #1,00FLAG
   BNE 51$
   ERROR +3
; SEE IF EXPECTING A TRAP
; BRANCH IF EXPECTING A TRAP
; FPP ERROR
; INSTRUCTION WASNT SUPPOSE TO TRAP
; REJOIN CODE
; SEE IF PC = INSTRUCTION
; CLEAN UP STACK
;
51$: JMP 2$
   MOV (SP)+,R4
   TST (SP),
   CMP #1$,R4
   BEQ 2$
   ERROR +3
; BRANCH IF GOOD COMPARE
; FPP ERROR
; PC WAS INCORRECT
;
; COMMON CODE FOR TRAP AND NO TRAP
; VERIFY STATUS
2$: STEPS R3
   MOV 16(R5),R2
   CMP R2,R3
   BEQ 3$
   ERROR +3
; SAVE FPS
; GET EXPECTED STATUS
; VERIFY STATUS
; BRANCH IF GOOD
; FPP ERROR
; BAD FPS
; POINT TO EXPECTED DATA
; VERIFY DATA
; BRANCH IF GOOD
; FPP ERROR
; BAD ACO
; SEE IF NEED TO CHECK FEC
; BRANCH IF NEED TO CHECK
; RETURN FROM TEST
;
3$: MOV R5,R4
   ADD #10,R4
4$: JSR R7,DATVER
   TST COUNT
   BEQ 5$
   ERROR +3
; BRANCH IF GOOD
; FPP ERROR
; BAD ACO
; SEE IF NEED TO CHECK FEC
; BRANCH IF NEED TO CHECK
; RETURN FROM TEST
;
; VERIFY FEC
7$: MOV #RECFEC,R4
   STST (R4)
   CMP (R4),#10
   BEQ 8$
   ERROR +3
; POINT TO FEC AREA
; SAVE FEC
; VERIFY FEC FOR OVERFLOW
; BRANCH IF GOOD
; FPP ERROR
; BAD FEC
; RETURN FROM TEST
;
8$: JMP 20(R5)
; HOP17;
;
;
;

```

```

14744 ;TEST STCFD - USING ILLEGAL ACCUMULATOR
14745 ;
14746 ;
14747 ;
14748 ;
14749 074102 MSFDI:
14750 ;
14751 ;
14752 074102 012701 040000 ; MOV #40000,R1 ;DISABLE INTERRUPTS
14753 074106 170101 ; LDFPS R1 ;
14754 074110 176006 ; STCFD ACO,AC6 ;*TEST ILLEGAL INSTRUCTION
14755 074112 170202 ; STEPS R2 ;SAVE STATUS
14756 074114 170303 ; STST R3 ;SAVE FEC
14757 074116 022702 140000 ; CMP #140000,R2 ;VERIFY FER SET
14758 074122 001401 ; BEQ 1$ ;BRANCH IF ERROR RECEIVED
14759 074124 104003 ; ERROR +3 ;FPP ERROR
14760 ; ;FER BIT NOT SET ON ILLEGAL INST.
14761 074126 022703 000002 1$: ; CMP #2,R3 ;VERIFY FEC - FLOATING OP CODE ERROR
14762 074132 001401 ; BEQ 2$ ;BRANCH IF GOOD
14763 074134 104003 ; ERROR +3 ;FPP ERROR
14764 ; ;FEC INCORRECT
14765 074136 2$:
14766 ;
14767 ;
14768 ;
14769 ;
14770 ;
14771 ;TEST CLRD
14772 ;
14773 ;
14774 ;
14775 ;
14776 074136 MCLRD:
14777 ;
14778 ;
14779 074136 012701 003704 ; MOV #TAB47,R1 ;POINT TO DATA
14780 074142 012704 000200 ; MOV #200,R4 ;SET FPP STATUS TO DOUBLE
14781 074146 170104 ; LDFPS R4 ;
14782 074150 172411 ; LDD (R1),ACO ;
14783 074152 012701 003102 ; MOV #RECDST,R1 ;POINT TO DATA BUFFER
14784 074156 174011 ; STD ACO,(R1) ;STORE GARBAGE
14785 074160 170411 ; CLRD (R1) ;CLEAR DATA BUFFER
14786 074162 012704 003254 ; MOV #TAB6,R4 ;VERIFY BUFFER =0
14787 074166 004767 037270 ; JSR R7,DATVER ;
14788 074172 005767 106662 ; TST COUNT ;
14789 074176 001401 ; BEQ 1$ ;BRANCH I RECDST = 0
14790 074200 104003 ; ERROR +3 ;FPP ERROR
14791 ; ;RECDST NOT CLEARED
14792 074202 170202 1$: ; STEPS R2 ;SAVE STATUS
14793 074204 020227 000204 ; CMP R2,#204 ;VERIFY STATUS
14794 074210 001401 ; BEQ 2$ ;BRANCH IF GOOD
14795 074212 104003 ; ERROR +3 ;FPP ERROR
14796 ; ;BAD STATUS
14797 074214 2$:
14798 ;
14799 ;

```



```

14800
14801
14802
14803
14804
14805
14806
14807
14808 074214 MCLR1:
14809
14810
14811 074214 012704 040200      MOV      #40200,R4      ;DISABLE INTERRUPTS
14812 074220 170104      LDHPS   R4              ;LOAD STATUS
14813 074222 170406      CLRD    R6              ;*TEST INSTRUCTION WITH ILLEGAL ACC
14814 074224 170203      STEPS   R3              ;SAVE STATUS
14815 074226 170305      STST    R5              ;SAVE FEC
14816 074230 022703 140200      CMP     #140200,R5      ;VERIFY ERROR
14817 074234 001401      BEQ     1$              ;BRANCH IF FER SET
14818 074236 104003      ERROR    +3              ;FPP ERROR
14819
14820 074240 022705 000002      1$:  CMP     #2,R5              ;ERROR IN FPS
14821 074244 001401      BEQ     2$              ;VERIFY FEC *2 OPCODE ERROR
14822 074246 104003      ERROR    +3              ;BRANCH IF GOOD
14823
14824 074250      2$:
14825
14826
14827
14828
14829
14830
14831
14832
14833
14834
14835 074250 MCLR1:
14836
14837
14838 074250 012704 003122      MOV     #TESTLOC,R4      ;POINT R4 TO RAM
14839 074254 012714 147757      MOV     #147757,(R4)      ;SETUP EXPECTED STATUS
14840 074260 012701 003072      MOV     #RRCST,R1      ;SET BUFFER FOR RECEIVED STATUS
14841 074264 012737 074334 000244      MOV     #10$,@#FVFC      ;SETUP TRAP VECTOR
14842 074272 170114      LDHPS   (R4)              ;*TEST INSTRUCTION
14843 074274 170211      STEPS   (R1)              ;*TEST INSTRUCTION
14844 074276 020427 003122      CMP     R4,#TESTLOC      ;VERIFY R4
14845 074302 001401      BEQ     1$              ;BRANCH IF GOOD
14846 074304 104003      ERROR    +3              ;FPP ERROR
14847
14848 074306 020127 003072      1$:  CMP     R1,#RRCST      ;VERIFY R1
14849 074312 001401      BEQ     2$              ;BRANCH IF GOOD
14850 074314 104003      ERROR    +3              ;FPP ERROR
14851
14852 074316 023727 003072 147757      2$:  CMP     @RRCST,#147757      ;VERIFY STATUS
14853 074324 001406      BEQ     3$              ;BRANCH IF GOOD
14854 074326 104003      ERROR    +3              ;FPP ERROR
14855
14856
14857
14858
14859
14860
14861
14862
14863
14864
14865
14866
14867
14868
14869
14870
14871
14872
14873
14874
14875
14876
14877
14878
14879
14880
14881
14882
14883
14884
14885
14886
14887
14888
14889
14890
14891
14892
14893
14894
14895
14896
14897
14898
14899
14900
14901
14902
14903
14904
14905
14906
14907
14908
14909
14910
14911
14912
14913
14914
14915
14916
14917
14918
14919
14920
14921
14922
14923
14924
14925
14926
14927
14928
14929
14930
14931
14932
14933
14934
14935
14936
14937
14938
14939
14940
14941
14942
14943
14944
14945
14946
14947
14948
14949
14950
14951
14952
14953
14954
14955
14956
14957
14958
14959
14960
14961
14962
14963
14964
14965
14966
14967
14968
14969
14970
14971
14972
14973
14974
14975
14976
14977
14978
14979
14980
14981
14982
14983
14984
14985
14986
14987
14988
14989
14990
14991
14992
14993
14994
14995
14996
14997
14998
14999

```

```

14856 074330 000167 000006          JMP      3$          ;GET OVER TRAP
14857          ;UNEXPECTED TRAP
14858 074334 012600          10$: MOV      (SP)+,R0          ;SAVE PC
14859 074335 012605          MOV      (SP)+,R5          ;SAVE PS
14860 074340 104003          ERROR    +3          ;FPP ERROR
14861          ;UNEXPECTED TRAP
14862 074342          3$:
14863
14864
14865
14866
14867
14868          ;
14869          ;TEST LDAPS, STEPS MODE 2
14870
14871
14872
14873 074342          MLS2:
14874
14875
14876 074342 012704 003122          MOV      @TSTLOC,R4          ;POINT R4 TO RAM
14877 074346 012714 145557          MOV      @145557,(R4)          ;SETUP EXPECTED STATUS
14878 074352 012701 003072          MOV      @RECST,R1          ;SET BUFFER FOR RECEIVED STATUS
14879 074356 012737 074426 000244  MOV      @10$,@FPVEC          ;SETUP TRAP VECTOR
14880 074364 170124          LDAPS    (R4),          ;*TEST INSTRUCTION
14881 074366 170221          STEPS    (R1),          ;*TEST INSTRUCTION
14882 074370 020427 003124          CMP      R4,@TSTLOC+2          ;VERIFY R4
14883 074374 001401          BEQ      1$          ;BRANCH IF GOOD
14884 074376 104003          ERROR    +3          ;FPP ERROR
14885
14886 074400 020127 003074          1$: CMP      R1,@RECST+2          ;VERIFY R1
14887 074404 001401          BEQ      2$          ;BRANCH IF GOOD
14888 074406 104003          ERROR    +3          ;FPP ERROR
14889          ;BAD R1
14890 074410 023727 003072 145557 2$: CMP      @@RECST,@145557          ;VERIFY STATUS
14891 074416 001406          BEQ      3$          ;BRANCH IF GOOD
14892 074420 104003          ERROR    +3          ;FPP ERROR
14893          ;BAD STATUS
14894 074422 000167 000006          JMP      3$          ;GET OVER TRAP
14895          ;UNEXPECTED TRAP
14896 074426 012600          10$: MOV      (SP)+,R0          ;SAVE PC
14897 074430 012605          MOV      (SP)+,R5          ;SAVE PS
14898 074432 104003          ERROR    +3          ;FPP ERROR
14899          ;UNEXPECTED TRAP
14900 074434          3$:
14901
14902
14903
14904
14905
14906          ;
14907          ;TEST LDAPS, STEPS MODE 3
14908
14909
14910
14911 074434          MLS3:

```

```

14912
14913
14914 074434 012704 003122      ;      MOV      #TSTLOC,R4      ;POINT R4 TO RAM
14915 074440 012737 003126 003122      MOV      #TSTLOC+4,#TSTLOC      ;TSTLOC= DEFERRED ADDRESS
14916 074446 012737 147501 003126      MOV      #147501,#TSTLOC+4      ;SETUP EXPECTED STATUS
14917 074454 012701 003132      MOV      #TSTLOC+10,R1      ;R1 POINTS TO TSTLOC+10
14918 074460 012737 003072 003132      MOV      #RECS1,#TSTLOC+10      ;SET DEFERRED BUFFER FOR RECEIVED STATUS
14919 074466 012737 074536 000244      MOV      #10,#FPVEC      ;SETUP TRAP VECTOR
14920 074474 170134      LDPS      @R4      ;*TEST INSTRUCTION
14921 074476 170231      STPS      @R1      ;*TEST INSTRUCTION
14922 074500 020427 003124      CMP      R4,#TSTLOC+2      ;VERIFY R4
14923 074504 001401      BEQ      1$      ;BRANCH IF GOOD
14924 074506 104003      ERROR      +3      ;FPP ERROR
14925
14926 074510 020127 003134      1$:      CMP      R1,#TSTLOC+12      ;VERIFY R1
14927 074514 001401      BEQ      2$      ;BRANCH IF GOOD
14928 074516 104003      ERROR      +3      ;FPP ERROR
14929
14930 074520 023727 003072 147501 2$:      CMP      #RECS1,#147501      ;VERIFY STATUS
14931 074526 001406      BEQ      3$      ;BRANCH IF GOOD
14932 074530 104003      ERROR      +3      ;FPP ERROR
14933
14934 074532 000167 000006      JMP      3$      ;BAD STATUS
14935      ;UNEXPECTED TRAP      ;GET OVER TRAP
14936 074536 012600      10$:      MOV      (SP)+,R0      ;SAVE PC
14937 074540 012605      MOV      (SP)+,R5      ;SAVE PS
14938 074542 104003      ERROR      +3      ;FPP ERROR
14939
14940 074544      3$:      ;UNEXPECTED TRAP
14941
14942
14943
14944
14945
14946      ;TEST LDPS, STPS MODE 4
14947
14948
14949
14950
14951 074544      MULS4:
14952
14953
14954 074544 012704 003124      ;      MOV      #TSTLOC+2,R4      ;POINT R4 TO RAM
14955 074550 012737 147757 003122      MOV      #147757,#TSTLOC      ;TSTLOC= STATUS ADDRESS
14956 074556 012701 003074      MOV      #RECS1+2,R1      ;SET BUFFER FOR RECEIVED STATUS
14957 074562 012737 074632 000244      MOV      #10,#FPVEC      ;SETUP TRAP VECTOR
14958 074570 170144      LDPS      @R4      ;*TEST INSTRUCTION
14959 074572 170241      STPS      @R1      ;*TEST INSTRUCTION
14960 074574 020427 003122      CMP      R4,#TSTLOC      ;VERIFY R4
14961 074600 001401      BEQ      1$      ;BRANCH IF GOOD
14962 074602 104003      ERROR      +3      ;FPP ERROR
14963
14964 074604 020127 003072      1$:      CMP      R1,#RECS1      ;VERIFY R1
14965 074610 001401      BEQ      2$      ;BRANCH IF GOOD
14966 074612 104003      ERROR      +3      ;FPP ERROR
14967
14967

```

M5

CORDBO RDJ11 B CLUSTER MACY11 30(1046) 05 APR-84 16:45 PAGE 272
COKDAB.P11 05 APR 84 16:45 FLOATING POINT TESTS

SEQ 0271

```

14968 074614 023727 003072 147757 2$:    CMP    @@RECST,#147757    ;VERIFY STATUS
14969 074622 001406                BEQ     3$              ;BRANCH F GOOD
14970 074624 104003                ERROR    +3              ;FPP ERROR
14971                                ;BAD STATUS\
14972 074626 000167 000006                JMP     3$              ;GET OVER TRAP
14973                                ;UNEXPECTED TRAP
14974 074632 012600                10$:    MOV     (SP)+,R0          ;SAVE PC
14975 074634 012605                MOV     (SP)+,R5          ;SAVE PS
14976 074636 104003                ERROR    +3              ;FPP ERROR
14977                                ;UNEXPECTED TRAP
14978 074640                3$:
14979
14980
14981
14982
14983
14984                                ;-----
14985                                ;TEST LDFPS, STFPS MODE 5
14986
14987
14988
14989 074640                MLS5:
14990
14991
14992 074640 012704 003124                ;
14993 074644 012737 003126 003122        MOV     @TSTLOC+2,R4      ;POINT R4 TO RAM
14994 074652 012737 147501 003126        MOV     @TSTLOC+4,@@TSTLOC    ;TSTLOC= DEFERRED ADDRESS
14995 074660 012701 003134                MOV     @147501,@@TSTLOC+4    ;SETUP EXPECTED STATUS
14996 074664 012737 003072 003132        MOV     @TSTLOC+12,R1      ;R1 POINTS TO 412
14997 074672 012737 074742 000244        MOV     @RECST,@@TSTLOC+10  ;SET DEFERRED BUFFER FOR RECEIVED STATUS
14998 074700 170154                MOV     @10$,@@FPVEC        ;SETUP TRAP VECTOR
14999 074702 170251                LDFPS    @-(R4)          ;*TEST INSTRUCTION
15000 074704 020427 003122                STFPS    @-(R1)          ;*TEST INSTRUCTION
15001 074710 001401                CMP     R4,@TSTLOC        ;VERIFY R4
15002 074712 104003                BEQ     1$              ;BRANCH IF GOOD
15003                                ERROR    +3              ;FPP ERROR
15004 074714 020127 003132                1$:    CMP     R1,@TSTLOC+10    ;VERIFY R1
15005 074720 001401                BEQ     2$              ;BRANCH IF GOOD
15006 074722 104003                ERROR    +3              ;FPP ERROR
15007                                ;BAD R1
15008 074724 023727 003072 147501 2$:    CMP     @@RECST,#147501    ;VERIFY STATUS
15009 074732 001406                BEQ     3$              ;BRANCH F GOOD
15010 074734 104003                ERROR    +3              ;FPP ERROR
15011                                ;BAD STATUS\
15012 074736 000167 000006                JMP     3$              ;GET OVER TRAP
15013                                ;UNEXPECTED TRAP
15014 074742 012600                10$:    MOV     (SP)+,R0          ;SAVE PC
15015 074744 012605                MOV     (SP)+,R5          ;SAVE PS
15016 074746 104003                ERROR    +3              ;FPP ERROR
15017                                ;UNEXPECTED TRAP
15018 074750                3$:
15019
15020
15021
15022
15023

```

N5

COKDABO RDJ11-B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 273
 COKDAB.P11 05-APR-84 16:45 FLOATING POINT TESTS

SEQ 0272

```

15024      ;TEST LDFPS, STFPS MODE 6
15025      ;
15026      ;
15027      ;
15028      ;
15029 074750  M.LS6:
15030
15031      ;
15032 074750 012704 003122      MOV    #TSTLOC,R4      ;POINT R4 TO RAM
15033 074754 012737 140001 003126  MOV    #140001,#TSTLOC+4      ;SETUP EXPECTED STATUS
15034 074762 012701 003232      MOV    #TSTLOC+110,R1      ;R1 WILL POINT TO TESTLOC+10
15035 074766 012737 075042 000244  MOV    #10$,#FPVEC      ;SETUP TRAP VECTOR
15036 074774 170164 000004      LDFPS  4(R4)      ;*TEST INSTRUCTION
15037 075000 170261 177700      STFPS  -100(R1)      ;*TEST INSTRUCTION
15038 075004 020427 003122      CMP     R4,#TSTLOC      ;VERIFY R4
15039 075010 001401      BEQ     1$      ;BRANCH IF GOOD
15040 075012 104003      ERROR    +3      ;FPP ERROR
15041
15042 075014 020127 003232 1$:  CMP     R1,#TSTLOC+110      ;VERIFY R1
15043 075020 001401      BEQ     2$      ;BRANCH IF GOOD
15044 075022 104003      ERROR    +3      ;FPP ERROR
15045      ;BAD R1
15046 075024 023727 003132 140001 2$:  CMP     #TSTLOC+10,#140001      ;VERIFY STATUS
15047 075032 001406      BEQ     3$      ;BRANCH IF GOOD
15048 075034 104003      ERROR    +3      ;FPP ERROR
15049      ;BAD STATUS
15050 075036 000167 000006      JMP     3$      ;GET OVER TRAP
15051      ;UNEXPECTED TRAP
15052 075042 012600 10$:  MOV     (SP)+,R0      ;SAVE PC
15053 075044 012605      MOV     (SP)+,R5      ;SAVE PS
15054 075046 104003      ERROR    +3      ;FPP ERROR
15055      ;UNEXPECTED TRAP
15056 075050 3$:
15057
15058
15059
15060
15061
15062      ;TEST LDFPS, STFPS MODE 7
15063      ;
15064      ;
15065      ;
15066      ;
15067 075050  M.LS7:
15068
15069      ;
15070 075050 012704 003222      MOV    #TSTLOC+100,R4      ;POINT R4 TO RAM
15071 075054 012737 003126 003122  MOV    #TSTLOC+4,#TSTLOC      ;TSTLOC = DEFERRED ADDRESS
15072 075062 012737 145501 003126  MOV    #145501,#TSTLOC+4      ;SETUP EXPECTED STATUS
15073 075070 012701 003032      MOV    #TSTLOC+70,R1      ;R1 POINTS TO TSTLOC+10
15074 075074 012737 003132 003124  MOV    #TSTLOC+10,#TSTLOC+2      ;
15075 075102 012737 075156 000244  MOV    #10$,#FPVEC      ;SETUP TRAP VECTOR
15076 075110 170174 177700      LDFPS  #100(R4)      ;*TEST INSTRUCTION
15077 075114 170271 000072      STFPS  #72(R1)      ;*TEST INSTRUCTION
15078 075120 020427 003222      CMP     R4,#TSTLOC+100      ;VERIFY R4
15079 075124 001401      BEQ     1$      ;BRANCH IF GOOD

```

COKDAH0 KDJ11 B CLUSTER MAC111 30(1046) 05 APR 84 16:45 PAGE 274
COKDAH,P11 05 APR 84 16:45 FLOATING POINT TESTS

SEQ 02/3

```

15080 075126 104003          ERROR 03          IFPP ERROR
15081
15082 075130 020127 003032    14:    CMP      R1,075130-70          IVERIFY R1
15083 075134 001401          BEQ      23          IBRANCH IF GOOD
15084 075136 104003          ERROR 03          IFPP ERROR
15085
15086 075140 023727 003132 145501 23:    CMP      075140+10,0145501 IVERIFY STATUS
15087 075146 001406          BEQ      34          IBRANCH IF GOOD
15088 075150 104003          ERROR 03          IFPP ERROR
15089
15090 075152 000167 000006          JMP      34          ILOAD STATUS
15091          IUNEXPECTED TRAP          IGET OVER TRAP
15092 075156 012600    104:    MOV      (SP)+,R0          ISAVE PC
15093 075160 012605          MOV      (SP)+,R5          ISAVE PS
15094 075162 104003          ERROR 03          IFPP ERROR
15095
15096 075164          34:          IUNEXPECTED TRAP
15097
15098
15099
15100
15101
15102          ITEST LDCID MODE 27
15103
15104
15105
15106
15107 075164          MLCID:
15108
15109
15110 075164 005001          CLR      R1          IINIT R1
15111 075166 012704 007700          MOV      07700,R4          IFPS+DOUBLE, LONG
15112 075172 170104          LDHPS   R4          I
15113 075174 012737 075230 000244          MOV      0103,000PVEC ISETUP WILD TRAP
15114 075202 177027          LDCID  (R7)+,ACO ITEST INSTRUCTION
15115 075204 005201          INC      R1          I
15116 075206 005201          INC      R1          I
15117 075210 005201          INC      R1          I
15118 075212 005201          INC      R1          I
15119 075214 020127 000003          CMP      R1,03          IVERIFY
15120 075220 001406          BEQ      13          IBRANCH IF GOOD
15121 075222 104003          ERROR 03          IFPP ERROR
15122
15123 075224 000167 000006          JMP      13          IINSTRUCTION FAILED
15124 075230 012600    104:    MOV      (SP)+,R0          IJUMP OVER WILD TRAP
15125 075232 012605          MOV      (SP)+,R5          ISAVE PS
15126 075234 104003          ERROR 03          IFPP ERROR
15127
15128 075236 012704 003264    14:    MOV      07AB6A,R4          IWILD TRAP ON INSTRUCTION
15129 075242 012701 003102          MOV      07EC05,R1          IPOINT TO EXPECTED DATA
15130 075246 174011          STD      ACO,(R1)          IPOINT TO DATA BUFFER
15131 075250 004767 036206          JSR      R7,DATVER          IVERIFY DATA
15132 075254 005767 105600          LST      COUNT          I
15133 075260 001401          BEQ      23          IBRANCH IF GOOD DATA
15134 075262 104003          ERROR 03          IFPP ERROR
15135          ILOAD DATA

```


15136	075264			23:		
15137						
15138						
15139						
15140						
15141						
15142						
15143				TEST LDCIF, LDCIF		
15144						
15145						
15146						
15147						
15148	075264			MLCF:		
15149						
15150						
15151				1/INT=0		
15152	075264	004767	000500	JSR R7,LCFSUB	DO TEST	
15153	075270	000000	000000	.WORD 0,0	FSRC	
15154	075274	000000	000000	.WORD 0,0	RESULT	
15155	075300	000000		.WORD 0	TEST FPS	
15156	075302	000004		.WORD 4	RESULT FPS	
15157				2/INT=0,1		
15158	075304	004767	000460	JSR R7,LCFSUB	DO TEST	
15159	075310	000000	177777	.WORD 0,1	FSRC	
15160	075314	000000	000000	.WORD 0,0	RESULT	
15161	075320	007440		.WORD 7440	TEST FPS	
15162	075322	007444		.WORD 7444	RESULT FPS	
15163				3/LONG=0		
15164	075324	004767	000440	JSR R7,LCFSUB	DO TEST	
15165	075330	000000	000000	.WORD 0,0	FSRC	
15166	075334	000000	000000	.WORD 0,0	RESULT	
15167	075340	000100		.WORD 100	TEST FPS	
15168	075342	000104		.WORD 104	RESULT FPS	
15169				4/INT=40000		
15170	075344	004767	000420	JSR R7,LCFSUB	DO TEST	
15171	075350	040000	000000	.WORD 40000,0	FSRC	
15172	075354	043600	000000	.WORD 43600,0	RESULT	
15173	075360	000017		.WORD 17	TEST FPS	
15174	075362	000000		.WORD 0	RESULT FPS	
15175				5/LONG=1		
15176	075364	004767	000400	JSR R7,LCFSUB	DO TEST	
15177	075370	000000	000001	.WORD 0,1	FSRC	
15178	075374	040200	000000	.WORD 40200,0	RESULT	
15179	075400	000117		.WORD 117	TEST FPS	
15180	075402	000100		.WORD 100	RESULT FPS	
15181				6/INT=PATTERN		
15182	075404	004767	000360	JSR R7,LCFSUB	DO TEST	
15183	075410	000252	025252	.WORD 252,252,252	FSRC	
15184	075414	042052	000000	.WORD 42052,0	RESULT	
15185	075420	000000		.WORD 0	TEST FPS	
15186	075422	000000		.WORD 0	RESULT FPS	
15187				7/INT=-40000		
15188	075424	004767	000340	JSR R7,LCFSUB	DO TEST	
15189	075430	140000	000000	.WORD 40000,0	FSRC	
15190	075434	143600	000000	.WORD 143600,0	RESULT	
15191	075440	000007		.WORD 7	TEST FPS	

15192	075442	000010		.WORD	10		;RESULT FPS
15193				;A/INT=1			
15194	075444	004767	000320	JSR	R7,LCFSUB		;DO TEST
15195	075450	177777	000000	.WORD	1.0		;FSRC
15196	075454	140200	000000	.WORD	140200.0		;RESULT
15197	075460	000007		.WORD	7		;TEST FPS
15198	075462	000010		.WORD	10		;RESULT FPS
15199				;9/INT=PATTERN			
15200	075464	004767	000300	JSR	R7,LCFSUB		;DO TEST
15201	075470	125252	125252	.WORD	125252.125252		;FSRC
15202	075474	143652	126000	.WORD	143652.126000		;RESULT
15203	075500	000007		.WORD	7		;TEST FPS
15204	075502	000010		.WORD	10		;RESULT FPS
15205				;10/LONG=40000			
15206	075504	004767	000260	JSR	R7,LCFSUB		;DO TEST
15207	075510	040000	000000	.WORD	40000.0		;FSRC
15208	075514	047600	000000	.WORD	47600.0		;RESULT
15209	075520	000117		.WORD	117		;TEST FPS
15210	075522	000100		.WORD	100		;RESULT FPS
15211				;11/LONG=1			
15212	075524	004767	000240	JSR	R7,LCFSUB		;DO TEST
15213	075530	000000	000001	.WORD	0.1		;FSRC
15214	075534	040200	000000	.WORD	40200.0		;RESULT
15215	075540	007557		.WORD	7557		;TEST FPS
15216	075542	007540		.WORD	7540		;RESULT FPS
15217				;12/LONG=PATTERN			
15218	075544	004767	000220	JSR	R7,LCFSUB		;DO TEST
15219	075550	000000	000252	.WORD	0.252		;FSRC
15220	075554	042052	000000	.WORD	42052.0		;RESULT
15221	075560	007557		.WORD	7557		;TEST FPS
15222	075562	007540		.WORD	7540		;RESULT FPS
15223				;13/LONG=-40000			
15224	075564	004767	000200	JSR	R7,LCFSUB		;DO TEST
15225	075570	140000	000000	.WORD	-40000.0		;FSRC
15226	075574	147600	000000	.WORD	147600.0		;RESULT
15227	075600	000107		.WORD	107		;TEST FPS
15228	075602	000110		.WORD	110		;RESULT FPS
15229				;14/LONG=-1			
15230	075604	004767	000160	JSR	R7,LCFSUB		;DO TEST
15231	075610	177777	177777	.WORD	-1.1		;FSRC
15232	075614	140200	000000	.WORD	140200.0		;RESULT
15233	075620	007500		.WORD	7500		;TEST FPS
15234	075622	007510		.WORD	7510		;RESULT FPS
15235				;15/LONG=PATTERN			
15236	075624	004767	000140	JSR	R7,LCFSUB		;DO TEST
15237	075630	125252	125252	.WORD	125252.125252		;FSRC
15238	075634	147652	125253	.WORD	147652.125253		;RESULT
15239	075640	000105		.WORD	105		;TEST FPS
15240	075642	000110		.WORD	110		;RESULT FPS
15241				;16/LONG=77777.177500			
15242	075644	004767	000120	JSR	R7,LCFSUB		;DO TEST
15243	075650	077777	177500	.WORD	77777.177500		;FSRC
15244	075654	047777	177777	.WORD	47777.177777		;RESULT
15245	075660	000117		.WORD	117		;TEST FPS
15246	075662	000100		.WORD	100		;RESULT FPS
15247				;17/LONG=40000.100			


```

15248 075664 004767 000100      JSR      R7,LCF SUB      ;DO TEST
15249 075670 040000 000100      .WORD    40000,100      ;FSRC
15250 075674 047600 000001      .WORD    47600,1      ;RESULT
15251 075700 007502      .WORD    7502      ; TEST FPS
15252 075702 007500      .WORD    7500      ;RESULT FPS
15253      ;18/LONG=40000,100 TRUNCATE
15254 075704 004767 000060      JSR      R7,LCF SUB      ;DO TEST
15255 075710 040000 000100      .WORD    40000,100      ;FSRC
15256 075714 047600 000000      .WORD    47600,0      ;RESULT
15257 075720 007557      .WORD    7557      ; TEST FPS
15258 075722 007540      .WORD    7540      ;RESULT FPS
15259      ;19/INT= MOST NEGATIVE
15260 075724 004767 000040      JSR      R7,LCF SUB      ;DO TEST
15261 075730 100000 000000      .WORD    100000,0      ;FSRC
15262 075734 144000 000000      .WORD    144000,0      ;RESULT
15263 075740 000007      .WORD    7      ; TEST FPS
15264 075742 000010      .WORD    10      ;RESULT FPS
15265      ;20/LONG= MOST NEGATIVE
15266 075744 004767 000020      JSR      R7,LCF SUB      ;DO TEST
15267 075750 100000 000000      .WORD    100000,0      ;FSRC
15268 075754 150000 000000      .WORD    150000,0      ;RESULT
15269 075760 000107      .WORD    107      ; TEST FPS
15270 075762 000110      .WORD    110      ;RESULT FPS
15271      ;
15272      ;
15273 075764 000167 000112      JMP      HOP18      ;GET OVER SUBROUTINE
15274      ;
15275      ;
15276      ;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
15277      ;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
15278      ;LDCIF, LCLF
15279      ;
15280      ;
15281      ;      FSRC
15282      ;      RESULT
15283      ;      FPS BEFORE EXECUTION
15284      ;      FPS AFTER EXECUTION
15285      ;
15286      ;NO TRAP CAN OCCUR
15287      ;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
15288      ;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
15289      ;
15290 075770 012602      LCF SUB: MOV      (SP),R2      ; RETURN ADDRESS TO USE AS POINTER
15291 075772 012737 076070 000244      MOV      @50$,@FPVEC      ;REDIRECT TRAP VECTOR
15292 076000 012701 003102      MOV      @RECOST,R1      ;POINT TO RESULT AREA
15293 076004 016200 000010      MOV      10(R2),R0      ;GET TEST FPS
15294 076010 170100      LDFPS      R0      ;LOAD TEST FPS
15295 076012 010204      MOV      R2,R4      ;POINT TO TEST DATA
15296 076014 177014      40$: LDCIF      (R4),AC0      ;*TEST INSTRUCTION (ACCORDING TO MODE)
15297      ;
15298      ;
15299 076016 170203      ;VERIFY STATUS
15300 076020 012700 000200      2$: STEPS      R3      ;SAVE FPS
15301 076024 170100      MOV      @P00,R0      ;SET FPS STATUS TO DOUBLE
15302 076026 174011      LDFPS      R0      ;
15303 076030 016200 000012      STD      AC0,(R1)      ;SAVE TEST RESULT INTO RECOST
15304      MOV      12(R2),R0      ;GET EXPECTED STATUS

```

```

15304 076034 020003          CMP      R0,R3          ;VERIFY STATUS
15305 076036 001401          BEQ      3$          ;BRANCH IF GOOD
15306 076040 104003          ERROR    +3          ;FPP ERROR
15307                                     ;BAD FPS
15308 076042 010204          3$:    MOV      R2,R4          ;POINT TO EXPECTED DATA
15309 076044 062704 000004          ADD      R4,R4
15310 076050 004767 035370          4$:    JSR      R7,DATVFR          ;VERIFY DATA
15311 076054 005767 105000          TST      COUNT
15312 076060 001401          BEQ      5$          ;BRANCH IF GOOD
15313 076062 104003          ERROR    +3          ;FPP ERROR
15314                                     ;BAD ACO
15315 076064 000162 000014          5$:    JMP      14(R2)          ;RETURN FROM TEST
15316                                     ;
15317          ;INSTRUCTION TRAPPED
15318 076070 012600          50$:    MOV      (SP)+,R0          ;SAVE PC
15319 076072 012605          MOV      (SP)+,R5          ;SAVE PS
15320 076074 104003          ERROR    +3          ;FPP ERROR
15321                                     ;INSTRUCTION WASNT SUPPOSE TO TRAP
15322 076076 000167 177762          JMP      5$          ;CONTINUE
15323 076102          HOP18:
15324
15325          ;
15326          ;
15327          ;-----
15328          ;TEST LDCID, LDCLD
15329          ;
15330          ;
15331          ;
15332          ;
15333 076102          MLCID:
15334          ;
15335          ;
15336          ;1/LONG=0
15337 076102 004767 000264          JSR      R7,LCDSUB          ;DO TEST
15338 076106 000000 000000          .WORD    0,0          ;FSRC
15339 076112 000000 000000 000000          .WORD    0,0,0,0          ;RESULT
15340 076120 000000
15341 076122 007313          .WORD    7313          ; TEST FPS
15342 076124 007304          .WORD    7304          ;RESULT FPS
15343          ;2/INT=0
15344 076126 004767 000240          JSR      R7,LCDSUB          ;DO TEST
15345 076132 000000 000001          .WORD    0,1          ;FSRC
15346 076136 040200 000000 000000          .WORD    40200,0,0,0          ;RESULT
15347 076144 000000
15348 076146 007757          .WORD    7757          ; TEST FPS
15349 076150 007740          .WORD    7740          ;RESULT FPS
15350          ;3/INT=40000
15351 076152 004767 000214          JSR      R7,LCDSUB          ;DO TEST
15352 076156 040000 177777          .WORD    40000, 1          ;FSRC
15353 076162 043600 000000 000000          .WORD    43600,0,0,0          ;RESULT
15354 076170 000000
15355 076172 007617          .WORD    7617          ; TEST FPS
15356 076174 007600          .WORD    7600          ;RESULT FPS
15357          ;4/INT=40000
15358 076176 004767 000170          JSR      R7,LCDSUB          ;DO TEST
15359 076202 140000 177777          .WORD    40000, 1          ;FSRC

```

Address	Hex	Hex	Hex	Hex	Instruction	Value	Comment
15360	076206	143600	000000	000000	.WORD	143600,0,0,0	;RESULT
15361	076214	000000					
15362	076216	007600			.WORD	7600	; TEST FPS
15363	076220	007610			.WORD	7610	;RESULT FPS
15364					;5/LONG=40000		
15365	076222	004767	000144		JSR	R7,LCDSUB	;DO TEST
15366	076226	040000	000000		.WORD	40000,0	;FSRC
15367	076232	047600	000000	000000	.WORD	47600,0,0,0	;RESULT
15368	076240	000000					
15369	076242	007757			.WORD	7757	; TEST FPS
15370	076244	007740			.WORD	7740	;RESULT FPS
15371					;6/LONG=1		
15372	076246	004767	000120		JSR	R7,LCDSUB	;DO TEST
15373	076252	000000	000001		.WORD	0,1	;FSRC
15374	076256	040200	000000	000000	.WORD	40200,0,0,0	;RESULT
15375	076264	000000					
15376	076266	000300			.WORD	300	; TEST FPS
15377	076270	000300			.WORD	300	;RESULT FPS
15378					;7/LONG=-2		
15379	076272	004767	000074		JSR	R7,LCDSUB	;DO TEST
15380	076276	177777	177776		.WORD	-1,-2	;FSRC
15381	076302	140400	000000	000000	.WORD	140400,0,0,0	;RESULT
15382	076310	000000					
15383	076312	007300			.WORD	7300	; TEST FPS
15384	076314	007310			.WORD	7310	;RESULT FPS
15385					;8/INT=PATTERN		
15386	076316	004767	000050		JSR	R7,LCDSUB	;DO TEST
15387	076322	123456	176543		.WORD	123456,176543	;FSRC
15388	076326	143661	122000	000000	.WORD	143661,122000,0,0	;RESULT
15389	076334	000000					
15390	076336	000200			.WORD	200	; TEST FPS
15391	076340	000210			.WORD	210	;RESULT FPS
15392					;9/LONG=PATTERN		
15393	076342	004767	000024		JSR	R7,LCDSUB	;DO TEST
15394	076346	125252	125252		.WORD	125252,125252	;FSRC
15395	076352	147652	125252	126000	.WORD	147652,125252,126000,0	;RESULT
15396	076360	000000					
15397	076362	000300			.WORD	300	; TEST FPS
15398	076364	000310			.WORD	310	;RESULT FPS
15399							
15400							
15401							
15402	076366	000167	000112		JMP	HOP19	;GET OVER SUBROUTINE
15403							
15404							
15405							
15406							
15407							
15408							
15409							
15410							
15411							
15412							
15413							
15414							
15415							

Address	Op Code	Op 2	Op 3	Op 4	Comments
15416	076372	012602			
15417	076374	012737	076472	000244	
15418	076402	012701	003102		
15419	076406	016200	000014		
15420	076412	170100			
15421	076414	010204			
15422					
15423					
15424	076416	177014			
15425					
15426					
15427	076420	170203			
15428	076422	012700	000200		
15429	076426	170100			
15430	076430	174011			
15431	076432	016200	000016		
15432	076436	020003			
15433	076440	001401			
15434	076442	104003			
15435					
15436	076444	010204			
15437	076446	062704	000004		
15438	076452	004767	035004		
15439	076456	005767	104376		
15440	076462	001401			
15441	076464	104003			
15442					
15443	076466	000162	000020		
15444					
15445					
15446	076472	012600			
15447	076474	012605			
15448	076476	104003			
15449					
15450	076500	000167	177762		
15451					
15452	076504				
15453					
15454					
15455					
15456					
15457					
15458					
15459					
15460					
15461					
15462					
15463	076504				
15464					
15465					
15466					
15467	076504	005037	002776		
15468	076510	004767	001140		
15469	076514	123456	067012	025252	
15470	076520	171717			
15471	076524	000010			

```

15472 076526 142056 067012 025252 .WORD 142056,67012,25252,171717 ;RESULT
15473 076534 171717
15474 076536 007757 .WORD 7757 ;TEST FPS
15475 076540 007750 .WORD 7750 ;RESULT FPS
15476 ;2/EXP=177 - ACO=POS
15477 076542 005037 002776 CLR 00FLAG ;NO INTERRUPTS
15478 076546 004767 001102 JSR R7,LXPSUB ;DO TEST
15479 076552 023456 070123 100000 .WORD 23456,70123,100000,1 ;ACO
15480 076560 000001
15481 076562 000177 .WORD 177 ;EXP
15482 076564 077656 070123 100000 .WORD 77656,70123,100000,1 ;RESULT
15483 076572 000001
15484 076574 007700 .WORD 7700 ;TEST FPS
15485 076576 007700 .WORD 7700 ;RESULT FPS
15486 ;3/EXP=56
15487 076600 005037 002776 CLR 00FLAG ;NO INTERRUPTS
15488 076604 004767 001044 JSR R7,LXPSUB ;DO TEST
15489 076610 055555 044444 033333 .WORD 55555,44444,33333,22222 ;ACO
15490 076616 022222
15491 076620 000056 .WORD 56 ;EXP
15492 076622 053555 044444 033333 .WORD 53555,44444,33333,22222 ;RESULT
15493 076630 022222
15494 076632 007757 .WORD 7757 ;TEST FPS
15495 076634 007740 .WORD 7740 ;RESULT FPS
15496 ;4/EXP=-151, ACO=UV
15497 076636 005037 002776 CLR 00FLAG ;NO INTERRUPTS
15498 076642 004767 001006 JSR R7,LXPSUB ;DO TEST
15499 076646 100077 177777 177777 .WORD 100077,-1,-1,-2 ;ACO
15500 076654 177776
15501 076656 177623 .WORD 155 ;EXP
15502 076660 104677 177777 177777 .WORD 104677,-1,-1,-2 ;RESULT
15503 076666 177776
15504 076670 007757 .WORD 7757 ;TEST FPS
15505 076672 007750 .WORD 7750 ;RESULT FPS
15506 ;5/EXP=-177
15507 076674 005037 002776 CLR 00FLAG ;NO INTERRUPTS
15508 076700 004767 000750 JSR R7,LXPSUB ;DO TEST
15509 076704 000177 177777 177777 .WORD 177,1,-1,-2 ;ACO
15510 076712 177775
15511 076714 177601 .WORD 177 ;EXP
15512 076716 000377 177777 177777 .WORD 377,-1,-1,-2 ;RESULT
15513 076724 177776
15514 076726 007700 .WORD 7700 ;TEST FPS
15515 076730 007700 .WORD 7700 ;RESULT FPS
15516 ;6/EXP=-200, UNDERFLOW
15517 076732 012737 000001 002776 MOV 01,00FLAG ; INTERRUPTS
15518 076740 004767 000710 JSR R7,LXPSUB ;DO TEST
15519 076744 030131 032334 035363 .WORD 30131,32334,35363,73031 ;ACO
15520 076752 073031
15521 076754 177600 .WORD 200 ;EXP
15522 076756 000131 032334 035363 .WORD 131,32334,35363,73031 ;RESULT
15523 076764 073031
15524 076766 007740 .WORD 7740 ;TEST FPS
15525 076770 107744 .WORD 107744 ;RESULT FPS
15526 076772 000012 .WORD 12 ;FEC
15527 ;7/EXP=LARGEST NEGATIVE

```

```

15528 076774 012737 000001 002776      MOV      #1, @R7FLAG      ;EXPECT INTERRUPTS
15529 077002 004767 000646      JSR      R7, LXP SUB      ;DO TEST
15530 077006 000000 000123 000456      .WORD    0, 123, 456, 1      ;ACO
15531 077014 000001
15532 077016 100000      .WORD    100000      ;EXP
15533 077020 040000 000123 000456      .WORD    40000, 123, 456, 1      ;RESULT
15534 077026 000001
15535 077030 002200      .WORD    2200      ;TEST FPS
15536 077032 102200      .WORD    102200      ;RESULT FPS
15537 077034 000012      .WORD    12      ;FEC
15538
;8/EXP=-200, NEG, ACO
15539 077036 012737 000001 002776      MOV      #1, @R7FLAG      ; INTERRUPTS
15540 077044 004767 000604      JSR      R7, LXP SUB      ;DO TEST
15541 077050 111111 100000 100000      .WORD    111111, 100000, 100000, -1      ;ACO
15542 077056 177777
15543 077060 177600      .WORD    -200      ;EXP
15544 077062 100111 100000 100000      .WORD    100111, 100000, 100000, 1      ;RESULT
15545 077070 177777
15546 077072 002217      .WORD    2217      ;TEST FPS
15547 077074 102214      .WORD    102214      ;RESULT FPS
15548 077076 000012      .WORD    12      ;FEC
15549
;9/EXP=-1743, FIU=0
15550 077100 012737 000002 002776      MOV      #2, @R7FLAG      ;NO INTERRUPTS
15551 077106 004767 000542      JSR      R7, LXP SUB      ;DO TEST
15552 077112 123456 012346 012346      .WORD    123456, 12346, 12346, 123      ;ACO
15553 077120 000123
15554 077122 176035      .WORD    -1743      ;EXP
15555 077124 000000 000000 000000      .WORD    0, 0, 0, 0      ;RESULT
15556 077132 000000
15557 077134 005700      .WORD    5700      ;TEST FPS
15558 077136 005704      .WORD    5704      ;RESULT FPS
15559 077140 000012      .WORD    12      ;FEC
15560
;10/EXP=-16616, FID=1
15561 077142 012737 000002 002776      MOV      #2, @R7FLAG      ;NO INTERRUPTS
15562 077150 004767 000500      JSR      R7, LXP SUB      ;DO TEST
15563 077154 000377 123456 065432      .WORD    377, 123456, 65432, 1      ;ACO
15564 077162 000001
15565 077164 161162      .WORD    -16616      ;EXP
15566 077166 074577 123456 065432      .WORD    74577, 123456, 65432, 1      ;RESULT
15567 077174 000001
15568 077176 047700      .WORD    47700      ;TEST FPS
15569 077200 147700      .WORD    147700      ;RESULT FPS
15570 077202 000012      .WORD    12      ;FEC
15571
;11/EXP=177, ACO=UNDEFINED VARIABLE
15572 077204 005037 002776      CLR      @R7FLAG      ;NO INTERRUPTS
15573 077210 004767 000440      JSR      R7, LXP SUB      ;DO TEST
15574 077214 100177 177777 177777      .WORD    100177, 1, -1, 1      ;ACO
15575 077222 177777
15576 077224 000177      .WORD    177      ;EXP
15577 077226 177777 177777 177777      .WORD    -1, 1, 1, -1      ;RESULT
15578 077234 177777
15579 077236 007700      .WORD    7700      ;TEST FPS
15580 077240 007710      .WORD    7710      ;RESULT FPS
15581
;12/EXP=150, FCO=POS
15582 077242 005037 002776      CLR      @R7FLAG      ;NO INTERRUPT
15583 077246 004767 000400      JSR      R7, LXP SUB      ;DO TEST

```


K65

CORRADO KDJ11 8 CLUSTER MACY11 30(1046) 05 APR-84 16:45 PAGE 283
 CORRAD.P11 05-APR-84 16:45 FLOATING POINT TESTS

SEQ 0282

```

15584 077252 000200 000100 000200 .WORD 200,100,200,300 ;ACO
15585 077260 000300
15586 077262 000150 .WORD 150 ;EXP
15587 077264 072000 000100 000200 .WORD 72000,100,200,300 ;RESULT
15588 077272 000300
15589 077274 007717 .WORD 7717 ;TEST FPS
15590 077276 007700 .WORD 7700 ;RESULT FPS
15591
;13/EXP=200, ACO=NEG
15592 077300 012737 000001 002776 MOV #1,00FLAG ;INTERRUPT
15593 077306 004767 000342 JSR R7,LXPSUB ;DO TEST
15594 077312 177777 177777 177777 .WORD -1,-1,-1,-1 ;ACO
15595 077320 177777
15596 077322 000200 .WORD 200 ;EXP
15597 077324 100177 177777 177777 .WORD 100177,-1,-1,-1 ;RESULT
15598 077332 177777
15599 077334 007705 .WORD 7705 ;TEST FPS
15600 077336 107716 .WORD 107716 ;RESULT FPS
15601 077340 000010 .WORD 10 ;FEC
15602
;14/EXP=400, FID
15603 077342 012737 000002 002776 MOV #2,00FLAG ;INTERRUPT
15604 077350 004767 000300 JSR R7,LXPSUB ;DO TEST
15605 077354 000555 177777 177776 .WORD 555,-1,-2,-3 ;ACO
15606 077362 177775
15607 077364 000400 .WORD 400 ;EXP
15608 077366 040155 177777 177776 .WORD 40155,-1,-2,-3 ;RESULT
15609 077374 177775
15610 077376 047700 .WORD 47700 ;TEST FPS
15611 077400 147702 .WORD 147702 ;RESULT FPS
15612 077402 000010 .WORD 10 ;FEC
15613
;15/EXP=11011 FIU=0
15614 077404 012737 000000 002776 MOV #0,00FLAG ;NO INTERRUPT
15615 077412 004767 000236 JSR R7,LXPSUB ;DO TEST
15616 077416 177773 177777 177776 .WORD 177773,-1,-2,-3 ;ACO
15617 077424 177775
15618 077426 011011 .WORD 11011 ;EXP
15619 077430 000000 000000 000000 .WORD 0,0,0,0 ;RESULT
15620 077436 000000
15621 077440 006700 .WORD 6700 ;TEST FPS
15622 077442 006706 .WORD 6706 ;RESULT FPS
15623
;16/EXP=LARGEST POSITIVE
15624 077444 012737 000001 002776 MOV #1,00FLAG ;INTERRUPT
15625 077452 004767 000176 JSR R7,LXPSUB ;DO TEST
15626 077456 123456 000100 000100 .WORD 123456,100,100,200 ;ACO
15627 077464 000200
15628 077466 077777 .WORD 77777 ;EXP
15629 077470 137656 000100 000100 .WORD 137656,100,100,200 ;RESULT
15630 077476 000200
15631 077500 007740 .WORD 7740 ;TEST FPS
15632 077502 107752 .WORD 107752 ;RESULT FPS
15633 077504 000010 .WORD 10 ;FEC
15634
;17/FLOATING
15635 077506 005037 002776 CLR 00FLAG ;NO INTERRUPT
15636 077512 004767 000136 JSR R7,LXPSUB ;DO TEST
15637 077516 123456 023465 000555 .WORD 123456,23465,555,444 ;ACO
15638 077524 000444
15639 077526 000050 .WORD 50 ;EXP

```

```

15640 077530 152056 023465 000555 .WORD 152056,23465,555,444 ;RESULT
15641 077536 000444
15642 077540 007500 .WORD 7500 ; TEST FPS
15643 077542 007510 .WORD 7510 ;RESULT FPS
15644 ;18/FLOATING UNDERFLOW
15645 077544 012737 000001 002776 MOV #1,00FLAG ; INTERRUPT
15646 077552 004767 000076 JSR R7,LXPSUB ;DO TEST
15647 077556 000333 000444 000555 .WORD 333,444,555,666 ;ACO
15648 077564 000666
15649 077566 177600 .WORD -200 ;EXP
15650 077570 000133 000444 000555 .WORD 133,444,555,666 ;RESULT
15651 077576 000666
15652 077600 007500 .WORD 7500 ; TEST FPS
15653 077602 107504 .WORD 107504 ;RESULT FPS
15654 077604 000012 .WORD 12 ;FEC
15655 ;19/FLOATING OVERFLOW
15656 077606 012737 000001 002776 MOV #1,00FLAG ; INTERRUPT
15657 077614 004767 000034 JSR R7,LXPSUB ;DO TEST
15658 077620 012346 000123 000345 .WORD 12346,123,345,456 ;ACO
15659 077626 000456
15660 077630 000400 .WORD 400 ;EXP
15661 077632 040146 000123 000345 .WORD 40146,123,345,456 ;RESULT
15662 077640 000456
15663 077642 007400 .WORD 7400 ; TEST FPS
15664 077644 107402 .WORD 107402 ;RESULT FPS
15665 077646 000010 .WORD 10 ;FEC
15666 ;
15667 ;
15668 077650 000167 000220 JMP HOP20 ;GET OVER SUBROUTINE
15669 ;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
15670 ;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
15671 ;LDEXP
15672 ;
15673 ; ACO
15674 ; EXPONENT
15675 ; RESULT
15676 ; FPS BEFORE EXECUTION
15677 ; FPS AFTER EXECUTION
15678 ; (FEC)
15679 ;
15680 ;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
15681 ;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
15682 077654 012602 LXPSUB: MOV (SP)+,R2 ; RETURN ADDRESS TO USE AS POINTER
15683 077656 012737 077740 000244 MOV #50$,00FPVEC ;REDIRECT TRAP VECTOR
15684 077664 012701 003102 MOV #RECDST,R1 ;POINT TO RESULT AREA
15685 077670 012700 000200 MOV #000,R0 ;SET FPS TO DOUBLE
15686 077674 170100 LDFPS R0 ;
15687 077676 010204 MOV R2,R4 ;POINT TO ACO DATA
15688 077700 172414 LDD (R4),ACO ;LOAD ACO
15689 077702 016203 000022 MOV 20(R2),R0 ;GET TEST FPS
15690 077706 170100 LDFPS R0 ;LOAD TEST FPS
15691 077710 016204 000010 MOV 10(R2),R4 ;POINT TO TEST DATA
15692 ;
15693 077714 176404 40$: LDEXP R4,ACO ;*TEST INSTRUCTION (ACCORDING TO MODE)
15694 077716 170327 1$: SET (PC)+ ;WAIT FOR POSSIBLE FPA TRAP.
15695 077720 000000 .WORD 0 ;STORE STATUS HERE

```



```

15696
15697
15698
15699 077722 032737 000001 002776
15700 077730 001420
15701 077732 104003
15702
15703 077734 000167 000032
15704
15705
15706 077740 032737 000001 002776
15707 077746 001003
15708 077750 104003
15709
15710 077752 000167 000014
15711 077756 012604
15712 077760 005726
15713 077762 022704 077716
15714 077766 001401
15715 077770 104003
15716
15717
15718
15719
15720 077772 170203
15721 077774 012700 000200
15722 100000 170100
15723 100002 174011
15724 100004 016200 000024
15725 100010 020003
15726 100012 001401
15727 100014 104003
15728
15729 100016 010204
15730 100020 062704 000012
15731 100024 004767 033432
15732 100030 005767 103024
15733 100034 001401
15734 100036 104003
15735
15736 100040 005737 002776
15737 100044 001002
15738 100046 000162 000026
15739
15740 100052 012704 003062
15741 100056 170314
15742 100060 021462 000026
15743 100064 001401
15744 100066 104003
15745
15746 100070 000162 000030
15747
15748 100074
15749
15750
15751
;
;
; INSTRUCTION DIDNT TRAP
;
; BIT #1,0#FLAG ;VERIFY A NO TRAP CONDITION
; BEQ 2$ ;BRANCH IF GOOD
; ERROR 13 ;FPP ERROR
; INSTRUCTION SHOULD HAVE TRAPPED
; REJOIN CODE
;
; INSTRUCTION TRAPPED
50$: ;SEE IF EXPECTING A TRAP
; BIT #1,0#FLAG ;BRANCH IF EXPECTING A TRAP
; BNE 51$ ;FPP ERROR
; ERROR 13 ;INSTRUCTION WASNT SUPPOSE TO TRAP
; JMP 2$ ;REJOIN CODE
51$: ;SEE IF PC = INSTRUCTION
; MOV (SP)+,R4 ;CLEAN UP STACK
; TST (SP)+
; CMP #1$,R4
; BEQ 2$ ;BRANCH IF GOOD COMPARE
; ERROR 13 ;FPP ERROR
; PC WAS INCORRECT
;
; COMMON CODE FOR TRAP AND NO TRAP
; VERIFY STATUS
2$: ;SAVE FPS
; STEPS R3 ;SETUP FPS
; MOV #200,R0 ;FPS=200
; LD FPS R0 ;GET RESULT
; STD ACO,(R1) ;GET EXPECTED STATUS
; MOV 24(R2),R0 ;VERIFY STATUS
; CMP R0,R3 ;BRANCH IF GOOD
; BEQ 3$ ;FPP ERROR
; ERROR 13 ;BAD FPS
3$: ;POINT TO EXPECTED DATA
; MOV R2,R4
; ADD #12,R4
4$: ;VERIFY DATA
; JSR R7,DATVER
; TST COUNT ;BRANCH IF GOOD
; BEQ 5$ ;FPP ERROR
; ERROR 13 ;BAD ACO
5$: ;SEE IF NEED TO CHECK FEC
; TST 0#FLAG ;BRANCH IF NEED TO CHECK
; BNE 7$ ;RETURN FROM TEST
; JMP 26(R2)
; VERIFY FEC
7$: ;POINT TO FEC AREA
; MOV #REC FEC,R4 ;SAVE FEC
; STST (R4) ;VERIFY FEC FOR OVERFLOW
; CMP (R4),26(R2) ;BRANCH IF GOOD
; BEQ 8$ ;FPP ERROR
; ERROR 13 ;BAD FEC
8$: ;RETURN FROM TEST
; JMP 30(R2)
;
; HOP.P0:
;
;

```

N6

COKDAB0 KDJ11 B CLUSTER MACY11 30(1046) 05 APR 84 16:45 PAGE 286
 COKDAB.P11 05-APR-84 16:45 FLOATING POINT TESTS

SEQ 0285

```

15752
15753
15754
15755
15756
15757
15758 100074
15759
15760
15761
15762 100074 005037 002776
15763 100100 004767 000610
15764 100104 000177 000000 000000
15765 100112 000000
15766 100114 000000 177777
15767 100120 007640
15768 100122 007644
15769
15770 100124 005037 002776
15771 100130 004767 000560
15772 100134 100177 177777 177777
15773 100142 177777
15774 100144 000000 000000
15775 100150 007700
15776 100152 007704
15777
15778 100154 005037 002776
15779 100160 004767 000530
15780 100164 020000 000000 000000
15781 100172 000000
15782 100174 000000 000000
15783 100200 000300
15784 100202 000304
15785
15786 100204 005037 002776
15787 100210 004767 000500
15788 100214 140177 177777 000001
15789 100222 000001
15790 100224 000000 000000
15791 100230 007700
15792 100232 007704
15793
15794 100234 005037 002776
15795 100240 004767 000450
15796 100244 047667 075757 157737
15797 100252 167773
15798 100254 055675 173757
15799 100260 007717
15800 100262 007700
15801
15802 100264 005037 002776
15803 100270 004767 000420
15804 100274 046400 000000 000000
15805 100302 000000
15806 100304 001000 000000
15807 100310 007700

;-----
;TEST STCDI, STCDL
;
;
;
MSCD:
;
;1/ACO=0, INT
CLR @FLAG ;NO INTERRUPTS
JSR R7,SCDSUB ;DO TEST
.WORD 0177,0,0,0 ;ACO
.WORD 0,-1 ;RESULT
.WORD 7640 ;TEST FPS
.WORD 7644 ;RESULT FPS
;2/ACO=-0, LONG
CLR @FLAG ;INTERRUPT
JSR R7,SCDSUB ;DO TEST
.WORD 100177,-1,-1,-1 ;ACO
.WORD 0,0 ;RESULT
.WORD 7700 ;TEST FPS
.WORD 7704 ;RESULT FPS
;3/EXP=100, LONG
CLR @FLAG ;NO INTERRUPT
JSR R7,SCDSUB ;DO TEST
.WORD 20000,0,0,0 ;ACO
.WORD 0,0 ;RESULT
.WORD 300 ;TEST FPS
.WORD 304 ;RESULT FPS
;4/EXP=200 BASED 0, INT, ROUND
CLR @FLAG ;INTERRUPT
JSR R7,SCDSUB ;DO TEST
.WORD 140177,177777,1,1 ;ACO
.WORD 0,0 ;RESULT
.WORD 7700 ;TEST FPS
.WORD 7704 ;RESULT FPS
;5/LONG
CLR @FLAG ;INTERRUPT
JSR R7,SCDSUB ;DO TEST
.WORD 47667,75757,157737,167773 ;ACO
.WORD 55675,173757 ;RESULT
.WORD 7717 ;TEST FPS
.WORD 7700 ;RESULT FPS
;6/LONG, EXP=2**32
CLR @FLAG ;NO INTERRUPT
JSR R7,SCDSUB ;DO TEST
.WORD 46400,0,0,0 ;ACO
.WORD 1000,0 ;RESULT
.WORD 7700 ;TEST FPS

```

15808	100312	007700			.WORD 7700	RESULT FPS
15809					17/LONG, EXP=2**32	
15810	100314	012737	000001	002776	MOV 01,00FLAG	INTERRUPT
15811	100322	004767	000366		JSR R7,SCDSUB	DO TEST
15812	100326	077607	000000	000000	.WORD 77607,0,0,0	JACO
15813	100334	000000				
15814	100336	000000	000000		.WORD 0,0	RESULT
15815	100342	007700			.WORD 7700	TEST FPS
15816	100344	107705			.WORD 107705	RESULT FPS
15817					18/INT, EXP=2**15	
15818	100346	005037	002776		CIR 00FLAG	NO INTERRUPTS
15819	100352	004767	000336		JSR R7,SCDSUB	DO TEST
15820	100356	043200	000000	000000	.WORD 43200,0,0,0	JACO
15821	100364	000000				
15822	100366	010000	177777		.WORD 10000, 1	RESULT
15823	100372	007600			.WORD 7600	TEST FPS
15824	100374	007600			.WORD 7600	RESULT FPS
15825					19/INT, EXP=2**15	
15826	100376	012737	000001	002776	MOV 01,00FLAG	INTERRUPT
15827	100404	004767	000304		JSR R7,SCDSUB	DO TEST
15828	100410	077777	177777	177777	.WORD 77777, 1, -1, -1	JACO
15829	100416	177777				
15830	100420	000000	177777		.WORD 0, 1	RESULT
15831	100424	007600			.WORD 7600	TEST FPS
15832	100426	107605			.WORD 107605	RESULT FPS
15833					110/INT, EXP=2**15, FID	
15834	100430	012737	000000	002776	MOV 00,00FLAG	NO INTERRUPT
15835	100436	004767	000252		JSR R7,SCDSUB	DO TEST
15836	100442	043300	000000	000000	.WORD 43300,0,0,0	JACO
15837	100450	000000				
15838	100452	000000	014000		.WORD 0,14000	RESULT
15839	100456	047700			.WORD 47700	TEST FPS
15840	100460	047700			.WORD 47700	RESULT FPS
15841					111/INT, EXP=2**15, FIC=0	
15842	100462	012737	000000	002776	MOV 00,00FLAG	NO INTERRUPT
15843	100470	004767	000220		JSR R7,SCDSUB	DO TEST
15844	100474	143300	177777	177777	.WORD 143300, 1, 1, 1	JACO
15845	100502	177777				
15846	100504	177777	163741		.WORD -1,163741	RESULT
15847	100510	007300			.WORD 7300	TEST FPS
15848	100512	007310			.WORD 7310	RESULT FPS
15849					112/LONG, EXP=2**32, FID	
15850	100514	012737	000002	002776	MOV 02,00FLAG	INTERRUPT
15851	100522	004767	000166		JSR R7,SCDSUB	DO TEST
15852	100526	050100	000000	000000	.WORD 50100,0,0,0	JACO
15853	100534	000000				
15854	100536	000000	000000		.WORD 0,0	RESULT
15855	100542	047700			.WORD 47700	TEST FPS
15856	100544	147705			.WORD 147705	RESULT FPS
15857					113/LONG, EXP=2**32, FIC=0	
15858	100546	012737	000000	002776	MOV 00,00FLAG	NO INTERRUPT
15859	100554	004767	000134		JSR R7,SCDSUB	DO TEST
15860	100560	050377	177777	177777	.WORD 50377, 1, 1, 1	JACO
15861	100566	177777				
15862	100570	000000	000000		.WORD 0,0	RESULT
15863	100574	007300			.WORD 7300	TEST FPS

```

15864 100576 002305          .WORD 7305          ;RESULT FPS
15865          ;14/LONG, EXP<0
15866 100600 005037 002776    CLR  00FLAG          ;NO INTERRUPTS
15867 100604 004767 000104    JSR  R7,SCDSUB          ;DO TEST
15868 100610 100200 177777 177777 .WORD 100200,-1,-1,-1    ;ACO
15869 100616 177777
15870 100620 000000 000000    .WORD 0,0          ;RESULT
15871 100624 007757          .WORD 7757          ;TEST FPS
15872 100626 007744          .WORD 7744          ;RESULT FPS
15873          ;15/INT, EXP<0
15874 100630 005037 002776    CLR  00FLAG          ;NO INTERRUPTS
15875 100634 004767 000054    JSR  R7,SCDSUB          ;DO TEST
15876 100640 037700 177777 177777 .WORD 37700,-1,1,2    ;ACO
15877 100646 177776
15878 100650 000000 177777    .WORD 0,1          ;RESULT
15879 100654 007600          .WORD 7600          ;TEST FPS
15880 100656 007604          .WORD 7604          ;RESULT FPS
15881          ;16/INT, EXP=10
15882 100660 005037 002776    CLR  00FLAG          ;NO INTERRUPTS
15883 100664 004767 000024    JSR  R7,SCDSUB          ;DO TEST
15884 100670 004377 177777 177777 .WORD 4377,1,1,-1    ;ACO
15885 100676 177777
15886 100700 000000 177777    .WORD 0,1          ;RESULT
15887 100704 007600          .WORD 7600          ;TEST FPS
15888 100706 007604          .WORD 7604          ;RESULT FPS
15889
15890
15891 100710 000167 000214      JMP  HOP21          ;GET OVER SUBROUTINE
15892      ;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
15893      ;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
15894      ;STCDI, STCDL, STCFI, STCFI
15895      ;
15896      ;          ACO
15897      ;          RESULT
15898      ;          FPS BEFORE EXECUTION
15899      ;          FPS AFTER EXECUTION
15900      ;          (FEC)
15901      ;
15902      ;TRAP ON CONVERSION FAILURE
15903      ;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
15904      ;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
15905      ;
15906 100714 012602          SCDSUB: MOV  (SP),R0          ; RETURN ADDRESS TO USE AS POINTER
15907 100716 012737 101004 000244 MOV  0501,00PVEC          ;REDIRECT TRAP VECTOR
15908 100724 012701 003104      MOV  0RECDST,R1          ;POINT TO RESULT AREA
15909 100730 012711 177777      MOV  0 1,(R1)          ;PRELOAD RECEIVE DATA BUFFER
15910 100734 012741 177777      MOV  0 1,(R1)
15911 100740 012700 000200      MOV  0,00,R0
15912 100744 170100          LDEPS R0          ;SET FPS TO DOUBLE
15913 100746 010204          MOV  R2,R4          ;POINT TO ACO DATA
15914 100750 172414          LDD  (R4),ACO          ;LOAD ACO
15915 100752 016200 000014      MOV  14(R2),R0          ;GET TEST FPS
15916 100756 170100          LDEPS R0          ;LOAD TEST FPS
15917
15918 100760 175411          403: STCDI ACO,(R1)          ;TEST INSTRUCTION ACCORDING TO MODE
15919 100762 170327          13: STST  (PC),          ;WAIT FOR POSSIBLE FPA TRAP
  
```

107

SEQ 0288

```

15920 100764 000000          .WORD 0          ;STORE STATUS HERE.
15921
15922
15923
15924 100766 032737 000001 002776 ;INSTRUCTION DIDNT TRAP
15925 100774 001420          BEQ 2$          ;VERIFY A NO TRAP CONDITION
15926 100776 104003          ERROR 3$        ;BRANCH IF GOOD
15927
15928 101000 000167 000032          JMP 2$        ;FPP ERROR
15929
15930
15931 101004 032737 000001 002776 50$: BIT 01,00FLAG ;INSTRUCTION SHOULD HAVE TRAPPED
15932 101012 001003          BNE 51$        ;REJOIN CODE
15933 101014 104003          ERROR 3$        ;SEE IF EXPECTING A TRAP
15934
15935 101016 000167 000014          JMP 2$        ;BRANCH IF EXPECTING A TRAP
15936 101022 012604          51$: MOV (SP)+,R4 ;FPP ERROR
15937 101024 005726          TST (SP)+        ;INSTRUCTION WASNT SUPPOSE TO TRAP
15938 101026 022704 100762          CMP 01$,R4 ;REJOIN CODE
15939 101032 001401          BEQ 2$          ;SEE IF PC IN INSTRUCTION
15940 101034 104003          ERROR 3$        ;CLEAN UP STACK
15941
15942
15943
15944 ;COMMON CODE FOR TRAP AND NO TRAP
15945 101036 170203          2$: STEPS R3        ;BRANCH IF GOOD COMPARE
15946 101040 016200 000016          MOV 16(R2),R0 ;FPP ERROR
15947 101044 020003          CMP R0,R3        ;PC WAS INCORRECT
15948 101046 001401          BEQ 3$          ;SAVE FPS
15949 101050 104003          ERROR 3$        ;GET EXPECTED STATUS
15950
15951 101052 010204          3$: MOV R2,R4        ;VERIFY STATUS
15952 101054 062704 000010          ADD 010,R4 ;BRANCH IF GOOD
15953 101060 004767 032360          JSR R7,DATVER ;FPP ERROR
15954 101064 005767 101770          TST COUNT ;BAD FPS
15955 101070 001401          BEQ 5$          ;POINT TO EXPECTED DATA
15956 101072 104003          ERROR 3$        ;VERIFY DATA
15957
15958 101074 005737 002776          5$: TST 00FLAG ;BRANCH IF GOOD
15959 101100 001002          BNE 7$          ;FPP ERROR
15960 101102 000162 000020          JMP 20(R2) ;BAD ACO
15961
15962 101106 012704 003062          ;VERIFY FEC ;SEE IF NEED TO CHECK FEC
15963 101112 170314          7$: MOV 0REC FEC,R4 ;BRANCH IF NEED TO CHECK
15964 101114 021427 000006          STST (R4) ;RETURN FROM TEST
15965 101120 001401          CMP (R4),06 ;POINT TO FEC AREA
15966 101122 104003          BEQ 8$          ;SAVE FEC
15967
15968 101124 000162 000020          8$: JMP 20(R2) ;VERIFY FEC FOR OVERFLOW
15969
15970 101130          ;RETURN FROM TEST
15971
15972
15973
15974
15975 ;TEST STCF1, STCF2

```

```

15976
15977
15978
15979
15980 101130 MSCF;
15981
15982
15983 ;1/LONG EXP =30
15984 101130 005037 002776 CLR 00FLAG ;NO INTERRUPTS
15985 101134 004767 177554 JSR R7,SCDSUB ;DO TEST
15986 101140 044541 052525 177777 .WORD 44541,52525,-1,-1 ;ACO
15987 101146 177777
15988 101150 000003 102525 .WORD 3,102525 ;RESULT
15989 101154 007517 .WORD 7517 ;TEST FPS
15990 101156 007500 .WORD 7500 ;RESULT FPS
15991 ;2/INT, EXP<0
15992 101160 005037 002776 CLR 00FLAG ;NO INTERRUPTS
15993 101164 004767 177524 JSR R7,SCDSUB ;DO TEST
15994 101170 002300 177777 177777 .WORD 2300,-1,-1,-1 ;ACO
15995 101176 177777
15996 101200 000000 177777 .WORD 0,-1 ;RESULT
15997 101204 007400 .WORD 7400 ;TEST FPS
15998 101206 007404 .WORD 7404 ;RESULT FPS
15999 ;3/LONG, EXP
16000 101210 012737 000001 002776 MOV 01,00FLAG ;INTERRUPT
16001 101216 004767 177472 JSR R7,SCDSUB ;DO TEST
16002 101222 070000 177777 177777 .WORD 70000,-1,-1,-1 ;ACO
16003 101230 177777
16004 101232 000000 000000 .WORD 0,0 ;RESULT
16005 101236 007540 .WORD 7540 ;TEST FPS
16006 101240 107545 .WORD 107545 ;RESULT FPS
16007 ;4/INT,EXP=5, FIC=0, FID=1
16008 101242 005037 002776 CLR 00FLAG ;NO INTERRUPTS
16009 101246 004767 177442 JSR R7,SCDSUB ;DO TEST
16010 101252 052000 000000 177777 .WORD 52000,0,-1,-1 ;ACO
16011 101260 177777
16012 101262 000000 177777 .WORD 0,1 ;RESULT
16013 101266 047000 .WORD 47000 ;TEST FPS
16014 101270 047005 .WORD 47005 ;RESULT FPS
16015
16016
16017
16018
16019
16020
16021 ;TEST STEXP
16022
16023
16024
16025
16026 101272 MSXP;
16027
16028
16029 ;1/EXP=100
16030 101272 004767 000154 JSR R7,EXPSUB ;DO TEST
16031 101276 020000 000000 000000 .WORD 20000,0,0,0 ;ACO

```



```

16032 101304 000000
16033 101306 177700      .WORD -100      ;RESULT
16034 101310 007740      .WORD 7740      ;TEST FPS
16035 101312 007750      .WORD 7750      ;RESULT FPS
16036      ;2/EXP=201 FLOAT, NEG
16037 101314 004767 000132 JSR R7,SXPSUB      ;DO TEST
16038 101320 140377 177777 .WORD 140377,-1,-1,0 ;ACO
16039 101326 000070
16040 101330 000001      .WORD 1      ;RESULT
16041 101332 007500      .WORD 7500      ;TEST FPS
16042 101334 007500      .WORD 7500      ;RESULT FPS
16043      ;3/EXP=-177
16044 101336 004767 000110 JSR R7,SXPSUB      ;DO TEST
16045 101342 000177 177777 .WORD 177,-1,-1,-1 ;ACO
16046 101350 177777
16047 101352 177600      .WORD 177600      ;RESULT
16048 101354 007700      .WORD 7700      ;TEST FPS
16049 101356 007710      .WORD 7710      ;RESULT FPS
16050      ;4/EXP=100
16051 101360 004767 000066 JSR R7,SXPSUB      ;DO TEST
16052 101364 020000 000000 .WORD 20000,0,-1,-1 ;ACO
16053 101372 177777
16054 101374 177700      .WORD -100      ;RESULT
16055 101376 040200      .WORD 40200      ;TEST FPS
16056 101400 040210      .WORD 40210      ;RESULT FPS
16057      ;5/EXP=200
16058 101402 004767 000044 JSR R7,SXPSUB      ;DO TEST
16059 101406 040000 000000 .WORD 40000,0,0,0 ;ACO
16060 101414 000000
16061 101416 000000      .WORD 0      ;RESULT
16062 101420 007700      .WORD 7700      ;TEST FPS
16063 101422 007704      .WORD 7704      ;RESULT FPS
16064      ;6/EXP=0
16065 101424 004767 000022 JSR R7,SXPSUB      ;DO TEST
16066 101430 000177 177777 .WORD 177,-1,-1,-1 ;ACO
16067 101436 177777
16068 101440 177600      .WORD 177600      ;RESULT
16069 101442 000000      .WORD 0      ;TEST FPS
16070 101444 000010      .WORD 10      ;RESULT FPS
16071
16072
16073 101446 000167 000104      JMP HOP22      ;GET OVER SUBROUTINE
16074      ;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
16075      ;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
16076      ;STEXP
16077      ;
16078      ; ACO
16079      ; EXPONENT RESULT
16080      ; FPS BEFORE EXECUTION
16081      ; FPS AFTER EXECUTION
16082      ;
16083      ; NO TRAPS CAN OCCUR
16084      ;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
16085      ;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
16086      ;
16087 101452 012602 SXPSUB: MOV (SP),R2      ; RETURN ADDRESS TO USE AS POINTER
16087 101454 012737 101544 000244 MOV #50$,@0FPVEC ;REDIRECT TRAP VECTOR

```

```

16088 101462 012701 003102      MOV    #RECDST,R1      ;POINT TO RESULT AREA
16089 101466 012700 000200      MOV    #200,R0      ;SET FPS TO DOUBLE
16090 101472 170100              LDFPS   R0              ;
16091 101474 010204              MOV     R2,R4              ;POINT TO ACO DATA
16092 101476 172414              LDD     (R4),ACO          ;LOAD ACO
16093 101500 016200 000012      MOV     12(R2),R0        ;GET TEST FPS
16094 101504 170100              LDFPS   R0              ;LOAD TEST FPS
16095
16096 101506 175011              ;
16097                                40$:   STEXP   ACO,(R1)      ;*TEST INSTRUCTION(ACCORDING TO MODE)
16098                                ;
16099                                ;VERIFY STATUS
16100 101510 170203              2$:   STEPS   R3              ;SAVE FPS
16101 101512 016200 000014      MOV     14(R2),R0          ;GET EXPECTED STATUS
16102 101516 020003              CMP     R0,R3              ;VERIFY STATUS
16103 101520 001401              BEQ     3$              ;BRANCH IF GOOD
16104 101522 104003              ERROR   +3              ;FPP ERROR
16105                                ;BAD FPS
16106 101524 016204 000010      3$:   MOV     10(R2),R4      ;POINT TO EXPECTED EXPONENT
16107 101530 020437 003102      CMP     R4,#RECDST      ;VERIFY EXPONENT
16108 101534 001401              BEQ     5$              ;BRANCH IF GOOD
16109 101536 104003              ERROR   +3              ;FPP ERROR
16110                                ;BAD ACO
16111 101540 000162 000016      5$:   JMP     16(R2)          ;RETURN FROM TEST
16112                                ;
16113                                ;INSTRUCTION TRAPPED
16114 101544 012600              50$:  MOV     (SP)+,R0      ;SAVE PC
16115 101546 012605              MOV     (SP)+,R5      ;SAVE OLD PS
16116 101550 104003              ERROR   +3              ;FPP ERROR
16117 101552 000167 177762      JMP     5$              ;WILD TRAP DURING STEXP
16118                                ;REJOIN CODE
16119                                ;
16120 101556              HOP22:
16121
16122
16123      .ENABL AMA
16124      .SBTTL TEST - CHECK REGISTER ACCESS
16125      ;CHECK REGISTER ACCESS - THIS TEST WILL VERIFY EACH OF THE THREE
16126      ;CACHE MEMORY SYSTEM REGISTERS CAN BE ACCESSED WITHOUT A TRAP TO
16127      ;LOCATION 4(NON-EXISTANT ADDRESS TRAP) OCCURRING. THE REGISTERS TO
16128      ;BE TESTED ARE:
16129      ;      CACHE CONTROL          17777746
16130      ;      MEMORY SYSTEM ERROR    17777744
16131      ;      HIT/MISS                17777752
16132      ;EACH REGISTER WILL BE ACCESSED WITH A WRITE CYCLE.
16133      ;
16134      ;BGNIST
16135      ;SAVE CONTENT OF LOCATION 4
16136      ;LET LOCATION 4 POINT TO ERROR ROUTINE
16137      ;INITIALIZE R TO TOP OF ADDRESS TABLE
16138      ;DO UNTIL FOR ALL REGISTERS
16139      ;.   TEST REGISTER UNDER TEST
16140      ;.   IF TIMEOUT DID HAPPEN
16141      ;.   .   ERROR
16142      ;.   ENDF
16143      ;ENDDO

```


H7

CORDBO RDJ11 B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 293
 CORDAB.P11 05-APR-84 16:45 TEST - CHECK REGISTER ACCESS

SEQ 0292

```

16144 ;RESTORE CONTENTS OF LOCATION 4
16145 ;EXIT TST
16146 ;
16147 ;ADDRESS TABLE: 17777746
16148 ;                  17777744
16149 ;                  17777752
16150 ;
16151 ;ENDTST
16152 ;ERROR ROUTINE: SET TIMEOUT FLAG
16153 ;
16154 ;
16155 ;*****
16156 101556 000004 TST2: SCOPE
16157 101560 012737 133124 000004 MOV @TOUT, @#4
16158 101566 013737 000004 002762 MOV @#4, SLOC00 ;SAVE CONTENTS OF VECTOR 4
16159 101574 012737 101700 000004 MOV @ERROUT, @#4 ;LET VECTOR 4 POINT TO ERROR ROUTINE
16160 101602 005001 CLR R1 ;CLEAR TIMEOUT FLAG
16161 101604 005737 177746 1$: TST CCR ;READ REGISTER UNDER TEST
16162 101610 005701 TST R1 ;TIMEOUT?
16163 101612 001404 BEQ 2$
16164 101614 012737 177746 001122 MOV @CCR, $BDADR ;STORE LOCATION
16165 101622 104130 ERROR +130 ;TIMEOUT ACCESSING CCR
16166 101624 005001 2$: CLR R1 ;CLEAR TIMEOUT FLAG
16167 101626 005737 177744 TST MSER ;READ MSER
16168 101632 005701 TST R1 ;TIMEOUT ?
16169 101634 001404 BEQ 3$
16170 101636 012737 177744 001122 MOV @MSER, $BDADR ;STORE LOCATION
16171 101644 104130 ERROR +130 ;TIMEOUT ACCESSING MSER
16172 101646 005001 3$: CLR R1 ;CLEAR TIMEOUT FLAG
16173 101650 005737 177752 TST HITMIS ;ACCESS HIT/MISS
16174 101654 005701 TST R1 ;TIMEOUT?
16175 101656 001404 BEQ 4$
16176 101660 012737 177752 001122 MOV @HITMIS, $BDADR ;STORE LOCATION
16177 101666 104130 ERROR +130 ;TIMEOUT ACCESSING HIT/MISS
16178 101670 013737 002762 000004 4$: MOV SLOC00, @#4 ;RESTORE VECTOR 4
16179 101676 000402 BR TST3 ;GO TO NEXT TEST
16180
16181 101700 ERRROUT:
16182 101700 005201 INC R1 ;FLAG TIMEOUT
16183 101702 000002 RTI
16184
16185

```

```

16186 .SBITL TEST - CCR REGISTER BIT TEST
16187 ;CCR REGISTER BIT TEST - THIS TEST WILL VERIFY THAT EACH READ/WRITE BIT OF
16188 ;THE CACHE CONTROL REGISTER CAN BE SET AND CLEARED INDIVIDUALLY AND THAT
16189 ;BITS 15 - 11 AND BIT 8 ARE ALWAYS READ AS ZEROS.
16190 ;
16191 ;BGNTST
16192 ;INITIALIZE GOOD DATA TO #1
16193 ;CLEAR CCR
16194 ;DO UNTIL ALL BITS TESTED
16195 ;. WRITE GOOD DATA TO CCR
16196 ;. READ CCR
16197 ;. IF CCR NOT EQUAL TO GOOD DATA THEN
16198 ;. . IF CCR EQUAL TO ZERO THEN
16199 ;. . . IF GOOD DATA NOT EQUAL TO BIT 15-11 OR BIT 8 THEN
16200 ;. . . . ERROR IN READ/WRITE BITS OF CCR
16201 ;. . . . ENDF
16202 ;. . ENDF
16203 ;. UPDATE TO NEXT BIT
16204 ;ENDDO
16205 ;ENDTST
16206
16207 ;*****
16208 101704 000004 TST3: SCOPE
16209 101706 012701 000001 MOV #1, R1 ;INITIALIZE GOOD DATA TO #1
16210 101712 005037 177746 CLR CCR ;CLEAR CCR
16211 101716 010137 177746 1$: MOV R1, CCR ;WRITE GOOD DATA TO CCR
16212 101722 023701 177746 CMP CCR, R1 ;IF CCR NOT EQUAL GOOD DATA
16213 101726 001407 BEQ 3$ ;THEN
16214 101730 005737 177746 TST CCR ;IF CCR EQUAL TO ZERO
16215 101734 001003 BNE 2$ ;THEN
16216 101736 032701 174400 BIT #174400,R1 ;IF GOOD DATA NE TO BIT 15-11 OR BIT 8
16217 101742 001001 BNE 3$ ;THEN
16218 101744 104004 2$: ERROR .4 ;ERROR IN READ/WRITE BITS OF CCR
16219 101746 006101 3$: ROL R1 ;UPDATE TO NEXT BIT
16220 101750 103362 BCC 1$ ;DO UNTIL ALL BITS TESTED
16221
16222

```

```

16223 .SBTTL TEST - FORCE MISS TEST
16224 ;FORCE MISS TEST - THIS TEST WILL VERIFY THAT ALL REFERENCES MADE
16225 ;WITH EITHER BIT<3> OR BIT<2> OF THE CCR SET CAUSE A CACHE MISS AND
16226 ;LEAVE THE CACHE ENTRY UNCHANGED. FIRST WRITE A TEST ADDRESS WITH
16227 ;BITS<3:2> CLEARED TO ALLOCATE CACHE AND SET A KNOWN DATA PATTERN
16228 ;INTO THE CACHE. THEN SET BIT<2> AND REWRITE THE SAME ADDRESS WITH
16229 ;NEW DATA. NEXT CLEAR BITS<3:2> AND READ THE TEST ADDRESS, THE DATA
16230 ;SHOULD EQUAL TO PATTERN 1. FINALLY READ THE TEST ADDRESS WITH BIT<3>
16231 ;SET, THE DATA SHOULD EQUAL PATTERN 2. A LAST WRITE MUST BE DONE WITH
16232 ;BOTH FORCE BITS CLEARED BECAUSE THE CACHE AND MAIN MEMORY HAVE
16233 ;DIFFERENT DATA.
16234 ;
16235 ;
16236 ;BGNTEST
16237 ;WRITE TEST ADDRESS WITH PATTERN 1
16238 ;SET CCR BITS<3:2> = 0,1
16239 ;WRITE TEST ADDRESS WITH PATTERN 2
16240 ;CLEAR CCR BIT<3>
16241 ;READ TEST ADDRESS
16242 ;SAVE HIT/MISS REGISTER DATA
16243 ;COMPARE RECEIVED DATA TO PATTERN 1
16244 ;IF DATA NOT EQUAL THEN
16245 ;. IF DATA EQUAL TO PATTERN 2 THEN
16246 ;. . IF HIT THEN
16247 ;. . . ERROR FORCE MISS WRITES TO CACHE
16248 ;. . ELSE
16249 ;. . . IF PARITY ERROR INDICATORS EQUAL 0 THEN
16250 ;. . . . ERROR FORCE MISS INVALIDATES CACHE
16251 ;. . . ELSE
16252 ;. . . . IF ALL PARITY INDICATORS SET THEN
16253 ;. . . . . ERROR PARITY ABORT AFTER FORCE MISS
16254 ;. . . . . ENDIF
16255 ;. . . . IF TAG PARITY ERROR THEN
16256 ;. . . . . ERROR TAG PARITY ERROR AFTER FORCE MISS
16257 ;. . . . ELSE
16258 ;. . . . . IF B0 AND B1 ERROR THEN
16259 ;. . . . . . ERROR DATA PARITY ERROR AFTER FORCE MISS
16260 ;. . . . . . ENDIF
16261 ;. . . . . IF B0 PARITY ERROR THEN
16262 ;. . . . . . ERROR LOW BYTE PARITY ERROR
16263 ;. . . . . . ENDIF
16264 ;. . . . . IF B1 PARITY ERROR THEN
16265 ;. . . . . . ERROR HIGH BYTE PARITY ERROR
16266 ;. . . . . . ENDIF
16267 ;. . . . . ENDIF
16268 ;. . . . . ENDIF
16269 ;. . . . . ENDIF
16270 ;. ELSE
16271 ;. . ERROR IN DATA PATH
16272 ;. . ENDIF
16273 ;ENDIF
16274 ;SET CCR<3:2> = 1,0
16275 ;READ TEST ADDRESS
16276 ;SAVE HIT/MISS REGISTER DATA
16277 ;COMPARE RECEIVED DATA TO PATTERN 2
16278 ;IF DATA NOT EQUAL THEN

```

```

16279      IF RECIEVED DATA EQUAL TO PATTERN 1 THEN
16280      .      IF HIT THEN
16281      .      .      ERROR FORCE MISS READS FROM CACHE
16282      .      .      ELSE
16283      .      .      .      ERROR FORCE MISS READS FROM CACHE AND MISS
16284      .      .      .      ENDIF
16285      .      ELSE
16286      .      .      ERROR IN DATA PATH
16287      .      .      ENDIF
16288      .      ENDIF
16289      .      CLEAR CCR<3:2>
16290      .      WRITE TEST ADDRESS
16291      .      ENDIST
16292      .
16293      .      ;*****
16294      TST4:  SCOPE
16295      MOV     @0114, SLOC00      ;SAVE CONTENTS OF VECTOR 114
16296      MOV     @FMPARR,@0114    ;SETUP PARITY VECTOR TO POINT TO HANDLER
16297      CLR     R3                ;CLEAR MSER SAVE LOCATION
16298      CLR     @01STLOC          ;WRITE TEST LOCATION WITH PATTERN 1
16299      MOV     @BIT02, CCR        ;SET CCR BITS<3:2> = 0,1
16300      MOV     @177777,@01STLOC  ;WRITE TEST LOCATION WITH PATTERN 2
16301      MOV     @200, CCR         ;CLEAR CCR BIT 3 AND SET PARITY ABORTS
16302      MOV     @177777,$GDDAT   ;SAVE DATA IN MEMORY
16303      BIC     @1000,BCSR        ;DISABLE HALT ON BREAK
16304      MOV     @01STLOC,R1       ;READ TEST ADDRESS
16305      MOV     @0HITMIS,R2      ;SAVE HIT/MISS DATA
16306      BIS     @1000,BCSR        ;ENABLE HALT ON BREAK
16307      TST     R1                ;COMPARE RECEIVED DATA TO PATTERN 1
16308      BEQ     1$                ;IF DATAS NOT EQUAL THEN
16309      CMP     @177777,R1        ;IF DATA EQUAL TO PATTERN 2
16310      BNE     106$              ;THEN
16311      BIT     @BIT01, R2        ;IF HIT
16312      BEQ     100$              ;THEN
16313      ERROR    +5                ;FORCE MISS WRITES TO CACHE
16314      BR      1$                ;ELSE
16315      BIT     @100340,R3        ;IF PARITY INDICATORS EQUAL 0
16316      BNE     101$              ;THEN
16317      ERROR    +6                ;FORCE MISS WRITE INVALIDATES CACHE
16318      BR      1$                ;ELSE
16319      CMP     @100340,R3        ;IF ALL PARITY INDICATORS SET
16320      BNE     102$              ;THEN
16321      ERROR    +7                ;PARITY ABORT AFTER FORCE MISS
16322      BR      1$                ;ELSE
16323      BIT     @BIT05, R3        ;IF TAG PARITY ERROR
16324      BEQ     103$              ;THEN
16325      ERROR    +10               ;TAG PARITY ERROR AFTER FORCE MISS
16326      BR      1$                ;ELSE
16327      MOV     R3, R4            ;COPY MSER INTO REG 4
16328      BIC     @177477,R4        ;MASK FOR B0 AND B1 DATA
16329      CMP     @300, R4          ;IF B0 AND B1 DATA
16330      BNE     104$              ;THEN
16331      ERROR    +11               ;DATA PARITY ERROR AFTER FORCE MISS
16332      BR      1$                ;ELSE
16333      BIT     @100, R3          ;IF B0 ERROR
16334      BEQ     105$              ;THEN
  
```

Address	Offset	Instruction	Comment
16335	102162	104012	
16336	102164	032703	000200
16337	102170	001403	
16338	102172	104013	
16339	102174	000401	
16340	102176	104014	
16341	102200	005003	
16342	102202	052737	000010 177746
16343	102210	042737	001000 177520
16344	102216	013701	003122
16345	102222	013702	177752
16346	102226	052737	001000 177520
16347	102234	020127	177777
16348	102240	001412	
16349	102242	005701	
16350	102244	001007	
16351	102246	032702	000002
16352	102252	001402	
16353	102254	104015	
16354	102256	000403	
16355	102260	104016	
16356	102262	000401	
16357	102264	104014	
16358	102266	013737	002762 000114
16359	102274	005037	177746
16360	102300	005037	003122
16361	102304	000405	
16362			
16363			
16364	102306	011637	001122
16365	102312	013703	177744
16366	102316	000002	
16367			

```

16368 .SHTTL TEST HIT/MISS REGISTER TEST PART 1
16369 ;HIT/MISS REGISTER TEST PART 1 - THIS TEST WILL VERIFY THAT THE HIT/MISS
16370 ;REGISTER CORRECTLY LOGS HITS AND MISSES.FIRST WRITE A TEST ADDRESS WITH
16371 ;BITS<3:2> CLEARED TO ALLOCATE CACHE AND SET A KNOWN DATA PATTERN INTO
16372 ;THE CACHE. THEN SET BIT<2> AND REWRITE THE SAME ADDRESS WITH NEW DATA.
16373 ;NEXT CLEAR CCR BITS<3:2> AND READ THE TEST ADDRESS, THE HIT/MISS REGISTER
16374 ;SHOULD LOGGED A HIT. FINALLY READ THE TEST ADDRESS PLUS 20000(8) THE
16375 ;HIT/MISS REGISTER SHOULD HAVE LOGGED A MISS.
16376 ;
16377 ;
16378 ;BGN1ST
16379 ;WRITE TEST ADDRESS WITH PATTERN 1
16380 ;SET CCR BITS<3:2> = 0,1
16381 ;WRITE TEST ADDRESS WITH PATTERN 2
16382 ;CLEAR CCR BIT<3>
16383 ;READ TEST ADDRESS
16384 ;IF HIT/MISS REGISTER BIT 1 NOT SET THEN
16385 ;. ERROR IN RECORDING HITS IN HIT/MISS
16386 ;ENDIF
16387 ;READ TEST ADDRESS + 20000(8)
16388 ;IF HIT/MISS REGISTER BIT 1 SET THEN
16389 ;. ERROR IN RECORDING HITS IN HIT/MISS
16390 ;ENDIF
16391 ;ENDTST
16392 ;
16393 ;*****
16394 102320 000004 TST5: SCOPE
16395 102322 013737 000114 002762 MOV @#114, SLOC00 ;SAVE CONTENTS OF VECTOR 114
16396 102330 012737 102460 000114 MOV @HMPARR,@#114 ;SETUP PARITY VECTOR TO POINT TO HANDLER
16397 102336 005037 003122 CLR @#1STLOC ;WRITE TEST LOCATION WITH PATTERN 1
16398 102342 012737 000004 177746 MOV @BIT02,CCR ;SET CCR BITS<3:2> = 0,1
16399 102350 012737 177777 003122 MOV @177777,@#1STLOC ;WRITE TEST LOCATION WITH PATTERN 2
16400 102356 012737 000200 177746 MOV @200,CCR ;CLEAR CCR BIT 3 AND SET PARITY ABORTS
16401 102364 042737 001000 177520 BIC @1000,BCSR ;DISABLE HALT ON BREAK
16402 102372 013701 003122 MOV @#1STLOC,R1 ;READ TEST ADDRESS
16403 102376 013737 177752 110504 MOV HITMIS,RECDAT ;STORE REGISTER
16404 102404 032737 000004 110504 BIT @BIT02,RECDAT ;IF HIT/MISS REGISTER BIT 1 NOT SET
16405 102412 001001 BNE 1$ ;THEN
16406 102414 104045 ERROR +45 ;ERROR IN RECORDING HITS IN HIT/MISS
16407 102416 013701 023122 1$: MOV @#1STLOC+8192,R1 ;READ TEST LOCATION + 20000(8)
16408 102422 013737 177752 110504 MOV HITMIS,RECDAT ;STORE REGISTER
16409 102430 032737 000004 110504 BIT @BIT02,RECDAT ;IF HIT/MISS REGISTER BIT 1 SET
16410 102436 001401 BEQ 2$ ;THEN
16411 102440 104045 ERROR +45 ;ERROR IN RECORDING HITS IN HIT/MISS
16412 102442 013737 002762 000114 2$: MOV SLOC00, @#114 ;RESTORE VECTOR 114
16413 102450 052737 001000 177520 BIS @1000,BCSR ;ENABLE HALT ON BREAK
16414 102456 000403 BR TST6 ;GO TO NEXT TEST
16415
16416 102460 013703 177744 HMPARR: MOV MSER, R3 ;SAVE CONTENTS OF MEMORY SYSTEM ERROR
16417 102464 000002 RTI ;REGISTER AND RETURN
16418

```


N7

COKDABO KDJ11-B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 299
 COKDAB.P11 05-APR-84 16:45 TEST - HIT/MISS REGISTER TEST

SEQ 0298

```

16419 .SBTTL TEST - HIT/MISS REGISTER TEST
16420 ;HIT/MISS REGISTER TEST - THIS TEST WILL VERIFY THAT THE HIT/MISS
16421 ;REGISTER CORRECTLY LOGS CACHE HITS AND MISSES. IT WILL ALSO VERIFY
16422 ;THAT EACH BIT OF THE REGISTER IS UNIQUE. THIS WILL BE DONE BY
16423 ;FLOATING A ZERO THROUGH A FIELD OF ONES. THE ROTATING WILL BE DONE
16424 ;BY EXECUTING A TST @#HM (WHERE HM IS THE ADDRESS OF THE HIT/MISS
16425 ;REGISTER) AT SUCCESSIVE POSITIONS IN A SET OF EIGHT NOPS. THE READ
16426 ;OF THE I/O PAGE ADDRESS OF THE HIT/MISS REGISTER SHOULD CAUSE A
16427 ;MISS TO BE RECORDED.
16428 ;
16429 ;BGN TST
16430 ;INITIALIZE LOOP INDICATOR
16431 ;INITIALIZE EXPECTED DATA
16432 ;DO UNTIL LOOP INDICATOR = SEVEN
16433 ;. PUT BACKGROUND DATA INTO EXECUTION BUFFER
16434 ;. PUT TST INSTRUCTION INTO TEST LOCATION
16435 ;. JUMP TO EXECUTE BUFFER
16436 ;. IF EXPECTED DATA NE RECEIVED DATA THEN
16437 ;. ERROR IN RECORDING HITS IN HIT/MISS
16438 ;.
16439 ;. ENDF
16440 ;. INCREMENT LOOP INDICATOR
16441 ;. UPDATE EXPECTED DATA
16442 ;ENDDO
16443 ;EXIT TST
16444 ;
16445 ;BACKGROUND DATA:NOP
16446 ;. NOP
16447 ;. NOP
16448 ;. NOP
16449 ;. NOP
16450 ;. NOP
16451 ;. NOP
16452 ;. MOV @#HM,R2
16453 ;. RTS PC
16454 ;ENDTST
16455 ;*****
16456 TST6: SCOPE
16457 CLR LOOPIN ;INITIALIZE LOOP INDICATOR
16458 MOV @#TSTLOC+20022,$TMP1 ;SAVE FOR LATER
16459 MOV @TSTLOC+2,R3 ;GET ADDRESS OF EXECUTION BUFFER
16460 MOV @EXPTBL,R4 ;GET ADDRESS OF EXPECTED DATA TABLE
16461 MOV @HITMIS,R5 ;PUT ADDRESS OF HIT/MISS REGISTER IN GPR
16462 BIC #1000,BCSR ;DISABLE HALT ON BREAK
16463 CMP LOOPIN,#7 ;DO UNTIL LOOP INDICATOR EQUALS 7
16464 BEQ ENDRHT ;THEN EXIT
16465 MOV #15,R2
16466 MOV @BACDAT,R0 ;PUT BACKGROUND DATA INTO
16467 MOV @TSTLOC,R1 ;EXECUTION BUFFER
16468 MOV (R0)+,(R1)+
16469 SOB R2,#2 ;LOOP FOR TEN WORDS
16470 CMP LOOPIN,#6 ;IS THIS LAST LOOP
16471 BNE #3 ;IF YES THEN
16472 MOV $TMP1,@#TSTLOC+20022 ;INVALIDATE FETCH OF PRECDAT
16473 BR #4 ;ELSE
16474 MOV #5/37,(R5)+ ;MOVE "TST" INSTRUCTION TO BUFFER

```

{38}

CORRADO KDJ11 B CLUSTER MAC111 3001046) 05 APR 84 16:45 PAGE 300
COKOAB.P11 05 APR 84 16:45 TEST HIT/MISS REGISTER TEST

SEQ 0299

16475	102600	012713	177752		MOV	0HITMISS,(R3)	MOVE HIT/MISS REG. ADD. TO BUFFER
16476	102604	004737	003122	43:	JSR	PC, TSTLOC	GO EXECUTE TEST CODE
16477	102610	021437	110504		CMP	(R4), RECDAT	IF EXPECTED DATA NOT EQUAL TO
16478	102614	001403			BEG	51	RECEIVED DATA THEN
16479	102616	011437	001124		MOV	(R4),IGDDAT	SAVE EXPECTED PATTERN
16480	102622	104017			ERROR	17	ERROR IN RECORDING HITS IN HIT/MISS
16481	102624	005724		53:	TST	(R4),	INCREMENT POINTER TO EXPECTED PATTERN
16482	102626	005237	003114		INC	LOOPIN	INCREMENT LOOP COUNTER
16483	102632	000734			BR	11	
16484	102634			ENDHRT:			
16485	102634	052737	001000 177520		BIS	01000,BCSR	ENABLE HALT ON BREAK
16486	102642	000422			BR	TST7	GO TO NEXT TEST
16487							
16488	102644	000240		BACDAT:	.WORD	NOP	
16489	102646	000240			.WORD	NOP	
16490	102650	000240			.WORD	NOP	
16491	102652	000240			.WORD	NOP	
16492	102654	000240			.WORD	NOP	
16493	102656	000240			.WORD	NOP	
16494	102660	000240			.WORD	NOP	
16495	102662	000240			.WORD	NOP	
16496	102664	011537	110504		MOV	(R5),RECDAT	SAVE CONTENTS OF HIT/MISS REGISTER
16497	102670	000207			RTS	PC	
16498							
16499	102672	000077		EXPTBL:	.WORD	77	
16500	102674	000037			.WORD	37	
16501	102676	000057			.WORD	57	
16502	102700	000067			.WORD	67	
16503	102702	000073			.WORD	73	
16504	102704	000075			.WORD	75	
16505	102706	000076			.WORD	76	
16506							


```

16507 .SHITL TEST - BYTE ALLOCATION TEST
16508 ;BYTE ALLOCATION TEST - THIS TEST WILL VERIFY THAT THE CACHE SYSTEM CORRECTLY
16509 ;HANDLES BYTE ACCESSSES. THE TEST WILL FIRST CHECK THAT A WRITE ACCESS WHICH
16510 ;IS A MISS DOES NOT UPDATE THE CACHE. THIS WILL BE DONE BY FIRST ASSURING
16511 ;A MISS AT A TEST LOCATION. THEN A WRITE BYTE ACCESS WILL BE DONE. FINALLY
16512 ;THE LOCATION WILL BE READ. THE WRITE BYTE SHOULD NOT HAVE ALLOCATED THE
16513 ;ADDRESS. NEXT THE TEST WILL VERIFY THAT IF A WRITE BYTE ACCESS IS A HIT
16514 ;THAT THE CACHE LOCATION IS UPDATED. THE TEST WILL FIRST WRITE A TEST LOCATION
16515 ;WITH A PATTERN TO ALLOCATE THE ADDRESS. THEN A SECOND PATTERN WILL BE WRITTEN
16516 ;TO THE HIGH BYTE OF THE TEST LOCATION. THE TEST LOCATION WILL THEN BE READ.
16517 ;IF THE HIGH BYTE OF THE TEST LOCATION IS EQUAL TO THE SECOND PATTERN THEN THE
16518 ;LOCATION WAS UPDATED. THE SECOND TEST WILL BE REPEATED TO TEST LOW BYTE
16519 ;ACCESSSES.
16520 ;
16521 ;BGNTST
16522 ;SAVE VECTOR 114
16523 ;LET VECTOR 114 POINT TO BYTE PARITY ROUTINE
16524 ;SET PARITY ABORT BIT IN CACHE CONTROL REGISTER
16525 ;READ TEST LOCATION + 4K
16526 ;WRITE LOW BYTE TO TEST ADDRESS
16527 ;READ LOW BYTE OF TEST ADDRESS
16528 ;IF HIT/MISS REGISTER BIT 1 SET THEN
16529 ;. ERROR WRITE BYTE ALLOCATES THE CACHE
16530 ;ENDIF
16531 ;WRITE PATTERN 1 TO TEST LOCATION
16532 ;WRITE BYTE PATTERN 2 TO TEST LOCATION+1
16533 ;IF BIT1 NOTSETIN HIT/MISS REGISTER THEN
16534 ;. IF B0 PARITY ERROR THEN
16535 ;. ERROR LOW BYTE PARITY ERROR ON WRITE BYTE HIT
16536 ;. ELSE
16537 ;. IF B1 PARITY ERROR THEN
16538 ;. ERROR HIGH BYTE PARITY ERROR ON WRITE BYTE HIT
16539 ;. ELSE
16540 ;. WRITE BYTE HIT DOES NOT RECORD A HIT
16541 ;. ENDF
16542 ;. ENDF
16543 ;ELSE
16544 ;. READ TEST LOCATION
16545 ;. IF RECIEVED DATA NOT EQUAL TO EXPECTED DATA THEN
16546 ;. . IF BYTE DATA REVERSED THEN
16547 ;. . ERROR BYTES REVERSED ON WRITE CICLES
16548 ;. . ELSE
16549 ;. . IF HIGH BYTE DATA ZERO THEN
16550 ;. . ERROR IN WRITING TO HIGH BYTE
16551 ;. . ELSE
16552 ;. . ERROR IN WRITING TO HIGH BYTE
16553 ;. . ENDF
16554 ;. ENDF
16555 ;. ENDF
16556 ;ENDIF
16557 ;WRITE PATTERN 1 TO TEST LOCATION
16558 ;WRITE BYTE PATTERN 2 TO TEST LOCATION LOW BYTE
16559 ;IF BIT1 NOTSETIN HIT/MISS REGISTER THEN
16560 ;. IF B0 PARITY ERROR THEN
16561 ;. ERROR LOW BYTE PARITY
16562 ;. ELSE

```

188

CORDBAR KDJ11 B CLUSTER MACY11 50(1046) 05 APR 84 16:45 PAGE 302
COKDAB,P11 05 APR 84 16:45 TEST BYTE ALLOCATION TEST

SEQ 0301

```

16563      IF B1 PARITY ERROR THEN
16564      .      ERROR HIGH BYTE PRITY
16565      .      ELSE
16566      .      WRITE BYTE HIT DOES NOT RECORD A HIT
16567      .      ENDIF
16568      .      ENDIF
16569      ELSE
16570      READ TEST LOCATION
16571      IF RECIEVED DATA NOT EQUAL TO EXPECTED DATA THEN
16572      .      IF BYTE DATA REVERSED THEN
16573      .      .      ERROR BYTES REVERSED
16574      .      .      ELSE
16575      .      .      IF LOW BYTE DATA ZERO THEN
16576      .      .      .      ERROR IN DATA PATH
16577      .      .      ELSE
16578      .      .      .      ERROR IN DATA PATH
16579      .      .      .      ENDIF
16580      .      .      ENDIF
16581      .      ENDIF
16582      .      ENDIF
16583      CLEAR CACHE CONTROL REGISTER
16584      RESTORE VECTOR 114
16585      EXIT TEST
16586      .
16587      BYTE PARITY ROUTINE:  SAVE MSR
16588      .      RETURN
16589      .
16590      .
16591      .
16592      *****
16593      TST7:  SCOPE
16594      MOV    @0114, SLOC00      ;SAVE VECTOR 114
16595      MOV    @08YPARR,@0114    ;LET VECTOR 114 POINT TO ABORT ROUTINE
16596      CLR    R3                ;CLEAR MSR SAVE REGISTER
16597      MOV    @0BIT07,CCR       ;SET PARITY ABORT BIT IN CCR
16598      TST    TSTLOC+8192,      ;READ TEST LOCATION + 4K TO CAUSE MISS
16599      BIC    @1000,BCSR        ;DISABLE HALT ON BREAK
16600      CLR    TSTLOC            ;WRITE LOW BYTE OF TEST LOCATION
16601      TST    TSTLOC            ;READ LOW BYTE OF TEST LOCATION
16602      MOV    HITMIS,RECDAT     ;STORE REGISTER
16603      BIT    @0BIT02,RECDAT    ;IF BIT1 OF HIT/MISS REGISTER SET
16604      BEQ    1$               ;THEN
16605      ERROR  +20               ;WRITE BYTE ALLOCATES CACHE
16606      CLR    TSTLOC            ;WRITE PATTERN 1 TO TEST LOCATION
16607      BIS    @377, @0TSTLOC+1 ;WRITE PATTERN 2 TO HIGH BYTE
16608      MOV    HITMIS,RECDAT     ;STORE REGISTER
16609      BIT    @0BIT02,RECDAT    ;IF BIT1 NOTSETIN HIT/MISS REGISTER
16610      BNE    3$               ;THEN
16611      ERROR  +21               ;WRITE BYTE HIT DOES NOT RECORD HIT
16612      BR     3$               ;ELSE
16613      CMP    @177400,@0TSTLOC  ;IF TEST LOCATION NOT EQUAL TO
16614      BEQ    3$               ;PATTERN 2 THEN
16615      ERROR  +22               ;BYTES REVERSED ON WRITE CYCLES
16616      CLR    TSTLOC            ;WRITE PATTERN 1 TO TEST LOCATION
16617      BIS    @377, @0TSTLOC    ;WRITE PATTERN 2 TO LOW BYTE
16618      MOV    HITMIS,RECDAT     ;STORE REGISTER
16619      BIT    @0BIT02,RECDAT    ;IF BIT1 NOTSETIN HIT/MISS REGISTER

```


COKDABO KDJ11 B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 304
 COKDAB.P11 05-APR-84 16:45 TEST - PDR BIT15 (BYPASS) TEST

SEQ 0303

```

16634 .SBTTL TEST - PDR BIT15 (BYPASS) TEST
16635 ;PDR BIT15 (BYPASS) TEST - THIS TEST WILL VERIFY THAT WHEN BIT<15> IS SET
16636 ;IN A PDR AND AN ACCESS IS MADE TO A LOCATION MAPPED BY THE SELECTED PDR
16637 ;THAT WOULD NORMALLY CAUSE A CACHE ACCESS, THE CACHE IS BYPASSED (I.E. THE
16638 ;CACHE LOCATION IS INVALIDATED AND A MAIN MEMORY IS READ. THIS WILL BE DONE
16639 ;BY FIRST WRITING A TEST LOCATION WITH A KNOWN PATTERN TO ALLOCATE THE ADDRESS.
16640 ;THEN THE LOCATION WILL BE WRITTEN AGAIN WITH THE MMU ENABLED AND BIT <15> SET
16641 ;IN THE CONTROLLING PDR WITH A NEW PATTERN. THE LOCATION WILL THEN BE READ
16642 ;AND A MISS SHOULD BE LOGGED IN THE HIT/MISS REGISTER. THIS READ WILL ALSO
16643 ;BRING VALID DATA INTO CACHE FROM MEMORY. THE MMU WILL AGAIN BE ENABLED
16644 ;WITH BIT <15> SET AND A READ CYCLE DONE. THIS SHOULD INVALIDATE THE CACHE.
16645 ;NEXT THE MMU WILL BE DISABLED AND THE LOCATION READ FOR A THIRD TIME. THIS
16646 ;WILL BE DONE WITH BIT<15> STILL SET. A MISS SHOULD ALSO BE LOGGED. LASTLY
16647 ;BIT<15> WILL BE CLEARED AND THE MMU ENABLED AND THE LOCATION AGAIN READ.
16648 ;THIS TIME A CACHE HIT SHOULD BE LOGGED.
16649 ;
16650 ;
16651 ;BGNTEST
16652 ;SETUP MMU REGISTERS
16653 ;WRITE TEST LOCATION WITH PATTERN 1
16654 ;SET BIT<15> IN PDR FOR TEST LOCATION
16655 ;ENABLE MMU
16656 ;WRITE TEST LOCATION WITH PATTERN 2
16657 ;DISABLE MMU AND CLEAR BIT<15> IN PDR
16658 ;READ TEST LOCATION
16659 ;IF HIT/MISS REGISTER BIT<1> SET THEN
16660 ;. ERROR CONDITIONAL BYPASS DOESN'T INVALIDATE CACHE
16661 ;ENDIF
16662 ;SET BIT<15> IN PDR FOR TEST LOCATION
16663 ;ENABLE MMU
16664 ;READ TEST LOCATION (SHOULD INVALIDATE CACHE)
16665 ;DISABLE MMU
16666 ;READ TEST LOCATION
16667 ;IF HIT/MISS REGISTER BIT<1> SET THEN
16668 ;. ERROR CONDITIONAL BYPASS DOESN'T INVALIDATE CACHE
16669 ;ENDIF
16670 ;CLEAR BIT<15> IN PDR
16671 ;TURN ON MMU
16672 ;READ TEST LOCATION
16673 ;IF HIT/MISS REGISTER BIT<1> NOT SET THEN
16674 ;. ERROR IN RECORDING HITS IN HIT/MISS
16675 ;ENDIF
16676 ;TURN OFF MMU
16677 ;ENDTEST
16678 ;
16679 ;
16680 *****
16680 TST10: SCOPE
16681 JSR PC, INITMM ;SETUP MEMORY MANAGEMENT REGISTERS
16682 CLR @015LOC ;WRITE TEST LOCATION WITH PATTERN 1
16683 MOV @177777,$GDDAT ;SAVE DATA IN MEMORY
16684 BIS @BIT15, KIPDR0 ;SET BIT15 IN PDR FOR TEST LOCATION
16685 INC SR0 ;TURN ON MEMORY MANAGEMENT
16686 MOV @177777,@015LOC ;WRITE TEST LOCATION WITH PATTERN 2
16687 CLR SR0 ;TURN OFF MEMORY MANAGEMENT
16688 BIC @BIT15, KIPDR0 ;CLEAR BIT15 IN PDR FOR TEST LOCATION
16689 BIC @1000,BCSR ;DISABLE HALT ON BREAK

```

```

MOV      @01STLOC,R1
MOV      HITMIS,RECDAT
BIT      @BIT02,RECDAT
BEQ      2$
ERROR    +23
BIS      @BIT15, KIPDRO
INC      SRO
TST      TSTLOC
BIC      @BIT15, KIPDRO
CLR      SRO
MOV      @01STLOC,R1
MOV      HITMIS,RECDAT
BIT      @BIT02,RECDAT
BEQ      3$
ERROR    +23
INC      SRO
TST      @01STLOC
MOV      HITMIS,RECDAT
BIT      @BIT02,RECDAT
BNE      4$
ERROR    +45
BIC      @BIT00, SRO
BIS      @1000,BCSR

```

```

;READ TEST LOCATION
;SAVE HIT/MISS REGISTER
;IF HIT/MISS REGISTER BIT 1 SET
;THEN
;CONDITIONAL BYPASS DOESN'T INVALIDATE CACHE
;SET BIT15 IN PDR FOR TEST LOCATION
;TURN ON MEMORY MANAGEMENT
;READ TEST LOCATION
;CLEAR BIT 15 IN PDR
;TURN OFF MMU
;READ TEST LOCATION
;STORE HIT/MISS TO REGISTER 2
;IF HIT/MISS REGISTER BIT 1 SET
;THEN
;CONDITIONAL BYPASS DOESN'T INVALIDATE 0 CACHE
;TURN ON MEMORY MANAGEMENT
;READ TEST LOCATION ONE MORE TIME
;STORE HIT/MISS TO REGISTER 2
;IF HIT/MISS REGISTER BIT 1 NOT SET
;THEN
;ERROR IN RECORDING HITS IN HIT/MISS
;TURN OFF MMU
;ENABLE HALT ON BREAK

```

COKDABO KDJ11-B CLUSTER MAC111 30(1046) 05-APR-84 16:45 PAGE 306
 COKDAB.P11 05-APR-84 16:45 TEST - FLUSH CACHE TEST

SEQ 0305

```

16714 .SBTTL TEST - FLUSH CACHE TEST
16715 ;FLUSH CACHE TEST - THIS TEST WILL VERIFY THAT WHEN CCR BIT<8> IS
16716 ;SET, THE ENTIRE CACHE IS INVALIDATED. FIRST 8K BYTES OF ADDRESSES
16717 ;WILL BE READ TO ALLOCATE CACHE. THEN THE CACHE WILL BE FLUSHED.
16718 ;THE SAME SET OF ADDRESS WILL THEN BE READ AND THE HIT/MISS REGISTER
16719 ;CHECKED AFTER EACH READ FOR A MISS TO BE LOGGED.
16720 ;
16721 ;BGNTST
16722 ;GET FIRST ADDRESS OF 8KB BUFFER
16723 ;INITIALIZE LOOP COUNTER TO 4KWORD COUNT
16724 ;DO UNTIL LOOP COUNTER = 0
16725 ;. READ @ADDRESS+
16726 ;ENDDO
16727 ;GET FIRST ADDRESS OF 8KB BUFFER
16728 ;INITIALIZE LOOP COUNTER TO 2KWORD COUNT
16729 ;INITIALIZE MISS COUNT TO 4KWORD COUNT
16730 ;FLUSH CACHE
16731 ;DO UNTIL LOOP COUNTER = 0
16732 ;. READ @ADDRESS+
16733 ;. INCREMENT ADDRESS TO READ EVERY 2WORDS
16734 ;. IF HIT/MISS REGISTER BIT<1> SET THEN
16735 ;. DECREMENT MISS COUNT
16736 ;. ENDF
16737 ;ENDDO
16738 ;GET FIRST ADDRESS OF 8KB BUFFER
16739 ;INITIALIZE LOOP COUNTER TO 4KWORD COUNT
16740 ;DO UNTIL LOOP COUNTER = 0
16741 ;. READ @ADDRESS+
16742 ;ENDDO
16743 ;GET LAST ADDRESS OF 8KB BUFFER
16744 ;INITIALIZE LOOP COUNTER TO 2KWORD COUNT
16745 ;FLUSH CACHE
16746 ;DO UNTIL LOOP COUNTER = 0
16747 ;. READ @ADDRESS
16748 ;. DECREMENT ADDRESS TO READ EVERY 2WORDS
16749 ;. IF HIT/MISS REGISTER BIT<1> SET THEN
16750 ;. DECREMENT MISS COUNT
16751 ;. ENDF
16752 ;ENDDO
16753 ;IF MISS COUNT NOT EQUAL TO 4096, THEN
16754 ;. ERROR HITS RECORDED AFTER FLUSHING CACHE.
16755 ;ENDIF
16756 ;ENDTST
16757 ;
16758 ;*****
16759 103372 000004 TST11: SCOPE
16760 103374 004737 132156 JSR PC, INITMM ;INITIALIZE MMU
16761 103400 012737 001600 172354 MOV #1600,KIPAR6 ;LET PAR6 POINT TO LOW 28K
16762 103406 012701 140000 MOV #140000,R1 ;GET ADDRESS OF 8K BYTE BUFFER
16763 103412 005237 177572 INC SRO ;ENABLE MMU
16764 103416 012702 010000 MOV #4096,, R2 ;INIT LOOP COUNTER TO 4K WORD COUNT
16765 103422 005721 1$: TST (R1)+ ;READ ADDRESS TO ALLOCATE CACHE
16766 103424 077202 SOB R2, 1$ ;DO UNTIL ALL LOCATIONS ALLOCATED
16767 103426 012701 140000 MOV #140000,R1 ;GET ADDRESS OF 8K BYTE BUFFER
16768 103432 012702 004000 MOV #2048,, R2 ;INIT LOOP COUNTER TO 2K WORD COUNT
16769 103436 012703 010000 MOV #4096,, R3 ;INITIALIZE MISS COUNT TO 4K

```


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

```

16803 ;SBTH TEST - UNCONDITIONAL BYPASS TEST
16804 ;UNCONDITIONAL BYPASS TEST - THIS TEST WILL VERIFY THAT WHEN CCR
16805 ;BIT<9> IS SET, A MEMORY REFERENCE IS FORCED TO MAIN MEMORY. THIS
16806 ;WILL ALSO VERIFY THAT READ AND WRITE HIT INVALIDATE THE CACHE
16807 ;LOCATIONS WHEN BYPASS IS SET.
16808 ;
16809 ;BGNIST
16810 ;READ TEST LOCATIONS TO SETUP POSSIBILITY OF HIT
16811 ;SET CCR BIT<9>
16812 ;READ FIRST TEST LOCATION
16813 ;IF HIT/MISS REGISTER BIT<1> NOT SET THEN
16814 ;. ERROR IN RECORDING HITS IN HIT/MISS
16815 ;ENDIF
16816 ;WRITE SECOND TEST LOCATION
16817 ;IF HIT/MISS REGISTER BIT<1> NOT SET THEN
16818 ;. ERROR IN RECORDING HITS IN HIT/MISS
16819 ;ENDIF
16820 ;CLEAR CCR BIT<9>
16821 ;READ FIRST LOCATION
16822 ;IF HIT/MISS REGISTER BIT<1> SET THEN
16823 ;. ERROR BYPASS DOESN'T INVALIDATE CACHE
16824 ;ENDIF
16825 ;READ SECOND LOCATION
16826 ;IF HIT/MISS REGISTER BIT<1> SET THEN
16827 ;. ERROR BYPASS DOESN'T INVALIDATE CACHE
16828 ;ENDIF
16829 ;ENDTST
16830 ;
16831 ;*****
16832 TST12: SCOPE
16833 TST @TSTLOC ;ALLOCATE FIRST TEST LOCATION
16834 IST @TSTLOC+2 ;ALLOCATE SECOND TEST LOCATION
16835 BIS @BIT09,CCR ;SET CCR BIT 9 (CACHE BYPASS)
16836 BIC #1000,BCSR ;DISABLE HALT ON BREAK
16837 TST @TSTLOC ;READ FIRST TEST LOCATION
16838 MOV HITMIS,RECDAT ;STORE HIT/MISS TO REGISTER 2
16839 BIT @BIT02,RECDAT ;IF HIT/MISS REG. BIT 1 NOT SET
16840 BNE 2$ ;THEN
16841 ERROR +45 ;ERROR IN RECORDING HITS IN HIT/MISS WITH CCR<9>
16842 CLR @TSTLOC+2 ;WRITE SECOND LOCATION
16843 MOV HITMIS,RECDAT ;STORE HIT/MISS TO REGISTER 2
16844 BIT @BIT02,RECDAT ;IF HIT/MISS REG. BIT 1 NOT SET
16845 BNE 3$ ;THEN
16846 ERROR +45 ;ERROR IN RECORDING HITS IN HIT/MISS WITH CCR<9>
16847 TST @TSTLOC ;READ FIRST TEST LOCATION
16848 MOV HITMIS,RECDAT ;STORE HIT/MISS TO REGISTER 2
16849 BIT @BIT02,RECDAT ;IF HIT/MISS REG. BIT 1 SET
16850 BEQ 4$ ;THEN
16851 ERROR +25 ;BYPASS DOESN'T INVALIDATE CACHE
16852 CLR @TSTLOC+2 ;WRITE SECOND LOCATION
16853 MOV HITMIS,RECDAT ;STORE HIT/MISS TO REGISTER 2
16854 BIT @BIT02,RECDAT ;IF HIT/MISS REG. BIT 1 SET
16855 BEQ 5$ ;THEN
  
```


K8

CORDBO RDJ11 B CLUSTER MAC111 30(1046) 05 APR-84 16:45 PAGE 309
COKDAB,P11 05 APR 84 16:45 TEST - UNCONDITIONAL BYPASS TEST
16856 103774 104025 ERROR *25
16857 103776 042737 001000 177746 5s: BIC #B1109,CCR
16858 104004 052737 001000 177520 BIS #1000,BCSR
16859

;BYPASS DOESN'T INVALIDATE CACHE
;RESTORE CCR
;ENABLE HALT ON BREAK

SEQ 0308

COKDABO KDJ11 B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 310
 COKDAB.P11 05-APR-84 16:45 TEST - WRITE WRONG DATA PARITY TEST

SEQ 0309

```

16860 ;GBITL TEST - WRITE WRONG DATA PARITY TEST
16861 ;WRITE WRONG DATA PARITY TEST - THIS TEST WILL VERIFY THAT WHEN CCR
16862 ;BIT<6> = 1 AND CCR BITS<7,0> = 0,1, A READ MISS OCCURS AFTER A
16863 ;WRITE. THE WRITE WITH CCR BIT<6> = 1 TO A LOCATION WILL CAUSE A
16864 ;CACHE UPDATE AND WRONG PARITY TO BE WRITTEN SO WHEN THE LOCATION
16865 ;IS READ INSTEAD OF A HIT BEING RECORDED THE CACHE PARITY ERROR
16866 ;WILL CAUSE A CACHE MISS. THIS TEST WILL BE DONE A BYTE AT A TIME.
16867 ;
16868 ;BGNIST
16869 ;SAVE CONTENTS OF VECTOR 114
16870 ;LET VECTOR 114 POINT TO ABORT ROUTINE
16871 ;CLEAR MSER
16872 ;IF MSER NOT CLEAR THEN
16873 ;. ERROR MSER DOESN'T CLEAR ON WRITE REFERENCE
16874 ;ENDIF
16875 ;WRITE TEST LOCATION
16876 ;SET BITS<6,0> IN CCR
16877 ;WRITE TEST LOCATION LOW BYTE
16878 ;CLEAR CCR BIT<6> (WRITE WRONG DATA PARITY)
16879 ;INITIALIZE ERROR INDICATORS
16880 ;READ TEST LOCATION LOW BYTE
16881 ;IF BIT<1> SET IN HIT/MISS REGISTER THEN
16882 ;. PARITY ERROR DOESN'T CAUSE MISS
16883 ;ELSE
16884 ;. IF MSER BITS <7:5> ZERO THEN
16885 ;. PARITY ERROR DOESN'T SET MSER PROPERLY
16886 ;. ELSE
16887 ;. SET ERROR IN LOW BYTE INDICATOR
16888 ;. ENDIF
16889 ;. ENDIF
16890 ;ENDIF
16891 ;CLEAR MSER
16892 ;IF MSER NOT CLEAR THEN
16893 ;. ERROR MSER DOESN'T CLEAR ON WRITE REFERENCE
16894 ;ENDIF
16895 ;SET CCR BIT<6>
16896 ;WRITE TEST LOCATION HIGH BYTE
16897 ;CLEAR CCR BIT<6>
16898 ;READ TEST LOCATION HIGH BYTE
16899 ;IF BIT<1> SET IN HIT/MISS REGISTER THEN
16900 ;. IF MSER BITS<7:5> NOT SET THEN
16901 ;. PARITY ERROR DOESN'T CAUSE A MISS
16902 ;. ELSE
16903 ;. SET ERROR IN HIGH BYTE INDICATOR
16904 ;. ENDIF
16905 ;ELSE
16906 ;. IF MSER BITS <7:5> ZERO THEN
16907 ;. PARITY ERROR DOESN'T SET MSER
16908 ;. ENDIF
16909 ;ENDIF
16910 ;RESTORE CONTENTS OF VECTOR 114
16911 ;IF ERROR INDICATORS SET THEN
16912 ;. IF ERROR IN BOTH BYTES THEN
16913 ;. PARITY ERROR IGNORED
16914 ;. ELSE
16915 ;. IF ERROR IN LOW BYTE THEN

```

M8

CORDBO RDJ11 B CLUSTER MAC11 30(1046) 05-APR-84 16:45 PAGE 311
 CORDAB.P11 05-APR-84 16:45 TEST WRITE WRONG DATA PARITY TEST

SEQ 0310

```

16916      .          .          .          .          .          .          .          .          .          .
16917      .          .          .          .          .          .          .          .          .          .
16918      .          .          .          .          .          .          .          .          .          .
16919      .          .          .          .          .          .          .          .          .          .
16920      .          .          .          .          .          .          .          .          .          .
16921      .          .          .          .          .          .          .          .          .          .
16922      .          .          .          .          .          .          .          .          .          .
16923      .          .          .          .          .          .          .          .          .          .
16924      .          .          .          .          .          .          .          .          .          .
16925      .          .          .          .          .          .          .          .          .          .
16926      .          .          .          .          .          .          .          .          .          .
16927      .          .          .          .          .          .          .          .          .          .
16928      .          .          .          .          .          .          .          .          .          .
16929      .          .          .          .          .          .          .          .          .          .
16930      104012 000004      .          .          .          .          .          .          .          .          .
16931      104014 013737 000114 002762      .          .          .          .          .          .          .          .
16932      104022 012737 104300 000114      .          .          .          .          .          .          .          .
16933      104030 005037 177744      .          .          .          .          .          .          .          .
16934      104034 005737 177744      .          .          .          .          .          .          .          .
16935      104040 001401      .          .          .          .          .          .          .          .
16936      104042 104026      .          .          .          .          .          .          .          .
16937      104044 005037 003122      .          .          .          .          .          .          .          .
16938      104050 012737 000101 177746      .          .          .          .          .          .          .          .
16939      104056 112737 000377 003122      .          .          .          .          .          .          .          .
16940      104064 042737 000100 177746      .          .          .          .          .          .          .          .
16941      104072 005002      .          .          .          .          .          .          .          .
16942      104074 042737 001000 177520      .          .          .          .          .          .          .          .
16943      104102 105737 003122      .          .          .          .          .          .          .          .
16944      104106 013703 177752      .          .          .          .          .          .          .          .
16945      104112 032703 000004      .          .          .          .          .          .          .          .
16946      104116 001402      .          .          .          .          .          .          .          .
16947      104120 104027      .          .          .          .          .          .          .          .
16948      104122 000405      .          .          .          .          .          .          .          .
16949      104124 032737 000340 177744      .          .          .          .          .          .          .          .
16950      104132 001001      .          .          .          .          .          .          .          .
16951      104134 104030      .          .          .          .          .          .          .          .
16952      104136 005037 177744      .          .          .          .          .          .          .          .
16953      104142 005737 177744      .          .          .          .          .          .          .          .
16954      104146 001401      .          .          .          .          .          .          .          .
16955      104150 104026      .          .          .          .          .          .          .          .
16956      104152 052737 000100 177746      .          .          .          .          .          .          .          .
16957      104160 112737 000377 003123      .          .          .          .          .          .          .          .
16958      104166 042737 000100 177746      .          .          .          .          .          .          .          .
16959      104174 105737 003123      .          .          .          .          .          .          .          .
16960      104200 013737 177752 110504      .          .          .          .          .          .          .          .
16961      104206 032737 000004 110504      .          .          .          .          .          .          .          .
16962      104214 001402      .          .          .          .          .          .          .          .
16963      104216 104027      .          .          .          .          .          .          .          .
16964      104220 000405      .          .          .          .          .          .          .          .
16965      104222 032737 000340 177744      .          .          .          .          .          .          .          .
16966      104230 001001      .          .          .          .          .          .          .          .
16967      104232 104030      .          .          .          .          .          .          .          .
16968      104234 005037 177746      .          .          .          .          .          .          .          .
16969      104240 013737 002762 000114      .          .          .          .          .          .          .          .
16970      104246 032703 000003      .          .          .          .          .          .          .          .
16971      104252 001401      .          .          .          .          .          .          .          .

TST13:  SCOPE
MOV      @0114, SLOC00      ;SAVE CONTENTS OF VECTOR 114
MOV      @DAPAB0,@0114      ;LET VECTOR POINT TO ABORT ROUTINE
CLR      MSER                ;CLEAR MSER
TST      MSER                ;IF MSER NOT CLEAR
BEQ      1$                  ;THEN
ERROR    +26                 ;MSER DOES NOT CLEAR ON WRITE REFERENCE
1$:      CLR      @01STLOC      ;WRITE TEST LOCATION TO ALLOCATE CACHE
MOV      @101, CCR           ;SET BITS<6,0> IN CCR
MOVB     @377, TSILOC        ;WRITE LOW BYTE WITH BAD PARITY
BIC      @BIT06, CCR         ;CLEAR WRITE WRONG DATA PARITY BIT
CLR      R2                  ;CLEAR ERROR INDICATORS
BIC      @1000,BCSR          ;DISABLE HALT ON BREAK
TSTB     @01STLOC           ;READ LOW BYTE OF TEST LOCATION
MOV      HITMIS, R3          ;SAVE HIT/MISS
BIT      @BIT02, R3          ;IF BIT 1 SET IN HIT/MISS REGISTER
BEQ      2$                  ;THEN
ERROR    +27                 ;PARITY ERROR DON'T CAUSE A MISS
BR       3$                  ;ELSE
2$:      BIT      @340, MSER   ;IF MSER BIT<7:5> NOT ZERO
BNE      3$                  ;THEN
ERROR    +30                 ;PARITY ERROR DON'T SET MSER WITH CCR<7>=0
3$:      CLR      MSER        ;CLEAR MSER
TST      MSER                ;IF MSER NOT CLEAR
BEQ      4$                  ;THEN
ERROR    +26                 ;MSER DOES NOT CLEAR ON WRITE REFERENCE
4$:      BIS      @BIT06, CCR   ;SET CCR BIT 6 (WRITE WRONG PARITY)
MOVB     @377, @01STLOC+1    ;WRITE HIGH BYTE OF TEST LOCATION
BIC      @BIT06, CCR         ;CLEAR WRITE WRONG PARITY BIT
TSTB     @01STLOC+1         ;READ HIGH BYTE OF TEST LOCATION
MOV      HITMIS,RECDAT       ;SAVE HIT/MISS
BIT      @BIT02,RECDAT       ;IF BIT 1 SET IN HIT/MISS REGISTER
BEQ      5$                  ;THEN
ERROR    +27                 ;PARITY ERROR DON'T CAUSE A MISS
BR       6$                  ;ELSE
5$:      BIT      @340, MSER   ;IF BITS <7:5> ZERO
BNE      6$                  ;THEN
ERROR    +30                 ;PARITY ERROR DON'T SET MSER WITH CCR<7>=0
6$:      CLR      CCR          ;CLEAR CCR BEFORE EXIT
MOV      SLOC00, @0114      ;RESTORE CONTENTS OF VECTOR 114
BIT      @3, R2              ;IF ERROR INDICATORS SET
BEQ      7$                  ;THEN

```


COKDABO K0J11 B CLUSTER MACY11 30(1046) 05 APR 84 16:45 PAGE 248
COKDAB.P11 05-APR 84 16:45 FLOATING POINT TESTS

SEQ 0247

13624	066610	004767	000424	JSR	R7,MLFSUB		DO TEST
13625	066614	040177	177777	.WORD	40177,-1	ACO	
13626	066620	040200	000000	.WORD	40200,0	FSRC	
13627	066624	040177	177777	.WORD	40177,1		RESULT
13628	066630	000040		.WORD	40		TEST FPS
13629	066632	000040		.WORD	40		RESULTANT FPS
13630							
13631	066634	005037	002776	16/NORMALIZE			
13632	066640	004767	000374	CLR	00FLAG		NO INTERRUPT
13633	066644	040100	000000	JSR	R7,MLFSUB		DO TEST
13634	066650	040100	000000	.WORD	40100,0	ACO	
13635	066654	040020	000000	.WORD	40100,0	FSRC	
13636	066660	000012		.WORD	40020,0		RESULT
13637	066662	000000		.WORD	12		TEST FPS
13638				.WORD	0		RESULTANT FPS
13639	066664	005037	002776	17/ROUND			
13640	066670	004767	000344	CLR	00FLAG		NO INTERRUPT
13641	066674	017500	000000	JSR	R7,MLFSUB		DO TEST
13642	066700	023652	125252	.WORD	17500,0	ACO	
13643	066704	003177	177777	.WORD	23652,125252	FSRC	
13644	066710	007417		.WORD	3177,-1		RESULT
13645	066712	007400		.WORD	7417		TEST FPS
13646				.WORD	7400		RESULTANT FPS
13647	066714	005037	002776	18/AC>0>FSRC ROUND			
13648	066720	004767	000314	CLR	00FLAG		NO INTERRUPT
13649	066724	040342	177777	JSR	R7,MLFSUB		DO TEST
13650	066730	176543	025252	.WORD	40342,-1	ACO	
13651	066734	176711	067324	.WORD	176543,025252	FSRC	
13652	066740	007500		.WORD	176711,67324		RESULT
13653	066742	007510		.WORD	7500		TEST FPS
13654				.WORD	7510		RESULTANT FPS
13655	066744	005037	002776	19/IAC<FSRC<0, ROUND			
13656	066750	004767	000264	CLR	00FLAG		NO INTERRUPT
13657	066754	144600	000000	JSR	R7,MLFSUB		DO TEST
13658	066760	154000	000000	.WORD	144600,0	ACO	
13659	066764	060400	000000	.WORD	154000,0	FSRC	
13660	066770	000017		.WORD	60400,0		RESULT
13661	066772	000000		.WORD	17		TEST FPS
13662				.WORD	0		RESULTANT FPS
13663	066774	005037	002776	10/AC<FSRC, ROUND			
13664	067000	004767	000234	CLR	00FLAG		NO INTERRUPT
13665	067004	060000	000000	JSR	R7,MLFSUB		DO TEST
13666	067010	140377	177776	.WORD	60000,0	ACO	
13667	067014	160177	177776	.WORD	140377,177776	FSRC	
13668	067020	000017		.WORD	160177,177776		RESULT
13669	067022	000010		.WORD	17		TEST FPS
13670				.WORD	10		RESULTANT FPS
13671	067024	005037	002776	11/AC>0>FSRC, TRUNCATE			
13672	067030	004767	000204	CLR	00FLAG		NO INTERRUPT
13673	067034	060000	000000	JSR	R7,MLFSUB		DO TEST
13674	067040	140377	177776	.WORD	60000,0	ACO	
13675	067044	160177	177776	.WORD	140377,177776	FSRC	
13676	067050	007547		.WORD	160177,177776		RESULT
13677	067052	007550		.WORD	7547		TEST FPS
13678				.WORD	7550		RESULTANT FPS
13679	067054	012737	000002 002776	12/UNDERFLOW, NO INTERRUPTS			
				MOV	02,00FLAG		NO INTERRUPT

(9)

CORDBO RDJ11 B CLUSTER MAC111 50(1046) 05 APR 84 16:45 PAGE 514
CORDBO.P11 05 APR 84 16:45 TEST WRITE WRONG TAG PARITY

SEQ 0313

1039
1040 104426 011637 001122 TAPABO: MOV (SP), \$BDADR
1041 104430 104000 ERROR *7
1042 104434 000002 RTI
1043

SAVE ADDRESS THAT CAUSE ABORT
ILLEGAL PARITY INTERRUPT

CORPABO RDJ11 B CLUSTER MACY11 30(1046) 05 APR 84 16:45 PAGE 315
 CORPAB.P11 05 APR 84 16:45 TEST PARITY ABORT TEST

SEQ 0314

```

17044 .SBTTL TEST - PARITY ABORT TEST
17045 ;PARITY ABORT TEST THIS TEST WILL VERIFY THAT WHEN CCR<7,0>
17046 ;1,0, AN ABORT OCCURS ON THE EXECUTION OF AN INSTRUCTION THAT HAS
17047 ;BEEN WRITTEN WITH WRONG PARITY.
17048 ;
17049 ;
17050 ;BGNIST
17051 ;SAVE VECTOR 114
17052 ;SAVE VECTOR 4
17053 ;LET VECTOR 114 POINT TO ABORT ROUTINE
17054 ;LET VECTOR 4 POINT TO ERROR ROUTINE
17055 ;CLEAR EXPECTING ABORT FLAG
17056 ;SET CCR<10> WRITE WRONG TAG PARITY BIT
17057 ;WRITE TEST ADDRESS
17058 ;CLEAR CCR<10>
17059 ;SET CCR TO 0200 ENABLE PARITY ABORTS
17060 ;SET EXPECTING ABORT FLAG
17061 ;READ TEST ADDRESS
17062 ;IF ABORT FLAG NE 0 THEN
17063 ;. ERROR IN PARITY ABORT LOGIC
17064 ;ENDIF
17065 ;RESTORE VECTOR 114
17066 ;RESTORE VECTOR 4
17067 ;EXIT TST
17068 ;
17069 ;ABORT ROUTINE: IF EXPECTING ABORT FLAG NOT SET THEN
17070 ;. ERROR NO ABORT SHOULD HAVE OCCURRED
17071 ;. ELSE
17072 ;. CLEAR (EXPECTING) ABORT FLAG
17073 ;. ENDIF
17074 ;IF MSER NOT EQUAL TO 100040 THEN
17075 ;. PARITY ABORT LOGIC DOESN'T SET MSER PROPERLY
17076 ;. ENDIF
17077 ;IF PC = UPDATED PC THEN
17078 ;. ILLEGAL PARITY ABORT
17079 ;. ENDIF
17080 ;RETURN
17081 ;
17082 ;ENDTST
17083 ;*****
17084 TST15: SCOPE
17085 MOV 00114, SLOC00 ;SAVE VECTOR 114
17086 MOV 000004, SLOC01 ;SAVE VECTOR 4
17087 MOV 104556, 000114 ;LET VECTOR 114 POINT TO ABORT ROUTINE
17088 ;
17089 ; TO AVOID CONFUSION ALLOCATE IN CACHE FOLLOWING INSTRUCTIONS
17090 ;
17091 MOV 0, R4 ;START WITH CURRENT
17092 TST (R4). ;READ A WORD
17093 CMP 0ABORTR, R4 ;GOT TO ABORT ROUTINE?
17094 BNE 11 ;IF NOT, KEEP ON ALLOCATING
17095 CLR R0 ;CLEAR EXPECTING ABORT FLAG
17096 MOV 0BIT10, CCR ;SET WRITE WRONG PARITY BIT
17097 CLR 0BIT10C ;WRITE TEST LOCATION WITH BAD PARITY
17098 MOV 0BIT07, CCR ;ENABLE ABORTS, CLEAR WWP BIT
17099 COM R0 ;SET EXPECTING ABORT FLAG

```

CORDBO KDJ11 B CLUSTER MACY11 30(1046) 05 APR-84 16:45 PAGE 316
 CORDAB,P11 05-APR-84 16:45 TEST PARITY ABORT TEST

SEQ 0315

```

17100 104522 005737 003122 ABORTI: IST @ISTLOC ;READ TEST LOCATION (SHOULD CAUSE ABORT)
17101 104526 005700          IST R0 ;IF ABORT FLAG NOT EQUAL ZERO
17102 104530 001401          BEQ 1$ ;THEN
17103 104532 104034          ERROR +34 ;PARITY ABORT LOGIC DOESN'T WORK
17104 104534 013737 002762 000114 1$: MOV SLOC00, @114 ;RESTORE VECTOR 114
17105 104542 013737 002764 000004      MOV SLOC01, @4 ;RESTORE VECTORE 4
17106 104550 005037 177744          CLR MSER
17107 104554 000426          BR TST16 ;GO TO NEXT TEST
17108
17109
17110 104556 013703 177744 ABORTR: MOV MSER,R3 ;SAVE MSER
17111 104562 005700          TST R0 ;IF EXPECTING ABORT FLAG NOT SET
17112 104564 001004          BNE 1$ ;THEN
17113 104566 011637 001122          MOV (SP), $BDADR ;SAVE ABORT ADDRESS
17114 104572 104007          ERROR +7 ;ILLEGAL PARITY INTERRUPT
17115 104574 000401          BR 2$ ;ELSE
17116 104576 005000          1$: CLR R0 ;CLEAR (EXPECTING) ABORT FLAG
17117 104600 022737 100040 177744 2$: CMP #100040,MSER ;IF MSER NOT EQUAL TO 100040
17118 104606 001404          BEQ 3$ ;THEN
17119 104610 012737 100040 001124      MOV #100040,$GDDAT ;SAVE PROPER MSER SETTING
17120 104616 104035          ERROR +35 ;PARITY ABORT DON'T SET MSER PROPERLY
17121 104620 021627 104526          3$: CMP (SP), $ABORTI+4 ;IF PC EQUAL TO UPDATE PC
17122 104624 001401          BEQ 4$ ;THEN
17123 104626 104007          ERROR +7 ;ILLEGAL PARITY INTERRUPT
17124 104630 000002          4$: RTI ;RETURN
17125

```


COKDABO KDJ11 B CLUSTER MACY11 50(1046) 05-APR-84 16:45 PAGE 317
 COKDAB.P11 05-APR 84 16:45 TEST - PARITY INTERRUPT TEST

SEQ 0316

```

17126 .SBTIL TEST - PARITY INTERRUPT TEST
17127 ;PARITY INTERRUPT TEST - THIS TEST WILL VERIFY THAT WHEN CCR<7,0> =
17128 ;0,0, A PARITY INTERRUPT OCCURS AFTER EXECUTION OF AN INSTRUCTION
17129 ;THAT HAS BEEN WRITTEN WITH WRONG PARITY.
17130 ;
17131 ;BGTST
17132 ;SAVE CONTENTS OF 114
17133 ;SETUP VECTOR 114 TO POINT TO INTERRUPT ROUTINE
17134 ;CLEAR EXPECTING INTERRUPT FLAG
17135 ;WRITE TEST ADDRESS WITH BAD PARITY
17136 ;SET EXPECTING INTERRUPT FLAG
17137 ;READ TEST ADDRESS
17138 ;IF INTERRUPT FLAG NE 0 THEN
17139 ;. PARITY INTERRUPT LOGIC DOESN'T WORK
17140 ;ENDIF
17141 ;RESTORE CONTENTS OF VECTOR 114
17142 ;EXIT TST
17143 ;
17144 ;INTERRUPT ROUTINE: IF EXPECTING INTERRUPT FLAG NE 1 THEN
17145 ;. ERROR NO INTERRUPT SHOULD HAVE OCCURRED
17146 ;. ELSE
17147 ;. CLEAR (EXPECTING) INTERRUPT FLAG
17148 ;. ENDIF
17149 ;IF SAVED PC NE TO UPDATED PC THEN
17150 ;. IF PC = TEST INSTRUCTION PC THEN
17151 ;. ERROR INSTRUCTION ABORTED
17152 ;. ELSE
17153 ;. ILLEGAL PARITY ABORT
17154 ;. ENDIF
17155 ;ENDIF
17156 ;IF MSER NE 0340 THEN
17157 ;. PARITY INTERRUPT DOESN'T SET MSER PROPERLY
17158 ;. ENDIF
17159 ;RETURN
17160 ;
17161 ;ENDTST
17162 ;*****
17163 104632 000004 TST16: SCOPE
17164 104634 013737 000114 002762 MOV 00114, SIUC00 ;SAVE CONTENTS OF VECTOR 114
17165 104642 012737 104730 000114 MOV 0INTERR,00114 ;LET VECTOR POINT TO INTERRUPT ROUTINE
17166 ;
17167 ; TO AVOID CONFUSION ALLOCATE IN CACHE FOLLOWING INSTRUCTIONS
17168 ;
17169 104650 012704 104650 MOV 0,R4 ;START WITH CURRENT
17170 104654 005724 1$; TST (R4)+ ;READ A WORD
17171 104656 022704 104730 CMP 0INTERR,R4 ;GOT TO INTERRUPT ROUTINE?
17172 104662 001374 BNE 1$ ;IF NOT, KEEP ON ALLOCATING
17173 104664 005001 CLR R1 ;CLEAR EXPECTING INTERRUPT FLAG
17174 104666 052737 000100 177746 BIS 0BIT06,CCR ;SET WRITE WRONG DATA PARITY
17175 104674 005037 003122 CLR 001TLOC ;WRITE LOCATION WITH BAD DATA PARITY
17176 104700 005037 177746 CLR CCR ;CLEAR WRITE WRONG DATA PARITY
17177 104704 005101 COM R1 ;SET EXPECTING INTERRUPT FLAG
17178 104706 005737 003122 INTRPC: TST 001TLOC ;READ TEST LOCATION
17179 104712 005701 TST R1 ;IF INTERRUPT FLAG NOT EQUAL ZERO
17180 104714 001401 BEQ 1$ ;THEN
17181 104716 104036 ERROR +36 ;PARITY INTERRUPT LOGIC DOESN'T WORK

```

(39)

COKDABO KDJ11-B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 318
 COKDAB.P11 05-APR-84 16:45 TEST - PARITY INTERRUPT TEST

SEQ 0317

```

17182 104720 013737 002762 000114 1$: MOV SLOC00, @#114 ;RESTORE VECTOR 114
17183 104726 000424 BR TST17 ;;GO TO NEXT TEST
17184
17185
17186 104730 005701 INTERR: TST R1 ;IF EXPECTING INTERRUPT FLAG NOT SET
17187 104732 001004 BNE 1$ ;THEN
17188 104734 011637 001122 MOV (SP), $BDADR ;SAVE INTERRUPT ADDRESS
17189 104740 104007 ERROR +7 ;ILLEGAL PARITY INTERRUPT
17190 104742 000401 BR 2$ ;ELSE
17191 104744 005001 1$: CLR R1 ;CLEAR (EXPECTING) INTERRUPT FLAG
17192 104746 021627 104712 2$: CMP (SP), $INTRPC+4 ;IF SAVED PC NOT EQUAL TO UPDATED PC
17193 104752 001401 BEQ 4$ ;THEN
17194 104754 104007 ERROR +7 ;ILLEGAL PARITY INTERRUPT
17195 104756 022737 000340 177744 4$: CMP $340, MSER ;IF MSER NOT EQUAL TO EXPECTED VALUE
17196 104764 001404 BEQ 5$ ;THEN
17197 104766 012737 000340 001124 MOV $340, $GDDAT ;SAVE PROPER MSER
17198 104774 104035 ERROR +35 ;PARITY INTERRUPT DON'T SET MSER PROPERLY
17199 104776 000002 5$: RTI ;RETURN
17200

```

COMDARO RDJ11-B CLUSTER MAC:11 30(1046) 05-APR-84 16:45 PAGE 319
 COKDAR.P11 05-APR-84 16:45 TEST - MISCELLANEOUS PARITY TEST

SEQ 0318

```

17201 ;SBTTL TEST - MISCELLANEOUS PARITY TEST
17202 ;MISCELLANEOUS PARITY TEST - THIS TEST CHECKS THAT BYPASS CYCLES WITH
17203 ;PARITY ERRORS CAUSE CACHE HIT RESPONSE AND THAT FORCE MISS CYCLES
17204 ;IGNORE PARITY ERRORS.
17205 ;
17206 ;BGMTST
17207 ;WRITE A LOCATION WITH BAD PARITY
17208 ;SET BYPASS IN CCR
17209 ;READ THE LOCATION BACK
17210 ;IF NO HIT OR MSER NOT SET THEN
17211 ;. ERROR
17212 ;ENDIF
17213 ;WRITE A LOCATION WITH BAD PARITY AND SET BYPASS
17214 ;IF MSER SET WHILE READING IT BACK
17215 ;. ERROR
17216 ;ENDIF
17217 ;ENDTST
17218 ;*****
17219 105000 000004 TST17: SCOPE
17220 105002 012703 105002 MOV #.,R3 ;START WITH CURRENT INSTRUCTION
17221 105006 005723 10$ TST (R3). ;READ A WORD
17222 105010 022703 105152 CMP #4$,R3 ;LAST WORD?
17223 105014 001374 BNE 10$ ;IF NOT, CONTINUE
17224 ;
17225 ; CHECK BYPASS AND BAD PARITY
17226 ;
17227 105016 013737 000114 002762 MOV @#114, SLOC00 ;SAVE PARITY VECTOR
17228 105024 012737 105060 000114 MOV #1$,@#114 ;POINT NEW VECTOR
17229 105032 052737 002101 177746 BIS #BIT10:BIT06:BIT00,CCR ;DATA AND TAG PAR., NO INT.
17230 105040 005037 003122 CLR @#15:LOC ;WRITE CYCLE
17231 105044 012737 001000 177746 MOV #BIT09,CCR ;SET BYPASS
17232 105052 005737 003122 TST @#15:LOC ;BYPASS WITH WRONG PARITY
17233 105056 000410 BR 2$ ;
17234 105060 062706 000004 1$ ADD #4,SP ;ADJUST STACK
17235 105064 104045 ERROR +4$ ;ERROR
17236 105066 032737 000340 177744 BIT #340,MSER ;MSER OK?
17237 105074 001001 BNE 2$ ;IF YES, BRANCH
17238 105076 104042 ERROR +42 ;BYPASS WRONG
17239 ;
17240 ; CHECK FORCE MISS AND BAD PARITY
17241 ;
17242 105100 005037 177744 2$ CLR MSER ;CLEAR MSER
17243 105104 005037 177746 CLR CCR ;CLEAR CCR
17244 105110 012737 105144 000114 MOV #3$,@#114 ;POINT NEW VECTOR
17245 105116 052737 002101 177746 BIS #BIT10:BIT06:BIT00,CCR ;DATA AND TAG PAR., NO INTER
17246 105124 005037 003122 CLR @#15:LOC ;FORCE MISS WITH PARITY
17247 105130 012737 000014 177746 MOV #14,CCR ;FORCE MISS
17248 105136 005737 003122 TST @#15:LOC ;ALLOCATE CACHE
17249 105142 000403 BR 4$ ;
17250 105144 104042 3$ ERROR +42 ;ADJUST STACK
17251 105146 062706 000004 ADD #4,SP ;CLEAR CCR
17252 105152 005037 177746 4$ CLR CCR ;RESTORE PARITY VECTOR
17253 105156 013737 002762 000114 MOV SLOC00,@#114 ;FLUSH CACHE
17254 105164 012737 000400 177746 MOV #BIT08,CCR
17255

```

COKDABO KDJ11 B CLUSTER MACY11 30(1046)
COKDAB.P11 05-APR-84 16:45

05 APR-84 16:45 PAGE 320
TEST - MEMORY SYSTEM ERROR REGISTER TEST

SEQ 0319

```

17256 .SBTTL TEST - MEMORY SYSTEM ERROR REGISTER TEST
17257 ;MEMORY SYSTEM ERROR REGISTER TEST - THIS TEST WILL VERIFY THE
17258 ;FUNCTIONALITY OF BITS <15> AND <7:5> OF THE MEMORY SYSTEM ERROR
17259 ;REGISTER. THIS TEST WILL USE THE WRITE WRONG PARITY BITS (BITS<10>
17260 ;AND <6> OF THE CCR) TO WRITE BAD PARITY INTO A LOCATION. THE
17261 ;LOCATION WILL THEN BE READ WITH CACHE TRAPS ENABLED AND THE MSER
17262 ;WILL BE CHECKED AFTER THE ABORT FOR THE CORRECT BIT(S) BEING SET.
17263 ;THIS WILL BE DONE FOR ALL COMBINATIONS OF BITS<10> AND <6> FOR WORD
17264 ;ACCESSES THEN REPEATED FOR ALL COMBINATIONS OF BITS<10> AND <6> FOR
17265 ;BYTE ACCESSES. THE TEST WILL THEN BE REPEATED A THIRD TIME FOR BYTE
17266 ;ACCESSES AND ABORTS DISABLED. THE MSER SHOULD CONTAIN BITS <7:5>
17267 ;SET TO 1'S AND BIT <15> A ZERO FOR ALL COMBINATIONS.
17268 ;
17269 ;BGNTST
17270 ;SAVE CONTENTS OF VECTOR 114
17271 ;SETUP VECTOR TO POINT TO ABORT ROUTINE
17272 ;INITIALIZE LOOP COUNTER
17273 ;DO UNTIL ALL WORD COMBINATIONS CHECKED
17274 ;.
17275 ;. INITIALIZE MSER
17276 ;. SETUP CCR FROM CCR TABLE
17277 ;. READ TEST LOCATION TO ALLOCATE LOCATION WITH DESIRED PARITY
17278 ;. CLEAR <10,6> FROM CCR
17279 ;. SETUP CCR FOR ABORT
17280 ;. READ TEST LOCATION ;THIS COULD CAUSE TRAP
17281 ;. IF EXPECTED DATA NE RECEIVED DATA THEN
17282 ;. ERROR IN SETTING MSER
17283 ;. ENDF
17284 ;. UPDATE LOOP COUNTER
17285 ;ENDDO
17286 ;INITIALIZE LOOP COUNTER
17287 ;DO UNTIL ALL BYTE COMBINATIONS CHECKED
17288 ;.
17289 ;. INITIALIZE MSER
17290 ;. SETUP CCR FROM CCR TABLE
17291 ;. READ TEST LOCATION TO ALLOCATE LOCATION WITH DESIRED PARITY
17292 ;. CLEAR <10,6> FROM CCR
17293 ;. SETUP CCR FOR ABORT
17294 ;. READ LOW BYTE TEST LOCATION
17295 ;. GET EXPECTED BYTE DATA FROM TABLE
17296 ;. IF EXPECTED DATA NE RECEIVED DATA THEN
17297 ;. ERROR IN SETTING MSER
17298 ;. ENDF
17299 ;. INITIALIZE MSER
17300 ;. READ HIGH BYTE TEST LOCATION
17301 ;. GET EXPECTED BYTE DATA FROM TABLE
17302 ;. IF EXPECTED DATA NE RECEIVED DATA THEN
17303 ;. ERROR IN SETTING MSER
17304 ;. ENDF
17305 ;. INCREMENT LOOP COUNTER
17306 ;ENDDO
17307 ;INITIALIZE LOOP COUNTER
17308 ;DO UNTIL ALL BYTE COMBINATIONS CHECKED
17309 ;.
17310 ;. INITIALIZE MSER
17311 ;. SETUP CCR FROM CCR TABLE
17312 ;. READ TEST LOCATION TO ALLOCATE LOCATION WITH DESIRED PARITY
17313 ;. CLEAR <10,6> FROM CCR
17314 ;. SETUP CCR FOR NO ABORTS

```

```

17312      ;.      READ LOW BYTE TEST LOCATION
17313      ;.      IF RECEIVED DATA NE TO #340 THEN
17314      ;.          ERROR IN SETTING MSER
17315      ;.      ENDIF
17316      ;.      INITIALIZE MSER
17317      ;.      READ HIGH BYTE TEST LOCATION
17318      ;.      IF RECEIVED DATA NE #340 THEN
17319      ;.          ERROR IN SETTING MSER
17320      ;.      ENDIF
17321      ;.      INCREMENT LOOP COUNTER
17322      ;ENDDO
17323      ;EXIT  TST
17324
17325      ;CCR TABLE:      0
17326      ;                  100
17327      ;                  2000
17328      ;                  2100
17329      ;EXPECTED WORD DATA:  0
17330      ;                  100300
17331      ;                  100040
17332      ;                  100340
17333      ;EXPECTED BYTE DATA:  0
17334      ;                  0
17335      ;                  100100
17336      ;                  100200
17337      ;                  100040
17338      ;                  100040
17339      ;                  100140
17340      ;                  100240
17341      ;ABORT ROUTINE: RTI
17342      ;
17343      ;ENDTST
17344      ;*****
17345      TST20:  SCOPE
17346      MOV     @#114, SLOC00      ;SAVE CONTENTS OF VECTOR 114
17347      MOV     @ABROUT,@#114    ;SETUP VECTOR TO POINT TO ABORT ROUTINE
17348      MOV     #4, R4            ;INITIALIZE LOOP COUNTER
17349      MOV     @CCRTBL,R0        ;GET ADDRESS OF CCR TABLE
17350      MOV     @EXPWDT,R1        ;GET ADDRESS OF EXPECTED DATA TABLE
17351      1$:     CLR     MSER        ;INITIALIZE MSER DATA
17352      MOV     (R0)+, CCR        ;SETUP CCR FROM CCR TABLE
17353      CLR     @#1STLOC          ;ALLOCATE CACHE LOC. WITH DESIRED PARITY
17354      MOV     @B1T07,CCR        ;SETUP CCR TO ABORT POSSIBLE BAD PARITY
17355      TST     @#1STLOC          ;READ TEST LOCATION
17356      CMP     (R1), MSER        ;IF RECEIVED DATA NOT EQUAL TO EXPECTED
17357      BEQ     ,3$              ;DATA THEN
17358      MOV     (R1), $GDDAT      ;SAVE PROPER MSER SETTING
17359      ERROR   ,3$              ;MSER NOT SET PROPERLY
17360      2$:     TST     (R1)+      ;INCREMENT POINTER THRU MSER TABLE
17361      TST     @#1STLOC+8192,    ;TO INSURE MISS ON THE NEXT LOOP
17362      SOB     R4, 1$           ;LOOP UNTIL ALL COMBINATIONS CHECKED
17363      ;CHECK BYTE OPERATIONS NOW
17364      MOV     #4, R4            ;INITIALIZE LOOP COUNTER
17365      MOV     @CCRTBL,R0        ;GET ADDRESS OF CCR TABLE
17366      MOV     @EXPBDT,R1        ;GET ADDRESS OF EXPECTED BYTE DATA TABLE
17367      3$:     CLR     MSER        ;INITIALIZE MSER

```

100

COKDABD KDJ11 8 CLUSTER MAC111 30(1046) 05-APR-84 16:45 PAGE 322
COKDAB.P11 05-APR-84 16:45 TEST MEMORY SYSTEM ERROR REGISTER TEST

SEQ 0321

17368	105316	005737	003122		TST	@#TSTLOC		;ALLOCATE TO HAVE WRITE BYTE HIT
17369	105322	011037	177746		MOV	(R0), CCR		;SETUP CCR FROM TABLE
17370	105326	105037	003122		CLRB	@#TSTLOC		;ALLOCATE CACHE LOC. WITH DESIRED PARITY
17371	105332	012737	000200	177746	MOV	#BIT07, CCR		;SETUP CCR TO ABORT POSSIBLE BAD PARITY
17372	105340	105737	003122		TSTB	@#TSTLOC		;READ LOW BYTE OF TEST LOCATION
17373	105344	021137	177744		CMP	(R1), MSER		;IF RECEIVED DATA NOT EQUAL TO EXPECTED
17374	105350	001403			BEQ	4\$		;THEN
17375	105352	011137	001124		MOV	(R1), \$GDDAT		;SAVE PROPER MSER SETTING
17376	105356	104035			ERROR	+35		;MSER NOT SET PROPERLY
17377	105360	005721		4\$:	TST	(R1),		;INCREMENT POINTER THRU MSER TABLE
17378	105362	005037	177744		CLR	MSER		;INITIALIZE MSER
17379	105366	005737	003122		TST	@#TSTLOC		;ALLOCATE TO HAVE WRITE BYTE HIT
17380	105372	012037	177746		MOV	(R0)+, CCR		;SETUP CCR FROM TABLE
17381	105376	105037	003123		CLRB	@#TSTLOC+1		;ALLOCATE CACHE LOC. WITH DESIRED PARITY
17382	105402	012737	000200	177746	MOV	#BIT07, CCR		;SETUP CCR TO ABORT POSSIBLE BAD PARITY
17383	105410	105737	003123		TSTB	@#TSTLOC+1		;READ HIGH BYTE OF TEST LOCATION
17384	105414	021137	177744		CMP	(R1), MSER		;IF RECEIVED DATA NOT EQUAL TO EXPECTED
17385	105420	001403			BEQ	5\$		;THEN
17386	105422	011137	001124		MOV	(R1), \$GDDAT		;SAVE PROPER MSER SETTING
17387	105426	104035			ERROR	+35		;MSER NOT SET PROPERLY
17388	105430	005721		5\$:	TST	(R1),		;INCREMENT POINTER THRU MSER TABLE
17389	105432	077451			SOB	R4, 3\$		;DO UNTIL ALL BYTE COMBINATIONS CHECKED
17390					;REPEAT WITHOUT ABORT			
17391	105434	012704	000003		MOV	#3, R4		;INITIALIZE LOOP COUNTER
17392	105440	012700	105614		MOV	@CCRTBL+2,R0		;GET ADDRESS OF CCR TABLE
17393	105444	005037	177744		CLRB	MSER		;INITIALIZE MSER
17394	105450	005737	003122		TST	@#TSTLOC		;ALLOCATE CACHE LOC.
17395	105454	011037	177746		MOV	(R0), CCR		;SETUP CCR FROM TABLE
17396	105460	105037	003122		CLRB	@#TSTLOC		;ALLOCATE CACHE LOC. WITH DESIRED PARITY
17397	105464	012737	000001	177746	MOV	#BIT00, CCR		;SETUP CCR TO NOT ABORT
17398	105472	105737	003122		TSTB	@#TSTLOC		;READ LOW BYTE OF TEST LOCATION
17399	105476	022737	000340	177744	CMP	#340, MSER		;IF RECEIVED DATA NOT EQUAL TO EXPECTED
17400	105504	001404			BEQ	7\$		;THEN
17401	105506	012737	000340	001124	MOV	#340, \$GDDAT		;SAVE PROPER MSER SETTING
17402	105514	104035			ERROR	+35		;MSER NOT SET PROPERLY
17403	105516	005037	177744		7\$:	CLR	MSER	;INITIALIZE MSER
17404	105522	005737	003122		TST	@#TSTLOC		;ALLOCATE CACHE LOC.
17405	105526	012037	177746		MOV	(R0)+, CCR		;SETUP CCR FROM TABLE
17406	105532	105037	003123		CLRB	@#TSTLOC+1		;ALLOCATE CACHE LOC. WITH DESIRED PAR.
17407	105536	012737	000001	177746	MOV	#BIT00, CCR		;SETUP CCR TO NOT ABORT
17408	105544	105737	003123		TSTB	@#TSTLOC+1		;READ HIGH BYTE OF TEST LOCATION
17409	105550	022737	000340	177744	CMP	#340, MSER		;IF RECEIVED DATA NOT EQUAL TO EXPECTED
17410	105556	001404			BEQ	8\$		;THEN
17411	105560	012737	000340	001124	MOV	#340, \$GDDAT		;SAVE PROPER MSER SETTING
17412	105566	104035			ERROR	+35		;MSER NOT SET PROPERLY
17413	105570	077453		8\$:	SOB	R4, 6\$		;DO UNTIL ALL BYTE COMBINATIONS CHECKED
17414	105572	005037	177744		CLR	MSER		;CLEAR ERROR REGISTER
17415	105576	013737	002762	000114	MOV	\$LOC00, @#114		;RESTORE VECTOR 114
17416	105604	005037	177746		CLR	CCR		;CLEAR ALL BIT IN CCR
17417	105610	000421			BR	TST21		;GO TO NEXT TEST
17418								
17419								
17420	105612	000000			CCRTBL :	.WORD	0	
17421	105614	000100				.WORD	100	
17422	105616	000200				.WORD	2000	
17423	105620	0002100				.WORD	2100	

COROAD0 RDJ11 B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 323
COROAD.P11 05-APR-84 16:45 TEST MEMORY SYSTEM ERROR REGISTER TEST

SEQ 0322

17424				
17425	105622	000000	EXPWDT: .WORD	0
17426	105624	100300	.WORD	100300
17427	105626	100040	.WORD	100040
17428	105630	100340	.WORD	100340
17429				
17430	105632	000000	EXPBDT: .WORD	0
17431	105634	000000	.WORD	0
17432	105636	100100	.WORD	100100
17433	105640	100200	.WORD	100200
17434	105642	100040	.WORD	100040
17435	105644	100040	.WORD	100040
17436	105646	100140	.WORD	100140
17437	105650	100240	.WORD	100240
17438				
17439	105652	000002	ABROUT: RTI	
17440				
17441				

COKDABO KD,J11-B CLUSTER MACY11 30(1046)
COKDAB.P11 05-APR-84 16:45

05-APR-84 16:45 PAGE 324
TEST - CHECK PARITY ABORTS BLOCKED BY NON-EXISTENT MEMORY ABORT

SEQ 0323

```

17442 .SBTTL TEST - CHECK PARITY ABORTS BLOCKED BY NON-EXISTENT MEMORY ABORT
17443 ;CHECK PARITY ABORTS BLOCKED BY NON-EXISTENT MEMORY ABORT - THIS TEST WILL
17444 ;VERIFY THAT IF A PARITY ERROR OCCURS ON THE SAME ADDRESS REFERENCE AS A
17445 ;NON-EXISTENT MEMORY ERROR THAT THE CACHE DATA PATH GATE ARRAY BLOCKS THE
17446 ;PARITY ERROR TO THE J-11 CHIP SET. THIS WILL BE DONE BY USING THE DIAGNOSTIC
17447 ;BIT TO CAUSE A PARITY ERROR IN A CACHE REFERENCE THAT DOES NOT HAVE A
17448 ;CORRESPONDING ADDRESS IN MAIN MEMORY. THE ADDRESS WILL THEN BE READ CAUSING
17449 ;BOTH A CACHE PARITY ERROR AND A NON-EXISTENT MEMORY ERROR. TO AVOID HAVING
17450 ;TO SIZE THE ENTIRE MEMORY TO FIND A NON-EXISTENT MEMORY ADDRESS THIS TEST
17451 ;WILL USE THE LARGEST NON-I/O ADDRESS (17757776). THE TEST WILL FIRST READ
17452 ;THIS ADDRESS AND IF A NXM TRAP OCCURS THE TEST WILL BE DONE. IF THE ACCESS
17453 ;TO THIS ADDRESS DOES NOT TRAP THEN THE TEST WILL BE SKIPPED.
17454 ;
17455 ;BGNTEST
17456 ;SAVE CONTENTS OF VECTOR 4
17457 ;SAVE CONTENTS OF VECTOR 114
17458 ;LET VECTOR 4 POINT TO CONTINUE TESTING (A:)
17459 ;LET PAR6 = #177400
17460 ;ACCESS ADDRESS 157776 (PHYSICAL 17757776)
17461 ;IF NO TRAP THEN
17462 ;. GOTO ENDTST
17463 ;ENDIF
17464 ;A:
17465 ;SET DIAGNOSTIC AND WRITE WRONG PARITY BITS IN CCR
17466 ;WRITE ADDRESS 157776
17467 ;CLEAR CCR
17468 ;SET PARITY ERROR ABORT BIT IN CCR
17469 ;LET VECTOR 4 POINT TO CONTINUE TESTING (B:)
17470 ;LET VECTOR 114 POINT TO NXM-PARITY ERROR ROUTINE
17471 ;READ ADDRESS 157776
17472 ;IF NO TRAP THEN
17473 ;. ERROR IN ABORT LOGIC
17474 ;ENDIF
17475 ;B:
17476 ;CLEAR CCR
17477 ;RESTORE CONTENTS OF VECTOR 114
17478 ;RESTORE CONTENTS OF VECTOR 4
17479 ;EXIT TST
17480 ;
17481 ;
17482 ;NXM-PARITY ERROR ROUTINE: RESET STACK AFTER TRAP
17483 ; PARITY ABORT NOT BLOCKED BY NXM
17484 ; GOTO B:
17485 ;ENDTST
17486 ;*****
17487 105654 000004 TST21: SCOPE
17488 105656 013737 000004 002762 MOV #4, SLOC00 ;SAVE CONTENTS OF VECTOR 4
17489 105664 013737 000114 002764 MOV #114, SLOC01 ;SAVE CONTENTS OF VECTOR 114
17490 105672 012737 105720 000004 MOV #15, #4 ;LET VECTOR 4 POINT TO CONTINUE TESTING
17491 105700 012737 177400 172354 MOV #177400,KIPAR6 ;LET PAR6 = OFFSET TO HIGHEST MEMORY
17492 105706 005237 177572 INC SRO ;TURN ON MMU
17493 105712 005737 157776 LST #157776 ;ACCESS ADDRESS 17757776
17494 105716 000426 BR ABOEXT ;IF NO TRAP SKIP TEST
17495 105720 062706 000004 1$: ADD #4, SP ;RESET STACK AFTER TRAP
17496 105724 012737 000102 177746 MOV #102, CCR ;SET DIAG. AND WRITE WRONG PARITY BITS
17497 105732 005037 157776 CLR #157776 ;WRITE TO ADDRESS 17757776

```


SEQ 0324

[illegible]

COKDABO KDJ11 B CLUSTER MACY11 30(1046)
COKDAB,P11 05-APR 84 16:45

05-APR-84 16:45 PAGE 326
TEST - MULTIPROCESSING INSTRUCTION TESTS

SEQ 0325

```

17517 ;SBTTL TEST - MULTIPROCESSING INSTRUCTION TESTS
17518 ;MULTIPROCESSING INSTRUCTION TESTS - THIS TEST WILL VERIFY THAT THE MULTI-
17519 ;PROCESSING INSTRUCTIONS DO A BYPASS OF THE CACHE. THIS TEST WILL NOT VERIFY
17520 ;THE REST OF THE FUNCTIONALITY OF THESE INSTRUCTIONS BECAUSE THAT WILL ALREADY
17521 ;HAVE BEEN CHECKED IN THE BASE INSTRUCTION TESTS. THE TEST WILL FIRST ALLOCATE
17522 ;AN ADDRESS IN CACHE THEN A TSTSET INSTRUCTION WILL BE DONE AT THAT ADDRESS.
17523 ;A HIT SHOULD BE RECORDED ON THE ACCESS. NEXT, THE ADDRESS WILL BE READ AND
17524 ;A MISS SHOULD BE RECORDED BECAUSE THE FORCED BYPASS ON THE TSTSET INSTRUCTION
17525 ;SHOULD HAVE INVALIDATED THE CACHE ENTRY. THE SAME SEQUENCE WILL THEN BE
17526 ;REPEATED FOR THE WRTCLK INSTRUCTION.
17527 ;
17528 ;BGN1ST
17529 ;READ TEST LOCATION TO ALLOCATE CACHE
17530 ;DO TSTSET INSTRUCTION
17531 ;IF HIT/MISS REGISTER BIT 3 NOT SET OR BIT 2 SET THEN
17532 ;. ERROR IN MULTI PROCESSOR HOOKS
17533 ;ENDIF
17534 ;READ TEST LOCATION (ALSO ALLOCATES CACHE FOR WRTCLK)
17535 ;DO WRTCLK INSTRUCTION
17536 ;IF HIT/MISS REGISTER BIT 3 NOT SET OR BIT 2 SET THEN
17537 ;. ERROR IN MULTI-PROCESSOR HOOKS
17538 ;ENDIF
17539 ;READ TEST LOCATION
17540 ;DO ASRB INSTRUCTION
17541 ;IF HIT/MISS REGISTER BIT 3 NOT SET OR BIT 2 SET THEN
17542 ;. ERROR IN MULTI-PROCESSOR HOOKS
17543 ;ENDIF
17544 ;END1ST
17545 ;NOTE: THE CODE IS POSITION DEPENDENT
17546
17547 ;*****
17548 106032 000004
17549 106034 042737 001000 177520
17550 106042 005737 003122
17551
17552 106046 007237
17553 106050 003122
17554 106052 013737 177752 110504
17555 106060 032737 000010 110504
17556 106066 001001
17557 106070 104045
17558 106072 032737 000004 110504
17559 106100 001404
17560 106102 013737 106046 001126
17561 106110 104041
17562
17563 106112 007337
17564 106114 003122
17565 106116 013737 177752 110504
17566 106124 032737 000010 110504
17567 106132 001001
17568 106134 104045
17569 106136 032737 000004 110504
17570 106144 001404
17571 106146 013737 106112 001126
17572 106154 104041

TST22: SCOPE
      BIC      01000,BCSR
      TST      TSTLOC
      TSTSET   TSTLOC
17573:  .WORD     7237
      .WORD     TSTLOC
      MOV      HITMISS,RECDAT
      BIT      08BIT3,RECDAT
      BNE      17574
      ERROR    145
17574:  BIT      08BIT2,RECDAT
      BEQ      17575
      MOV      0011, $HDDAT
      ERROR    141
17575:  WRTCLK   TSTLOC
27576:  .WORD     7237
      .WORD     TSTLOC
      MOV      HITMISS,RECDAT
      BIT      08BIT3,RECDAT
      BNE      17577
      ERROR    145
17577:  BIT      08BIT2,RECDAT
      BEQ      17578
      MOV      0011, $HDDAT
      ERROR    141

;DISABLE HALT ON BREAK
;READ TEST LOCATION TO ALLOCATE CACHE
;DO TSTSET INSTRUCTION
;THESE NEXT TWO LOCATIONS ARE THE TSTSET
;INSTR. BECAUSE THE ASSEMBLER WAS NOT READY
;STORE REGISTER
;IF HIT/MISS REGISTER BIT 3 NOT SET
;THEN
;ERROR IN RECORDING HITS IN HIT/MISS
;IF HIT/MISS REGISTER BIT 2 SET
;THEN
;SAVE OPCODE
;MULTI-PROCESSOR HOOK INSTRUCTION DOESN'T CAUSE
;DO WRTCLK INSTRUCTION
;THESE NEXT TWO WORDS ARE THE WRTCLK
;INSTR. BECAUSE THE ASSEMBLER WAS NOT READY
;STORE REGISTER
;IF HIT/MISS REGISTER BIT 3 NOT SET
;THEN
;ERROR IN RECORDING HITS IN HIT/MISS
;IF HIT/MISS REGISTER BIT 2 SET
;THEN
;SAVE OPCODE
;MULTI-PROCESSOR HOOK INSTRUCTION DOESN'T CAUSE

```

C10

COKDABO RDJ11 B CLUSTER MACY11 30(1046) 05 APR-84 16:45 PAGE 327
COKDAB.P11 05-APR-84 16:45 TEST MULTIPROCESSING INSTRUCTION TESTS

St.Q 0326

PC	OPCODE	OP	RS	RT	RD	DATA	COMMENT
17573	106156	106237	003122			4\$:	ASRB TSTLOC ;THE THIRD BYPASS INSTRUCTION
17574	106162	013737	177752	110504			MOV HITMISS,RECDAT ;STORE REGISTER
17575	106170	032737	000010	110504			BIT @BIT3,RECDAT ;IF HIT/MISS REGISTER BIT 3 NOT SET
17576	106176	001001					BNE 5\$;THEN
17577	106200	104045					ERROR +45 ;ERROR IN RECORDING HITS IN HIT/MISS
17578	106202	052737	001000	177520		5\$:	BIS @1000,RCNR ;ENABLE HALT ON BREAK
17579	106210	032737	000004	110504			BIT @BIT2,RECDAT ;IF HIT/MISS REGISTER BIT 2 SET
17580	106216	001404					BEQ TST23 ;EXIT TEST
17581	106220	013737	106156	001126			MOV @04\$, \$BDDAT ;SAVE OPCODE
17582	106226	104041					ERROR +41 ;MULTI-PROCESSOR HOOK INSTRUCTION DOESN'T CAUSE
17583							
17584							
17585							

D10

COKDABO KDJ11 B CLUSTER MACY11 30(1046)
COKDAB.P11 05-APR-84 16:45

05 APR-84 16:45 PAGE 328
TEST DATA STORE RAM TESTS

SEQ 0327

17586
17587
17588
17589
17590
17591
17592
17593
17594
17595
17596
17597
17598
17599
17600
17601
17602
17603
17604
17605
17606
17607
17608
17609
17610
17611
17612
17613
17614
17615
17616
17617
17618
17619
17620
17621
17622
17623
17624
17625
17626
17627
17628
17629
17630
17631
17632
17633
17634
17635
17636
17637
17638
17639 106230 000004
17640 106232 004737 132156
17641 106236 012737 002000 172354

```

;SHTTL TEST - DATA STORE RAM TESTS
;DATA STORE RAM TESTS - THERE ARE TWO TESTS FOR THE DATA STORE RAM.
;THE FIRST TEST WILL BE A NO DUAL ADDRESSING TEST AND THE SECOND
;TEST WILL BE A DATA RELIABILITY TEST. THE NO DUAL ADDRESSING TEST
;WILL FIRST WRITE EACH WORD OF THE CACHE RAM WITH ITS WORD ADDRESS
;AND VERIFY THE CONTENTS WITH A READ OF EACH ADDRESS. THEN EACH
;BYTE WILL BE WRITTEN WITH ITS ADDRESS AND CHECKED. THE DATA
;RELIABILITY TEST THAT WILL BE USED IS A TEST CALLED MOVING INVERSIONS.
;
;RGNTST 1
;SETUP MMU REGISTERS TO HAVE BYPASS ON KERNAL SPACE AND NO
;    BYPASS ON USER SPACE
;LET PS EQUAL KERNAL FOR CURRENT MODE AND USER FOR PREVIOUS
;    MODE
;ENABLE MMU
;GET FIRST ADDRESS OF 4K WORD DATA BUFFER
;CLEAR DATA TO BE WRITTEN
;SET DIAGNOSTIC BIT (BIT<1>) IN CCR
;DO UNTIL DATA TO BE WRITTEN EQUALS 20000(8)
;..    WRITE DATA TO ADDRESS
;..    ADD 2 TO ADDRESS
;..    ADD 2 TO DATA TO BE WRITTEN
;ENDDO
;GET FIRST ADDRESS OF 4K WORD DATA BUFFER
;CLEAR EXPECTED DATA
;DO UNTIL EXPECTED DATA EQUALS 20000(8)
;..    READ ADDRESS
;..    IF RECEIVED DATA NE EXPECTED DATA THEN
;..        GOTO DATA STORE PARITY ERROR ROUTINE
;..    ENDIF
;..    ADD 2 TO EXPECTED DATA
;..    ADD 2 TO ADDRESS
;ENDDO
;GET FIRST ADDRESS OF 8K BYTE DATA BUFFER
;CLEAR DATA TO BE WRITTEN
;DO UNTIL ALL BYTES CHECKED
;..    WRITE DATA TO ADDRESS
;..    ADD 1 TO ADDRESS
;..    ADD 1 TO DATA TO BE WRITTEN
;ENDDO
;GET FIRST ADDRESS OF 8K BYTE DATA BUFFER
;CLEAR EXPECTED DATA
;DO UNTIL EXPECTED DATA EQUALS 20000(8)
;..    READ ADDRESS
;..    IF RECEIVED DATA NE EXPECTED DATA THEN
;..        GOTO DATA STORE PARITY ERROR ROUTINE
;..    ENDIF
;..    ADD 1 TO EXPECTED DATA
;..    ADD 1 TO ADDRESS
;ENDDO
;ENDTST
;
;*****
TSTP3:  SCOPE
        JSR    PC,    INITMM
        MOV    02000, KIPAR6
        ;SETUP MMU REGISTERS
        ;LET PAR6 MAP TO ADDRESS 200000

```

Address	Offset	Value	Label	Instruction	Comment
17642	106244	005237	177572	INC SRO	TURN ON MEMORY MANAGEMENT
17643	106250	012702	140000	MOV #140000,R2	GET FIRST ADDRESS OF 4K WORD BUFFER
17644	106254	005001		CLR R1	INIT DATA TO BE WRITTEN
17645	106256	052737	000002 177746	BIS #BIT01,CCR	SET DIAG BIT IN CASE NO MEMORY PRESENT
17646	106264	042737	001000 177520	BIC #1000,BCSR	DISABLE HALT ON BREAK
17647	106272	020127	020000	CMP R1, #20000	DO UNTIL ALL ADDRESSES WRITTEN
17648	106276	001404		BEQ 2\$	IF DONE GO CHECK DATA
17649	106300	010122		MOV R1, (R2)+	WRITE DATA TO ADDRESS
17650	106302	062701	000002	ADD #2, R1	UPDATE DATA BY 2 (WORD BOUNDARY)
17651	106306	000771		BR 1\$	ENDDD
17652	106310	042737	000002 177746	BIC #BIT01,CCR	CLEAR DIAGNOSTIC BIT
17653	106316	012702	140000	MOV #140000,R2	GET FIRST ADDRESS OF 4K WORD BUFFER
17654	106322	005001		CLR R1	INIT DATA TO BE CHECKED
17655	106324	020127	020000	CMP R1, #20000	DO UNTIL ALL ADDRESSES CHECKED
17656	106330	001426		BEQ 5\$	IF DONE GO DO BYTES
17657	106332	010137	001122	MOV R1, \$BDADR	STORE FOR ERRORS
17658	106336	052737	100000 001122	BIS #BIT15, \$BDADR	
17659	106344	012237	110504	MOV (R2)+, RECDAT	READ TEST LOCATION
17660	106350	013737	177752 001160	MOV HITMIS, \$TMPO	STORE REGISTER
17661	106356	032737	000020 001160	BIT #BIT04,\$TMPO	HIT?
17662	106364	001001		BNE 100\$	IF YES, BRANCH
17663	106366	104045		ERROR .45	DATA RAM ERROR
17664	106370	020137	110504	CMP R1, RECDAT	IF RECIEVED DATA NOT EQUAL TO EXPECTED
17665	106374	001401		BEQ 4\$	THEN
17666	106376	104043		ERROR .43	DATA RAM ERROR
17667	106400	062701	000002	ADD #2, R1	UPDATE EXPECTED DATA BY 2
17668	106404	000747		BR 3\$	ENDDD
17669	106406	012702	140000	MOV #140000,R2	GET FIRST ADDRESS OF 4K WORD BUFFER
17670	106412	005001		CLR R1	INIT DATA TO BE WRITTEN
17671	106414	052737	000002 177746	BIS #BIT01,CCR	SET DIAG BIT IN CASE NO MEMORY PRESENT
17672	106422	020127	020000	CMP R1, #20000	DO UNTIL ALL ADDRESSES WRITTEN
17673	106426	001404		BEQ 7\$	IF DONE GO CHECK DATA
17674	106430	110122		MOVB R1, (R2)+	WRITE DATA TO ADDRESS
17675	106432	062701	000001	ADD #1, R1	UPDATE DATA BY 1 (BYTE BOUNDARY)
17676	106436	000771		BR 6\$	ENDDD
17677	106440	042737	000002 177746	BIC #BIT01,CCR	CLEAR DIAGNOSTIC BIT
17678	106446	012702	140000	MOV #140000,R2	GET FIRST ADDRESS OF 4K WORD BUFFER
17679	106452	005001		CLR R1	INIT DATA TO BE CHECKED
17680	106454	020127	020000	CMP R1, #20000	DO UNTIL ALL ADDRESSES CHECKED
17681	106460	001426		BEQ 10\$	IF DONE EXIT TEST
17682	106462	010137	001122	MOV R1, \$BDADR	STORE FOR ERRORS
17683	106466	052737	100000 001122	BIS #BIT15, \$BDADR	
17684	106474	112237	110504	MOVB (R2)+, RECDAT	READ TEST LOCATION
17685	106500	013737	177752 001160	MOV HITMIS, \$TMPO	STORE REGISTER
17686	106506	032737	000020 001160	BIT #BIT04,\$TMPO	HIT?
17687	106514	001001		BNE 101\$	IF YES, BRANCH
17688	106516	104045		ERROR .45	DATA RAM ERROR
17689	106520	120137	110504	CMPB R1, RECDAT	IF RECIEVED DATA NOT EQUAL TO EXPECTED
17690	106524	001401		BEQ 9\$	THEN
17691	106526	104043		ERROR .43	DATA RAM ERROR
17692	106530	062701	000001	ADD #1, R1	UPDATE EXPECTED DATA BY 1
17693	106534	000747		BR 8\$	ENDDD
17694	106536	052737	001000 177520	B	

G10

COKDABO KDJ11 B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 331
 COKDAB.P11 05-APR-84 16:45 TEST TAG STORE RAM TESTS

SEQ 0330

```

17753 ;*****
17754 106550 000004 TST24: SCOPE
17755 106552 013737 000004 001160 MOV #04,$TMP0 ;STORE TIMEOUT VECTOR
17756 106560 012737 107304 000004 MOV #20,$074 ;POINT NEW
17757 106566 012737 140000 002766 MOV #140000,LOWADD ;INITIALIZE LOW ADDRESS (USE PAR6)
17758 106574 004737 132156 JSR PC, INITMM ;INITIALIZE MMU
17759 106600 005237 177572 INC SR0 ;ENABLE MMU
17760 106604 012737 000020 172516 MOV #BIT04,MMR3 ;ENABLE 22-BIT MAPPING
17761 106612 012737 177770 003114 MOV #10, LOOPIN ;DO UNTIL ALL BLOCKS TESTED
17762 106620 012701 000000 1$: MOV #0,R1 ;DO IN 2 WORDS TO AVOID PMI MEMORY
17763 106624 042737 001000 177520 9$: BIC #1000,BCSR ;DISABLE HALT ON BREAK
17764 106632 005037 172354 CLR KIPAR6 ;SET UP PAR6 FOR THIS TEST
17765 106636 012737 000402 177746 MOV #402, CCR ;FLUSH CACHE AND SET DIAG BIT
17766 106644 013737 002766 110524 MOV LOWADD, CURADD ;GET FIRST ADDRESS IN CURRENT BLOCK
17767 106652 012737 177400 003112 MOV #400, ALLCTR ;DO UNTIL ALL ADDRESSES ALLOCATED
17768 106660 022737 002000 172354 2$: CMP #2000, KIPAR6 ;IF ADDRESS LESS THAN 32K
17769 106666 002413 BLT 3$ ;THEN
17770 106670 052737 100000 172314 BIS #BIT15,KIPDR6 ;SET BYPASS
17771 106676 017702 001622 MOV #CURADD,R2 ;STORE CURRENT DATA
17772 106702 042737 100000 172314 BIC #BIT15,KIPDR6 ;ALLOCATE NEXT ACCESS
17773 106710 010277 001610 MOV R2,CURADD ;WRITE ALLOCATE
17774 106714 000402 BR 4$ ;ELSE
17775 106716 005077 001602 3$: CLR #CURADD ;WRITE CURRENT ADDRESS
17776 106722 062737 000002 110524 4$: ADD #2, CURADD ;UPDATE CURRENT ADDRESS
17777 106730 062737 000200 172354 ADD #200, KIPAR6
17778 106736 022737 177600 172354 CMP #177600,KIPAR6 ;REACHED I/O PAGE?
17779 106744 001003 BNE 10$ ;BRANCH IF NOT
17780 106746 162737 000200 172354 SUB #200,KIPAR6 ;DON'T UPDATE PAR FOR I/O PAGE
17781 106754 005237 003112 10$: INC ALLCTR ;IF ALL ADDRESSES ALLOCATED
17782 106760 002737 BLT 2$ ;ENDDO
17783 106762 052737 000004 177746 BIS #BIT02, CCR ;RUN WITH FORCE MISS
17784 106770 013737 002766 002770 MOV LOWADD, GOODAD ;INITIALIZE GOOD ADDRESS
17785 106776 060137 002770 ADD R1, GOODAD ;
17786 107002 012737 140000 110524 MOV #140000,CURADD ;GET FIRST ADDRESS
17787 107010 060137 110524 ADD R1, CURADD ;START WITH 1 OR 2 LOCATION
17788 107014 005037 172354 CLR KIPAR6 ;
17789 107020 072127 000006 ASH #6,R1 ;
17790 107024 060137 172354 ADD R1,KIPAR6 ;MAKE SURE ON THE RIGHT BOUNDARY
17791 107030 012737 177600 003112 MOV #200, ALLCTR ;DO UNTIL ALL LOCATIONS CHECKED
17792 107036 012737 177770 110500 5$: MOV #10, DCOUNT ;DO UNTIL ADDRESS IN EACH BLOCK CHECKED
17793 107044 013737 110524 001122 6$: MOV CURADD, $BDADR ;IN CASE OF ERRORS
17794 107052 042737 160000 001122 BIC #160000,$BDADR ;CLEAR PAR BITS
17795 107060 042737 000004 177746 BIC #BIT02, CCR ;CLEAR FORCE MISS
17796 107066 005777 001432 TST #CURADD ;READ CURRENT ADDRESS
17797 107072 013737 177752 110504 MOV HITMIS, RECDAT ;STORE REGISTER
17798 107100 032737 000004 110504 BIT #BIT2, RECDAT ;IF ACCESS WAS A HIT
17799 107106 001406 BEQ 7$ ;THEN
17800 107110 023737 110524 002770 CMP CURADD, GOODAD ;IF CURRENT ADDRESS NOT EQUAL TO GOOD
17801 107116 001407 BEQ 8$ ;ADDRESS THEN
17802 107120 104046 ERROR 146 ;ERROR IN TAG STORE
17803 107122 000405 BR 8$ ;
17804 107124 023737 110524 002770 7$: CMP CURADD, GOODAD ;IF CURRENT ADDRESS EQUAL GOOD ADDRESS
17805 107132 001001 BNE 8$ ;THEN
17806 107134 104047 ERROR 147 ;ERROR IN TAG STORE
17807 107136 062737 001000 110524 8$: ADD #1000, CURADD ;UPDATE TO NEXT BLOCK
17808 107144 052737 000004 177746 BIS #BIT02, CCR ;RUN WITH FORCE MISS

```

H10

COKDABO KDJ11 B CUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 332
COKDAB.P11 05-APR-84 16:45 TEST - TAG STORE RAM TESTS

SEQ 0331

PC	Op	Op1	Op2	Op3	Op4	Op5	Op6	Op7	Op8	Op9	Op10	Op11	Op12	Op13	Op14	Op15	Op16	Op17	Op18	Op19	Op20	Op21	Op22	Op23	Op24	Op25	Op26	Op27	Op28	Op29	Op30	Op31	Op32	Op33	Op34	Op35	Op36	Op37	Op38	Op39	Op40	Op41	Op42	Op43	Op44	Op45	Op46	Op47	Op48	Op49	Op50	Op51	Op52	Op53	Op54	Op55	Op56	Op57	Op58	Op59	Op60	Op61	Op62	Op63	Op64	Op65	Op66	Op67	Op68	Op69	Op70	Op71	Op72	Op73	Op74	Op75	Op76	Op77	Op78	Op79	Op80	Op81	Op82	Op83	Op84	Op85	Op86	Op87	Op88	Op89	Op90	Op91	Op92	Op93	Op94	Op95	Op96	Op97	Op98	Op99	Op100	Op101	Op102	Op103	Op104	Op105	Op106	Op107	Op108	Op109	Op110	Op111	Op112	Op113	Op114	Op115	Op116	Op117	Op118	Op119	Op120	Op121	Op122	Op123	Op124	Op125	Op126	Op127	Op128	Op129	Op130	Op131	Op132	Op133	Op134	Op135	Op136	Op137	Op138	Op139	Op140	Op141	Op142	Op143	Op144	Op145	Op146	Op147	Op148	Op149	Op150	Op151	Op152	Op153	Op154	Op155	Op156	Op157	Op158	Op159	Op160	Op161	Op162	Op163	Op164	Op165	Op166	Op167	Op168	Op169	Op170	Op171	Op172	Op173	Op174	Op175	Op176	Op177	Op178	Op179	Op180	Op181	Op182	Op183	Op184	Op185	Op186	Op187	Op188	Op189	Op190	Op191	Op192	Op193	Op194	Op195	Op196	Op197	Op198	Op199	Op200	Op201	Op202	Op203	Op204	Op205	Op206	Op207	Op208	Op209	Op210	Op211	Op212	Op213	Op214	Op215	Op216	Op217	Op218	Op219	Op220	Op221	Op222	Op223	Op224	Op225	Op226	Op227	Op228	Op229	Op230	Op231	Op232	Op233	Op234	Op235	Op236	Op237	Op238	Op239	Op240	Op241	Op242	Op243	Op244	Op245	Op246	Op247	Op248	Op249	Op250	Op251	Op252	Op253	Op254	Op255	Op256	Op257	Op258	Op259	Op260	Op261	Op262	Op263	Op264	Op265	Op266	Op267	Op268	Op269	Op270	Op271	Op272	Op273	Op274	Op275	Op276	Op277	Op278	Op279	Op280	Op281	Op282	Op283	Op284	Op285	Op286	Op287	Op288	Op289	Op290	Op291	Op292	Op293	Op294	Op295	Op296	Op297	Op298	Op299	Op300	Op301	Op302	Op303	Op304	Op305	Op306	Op307	Op308	Op309	Op310	Op311	Op312	Op313	Op314	Op315	Op316	Op317	Op318	Op319	Op320	Op321	Op322	Op323	Op324	Op325	Op326	Op327	Op328	Op329	Op330	Op331	Op332	Op333	Op334	Op335	Op336	Op337	Op338	Op339	Op340	Op341	Op342	Op343	Op344	Op345	Op346	Op347	Op348	Op349	Op350	Op351	Op352	Op353	Op354	Op355	Op356	Op357	Op358	Op359	Op360	Op361	Op362	Op363	Op364	Op365	Op366	Op367	Op368	Op369	Op370	Op371	Op372	Op373	Op374	Op375	Op376	Op377	Op378	Op379	Op380	Op381	Op382	Op383	Op384	Op385	Op386	Op387	Op388	Op389	Op390	Op391	Op392	Op393	Op394	Op395	Op396	Op397	Op398	Op399	Op400	Op401	Op402	Op403	Op404	Op405	Op406	Op407	Op408	Op409	Op410	Op411	Op412	Op413	Op414	Op415	Op416	Op417	Op418

COKDABO RDJ11 B CLUSTER MACY11 50(1046) 05-APR-84 16:45 PAGE 333
 COKDAB.P11 05-APR-84 16:45 TEST STANDALONE MODE TEST

SEQ 0332

```

17834 .SBTTL TEST - STANDALONE MODE TEST
17835 ;THIS TEST VERIFIES THAT NMX CAN BE CREATED IN STANDALONE MODE.
17836 ;
17837 ;ALLOCATE INSTRUCTIONS IN CACHE
17838 ;GUARANTEE MISS ON TEST LOCATION
17839 ;IN STANDALONE MODE ACCESS TEST LOCATION
17840 ;IF NO TIMEOUT THEN
17841 ;. ERROR
17842 ;ENDIF
17843 ;
17844 ;*****
17845 107306 000004 TST25: SCOPE
17846 107310 012737 000400 177746 MOV #400,CCR ;FLUSH CACHE
17847 107316 012701 107316 MOV #.,R1 ;START WITH CURRENT INSTRUCTION
17848 107322 005721 TST (R1). ;READ A WORD
17849 107324 022701 107422 1$: CMP #10$,R1 ;DONE?
17850 107330 001374 BNE 1$ ;IF NOT, CONTINUE
17851 107332 005737 020000 TST @#20000 ;TO GUARANTY MISS ON 0
17852 107336 013702 000004 MOV @#4,R2 ;STORE TIMEOUT VECTOR
17853 107342 012737 107376 000004 MOV #5$,@#4 ;POINT NEW TO THE TEST
17854 107350 012737 000340 000006 MOV #340,@#6 ;AT PRIORITY 7
17855 107356 042737 001000 177520 BIC #BIT09,BCSR ;DISABLE HALT ON BREAK
17856 107364 052737 000400 177520 BIS #BIT08,BCSR ;GO TO STANDALONE MODE
17857 107372 005737 000000 TST @#0 ;MISS, SHOULD TIMEOUT
17858 107376 042737 000400 177520 5$: BIC #BIT08,BCSR ;CLEAR STANDALONE BIT
17859 107404 052737 001000 177520 BIS #BIT09,BCSR ;ENABLE HALT ON BREAK
17860 107412 022716 107376 CMP #5$, (SP) ;TIMEOUT?
17861 107416 001401 BEQ 10$ ;IF YES, BRANCH
17862 107420 104044 ERROR -44
17863 107422 012706 001100 10$: MOV #1100,SP ;RESTORE STACK
17864 107426 010237 000004 MOV R2,@#4 ;AND TIMEOUT VECTOR

```

```

17865 ;SBTTL TEST - MOVING INVERSIONS TEST FOR DATA RAMS
17866 ;MOVING INVERSIONS TEST FOR DATA RAMS - THE TEST IS STARTED AFTER LOADING THE
17867 ;RAM STORE WITH 0'S. EACH ADDRESS IS READ AND VERIFIED TO BE ALL 0'S. THEN A
17868 ;1 IS SUBSTITUTED IN A BIT POSITION AND THE NEW WORD IS WRITTEN. NEXT THE
17869 ;ADDRESS IS READ TO VERIFY THE NEW CONTENTS. THIS IS REPEATED FOR EACH BIT OF
17870 ;THE WORD LEAVING THE ARRAY FILLED WITH 1'S. THE WHOLE PROCESS IS REPEATED
17871 ;PLUGGING IN 0'S TO THE 1'S AND REPEATED TWICE MORE ADDRESSING IN THE DOWNWARD
17872 ;DIRECTION. FINALLY EVERYTHING IS REPEATED FOR EACH BIT POSITION BEING THE
17873 ;LSB. TO SAVE TIME AND KNOWING THE LAYOUT OF THE RAM CHIPS INSTEAD OF DOING
17874 ;ONLY A SINGLE BIT AT A TIME EVERY FOURTH BIT WILL BE DONE CONCURRENTLY.
17875 ;THIS TEST RUNS IN STANDALONE MODE.
17876 ;
17877 ;BGNTEST
17878 ;SETUP AND ENABLE MMU
17879 ;SETUP CCR TO ABORT PARITY ERRORS
17880 ;CLEAR CACHE
17881 ;DO IN STANDALONE MODE FOR EACH HALF SEPARATELY
17882 ;LET FWDSEQ = #1
17883 ;LET ADDLSB = #1
17884 ;DO UNTIL ADDLSB EQ #20000
17885 ;.
17886 ;.   LET CURDAT = 0
17887 ;.   LET RITEDA = #1
17888 ;.   LET NEWDAT = #1
17889 ;.   IF FWDSEQ = #1 THEN
17890 ;.       LET FSTADD EQUAL FIRST ADDRESS OF 4K BYTE BUFFER
17891 ;.       LET LASTAD EQUAL LAST ADDRESS OF 4K BYTE BUFFER
17892 ;.   ELSE
17893 ;.       LET FSTADD EQUAL LAST ADDRESS OF 4K BYTE BUFFER
17894 ;.       LET LASTAD EQUAL FIRST ADDRESS OF 4K BYTE BUFFER
17895 ;.   ENDIF
17896 ;.   LET CURADD = FSTADD
17897 ;.   LET DCOUNT = #0
17898 ;.   SAVE CONTENTS OF VECTOR 114
17899 ;.   LET VECTOR 114 POINT TO DATA STORE PARITY ABORT ROUTINE
17900 ;.   DO UNTIL DCOUNT EQ #10
17901 ;.       LET RECDAT = @CURADD
17902 ;.       IF RECDAT NE CURDAT THEN
17903 ;.           LET R1 EQUAL CURRENT DATA
17904 ;.           GOTO DATA STORE PARITY ERROR ROUTINE
17905 ;.       ENDIF
17906 ;.       IF DCOUNT GT #3 THEN
17907 ;.           LET @CURADD = @CURADD CLEARBY RITEDA
17908 ;.       ELSE
17909 ;.           LET @CURADD = @CURADD SETBY RITEDA
17910 ;.       ENDIF
17911 ;.       LET RECDAT = @CURADD
17912 ;.       IF RECDAT NE NEWDAT THEN
17913 ;.           LET R1 EQUAL NEW DATA
17914 ;.           GOTO DATA STORE PARITY ERROR ROUTINE
17915 ;.       ENDIF
17916 ;.       IF CURADD EQ LASTAD THEN
17917 ;.           IF RITEDA = #010 THEN
17918 ;.               IF DCOUNT NE #7 THEN
17919 ;.                   LET CURDAT = #377
17920 ;.                   LET RITEDA = #1
17921 ;.                   LET NEWDAT = #376

```

17921									ENDIF
17922								ELSE	
17923									LET CURDAT = NEWDAT
17924									ROTATE RITEDA
17925									IF DCOUNT GT #3 THEN
17926									LET NEWDAT = NEWDAT CLEAREDBY RITEDA
17927								ELSE	
17928									LET NEWDAT = NEWDAT SETBY RITEDA
17929								ENDIF	
17930								ENDIF	
17931								INCREMENT DCOUNT	
17932								LET CURADD = FSTADD	
17933								ELSE	
17934									IF FWDSEQ = #1 THEN
17935									LET CURADD = CURADD + ADDLSB
17936									IF CARRY THEN
17937									LET CURADD = CURADD + #1
17938								ENDIF	
17939								ELSE	
17940									LET CURADD = CURADD - ADDLSB
17941									IF CARRY THEN
17942									LET CURADD = LASTAD - ADDLSB
17943									LET CURADD = CURADD - #1
17944								ENDIF	
17945								ENDIF	
17946								ENDIF	
17947								ENDDO	
17948								IF FWDSEQ EQ #1 THEN	
17949								LET FWDSEQ = #0	
17950								ELSE	
17951									ROTATE ADDLSB
17952									LET FWDSEQ = #1
17953								ENDIF	
17954								ENDDO	
17955								RESTORE VECTOR 114	
17956								ENDTST	
17957									
17958									
17959	107432	000004						TST26:	SCOPE
17960	107434	032777	000200	071476				BIT	#BIT07,@SWR ;RUN THIS TEST?
17961	107442	001002						BNE	100\$;IF SET, GO DO IT
17962	107444	000137	110526					JMP	ENDMOV ;OTHERWISE, GO TO NEXT TEST
17963	107450	042737	001000	177520	100\$:			BIC	#1000,BCSR ;DISABLE HALT ON BREAK
17964	107456	004737	132156					JSR	PC, INITMM ;SETUP MEMORY MANAGEMENT
17965	107462	012737	002000	172354				MOV	#2000,KIPAR6 ;START ON 32K BOUNDARY
17966	107470	005237	177572					INC	SRO ;TURN ON MMU
17967	107474	052737	000002	177746				BIS	#2,CCR ;SET DIAG. BIT
17968	107502	013737	000004	001160				MOV	#4,\$TMP0 ;SAVE 4
17969									
17970									;STORE TEST IN THE FIRST 2K AND THEN IN THE SECOND 2K
17971									
17972	107510	012703	140000					MOV	#140000,R3 ;START FOR THE TEST
17973	107514	012704	150000					MOV	#150000,R4 ;LOWER BOUNDARY TEST AREA
17974	107520	012705	157777					MOV	#157777,R5 ;HIGH BOUNDARY TEST AREA
17975	107524	000406						BR	\$
17976	107526	012703	150000		1\$:			MOV	#150000,R3 ;START OF THE TEST

COKDABO KDJ11-B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 336
 COKDAB.P11 05-APR-84 16:45 TEST MOVING INVERSIONS TEST FOR DATA RAMS

SEQ 0335

17977	107532	012704	140000		MOV	#140000,R4	;LOWER BOUNDARY TEST AREA	
17978	107536	012705	147777		MOV	#147777,R5	;HIGH BOUNDARY	
17979	107542	012702	107662	2\$:	MOV	#STMOVI,R2	;START WITH CURRENT	
17980	107546	010300			MOV	R3,R0	;MOVE TO UPPER 4K	
17981	107550	012220		3\$:	MOV	(R2)+,(R0)+	;WORD BY WORD	
17982	107552	022702	110526		CMP	#ENDMOV,R2	;ALL DONE	
17983	107556	001374			BNE	3\$		
17984	107560	010400			MOV	R4,R0	;CLEAR CACHE UNDER TEST	
17985	107562	012701	004000		MOV	#4000, R1		
17986	107566	005020		4\$:	CLR	(R0)+		
17987	107570	077102			SOB	R1, 4\$		
17988	107572	004713			JSR	PC,(R3)	;GO DO THE ROUTINE	
17989	107574	005702			TST	R2	;ANY ERRORS?	
17990	107576	001411			BEQ	5\$	;IF NOT, CONTINUE	
17991	107600	010037	110504		MOV	R0,RECDAT	;DATA RECEIVED	
17992	107604	010237	001122		MOV	R2,\$BDADR	;ADDRESS RECIEVED	
17993	107610	042737	140000	001122	BIC	#140000,\$BDADR	;STRIP OF PAR BITS	
17994	107616	104043			ERROR	+43		
17995	107620	000403			BR	6\$	;EXIT	
17996	107622	022703	150000	5\$:	CMP	#150000,R3	;DONE FOR BOTH HALVES?	
17997	107626	001337			BNE	1\$	;IF NOT, DO AGAIN	
17998	107630	052737	001000	177520	6\$:	BIS	#BIT09,\$BCSR	;ENABLE HALT ON BREAK
17999	107636	013737	001160	000004	MOV	\$TMP0,\$#4	;RESTORE TIMEOUT VECTOR	
18000	107644	012737	000400	177746	MOV	#400,CCR	;INIT CCR FOR EXIT	
18001	107652	005037	177572		CLR	SRO	;TURN OFF MMU	
18002	107656	000137	110526		JMP	ENDMOV	;GO TO NEXT TEST	
18003								
18004					.DSABL AMA			
18005	107662	052737	000400	177520	STMOVI: BIS	#BIT08,\$#BCSR	;STANDALONE MODE	
18006	107670	005002			CLR	R2	;ERROR INDICATOR	
18007	107672	012767	000001	000606	MOV	#1, FWDSEQ	;INIT UPWARD ADDRESSING INDICATOR	
18008	107700	012767	000001	000602	MOV	#1, ADDLSB	;INIT LSB AND DO LOOP UNTIL SHIFTED OUT	
18009	107706	042737	100000	172300	BIC	#100000,\$#KIPDR0	;NO BYPASS	
18010	107714	012737	172360	000004	MOV	#KDPAR0,\$#4	;ALLOCATE TIMEOUT VECTOR	
18011	107722	012737	000340	000006	MOV	#340, \$#6	;AT PRIORITY 7	
18012	107730	012737	000006	172360	MOV	#6, \$#KDPAR0	;PUT RETURN	
18013	107736	052737	100000	172300	BIS	#100000,\$#KIPDR0	;BYPASS	
18014	107744	005067	000546		CLR	CURDAT	;INIT CURRENT DATA	
18015	107750	012767	000221	000534	MOV	#21, RITEDA	;INIT DATA TO BE WRITTEN	
18016	107756	012767	000221	000530	MOV	#21, NEWDAT	;INIT EXPECTED DATA	
18017	107764	005767	000216		TST	FWDSEQ	;IF ADDRESSING UPWARD	
18018	107770	001405			BEQ	1\$	;THEN	
18019	107772	010467	000520		MOV	R4,FSTADD	;LET FIRST ADDRESS EQUAL LOWEST VALUE	
18020	107776	010567	000520		MOV	R5,LSTADD	;LET LAST ADDRESS EQUAL HIGHEST VALUE	
18021	110002	000404			BR	2\$	;ELSE	
18022	110004	010567	000510		1\$:	MOV	R5,FSTADD	
18023	110010	010467	000506		MOV	R4,LSTADD	;LET FIRST ADDRESS EQUAL HIGHEST VALUE	
18024	110014	016767	000500	000502	2\$:	MOV	FSTADD, CURADD	
18025	110022	005067	000452		CLR	DCOUNT	;LET LAST ADDRESS EQUAL LOWEST VALUE	
18026	110026	022767	000010	000444	BGNTLP: CMP	#10, DCOUNT	;LET CURRENT ADDRESS EQUAL FIRST ADDRESS	
18027	110034	001567			BEQ	ENDTLP	;INIT LOOP COUNTER	
18028							;DO LOOP 8 TIMES (4 TIMES TO WRITE 1'S	
18029							;AND 4 TIMES TO WRITE BACK 0'S)	
18030					; DON'T REWRITE TIMEOUT VECTOR			
18031	110036	016700	000460		MOV	CURADD, R0	;STORE CURRENT ADDRESS	
18032	110042	042700	170000		BIC	#170000,R0	;LEAVE ONLY LOW 4K BITS	

M10

COKDABO KDJ11-B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 337
 COKDAB.P11 05-APR-84 16:45 TEST - MOVING INVERSIONS TEST FOR DATA RAMS

SEQ 0336

18033	110046	022700	000004		CMP	#4,	R0		;TIMOUT VECTOR?
18034	110052	001531			BEQ	8\$			;IF SO, DON'T REWRITE IT
18035	110054	022700	000005		CMP	#5,	R0		
18036	110060	001526			BEQ	8\$			
18037	110062	022700	000006		CMP	#6,	R0		
18038	110066	001523			BEQ	8\$			
18039	110070	022700	000007		CMP	#7,	R0		
18040	110074	001520			BEQ	8\$			
18041									
18042									
18043									
18044	110076	016701	000414		MOV	CURDAT,	R1		;MOVE GOOD DATA TO R1
18045	110102	117767	000416	000374	MOVB	@CURADD,	RECDAT		;FIRST READ LOCATION
18046	110110	126767	000370	000400	CMPB	RECDAT,	CURDAT		;IF RECIEVED DATA NOT EQUAL TO EXPECTED
18047	110116	001402			BEQ	1\$			;DATA THEN
18048	110120	000167	000334		JMP	EXBAD			;EXIT TEST
18049	110124	016701	000364		MOV	NEWDAT,	R1		;MOVE GOOD DATA TO R1
18050	110130	022767	000003	000342	CMP	#3,	DCOUNT		;IF LOOP COUNT GREATER THAN 3
18051	110136	002004			BGE	2\$			;THEN
18052	110140	146777	000346	000356	BICB	RITEDA,	@CURADD		;CLEAR TEST DATA BY WRITE DATA
18053	110146	000403			BR	3\$			;ELSE
18054	110150	156777	000336	000346	BISB	RITEDA,	@CURADD		;SET TEST DATA BY WRITE DATA
18055	110156	117767	000342	000320	MOVB	@CURADD,	RECDAT		;DO READ AFTER WRITE
18056	110164	126767	000314	000322	CMPB	RECDAT,	NEWDAT		;IF RECIEVED DATA NOT EQUAL TO EXPECTED
18057	110172	001402			BEQ	4\$			;DATA THEN
18058	110174	000167	000260		JMP	EXBAD			;EXIT TEST
18059	110200	026767	000320	000314	CMP	CURADD,	LSTADD		;IF CURRENT ADDRESS EQUALS LAST ADDRESS
18060	110206	001053			BNE	8\$			;THEN
18061	110210	022767	000210	000274	CMP	#210,	RITEDA		;AND IF WRITE PATTERN EQUALS LAST
18062	110216	001016			BNE	5\$			;PATTERN THEN
18063	110220	022767	000007	000252	CMP	#7,	DCOUNT		;AND TEST IS NOT ON LAST LOOP
18064	110226	001435			BEQ	7\$			;THEN
18065	110230	012767	000377	000260	MOV	#377,	CURDAT		;SET UP TO WRITE 0'S
18066	110236	012767	000021	000246	MOV	#21,	RITEDA		
18067	110244	012767	000356	000242	MOV	#356,	NEWDAT		
18068	110252	000423			BR	7\$			;ELSE
18069	110254	016767	000234	000234	MOV	NEWDAT,	CURDAT		;SET UP FOR NEXT DATA PATTERN
18070	110262	000241			CLC				
18071	110264	106167	000222		ROLB	RITEDA			
18072	110270	005567	000216		ADC	RITEDA			;GET A PATTERN
18073	110274	022767	000003	000176	CMP	#3,	DCOUNT		;AND IF DOWNWARD ADDRESSING
18074	110302	002004			BGE	6\$			;THEN
18075	110304	046767	000202	000202	BIC	RITEDA,	NEWDAT		;LET NEWDAT BE CLEARED BY WRITE DATA
18076	110312	000403			BR	7\$			;ELSE
18077	110314	056767	000172	000172	BIS	RITEDA,	NEWDAT		;LET NEWDAT BE SET BY WRITE DATA
18078	110322	005267	000152		INC	DCOUNT			;UPDATE LOOP COUNTER
18079	110326	016767	000166	000170	MOV	FSTADD,	CURADD		;REINIT FIRST ADDRESS FOR THIS PASS
18080	110334	000426			BR	10\$			;ELSE
18081	110336	005767	000144		IST	FWDSEQ			;IF ADDRESSING UPWARD
18082	110342	001412			BEQ	9\$			;THEN
18083	110344	066767	000140	000152	ADD	ADDLSB,	CURADD		;CALCULATE NEXT HIGHER ADDRESS
18084	110352	020567	000146		CMP	RS,CURADD			;IF CURRENT ADDRESS HAS BEEN INCREASED
18085	110356	002015			BGE	10\$			;ABOVE HIGHEST ADDRESS THEN
18086	110360	162767	001777	000136	SUB	#7777,	CURADD		;ROLL ADDRESS BACK
18087	110366	000411			BR	10\$			;ELSE
18088	110370	166767	000114	000126	SUB	ADDLSB,	CURADD		;CALCULATE NEXT LOWER ADDRESS

N10

CORDBO RDJ11-B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 338
 CORDAB.P11 05-APR-84 16:45 TEST - MOVING INVERSIONS TEST FOR DATA RAMS

SEQ 0337

18089	110376	020467	000122		CMP	R4,CURADD		;IF CURRENT ADDRESS HAS BEEN DECREASED
18090	110402	003403			BLE	10\$		;BELOW LOWEST ADDRESS THEN
18091	110404	062767	007777	000112	ADD	#7777, CURADD		;ROLL ADDRESS BACK
18092	110412	000605			BR	BGMTLP		;ENDDO
18093	110414	005767	000066		ENDTLP;	TST	FWDSEQ	;IF ADDRESSING UPWARD FINISHED
18094	110420	001404			BEQ	1\$		;THEN
18095	110422	005067	000060		CLR	FWDSEQ		;DO ADDRESSING DOWNWARD
18096	110426	000167	177312		JMP	TSTLUP		
18097	110432	012767	000001	000046	1\$;	MOV	#1, FWDSEQ	;SET ADDRESSING UPWARD INDICATOR
18098	110440	006167	000044		ROL	ADDLSB		;UPDATE LSB TO NEXT POSITION
18099	110444	022767	020000	000036	ENDLUP;	CMP	#20000,ADDLSB	;ALL DONE?
18100	110452	001406			BEQ	EXITST		;ENDDO
18101	110454	000167	177264		JMP	TSTLUP		
18102	110460	016700	000020		EXBAD;	MOV	RECDAT,R0	;STORE RECEIVED DATA
18103	110464	016702	000034		MOV	CURADD,R2		;STORE ADDRESS
18104	110470	042737	000400	177520	EXITST;	RIC	#BIT08,#BCSR	;OUT OF STANDALONE
18105	110476	000207			RTS	PC		
18106					.ENABL	AMA		
18107								
18108	110500	000000			DCOUNT;	.WORD	0	
18109	110502	000000			EXPDAT;	.WORD	0	;STORES EXPECTED (GOOD) DATA FOR COMPARISONS
18110	110504	000000			RECDAT;	.WORD	0	;STORES RECEIVED DATA TO BE VERIFIED
18111	110506	000000			FWDSEQ;	.WORD	0	;USED TO INDICATE DIRECTION OF ADDRESSING
18112	110510	000000			ADDLSB;	.WORD	0	;STORES LEAST SIGNIFICANT BIT FOR RAM TESTS
18113	110512	000000			RITEDA;	.WORD	0	;STORES WRITE DATA FOR RAM TESTS
18114	110514	000000			NEWDAT;	.WORD	0	;DATA STORE FOR RAM TESTS
18115	110516	000000			CURDAT;	.WORD	0	;DATA STORE FOR RAM TESTS
18116	110520	000000			FSTADD;	.WORD	0	;STORES FIRST ADDRESS IN ADDRESSING SEQUENCE
18117	110522	000000			LSTADD;	.WORD	0	;STORES LAST ADDRESS IN ADDRESSING SEQUENCE
18118	110524	000000			CURADD;	.WORD	0	;STORES CURRENT ADDRESS FOR RAM TESTS
18119								
18120	110526				ENDMOV;			

05 APR-84 16:45 PAGE 339
TEST MOVING INVERSIONS TEST FOR TAG STORE

181131
181132
181133
181134
181135
181136
181137
181138
181139
181130
181131
181132
181133
181134
181135
181136
181137
181138
181139
181140
181141
181142
181143
181144
181145
181146
181147
181148
181149
181150
181151
181152
181153
181154
181155
181156
181157
181158
181159
181160
181161
181162
181163
181164
181165
181166
181167
181168
181169
181170
181171
181172
181173
181174
181175
181176

```

;SBITL TEST - MOVING INVERSIONS TEST FOR TAG STORE
;MOVING INVERSIONS TEST - THE TEST IS STARTED AFTER LOADING THE RAM STORE
;WITH 0'S. EACH ADDRESS IS READ AND VERIFIED TO BE ALL 0'S, THEN A 1 IS
;SUBSTITUTED IN A BIT POSITION AND THE NEW WORD IS WRITTEN. NEXT THE ADDRESS
;IS READ TO VERIFY THE NEW CONTENTS. THIS IS REPEATED FOR EACH BIT OF THE
;WORD LEAVING THE ARRAY FILLED WITH 1'S. THE WHOLE PROCESS IS REPEATED PLUGGING
;IN 0'S TO THE 1'S AND REPEATED TWICE MORE ADDRESS IN THE DOWARD DIRECTION.
;FINALLY EVERYTHING IS REPEATED FOR EACH BIT POSITION BEING THE LSB. TO SAVE
;TIME AND KNOWING THE LAYOUT OF THE RAM CHIPS INSTEAD OF DOING ONLY A SINGLE
;BIT AT A TIME EVERY THIRD BIT WILL BE DONE CONCURRENTLY. ALSO NOTE THAT
;SINCE THE TAG STORE CANNOT BE DIRECTLY ACCESSED THE ENTIRE PATTERN MUST BE
;DONE BY DOING MEMORY CYCLES TO THE CORRECT BUS ADDRESSES. TO DO THIS THE
;TEST WILL BE DONE WITH THE DIAGNOSTIC BIT IN THE CCR (BIT 1) SET TO A 1.
;THIS TEST RUNS IN STANDALONE MODE.
;
;BGNTEST
;SETUP AND ENABLE MMU
;SETUP CCR TO ABORT PARITY ERRORS
;DO IN STANDALONE MODE FOR EACH HALF SEPARATELY
;LET FWDSEQ = 01
;LET ADDLSB = 01
;DO UNTIL ADDLSB EQ 020000
;.   LET NEWDAT = 22200
;.   LET CURDAT = 0
;.   IF FWDSEQ = 01 THEN
;.       LET FSTADD EQUAL FIRST ADDRESS OF 4K BYTE BUFFER
;.       LET LASTAD EQUAL LAST ADDRESS OF 4K BYTE BUFFER
;.   ELSE
;.       LET FSTADD EQUAL LAST ADDRESS OF 4K BYTE BUFFER
;.       LET LASTAD EQUAL FIRST ADDRESS OF 4K BYTE BUFFER
;.   ENDIF
;.   LET DCOUNT = 00
;.   DO UNTIL DCOUNT EQ 010
;.       READ CURADD USING CURDAT AS PAR
;.       IF MISS THEN
;.           ERROR
;.       ENDIF
;.       WRITE CURADD USING NEWDAT AS PAR
;.       ENDIF
;.       IF HIT THEN
;.           ERROR
;.       ENDIF
;.       READ CURADD USING NEWDAT AS PAR
;.       IF MISS THEN
;.           ERROR
;.       ENDIF
;.       IF CURADD EQ LASTAD THEN
;.           LET CURDAT = NEWDAT
;.           IF DCOUNT = 01 THEN
;.               LET NEWDAT = NEWDAT SET BY RITEDA ROTATED LEFT
;.           ENDIF
;.           IF DCOUNT = 01 THEN
;.               LET NEWDAT = NEWDAT CLR BY RITEDA ROTATED LEFT
;.           ELSE
;.               LET NEWDAT = 0155400 (NO ALL 1'S)
;.           ENDIF
;.       ENDIF
;.   ENDUNTIL
;.   LET ADDLSB = ADDLSB + 01
;.   LET FWDSEQ = FWDSEQ + 01
;.   IF FWDSEQ EQ 02 THEN
;.       LET FWDSEQ = 01
;.       LET ADDLSB = 01
;.   ENDIF
;. ENDUNTIL

```

						. . . LET RITEOA = 02200
						. . . ENDIF
						. . . INCREMENT DCOUNT
						. . . ELSE
						. IF FWDSEQ = 01 THEN
						. . . LET CURADD = CURADD + ADDLSB
						. . . IF CURADD GE HIGH ADDRESS THEN
						. . . LET CURADD = CURADD + 07776
						. . . ENDIF
						. . . ELSE
						. . . LET CURADD = CURADD - ADDLSB
						. . . IF CURADD LE LOW ADDRESS THEN
						. . . LET CURADD = CURADD - 07776
						. . . ENDIF
						. . . ENDIF
						. . . ENDDO
						. IF FWDSEQ EQ 01 THEN
						. . . LET FWDSEQ = 00
						. . . ELSE
						. . . ROTATE ADDLSB
						. . . LET FWDSEQ = 01
						. . . ENDIF
						. . . ENDDO
						. RESTORE PARITY ABORT VECTOR
						. ENDIST
18204						*****
18205	110526	000004			TST27:	SCOPE
18206	110530	032777	000100	070402		BIT 0BIT06,BSWR ;RUN THIS TEST?
18207	110536	001002			BNE	1003 ;IF SET, GO DO IT
18208	110540	000137	111630		JMP	FNDTAG ;OTHERWISE, GO TO NEXT TEST
18209	110544	042737	001000	177520	1003:	BIC #1000,BCSR ;DISABLE HALT ON BREAK
18210	110552	004737	132156		JSR	PC, INITMM ;SETUP MEMORY MANAGEMENT
18211	110556	005237	177572		INC	SRO ;TURN ON MMU
18212	110562	012737	000020	172516	MOV	#BIT04,MMR3 ;ENABLE 32 BIT MAPPING
18213	110570	052737	000002	177746	RIS	#2, CCR ;SET DIAG. BIT
18214	110576	013737	000004	001160	MOV	#04, \$TMPO ;STORE TIMEOUT VECTOR
18215					.	
18216					.	
18217					.	
18218	110604	012703	140000		.	
18219	110610	012704	130000		MOV	#140000,R3 ;START FOR THE TEST
18220	110614	012705	137776		MOV	#130000,R4 ;LOWER BOUNDARY TEST AREA
18221	110620	000406			MOV	#137776,R5 ;HIGH BOUNDARY TEST AREA
18222	110622	012703	150000		BR	\$;
18223	110626	012704	120000		1\$: MOV	#150000,R3 ;START OF THE TEST
18224	110632	012705	127776		MOV	#120000,R4 ;LOWER BOUNDARY TEST AREA
18225	110636	012737	002000	172354	MOV	#127776,R5 ;HIGH BOUNDARY
18226	110644	012702	110470		2\$: MOV	#2000,KIPAR6 ;ABOVE 3K
18227	110650	010300			MOV	#EXITST,RP ;START WITH CURRENT
18228	110652	012220			MOV	R3,RO ;MOVE TO UPPER 4K
18229	110654	022702	111630		3\$: MOV	(RP), (RO) ;WORD BY WORD
18230	110660	001374			CMP	#FNDTAG,RP ;ALL DONE?
18231	110662	012700	110764		BNE	\$;
18232	110666	162700	110470		MOV	#TIMOVLT,RO ;ADDRESS OF ROUTINE
					SUB	#EXITST,RO ;PROPER OFFSET

011

CORPANO KDJ11 B CLUSTER MAC111 30(1046) 05 APR 84 16:45 PAGE 341
COKDAB,P11 05-APR 84 16:45 TEST MOVING INVERSIONS TEST FOR TAG STORE

SEQ 0340

18233	110672	050003			BIS	R0,R3	
18234	110674	004713			JSR	PC,(R3)	
18235	110676	005700			TST	R0	
18236	110700	001407			BEQ	4:	
18237	110702	010037	001122		MOV	R0,\$BDADR	
18238	110706	042737	160000	001122	BIC	\$160000,\$BDADR	
18239	110714	104050			ERROR	50	
18240	110716	000403			BR	5:	
18241	110720	022703	150000		4\$: CMP	\$150000,R3	
18242	110724	103736			BLO	1:	
18243	110726	013737	001160	000004	5\$: MOV	\$TMP0, @4	
18244	110734	012737	000400	177746	MOV	@400, CCR	
18245	110742	005037	177572		CLR	540	
18246	110746	005037	172516		CLR	MMR3	
18247	110752	052737	001000	177520	BIS	\$1000,BCSR	
18248	110760	000137	111630		JMP	ENDTAG	
18249							
18250					.DSABL AMA		
18251	110764	052737	000400	177520	STMOVT: BIS	\$BIT08,\$BCSR	
18252	110772	042737	100000	172312	BIC	\$BIT15, \$KIPDR5	
18253	111000	042737	100000	172300	BIC	\$100000,\$KIPDR0	
18254	111006	012737	172360	000004	MOV	\$KDPAR0,\$4	
18255	111014	012737	000340	000006	MOV	\$340, @6	
18256	111022	012737	000006	172360	MOV	@6, \$KDPAR0	
18257	111030	052737	100000	172300	BIS	\$100000,\$KIPDR0	
18258	111036	005037	172352		CLR	\$KIPARS	
18259	111042	010400			MOV	R4,R0	
18260	111044	012701	004000		MOV	\$4000, R1	
18261	111050	005020			4\$: CLR	(R0)+	
18262	111052	077102			SOB	R1, 4:	
18263	111054	005000			CLR	R0	
18264	111056	012767	000001	177422	MOV	@1, FWDSEQ	
18265	111064	012767	000002	177416	MOV	@2, ADDLSB	
18266	111072	005067	177420		TSL00P: CLR	CURDAT	
18267	111076	012767	022200	177410	MOV	\$22200,NEWDA1	
18268	111104	012767	022200	177400	MOV	\$22200,RITEDA	
18269	111112	005767	177370		TST	FWDSEQ	
18270	111116	001405			BEQ	1:	
18271	111120	010467	177374		MOV	R4, FSTADD	
18272	111124	010567	177372		MOV	R5, LSTADD	
18273	111130	000404			BR	2:	
18274	111132	010567	177362		1\$: MOV	R5, FSTADD	
18275	111136	010467	177360		MOV	R4, LSTADD	
18276	111142	005067	177332		2\$: CLR	DCOUNT	
18277	111146	016767	177346	177350	MOV	FSTADD, CURADD	
18278							
18279							
18280							
18281	111154	016700	177344		3\$: MOV	CURADD, R0	
18282	111160	042700	170000		BIC	\$170000,R0	
18283	111164	022700	000004		CMP	@4, R0	
18284	111170	001526			BEQ	16:	
18285	111172	022700	000006		CMP	@6, R0	
18286	111176	001523			BEQ	16:	
18287	111200	016737	177312	172352	MOV	CURDAT, \$KIPARS	
18288	111206	005777	177312		TST	\$CURADD	

F 11

COKDABO KDJ11-B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 342
COKDAR.P11 05-APR-84 16:45 TEST - MOVING INVERSIONS TEST FOR TAG STORE

SEQ 0341

18289	111212	013767	177752	177264		MOV	00HITMIS,RECDAT	
18290	111220	032767	000004	177256	5\$:	BIT	0BIT02, RECDAT	;IF ACCESS WAS A MISS
18291	111226	001002				BNE	6\$	;THEN
18292	111230	000167	000346			JMP	EXBAD2	;ERROR, EXIT
18293	111234	016737	177254	172352	6\$:	MOV	NEWDAT, 00KIPAR5	;GET NEW PATTERN
18294	111242	005077	177256			CLR	0CURADD	;REGISTER AND WRITE LOCATION
18295	111246	013767	177752	177230		MOV	00HITMIS,RECDAT	
18296	111254	032767	000004	177222	8\$:	BIT	0BIT02, RECDAT	;IF ACCESS WAS A HIT
18297	111262	001406				BEQ	9\$	;THEN
18298	111264	032767	000010	177212	10\$:	BIT	0BIT03, RECDAT	;MISS?
18299	111272	001402				BEQ	9\$	;IF SO, CONTINUE
18300	111274	000167	000302			JMP	EXBAD2	;ERROR
18301	111300	005777	177220		9\$:	TST	0CURADD	;REGISTER AND READ LOCATION
18302	111304	013767	177752	177172		MOV	00HITMIS,RECDAT	
18303	111312	032767	000004	177164	11\$:	BIT	0BIT02, RECDAT	;IF ACCESS WAS A MISS
18304	111320	001002				BNE	12\$	;THEN
18305	111322	000167	000254			JMP	EXBAD2	;ERROR, EXIT
18306	111326	026767	177170	177170	12\$:	CMP	LSTADD, CURADD	;IF CURRENT ADDRESS IS LAST ADDRESS
18307	111334	001044				BNE	16\$	;THEN
18308	111336	016767	177152	177152		MOV	NEWDAT,CURDAT	;NEWKIPAR5
18309	111344	022767	000001	177126		CMP	01, DCOUNT	;IF LOOP COUNTER LESS THAN 1
18310	111352	003406				BLE	13\$	;THEN
18311	111354	006367	177132			ASL	RITEDA	;UPDATE OF NEW DATA....
18312	111360	056767	177126	177126		BIS	RITEDA, NEWDAT	;BY SETTING BITS
18313	111366	000421				BR	15\$	;ENDIF
18314	111370	022767	000001	177102	13\$:	CMP	01, DCOUNT	;CHANGE PATTERN?
18315	111376	001007				BNE	14\$	;IF SO, BRANCH
18316	111400	012767	022200	177104		MOV	022200, RITEDA	;START WITH THE SAME
18317	111406	012767	155400	177100		MOV	0155400,NEWDAT	;DON'T DO ALL 1'S
18318	111414	000406				BR	15\$	
18319	111416	006367	177070		14\$:	ASL	RITEDA	;UPDATE OF NEW DATA....
18320	111422	046767	177064	177064		BIC	RITEDA, NEWDAT	;BY CLEARING BITS
18321	111430	000400				BR	15\$	;ELSE
18322	111432	005267	177042		15\$:	INC	DCOUNT	;INCREMENT THE LOOP COUNTER
18323	111436	016767	177056	177060		MOV	FSTADD, CURADD	;FINISH FIRST CURRENT ADDRESS
18324	111444	000426				BR	18\$	;ELSE
18325	111446	005767	177034		16\$:	TST	FWDSEQ	;IF ADDRESSING UPWARD
18326	111452	001412				BEQ	17\$	;THEN
18327	111454	066767	177030	177042		ADD	ADDLSB, CURADD	;ADD THE VALUE OF THE CURRENT LSB
18328	111462	020567	177036			CMP	R5, CURADD	;IF CURRENT ADDRESS GREATER THAN HIGHEST
18329	111466	002015				BGE	18\$	;VIRTUAL ADDRESS THEN
18330	111470	162767	007776	177026		SUB	07776, CURADD	;ROLL IT BACK
18331	111476	000411				BR	18\$	;ELSE
18332	111500	166767	177004	177016	17\$:	SUB	ADDLSB, CURADD	;IF ADDRESSING DOWNWARD THEN SUBTRACT LSB
18333	111506	020467	177012			CMP	R4, CURADD	;IF CURRENT ADDRESS LESS THEN LOWEST
18334	111512	003403				BLE	18\$	;VIRTUAL ADDRESS THEN
18335	111514	062767	007776	177002		ADD	07776, CURADD	;ROLL IT BACK
18336	111522	022767	000005	176750	18\$:	CMP	05, DCOUNT	;IF LOOP COUNTER LESS THAN 1
18337	111530	003402				BLE	181\$	;THEN LOOP BACK FOR NEXT BIT POSITION
18338	111532	000167	177416			JMP	3\$	
18339	111536	005767	176744		181\$:	TST	FWDSEQ	;IF ADDRESSING UPWARD
18340	111542	001404				BEQ	19\$	;THEN
18341	111544	005067	176736			CLR	FWDSEQ	;START ADDRESSING DOWNWARD
18342	111550	000167	177316			JMP	TSLOOP	;ELSE
18343	111554	012767	000001	176724	19\$:	MOV	01, FWDSEQ	;START ADDRESSING UPWARD
18344	111562	006167	176722			ROL	ADDLSB	;ROTATE LSB TO NEXT POSITION

1.1.

COKDABO KDJ11-R CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 343
COKDAB,P11 05-APR-84 16:45 TEST - MOVING INVERSIONS TEST FOR TAG STORE

SEQ 0342

18345	111566	022767	020000	176714	CMP	#20000, ADDLSB	; ALL DONE?
18346	111574	001406			BEQ	TSEND	; EXIT
18347	111576	000167	177270		JMP	TSLOOP	; ENDDO
18348	111602	013701	172352	FXBAD2:	MOV	@KIPAR5, R1	; STORE PAR
18349	111606	016700	176712		MOV	CURADD, R0	; AND ADDRESS
18350	111612	012737	001200	172352	TSEND:	MOV	#1200, @KIPAR5
18351	111620	042737	000400	177520		BIC	#BIT08, @BCSR
18352	111626	000207			RTS	PC	; OUT OF STANDALONE MODE
18353	111630				ENDTAG:		; RETURN
18354					.ENABL	AMA	
18355							

611

COKDABO KDJ11-B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 344
COKDAB.P11 05-APR-84 16:45 TEST - PCR READ/WRITE BITS

SEQ 0343

18356	.SBTTL TEST - PCR READ/WRITE BITS		
18357	;PCR AND BCSR READ/WRITE BITS		
18358	;THE FIRST TEST WILL CHECK THAT PCR REGISTER IS BOTH WORD AND		
18359	;BYTE ADDRESSABLE. BITS 14-09 AND 06-01 WILL BE WRITTEN AND READ		
18360	;AS ZEROS AND ONES. THE REST OF THE BITS HAVE TO BE ALL 0'S.		
18361	;ROUTINE TEST		
18362	.. SAVE PCR		
18363	.. LET PCR=0		
18364	.. DO FOR PATTERN=001111,110011,101010,010101		
18365	.. WRITE PCR<14-09>=PATTERN		
18366	.. WRITE PCR<06-01>=PATTERN		
18367	.. IF PCR<14-09> NE PATTERN OR PCR<06-01> NE PREVIOUS		
18368	.. PATTERN		
18369	.. THEN ERROR		
18370	.. ENDIF		
18371	.. WRITE PCR<06-01>=PATTERN		
18372	.. IF PCR<14-09> NE PATTERN OR PCR<06-01> NE PATTERN		
18373	.. THEN ERROR		
18374	.. ENDIF		
18375	.. ENDDO		
18376	.. WRITE PCR=01010101010101		
18377	.. IF PCR NE 0101010001010100 THEN		
18378	.. ERROR		
18379	.. ENDIF		
18380	;ENDROUTINE		
18381			
18382	;*****		
18383	111630	000004	TST30: SCOPE
18384	111632	005037	CLR PCR
18385	111636	012702	MOV #SIXBIT,1,R2
18386	111642	012704	MOV #SIXBIT,R4
18387	111646	012703	MOV #4,R3
18388			; INITIALIZE PCR TO 0
18389			;R2->TABLE OF PATTERNS FOR 6 R/W BITS IN
18390			;R3 POINTER TO PREVIOUS PATTERN
18391			;DO 4 TIMES
18392			; WRITE TO HIGH BYTE FIRST
18393	111652	111237	1\$: MOVB (R2),PCR+1
18394	111656	121237	CMPB (R2),PCR+1
18395	111662	001003	BNE 2\$
18396	111664	121437	CMPB (R4),PCR
18397	111670	001405	BEQ 3\$
18398	111672	111237	2\$: MOVB (R2),%GDDAT+1
18399	111676	111437	MOVB (R4),%GDDAT
18400	111702	104051	ERROR +51
18401	111704	105724	3\$: TSTB (R4)+
18402			; INCREMENT POINTER FOR OLD
18403			; WRITE TO LOW BYTE
18404	111706	111237	4\$: MOVB (R2),PCR
18405	111712	121237	CMPB (R2),PCR
18406	111716	001003	BNE 4\$
18407	111720	121237	CMPB (R2),PCR
18408	111724	001405	BEQ 5\$
18409	111726	111237	5\$: MOVB (R2),%GDDAT
18410	111732	111237	MOVB (R2),%GDDAT
18411	111736	104051	ERROR +51
	111740	105722	5\$: TSTB (R2)+
			; INCREMENT POINTER FOR NEW PATTERN

HLL

COKDABO KDJ11 B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 345
COKDAB.P11 05-APR-84 16:45 TEST - PCR READ/WRITE BITS

SEQ 0344

18412	111742	077335				SOB	R3,1\$				
18413											
18414						:					
18415						:	NOW TRY WORD ADDRESSING				
18416	111744	012737	052525	177522		MOV	#52525,PCR				
18417	111752	022737	052124	177522		CMP	#52124,PCR				
18418	111760	001404				BEQ	6\$				
18419	111762	112737	052124	001124		MOVB	#52124,\$GDDAT				
18420	111770	104051				ERROR	*51				
18421	111772					6\$:					
18422	111772	000403				BR	TST31				
18423											
18424	111774	000	036	146	SIXBIT:	.BYTE	0,\$6,146,124,52				
18425	111777	124	052								
18426											
18427		112002									
18428							.EVEN				

;DO FOR ALL 4 PATTERNS

;WRITE A PATTERN

;ALL BUT BITS <8,7,0> OK?

;IF SO, BRANCH

;EXPECTED PATTERN

;ERROR PCR READ/WRITE BITS

;;GO TO NEXT TEST

;001111,110011,101010,010101

;BINARY DIVIDE FOR 6 BITS

```

18429 .SBTTL TEST - BCSR READ/WRITE BITS
18430 ;THE SECOND TEST WILL CHECK THAT BCSR<7-5;2-0> CAN BE WRITTEN AND
18431 ;READ AS ZEROES AND ONES. BCSR<14,03> SHOULD BE 0'S. BCSR<04>
18432 ;SHOULD BE CLEARED BY RESET INSTRUCTION.
18433 ;ROUTINE TEST
18434 ;. FOR PATTERN=011,010,101 DO
18435 ;. . WRITE BCSR<7-5>=PATTERN
18436 ;. . IF BCSR<7-5> NE PATTERN THEN
18437 ;. . . ERROR
18438 ;. . . ENDF
18439 ;. . WRITE BCSR<2-0>=PATTERN
18440 ;. . IF BCSR<2-0> NE PATTERN THEN
18441 ;. . . ERROR
18442 ;. . . ENDF
18443 ;. ENDDO
18444 ;. IF BCSR<14,03> NE <0,0> THEN
18445 ;. . ERROR
18446 ;. ENDF
18447 ;. LET BCSR<04>=1
18448 ;. IF BCSR<04> NE #1 THEN
18449 ;. . ERROR
18450 ;. ENDF
18451 ;. EXECUTE "RESET"
18452 ;. IF BCSR<04> NE 0 THEN
18453 ;. . ERROR
18454 ;. ENDF
18455 ;. LET BCSR<04>=0 (THIS BIT IS WRITE ENABLE FOR EAROM)
18456 ;ENDROUTINE
18457
18458 ;*****
18459 112002 000004 TST31: SCOPE
18460 112004 013737 177520 002710 MOV BCSR,SAVBR ;SAVE BCSR
18461 112012 005037 177520 CLR BCSR ;CLEAR BCSR
18462
18463 ; WRITE TO BITS <7-5> AND <2-0>
18464 ;
18465 112016 012703 112252 MOV #THRBIT,R3 ;POINTER FOR PATTERN TABLE
18466 112022 005037 001124 CLR $GDDAT ;CLEAR A LOCATION
18467 112026 012702 000003 MOV #3,R2 ;DC FOR ALL PATTERNS
18468 112032 111337 177520 1$: MOVB (R3),BCSR ;WRITE TO BITS <7-5>
18469 112036 111337 001124 MOVB (R3),$GDDAT ;EXPECTED PATTERN
18470 112042 122337 177520 CMPB (R3)+,BCSR ;BITS WRITTEN OK?
18471 112046 001401 BEQ 2$ ;IF SO, BRANCH
18472 112050 104052 ERROR +52 ;ERROR IN BCSR READ/WRITE BITS
18473 112052 111337 177520 2$: MOVB (R3),BCSR ;WRITE TO BITS <2-0>
18474 112056 111337 001124 MOVB (R3),$GDDAT ;EXPECTED PATTERN FOR ERRORS
18475 112062 122337 177520 CMPB (R3)+,BCSR ;BITS WRITTEN OK?
18476 112066 001401 BEQ 3$ ;IF SO, BRANCH
18477 112070 104052 ERROR +52 ;ERROR IN BCSR READ/WRITE BITS
18478 112072 077221 3$: SOB R2,1$ ;CONTINUE TILL ALL PATTERNS DONE
18479
18480 ; CHECK UNUSED BITS <3>
18481 ;
18482 112074 012737 000010 177520 MOV #10,BCSR ;WRITE TO BIT <3>
18483 112102 032737 000010 177520 BIT #BIT03,BCSR ;ALL ZEROES?
18484 112110 001403 BEQ 4$ ;IF YES, BRANCH

```


K11

COKDABO RDJ11-B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 348
 COKDAB.P11 05-APR-84 16:45 TEST - 16 BIT ROM CHECKSUM TEST

SEQ 0347

```

18517 .SBITL TEST - 16 BIT ROM CHECKSUM TEST
18518 ;ROM'S CHECKSUMS
18519 ;
18520 ;16 BIT ROM TEST
18521 ;THE FIRST TEST WILL CLEAR BCSR<07>, LOAD PCR<14-09> WITH
18522 ;ROM ADDRESS BITS <14-09>, AND CHECK CHECKSUMS OF 16-BIT ROM
18523 ;BY ACCESSING IT THRU BUS ADDRESSES 173000-173776. THEN WITH
18524 ;BCSR<06;05> BOTH CLEAR, PCR<06-01> USED AS ADDRESS BITS 14-09,
18525 ;THE SAME THING WILL BE DONE BY ADDRESSING 16-BIT ROM THRU BUS ADDRESSES
18526 ;165000-165776. THE RESULTS SHOULD BE THE SAME AND SHOULD COMPARE
18527 ;WITH THAT STORED IN THE BOOT AND DIAGNOSTIC ROM.
18528 ;
18529 ;BCSR <07> DISABLE 17773000
18530 ; <06> DISABLE 17765000
18531 ; <05> ROM SOCKET 3 AT 17765000
18532 ;
18533 ;ROUTINE TEST
18534 ;. LET BCSR<7,6,5>=0,0,0
18535 ;. DO FOR R1 FROM #0 TO #31. BY #1 DO
18536 ;. . LET PCR<14-09>=R1
18537 ;. . DO FOR R2 FROM #0 TO #776 BY 2
18538 ;. . . CALCULATE CHECKSUM THRU 173000
18539 ;. . . ENDDO
18540 ;. . ENDDO
18541 ;. IF CHECKSUM NE #0 THEN
18542 ;. . ERROR
18543 ;. . ENDF
18544 ;. DO FOR R1 FROM #0 TO #31. BY #1
18545 ;. . LET PCR<06-01>=R1
18546 ;. . DO FOR R2 FROM #0 TO #776 BY 2
18547 ;. . . CALCULATE CHECKSUM THRU 165000
18548 ;. . . ENDDO
18549 ;. . ENDDO
18550 ;. IF CHECKSUM NE #0 THEN
18551 ;. . ERROR
18552 ;. . ENDF
18553 ;. ENDRoutine
18554 ;
18555 ;*****
18556 TST32: SCOPE
18557 112260 000004 MOV BCSR,SAVBR ;SAVE BCSR
18558 112262 013737 177520 002710 BIC #BIT07!BIT06!BIT05,BCSR ;READ 16 BIT ROM
18559 112270 042737 000340 177520 ;
18560 ; CALCULATE LOW BYTE CHECKSUM'S THRU 173000 AND 165000
18561 ;
18562 112276 005001 CLR R1 ;PAGE COUNT FOR ALL 8K
18563 112300 005037 177522 CLR PCR ;CLEAR PAGE CONTROL REGISTER
18564 112304 005037 002704 1$: CLR ACTCHS ;CLEAR CHECKSUM AT 173000
18565 112310 005037 001160 CLR $TMP0 ;AT 165000
18566 112314 005002 CLR R2 ;CLEAR COUNTER THRU A PAGE
18567 112316 016203 173000 2$: MOV 173000(R2),R3 ;GET LOW BYTE THRU 173000
18568 112322 016204 165000 MOV 165000(R2),R4 ;THRU 165000
18569 112326 060337 002704 ADD R3,ACTCHS ;CALCULATE CHECKSUM THRU 173000
18570 112332 060437 001160 ADD R4,$TMP0 ;CALCULATE CHECKSUM THRU 165000
18571 112336 005722 TST (R2); ;GET NEXT WORD
18572 112340 022702 000776 CMP #776,R2 ;WORD BEFORE LAST?

```


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

M11

COKDABO KDJ11 B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 350
 COKDAB.P11 05-APR-84 16:45 TEST - 8 BIT ROM CRC TEST

SEQ 0349

```

18624 ;SBTTL TEST - 8 BIT ROM CRC TEST
18625 ;THIS TEST WILL CLEAR BCSR<6>, SET BCSR<5>, LOAD PCR<5-1>
18626 ;WITH ADDRESS BITS 13-09, AND CHECK CRC PATTERNS OF 8-BIT ROM BY
18627 ;ACCESSING IT THRU ADDRESSES 165000-165776. THIS TEST VERIFIES THE
18628 ;RESPONSE ONLY OF THE BASE AREA.
18629 ;ROUTINE TEST
18630 ;; LET BCSR<5>=1
18631 ;; LET OLDCRC=0
18632 ;; LET PCR<05-01>=0
18633 ;; CALCULATE CHECKSUM FOR THE FIRST 320 LOCATIONS
18634 ;; IF RESULTING CHECKSUM NOT ZERO THEN
18635 ;; ERROR
18636 ;; ENDIF
18637 ;ENDROUTINE
18638
18639 ;*****
18640 TST33: SCOPE
18641 MOV BCSR,SAVBR ;SAVE BCSR
18642 BIC #BIT06,BCSR ;ENABLE INTERNAL RESPONSE
18643 BIS #BIT05,BCSR ;SELECT 8-BIT ROM
18644 CLR $TMP0 ;CLEAR SUM
18645
18646 ; CALCULATE LOW BYTE CHECKSUM'S THRU 165000
18647 ;
18648 1$: CLR R1 ;PAGE COUNT FOR ALL 8K
18649 CLR PCR ;CLEAR PAGE CONTROL REGISTER
18650 2$: CLR ACTCHS ;CLEAR CHECKSUM AT 165000
18651 CLR R2 ;CLEAR COUNTER THRU A PAGE
18652 3$: MOVB 165000(R2),R4 ;GET A BYTE THRU 165000
18653 ADD R4,$TMP0 ;CALCULATE CHECKSUM THRU 165000
18654 TST (R2)+ ;GET NEXT WORD
18655 CMP #316,R2 ;WORD BEFORE LAST?
18656 BNE 4$ ;IF NOT, BRANCH
18657 CMPB #252,R4 ;314 SHOULD HAVE 252
18658 BEQ 3$ ;IF YES, BRANCH
18659 ERROR +55 ;PAGE NUMBER STORED WRONG
18660 4$: CMP #322,R2 ;LAST WORD IN A PAGE?
18661 BGT 3$ ;IF NOT, BRANCH
18662 MOVB 165010,$TMP1 ;STORE SIZE,<3>-1 BK
18663 TSTB $TMP0 ;CHECKSUM 0 AT 165000?
18664 BEQ 5$ ;IF YES, BRANCH
18665 ERROR +55 ;IN CHECKSUM AT 165000
18666 5$: CLR PCR
18667 MOV SAVBR,BCSR ;RESTORE BCSR
18668
18669

```

N11

COKDABO KDJ11-B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 351
 COKDAB.P11 05-APR-84 16:45 TEST - LKS BIT 7

SEQ 0350

```

18670 .SBTTL TEST - LKS BIT 7
18671 ;THESE TESTS WILL CHECK FUNCTIONALITY OF LKS<7-6>
18672 ;
18673 ;ACCESS LKS, CHECK THAT READY LKS<07> CAN BE 0 AND 1.
18674 ;
18675 ;LKS <07> LINE CLOCK MONITOR
18676 ; <06> LINE CLOCK INTERRUPT ENABLE
18677 ;ROUTINE TEST
18678 ;IF UFD AND LKS IS DISABLED THEN
18679 ;. EXIT TEST
18680 ;ENDIF
18681 ;. READ LKS TO SEE IF IT TIMEOUTS
18682 ;.(CHECK LKS<07>)
18683 ;. LET R1=#77777(WORST CASE COUNTER FOR SLOW LKS)
18684 ;. LET 100=#ADDRESS OF LINE_CLOCK_INTERRUPT
18685 ;. CLEAR INTERRUPT_FLAG
18686 ;. DO 3 TIMES
18687 ;. . REPEAT
18688 ;. . . DECREMENT R1
18689 ;. . . UNTIL R1 NE #0 OR LOW BYTE OF LKS LT #0
18690 ;. . . IF LOW BYTE OF LKS GE #0 THEN (READY DIDN'T COME UP)
18691 ;. . . ERROR
18692 ;. . . ENDDO
18693 ;. . . ENDDO
18694 ;. IF INTERRUPT_FLAG NE #0 THEN (INTERUPT W/O LKS<6>=1)
18695 ;. . ERROR
18696 ;. . ENDDO
18697 ;ENDROUTINE
18698
18699 ;*****
18700 112716 000004 TST34: SCOPE
18701 112720 032737 000100 000052 BIT #BIT06,#52 ;UFD MODE?
18702 112726 001404 BEQ 1$ ;IF NOT, GO DO TEST
18703 112730 032737 010000 177520 BIT #BIT12,BCSR ;LKS DISABLED?
18704 112735 001053 BNE TST35 ;IF DISABLED, EXIT TEST
18705
18706 ; TRY TO ACCESS LKS WITHOUT TIMEOUT
18707 ;
18708 112740 013737 000004 001160 1$: MOV ERRVEC,$TMP0 ;SAVE TIMEOUT VECTOR
18709 112746 012737 112770 000004 MOV #2$,ERRVEC ;POINT NEW VECTOR TO PROGRAM
18710 112754 012737 000340 000006 MOV #340,ERRVEC+2 ;AT PRIORITY 7
18711 112762 005737 177546 TST LKS ;ACCESS LKS
18712 112766 000403 BR 3$ ;IF NO TIMEOUT, CONTINUE
18713 112770 104056 2$: ERROR #56 ;TIMEOUT READING LKS
18714 112772 005726 TST (SP)+ ;ADJUST STACK POINTER
18715 112774 005726 TST (SP)+
18716 112776 013737 001160 000004 3$: MOV $TMP0,ERRVEC ;RESTORE TIMEOUT VECTOR
18717
18718 ; CHECK THAT READY BIT LKS<7> CAN BE 1
18719 ;
18720 113004 005037 002702 CLR LKSFL ;CLEAR INTERRUPT FLAG
18721 113010 012737 132444 000100 MOV #LKSINT,100 ;POINT VECTOR TO INTERRUPT ROUTINE
18722 113016 012737 000340 000102 MOV #340,100 ;AT PRIORITY 7
18723 113024 012702 000003 MOV #3,R2 ;DO 3 TIMES TO SYNCHRONISE
18724 113030 012701 077777 4$: MOV #77777,R1 ;COUNTER FOR SLOW CLOCKS
18725 113034 105737 177546 5$: TSTB LKS ;READY LKS<7>=1?

```

1312

CORDBO RDJ11 B CLUSTER MACY11 30(1046) 05 APR-84 16:45 PAGE 352
CORDBO P11 05-APR 84 16:45 TEST LKS BIT 7

SEQ 0351

18726	113040	100401			BMI	63
18727	113042	077104			SOB	R1,53
18728	113044	105737	117546	63:	TSIB	LKS
18729	113050	100401			BMI	73
18730	113052	104057			ERROR	57
18731	113054	077213		73:	SOB	R2,43
18732	113056	005737	002702	83:	TSI	LKS4
18733	113062	001401			REQ	TS135
18734	113064	104061			ERROR	61
18735						

IF 50, DO NEXT LOOP
OTHERWISE, GO THRU COUNT
WAS READY 1?
IF YES, BRANCH
LKS<07> DOES NOT BECOME 1
DO ALL 3 TIMES
ANY INTERRUPTS W/O LKS<6>=1?
IF NONE, EXIT TEST
ILLEGAL CLOCK INTERRUPTS

05 APR 84 16:45 PAGE 355
TEST LKS INTERRUPT PRIORITY

CLP

```

18736          .SMTTL TEST - LKS INTERRUPT PRIORITY
18737          .CHECK THAT LKS INTERRUPTS HAPPEN AT PRIORITY 5 CLEARING LKS<07>
18738          .AND DON'T HAPPEN AT PRIORITY 6.
18739          .ROUTINE TEST
18740          .IF W/D AND LKS IS DISABLED THEN
18741          .      .EXIT TEST
18742          .ENDIF
18743          .      .SET PRIORITY TO 5
18744          .      .CLEAR INTERRUPT FLAG
18745          .      .LET LKS<06>=01 (ENABLE INTERRUPTS)
18746          .      .SET COUNTER TO WAIT FOR 3 INTERRUPTS
18747          .      .REPEAT
18748          .          .      .DECREMENT COUNTER
18749          .          .UNTIL INTERRUPT_FLAG EQ 03 OR COUNTER EQ 00
18750          .          .CLEAR LKS<06>
18751          .          .IF LKS<07> EQ 01 THEN
18752          .              .      .ERROR (WAS NOT CLEARED ON INTERRUPT)
18753          .          .ENDIF
18754          .          .IF COUNTER LT TIME_REQUIRED FOR 3 INTERRUPTS FOR 800HZ
18755          .              .      .ERROR (INTERRUPTS NEVER GO LOW)
18756          .          .ENDIF
18757          .          .IF INTERRUPT_FLAG LT 03 THEN
18758          .              .      .ERROR (INTERRUPTS DON'T HAPPEN)
18759          .          .ENDIF
18760          .          .CLEAR INTERRUPT_FLAG
18761          .          .WAIT FOR LKS<7>=1
18762          .          .LIT LKS<7>=0
18763          .          .IF LKS<7> NE 00 THEN
18764          .              .      .ERROR (LKS<7> NOT CLEARED)
18765          .          .ENDIF
18766          .          .SET PRIORITY TO 6
18767          .          .SET COUNTER TO 1 SLOW CLOCK INTERRUPT
18768          .          .SET LKS<06>
18769          .          .REPEAT
18770          .              .      .DECREMENT COUNTER
18771          .              .UNTIL COUNTER EQ 00 OR INTERRUPT_FLAG NE 00
18772          .              .IF INTERRUPT_FLAG NE 00 THEN
18773          .                  .      .ERROR (INTERRUPT WAS AT WRONG PRIORITY)
18774          .              .ENDIF
18775          .              .RESTORE ORIGINAL PRIORITY
18776          .      .ENDROUTINE
18777          .
18778          .ROUTINE LINE_CLOCK_INTERRUPT
18779          .      .INCREMENT INTERRUPT_FLAG
18780          .      .ENDROUTINE
18781
18782          .*****
18783          113066 000004
18784          113070 032737 000100 000052
18785          113076 001404
18786          113100 032737 010000 177520
18787          113106 001121
18788
18789          .
18790          .ROUTINE LINE_CLOCK_INTERRUPT
18791          .      .INCREMENT INTERRUPT_FLAG
18792          .      .ENDROUTINE
18793
18794          .*****
18795          113135: SCOPE
18796          .      .BIT 0B1106,0052
18797          .      .BEQ 13
18798          .      .BIT 0B1112,BCSR
18799          .      .BNE 13136
18800          .      .IF NOT, DO DO TEST
18801          .      .IF NOT, DO DO TEST
18802          .      .IF NOT, DO DO TEST
18803          .      .IF NOT, DO DO TEST
18804          .      .IF NOT, DO DO TEST
18805          .      .IF NOT, DO DO TEST
18806          .      .IF NOT, DO DO TEST
18807          .      .IF NOT, DO DO TEST
18808          .      .IF NOT, DO DO TEST
18809          .      .IF NOT, DO DO TEST
18810          .      .IF NOT, DO DO TEST
18811          .      .IF NOT, DO DO TEST
18812          .      .IF NOT, DO DO TEST
18813          .      .IF NOT, DO DO TEST
18814          .      .IF NOT, DO DO TEST
18815          .      .IF NOT, DO DO TEST
18816          .      .IF NOT, DO DO TEST
18817          .      .IF NOT, DO DO TEST
18818          .      .IF NOT, DO DO TEST
18819          .      .IF NOT, DO DO TEST
18820          .      .IF NOT, DO DO TEST
18821          .      .IF NOT, DO DO TEST
18822          .      .IF NOT, DO DO TEST
18823          .      .IF NOT, DO DO TEST
18824          .      .IF NOT, DO DO TEST
18825          .      .IF NOT, DO DO TEST
18826          .      .IF NOT, DO DO TEST
18827          .      .IF NOT, DO DO TEST
18828          .      .IF NOT, DO DO TEST
18829          .      .IF NOT, DO DO TEST
18830          .      .IF NOT, DO DO TEST
18831          .      .IF NOT, DO DO TEST
18832          .      .IF NOT, DO DO TEST
18833          .      .IF NOT, DO DO TEST
18834          .      .IF NOT, DO DO TEST
18835          .      .IF NOT, DO DO TEST
18836          .      .IF NOT, DO DO TEST
18837          .      .IF NOT, DO DO TEST
18838          .      .IF NOT, DO DO TEST
18839          .      .IF NOT, DO DO TEST
18840          .      .IF NOT, DO DO TEST
18841          .      .IF NOT, DO DO TEST
18842          .      .IF NOT, DO DO TEST
18843          .      .IF NOT, DO DO TEST
18844          .      .IF NOT, DO DO TEST
18845          .      .IF NOT, DO DO TEST
18846          .      .IF NOT, DO DO TEST
18847          .      .IF NOT, DO DO TEST
18848          .      .IF NOT, DO DO TEST
18849          .      .IF NOT, DO DO TEST
18850          .      .IF NOT, DO DO TEST
18851          .      .IF NOT, DO DO TEST
18852          .      .IF NOT, DO DO TEST
18853          .      .IF NOT, DO DO TEST
18854          .      .IF NOT, DO DO TEST
18855          .      .IF NOT, DO DO TEST
18856          .      .IF NOT, DO DO TEST
18857          .      .IF NOT, DO DO TEST
18858          .      .IF NOT, DO DO TEST
18859          .      .IF NOT, DO DO TEST
18860          .      .IF NOT, DO DO TEST
18861          .      .IF NOT, DO DO TEST
18862          .      .IF NOT, DO DO TEST
18863          .      .IF NOT, DO DO TEST
18864          .      .IF NOT, DO DO TEST
18865          .      .IF NOT, DO DO TEST
18866          .      .IF NOT, DO DO TEST
18867          .      .IF NOT, DO DO TEST
18868          .      .IF NOT, DO DO TEST
18869          .      .IF NOT, DO DO TEST
18870          .      .IF NOT, DO DO TEST
18871          .      .IF NOT, DO DO TEST
18872          .      .IF NOT, DO DO TEST
18873          .      .IF NOT, DO DO TEST
18874          .      .IF NOT, DO DO TEST
18875          .      .IF NOT, DO DO TEST
18876          .      .IF NOT, DO DO TEST
18877          .      .IF NOT, DO DO TEST
18878          .      .IF NOT, DO DO TEST
18879          .      .IF NOT, DO DO TEST
18880          .      .IF NOT, DO DO TEST
18881          .      .IF NOT, DO DO TEST
18882          .      .IF NOT, DO DO TEST
18883          .      .IF NOT, DO DO TEST
18884          .      .IF NOT, DO DO TEST
18885          .      .IF NOT, DO DO TEST
18886          .      .IF NOT, DO DO TEST
18887          .      .IF NOT, DO DO TEST
18888          .      .IF NOT, DO DO TEST
18889          .      .IF NOT, DO DO TEST
18890          .      .IF NOT, DO DO TEST
18891          .      .IF NOT, DO DO TEST
18892          .      .IF NOT, DO DO TEST
18893          .      .IF NOT, DO DO TEST
18894          .      .IF NOT, DO DO TEST
18895          .      .IF NOT, DO DO TEST
18896          .      .IF NOT, DO DO TEST
18897          .      .IF NOT, DO DO TEST
18898          .      .IF NOT, DO DO TEST
18899          .      .IF NOT, DO DO TEST
18900          .      .IF NOT, DO DO TEST
18901          .      .IF NOT, DO DO TEST
18902          .      .IF NOT, DO DO TEST
18903          .      .IF NOT, DO DO TEST
18904          .      .IF NOT, DO DO TEST
18905          .      .IF NOT, DO DO TEST
18906          .      .IF NOT, DO DO TEST
18907          .      .IF NOT, DO DO TEST
18908          .      .IF NOT, DO DO TEST
18909          .      .IF NOT, DO DO TEST
18910          .      .IF NOT, DO DO TEST
18911          .      .IF NOT, DO DO TEST
18912          .      .IF NOT, DO DO TEST
18913          .      .IF NOT, DO DO TEST
18914          .      .IF NOT, DO DO TEST
18915          .      .IF NOT, DO DO TEST
18916          .      .IF NOT, DO DO TEST
18917          .      .IF NOT, DO DO TEST
18918          .      .IF NOT, DO DO TEST
18919          .      .IF NOT, DO DO TEST
18920          .      .IF NOT, DO DO TEST
18921          .      .IF NOT, DO DO TEST
18922          .      .IF NOT, DO DO TEST
18923          .      .IF NOT, DO DO TEST
18924          .      .IF NOT, DO DO TEST
18925          .      .IF NOT, DO DO TEST
18926          .      .IF NOT, DO DO TEST
18927          .      .IF NOT, DO DO TEST
18928          .      .IF NOT, DO DO TEST
18929          .      .IF NOT, DO DO TEST
18930          .      .IF NOT, DO DO TEST
18931          .      .IF NOT, DO DO TEST
18932          .      .IF NOT, DO DO TEST
18933          .      .IF NOT, DO DO TEST
18934          .      .IF NOT, DO DO TEST
18935          .      .IF NOT, DO DO TEST
18936          .      .IF NOT, DO DO TEST
18937          .      .IF NOT, DO DO TEST
18938          .      .IF NOT, DO DO TEST
18939          .      .IF NOT, DO DO TEST
18940          .      .IF NOT, DO DO TEST
18941          .      .IF NOT, DO DO TEST
18942          .      .IF NOT, DO DO TEST
18943          .      .IF NOT, DO DO TEST
18944          .      .IF NOT, DO DO TEST
18945          .      .IF NOT, DO DO TEST
18946          .      .IF NOT, DO DO TEST
18947          .      .IF NOT, DO DO TEST
18948          .      .IF NOT, DO DO TEST
18949          .      .IF NOT, DO DO TEST
18950          .      .IF NOT, DO DO TEST
18951          .      .IF NOT, DO DO TEST
18952          .      .IF NOT, DO DO TEST
18953          .      .IF NOT, DO DO TEST
18954          .      .IF NOT, DO DO TEST
18955          .      .IF NOT, DO DO TEST
18956          .      .IF NOT, DO DO TEST
18957          .      .IF NOT, DO DO TEST
18958          .      .IF NOT, DO DO TEST
18959          .      .IF NOT, DO DO TEST
18960          .      .IF NOT, DO DO TEST
18961          .      .IF NOT, DO DO TEST
18962          .      .IF NOT, DO DO TEST
18963          .      .IF NOT, DO DO TEST
18964          .      .IF NOT, DO DO TEST
18965          .      .IF NOT, DO DO TEST
18966          .      .IF NOT, DO DO TEST
18967          .      .IF NOT, DO DO TEST
18968          .      .IF NOT, DO DO TEST
18969          .      .IF NOT, DO DO TEST
18970          .      .IF NOT, DO DO TEST
18971          .      .IF NOT, DO DO TEST
18972          .      .IF NOT, DO DO TEST
18973          .      .IF NOT, DO DO TEST
18974          .      .IF NOT, DO DO TEST
18975          .      .IF NOT, DO DO TEST
18976          .      .IF NOT, DO DO TEST
18977          .      .IF NOT, DO DO TEST
18978          .      .IF NOT, DO DO TEST
18979          .      .IF NOT, DO DO TEST
18980          .      .IF NOT, DO DO TEST
18981          .      .IF NOT, DO DO TEST
18982          .      .IF NOT, DO DO TEST
18983          .      .IF NOT, DO DO TEST
18984          .      .IF NOT, DO DO TEST
18985          .      .IF NOT, DO DO TEST
18986          .      .IF NOT, DO
```

```

18792 113114 012737 132444 000100      MOV      0LKSINT,100      ;POINT VECTOR TO ROUTINE
18793 113122 012701 077777      MOV      077777,R1      ;COUNTER FOR SLOW CLOCK
18794 113126 052737 000100 177546      BIS      0BIT06,LKS      ;SET INTERRUPT ENABLE BIT
18795 113134 032737 000100 177546      BIT      0BIT06,LKS      ;BIT SET OK?
18796 113142 001001      BNE      2$      ;IF YES, BRANCH
18797 113144 104131      ERROR    +131      ;ERROR WRITING 1 TO LKS<6>
18798 113146 106427 000240      MIPS     0240      ;SET PRIORITY TO 5
18799 113152 022737 000003 002702 3$:  CMP      03,LKSFL      ;3 INTERRUPTS HAPPENED?
18800 113160 001401      BEQ      4$      ;IF YES, BRANCH
18801 113162 077105      SOB      R1,3$      ;STAY IN A LOOP
18802
18803      ;
18804      ; DISABLE INTERRUPTS AND CHECK THAT PROPER CONDITIONS ARE MET
18805 113164 106427 000340      4$:      MIPS     0340      ;RAISE PRIORITY
18806 113170 042737 000100 177546      BIC      0BIT06,LKS      ;DISABLE INTERRUPTS
18807 113176 032737 000200 177546      BIT      0BIT07,LKS      ;LKS<7> CLEARED AFTER INTERRUPTS?
18808 113204 001401      BEQ      5$      ;IF 0, BRANCH
18809 113206 104062      ERROR    +62      ;INTERRUPTS DON'T CLEAR LKS<7>
18810 113210 105737 177546      5$:      TSTB     LKS      ;LKS<7>=1?
18811 113214 100375      BPL      5$      ;IF NOT, WAIT
18812 113216 005037 177546      CLR      LKS      ;CLEAR LKS<7>
18813 113222 032737 000200 177546      BIT      0BIT07,LKS      ;LKS<7> CLEARED?
18814 113230 001401      BEQ      6$      ;IF YES, BRANCH
18815 113232 104060      ERROR    +60      ;LKS<7> NOT CLEARED ON WRITE
18816 113234 032737 000100 177546 6$:  BIT      0BIT06,LKS      ;LKS<6>=0?
18817 113242 001401      BEQ      7$      ;IF YES, BRANCH
18818 113244 104131      ERROR    +131      ;ERROR WRITING 0 TO LKS<6>
18819 113246 022701 077737      7$:      CMP      077737,R1      ;COUNTER AT LESS THAN 800H2?
18820 113252 002001      BGE      8$      ;IF NOT, BRANCH
18821 113254 104063      ERROR    +63      ;READY LINE DOES NOT GO LOW
18822 113256 022737 000003 002702 8$:  CMP      03,LKSFL      ;DID 3 INTERRUPTS HAPPEN?
18823 113264 001404      BEQ      9$      ;IF YES, BRANCH
18824 113266 012737 000003 001124      MOV      03,$GDDAT      ;3 INTERRUPTS EXPECTED
18825 113274 104064      ERROR    +64      ;INTERRUPTS DON'T HAPPEN
18826
18827      ;
18828      ; CHECK WHETHER INTERRUPTS HAPPEN AT PRIORITY 6
18829 113276 005037 002702      9$:      CLR      LKSFL      ;CLEAR INTERRUPT FLAG
18830 113302 106427 000300      MIPS     0300      ;RAISE PRIORITY TO 6
18831 113306 012701 077777      MOV      077777,R1      ;COUNTER FOR SLOW CLOCK
18832 113312 052737 000100 177546      BIS      0BIT06,LKS      ;SET INTERRUPT ENABLE BIT
18833 113320 005737 002702      10$:     TST      LKSFL      ;ANY INTERRUPTS?
18834 113324 001001      BNE      11$      ;IF YES, EXIT LOOP
18835 113326 077104      SOB      R1,10$      ;CONTINUE WITH COUNT
18836 113330 005737 002702      11$:     TST      LKSFL      ;ANY INTERRUPTS?
18837 113334 001404      BEQ      12$      ;IF NO, BRANCH
18838 113336 012737 000005 001124      MOV      05,$GDDAT      ;STORE PRIORITY FOR TYPE OUT
18839 113344 104065      ERROR    +65      ;INTERRUPTS HAPPEN AT WRONG PRIORITY
18840 113346 106427 000340      12$:     MIPS     0340      ;RESTORE PRIORITY
18841

```



```

18842 ;SBTTL TEST - LINE CLOCK DISABLE
18843 ;LINE CLOCK DISABLE(*)
18844 ;THIS TEST WILL CHECK THAT BCSR<12> DISABLES RESPONSE
18845 ;OF LKS REGISTER.
18846 ;
18847 ;BCSR <12> LINE CLOCK STATUS REGISTER DISABLE
18848 ;ROUTINE TEST
18849 ;IF UFD AND LKS IS NOT DISABLED THEN
18850 ;. EXIT TEST
18851 ;ENDIF
18852 ;. WRITE BCSR<12>=1
18853 ;. LET 4=ADDRESS OF LKS_TRAP
18854 ;. LET TRAP_LKS=0
18855 ;. READ LKS
18856 ;. IF TRAP_LKS NE 1 THEN
18857 ;. ERROR
18858 ;. ENDF
18859 ;ENDROUTINE
18860 ;
18861 ;ROUTINE LKS_TRAP
18862 ;. LET TRAP_LKS=1
18863 ;. LET BCSR<12>=0
18864 ;RTI
18865
18866 ;*****
18867 113352 000004 TST36: SCOPE
18868 113354 032737 000100 000052 BIT 0BIT06,0052 ;UFD MODE?
18869 113362 001404 BEQ 1$ ;IF NOT, BRANCH
18870 113364 032737 010000 177520 BIT 0BIT12,BCSR ;LKS DISABLED?
18871 113372 001052 BNE TST37 ;IF DISABLED, EXIT TEST
18872 ;
18873 ; CHECK BCSR<12> TO BE 0 AND 1
18874 ;
18875 113374 013737 177520 002710 1$: MOV BCSR,SAVBR ;SAVE BCSR REGISTER
18876 113402 042737 010000 177520 BIC 0BIT12,BCSR ;CLEAR BCSR
18877 113410 032737 010000 177520 BIT 0BIT12,BCSR ;<12>=0?
18878 113416 001403 BEQ 2$ ;IF OK, BRANCH
18879 113420 005037 001124 CLR $GDDAT ;CLEAR EXPECTED PATTERN
18880 113424 104052 ERROR +52 ;ERROR BCSR READ/WRITE BITS
18881 113426 052737 010000 177520 2$: BIS 0BIT12,BCSR ;SET BIT 12
18882 113434 032737 010000 177520 BIT 0BIT12,BCSR ;GOT SET OK?
18883 113442 001004 BNE 3$ ;IF OK, BRANCH
18884 113444 012737 010000 001124 MOV 0BIT12,$GDDAT ;EXPECTED PATTERN
18885 113452 104052 ERROR +52 ;ERROR BCSR READ/WRITE BITS
18886 ;
18887 ; TRY TO ACCESS LKS TO GET A TIMEOUT WITH BCSR<12>=1
18888 ;
18889 113454 013701 000004 3$: MOV ERRVEC,R1 ;SAVE TIMEOUT VECTOR
18890 113460 012737 113502 000004 MOV 04$,ERRVEC ;POINT TO PROGRAM AREA
18891 113466 012737 000340 000006 MOV 0340,ERRVEC+2 ;AT PRIORITY 7
18892 113474 005737 177546 TST LKS ;ACCESS LKS REGISTER
18893 113500 104066 ERROR +66 ;BCSR<12> DOES NOT DISABLE LKS

```

F12

COKDABO KDJ11 B CLUSTER MACY11 30(1046) 05 APR-84 16:45 PAGE 356
COKDAB.P11 05-APR-84 16:45 TEST : LINE CLOCK DISABLE

SEQ 0355

18894	113502	005726		4\$:	TST	(SP)+	;RESTORE STACK POINTER
18895	113504	005726			TST	(SP)+	
18896	113506	010137	000004		MOV	R1,ERRVEC	;RESTORE TIMEOUT VECTOR
18897	113512	013737	002710 177520		MOV	SAVBR,BCSR	;RESTORE BCSR
18898							
18899							

COKDABO KDJ11-B CLUSTER MACY11 30(1046)
COKDAB.P11 05-APR-84 16:45

05-APR-84 16:45 PAGE 357
TEST - UNCONDITIONAL CLOCK LINE INTERRUPTS

SEQ 0356

```

18900 .SBITL TEST - UNCONDITIONAL CLOCK LINE INTERRUPTS
18901 ;UNCONDITIONAL CLOCK LINE INTERRUPTS(*)
18902 ;THIS TEST WILL CHECK THAT SETTING BCSR<13> TO 1 WILL
18903 ;REQUEST INTERRUPTS WHENEVER A CLOCK LINE IS ASSERTED. THIS
18904 ;SHOULD HAPPEN WITHOUT ACCESSING LKS, THEREFORE, EVEN WITH BCSR<12>=1
18905 ;INTERRUPTS SHOULD HAPPEN.
18906 ;
18907 ;BCSR <13> FORCE LINE CLOCK INTERRUPT ENABLE
18908 ;
18909 ;
18910 ;ROUTINE TEST
18911 ;IF UFD AND FORCE LKS NOT DISABLED THEN
18912 ;. EXIT TEST
18913 ;ENDIF
18914 ;. LET 100=ADDRESS OF UNCONDITIONAL_INTERRUPT_ROUTINE
18915 ;. DO FOR BCSR<12> FROM #0 TO #1(LKS DISABLED AND ENABLED)
18916 ;. (IF UFD DO ONLY FOR SELECTED LINE CLOCK)
18917 ;. . CLEAR UNCONDITIONAL_INTERRUPT
18918 ;. . SET COUNTER TO WAIT FOR 3 INTERRUPTS
18919 ;. . LET BCSR<13>=1
18920 ;. . REPEAT
18921 ;. . . DECREMENT COUNTER
18922 ;. . . UNTIL UNCONDITIONAL_INTERRUPT EQ #3 OR COUNTER EQ #0
18923 ;. . . IF COUNTER GT TIME_REQUIRED_FOR_3_INTERRUPTS_FOR_800HZ
18924 ;. . . ERROR (INTERRUPTS NEVER GO LOW)
18925 ;. . . ENDIF
18926 ;. . . IF UNCONDITIONAL_INTERRUPT LT #3 THEN
18927 ;. . . ERROR (INTERRUPTS DON'T HAPPEN)
18928 ;. . . ENDIF
18929 ;. . LET BCSR<13>=#0
18930 ;. . ENDDO
18931 ;ENDROUTINE
18932 ;
18933 ;ROUTINE UNCONDITIONAL_INTERRUPT_ROUTINE
18934 ;. INCREMENT UNCONDITIONAL_INTERRUPT
18935 ;RETURN
18936 ;
18937 ;*****
18938 113520 000004 TST37: SCOPE
18939 113522 032737 000100 000052 BIT #BIT06,#052 ;UFD MODE?
18940 113530 001404 BEQ 1$ ;IF NOT, BRANCH
18941 113532 032737 020000 177520 BIT #BIT13,BCSR ;FORCE INTERRUPT SET?
18942 113540 001525 BEQ TST40 ;IF SET, EXIT TEST
18943 ;
18944 ; CHECK BCSR<13> TO BE 0 AND 1
18945 ;
18946 113542 013737 177520 002710 1$: MOV BCSR,SAVBR ;SAVE BCSR REGISTER
18947 113550 042737 020000 177520 BIC #BIT13,BCSR ;CLEAR BCSR
18948 113556 032737 020000 177520 BIT #BIT13,BCSR ;<13>=0?
18949 113564 001403 BEQ 2$ ;IF OK, BRANCH
18950 113566 005037 001124 CLR $GDDAT ;CLEAR EXPECTED PATTERN
18951 113572 104052 ERROR +52 ;ERROR BCSR READ/WRITE BITS
18952 113574 052737 020000 177520 2$: BIS #BIT13,BCSR ;SET BIT 13
18953 113602 032737 020000 177520 BIT #BIT13,BCSR ;GOT SET OK?
18954 113610 001004 BNE 3$ ;IF OK, BRANCH
18955 113612 012737 020000 001124 MOV #BIT13,$GDDAT ;EXPECTED PATTERN

```

COKDABO KDJ11 B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 358
COKDAB.P11 05-APR-84 16:45 TEST - UNCONDITIONAL CLOCK LINE INTERRUPTS

SEQ 0357

Address	Hex Address	Hex Address	Hex Address	Hex Address	Label	Instruction	Comment
18956	113620	104052				ERROR +52	;ERROR BCSR READ/WRITE BITS
18957							
18958							
18959							
18960	113622	012737	132444	000100	3\$:	MOV #LKSINT, @#100	;SET UP INTERRUPT VECTOR
18961	113630	012737	000340	000102		MOV #340, @#102	;AT PRIORITY 7
18962	113636	052737	010000	177520		BIS #BIT12,BCSR	;FOR 1ST TIME DISABLE LKS
18963	113644	000403				BR 5\$	;GO DO IT
18964	113646	042737	010000	177520	4\$:	BIC #BIT12,BCSR	;FOR THE 2ND TIME, ENABLE LKS
18965	113654	005037	002702		5\$:	CLR LKSFL	;CLEAR INTERRUPTS FLAG
18966	113660	012702	077777			MOV #77777,R2	;COUNTER TO WAIT FOR INTERRUPTS
18967	113664	106427	000240			MTPS #240	;LOWER PRIORITY TO 5
18968	113670	022737	000003	002702	6\$:	CMP #3,LKSFL	;3 INTERRUPTS HAPPENED?
18969	113676	001401				BEQ 7\$	;EXIT LOOP, IF SO
18970	113700	077205				S0B R2,6\$	;OTHERWISE, KEEP WAITING
18971	113702	106427	000340		7\$:	MTPS #340	;RAISE PRIORITY TO 7
18972	113706	022702	077700			CMP #77700,R2	;INTERRUPTS HAPPEN TOO OFTEN?
18973	113712	002001				BGE 8\$	;IF NOT, BRANCH
18974	113714	104063				ERROR +63	;READY LINE DOESN'T GO LOW
18975	113716	022737	000003	002702	8\$:	CMP #3,LKSFL	;AT LEAST 3 INTERRUPTS HAPPENED?
18976	113724	002004				BGE 9\$	;IF SO, BRANCH
18977	113726	012737	000003	001124		MOV #3,\$GDDAT	;EXPECTED DATA
18978	113734	104064				ERROR +64	;INTERRUPTS DON'T HAPPEN
18979	113736	032737	010000	177520	9\$:	BIT #BIT12,BCSR	;SECOND TIME THRU THE LOOP?
18980	113744	001340				BNE 4\$	;IF NOT, DO IT AGAIN
18981	113746	032737	000100	000052		BIT #BIT06, @#52	;UFD MODE?
18982	113754	001404				BEQ 10\$	;IF NOT, BRANCH
18983	113756	032737	010000	177520		BIT #BIT12,BCSR	;IF UFD AND LKS DISABLED?
18984	113764	001010				BNE 12\$	;DON'T CHECK LKS
18985	113766	032737	000100	177546	10\$:	BIT #BIT06,LKS	;INTERRUPT ENABLE LINE HOLD 1?
18986	113774	001001				BNE 11\$	;IF SO, BRANCH
18987	113776	104067				ERROR +67	;BCSR<13> DOESN'T SET LKS<6>
18988	114000	042737	000100	177546	11\$:	BIC #BIT06,LKS	;DISABALE LKS INTERRUPTS
18989	114006	013737	002710	177520	12\$:	MOV SAVBR,BCSR	;RESTORE BCSR
18990							
18991							

COKDABO KDJ11-B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 359
 COKDAB.P11 05-APR-84 16:45 TEST - RESETTING LKS

SEQ 0358

```

18992 .SBTTL TEST - RESETTING LKS
18993 ;RESETTING LKS(*)
18994 ;THIS TEST WILL PROVE THAT RESET INSTRUCTION SETS LKS<07> AND
18995 ;CLEARS LKS<06>.
18996 ;ROUTINE TEST
18997 ;IF UFD AND LKS IS DISABLED THEN
18998 ;. EXIT TEST
18999 ;ENDIF
19000 ;. POINT LKS VECTOR 100 TO ERROR_LKS_ILLEGAL_INTERRUPT
19001 ;. SYNCHRONIZE LKS BY WAITING FOR 3 PULSES
19002 ;. LET LKS<06>=#1
19003 ;. CLEAR LKS (CLEARS LKS<07>)
19004 ;. EXECUTE "RESET"
19005 ;. IF LKS<7> NE #1 OR LKS<6> NE #0 THEN
19006 ;. ERROR
19007 ;. ENDIF
19008 ;. IF ILLEGAL_LINE_CLOCK_INTERRUPT NE 0 THEN
19009 ;. ERROR
19010 ;. ENDIF
19011 ;ENDROUTINE
19012 ;
19013 ;ROUTINE ERROR_LKS_ILLEGAL_INTERRUPT
19014 ;. FLAG ILLEGAL_LINE_CLOCK_INTERRUPT
19015 ;RETURN
19016
19017 ;*****
19018 114014 000004 TST40: SCOPE
19019 114016 122737 000001 001220 CMPB #APTENV,$ENV ;RUNNING IN APT MODE?
19020 114024 001013 BNE 1$ ;NO, GO DO TEST
19021 114026 005737 001206 TST $PASS ;FIRST PASS?
19022 114032 001057 BNE TST41 ;IF APT AND NOT FIRST PASS, EXIT TEST
19023 114034 032737 000100 000052 BIT #BIT06,#052 ;UFD MODE?
19024 114042 001404 BEQ 1$ ;IF NOT, BRANCH
19025 114044 032737 010000 177520 BIT #BIT12,BCSR ;LKS DISABLED?
19026 114052 001447 BEQ TST41 ;IF DISABLED, EXIT TEST
19027
19028 ; SYNCHRONISE WITH LINE TIME CLOCK BY WAITING FOR 3 INTERRUPTS
19029 ;
19030 114054 013737 177520 002710 1$: MOV BCSR,SAVBR ;SAVE BCSR
19031 114062 042737 010000 177520 BIC #BIT12,BCSR ;ENABLE LKS RESPONSE
19032 114070 012737 132444 000100 MOV #LKSINT,#0100 ;SET UP INTERRUPT VECTOR
19033 114076 012737 000340 000102 MOV #340,#0102 ;AT PROIRITY 7
19034 114104 052737 000100 177546 BIS #BIT06,LKS ;SET INTERRUPT ENABLE BIT
19035 114112 005037 002702 CLR LKSFL ;CLEAR INTERRUPTS FLAG
19036 114116 012702 077777 MOV #77777,R2 ;COUNTER TO WAIT FOR INTERRUPTS
19037 114122 106427 000240 MTPS #240 ;LOWER PRIORITY TO 5
19038 114126 022737 000003 002702 2$: CMP #3,LKSFL ;3 INTERRUPTS HAPPENED?
19039 114134 001401 BEQ 3$ ;EXIT LOOP, IF SO
19040 114136 077205 SOB R2,2$ ;OTHERWISE, KEEP WAITING
19041 114140 106427 000340 3$: MTPS #340 ;RAISE PRIORITY TO 7
19042 114144 000005 RESET ;EXECUTE RESET
19043 114146 032737 000200 177546 BIT #BIT07,LKS ;READY BIT SET?
19044 114154 001001 BNE 4$ ;IF SO, BRANCH
19045 114156 104070 ERROR #70 ;RESET DOESN'T SET LKS<07>
19046 114160 032737 000100 177546 4$: BIT #BIT06,LKS ;INTERRUPT ENABLE BIT CLEARED
19047 114166 001401 BEQ TST41 ;IF SO, EXIT TEST

```

312
COKDABO KDJ11 B CLUSTER MAC111 30(1046) 05-APR-84 16:45 PAGE 360
COKDAB.P11 05-APR-84 16:45 TEST - RESETTING LKS

SEQ 0359

19048 114170 104071
19049
19050

ERROR +71

;RESET DOESN'T CLEAR LKS

K12

COKDABO KDJ11-B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 361
 COKDAB.P11 05-APR-84 16:45 TEST - LINE CLOCK INTERRUPTS

SEQ 0360

```

19051 .SBTTL TEST - LINE CLOCK INTERRUPTS
19052 ;LINE CLOCK INTERRUPTS(*)
19053 ;BY SETTING TO 1 LKS<06>, THIS TEST WILL CHECK FOR INTERRUPTS
19054 ;FROM BEVENT LINE AND FROM KDJ11-B 50HZ, 60HZ, 800HZ ON BOARD SIGNALS
19055 ;(THE LATTER SIGNALS WILL BE ACCESSED BY SETTING BCSR<11-10>).
19056 ;
19057 ;BCSR <11> <10>          CLOCK SELECT BITS 1 AND 0
19058 ;
19059 ;      0      0          EXTERNAL BEVENT LINE
19060 ;      0      1          ON-BOARD 50 HZ
19061 ;      1      0          ON-BOARD 60 HZ
19062 ;      1      1          ON-BOARD 800 HZ
19063 ;
19064 ;ROUTINE TEST
19065 ;IF UFD THEN
19066 ;.    IF LKS DISABLED THEN
19067 ;.      EXIT TEST
19068 ;.    ENDIF
19069 ;.    SET FLAGS TO RUN ONLY WHAT SPECIFIED IN EAPROM
19070 ;ENDIF
19071 ;.    LET 100=ADDRESS OF LKS_INTERRUPT
19072 ;.    DO FOR BCSR<11;10> FROM #0 TO #3
19073 ;.      LET LKS<06>=#1
19074 ;.      WAIT FOR 10 INTERRUPTS FOR EACH CLOCK
19075 ;.      STORE ACTUAL NUMBER OF INTERRUPTS FOR EACH CLOCK
19076 ;.      LET INTERRUPT_FLAG=0
19077 ;.    ENDDO
19078 ;.    COMPARE NUMBER OF INTERRUPTS FOR EACH CLOCK
19079 ;ENDROUTINE
19080 ;
19081 ;ROUTINE LKS_INTERRUPT
19082 ;.    INCREMENT INTERRUPT_FLAG
19083 ;RETURN
19084 ;
19085 ;*****
19086 114172 000004 TST41: SCOPE
19087 114174 032737 000100 000052 BIT    #BIT06,#052          ;UFD MODE?
19088 114202 001411 BEQ     1$              ;IF NOT, GO DO THE TEST
19089 114204 032737 010000 177520 BIT    #BIT12,BCSR          ;LKS IS DISABLED?
19090 114212 001123 BNE     TST42              ;;IF DISABLED, EXIT TESTS
19091 114214 005002 CLR     R2                  ;CLEAR R2 TO SET FLAGS
19092 114216 053702 177520 BIS     BCSR,R2          ;SET R2 ACCORDING TO BCSR
19093 114222 042702 171777 BIC     #171777,R2        ;LEAVE ONLY BITS <11-10>
19094 ;
19095 ; SETUP FOR INTERRUPTS, IN UFD MODE ONLY FROM THE CLOCK SPECIFIED IN BCSR<11-10>
19096 ;
19097 114226 013737 177520 002710 1$: MOV    BCSR,SAVBR          ;STORE BCSR
19098 114234 012737 132444 000100 MOV    #LKSINT,100        ;SET UP LKS VECTOR
19099 114242 012737 000340 000102 MOV    #340,102          ;AT PRIORITY 7
19100 114250 012705 114452 MOV    #TIMDEL,R5        ;POINTER TO DEL
19101 114254 012704 000004 MOV    #4,R4             ;R4 IS THE COUNTER FOR ALL CLOCKS
19102 114260 005037 177520 CLR    BCSR              ;DO FOR BEVENT LINE INTERRUPTS
19103 114264 000403 BR      3$                  ;GO DO THE LOOP
19104 114266 062737 002000 177520 2$: ADD    #2000,BCSR        ;SET UP FOR THE NEXT CLOCK LINE
19105 114274 032737 000100 000052 3$: BIT    #BIT06,#052          ;UFD MODE?
19106 114302 00140 BEQ     4$              ;IF NOT, BRANCH

```

COKDABO KDJ11-B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 362
 COKDAB,P11 05-APR-84 16:45 TEST - LINE CLOCK INTERRUPTS

SEQ 0361

```

19107 114304 010237 177520      MOV      R2,PCSR
19108 114310 005037 002702      CLR      LKSF'
19109 114314 052737 000100 177546 4$:    BIS      @BIT06,LKS
19110 114322 012703 000010      MOV      @10,R3
19111 114326 012701 177777      5$:    MOV      @177777,R1
19112 114332 106427 000240      MIPS     @240
19113 114336 023703 002702      6$:    CMP      LKSF',R3
19114 114342 001401              BEQ      7$
19115 114344 077104              SOB      R1,6$
19116 114346 010125              7$:    MOV      R1,(R5)+
19117 114350 032737 000100 000052      BIT      @BIT06,@052
19118 114356 001026              BNE      10$
19119 114360 077436              SOB      R4,2$
19120
19121      ; CHECK THE DELAY VALUES FOR ALL CLOCKS
19122      ;
19123 114362 106427 000340      MIPS     @340
19124 114366 042737 000100 177546      BIC      @BIT06,LKS
19125 114374 012705 114452      MOV      @TIMDEL,R5
19126 114400 021565 000006      CMP      (R5),6(R5)
19127 114404 103401              BLO      8$
19128 114406 104064              ERROR    +64
19129 114410 005725              8$:    TST      (R5)+
19130 114412 021565 000002      CMP      (R5),2(R5)
19131 114416 103401              BLO      9$
19132 114420 104064              ERROR    +64
19133 114422 005725              9$:    TST      (R5)+
19134 114424 021565 000002      CMP      (R5),2(R5)
19135 114430 103401              BLO      10$
19136 114432 104064              ERROR    +64
19137 114434 013737 002710 177520 10$:   MOV      SAVBR,BCSR
19138 114442 042737 000100 177546      BIC      @BIT06,LKS
19139 114450 000404              BR       IST42
19140
19141 114452 000004              TIMDEL: .BLKW 4
19142

```

```

; IN UFD, DO ONLY FOR SPECIFIED
; CLEAR INTERRUPT FLAG
; SET INTERRUPT ENABLE BIT
; START COUNTER FOR 10 INTERRUPTS
; START COUNTER TO WAIT FOR INTERRUPT
; LOWER PRIORITY TO 5
; NEW INTERRUPT HAPPENED?
; IF SO, EXIT WAIT LOOP
; OTHERWISE, KEEP WAITING
; STORE DELAY FOR EACH CLOCK
; UFD MODE?
; IF UFD, DON'T DO FOR ANY OTHER
; ALL LINE CLOCKS DONE?

```

```

; RAISE PRIORITY
; DISABLE INTERRUPTS
; POINTER TO DELAY TABLE
; DELAY FOR BEVENT AND 800HZ?
; BEVENT IS NOT 800HZ
; WRONG # OF INTERRUPTS
; INCREMENT POINTER
; DELAY FOR 50HZ AND 60HZ?
; IF FIRST BIGGER, BRANCH
; WRONG # OF INTERRUPTS
; INCREMENT POINTER
; DELAY FOR 50HZ AND 800HZ
; IF FIRST BIGGER, BRANCH
; WRONG # OF INTERRUPTS
; RESTORE BCSR
; DISABLE INTERRUPTS

```

;;EXIT TEST

M12

COKDABO KDJ11 B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 363
COKDAB.P11 05-APR-84 16:45 TEST - MAINTENANCE REGISTER TEST

SEQ 0362

```

19143 .SBTTL TEST - MAINTENANCE REGISTER TEST
19144 ;MAINTENANCE REGISTER TEST
19145 ;THIS TEST WILL ADDRESS MAINTENANCE REGISTER AND CHECK BITS
19146 ;7-4 TO BE 0010, 2-1 TO BE 10, AND READ BITS 10-08, 03, 00
19147 ;FOR FUTURE USE. THOSE BITS REPRESENT THE FOLLOWING SIGNALS:
19148 ;MULTIPROCESSOR SLAVE, UNIBUS SYSTEM, FPA AVAILABLE, HALT/TRAP
19149 ;OPTION, AND AC POWER OKAY.
19150 ;ROUTINE TEST
19151 ;. IF MAINT. REG. BITS <7-4> NE 0010 OR <2-1> NE 10 THEN
19152 ;. ERROR
19153 ;. ENDIF
19154 ;. READ MAINT.REG. BITS <10-08,03,00>
19155 ;ENDROUTINE
19156
19157 ;*****
19158 114462 000004 TST42: SCOPE
19159 114464 032737 174000 177750 BIT #174000,MAIREG ;UNUSED BITS ALL ZEROS?
19160 114472 001401 BEQ 1$ ;IF OK, BRANCH
19161 114474 104132 ERROR +132 ;MAINTENANCE REGISTER ERROR
19162 114476 032737 000044 177750 1$: BIT #44,MAIREG ;<5,2> SET ?
19163 114504 001001 BNE 2$ ;IF SO, BRANCH
19164 114506 104132 ERROR +132 ;MAINTENANCE REGISTER ERROR
19165 114510 032737 000322 177750 2$: BIT #322,MAIREG ;<7,6,4,1> CLEAR?
19166 114516 001401 BEQ TST43 ;IF YES, BRANCH
19167 114520 104132 ERROR +132
19168
19169

```

N12

COKDABO KDJ11-B CLUSTER MACY11 30(1046)
COKDAB.P11 05-APR-84 16:45

05-APR-84 16:45 PAGE 364
TEST - SERIAL LINE UNIT REGISTERS

SEQ 0363

```

19170 .SBTTL TEST - SERIAL LINE UNIT REGISTERS
19171 ;SERIAL LINE UNIT TEST(*)
19172 ;BCR<2-0> WILL BE READ TO FIND OUT BAUD RATE. SLU WILL BE PROG-
19173 ;RAMMED TO CHECK THE INTERRUPT LEVELS BY SETTING BIT<06> IN
19174 ;RCSR AND XMIT. LOOP BACK CAPABILITIES WILL BE TESTED BY SETTING
19175 ;TO 1 XCSR<02>. THE LINE CLOCK INTERRUPT SUBROUTINE WILL BE
19176 ;USED TO RETURN TO THE EXECUTION OF THE DIAGNOSTICS, IF THE
19177 ;PROGRAM HANGS IN THE LOOP BACK MODE.
19178 ;ROUTINE TEST
19179 ;IF UFD AND CONSOLE NOT PRESENT
19180 ;. GO TO TEST_22
19181 ;ENDIF
19182 ;. IF BCR<07> EQ #0 THEN
19183 ;. READ BCR<2-0> TO GET BAUD RATE
19184 ;. ENDIF
19185 ;. LET 4-ADDRESS_OF_TIMEOUT_ROUTINE
19186 ;. DO FOR RCSR,XCSR,RBUF,XBUF
19187 ;. READ XCSR,XCSR,RBUF,XBUF
19188 ;. IF TIMEOUT_FLAG NE #0 THEN
19189 ;. ERROR
19190 ;. ENDIF
19191 ;. ENDDO
19192 ;ENDROUTINE
19193 ;
19194 ;ROUTINE TIMEOUT
19195 ;. LET TIMEOUT_FLAG=#1
19196 ;ENDROUTINE
19197 ;
19198 ;*****
19199 114522 000004 TST43: SCOPE
19200 114524 032737 000100 000052 BIT #BIT06,#052 ;UFD MODE?
19201 114532 001406 BEQ 1$ ;IF NOT, DO THE TEST
19202 114534 032737 000200 177524 BIT #BIT07,BCR ;IF UFD AND CONSOLE NOT PRESENT
19203 114542 001402 BEQ 1$ ;NOT TRUE, DO THE TEST
19204 114544 000137 116510 JMP $END ;IF TRUE, SKIP ALL SLU TESTS
19205 ;
19206 ; TRY TO ACCESS SLU REGISTERS
19207 ;
19208 114550 013701 000004 1$: MOV ERRVEC,R1 ;SAVE TIMEOUT VECTOR
19209 114554 012737 114600 000004 MOV #3$,ERRVEC ;POINT NEW ONE TO PROGRAM AREA
19210 114562 012737 000340 000006 MOV #340,ERRVEC+2 ;AT PRIORITY 7
19211 114570 012702 177560 MOV #RCSR,R2 ;START ACCESSING WITH RCSR
19212 114574 005712 2$: IST (R2) ;ACCESS SLU REGISTER
19213 114576 000403 BR 4$ ;IF NO TIMEOUT, CONTINUE
19214 114600 010237 001126 3$: MOV R2,$BDDAT ;STORE ADDRESS THAT TIMED OUT
19215 114604 104072 ERROR +72 ;TIMEOUT ACCESSING SLU REGISTER
19216 114606 022722 177566 4$: CMP #XBUF,(R2)+ ;LAST REGISTER ACCESSED?
19217 114612 103770 BLO 2$ ;IF NOT, BRANCH
19218 114614 010137 000004 MOV R1,ERRVEC ;RESTORE TIMEOUT VECTOR
19219

```


1313

COKDABO RDJ11 B CLUSTER MACY11 30(1046) 05 APR 84 16:45 PAGE 365
COKDAB.P11 05 APR 84 16:45 TEST XCSR BIT 7

SEQ 03F4

```

19230 .SBTTL TEST XCSR BIT 7
19231 ;CHECK THAT XCSR<07> CAN BE 0 AND 1.
19232 ;
19233 ;XCSR <07> TRANSMITTER READY
19234 ;
19235 ;ROUTINE TEST
19236 ;. WAIT FOR XCSR<07>=01 NO MORE THAN 200MSEC
19237 ;. IF XCSR<07> NE 01 THEN
19238 ;. ERROR
19239 ;. ENDIF
19240 ;. LET /BUF=0NULL
19241 ;. WAIT FOR XCSR<07>=01
19242 ;. LET XBUF=0NULL
19243 ;. IF XCSR<07> NE 0 THEN
19244 ;. ERROR (READY DIDN'T GO LOW)
19245 ;. ENDIF
19246 ;ENDROUTINE
19247
19248 *****
19249 TST44: SCOPE
19250 MOV 01000,R1 ;COUNTER FOR ABOUT 200MICROSEC.
19251 CMPB 0APTEHV,$ENV ;RUNNING IN APT MODE?
19252 BNE 1$ ;NO, GO DO TEST
19253 TST $PASS ;FIRST PASS?
19254 BNE TST45 ;IF APT AND NOT FIRST PASS, EXIT TEST
19255 TSTB XCSR ;XCSR<7> READY?
19256 BMI 2$ ;IF SO, EXIT WAIT LOOP
19257 SOB R1,1$ ;IF NOT 1, CONTINUE WAITING
19258 TSTB XCSR ;XCSR<7>=1?
19259 BMI 3$ ;IF YES, BRANCH
19260 ERROR 223 ;XCSR<7> DOES NOT BECOME 1
19261 MOV 0NULL,XBUF ;TRY TO TRANSMIT NULL CHARACTER
19262 TSTB XCSR ;XCSR<7>=0
19263 BPL TST45 ;IF YES, EXIT TEST
19264 ERROR 223 ;XMIT READY DIDN'T GO LOW
19265
19266

```

C15

COKDABO RDJ11 B CLUSTER MAC111 5001046)
COKDAB.P11 05 APR 84 16:45

05 APR 84 16:45 PAGE 366
TEST RCSR BIT 7 AND XCSR BIT 2

SEQ 0365

19257
19258
19259
19260
19261
19262
19263
19264
19265
19266
19267
19268
19269
19270
19271
19272
19273
19274
19275
19276
19277
19278
19279
19280
19281
19282
19283 114702 000004
19284 114704 012701 000013
19285 114710 122737 000001 001220
19286 114716 001003
19287 114720 005737 001206
19288 114724 001074
19289 114726 105737 177564
19290 114732 100375
19291 114734 052737 000004 177564
19292 114742 032737 000004 177564
19293 114750 001004
19294 114752 005037 177564
19295 114756 104114
19296 114760 000456
19297
19298
19299
19300 114762 012701 002000
19301 114766 105737 177560
19302 114772 100402
19303 114774 077104
19304 114776 000402
19305 115000 005737 177562
19306
19307
19308
19309 115004 012737 000021 177566
19310 115012 012701 002000
19311 115016 105737 177560
19312 115022 100401

```

;SBIT TEST RCSR BIT 7 AND XCSR BIT 2
;CHECK THAT RCSR<07> CAN BE 0 AND 1 AND THAT XCSR<02> WORKS PROPERLY.
;
;RCSR <07> RECEIVER DONE
;XCSR <02> MAINTENANCE
;
;ROUTINE TEST
;.(CHECK RCSR<07> AND XCSR<07>)
;
;    WAIT FOR XCSR<07>=01
;    LET XCSR<02>=01 (LOOP BACK MODE)
;    LET XBUF=0125
;    WAIT FOR RCSR<07>=01 NO MORE THAN 200MSEC
;    IF RCSR<07> NE 01 THEN
;        . ERROR (RCSR<07> DOES NOT BECOME 1 OR XCSR<02> DOES NOT
;        . WORK)
;    ENDIF
;    IF RBUF NE 0125 THEN
;        . ERROR
;    ENDIF
;    IF RCSR<07> NE 00 THEN
;        . ERROR (RCSR<07> DOES NOT GO LOW)
;    ENDIF
;    LET XCSR<02>=00
;ENDROUTINE

```

```

;*****
;TST45: SCOPE
;MOV 015,R1 ;COUNTER FOR ABOUT 200MICROSEC.
;CMPB 0APTENV,ENV ;RUNNING IN APT MODE?
;BNE 11 ;NO, GO DO TEST
;TST 0PASS ;FIRST PASS?
;BNE TST46 ;IF APT AND NOT FIRST PASS, EXIT TEST
;TSTB XCSR ;XCSR<7> READY?
;BPL 11 ;IF NOT 1, CONTINUE WAITING
;BIT 0BIT02,XCSR ;SET LOOP BACK MODE
;BIT 0BIT02,XCSR ;NOT SET OK?
;BNE 31 ;IF YES, BRANCH
;CLR XCSR ;RESET TO PRINT ERROR
;ERROR 0114 ;XCSR<2> DOES NOT BECOME 1
;BR TST46 ;EXIT TEST
;
;STALL FOR A WHILE IN CASE XCSR<2> CAUSES RCSR<7> TO BE 1
;
;31: MOV 02000,R1 ;STALL IN CASE XCSR<2> SETS READY
;41: TSTB RCSR ;IF RECEIVER READY SET?
;BMI 51 ;IF SET, BRANCH
;SOB R1,41 ;OTHERWISE, STAY FOR A WHILE
;BR 01 ;IF NOT READY, BRANCH
;51: TST RBUF ;READ RBUF
;
;TRANSMIT XON AND CHECK RCSR<7>
;
;61: MOV 021,XBUF ;TRANSMIT A CHARACTER
;MOV 02000,R1 ;COUNTER TO WAIT
;71: TSTB RCSR ;RCSR<7> READY?
;BMI 81 ;IF YES, EXIT WAIT LOOP

```

115

CORDABO ADJ11 B CLUSTER MACY11 30(1046) 05 APR 84 16:45 PAGE 367
CORDAB,P11 05-APR 84 16:45 :ESI RCSR BIT 7 AND XCSR BIT 2

SEQ 0366

19313	115024	077104			SOR	R1,75	;OTHERWISE, CONTINUE WAITING
19314	115026	105737	177560		TSIB	RCSR	;RCSR<7>*1?
19315	115032	100403		8\$:	BMI	9\$	;IF YES, BRANCH
19316	115034	005037	177564		CLR	XCSR	;RESET XCSR<2>
19317	115040	104074			ERROR	*74	;RECEIVER READY: DIDN'T COME UP
19318	115042	013737	177562	001126	MOV	RBUF, \$BDDAT	;STORE RECEIVED DATA
19319	115050	022737	000021	001126	CMP	@21, \$BDDAT	;DATA RECEIVED OK?
19320	115056	001406			BEQ	10\$	;IF YES, BRANCH
19321	115060	012737	000021	001124	MOV	@21, \$GDDAT	
19322	115066	005037	177564		CLR	XCSR	;RESET TO ENABLE SLO
19323	115072	104075			ERROR	*75	;WRONG CHARACTER RECEIVED
19324	115074	105737	177560		TSIB	RCSR	;RCSR<7>*0?
19325	115100	100003			BPL	11\$	;IF ZERO, BRANCH
19326	115102	005037	177564		CLR	XCSR	;RESET TO ENABLE SLO
19327	115106	104076			ERROR	*76	;RCSR<0?>*0 AFTER READING RBUF
19328	115110	042737	000004	177564	BIC	@BIT02,XCSR	;DISABLE LOOP BACK MODE
19329							
19330							

COKDABO KDJ11-8 CLUSTER MACY11 30(1046) 05 APR-84 16:45 PAGE 368
COKDAB.P11 05-APR-84 16:45 TEST - RESET AND XCSR<2!0>

SEQ 0367

```

19331 .SRTTL TEST - RESET AND XCSR<2!0>
19332 ;CHECK THAT RESET CLEARS XCSR<0!2>.
19333 ;ROUTINE TEST
19334 ;.(CHECK RCSR<07> AND XCSR<07> AND RESET)
19335 ;. LET XCSR<02,00>=#1 (LOOP BACK MODE)
19336 ;. EXECUTE "RESET"
19337 ;. IF XCSR<02!00> NE #0 THEN
19338 ;. ERROR
19339 ;. ENDIF
19340 ;. LET XCSR<02>=#0
19341 ;ENDROUTINE
19342
19343 ;*****
19344 115116 000004 TST46: SCOPE
19345 115120 122737 000001 001220 CMPB 0APTENV,$ENV ;RUNNING IN APT MODE?
19346 115126 001003 BNE 1$ ;NO, GO DO TEST
19347 115130 005737 001206 TST $PASS ;FIRST PASS?
19348 115134 001011 BNE TST47 ;IF APT AND NOT FIRST PASS, EXIT TEST
19349 115136 052737 000005 177564 1$: BIS 0BIT02!BIT00,XCSR ;LOOP BACK MODE
19350 ;
19351 ; EXECUTE RESET AND VALIDATE THAT XCSR<7,2> BECOMES <1,0>
19352 ;
19353 115144 000005 RESET ;EXECUTE RESET
19354 115146 032737 000005 177564 BIT 0BIT02!BIT00,XCSR ;XCSR<2,0> CLEAR?
19355 115154 001401 BEQ TST47 ;IF YES, BRANCH
19356 115156 104102 ERROR #102 ;XCSR<2,0> NOT CLEARED ON RESET
19357
19358

```

COKDABO R0J11 B CLUSTER MACY11 30(1046)
COKDAB.P11 05-APR-84 16:45

05 APR 84 16:45 PAGE 369
TEST RESET AND INTERRUPT ENABLE BITS

SEQ 0368

```

19359 .SBTTL TEST - RESET AND INTERRUPT ENABLE BITS
19360 ;CHECK THAT INTERRUPTS DON'T HAPPEN AT PRIORITY 4 AND THAT RESET
19361 ;CLEARS XCSR<06> AND RCSR<06>.
19362 ;
19363 ;RCSR <06> RECEIVER INTERRUPT ENABLE
19364 ;XCSR <06> TRANSMITTER INTERRUPT ENABLE
19365 ;
19366 ;ROUTINE TEST
19367 ;. LET 60=ADDRESS OF ILLEGAL INTERRUPT XCSR
19368 ;. LET 64=ADDRESS OF ILLEGAL INTERRUPT XCSR
19369 ;. SET PRIORITY TO 4
19370 ;. LET XCSR<02>=01 (LOOPBACK MODE)
19371 ;. LET XCSR<06>=01 (ENABLE TRANSMIT INTERRUPTS)
19372 ;. LET RCSR<06>=01 (ENABLE RECEIVE INTERRUPTS)
19373 ;. WAIT FOR XCSR<07>=01 (READY TO TRANSMIT)
19374 ;. LET XBUF=0 (SEND A CHARACTER)
19375 ;. WAIT FOR ILLEGAL INTERRUPTS (ABOUT 200MSEC)
19376 ;. EXECUTE "RESET"
19377 ;. IF XCSR<06> NE 00 OR RCSR<06> NE 00 OR XCSR NE 00 THEN
19378 ;. ERROR
19379 ;. ENDIF
19380 ;. RESTORE PRIORITY TO NORMAL
19381 ;ENDROUTINE
19382 ;
19383 ;ROUTINE ILLEGAL_INTERRUPT_XCSR
19384 ;. INCREMENT XCSR
19385 ;ENDROUTINE
19386 ;
19387 ;*****
19388 115160 000004 TST47: SCOPE
19389 115162 122737 000001 001220 CMPB 0APTENV,$ENV ;RUNNING IN APT MODE?
19390 115170 001003 BNE 1$ ;NO, GO DO TEST
19391 115172 005737 001206 TST $PASS ;FIRST PASS?
19392 115176 001033 BNE 5$ ;SKIP RESET PART OF THE TEST
19393 ;
19394 ; CHECK THAT INTERRUPTS ENABLE BITS FOR RECEIVER AND TRASMITTER OF SLU
19395 ; ARE CLEARED BY RESET
19396 ;
19397 115200 052737 000100 177564 1$: BIS 0BIT06,XCSR ;SET INTERRUPT ENABLE BIT IN XCSR
19398 115206 032737 000100 177564 BIT 0BIT06,XCSR ;GOT SET OK?
19399 115214 001001 BNE 2$ ;IF YES, BRANCH
19400 115216 104110 ERROR +110 ;IN BIT 6 OF XCSR
19401 115220 052737 000100 177560 2$: BIS 0BIT06,RCSR ;SET INTERRUPT ENABLE BIT IN RCSR
19402 115226 032737 000100 177560 BIT 0BIT06,RCSR ;GOT SET OK?
19403 115234 001001 BNE 3$ ;IF YES, BRANCH
19404 115236 104110 ERROR +110 ;IN BIT 6 OF RCSR
19405 115240 000005 3$: RESET ;INLINE BUS RESET
19406 115242 032737 000100 177564 BIT 0BIT06,XCSR ;XMIT INTERRUPT ENABLE BIT CLEARED?
19407 115250 001401 BEQ 4$ ;IF CLEARED, BRANCH
19408 115252 104102 ERROR +102 ;INTERRUPT ENABLE NOT CLEARED ON RESET
19409 115254 032737 000100 177560 4$: BIT 0BIT06,RCSR ;RECEIVE INTERRUPT ENBLE CLEARED?
19410 115262 001401 BEQ 5$ ;IF CLEARED, BRANCH
19411 115264 104102 ERROR +102 ;INTERRUPT ENABLE NOT CLEARED ON RESET
19412 ;
19413 ; CHECK THAT TRANSMIT INTERRUPTS DON'T HAPPEN AT PRIORITY HIGHER THAN 3
19414 ;

```

COKDABO RDJ11 B CLUSTER MACY11 30(1046) 05 APR-84 16:45 PAGE 370
COKDAB,P11 05-APR 84 16:45 TEST - RESET AND INTERRUPT ENABLE BITS

SEQ 0369

Address	Offset	Hex	Label	Instruction	Comment
19415	115266	012737	115334	000064	5\$: MOV #9\$, @#64 ;POINT XMIT VECTOR TO PROGRAM AREA
19416	115274	012737	000340	000066	MOV #340, @#66 ;AT PRIORITY 7
19417	115302	052737	000100	177564	BIS #BIT06, XCSR ;SET INTERRUPT ENABLE BIT IN XCSR
19418	115310	012702	000340		MOV #340, R2 ;SET PRIORITY TO 7
19419	115314	000402			BR 7\$;GO WAIT IN CASE OF INTERRUPTS
19420	115316	162702	000040	6\$: SUB #40, R2 ;LOWER PRIORITY LEVEL	
19421	115322	106402		7\$: MTPS R2 ;SET PRIORITY	
19422	115324	012703	000026		MOV #26, R3 ;TIME DELAY
19423	115330	077301		8\$: SOB R3, 8\$;WAIT FOR INTERRUPTS	
19424	115332	000403			BR 10\$;IF INTERRUPTS DIDN'T HAPPENED, BRANCH
19425	115334	104101		9\$: ERROR +101 ;INTERRUPTS HAPPEN AT WRONG PRIORITY	
19426	115336	005726			TST (SP)+ ;CLEAN UP THE STACK
19427	115340	005726			TST (SP)+
19428	115342	022702	000200	10\$: CMP #200, R2 ;AT PRIORITY 4?	
19429	115346	001363			BNE 6\$;IF NOT LAST ONE, CONTINUE
19430	115350	106427	000340		MTPS #340 ;RESTORE PRIORITY 7
19431	115354	042737	000100	177564	BIC #BIT06, XCSR ;CLEAR INTERRUPT ENABLE BIT
19432					;
19433					; CHECK THAT RECEIVE INTERRUPTS DON'T HAPPEN AT PRIORITY HIGHER THAN 3
19434					;
19435	115362	012737	115452	000060	MOV #15\$, @#60 ;POINT RECEIVE VECTOR TO PROGRAM AREA
19436	115370	012737	000340	000062	MOV #340, @#62 ;AT PRIORITY 7
19437	115376	105737	177564		11\$: TSTB XCSR ;TRANSMITTER READY
19438	115402	100375			BPL 11\$;IF NOT, WAIT
19439	115404	012737	000000	177566	MOV #NULL, XBUF ;TRY TO TRANSMIT NULL
19440	115412	052737	000100	177560	BIS #BIT06, RCSR ;SET INTERRUPT ENABLE BIT IN RCSR
19441	115420	012702	000340		MOV #340, R2 ;SET PRIORITY TO 7
19442	115424	000402			BR 13\$;GO WAIT IN CASE OF INTERRUPTS
19443	115426	162702	000040		12\$: SUB #40, R2 ;LOWER PRIORITY LEVEL
19444	115432	052737	000004	177564	13\$: BIS #BIT02, XCSR ;SET LOOP BACK MODE
19445	115440	106402			MTPS R2 ;SET PRIORITY
19446	115442	012703	000100		MOV #100, R3 ;TIME DELAY
19447	115446	077301		14\$: SOB R3, 14\$;WAIT FOR INTERRUPTS	
19448	115450	000406			BR 16\$;IF INTERRUPTS DIDN'T HAPPENED, BRANCH
19449	115452	042737	000004	177564	15\$: BIC #BIT02, XCSR ;CLEAR LOOP BACK MODE
19450	115460	104101			ERROR +101 ;INTERRUPTS HAPPEN AT WRONG PRIORITY
19451	115462	005726			TST (SP)+ ;CLEAN UP THE STACK
19452	115464	005726			TST (SP)+
19453	115466	022702	000200		16\$: CMP #200, R2 ;AT PRIORITY 4?
19454	115472	001355			BNE 12\$;IF NOT LAST ONE, CONTINUE
19455					;
19456					; CLEAN UP BEFORE NEXT TEST
19457					;
19458	115474	106427	000340		MTPS #340 ;RESTORE PRIORITY 7
19459	115500	042737	000100	177560	BIC #BIT06, RCSR ;CLEAR INTERRUPT ENABLE BIT
19460	115506	012702	000300		MOV #400, R2 ;STALL DELAY
19461	115512	105737	177560		17\$: TSTB RCSR ;RECEIVE READY?
19462	115516	100401			BMI 18\$;STOP WAITING, IF SO
19463	115520	077204			SOB R2, 17\$;OTHERWISE, STAY IN THE LOOP
19464	115522	005737	177562		18\$: TST RBUF ;READ CHARACTER TRANSMITTED
19465	115526	042737	000004	177564	BIC #BIT02, XCSR ;CLEAR LOOP BACK MODE
19466					

H13

COKDABO RDJ11 B CLUSTER MACY11 50(1046) 05-APR-84 16:45 PAGE 371
 COKDAB.P11 05-APR-84 16:45 TEST INTERRUPT PRIORITY FOR SLU

SEQ 0370

```

19467 .SBTTL TEST - INTERRUPT PRIORITY FOR SLU
19468 ;CHECK THAT INTERRUPTS HAPPEN AT PRIORITY 3 AND THAT THEY CLEAR
19469 ;RCSR<06> AND XCSR<06>.
19470 ;
19471 ;ROUTINE TEST
19472 ;. LET 60=#ADDRESS_OF_LEGAL_RINTERRUPT
19473 ;. LET 64=#ADDRESS_OF_LEGAL_XINTERRUPT
19474 ;. LET XCSR<02>=#1
19475 ;. SET PRIORITY TO #3
19476 ;. WAIT FOR XINTERRUPT=#3
19477 ;. IF XCSR<07> EQ #1 THEN
19478 ;. ERROR
19479 ;. ENDF
19480 ;. WAIT FOR RINTERRUPT=#3
19481 ;. IF RCSR<07> EQ #0 THEN
19482 ;. ERROR
19483 ;. ENDF
19484 ;. LET XCSR<02>=#0
19485 ;. SET PRIORITY TO NORMAL
19486 ;ENDROUTINE
19487 ;
19488 ;ROUTINE LEGAL_XINTERRUPT
19489 ;. LET XBUF=#CHARACTER
19490 ;. INCREMENT XINTERRUPT
19491 ;ENDROUTINE
19492 ;
19493 ;ROUTINE LEGAL_RINTERRUPT
19494 ;. READ RCSR
19495 ;. INCREMENT RINTERRUPT
19496 ;ENDROUTINE
19497 ;
19498 ;*****
19499 115534 000004 TST50: SCOPE
19500 115536 122737 000001 001220 CMPB #APTENV,$ENV ;RUNNING IN APT MODE?
19501 115544 001003 BNE 100$ ;NO, GO DO TEST
19502 115546 005737 001206 TST $PASS ;FIRST PASS?
19503 115552 001113 BNE TST51 ;IF APT AND NOT FIRST PASS, EXIT TEST
19504 ;
19505 ; GET READY FOR INTERRUPTS
19506 ;
19507 115554 012737 115750 000060 100$: MOV #8,$@#60 ;STORE RECEIVER VECTOR
19508 115562 012737 115674 000064 MOV #6,$@#64 ;STORE TRANSMITTER VECTOR
19509 115570 012737 000340 000062 MOV #340,$@#62 ;AT PRIORITY 7
19510 115576 012737 000340 000066 MOV #340,$@#66 ;FOR RECEIVER AND TRANSMITTER
19511 115604 052737 000004 177564 BIS #BIT02,XCSR ;SET LOOP BACK MODE
19512 115612 012701 000100 MOV #100,R1 ;DELAY FOR UNEXPECTED CHARACTERS
19513 115616 105737 177560 1$: TSTR RCSR ;RECEIVER READY?
19514 115622 100401 BML 1$ ;IF YES, BRANCH
19515 115624 077104 SOB R1,1$ ;OTHERWISE, WAIT JUST IN CASE
19516 115626 005737 177562 2$: TST RBUF ;READ RECEIVER
19517 ;
19518 ; SET PRIORITIES AND XMIT INTERRUPTS
19519 ;
19520 115632 012702 000140 MOV #140,R2 ;START WITH PRIORITY 3
19521 115636 000402 BR 4$ ;TRY TO DO IT
19522 115640 162702 000040 3$: SOB #40,R2 ;LOWER PRIORITY

```

COKDABO KDJ11-B CLUSTER MAC11 30(1046) 05 APR-84 16:45 PAGE 372
 COKDAB.P11 05-APR-84 16:45 TEST - INTERRUPT PRIORITY FOR SLU

SEQ 0371

```

19523 115644 106402      4$:  MTPS    R2                ;TRY TO DO AT LOWER PRIORITY
19524 115646 052737 000100 177564  BIS    #BIT06,XCSR      ;LOOP BACK & INTERRUPT ENABLE
19525 115654 012703 000100      MOV    #100,R3          ;WAIT DELAY FOR INTERRUPTS
19526 115660 077301      5$:  SOB     R3,5$          ;WAIT FOR XMIT INTERRUPTS
19527 115662 042737 000004 177564  BIC    #BIT02,XCSR      ;CLEAR LOOP BACK BIT
19528 115670 104107      ERROR   +107          ;NO XMIT INTERRUPTS
19529 115672 000443      BR      TST51          ;;IF ERROR, EXIT TEST
19530
19531      ; TRANSMITTER INTERRUPT HERE
19532
19533 115674 005726      6$:  TST     (SP)+          ;CLEAN UP STACK
19534 115676 005726      TST     (SP)+
19535 115700 042737 000100 177564  BIC    #BIT06,XCSR      ;CLEAR INTERRUPT ENABLE
19536 115706 012737 000000 177566  MOV    #NULL,XBUF      ;TRANSMIT NULL
19537 115714 052737 000100 177560  BIS    #BIT06,RCSR      ;SET RECEIVE INTERRUPT
19538 115722 106402      MTPS    R2                ;SET NEXT PRIORITY
19539 115724 012703 020000      MOV    #20000,R3        ;WAIT DELAY FOR INTERRUPTS
19540 115730 077301      7$:  SOB     R3,7$          ;WAIT FOR RECEIVE INTERRUPTS
19541 115732 042737 000004 177564  BIC    #BIT02,XCSR      ;CLEAR LOOP BACK MODE BIT
19542 115740 104107      ERROR   +107          ;NO RECEIVE INTERRUPTS
19543 115742 000406      BR      9$                ;DON'T TOUCH STACK
19544 115744 106427 000340      MTPS    #340          ;RAISE PRIORITY
19545
19546      ; RECEIVER INTERRUPT HERE
19547
19548 115750 005726      8$:  TST     (SP)+          ;CLEAN UP STACK
19549 115752 005726      TST     (SP)+
19550 115754 005737 177562      TST    RBUF          ;READ RECEIVER BUFFER
19551 115760 005702      9$:  TST     R2                ;PRIORITY 0
19552 115762 001326      BNE     3$                ;IF NOT YET, CONTINUE
19553 115764 106427 000340      MTPS    #340          ;RAISE PRIORITY TO 7
19554 115770 042737 000100 177560  BIC    #BIT06,RCSR      ;CLEAR RECEIVE INTER. ENABLE
19555 115776 005037 177564      CLR     XCSR          ;CLEAR XCSR
19556

```


COKDABO KDJ11 B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 375
 COKDAB.P11 05-APR-84 16:45 TEST - BREAK CONDITION

SEQ 0372

```

19557 .SBTTL TEST - BREAK CONDITION
19558 ;CHECK THAT SENDING BREAK CAUSES FRAMING ERROR.
19559 ;
19560 ;RCSR <15> ERROR
19561 ; <13> FRAMING ERROR
19562 ; <11> RECEIVED BREAK
19563 ;
19564 ;XCSR <00> TRANSMIT BREAK
19565 ;
19566 ;ROUTINE TEST
19567 ;. LET XCSR<02>=#1
19568 ;. LET XCSR<00>=#1
19569 ;. WAIT FOR RCSR<07>=#1
19570 ;. IF RBUF<15!13!11> NE #1 THEN
19571 ;. ERROR (ERROR, FRAMING ERROR, RECEIVE BREAK NE 1)
19572 ;. ENDIF
19573 ;. LET XCSR<00>=#0
19574 ;. IF XCSR<00> NE #0 THEN
19575 ;. ERROR (XCSR<00> DOES NOT GO LOW)
19576 ;. ENDIF
19577 ;. WAIT FOR XCSR<07>=#1
19578 ;. LET XBUF=#NULL (SEND NULL CHARACTER TO SEE ERROR CLEARED)
19579 ;. WAIT FOR RCSR<07>=#1
19580 ;. IF RBUF<15!13!11> NE #0 THEN
19581 ;. ERROR
19582 ;. ENDIF
19583 ;. LET XCSR<00>=#1
19584 ;. EXECUTE "RESET"
19585 ;. IF XCSR<00> NE #0 THEN
19586 ;. ERROR
19587 ;. ENDIF
19588 ;. LET XCSR<02>=#0
19589 ;ENDROUTINE
19590
19591 ;*****
19592 116002 000004 TST51: SCOPE
19593 ;
19594 ; DECIDE WHETHER TO RUN THIS TEST
19595 ;
19596 116004 032737 000200 000052 BIT #B1107,#052 ;UFD MODE?
19597 116012 001127 BNE TST52 ;;IN UFD MODE, EXIT TEST
19598 116014 005737 001206 TST $PASS ;FIRST PASS?
19599 116020 001124 BNE TST52 ;;IF APT AND NOT FIRST PASS, EXIT TEST
19600 ;
19601 ; SEND BREAK AND CHECK ERROR BITS IN RBUF
19602 ;
19603 116022 052737 000004 177564 1$: BIS #B1102,XCSR ;TRANSMIT IN LOOP BACK
19604 116030 013737 177520 002710 MOV BCSR,SAVBR ;SAVE BCSR
19605 116036 042737 001000 177520 RLC #B1109,BCSR ;DISABLE HALT ON BREAK
19606 116044 052737 000001 177564 BIS #B1100,XCSR ;SET SEND BREAK BIT
19607 116052 032737 000001 177564 BIT #B1100,XCSR ;GOT SET OK?
19608 116060 001001 BNE 2$ ;IF YES, BRANCH
19609 116062 104110 ERROR +110 ;WRITING 1 TO XCSR<0>
19610 116064 012701 000100 1$: MOV #100,R1 ;STALL DELAY
19611 116070 105737 177560 4$: TSTB RCSR ;RECEIVER READY?
19612 116074 100401 BML 5$ ;IF YES, BRANCH

```


CORDBAO KDJ11 H CLUSTER MACY11 30(1046) 05 APR-84 16:45 PAGE 375
 CORDAB.P11 05-APR-84 16:45 TEST - OVERRUN CONDITION

SEQ 0374

```

19649 .SBTTL TEST - OVERRUN CONDITION
19650 ;CHECK OVERRUN CONDITION
19651 ;
19652 ;RCSR <14> OVERRUN ERROR
19653 ;
19654 ;ROUTINE TEST
19655 ;. LET XCSR<02>=#1 (LOOPBACK MODE)
19656 ;. WAIT FOR XCSR<07>=#1
19657 ;. LET XBUF=#252
19658 ;. WAIT FOR XCSR<07>=#1
19659 ;. LET XBUF=#125 (SEND THE 2ND W/O READING THE 1ST CHARACTER)
19660 ;. WAIT FOR RCSR<07>=#1
19661 ;. STALL FOR LOWEST BAUD RATE TO GET 2ND CHARACTER
19662 ;. IF LOW BYTE OF RBUF NE #125 THEN
19663 ;. ERROR (1ST CHARACTER WASN'T OVERRUN)
19664 ;.
19665 ;. ENDIF
19666 ;. IF RBUF<15:14> NE #1 THEN
19667 ;. ERROR (NO OVERRUN BIT SET)
19668 ;.
19669 ;. ENDIF
19670 ;. WAIT FOR XCSR<07>=#1
19671 ;. LET XBUF=NULL
19672 ;. WAIT FOR RCSR<07>=#1
19673 ;. IF RBUF<15:14> NE #0 THEN
19674 ;. ERROR (WASN'T CLEARED ON THE NEXT CHARACTER RECEIVED)
19675 ;.
19676 ;. ENDIF
19677 ;. LET XCSR<02>=#0
19678 ;ENDROUTINE
19679 ;*****
19680 TST52: SCOPE
19681 CMPB #APTEENV,$ENV ;RUNNING IN APT MODE?
19682 BNE 100$ ;NO, GO DO TEST
19683 TST $PASS ;FIRST PASS?
19684 BNE TST53 ;IF APT AND NOT FIRST PASS, EXIT TEST
19685 BIS #BIT02,XCSR ;SET LOOP BACK MODE
19686 1$: TSTB XCSR ;READY TO TRANSMIT?
19687 BPL 1$ ;IF NOT, WAIT
19688 MOV #21,XBUF ;TRANSMIT A CHARACTER
19689 TSTB RCSR ;RECEIVE READY?
19690 BPL 2$ ;IF NOT, WAIT
19691 TSTB XCSR ;READY TO TRANSMIT?
19692 BPL 3$ ;IF NOT, WAIT
19693 MOV #177,XBUF ;TRANSMIT THE 2ND CHARACTER
19694 MOV #10000,R3 ;STALL FOR THE 2ND CHARACTER
19695 2$: TSTB R3,4$ ;WAIT A WHILE
19696 MOV #140177,$BDDAT ;STORE RECEIVED DATA
19697 CMPB #177,$BDDAT ;EXPECTED PATTERN
19698 BEQ 5$ ;2ND CHARACTER RECEIVED?
19699 BIC #BIT102,XCSR ;IF YES, BRANCH
19700 ERROR +111 ;RESET TO ENABLE SLO
19701 3$: CMPB #BIT7:BIT6,$BDDAT+1 ;2ND CHARACTER DIDN'T OVERRUN 1ST
19702 BEQ 6$ ;OVERRUN ERROR BITS SET?
19703 CLR XCSR ;IF YES, BRANCH
19704 ERROR +112 ;RESET TO ENABLE SLO
19705 BR TST53 ;OVERRUN DOES NOT SET ERROR BITS
19706 ;EXIT

```

M13

COKDABO KDJ11 B CLUSTER MACY11 30(1046) 05 APR-84 16:45 PAGE 376
COKDAB.P11 05-APR-84 16:45 TEST - OVERRUN CONDITION

SEQ 0375

19705					:			
19706					:	; SEND NEXT CHARACTER TO CLEAR OVERRUN CONDITIONS		
19707					:			
19708	116440	105737	177564		6\$:	ISTB	XCSR	; TRANSMITTER READY?
19709	116444	100375				BPL	6\$	; IF NOT, BRANCH AND WAIT
19710	116446	012737	000000	177566		MOV	#NULL,XBUF	; TRANSMIT NULL CHARACTER
19711	116454	105737	177560		7\$:	ISTB	RCSR	; RECEIVER READY?
19712	116460	100375				BPL	7\$	; IF NOT, BRANCH AND WAIT
19713	116462	032737	140000	177562		BIT	#BIT15!BIT14,RBUF	; ANY ERRORS SET?
19714	116470	001404				BEQ	8\$	; IF NOT, BRANCH
19715	116472	042737	000004	177564		BIC	#BIT02,XCSR	; RESET TO ENABLE SLU
19716	116500	104113				ERROR	.113	; OVERRUN NOT CLEARED ON NEXT CHAR.
19717	116502	042737	000004	177564	8\$:	BIC	#BIT02,XCSR	; CLEAR LOOP BACK MODE BIT
19718								
19719	116510				SLEND:			; LAST SLU TEST

N13

COKDABO KDJ11 B CLUSTER MACY11 30(1046) 05 APR-84 16:45 PAGE 377
COKDAB.P11 05-APR-84 16:45 TEST - LED'S ON

SEQ 0376

```

19720 ;SMTL TEST - LED'S ON
19721 ;LED'S ON
19722 ;THIS TEST WILL INITIALIZE BDR TO CONTAIN A ROTATING PATTERN
19723 ;DISPLAYED IN LED'S.
19724 ;
19725 ;ROUTINE TEST
19726 ;. WHILE A KEY NOT RECEIVED FROM KEYBOARD DO
19727 ;. . STALL ALLOWING TIME TO SEE PATTERN
19728 ;. . ROTATE LEFT TO LIGHT UP NEXT LED'S
19729 ;. ENDDO
19730 ;ENDROUTINE
19731
19732 ;*****
19733 116510 000004 TST53: SCOPE
19734 116512 005005 CLR R5 ;FLAG IN NO INTERRUPT MODE
19735 116514 032737 000001 000052 BIT 0BIT00,0052 ;IF RUNNING IN CHAIN MODE
19736 116522 001427 BEQ 1$ ;SKIP PRINTOUTS
19737 116524 005737 001206 IST $PASS ;1ST PASS?
19738 116530 001024 BNE 1$ ;IF NOT, SKIP PRINTOUTS
19739 116532 122737 000001 001220 CMPB 0APTENV,$ENV ;APT MODE?
19740 116540 001420 BEQ 1$ ;YES, SKIP PRINTOUTS
19741 116542 005105 COM R5 ;CLEAR FLAG IN INTERRUPT MODE
19742 116544 104401 001175 TYPE , $CRLF
19743 116550 104401 116672 TYPE ,LEDS ;IDENTIFY THE TEST
19744 116554 012737 116652 MOV 05$,0050 ;RECEIVE SLU VECTOR
19745 116562 012737 000340 000062 MOV 0340,0062 ;AT PRIORITY 7
19746 116570 052737 000100 177560 BIS 0BIT06,RCSR ;ENABLE INTERRUPTS
19747 116576 106427 000140 MTPS 0140 ;LOWER PRIORITY
19748 116602 012704 000006 1$: MOV 06,R4 ;FOR EACH LOOP
19749 116606 012701 000076 MOV 076,R1 ;START WITH 1
19750 116612 110137 177524 2$: MOVB R1,BDR ;TURN OFF FIRST LED
19751 116616 012703 000004 MOV 04,R3 ;STALL DELAY
19752 116622 012702 177777 3$: MOV 017777,R2 ;STALL DELAY
19753 116626 077201 4$: SOB R2,4$ ;WAIT A WHILE
19754 116630 077304 SOB R3,3$ ;WAIT A WHILE
19755 116632 000261 SEC ;SET CARRY
19756 116634 006101 ROL R1 ;GET ANOTHER LED
19757 116636 077413 SOB R4,2$ ;DO A FEW TIMES
19758 116640 005705 IST R5 ;RUNNING IN INTERACTIVE MODE?
19759 116642 001407 BEQ 6$ ;IF NOT, EXIT
19760 116644 104401 001170 TYPE , $BELL. ;
19761 116650 000754 BR 1$ ;REPEAT PATTERN
19762 116652 005737 177562 5$: IST RBUF ;READ BUFFER
19763 116656 062706 000004 ADD 04,$P ;ADJUST STACK
19764 116662 112737 000377 177524 6$: MOVB 0377,BDR ;NO MORE
19765 116670 000452 BR TST54 ;EXIT TEST
19766
19767 116672 006412 044124 051511 LEDS: .ASCII <12><15>/THIS IS A TEST FOR ON-BOARD LED'S/<12><15>
19768 116700 044440 020123 020101
19769 116706 042524 052123 043040
19770 116714 051117 047440 026516
19771 116722 047502 051101 020104
19772 116730 042514 023504 005123
19773 116736 015
19774 116737 124 050131 020105 .ASCII /TYPE ANY CHARACTER ON A KEYBOARD TO CONTINUE/<12><15>
19775 116744 047101 020131 044103

```

B14

CORDBO RDJ11 H CLUSTER MAC111 50(1046) 05 APR 84 16:45 PAGE 378
 CORDAB.P11 05 APR 84 16:45 TEST LED'S ON

SEQ 0377

19776	116752	051101	041501	042524
19777	116760	020122	047117	040440
19778	116766	045440	054505	047502
19779	116774	051101	020104	047524
19780	117002	041440	047117	044524
19781	117010	052516	005105	000015
19782				
19783				

.EVEN

014

19784
19785
19786
19787
19788
19789
19790
19791
19792
19793
19794
19795
19796
19797
19798
19799
19800
19801
19802
19803
19804
19805
19806
19807
19808
19809
19810
19811
19812
19813
19814
19815
19816
19817
19818
19819
19820
19821
19822
19823
19824
19825 117016 000004
19826 117020 052737 000001 000052
19827 117026 001561
19828 117030 005737 001206
19829 117034 001156
19830 117036 122737 000001 001220
19831 117044 001552
19832
19833
19834
19835 117046 012737 000400 122746
19836 117054 013704 000004
19837 117060 012737 117230 000004
19838 117062 012737 000340 000006
19839 117074 004737 132156

```

;TITLE TEST MEMORY MAPPING
;MEMORY MAPPING
;THIS TEST WILL AUTOSIZE MEMORY IN PK BYTES. EVERY PAGE WILL BE
;CHECKED FOR WHAT TYPE IT IS: Q-BUS, UNIBUS OR PMT BUS MEMORY BY
;SEEING HOW IT CAN BE CACHED.
;ROUTINE TEST
1. LET A=ADDRESS OF NON EXISTENT MEMORY
1. IF KMCR<05 00> NE 01 THEN (NOT ALL UNIBUS MEMORY)
1. TURN ON MMU AND REMAP THE PROGRAM AREA
1. REPEAT
1. DO FOR KDPARO FROM 00 TO 0177600
1. DO FOR R1 FROM 00 TO 020000 BY 04000
1. READ (R1)
1. IF ABORT NON EXISTENT MEMORY NE 1
1. MEMORY EXISTS
1. READ (R1)
1. READ (R1)
1. IF HIT/MISS EQ 2 HITS THEN
1. PMT MEMORY
1. ELSE
1. IF HIT/MISS EQ HIT THEN
1. Q BUS MEMORY
1. ELSE
1. ERROR
1. ENDDIF
1. ENDDIF
1. ENDDIF
1. LET ABORT NON EXISTENT MEMORY=00
1. ENDDO
1. ENDDO
1. UNTILL ABORT NON-EXISTANT MEMORY EQ 01
1. ENDDIF
;ENDROUTINE
;ROUTINE NON EXISTENT MEMORY
1. LET ABORT NON EXISTENT MEMORY=01
1. RETURN
;ENDROUTINE

*****
TEST54: SCOPF
BIT 0BIT00,0052 ;CHAIN MODE?
REQ 15155 ;IF NO, EXIT TEST
TST $PASS ;FIRST PASS?
BNE 15155 ;IF APT AND NOT FIRST PASS, EXIT TEST
CMPB 0APTENV,0ENV ;APT MODE?
REQ 15155 ;YES, SKIP PRINTOUTS

; SETUP ALL REGISTERS FOR MAPPING
1. MOV 0400,CCR ;FLUSH THE CACHE
1. MOV 0ERRVEC,R4 ;STORE TIMEOUT VECTOR
1. MOV 073,00ERRVEC ;POINT TO PROGRAM AREA
1. MOV 0540,0ERRVEC+2 ;AT PRIORITY 4
1. JSR PC,INITMM ;REMAP PROGRAM AREA

```

CORDARO RDJ11 8 CLUSTER MACY11 50(1046) 05 APR 84 16:45 PAGE 580
 CORDAR,P11 05 APR 84 16:45 TEST MEMORY MAPPING

SEQ 0379

```

19840 117100 005037 172354 CLR KIPAR6 ;USED FOR MAPPING MEMORY
19841 117104 005002 CLR R2 ;CLEAR PAGE COUNT FOR PMI
19842 117106 005003 CLR R3 ;CLEAR PAGE COUNT FOR QBUS
19843 117110 005237 177572 INC MMRO ;ENABLE MEMORY MANGEMENT
19844 117114 052737 000020 172516 BIS #BIT04,MMR3 ;ENABLE 22 BITS
19845 117122 000403 BR 24 ;GO ACCESS
19846 ;
19847 ; TRY TO MAP ALL PAGES
19848 ;
19849 117124 062737 000200 172354 14: ADD #200,KIPAR6 ;INCREMENT BY 4K WORDS
19850 117132 012701 140000 24: MOV #140000,R1 ;ACCESS THRU KIPAR6
19851 117136 000402 BR 44 ;START DOING IT
19852 117140 062701 003776 54: ADD #3776,R1 ;INCREMENT BY 1K WORDS
19853 117144 005721 44: TST (R1) ;ACCESS 1ST LOCATION
19854 117148 042737 001000 177520 BIC #1000,BCSR ;DISABLE HALT ON BREAK
19855 117154 005711 TST (R1) ;ACCESS 2ND LOCATION
19856 117156 013737 177752 110504 MOV HITMIS,RECDAT ;STORE REGISTER
19857 117164 052737 001000 177520 BIS #1000,BCSR ;ENABLE HALT ON BREAK
19858 117172 032737 000004 110504 BIT #BIT02,RECDAT ;LAST (R1) HIT?
19859 117200 001402 BEQ 54 ;IF NOT, QBUS MEMORY
19860 117202 005202 INC R2 ;INCREMENT 1K COUNT FOR PMI
19861 117204 000401 BR 64 ;GO CONTINUE
19862 117206 005203 54: INC R3 ;INCREMENT COUNT FOR QBUS MEM.
19863 117210 022701 154200 64: CMP #154000,R1 ;LAST IN 4K PAGE?
19864 117214 101351 BHT 34 ;IF NOT, BRANCH
19865 117216 022737 177600 172354 CMP #177600,KIPAR6 ;PM BOUNDARY?
19866 117224 001357 BNE 14 ;IF NOT, BRANCH
19867 117226 000402 BR 84 ;IF PM, DON'T TOUCH STACK
19868 ;
19869 ; MAPPING IS DONE, FIND OUT HOW MANY PAGES WERE THERE
19870 ;
19871 117230 005726 74: TST (SP) ;ADJUST STACK
19872 117232 005726 TST (SP)
19873 117234 010437 000004 84: MOV R4,ERRVEC ;RESTORE ERROR VECTOR
19874 117240 005037 177572 CLR MMRO ;DISABLE MEMORY MANGEMENT
19875 117244 042737 000020 172516 BIC #BIT04,MMR3 ;DISABLE 22 BITS
19876 117252 005702 TST R2 ;ANY PMI MEMORY?
19877 117254 001405 BEQ 94 ;IF NOT, GO CHECK QBUS MEMORY
19878 117256 006302 ASL R2 ;TRANSLATE TO BYTES
19879 117260 010246 MOV R2,(SP) ;STORE 1K # ON STACK
19880 117262 104405 TYPDS ;TYPE # OF PAGES
19881 117264 104401 117310 TYPE ,MEMK ;TYPE ASCII
19882 117270 005703 94: TST R3 ;ANY Q BUS MEMORY?
19883 117272 001405 BEQ 104 ;IF NOT, BRANCH
19884 117274 006303 ASL R3 ;TRANSLATE TO BYTE
19885 117276 010346 MOV R3,(SP) ;STORE 1K # ON STACK
19886 117300 104405 TYPDS ;TYPE # OF PAGES
19887 117302 104401 117340 TYPE ,MEMQ ;TYPE ASCII
19888 117306 104:
19889 117306 000431 BR TEST ;IF IT TEST
19890 ;
19891 117310 020113 054502 042524 MEMK: .ASCII 7K BYTES OF PMI MEMORY (1-15)
19892 117316 020123 043117 050040
19893 117324 044515 046440 046505
19894 117332 051117 005131 000015
19895 117340 020113 054502 042524 MEMQ: .ASCII 7K BYTES OF Q-BUS MEMORY (1-15)

```


CORDARO RD,11 B CLUSTER MACY11 30(1046) 05 APR-84 16:45 PAGE 381
 CORDAB,P11 05-APR-84 16:45 TEST MEMORY MAPPING

SEQ 0380

19896 117346 020125 043117 050440
 19897 117354 041055 051525 046440
 19898 117362 046505 051117 005131
 19899 117370 000015

,EVEN

,UNTIL WRONG PARITY ABORT TEST

,WRONG PARITY ABORT

,THIS TEST VERIFIES ABORT TO 114 USING PARITY OR ECC MEMORY CSR.

,IF MORE THEN 1 CSR PRESENT, ALL OF THEM WILL BE WRITTEN AT THE

,SAME TIME.

;

,ROUTINE TEST

;; SIZE FOR ALL POSSIBLE MEMORY CSR'S

;; STORE UP TO 16 CSR STARTING FROM TEMP

;; WRITE ALL WITH WRONG PARITY

;; WRITE TO 0

;; READ IT BACK

;; IF NO ABORT TO 114 THEN

;; ERROR IN PARITY ABORT LOGIC

;; ENDIF

;; RESTORE CSR'S

;ENDROUTINE

;*****

;TEST5: SCOPE

;

; FIND OUT ALL POSSIBLE MEMORY CSR LOCATIONS UP TO 16

;

MOV ERRVEC,R1

;STORE TIMEOUT VECTOR

MOV 02\$,ERRVEC

;POINT NEW TO PROGRAM

MOV 0340,ERRVEC+2

;AT PRIORITY 2

CLR R4

;COUNT FOR CSR'S

MOV 0172100,R2

;FIRST POSSIBLE CSR

MOV 0TEMP,R3

;STORAGE LOCATION

1\$: TST (R2)

;IS CSR THERE?

MOV R2,(R3)+

;IF THERE, STORE

INC R4

;INCREMENT COUNT FOR CSR'S

BR 3\$

;BRANCH AROUND

2\$: TST (SP)

;RESTORE STACK

TST (SP)

3\$: ADD 0\$,R2

;POINT TO NEW CSR

CMP 0172136,R2

;ALL DONE?

BHI 1\$

;IF NOT, BRANCH

MOV R1,ERRVEC

;RESTORE ERROR VECTOR

BIS 0BIT09,CCR

;SET CACHE BYPASS

;

; WRITE ALL CSR'S WITH WRONG ECC CODE

; NOTE: IN PARITY MEMORY THOSE BITS ARE READ ONLY AND

; DIAGNOSTIC MODE BIT FOR ECC IS THE SAME AS WRONG PARITY

;

MOV 00114,R1

;STORE PARITY ABORT VECTOR

MOV 00\$,00114

;POINT NEW TO PROGRAM

MOV 0340,00116

;AT PRIORITY 2

MOV R4,R2

;STORE CSR COUNT

MOV 0TEMP,R3

;START WITH 1ST CSR

19900

19901

19902

19903

19904

19905

19906

19907

19908

19909

19910

19911

19912

19913

19914

19915

19916

19917

19918

19919

19920

19921

19922

19923

19924

19925

19926

19927

19928

19929

19930

19931

19932

19933

19934

19935

19936

19937

19938

19939

19940

19941

19942

19943

19944

19945

19946

19947

19948

19949

19950

19951

117372 000004

117374 013701 000004

117400 012737 117436 000004

117406 012737 000340 000006

117414 005004

117416 012702 172100

117422 012703 002710

117426 005712

117430 010223

117432 005204

117434 000402

117436 005726

117440 005726

117442 062702 000002

117446 022702 172136

117452 101365

117454 010137 000004

117460 052737 001000 177746

19942

19943

19944

19945

19946

19947

19948

19949

19950

19951

19952

19953

19954

19955

19956

19957

19958

19959

19960

19961

19962

19963

19964

19965

19966

19967

19968

19969

19970

```

;DIAGNOSTIC OR WRONG PARITY
;CLEAR <10 5> CHECK BITS
;DO FOR ALL CSRS
;CLEAR TEST LOCATION WITH WRONG PAR
;RESTORE COUNTER
;START WITH 1ST CSR
;CLEAR ALL WRONG PARITY
;IN ALL CSRS
;READ BACK WRONG PARITY
;NO WRONG PARITY ABORT

;ADJUST STACK

;START WITH 1ST CSR
;CLEAR ALL WRONG PARITY
;IN ALL CSRS
;ABORT REFLECTED IN MSER?
;IF YES, BRANCH
;
;MSER<15> NOT SET ON PARITY ABORT
;CLEAR CACHE BIPASS
;CLEAR WRONG PARITY
;CLEAR MSER
;RESTORE PARITY ABORT

```

COKDABO KDJ11 B CLUSTER MACY11 30(1046)
COKDAB,P11 05-APR-84 16:45

05-APR-84 16:45 PAGE 383
TEST DMA TAG PARITY IN STANDALONE MODE

SEQ 0382

```

19978 ;SBTTL TEST - DMA TAG PARITY IN STANDALONE MODE
19979 ;CHECK DMA TAG STORE PARITY BIT.
19980 ;ROUTINE TEST
19981 ;.    CACHE DMA PARITY
19982 ;.    LET BCSR<08>=01
19983 ;.    REPORT ALL ERRORS
19984 ;ENDROUTINE
19985 ;
19986 ;ROUTINE DMA_PARITY
19987 ;.    GENERATE PARITY ERRORS
19988 ;.    CHECK MSER<13>
19989 ;.    LET BCSR<08>=00
19990 ;ENDROUTINE
19991
19992 ;*****
19993 117640 000004 TST56: SCOPE
19994 ;
19995 ; ALLOCATE CODE IN CACHE
19996 ;
19997 117642 012702 117670      MOV    #DMAPAR,R2      ;POINT TO STANDALONE CODE
19998 117646 005722      TST     (R2),          ;ALLOCATE IN CACHE
19999 117650 022702 117730      CMP     #DPAREN,R2      ;LAST ADDRESS?
20000 117654 001374      BNE     1$              ;IF NOT, BRANCH
20001 117656 005737 002710      TST     TEMP          ;ALLOCATE TEST LOCATION
20002 117662 013737 177520 002710      MOV     BCSR,SAVBR      ;SAVE BCSR
20003 ;
20004 ; IN STANDALONE MODE TRY TO VERIFY RESPONSE TO PARITY ERRORS
20005 ;
20006 117670 052737 000400 177520 DMAPAR: BIS     #BIT08,BCSR      ;SET STANDALONE MODE BIT
20007 117676 052737 002001 177746      BIS     #BIT10:BIT00,CCR      ;WRITE WRONG TAG PARITY
20008 117704 005037 002710      CLR     TEMP          ;WRITE HIT WITH WRONG PARITY
20009 117710 013705 177744      MOV     MSER,R5          ;STORE MSER
20010 117714 042737 002000 177746      BIC     #BIT10,CCR      ;CLEAR WRONG PARITY BIT
20011 117722 042737 000400 177520 2$: BIC     #BIT08,BCSR      ;CLEAR STANDALONE BIT
20012 ;
20013 ; RETURN FROM STANDALONE MODE
20014 ;
20015 117730      DPAREN:
20016 117730 022705 060020      CMP     #0000,R5          ;WRONG DMA PARITY?
20017 117734 001401      BEQ     4$              ;IF OK, BRANCH
20018 117736 104117      ERROR  +117              ;MSER NOT SET IN STANDALONE MODE
20019 117740 005037 177744      CLR     MSER
20020 117744 012737 000400 177746 4$: MOV     #400,CCR          ;FLUSH THE CACHE
20021 117752 013737 002710 177520      MOV     SAVBR,BCSR      ;RESTORE BCSR
20022

```

CORRADO KDJ11 B CLUSTER MACY11 30(1046)
 CORRADO P11 05-APR-84 16:45

05 APR-84 16:45 PAGE 384
 TEST DMA TAG PARITY W/O STANDALONE MODE

SEQ 0383

```

20023 .SBITI TEST - DMA TAG PARITY W/O STANDALONE MODE
20024 ;CHECK DMA TAG PARITY BIT WITHOUT GOING INTO STANDALONE MODE.
20025 ;
20026 ;CCR <10> WRITE WRONG TAG PARITY
20027 ;
20028 ;ROUTINE TEST
20029 ;. LET CCR<10>=01
20030 ;. ALLOCATE LOCATION IN CACHE
20031 ;. INITIATE DMA WRITE TRANSFERS
20032 ;. IF MSER<04> NE 01 THEN
20033 ;. ERROR
20034 ;. ENDIF
20035 ;ENDROUTINE
20036
20037 ;*****
20038 117760 000004 TST57: SCOPE
20039 117762 032737 000200 000052 BIT 0BIT07,0052 ;USED MODE?
20040 117770 001402 BEQ 110$ ;IF NOT, BRANCH
20041 117772 000137 133514 JMP $EOP ;OTHERWISE, NEXT PASS
20042 117776 005737 002644 110$: TST CSR1 ;AT LEAST ONE Q22BE FOUND?
20043 120002 001457 BEQ TST60 ;;IF NOT, EXIT TEST
20044 ;
20045 ; ALLOCATE TEST IN CACHE
20046 ;
20047 120004 012703 120004 11$: MOV 0,R3 ;START WITH CURRENT INSTRUCTION
20048 120010 005723 10$: TST (R3) ;READ A WORD
20049 120012 022703 120122 CMP 04$,R3 ;ALL DONE?
20050 120016 001374 BNE 10$ ;IF NOT, CONTINUE
20051 ;
20052 ; WRITE A WORD WITH WRONG PARITY
20053 ;
20054 120020 013737 000114 001160 1$: MOV 00114,$TMP0 ;STORE PARITY VECTOR
20055 120026 012737 120074 000114 MOV 02$,00114 ;POINT TO TEST AREA
20056 120034 052737 002000 177746 BIS 0BIT10,CCR ;WRITE WRONG TAG PARITY
20057 120042 005037 002710 CLR TEMP ;WRITE MISS WITH WRONG TAG PARITY
20058 120046 042737 001000 177746 BIC 0BIT10,CCR ;CLEAR WRONG TAG PARITY
20059 120054 052737 000200 177746 BIS 0BIT07,CCR ;PARITY ABORT
20060 120062 005000 CLR R0 ;FLAG TO DO 1 TRANSFER
20061 120064 004737 132752 JSR PC,DMA TRN ;DO DMA WRITE TO TEMP THRU Q22BE
20062 120070 104116 ERROR +116 ;NO PARITY ABORT
20063 120072 000413 BR 4$ ;BRANCH TO TEST MSER
20064 120074 013704 177744 2$: MOV MSER,R4 ;STORE REGISTER
20065 120100 005037 177744 CLR MSER ;CLEAR MSER
20066 120104 005726 TST (SP) ;
20067 120106 005726 TST (SP) ;RESTORE STACK
20068 120110 005726 TST (SP) ;
20069 120112 032704 000020 3$: BIT 0BIT04,R4 ;MSER OK?
20070 120116 001001 BNE 4$ ;IF SET, BRANCH
20071 120120 104116 ERROR +116 ;MSER<4> NOT SET
20072 120122 005037 177744 4$: CLR MSER ;CLEAR MSER
20073 120126 012737 000400 177746 MOV 0400,CCR ;FLUSH CACHE
20074 120134 013737 001160 000114 MOV $TMP0,00114 ;RESTORE VECTOR

```

CORDBO RDJ11-B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 385
 CORDAB.P11 05-APR-84 16:45 TEST - DMA WRITE HIT CYCLES

SEQ 0384

```

20075 ;SBIH TEST - DMA WRITE HIT CYCLES
20076 ;CHECK THAT DMA WRITE HITS INVALIDATE CACHE.
20077 ;ROUTINE TEST
20078 ;; ALLOCATE A LOCATION IN CACHE
20079 ;; INITIATE DMA WRITE TO THIS LOCATION
20080 ;; READ THIS LOCATION BACK
20081 ;; IF IT IS CHANGED OR HIT/MISS EQ HIT
20082 ;; ERROR
20083 ;; ENDIF
20084 ;; WRITE TO THE FIRST 16 LOCATION THEIR ADDRESS
20085 ;; DO BLOCK MODE TRANSFER TO THOSE LOCATIONS
20086 ;; START READ WITH THE LAST LOCATION
20087 ;; IF RECORD ANY HITS THEN
20088 ;; ERROR
20089 ;; ENDIF
20090 ;; INITIATE READ DMA TO THE SAME TEST LOCATIONS
20091 ;; READ THEM BACK
20092 ;; IF HIT/MISS NE HIT THEN
20093 ;; ERROR
20094 ;; ENDIF
20095 ;ENDROUTINE
20096
20097 ;*****
20098 120142 000004 TST60; SCOPE
20099 120144 032737 000200 000052 BIT 0BIT07,0052 ;USED MODE?
20100 120152 001402 BEQ 1$ ;IF NOT, BRANCH
20101 120154 000137 133514 JMP $EOP ;OTHERWISE, NEXT PASS
20102 120160 005737 002644 1$; TST CSR1 ;AT LEAST 1 Q2BE?
20103 120164 001002 BNE 2$ ;IF YES, BRANCH
20104 120166 000137 133514 JMP $EOP ;OTHERWISE, NEXT PASS
20105
20106 ; TRY TO DO DMA WRITE AT ALL DIFFERENT PRIORITIES
20107 ;
20108 120172 012702 000340 2$; MOV 0340,R2 ;START WITH 7
20109 120176 000402 BR 4$ ;GO DO IT
20110 120200 162702 000040 3$; SUB 040,R2 ;LOWER PRIORITY
20111 120204 005037 002710 4$; CLR TEMP ;CLEAR TEST LOCATION
20112 120210 005000 CLR R0 ;DO JUST 1 WORD
20113 120212 106402 MTPS R2 ;LOWER PRIORITY
20114 120214 004737 132752 JSR PC,DMAWRN ;DO DATA
20115 120220 042737 001000 177520 BIC 01000,BCSR ;DISABLE HALT ON BREAK
20116 120226 005737 002710 TST TEMP ;STILL CACHED?
20117 120232 013737 177520 110504 MOV HITMISS,RECDAT ;STORE HIT/MISS
20118 120240 052737 001000 177520 BIS 01000,BCSR ;ENABLE HALT ON BREAK
20119 120246 032737 000004 17504 BIT 0BIT02,RECDAT ;LAST ACCESS HIT?
20120 120254 001401 BEQ 5$ ;IF MISS, BRANCH
20121 120256 104120 ERROR 0120
20122 120260 022737 012525 002710 5$; CMP 012525,TEMP ;DATA OK?
20123 120266 001401 BEQ 6$ ;IF SO, BRANCH
20124 120270 104123 ERROR 0123 ;DATA
20125 120272 005702 6$; TST R2 ;LAST PRIORITY OK?
20126 120274 001341 BNE 3$ ;IF NOT, BRANCH
20127 120276 106427 000340 MTPS 0340 ;RESTORE PRIORITY
20128
20129 ; VERIFY THAT DMA WITH FORCE MISS DOES NOT INVALIDATE CACHE
20130 ;

```

CORDBO RDJ11 B CLUSTER MACY11 30(1046) 05 APR-84 16:45 PAGE 386
 CORDAB,P11 05-APR-84 16:45 TEST - DMA WRITE HIT CYCLES

SEQ 0385

20131	120302	005037	002710		CLR	TEMP			
20132	120306	012737	000004	177746	MOV	#BIT2,CCR			;ALLOCATE CACHE
20133	120314	004737	132752		JSR	PC,DMA TRN			;SET FORCE MISS
20134	120320	005037	177746		CLR	CCR			;DO DMA DATI
20135	120324	042737	001000	177520	BIC	#1000,BCSR			;CLEAR FORCE MISS
20136	120332	005737	002710		IST	TEMP			;DISABLE HALT ON BREAK
20137	120336	013737	177752	110504	MOV	HITMIS,RECDAT			;STILL IN CACHE
20138	120344	052737	001000	177520	BIS	#1000,BCSR			;STORE REGISTER
20139	120352	032737	000004	110504	BIT	#BIT02,RECDAT			;ENABLE HALT ON BREAK
20140	120360	001001			ENE	7\$			;HIT?
20141	120362	104123			ERROR	+123			;IF SO, BRANCH
20142	120364	005737	002710		7\$: IST	TEMP			;READ FROM CACHE?
20143	120370	001401			BEQ	8\$			;IF STILL 0, BRANCH
20144	120372	104123			ERROR	+123			
20145									
20146									; VERIFY THAT BYPASS WITH DMA DATI INVALIDATES CACHE
20147									
20148	120374	012737	001000	177746	8\$: MOV	#BIT09,CCR			;SET BYPASS
20149	120402	004737	132752		JSR	PC,DMA TRN			;DO DMA DATI
20150	120406	005037	177746		CLR	CCR			;CLEAR BYPASS
20151	120412	042737	001000	177520	BIC	#1000,BCSR			;DISABLE HALT ON BREAK
20152	120420	005737	002710		IST	TEMP			;IN CACHE?
20153	120424	013737	177752	110504	MOV	HITMIS,RECDAT			;STORE REGISTER
20154	120432	052737	001000	177520	BIS	#1000,BCSR			;ENABLE HALT ON BREAK
20155	120440	032737	000004	110504	BIT	#BIT02,RECDAT			;TEMP WAS A HIT?
20156	120446	001401			BEQ	DATB0			;IF NOT, BRANCH
20157	120450	104123			ERROR	+123			
20158									
20159									; DO DATB0
20160									
20161	120452	012701	000010		DATB0: MOV	#10,R1			;COUNTER FOR 8
20162	120456	012702	002710		MOV	#TEMP,R2			;START WITH TEMP
20163	120462	005022			1\$: CLR	(R2)+			;CLEAR ALL 16
20164	120464	077102			SUB	R1,1\$			;DO ALL 16
20165	120466	005200			INC	R0			;FLAG BLOCK MODE
20166	120470	012777	002710	062152	MOV	#TEMP,0BA			;LOAD DMA ADDRESS
20167	120476	012777	177770	062146	MOV	#177770,0WC			;DO 8 WORDS
20168	120504	012777	001701	062132	MOV	#1701,0CSR1			;16 WORDS FROM 3,1K
20169	120512	004737	132752		JSR	PC,DMA TRN			;DO DATB0
20170	120516	032777	010000	062122	BIT	#BIT12,0CSR2			;NO BLOCK MODE SLAVE?
20171	120524	001401			BEQ	2\$			;IF NOT, BRANCH
20172	120526	104123			ERROR	+123			;NO BLOCK MODE SLAVE
20173	120530	012701	000004		2\$: MOV	#4,R1			;COUNTER FOR 4 LOCATIONS
20174	120534	012702	002710		MOV	#TEMP,R2			;START WITH TEMP
20175	120540	042737	001000	177520	3\$: BIC	#1000,BCSR			;DISABLE HALT ON BREAK
20176	120546	005712			IST	(R2)			;ACCESS A LOCATION
20177	120550	013737	177752	110504	MOV	HITMIS,RECDAT			;STORE REGISTER
20178	120556	052737	001000	177520	BIS	#1000,BCSR			;ENABLE HALT ON BREAK
20179	120564	032737	000004	110504	BIT	#BIT02,RECDAT			;HIT?
20180	120572	001401			BEQ	4\$			;IF NOT, BRANCH
20181	120574	104121			ERROR	+121			;DMA DOES NOT INVALIDATE
20182	120576	022702	012525		4\$: CMP	#12525,(R2)+			;DAT0 OK?
20183	120602	001401			BEQ	5\$			;IF SO, BRANCH
20184	120604	104123			ERROR	+123			;IN DMA
20185	120606	062702	000002		5\$: ADD	#2,R2			;DO IN 2 WORDS
20186	120612	077126			SUB	R1,3\$			;DO ALL 16

CORDBA0 RDJ11 B CLUSTER MACY11 30(1046) 05 APR-84 16:45 PAGE 387
 CORDAB.P11 05-APR-84 16:45 TEST DMA WRITE HIT CYCLES

SEQ 0386

```

20187
20188
20189 ; DO 4K OF DATO AND CHECK FOR MISS
20190 ;
20191 120614 004737 132156 JSR PC,INITMM ;SET UP MMU
20192 120620 012737 002000 172354 MOV #2000,KIPAR6 ;START AT 32K
20193 120626 042737 100000 172314 BIC #BIT15,KIPDR6 ;NO BYPASS
20194 120634 052737 000001 177572 BIS #BIT00,MMR0 ;ENABLE MMU
20195 120642 012737 121002 000004 MOV #DATI,#04 ;IF 32K NXW
20196 120650 012702 140000 MOV #140000,R2 ;START WITH 32K
20197 120654 005712 ISL (R2) ;EXIT
20198 120656 012701 010000 MOV #10000,R1 ;COUNTER FOR 4K
20199 120662 005022 6$: CLR (R2)+ ;CLEAR ALL 16
20200 120664 077102 SOB R1,6$ ;DO ALL 16
20201 120666 005200 INC R0 ;FLAG BLOCK MODE
20202 120670 012777 000000 061752 MOV #0,#BA ;LOAD DMA ADDRESS
20203 120676 012777 170000 061746 MOV #10000,#WC ;DO 4K
20204 120704 012777 003701 061732 MOV #3701,#CSR1 ;16 WORDS FROM 32K
20205 120712 004737 132752 JSR PC,DMAIRN ;DO DATB0
20206 120716 012701 004000 7$: MOV #4000,R1 ;COUNTER FOR 4K LOCATIONS
20207 120722 012702 140000 MOV #140000,R2 ;START WITH 0
20208 120726 042737 001000 177520 8$: BIC #1000,BCSR ;DISABLE HALT ON BREAK
20209 120734 005712 ISL (R2) ;ACCESS A LOCATION
20210 120736 013737 177752 110504 MOV HITMIS,RECDAT ;STORE REGISTER
20211 120744 052737 001000 177520 BIS #1000,BCSR ;ENABLE HALT ON BREAK
20212 120752 032737 000004 110504 BIT #BIT02,RECDAT ;HIT?
20213 120760 001401 BEQ 9$ ;IF NOT, BRANCH
20214 120762 104121 ERROR +121 ;DMA DOES NOT INVALIDATE
20215 120764 022722 012525 9$: CMP #12525,(R2)+ ;DATO OK?
20216 120770 001401 BEQ 10$ ;IF SO, BRANCH
20217 120772 104123 ERROR +123 ;IN DMA
20218 120774 062702 000002 10$: ADD #2,R2 ;DO IN 2 WORDS
20219 121000 077126 SOB R1,8$ ;DO ALL OF THEM
20220
20221 ; CHECK DATI
20222 ;
20223 121002 005037 177572 DATI: CLR MMR0 ;DISABLE MMU
20224 121006 012706 001100 MOV #1100,SP ;RESTORE STACK
20225 121012 012737 133124 000004 MOV #1001,#04 ;AND TIMEOUT
20226 121020 012737 052525 002710 MOV #52525,TEMP ;LOAD MEMORY
20227 121026 005000 CLR R0 ;JUST 1 WORD
20228 121030 004737 133034 JSR PC,DMARD ;DO DATI
20229 121034 022777 052525 061612 CMP #52525,#DATA ;DATI OK?
20230 121042 001401 BEQ 11$ ;IF YES, BRANCH
20231 121044 104123 ERROR +123 ;DATI
20232
20233 ; DO DATB1
20234 ;
20235 121046 012701 000010 11$: MOV #10,R1 ;COUNTER FOR 16 LOCATIONS
20236 121052 012702 002710 MOV #TEMP,R2 ;START WITH TEMP
20237 121056 012703 002710 MOV #TEMP,R3 ;START WITH TEMP
20238 121062 010322 12$: MOV R3,(R2)+ ;PUT ADDRESSES
20239 121064 005723 ISL (R3)+ ;INCREMENT ADDRESS
20240 121066 077103 SOB R1,12$ ;DO ALL 16
20241 121070 005200 INC R0 ;FLAG BLOCK MODE
20242 121072 004737 133034 JSR PC,DMARD ;DO DATB1

```


COKDABO KDJ11-B CLUSTER MAC111 30(1046) 05-APR-84 16:45 PAGE 389
 COKDAB,P11 05-APR-84 16:45 TEST - DIFFERENT LEVELS OF INTERRUPTS

SEQ 0388

```

20258 ;SBTIL TEST - DIFFERENT LEVELS OF INTERRUPTS
20259 ;DIFFERENT LEVELS OF INTERRUPTS
20260 ;THIS TEST WILL PROGRAM Q22 BUS EXERCISER TO INTERRUPT AT DIFFERENT
20261 ;LEVELS. ARBITRATION BETWEEN DIFFERENT LEVELS OF INTERRUPTS AND
20262 ;PIRQ'S WILL BE TESTED.
20263 ;
20264 ;CHECK DIFFERENT LEVELS OF INTERRUPTS.
20265 ;ROUTINE TEST
20266 ;. SET VECTOR TO INTERRUPT_DMA
20267 ;. FOR INTERRUPTS FROM 4 TO 7 DO
20268 ;. . ENABLE INTERRUPTS
20269 ;. . SET PRIORITY=INTERUPT
20270 ;. . IF INTERRUPT_FLAG SET THEN
20271 ;. . . ERROR
20272 ;. . . ENDDIF
20273 ;. . . ENABLE INTERRUPTS
20274 ;. . . SET PRIORITY=INTERUPT 1
20275 ;. . . IF INTERRUPT_FLAG NOTSET THEN
20276 ;. . . . ERROR
20277 ;. . . . ENDDIF
20278 ;. . . LET INTERRUPT_DMA=0
20279 ;. ENDDO
20280 ;ENDROUTINE
20281 ;
20282 ;ROUTINE INTERRUPT_DMA
20283 ;. LET INTERRUPT_FLAG=1
20284 ;RETURN
20285 ;ENDROUTINE
20286
20287 ;*****
20288 121162 000004 TST61: SCOPE
20289 121164 052737 000200 000052 BIT #B1107,0052 ;UFD MODE?
20290 121172 001122 BNE 15162 ;IF NO, EXIT TEST
20291 121174 005737 002644 TST CSR1 ;AT LEAST ONE Q22BE FOUND?
20292 121200 001517 BEQ 15162 ;IF NO1, EXIT TEST
20293 ;
20294 ; SETUP INITIAL PRIORITY TO 7
20295 ;
20296 121202 013703 002644 MOV CSR1,R3 ;DO FOR FIRST FOUND Q22BE
20297 121206 012777 121502 061442 MOV #53,0VQBE1 ;POINT INTERRUPT VECTOR TO PROGRAM
20298 121214 012777 000340 061436 MOV #340,0VQPR1 ;AT PRIORITY 7
20299 121222 012700 002752 MOV #Q22PEN,R0 ;START WITH 7 FOR INTERRUPTS
20300 121226 012701 000340 MOV #340,R1 ;LOW BOUNDARY FOR NO INTERRUPTS
20301 121232 000402 BR 23 ;TRY TO DO IT FOR FIRST
20302 121234 162701 000040 13: SUB #40,R1 ;LOWER LOW BOUNDARY
20303 ;
20304 ; CHECK THAT INTERRUPTS DON'T HAPPEN AT PRIORITY HIGHER THAN BR
20305 ;
20306 121240 012737 000340 001160 23: MOV #340,$TMP0 ;TOP PRIORITY FOR NO INTERRUPTS
20307 121246 000403 BR 43 ;DO FIRST ONE
20308 121250 162737 000040 001160 33: SUB #40,$TMP0 ;DO AT NEXT LEVEL
20309 121256 106437 001160 43: MTPC $TMP0 ;SET PRIORITY NOT TO INTERRUPT
20310 121262 004737 132732 JBR PC,Q22INT ;ENABLE INTERRUPTS
20311 121266 012077 061354 MOV (R0)+,0CSR2 ;CLEAR GO BIT
20312 121272 000240 NOP
20313 121274 000240 NOP

```

N14

COKDABO KDJ11-B CLUSTER MACY11 30(1046) 05 APR-84 16:45 PAGE 390
COKDAB.P11 05-APR-84 16:45 TEST - DIFFERENT LEVELS OF INTERRUPTS

SEQ 0389

```

20314 121276 000240      NOP
20315 121300 000403      BR      6$
20316 121302 104126      5$:  ERROR  +126      ; IF NO INTERRUPT, BRANCH
20317 121304 005726      TST      (SP),      ; INTERRUPTS HAPPEN
20318 121306 005726      TST      (SP),      ; RESTORE STACK
20319 121310 020137 001160 6$:  CMP      R1,$TMP0      ; LAST ONE?
20320 121314 001355      BNE      3$      ; IF NOT BRANCH
20321 121316 022710 000002  CMP      #2,(R0)      ; AT BR4?
20322 121322 001344      BNE      1$      ; IF NOT LAST ONE, BRANCH
20323
20324      ;
20325      ; INTERRUPT AT ALL LEVELS
20326 121324 012777 121420 061324 INQ22: MOV      #5$,@VQBE1      ; POINT INTERRUPT VECTOR TO PROGRAM
20327 121332 012777 000340 061320  MOV      #340,@VQPR1      ; AT PRIORITY 7
20328 121340 012700 002752      MOV      #Q22EN,R0      ; START WITH 7 FOR INTERRUPTS
20329 121344 012701 000300      MOV      #300,R1      ; TOP BOUNDARY FOR INTERRUPTS
20330 121350 000402      BR      2$      ; TRY TO DO IT FOR FIRST
20331 121352 162701 000040 1$:  SUB      #40,R1      ; LOWER TOP BOUNDARY
20332
20333      ;
20334      ; CHECK THAT INTERRUPTS HAPPEN AT PRIORITY LOWER THAN BR
20335 121356 010137 001160 2$:  MOV      R1,$TMP0      ; PRIORITY FOR INTERRUPTS
20336 121362 000403      BR      4$      ; DO FIRST ONE
20337 121364 162737 000040 001160 3$:  SUB      #40,$TMP0      ; DO AT NEXT LEVEL
20338 121372 106437 001160 4$:  MTPS      $TMP0      ; SET PRIORITY NOT TO INTERRUPT
20339 121376 004737 132732      LSR      PC,Q22INT      ; ENABLE INTERRUPTS
20340 121402 011077 061240      MOV      (R0),@CSR2      ; CLEAR GO BIT
20341 121406 000240      NOP
20342 121410 000240      NOP
20343 121412 000240      NOP
20344 121414 104126      ERROR  +126      ; INTERRUPTS DON'T HAPPEN
20345 121416 000402      BR      6$      ; DON'T RESTORE STACK
20346 121420 005726      5$:  TST      (SP),      ; RESTORE STACK
20347 121422 005726      TST      (SP),
20348 121424 005737 001160 6$:  TST      $TMP0      ; LAST ONE?
20349 121430 001355      BNE      3$      ; IF NOT BRANCH
20350 121432 022720 000002  CMP      #2,(R0)+      ; AT BR4?
20351 121436 001345      BNE      1$      ; IF NOT LAST ONE, BRANCH
20352
20353

```

1PIRQ-507-40

015

CONRAD, R. J. 11 B. CLUSTER MAC111 30010461 05 APR 84 16:45 PAGE 392
 CONRAD, R. J. 05 APR 84 16:45 TEST POWER DOWN TEST

SEQ 0391

```

20406 ;SBITL TEST POWER DOWN TEST
20407 ;USING Q22BF THIS TEST WILL CHECK THAT ON POWER DOWN CONDITION IF
20408 ;POWER UP CODE 00 IS SELECTED THE CPU TRAPS THRU 24
20409 ;ROUTINE TEST
20410 1. IF 00D OR POWER UP CODE 00 NOT SELECTED THEN
20411 1. EXIT TEST
20412 1. ENDIF
20413 1. SET 24 TO POINT TO TEST AREA
20414 1. LET CSR2<5> = 01 TO NEGATE BPOK
20415 1. IF NO TRAP TO 24 THEN
20416 1. ERROR IN POWER DOWN CYCLE
20417 1. ENDIF
20418 1. IF TRAP TO 24 THEN
20419 1. LET CSR2<5> = 00
20420 1. ENDIF
20421 ;ENDROUTINE
20422
20423 1.*****
20424 121624 000004 TST63: SCOPE
20425 121626 012737 000200 000052 BIT 0BIT07,0052 ;00D MODE?
20426 121634 001033 BNE TST64 ;EXIT TEST IN 00D MODE
20427 121636 005737 002644 TST CSR1 ;AT LEAST ONE Q22BF FOUND?
20428 121642 001430 BEQ TST64 ;IF NOT, EXIT TEST
20429 121644 013737 000024 001160 MOV PWRVEC,$TMP0 ;SAVE POWER UP VECTOR
20430 121652 012737 121706 000024 MOV 01$,PWRVEC ;POINT NEW TO PROGRAM
20431 121660 012737 000340 000016 MOV 0340,PWRVEC+2 ;AT PRIORITY 7
20432 121666 012777 000040 060752 MOV 0BIT05,0CSR2 ;DO POWER DOWN
20433 121674 000240 NOP
20434 121676 005077 060744 CLR 0CSR2 ;CLEAR POWER DOWN BIT
20435 121702 104125 ERROR 1,15 ;NO POWER DOWN TRAP
20436 121704 000404 BR 24 ;SKIP RESTORING STACK
20437 121706 005077 060734 1$: CLR 0CSR2 ;CLEAR POWER DOWN BIT
20438 121712 005726 TST (SP)+ ;RESTORE STACK POINTER
20439 121714 005726 TST (SP)+
20440 121716 013737 001160 000024 2$: MOV $TMP0,PWRVEC ;RESTORE POWER VECTOR
20441
20442
20443

```

CORDABO RDJ11 B CLUSTER MACY11 3001046)
CORDABO.P11 05 APR 84 16:45

05 APR 84 16:45 PAGE 393
TEST ARBITRATION BETWEEN DIFFERENT LEVELS OF INTERRUPTS

SEQ 0392

```

20444 ;SMTL TEST ARBITRATION BETWEEN DIFFERENT LEVELS OF INTERRUPTS
20445 ;IF TWO Q22BE ARE AVAILABLE, THIS TEST WILL CHECK THAT HIGHER LEVEL
20446 ;INTERRUPT REQUESTS TAKE PRIORITY OVER LOWER LEVEL ONES
20447 ;ROUTINE TEST
20448 ;. IF 1ST OR 2ND Q22BE NOT AVAILABLE THEN
20449 ;. EXIT TEST
20450 ;. ENDDI
20451 ;. DO FOR PRIORITY LEVELS FROM 4 TO 6
20452 ;. . SET UP 1ST Q22BE TO INTERRUPT AT PRIORITY LEVEL
20453 ;. . SET UP 2ND Q22BE TO INTERRUPT AT PRIORITY LEVEL +1
20454 ;. . SET UP CPU PRIORITY TO PRIORITY LEVEL +1
20455 ;. . IF INTERRUPTS FORM 1ST Q22BE HAPPENED BEFORE 1ST THEN
20456 ;. . . ERROR
20457 ;. . . ENDDI
20458 ;. ENDDO
20459 ;ENDROUTINE
20460
20461 ;*****
20462 121724 000004 TST64: SCOPE
20463 121726 052757 000200 000052 BIT 0BIT07,0052 ;CPU MODE?
20464 121734 001064 BNE TST65 ;IF SO, EXIT TEST
20465 121736 005737 002664 TST CSR12 ;SECOND Q22BE AVAILABLE?
20466 121742 001002 BNE 13 ;IF YES, GO DO TEST
20467 121744 000137 133514 JMP $EOP ;OTHERWISE, GOTO EOP
20468
20469 ; INITIALISE VECTORS FOR Q22BE'S
20470 ;
20471 121750 012777 122064 060700 13: MOV 041,0VQBE1 ;VECTOR FOR 1ST ONE
20472 121756 012777 000340 060674 MOV 0340,0VQPR1 ;AT PRIORITY 7
20473 121764 012777 122074 060704 MOV 051,0VQBE2 ;VECTOR FOR 2ND ONE
20474 121772 012777 000340 060700 MOV 0340,0VQPR2 ;AT PRIORITY 7
20475 122000 012700 002754 MOV 0Q22EN2,R0 ;POINTER FOR Q22BE BRG
20476 122004 012702 000240 MOV 0240,R2 ;CPU AT 5
20477 122010 000402 BR 31 ;START DOING ARBITRATION
20478
20479 ; DO FOR CPU PRIORITIES 5-3
20480 ;
20481 122012 162702 000040 23: SUB 040,R2 ;LOWER CPU PRIORITY
20482 122016 106427 000340 33: MTP5, 0340 ;CPU AT 7
20483 122022 013703 002664 MOV CSR12,R3 ;Q22BE 2 AT HIGHER BR
20484 122026 004757 132752 JSR PC,Q22INT ;INITIALISE TO INTERRUPTS
20485 122032 011077 060610 MOV (R0),0CSR2 ;START 1ST ONE
20486 122036 013703 002664 MOV CSR1,R3 ;Q22BE 1 AT LOWER BR
20487 122042 005740 TST (R0) ;HIGHER PRIORITY
20488 122044 004757 132752 JSR PC,Q22INT ;INITIALISE
20489 122050 012077 060612 MOV (R0),0CSR2 ;START 2ND ONE
20490 122054 106402 MTP5, R2 ;LOWER CPU PRIORITY
20491 122056 000240 NOP ;WAIT A WHILE
20492 122060 000240 NOP
20493 122062 000240 NOP
20494 122064 104126 43: ERROR +126 ;INTERRUPTS IN WRONG ORDER
20495 122066 005726 TST (SP) ;RESTORE STACK
20496 122070 005726 TST (SP)
20497 122072 000402 BR 63
20498 122074 005726 53: TST (SP) ;RESTORE STACK
20499 122076 005726 TST (SP)

```

CORDABO RDJ11 B CLUSTER MAC111 3001046) 05 APR-84 16:45 PAGE 394
CORDAB,P11 05 APR 84 16:45

TEST ARBITRATION BETWEEN DIFFERENT LEVELS OF INTERRUPTS

SEQ 0393

20500 122100 002102 000140
20501 122104 001342
20502

65: CMP #140,R2
BNE 25

;PRIORITY 3 CALL?
;IF NOT, BRANCH TO CONTINUE

CORDBARO K0311 B CLUSTER MACY11 30(1046)
COKDAR.P11 05-APR-84 16:45

05 APR 84 16:45 PAGE 395
TEST - PMG COUNTER

SEQ 0394

```
20503 .SRTTL TEST - PMG COUNTER
20504 ;USING 2 Q22BE THIS TEST WILL VALIDATE THAT PMG COUNTER IS REALLY
20505 ;CAPABLE OF GRANTING CPU BUS MASTERSHIP WHEN DMA REQUESTS ARE STILL
20506 ;PENDING.
20507 ;ROUTINE TEST
20508 ;; IF OFD OR NO 2ND Q22BE THEN
20509 ;; . EXIT TEST
20510 ;; .
20511 ;; . ENDIF
20512 ;; SET PMG COUNT TO SOME VALUE
20513 ;; PROGRAM BOTH Q22BE TO INTERRUPT AT THE SAME LEVEL
20514 ;; DETERMINE WHICH HAS HIGHER PRIORITY ON THE BUS
20515 ;; PROGRAM Q22BE WITH HIGHER PRIORITY TO DO HOG MODE
20516 ;; PROGRAM 2ND ONE TO DO A CYCLE
20517 ;; CACHE INSTRUCTION THAT WILL STOP 2ND DMA
20518 ;; INITIATE BOTH DMA CYCLES
20519 ;; STOP 2ND DMA (RESET IS "STOLEN" CPU BUS CYCLE)
20520 ;; IF 2ND DMA HAPPENED THEN
20521 ;; . ERROR
20522 ;; .
20523 ;; . ENDIF
20524 ;; CLEAR PMG COUNT
20525 ;; PROGRAM Q22BE WITH HIGHER PRIORITY TO DO HOG MODE
20526 ;; PROGRAM 2ND ONE TO DO A CYCLE
20527 ;; CACHE INSTRUCTION THAT WILL STOP 2ND DMA
20528 ;; INITIATE BOTH DMA CYCLES
20529 ;; STOP 2ND DMA (CLEARING OF CSR2:05 IS "STOLEN" CPU BUS CYCLE)
20530 ;; IF 2ND DMA DIDN'T HAPPENED THEN
20531 ;; . ERROR
20532 ;; .
20533 ;; . ENDIF
20534 ;ENDROUTINE
20535 ;*****
20536 TST65: SCOPE
20537 BIT 081107,0052 ;OFD MODE?
20538 BEQ 1003 ;IF NOT, CONTINUE
20539 JMP $EOP ;EXIT
20540 1003: TST CSR12 ;SECOND Q22BE AVAILABLE?
20541 BNE 1103 ;IF YES, GO DO TEST
20542 JMP $EOP ;OTHERWISE, GOTO EOP
20543 ;
20544 ; DETERMINE WHICH Q22BE IS CLOSER TO CPU BY DOING DATA
20545 ; THE CSR1 OF THE ONE WITH HIGHER PRIORITY IS IN R4, THE SECOND IN R5
20546 ;
20547 1103: CLR 0CSR2 ;CLEAR JUST IN CASE
20548 CLR 0CSR12
20549 MOV 0TEMP,0BA ;ADDRESS TO BE USED
20550 MOV 0TEMP,0BA2 ;FOR BOTH OF THEM
20551 MOV 017777,0WC ;DO FOR 1 WORD
20552 MOV 017777,0WC2 ;IN BOTH Q22BES
20553 MOV 012525,0DATA ;DATA FOR 1ST
20554 MOV 052525,0DATA2 ;DATA FOR 2ND
20555 MOV 01601,0CSR1 ;1 DATA FOR 1ST
20556 MOV 01601,0CSR12 ;AND 2ND
20557 MOV 01,05IM6DA ;BOTH GO
20558 13: TSTB 0CSR1 ;FIRST DONE?
20559 BPL 13 ;IF NOT, WAIT
20560 23: TSTB 0CSR12 ;SECOND DONE
```

CORRADO RDJ11 B CLUSTER MACY11 50(1046) 05 APR-84 16:45 PAGE 396
 CORRADO P11 05-APR-84 16:45 TEST PMG COUNTER

SEQ 0395

```

20559 122246 100375          BPL      2$
20560 122250 022737 052525 002710  CMP      052525,TEMP
20561 122256 001405          BEQ      3$
20562 122260 013704 002664      MOV      CSR1,R4
20563 122264 013702 002644      MOV      CSR1,R2
20564 122270 000404          BR       4$
20565 122272 013704 002644 3$:    MOV      CSR1,R4
20566 122276 013702 002664      MOV      CSR1,R2
20567
20568      ;
20569      ; INITIATE DMA CYCLES TO WORK WITH PMG COUNTER
20570
20571 122302 013737 177520 002710 4$:    MOV      BCSR,SAVBR
20572 122310 012700 000001      MOV      0BIT00,R0
20573 122314 050037 177520 5$:    BIS      R0,BCSR
20574 122320 005737 122422      TST      6$
20575 122324 005737 122424      TST      6$+2
20576 122330 005737 122426      TST      6$+4
20577 122334 005737 122430      TST      6$+6
20578 122340 012714 001407      MOV      01407,(R4)
20579 122344 005064 000002      CLR      2(R4)
20580 122350 012764 002710 000004  MOV      0TEMP,4(R4)
20581 122356 012764 000000 000006  MOV      00,6(R4)
20582 122364 012712 001407      MOV      01407,(R2)
20583 122370 005062 000002      CLR      2(R2)
20584 122374 012762 002710 000004  MOV      0TEMP,4(R2)
20585 122402 012762 177777 000006  MOV      017777,6(R2)
20586 122410 005062 000010      CLR      10(R2)
20587 122414 012762 000001 000016  MOV      01,16(R2)
20588 122424 023762 002710 000010 6$:    RESET
20589 122432 001001          CMP      TEMP,10(R2)
20590 122434 104127          BNE      7$
20591 122436 062700 000003 7$:    ADD      03,R0
20592 122442 040037 177520          BIC      R0,BCSR
20593 122446 022700 000007          CMP      07,R0
20594 122452 002320          BGE      5$
20595
20596      ;
20597      ; TRY WITHOUT PMG COUNTER OPERATING
20598
20599 122454 042737 000007 177520      BIC      07,BCSR
20600 122462 005737 122564          TST      8$
20601 122466 005737 122566          TST      8$+2
20602 122472 005737 122570          TST      8$+4
20603 122476 005737 122572          TST      8$+6
20604 122502 012714 001407      MOV      01407,(R4)
20605 122506 005064 000002      CLR      2(R4)
20606 122512 012764 002710 000004  MOV      0TEMP,4(R4)
20607 122520 012764 000000 000006  MOV      00,6(R4)
20608 122526 012712 001407      MOV      01407,(R2)
20609 122532 005062 000002      CLR      2(R2)
20610 122536 012762 002710 000004  MOV      0TEMP,4(R2)
20611 122544 012762 177777 000006  MOV      017777,6(R2)
20612 122552 005062 000010      CLR      10(R2)
20613 122556 012777 000001 000076  MOV      01,255ING0A
20614 122566 023762 002710 000010 8$:    RESET
          CMP      TEMP,10(R2)

```

```

;WAIT FOR PND
;SECOND FINISHED LAST?
;IF SO, BRANCH
;SECOND ONE AT HIGHER LEVEL
;FIRST AT LOWER
;DO PMG PART
;FIRST ONE AT HIGHER LEVEL
;SECOND AT LOWER

;STORE BCSR REGISTER
;FIRST VALUE FOR PMG 0.4MSEC
;CHANGE PMG COUNTER
;CACHE RESET INSTRUCTION
;AND THE FOLLOWING FEW
;
;DATA IN HOG MODE FOR HIGHER ONE
;CLEAR CSR2
;ADDRESS TO START DATA
;DO 128 DATAS IN HOG MODE
;DATA FOR LOWER LEVEL ONE
;CLEAR JUST IN CASE
;USE THE SAME ADDRESS
;DO JUST ONE DATA
;CLEAR DATA OF PND Q2PBE
;BOTH GO
;STOP Q2PBE AT R4
;SECOND DMA DONE?
;IF SET, BRANCH
;NO CYCLE STEALING
;DO FOR 1,3,7 IN PMG
;CLEAR PREVIOUS BITS IN BCSR
;LAST ONE?
;IF NOT, BRANCH

;TURN OF PMG COUNTER
;CACHE RESET FECH
;AND THE FOLLOWING FEW
;
;DATA IN HOG MODE FOR HIGHER ONE
;CLEAR CSR1
;ADDRESS TO START DATA
;DO 128 DATAS IN HOG MODE
;DATA FOR LOWER LEVEL ONE
;CLEAR CSR2 OF PND
;USE THE SAME ADDRESS
;DO JUST ONE DATA
;CLEAR DATA OF PND Q2PBE
;BOTH GO
;IF NOT WORKING, STOPS PND
;END DMA HAPPENED?

```


1115

CORRADO KDJ11 B CLUSTER MACY11 50(1046) 05-APR-84 16:45 PAGE 397
CORRADO.P11 05-APR 84 16:45 TEST PMG COUNTER

SEQ 0396

```
20615 122574 001401      BEQ      9$      ; IF YES, BRANCH
20616 122576 104127      ERROR    9127     ; IN PMG COUNTER
20617 122600 013737 002710 177520 9$;      MOV      SAVBR,BCSR    ; RESTORE BCSR
20618
20619
20620
20621 122606 000004      ;*****
20622 122606 000004      TST66; SCOPE
20623 122610 000137 133514      JMP      $EOP
```

COKDABO RDJ11 B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 398
 COKDAB.P11 05-APR-84 16:45 GLOBAL ERROR MESSAGES

SEQ 0397

					SHIFT	GLOBAL ERROR MESSAGES
20624					EM1:	.ASCIIZ /BASIC INSTRUCTION SET ERROR/
20625	122614	040502	044523	020103		
20626	122622	047111	052123	052522		
20627	122630	052103	047511	020116		
20628	122636	042523	020124	051105		
20629	122644	047522	000122			
20630	122650	046515	020125	051105	EM2:	.ASCIIZ /MMU ERROR/
20631	122656	047522	000122			
20632	122662	050106	020120	051105	EM3:	.ASCIIZ /FPP ERROR/
20633	122670	047522	000122			
20634	122674	051105	047522	020122	EM4:	.ASCIIZ /ERROR IN READ-WRITE BITS OF CCR/
20635	122702	047111	051040	040505		
20636	122710	026504	051127	052111		
20637	122716	020105	044502	051524		
20638	122724	047440	020106	041503		
20639	122732	000122				
20640	122734	047506	041522	020105	EM5:	.ASCIIZ /FORCE MISS WRITES TO CACHE/
20641	122742	044515	051523	053440		
20642	122750	044522	042524	020123		
20643	122756	047524	041440	041501		
20644	122764	042510	000			
20645	122767	106	051117	042503	EM6:	.ASCIIZ /FORCE MISS WRITE INVALIDATES CACHE/
20646	122774	046440	051511	020123		
20647	123002	051127	052111	020105		
20648	123010	047111	040526	044514		
20649	123016	040504	042524	020123		
20650	123024	040503	044103	000105		
20651	123032	047125	054105	042520	EM7:	.ASCIIZ /UNEXPECTED PARITY INTERRUPT/
20652	123040	052103	042105	050040		
20653	123046	051101	052111	020131		
20654	123054	047111	042524	051122		
20655	123062	050125	000124			
20656	123066	040524	020107	040520	EM10:	.ASCIIZ /TAG PARITY ERROR/
20657	123074	044522	054524	042440		
20658	123102	051122	051117	000		
20659	123107	104	052101	020101	EM11:	.ASCIIZ /DATA PARITY ERROR/
20660	123114	040520	044522	054524		
20661	123122	042440	051122	051117		
20662	123130	000				
20663	123131	114	053517	041040	EM12:	.ASCIIZ /LOW BYTE PARITY ERROR/
20664	123136	052131	020105	040520		
20665	123144	044522	054524	042440		
20666	123152	051122	051117	000		
20667	123157	110	043511	020110	EM13:	.ASCIIZ /HIGH BYTE PARITY ERROR/
20668	123164	054502	042524	050040		
20669	123172	051101	052111	020131		
20670	123200	051105	047522	000122		
20671	123206	051105	047522	020122	EM14:	.ASCIIZ /ERROR IN DATA PATH/
20672	123214	047111	042040	052101		
20673	123222	020101	040520	044124		
20674	123230	000				
20675	123231	106	051117	042503	EM15:	.ASCIIZ /FORCE MISS READS FROM CACHE/
20676	123236	046440	051511	020123		
20677	123244	042522	042101	020123		
20678	123252	051105	047517	041440		
20679	123260	041501	042510	000		

CORRADO RDJ11 B CLUSTER MACY11 3001046) 05 APR 84 16:45 PAGE 399
CORRADO P11 05-APR 84 16:45 GLOBAL ERROR MESSAGES

SEQ 0398

20680	123265	106	051117	042503	EM16:	.ASCIZ /FORCE MISS READS FROM CACHE AND MISS/
20681	123272	046440	051511	020123		
20682	123300	042522	042101	020123		
20683	123306	051106	046517	041440		
20684	123314	041501	042510	040440		
20685	123322	042116	046440	051511		
20686	123330	000123				
20687	123332	051105	042522	020123	EM17:	.ASCIZ /ERROR IN RECORDING HITS IN HIT/MISS/
20688	123340	047111	051040	041505		
20689	123346	051117	044504	043516		
20690	123354	044040	052111	020123		
20691	123362	047111	044040	052111		
20692	123370	046457	051511	000123		
20693	123376	051127	052111	020105	EM20:	.ASCIZ /WRITE BYTE ALLOCATES CACHE/
20694	123404	054502	042524	040440		
20695	123412	046114	041517	052101		
20696	123420	051505	041440	041501		
20697	123426	042510	000			
20698	123431	127	044522	042524	EM21:	.ASCIZ /WRITE BYTE HIT DOES NOT RECORD HIT/
20699	123436	041040	052131	020105		
20700	123444	044510	020124	047504		
20701	123452	051505	047040	052117		
20702	123460	051040	041505	051117		
20703	123466	020104	044510	000124		
20704	123474	054502	042524	020123	EM22:	.ASCIZ /BYTES REVERSED ON WRITE CYCLES/
20705	123502	042522	042526	051522		
20706	123510	042105	047440	020116		
20707	123516	051127	052111	020105		
20708	123524	054503	046103	051505		
20709	123532	000				
20710	123533	103	047117	044504	EM23:	.ASCIZ /CONDITIONAL BYPASS DOESN'T INVALIDATE CACHE/
20711	123540	044524	047117	046101		
20712	123546	041040	050131	051501		
20713	123554	020123	047504	051505		
20714	123562	023516	020124	047111		
20715	123570	040526	044514	040504		
20716	123576	042524	041440	041501		
20717	123604	042510	000			
20718	123607	110	052111	020123	EM24:	.ASCIZ /HITS RECORDED AFTER FLUSHING CACHE/
20719	123614	042522	047503	042122		
20720	123622	042105	040440	052106		
20721	123630	051105	043040	052514		
20722	123636	044123	047111	020107		
20723	123644	040503	044103	000105		
20724	123652	054502	040520	051523	EM25:	.ASCIZ /BYPASS DOESN'T INVALIDATE CACHE/
20725	123660	042040	042517	047123		
20726	123666	052047	044440	053116		
20727	123674	046101	042111	052101		
20728	123702	020105	040503	044103		
20729	123710	000105				
20730	123712	051515	051105	042040	EM26:	.ASCIZ /MSR DOES NOT CLEAR ON WRITE REFERENCE/
20731	123720	042517	020123	047516		
20732	123726	020124	046103	040505		
20733	123734	020122	047117	053440		
20734	123742	044522	042524	051040		
20735	123750	043105	051105	047105		

COKDABO KDJ11 B CLUSTER MACY11 50(1046) 05-APR-84 16:45 PAGE 400
 COKDAB.P11 05-APR-84 16:45 GLOBAL ERROR MESSAGES

SEQ 0399

20736	123756	042503	000		
20737	123761	120	051101	052111	EM27: ,ASCIIZ /PARITY ERROR DON'T CAUSE A MISS/
20738	123766	020131	051105	047522	
20739	123774	020122	047504	023516	
20740	124002	020124	040503	051525	
20741	124010	020105	020101	044515	
20742	124016	051523	000		
20743	124021	120	051101	052111	EM30: ,ASCIIZ /PARITY ERROR DON'T SET MSER WITH CCR<7>=0/
20744	124026	020131	051105	047522	
20745	124034	020122	047504	023516	
20746	124042	020124	042523	020124	
20747	124050	051515	051105	053440	
20748	124056	052111	020110	041503	
20749	124064	036122	037067	030075	
20750	124072	000			
20751	124073	120	051101	052111	EM31: ,ASCIIZ /PARITY ERROR IGNORED/
20752	124100	020131	051105	047522	
20753	124106	020122	043511	047516	
20754	124114	012522	000104		
20755	124120	040520	044522	054524	EM32: ,ASCIIZ /PARITY ERROR IGNORED ON LOW BYTE/
20756	124126	042440	051122	051117	
20757	124134	044440	047107	051117	
20758	124142	042105	047440	020116	
20759	124150	047514	020127	054502	
20760	124156	042524	000		
20761	124161	120	051101	052111	EM33: ,ASCIIZ /PARITY ERROR IGNORED ON HIGH BYTE/
20762	124166	020131	051105	047522	
20763	124174	020122	043511	047516	
20764	124202	042522	020104	047117	
20765	124210	044040	043511	020110	
20766	124216	054502	042524	000	
20767	124223	120	051101	052111	EM34: ,ASCIIZ /PARITY ABORT LOGIC DOESN'T WORK/
20768	124230	020131	041101	051117	
20769	124236	020124	047514	044507	
20770	124244	020103	047504	051505	
20771	124252	023516	020124	047527	
20772	124260	045522	000		
20773	124263	115	042523	020122	EM35: ,ASCIIZ /MSER NOT SET PROPERLY/
20774	124270	047516	020124	042523	
20775	124276	020124	051120	050117	
20776	124304	051105	054514	000	
20777	124311	120	051101	052111	EM36: ,ASCIIZ /PARITY INTERRUPT LOGIC DOESN'T WORK/
20778	124316	020131	047111	042524	
20779	124324	051122	050125	020124	
20780	124332	047514	044507	020103	
20781	124340	047504	051505	023516	
20782	124346	020124	047527	045522	
20783	124354	000			
20784	124355	116	046530	040440	EM37: ,ASCIIZ /NXM AND PARITY ABORT DIN'T HAPPEN/
20785	124362	042116	050040	051101	
20786	124370	052111	020131	041101	
20787	124376	051117	020124	044504	
20788	124404	023516	020124	040510	
20789	124412	050120	047105	000	
20790	124417	120	051101	052111	EM40: ,ASCIIZ /PARITY ABORT NOT BLOCKED BY NXM TRAP/
20791	124424	020131	041101	051117	

CORDABO RDJ11 8 CLUSTER MACY11 30(1046) 05 APR 84 16:45 PAGE 401
 CORDAB.P11 05 APR 84 16:45 GLOBAL ERROR MESSAGES

SEQ 0400

20792	124432	020124	047516	020124	
20793	124440	046102	041517	042513	
20794	124446	020104	054502	047040	
20795	124454	046530	052040	040522	
20796	124462	000120			
20797	124464	052515	052114	026511	EM41: .ASCIZ /MULTI-PROCESSOR HOOK INSTRUCTION DOESN'T CAUSE MISS/
20798	124472	051120	041517	051505	
20799	124500	047523	020122	047510	
20800	124506	045517	044440	051516	
20801	124514	051124	041525	044524	
20802	124522	047117	042040	042517	
20803	124530	047123	052047	041440	
20804	124536	052501	042523	046440	
20805	124544	051511	000123		
20806	124550	051105	047522	020122	EM42: .ASCIZ /ERROR IN PARITY LOGIC/
20807	124556	047111	050040	051101	
20808	124564	052111	020131	047514	
20809	124572	044507	000103		
20810	124576	051105	047522	020122	EM43: .ASCIZ /ERROR IN CACHE DATA RAMS/
20811	124604	047111	041440	041501	
20812	124612	042510	042040	052101	
20813	124620	020101	040522	051515	
20814	124626	000			
20815	124627	105	051122	051117	EM44: .ASCIZ /ERROR IN NXM IN STANDALONE MODE/
20816	124634	044440	020116	054116	
20817	124642	020115	047111	051440	
20818	124650	040524	042116	046101	
20819	124656	047117	020105	047515	
20820	124664	042504	000		
20821	124667	105	051122	051117	EM45: .ASCIZ /ERROR IN RECORDING HITS THROUGH HIT/MISS REGISTER/
20822	124674	044440	020116	042522	
20823	124702	047503	042122	047111	
20824	124710	020107	044510	051524	
20825	124716	052040	051110	052517	
20826	124724	044107	044040	052111	
20827	124732	046457	051511	020123	
20828	124740	042522	044507	052123	
20829	124746	051105	000		
20830	124751	110	052111	051040	EM46: .ASCIZ /HIT RECORDED FOR A LOCATION THAT SHOULD NOT BE IN CACHE/
20831	124756	041505	051117	042504	
20832	124764	020104	047506	020122	
20833	124772	020101	047514	040503	
20834	125000	044524	047117	052040	
20835	125006	040510	020124	044123	
20836	125014	052517	042114	047040	
20837	125022	052117	041040	020105	
20838	125030	047111	041440	041501	
20839	125036	042510	000		
20840	125041	115	051511	020123	EM47: .ASCIZ /MISS RECORDED FOR A LOCATION THAT SHOULD BE IN CACHE/
20841	125046	042522	047503	042122	
20842	125054	042105	043040	051117	
20843	125062	040440	046040	041517	
20844	125070	052101	047511	020116	
20845	125076	044124	052101	051440	
20846	125104	047510	046125	020104	
20847	125112	042502	044440	020116	

M15

CORRADO KDJ11 B CLUSTER MACY11 30(1046) 05 APR 84 16:45 PAGE 402
 CORRAB.P11 05-APR-84 16:45 GLOBAL ERROR MESSAGES

SEQ 0401

20848	125120	040503	044103	000105		
20849	125126	051105	047522	020122	EM50:	.ASCIZ /ERROR IN TAG STORE/
20850	125134	047111	05204	043501		
20851	125142	051440	047524	042522		
20852	125150	000				
20853	125151	105	051122	051117	EM51:	.ASCIZ /ERROR PCR READ-WRITE BITS/
20854	125156	050040	051103	051040		
20855	125164	040505	026504	051127		
20856	125172	052111	020105	044502		
20857	125200	051524	000			
20858	125203	105	051122	051117	EM52:	.ASCIZ /ERROR IN BCSR READ-WRITE BITS/
20859	125210	044440	020116	041502		
20860	125216	051123	051040	040505		
20861	125224	026504	051127	052111		
20862	125232	020105	044502	051524		
20863	125240	000				
20864	125241	122	051505	052105	EM53:	.ASCIZ /RESET DOESN'T CLEAR BCSR<4>/
20865	125246	042040	042517	047123		
20866	125254	052047	041440	042514		
20867	125262	051101	041040	051503		
20868	125270	036122	037064	000		
20869	125275	103	042510	045503	EM54:	.ASCIZ /CHECKSUM ERROR IN 16-BIT ROM /
20870	125302	052523	020115	051105		
20871	125310	047522	020122	047111		
20872	125316	030440	026466	044502		
20873	125324	020124	047522	020115		
20874	125332	000				
20875	125333	103	042510	045503	EM55:	.ASCIZ /CHECKSUM ERROR IN 8 BIT ROM/
20876	125340	052523	020115	051105		
20877	125346	047522	020122	047111		
20878	125354	034040	041055	052111		
20879	125362	051040	046517	000		
20880	125367	124	046511	047505	EM56:	.ASCIZ /TIMEOUT READING LKS/
20881	125374	052125	051040	040505		
20882	125402	044504	043516	046040		
20883	125410	051513	000			
20884	125413	114	051513	030074	EM57:	.ASCIZ /LKS<07> DOES NOT BECOME 1/
20885	125420	037067	042040	042517		
20886	125425	020123	047516	020124		
20887	125434	042502	047503	042515		
20888	125442	030440	000			
20889	125445	127	044522	042524	EM60:	.ASCIZ /WRITE REFERENCE DOESN'T CLEAR LKS<07>/
20890	125452	051040	043105	051105		
20891	125460	047105	042503	042040		
20892	125466	042517	047123	052047		
20893	125474	041440	042514	051101		
20894	125502	046040	051513	030074		
20895	125510	037067	000			
20896	125513	111	046114	043505	EM61:	.ASCIZ /ILLEGAL LKS INTERRUPTS/
20897	125520	046101	046040	051513		
20898	125526	044440	052116	051105		
20899	125534	052522	052122	000123		
20900	125542	051120	041517	051505	EM62:	.ASCIZ /PROCESSOR INTERRUPTS DON'T CLEAR LKS<07>/
20901	125550	047523	020122	047111		
20902	125556	042524	051122	050125		
20903	125564	051524	042040	047117		

N15

COKDABO RDJ11-B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 403
 COKDAB.P11 05-APR-84 16:45 GLOBAL ERROR MESSAGES

SEQ 0402

20904	125572	052047	041440	042514	
20905	125600	051101	046040	051513	
20906	125606	030074	037067	000	
20907	125613	114	051513	051040	EM63: .ASCIZ /LKS READY DOESN'T GO LOW/
20908	125620	040505	054504	042040	
20909	125626	042517	047123	052047	
20910	125634	043440	020117	047514	
20911	125642	000127			
20912	125644	051127	047117	020107	EM64: .ASCIZ /WRONG NUMBER OF LKS INTERRUPTS/
20913	125652	052516	041115	051105	
20914	125660	047440	020106	045514	
20915	125666	020123	047111	042524	
20916	125674	051122	050125	051524	
20917	125702	000			
20918	125703	114	051513	044440	EM65: .ASCIZ /LKS INTERRUPTS HAPPEN AT WRONG PRIORITY/
20919	125710	052116	051105	052522	
20920	125716	052120	020123	040510	
20921	125724	050120	047105	040440	
20922	125732	020124	051127	047117	
20923	125740	020107	051120	047511	
20924	125746	044522	054524	000	
20925	125753	102	051503	036122	EM66: .ASCIZ /BCSR<12> DOES NOT DISABLES LKS/
20926	125760	031061	020076	047504	
20927	125766	051505	047040	052117	
20928	125774	042040	051511	041101	
20929	126002	042514	020123	045514	
20930	126010	000123			
20931	126012	041502	051123	030474	EM67: .ASCIZ /BCSR<13> DOESN'T SET LKS<06>/
20932	126020	037067	042040	042517	
20933	126026	047123	052047	05146	
20934	126034	052105	046040	0515	
20935	126042	030074	037066	000	
20936	126047	122	051505	052105	EM70: .ASCIZ /RESET DOESN'T SET LKS<7>/
20937	126054	042040	042517	047123	
20938	126062	052047	051440	052105	
20939	126070	046040	051513	033474	
20940	126076	000076			
20941	126100	042522	042523	020124	EM71: .ASCIZ /RESET DOESN'T CLEAR LKS<06>/
20942	126106	047504	051505	023516	
20943	126114	020124	046103	040505	
20944	126122	020122	045514	036123	
20945	126130	033060	000076		
20946	126134	044524	042515	052517	EM72: .ASCIZ /TIMEOUT READING SLU REGISTERS/
20947	126142	020124	042522	042101	
20948	126150	047111	020107	046123	
20949	126156	020125	042522	044507	
20950	126164	052123	051105	000123	
20951	126172	051105	047522	020122	EM73: .ASCIZ /ERROR IN XMIT READY/
20952	126200	047111	054040	044515	
20953	126206	020124	042522	042101	
20954	126214	000131			
20955	126216	041522	051123	033474	EM74: .ASCIZ /BCSR<7> DOESN'T BECOME 1/
20956	126224	020076	047504	051505	
20957	126232	023516	020124	042507	
20958	126240	047507	042515	030440	
20959	126246	000			

CORRADO RDJ11 B CLUSTER MAC111 50(1046) 05 APR 84 16:45 PAGE 404
CORRADO P11 05 APR 84 16:45 GLOBAL ERROR MESSAGES

SEQ 0403

20960	126247	127	047522	043516	EM75:	,ASCII2 /WRONG CHARACTER RECEIVED/
20961	126254	041440	040510	040522		
20962	126262	052105	051105	051040		
20963	126270	041505	044505	042526		
20964	126276	000104				
20965	126300	041522	051125	030074	EM76:	,ASCII2 /RCSR<07> NOT CLEARED AFTER READING RBUF /
20966	126306	037067	047040	052117		
20967	126314	041440	042514	051101		
20968	126322	042105	040440	052106		
20969	126330	051105	051040	040505		
20970	126336	044504	043516	051040		
20971	126344	052502	000106			
20972	126350	041530	051125	030074	EM77:	,ASCII2 /XCSR<07> NOT SET ON RESET /
20973	126356	037067	047040	052117		
20974	126364	051440	052105	047440		
20975	126372	020116	042522	042523		
20976	126400	000124				
20977	126402	041522	051125	030074	EM100:	,ASCII2 /RCSR<07> NOT CLEARED ON RESET /
20978	126410	037067	047040	052117		
20979	126416	041440	042514	051101		
20980	126424	042105	047440	020116		
20981	126432	042522	042523	000124		
20982	126440	046123	020125	047111	EM101:	,ASCII2 /SIO INTERRUPTS HAPPEN AT 4 /
20983	126446	042524	051122	050125		
20984	126454	051524	044040	050101		
20985	126462	042520	020116	052101		
20986	126470	032040	000			
20987	126473	127	051505	052105	EM102:	,ASCII2 /RESET DOES NOT CLEAR PROPER BITS IN SIO REGISTERS /
20988	126500	042040	042517	020123		
20989	126506	047516	020124	046103		
20990	126514	040505	020122	051170		
20991	126522	050117	051105	041040		
20992	126530	052111	020123	047111		
20993	126536	051440	052514	051040		
20994	126544	043505	051511	042524		
20995	126552	051522	000			
20996	126555	124	040522	051516	EM103:	,ASCII2 /TRANSMIT INTERRUPT DOES NOT CLEAR XCSR<0> /
20997	126562	044515	020124	047111		
20998	126570	042524	051122	050125		
20999	126576	020124	047504	051505		
21000	126604	047040	052117	041440		
21001	126612	042514	051101	054040		
21002	126620	051503	036122	033460		
21003	126626	000076				
21004	126630	042522	042503	053111	EM104:	,ASCII2 /RECEIVE INTERRUPTS DON'T CLEAR RCSR<0> /
21005	126636	020105	047111	042524		
21006	126644	051122	050125	051524		
21007	126652	042040	047117	052047		
21008	126660	041440	042514	051101		
21009	126666	051040	051503	036122		
21010	126674	033460	000076			
21011	126700	051102	040505	020113	EM105:	,ASCII2 /BREAK CONDITION DOES NOT SET RBUF PROPERLY /
21012	126706	047503	042116	052111		
21013	126714	047511	020116	042504		
21014	126722	051505	047040	052117		
21015	126730	051440	052105	051040		

CORDARO RD011 H CLUSTER MAC111 30(1046) 05 APR 84 16:45 PAGE 405
 CORDARO P11 05 APR 84 16:45 GLOBAL ERROR MESSAGE'S

SEQ 0404

21016	126736	052502	020106	051120	
21017	126744	050117	051105	054514	
21018	126752	000			
21019	126753	122	052502	020106	FM106: .ASCIIZ /RBUF - 15 11- WASN'T CLEARED ON NEXT CHARACTER/
21020	126760	030474	026465	030461	
21021	126766	020076	040527	047123	
21022	126774	052047	041440	042514	
21023	127002	051101	042105	047440	
21024	127010	020116	042516	052130	
21025	127016	041440	040510	040522	
21026	127024	052103	051105	000	
21027	127031	123	052514	044440	FM107: .ASCIIZ /SQU INTERRUPTS DON'T HAPPEN/
21028	127036	052116	051105	052522	
21029	127044	052120	020123	047504	
21030	127052	023516	020124	040510	
21031	127060	050120	042516	000116	
21032	127066	051105	047522	020122	FM110: .ASCIIZ /ERROR IN WRITING TO SQU REGISTERS/
21033	127074	047111	053440	044522	
21034	127102	044524	043516	052040	
21035	127110	020117	046123	020125	
21036	127116	042522	044507	052123	
21037	127124	051105	000123		
21038	127130	044506	051522	020124	FM111: .ASCIIZ /FIRST CHARACTER WAS NOT OVERRUN BY THE SECOND/
21039	127136	044103	051101	041501	
21040	127144	042524	020122	040527	
21041	127152	020123	047516	020124	
21042	127160	053117	051105	052522	
21043	127166	020116	054502	052040	
21044	127174	042510	051440	041505	
21045	127202	047117	000104		
21046	127206	053117	051105	052522	FM112: .ASCIIZ /OVERRUN CONDITION DOES NOT SET PROPER BITS IN RBUF/
21047	127214	020116	047503	042116	
21048	127222	052111	047511	020116	
21049	127230	047504	051505	047040	
21050	127236	052117	051440	052105	
21051	127244	050040	047522	042520	
21052	127252	020122	044502	051524	
21053	127260	044440	020116	041122	
21054	127266	043125	000		
21055	127271	117	042526	051122	FM113: .ASCIIZ /OVERRUN BITS WERE NOT CLEARED ON THE NEXT CHARACTER
21056	127276	047125	041040	052111	
21057	127304	020123	042527	042527	
21058	127312	047040	052117	041440	
21059	127320	042514	051101	042105	
21060	127326	047440	020116	044124	
21061	127334	020105	042516	052130	
21062	127342	041440	040510	040522	
21063	127350	052103	051105	000	
21064	127355	105	051122	051117	FM114: .ASCIIZ /ERROR ON XCSR, PXX
21065	127362	047440	020116	041530	
21066	127370	011123	031074	000036	
21067	127376	051105	047522	020122	FM115: .ASCIIZ /ERROR IN TAG STORE FROM STANDALONE MODE/
21068	127404	047111	052040	043501	
21069	127412	051440	047524	042527	
21070	127420	043040	047527	020117	
21071	127426	052123	047101	040504	

0165

CORDBAR RD011 H CLUSTER MAC111 50(1046) 05 APR 84 18:45 PAGE 406
COKDAB.P11 05 APR 84 18:45 GLOBAL ERROR MESSAGE'S

SEQ 0405

21072	127434	047514	042516	046440	
21073	127442	042117	000105		
21074	127446	046504	020101	040524	EM116: ,ASCII/ /DMA TAG PARITY DOES NOT GENERATE PROPER RESPONSE/
21075	127454	020107	040520	044522	
21076	127462	054524	042040	047517	
21077	127470	020123	047516	020124	
21078	127476	042507	042516	040522	
21079	127504	042524	050040	047522	
21080	127512	042520	020122	042522	
21081	127520	050123	047117	047523	
21082	127526	000			
21083	127537	115	042523	036122	EM117: ,ASCII/ /MSER<13>NOT SET IN STANDALONE MODE/
21084	127534	031461	047076	052117	
21085	127542	051440	052105	044440	
21086	127550	020116	052123	047101	
21087	127556	040504	047514	042516	
21088	127564	046440	042117	000105	
21089	127572	046504	020101	051127	EM120: ,ASCII/ /DMA WRITE HITS DON'T INVALIDATE CACHE/
21090	127600	052111	020105	044510	
21091	127606	051524	042040	047117	
21092	127614	052047	044440	053116	
21093	127622	046101	042111	052101	
21094	127630	020105	040503	044103	
21095	127636	000105			
21096	127640	047111	041040	047514	EM121: ,ASCII/ /IN BLOCK MODE ON WRITE DMA HITS NOT EVERYTHING IS INVALIDATED/
21097	127646	045503	046440	042117	
21098	127654	020105	047117	053440	
21099	127662	044522	042524	042040	
21100	127670	040515	044040	052111	
21101	127676	020123	047516	020124	
21102	127704	053105	051105	052131	
21103	127712	044510	043516	044440	
21104	127720	020123	047111	040526	
21105	127726	044514	040504	042524	
21106	127734	000104			
21107	127736	042522	042101	042040	EM122: ,ASCII/ /READ DMA HIT IS WRONG
21108	127744	040515	044040	052111	
21109	127752	044440	020123	051127	
21110	127760	047117	000107		
21111	127764	051105	047522	020122	EM123: ,ASCII/ /ERROR IN Q2PBE DMA CYCLES/
21112	127772	047111	050440	031062	
21113	130000	047502	042040	040515	
21114	130006	041440	041531	042514	
21115	130014	000123			
21116	130016	044520	050522	044440	EM124: ,ASCII/ /PIRQ INTERRUPTS DON'T TAKE PRIORITY OVER Q BUS INTERRUPTS
21117	130024	052116	051105	052522	
21118	130032	052117	020123	047504	
21119	130040	023516	020124	040524	
21120	130046	042513	050040	044522	
21121	130054	051117	052111	020131	
21122	130062	053117	051105	050440	
21123	130070	041040	051525	044440	
21124	130076	052116	051105	052522	
21125	130104	052120	000123		
21126	130110	047516	05004	053517	EM125: ,ASCII/ /NO POWER DOWN TRAP TO J4 OCCUR
21127	130116	051105	042040	053517	

CORDABO RDJ11 B CLUSTER MAC111 30(1046) 05 APR 84 16:45 PAGE 407
 CORDAB, P11 05-APR 84 16:45 GLOBAL ERROR MESSAGE S

SEQ 0406

21128	130124	020116	051124	050101	
21129	130132	052040	020117	032062	
21130	130140	047440	041503	051125	
21131	130146	000			
21132	130147	105	051122	051117	EM126: .ASCII2 /ERROR DOING Q22BE INTERRUPT/
21133	130154	042040	044517	043516	
21134	130162	050440	031062	042502	
21135	130170	044440	052116	051105	
21136	130176	052522	052120	000123	
21137	130204	051105	047522	020122	EM127: .ASCII2 /ERROR IN OPERATION OF PMG COUNTER/
21138	130212	047111	047440	042520	
21139	130220	040522	044524	047117	
21140	130226	047440	020106	046520	
21141	130234	020107	047503	047125	
21142	130242	042524	000122		
21143	130246	047125	054105	042520	EM130: .ASCII2 /UNEXPECTED TRAP TO 4/
21144	130254	052103	042105	052040	
21145	130262	040522	020120	047524	
21146	130270	032040	000		
21147	130273	105	051122	051117	EM131: .ASCII2 /ERROR WRITING TO LKS<6>/
21148	130300	053440	044522	044524	
21149	130306	043516	052040	020117	
21150	130314	045514	036123	037066	
21151	130322	000			
21152	130323	105	051122	051117	EM132: .ASCII2 /ERROR IN MAINTENANCE REGISTER/
21153	130330	044440	020116	040515	
21154	130336	052111	047105	047101	
21155	130344	042503	051040	043505	
21156	130352	051511	042524	000122	
21157					
21158	130360	052040	051505	004524	DH1: .ASCII1 / TEST ERROR/<15><12>
21159	130366	051105	047522	006522	
21160	130374	012			
21161	130375	040	021440	020011	.ASCII2 / 0 PC
21162	130402	041520	000		
21163	130405	040	042524	052123	DH4: .ASCII1 / TEST ERROR EXPECTED RECEIVED/<15><12>
21164	130412	042411	051122	051117	
21165	130420	042411	050130	052103	
21166	130425	042105	051011	041505	
21167	130434	044505	042526	006504	
21168	130432	012			
21169	130443	040	021440	020011	.ASCII2 / 0 PC DATA DATA
21170	130450	041520	020011	040504	
21171	130456	040524	020040	020040	
21172	130464	042040	052101	000101	
21173	130472	052040	051505	004524	DH5: .ASCII1 / TEST ERROR WITHIN DATA IN DATA IN/<15><12>
21174	130500	051105	047522	004522	
21175	130506	044510	042524	051511	
21176	130514	042011	052101	020101	
21177	130522	047111	042011	052101	
21178	130530	020101	047111	005015	
21179	130536	020040	004443	050040	.ASCII2 / 0 PC REG. CACHE MEMORY
21180	130544	004503	051040	043505	
21181	130552	004456	040503	044103	
21182	130556	004503	042515	047515	
21183	130556	054522	000		

CDPABO RD111 B CLEVER MAC111 50010463 05 APR 84 16:45 PAGE 408
CORDAB, P11 05-APR 84 16:45 GLOBAL ERROR MESSAGES

SEQ 0407

21184	130571	040	042524	052123	DH7:	,ASCII	/	TEST	ERROR	ADDRESS	MSER/	<15><12>
21185	130576	042411	051122	051117								
21186	130604	040411	042104	042522								
21187	130612	051523	046411	042523								
21188	130620	006522	012									
21189	130623	040	021440	020011		,ASCII2	/	#	PC	ACCESSED/		
21190	130630	041520	040411	041505								
21191	130636	051505	042523	000104								
21192	130644	052040	051505	004524	DH24:	,ASCII	/	TEST	ERROR	NUMBER/	<15><12>	
21193	130652	051105	047522	004522								
21194	130660	052516	041115	051105								
21195	130666	005015										
21196	130670	020040	004443	050040		,ASCII2		#	PC	OF HITS/		
21197	130676	004503	043117	044040								
21198	130704	052111	000123									
21199	130710	051105	047522	004522	DH27:	,ASCII		ERROR	ERROR	MSER	HIT/MISS/	<15><12>
21200	130716	051105	047522	004522								
21201	130724	051515	051105	044011								
21202	130732	052111	046457	051511								
21203	130740	006523	012									
21204	130743	040	021440	020011		,ASCII2	/	#	PC			
21205	130750	041520	000									
21206	130753	040	042524	052123	DH41:	,ASCII	/	TEST	ERROR	INSTRUCTION/	<15><12>	
21207	130760	042411	051122	051117								
21208	130766	044411	051516	051124								
21209	130774	041525	044524	047117								
21210	131002	005015										
21211	131004	020040	004443	050040		,ASCII2	/	#	PC	OPCODE/		
21212	131012	004503	047440	041520								
21213	131020	042117	000105									
21214	131024	052040	051505	004524	DH43:	,ASCII	/	TEST	ERROR	EXPECTED RECEIVED CACHE/	<15><12>	
21215	131032	051105	047522	004522								
21216	131040	054105	042520	052103								
21217	131046	004504	042522	042507								
21218	131054	053111	004504	040503								
21219	131062	044103	006505	012								
21220	131067	040	021440	020011		,ASCII2	/	#	PC	DATA DATA LOCATION/		
21221	131074	041520	020011	040504								
21222	131102	040524	020011	040504								
21223	131110	040524	046011	041517								
21224	131116	052101	047511	000116								
21225	131124	052040	051505	004524	DH47:	,ASCII	/	TEST	ERROR	ADDRESS ADDRESS/	<15><12>	
21226	131132	051105	047522	004522								
21227	131140	042101	051104	051505								
21228	131146	004523	042101	051104								
21229	131154	051505	006523	012								
21230	131161	040	021440	020011		,ASCII2		#	PC	<21 16> <15 0>		
21231	131166	041520	036011	030462								
21232	131174	030455	037066	020011								
21233	131202	030474	026465	037060								
21234	131210	000										
21235	131211	040	042524	052123	DH65:	,ASCII	/	TEST	ERROR	PRIORITY/	<15><12>	
21236	131216	042411	051122	051117								
21237	131224	050011	044522	051117								
21238	131232	052111	006531	012								
21239	131237	040	021440	020011		,ASCII2	/	#	PC	LEVEL		

CORDARO KDJ11 B CLUSTER MACY11 30(1046) 05 APR 84 16:45 PAGE 409
CORDAR,P11 05-APR-84 16:45 GLOBAL ERROR MESSAGES

SEQ 0408

```

21240 131244 041520 046011 053105
21241 131252 046105 000
21242 131255 040 042524 052123 DH72: ,ASCII / TEST ERROR ADDRESS/ <15><12>
21243 131262 042411 051122 051117
21244 131270 040411 042104 042522
21245 131276 051523 005015
21246 131302 020040 004443 050040 ,ASCII/ / # PC FAILED/
21247 131310 004503 040506 046111
21248 131316 042105 000
21249 131321 040 042524 052123 DH105: ,ASCII / TEST ERROR RBUF / <15><12>
21250 131326 042411 051122 051117
21251 131334 051011 052502 005506
21252 131342 012
21253 131343 040 021440 020011 ,ASCII/ / # PC
21254 131350 041520 000
21255 131353 040 042524 052123 DH115: ,ASCII / TEST ERROR MSER ADDRESS/ <15><12>
21256 131360 042411 051122 051117
21257 131366 046411 042523 004522
21258 131374 042101 051104 051505
21259 131402 006523 012
21260 131405 040 021440 020011 ,ASCII/ / # PC ACCESSED/
21261 131412 041520 004440 040411
21262 131420 041503 051505 042523
21263 131426 000104
21264
21265 131430 001162 001116 000000 DT1: ,EVEN
21266 131436 001162 001116 000001 DT4: ,WORD $TMP1,$ERRPC,0
21267 131444 177746 000000 ,WORD $TMP1,$ERRPC,R1,CCR,0
21268 131450 001162 001116 000002 DT5: ,WORD $TMP1,$ERRPC,R1,R1,$GDDAT,0
21269 131456 000001 001124 000000
21270 131464 001162 001116 001122 DT7: ,WORD $TMP1,$ERRPC,$BDADR,MSER,0
21271 131472 177744 000000
21272 131476 001162 001116 001124 DT14: ,WORD $TMP1,$ERRPC,$GDDAT,TESTPC,0
21273 131504 003122 000000
21274 131510 001162 001116 001124 DT17: ,WORD $TMP1,$ERRPC,$GDDAT,RECDAT,0
21275 131516 110504 000000
21276 131522 001162 001116 000003 DT24: ,WORD $TMP1,$ERRPC,R3,0
21277 131530 000000
21278 131532 001162 001116 177744 DT27: ,WORD $TMP1,$ERRPC,MSER,R3,0
21279 131540 000003 000000
21280 131544 001162 001116 001124 DT35: ,WORD $TMP1,$ERRPC,$GDDAT,MSER,0
21281 131552 177744 000000
21282 131556 001162 001116 001126 DT41: ,WORD $TMP1,$ERRPC,$BDADR,0
21283 131564 000000
21284 131566 001162 001116 000001 DT43: ,WORD $TMP1,$ERRPC,R1,RECDAT,$BDADR,0
21285 131574 110504 001122 000000
21286 131602 001162 001116 172354 DT47: ,WORD $TMP1,$ERRPC,KIDPAR6,$BDADR,0
21287 131610 001122 000000
21288 131614 001162 001116 000001 DT50: ,WORD $TMP1,$ERRPC,R1,$BDADR,0
21289 131622 001122 000000
21290 131626 001162 001116 001124 DT51: ,WORD $TMP1,$ERRPC,$GDDAT,PER,0
21291 131634 177522 000000
21292 131640 001162 001116 001124 DT52: ,WORD $TMP1,$ERRPC,$GDDAT,BCSR,0
21293 131646 177520 000000
21294 131652 001162 001116 001124 DT64: ,WORD $TMP1,$ERRPC,$GDDAT,CXSEL,0
21295 131660 002702 000000
```

FILE

CORDARO RDJ11 B CLUSTER MAC111 30010463 05 APR 84 16:45 PAGE 410
CORDAR,P11 05 APR 84 16:45 GLOBAL ERROR MESSAGE

SEQ 0409

21296	131664	001162	001116	001124	DT65:	.WORD	\$TMP1,\$ERRPC,\$GDDAT,0
21297	131672	000000					
21298	131674	001162	001116	001124	DT75:	.WORD	\$TMP1,\$ERRPC,\$GDDAT,\$BDDAT,0
21299	131702	001126	000000				
21300	131706	001162	001116	127562	DT105:	.WORD	\$TMP1,\$ERRPC,RBUF,0
21301	131714	000000					
21302	131716	001162	001116	001126	DT115:	.WORD	\$TMP1,\$ERRPC,\$BDDAT,KIPAR6,\$BDADR,0
21303	131724	127564	001122	000000			
21304	131732	001162	001122	000000	DT130:	.WORD	\$TMP1,\$BDADR,0


```

21505 .SBIH MODIFIED ERROR MESSAGE TIMEOUT ROUTINE
21506 ;*****
21507 ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
21508 ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
21509 ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
21510 ;*
21511 ;*THE ONLY DIFFERENCE BETWEEN THIS ROUTINE AND THE ORIGINAL "$ERRTP" FROM
21512 ;*SYSTEMAC IS THAT YOU CAN PASS INFORMATION IN GENERAL PURPOSE REGISTERS TO THIS
21513 ;*ROUTINE. THE GENERAL PURPOSE REGISTERS USED ARE TO BE SPECIFIED IN DI*
21514 ;*FORMAT. RO SHOULD NOT BE USED.
21515
21516 131740 005037 001162 001162
21517 131740 113737 001102 001162
21518 131744 104401 001175
21519 131752 010046
21520 131756 005000
21521 131760 153700 001114
21522 131762 001004
21523 131766 013746 001116
21524 131770 104402
21525 131774 000426
21526 131776 005300
21527 132000 006300
21528 132002 006300
21529 132004 006300
21530 132006 006300
21531 132010 062700 001324
21532 132014 012037 132024
21533 132020 001404
21534 132022 104401
21535 132024 000000
21536 132026 104401 001175
21537 132032 012037 132042
21538 132036 001404
21539 132040 104401
21540 132042 000000
21541 132044 104401 001175
21542 132050 011000
21543 132052 001004
21544 132054 012600
21545 132056 104401 001175
21546 132062 000207
21547 132064
21548 132066 021027 000005
21549 132070 101021
21550 132072 042737 000700 132126
21551 132100 011037 001160
21552 132104 000437 001160
21553 132110 006237 001160
21554 132114 006237 001160
21555 132120 013737 001160 132126
21556 132122 010046
21557 132130 005720

```

```

ERTYPE:
CLR $TMP1 ;JUST CLEAR IT
MOVB $STNM,$TMP1 ;STORE TEST NUMBER
TYPE , $CRUF ;"CARRIAGE RETURN" & "LINE FEED"
MOV RO, -(SP) ;SAVE RO
CLR RO ;PICKUP THE ITEM INDEX
BISB @ $ITEMB, RO
BNE 1$ ;IF ITEM NUMBER IS ZERO, JUST
;TYPE THE PC OF THE ERROR
MOV $ERRPC, -(SP) ;SAVE $ERRPC FOR TIMEOUT
;ERROR ADDRESS
;GO TYPE--OCTAL ASCII(ALL DIGITS)
BR 1$ ;GET OUT
DEC RO ;ADJUST THE INDEX SO THAT IT WILL
;WORK FOR THE ERROR TABLE
ASL RO
ASL RO
ASL RO
ADD @ $ERRTB, RO ;FORM TABLE POINTER
MOV (RO), 2$ ;PICKUP "ERROR MESSAGE" POINTER
BEQ 3$ ;SKIP TIMEOUT IF NO POINTER
;WORD 0
;TYPE , $CRUF ;"CARRIAGE RETURN" & "LINE FEED"
MOV (RO), 4$ ;PICKUP "DATA HEADER" POINTER
BEQ 5$ ;SKIP TIMEOUT IF 0
;TYPE THE "DATA HEADER"
;WORD 0
;TYPE , $CRUF ;"CARRIAGE RETURN" & "LINE FEED"
MOV (RO), RO ;PICKUP "DATA TABLE" POINTER
BNE 7$ ;GO TYPE THE DATA
MOV (RO), RO ;RESTORE RO
TYPE , $CRUF ;"CARRIAGE RETURN" & "LINE FEED"
RTS PC ;RETURN
7$:
CMP (RO), 0$ ;GENERAL PURPOSE REGISTER?
BNE 9$ ;IF NOT, GO TYPE DATA
BIC $BIT8:$BIT7:$BIT6, 8$ ;CLEAR BITS FOR SOURCE REGISTER
MOV (RO), $TMP0 ;SAVE (RO)
SWAB $TMP0 ;GET REGISTER NUMBER TO HIGH BITS
ASR $TMP0 ;GET REGISTER NUMBER TO BITS 8-6
ASR $TMP0
RCL $TMP0, 9$ ;SET BITS IN MOV INSTRUCTION
;ACCORDING TO REGISTER NUMBER
MOV RO, -(SP) ;MOVE CONTENT OF REGISTER TO STACK
TYPE (RO) ;ADVANCE POINTER

```

21361	132132	000401		BR	10\$	::GO TYPE
21362	132134	013046		9\$: MOV	@(RO)+, -(SP)	::IF NOT GPR, SAVE @(RO)+ FOR TYPEOUT
21363	132136	104402		10\$: TYPEOC		::GO TYPE OCIAL ASCII (ALL DIGITS)
21364	132140	005710		TEST	(RO)	::IS THERE ANOTHER NUMBER?
21365	132142	001744		BEQ	6\$	::BR IF NO
21366	132144	104401	132152	TYPE	,11\$	::TYPE TWO(2) SPACES
21367	132150	000745		BR	7\$	
21368	132152	020040	000	11\$: ASCII2	/ /	::TWO(2) SPACES
21369		132156		LEVEN		

BTTL GLOBAL SUBROUTINES SECTION

::
 :: THE GLOBAL SUBROUTINES SECTION CONTAINS THE SUBROUTINES
 :: THAT ARE USED IN MORE THAN ONE TEST.
 ::

::
 :: FUNCTIONAL DESCRIPTION:
 :: SUBROUTINE TO INITIALIZE ALL THE MMU REGISTERS

:: INPUTS: NONE

:: OUTPUTS: NONE

:: SUBORDINATE ROUTINES USED: LOAD PARS
 :: LOAD PDRS

:: FUNCTIONAL SIDE EFFECTS: NONE

:: CALLING SEQUENCE: JSR PC,INITMM

21394	132156	012701	172240	INITMM: MOV	\$172240,R1	::BASE ADDRESS OF SIPARS
21395	132162	004737	132320	JSR	PC, LDPARS	
21396	132166	012701	172260	MOV	\$172260,R1	::BASE ADDRESS OF SDPARS
21397	132172	004737	132320	JSR	PC, LDPARS	
21398	132176	012701	172340	MOV	\$172340,R1	::BASE ADDRESS OF KIPARS
21399	132202	004737	132320	JSR	PC, LDPARS	
21400	132206	012701	172360	MOV	\$172360,R1	::BASE ADDRESS OF KDPARS
21401	132212	004737	132320	JSR	PC, LDPARS	
21402	132216	012701	172640	MOV	\$172640,R1	::BASE ADDRESS OF UIPARS
21403	132222	004737	132320	JSR	PC, LDPARS	
21404	132226	012701	172660	MOV	\$172660,R1	::BASE ADDRESS OF UDPARS
21405	132232	004737	132320	JSR	PC, LDPARS	
21406	132236	012701	172600	MOV	\$172600,R1	::BASE ADDRESS OF UIPDRS
21407	132242	004737	132350	JSR	PC, LDPDRS	
21408	132246	012701	172620	MOV	\$172620,R1	::BASE ADDRESS OF UDPDRS
21409	132252	004737	132350	JSR	PC, LDPDRS	
21410	132256	012701	172300	MOV	\$172300,R1	::BASE ADDRESS OF KIPDRS
21411	132262	004737	132350	JSR	PC, LDPDRS	
21412	132266	012701	172320	MOV	\$172320,R1	::BASE ADDRESS OF KDPDRS
21413	132272	004737	132350	JSR	PC, LDPDRS	
21414	132276	012701	172200	MOV	\$172200,R1	::BASE ADDRESS OF SIPDRS
21415	132302	004737	132350	JSR	PC, LDPDRS	
21416	132306	012701	172220	MOV	\$172220,R1	::BASE ADDRESS OF SDPDRS

16

CORPABO RDJ11 B CLUSTER MACY11 30(1046) 05 APR 84 16:45 PAGE 415
CORPAB.P11 05 APR 84 16:45 GLOBAL SUBROUTINE'S SECTION

SEQ 0412

21417 132312 004737 132350
21418 132316 000207

JSR PC, LDPDR5
RTN PC

;RE TURN

CORDBO RDJ11-B CLUSTER MACY11 30(1046) 05 APR 84 16:45 PAGE 414
 CORDAB.P11 05-APR-84 16:45 GLOBAL SUBROUTINES SECTION

SEQ 0413

```

21419
21420
21421
21422
21423
21424
21425
21426
21427
21428
21429
21430
21431
21432
21433
21434
21435
21436
21437
21438
21439
21440
21441
21442
21443
21444
21445
21446
21447
21448
21449
21450
21451
21452
21453
21454
21455
21456
21457
21458
21459 132320 012702 000006
21440 132324 005003
21441 132326 010321
21442 132330 062703 000200
21443 132334 077204
21444 132336 012721 002000
21445 132340 012711 177600
21446 132346 000207

***
; FUNCTIONAL DESCRIPTION:
;   SUBROUTINE TO INITIALIZE ALL THE MMU PAGE ADDRESS REGISTERS (PARS).
;   THIS ROUTINE WILL INITIALIZE 8 PARS STARTING AT A BASE ADDRESS
;   SUPPLIED BY THE CALLING ROUTINE. PARS 0-5 WILL BE MAPPED FROM
;   ADDRESS 0 TO ADDRESS 137777 (0-24K). PAR 6 WILL BE MAPPED FROM
;   ADDRESS 200000 TO 217777 AND PAR 7 WILL BE MAPPED TO THE I/O
;   PAGE.
;
; INPUTS:
;   R1 CONTAINS THE BASE ADDRESS OF THE NEXT 8 PARS TO BE INITIALIZED
;
; OUTPUTS: NONE
;
; SUBORDINATE ROUTINES USED: NONE
;
; FUNCTIONAL SIDE EFFECTS: NONE
;
; CALLING SEQUENCE:      JSR      PC,LDPARS
LDPARS: MOV      #6,      R2
; LET LOOP COUNTER COUNT FIRST 6 PARS
CLR      R3
; INITIALIZE INDEX VALUE
1$: MOV      R3,      (R1)+
; LOAD PARS
ADD      #200,      R3
; INDEX IN 4K INCREMENTS
SOB      R2,      1$
; LOAD FIRST SIX PARS
MOV      #2000,      (R1)+
; LET PAR6 MAP TO 200000
MOV      #177600,(R1)+
; LET PAR7 MAP TO I/O PAGE
RTS      PC
; RETURN

```

COKDABO KDJ11 B CLUSTER MACY11 30(1046) 05 APR 84 16:45 PAGE 415
 COKDAB.P11 05-APR-84 16:45 GLOBAL SUBROUTINES SECTION

SEQ 0414

```

21447
21448
21449
21450
21451
21452
21453
21454
21455
21456
21457
21458
21459
21460
21461
21462
21463
21464
21465
21466
21467
21468 132350 012702 000006
21469 132354 012721 177406
21470 132360 077203
21471 132362 012721 077406
21472 132366 012711 077406
21473 132372 000207

; **
; FUNCTIONAL DESCRIPTION:
;   SUBROUTINE TO INITIALIZE ALL THE MMU PAGE DESCRIPTOR REGISTERS (PDRS).
;   THIS ROUTINE WILL INITIALIZE 8 PDRS STARTING AT A BASE ADDRESS
;   SUPPLIED BY THE CALLING ROUTINE. PDRS 0-5 WILL BE INITIALIZED TO
;   4K READ/WRITE BYPASS AND PDRS 6 AND 7 WILL BE INITIALIZED TO
;   4K READ/WRITE NO BYPASS.
;   NOTE: THERE IS NO NEED TO BYPASS ON I/O PAGE REFERENCES BECAUSE
;   THE CACHE DOES NOT ALLOCATE ANY OF THESE REFERENCES.

; INPUTS:
;   R1 CONTAINS THE BASE ADDRESS OF THE NEXT 8 PDRS TO BE INITIALIZED

; OUTPUTS: NONE

; SUBORDINATE ROUTINES USED: NONE

; FUNCTIONAL SIDE EFFECTS: NONE

. CALLING SEQUENCE:      JSR      PC,LDPARS

LDPDRS: MOV      #6,      R1
1$:      MOV      #177406,(R1)+
        SOB      R2,      1$
        MOV      #77406,(R1)+
        MOV      #77406,(R1)
        RTS      PC

; LET LOOP COUNTER COUNT FIRST 6 PDRS
; LOAD PDRS WITH 4K READ/WRITE BYPASS
; LOAD FIRST SIX PDRS
; LET PARS BE 4K READ/WRITE NO BYPASS
; LET PAR7 BE 4K READ/WRITE NO BYPASS ALSO
; RETURN

```



```

21474      ***
21475      ; FUNCTIONAL DESCRIPTION:
21476      ;   SUBROUTINE TO HANDLE PARITY ERROR ABORTS FROM THE RAM STORE RAM TESTS.
21477
21478      ; INPUTS:
21479      ;   MEMORY SYSTEM ERROR REGISTER CONTAINS BITS INDICATING FAILURE
21480
21481      ; OUTPUTS: NONE
21482
21483      ; SUBORDINATE ROUTINES USED: NONE
21484
21485      ; FUNCTIONAL SIDE EFFECTS: NONE
21486
21487      ; CALLING SEQUENCE: CALLED BY PARITY ABORT
21488      ;   MOV     00114, SLOC00 ;SAVE CONTENTS OF PARITY ABORT VECTOR
21489      ;   MOV     00SPAR, 00114 ;LET VECTOR POINT TO PARITY ABORT ROUTINE
21490
21491      ;   (CACHE PARITY ERROR OCCURS)
21492
21493      132374 011637 001122      RAMPAR: MOV     (SP), 0BDADR      ;STOR ADDRESS TRAPPED
21494      132400 032737 000100 177744      BIT     0BIT06, MSER      ;IF LOW BYTE PARITY ERROR
21495      132406 001401              BEQ     11              ;THEN
21496      132410 104007              ERROR    17              ;ERROR
21497      132412 032737 000200 177744 11:   BIT     0BIT07, MSER      ;IF HIGH BYTE PARITY ERROR
21498      132420 001401              BEQ     21              ;THEN
21499      132422 104007              ERROR    17              ;ERROR
21500      132424 032737 000040 177744 21:   BIT     0BIT05, MSER      ;IF TAG PARITY ERROR
21501      132432 001401              BEQ     31              ;THEN
21502      132434 104007              ERROR    17              ;ERROR
21503      132436 005037 177744      31:   CLR     MSER          ;INITIALIZE MSER AFTER ERROR
21504      132442 000002              RTI              ;RETURN
21505      132444 005237 002702      LKSINT: INC     LKSEL          ;INCREMENT FLAG
21506      132450 000002              RTI
21507
21508      ; SBTTL Q22BE SIZE ROUTINE
21509      ; THIS ROUTINE WILL AUTOSIZE FOR UP TO TWO Q22 BUS EXERCISERS. IF NONE
21510      ; FOUND LOCATIONS CSR1 AND CSR12 WILL BE LEFT ZEROES. THIS ROUTINE WILL
21511      ; ONLY RUN IN NOT L&D MODE.
21512
21513      132452 032737 001000 177750 Q22S12: BIT     0BIT09, MAIREG      ;UNIBUS SYSTEM?
21514      132460 001401              BEQ     11              ;IF NOT, ADVANCE TO ROUTINE
21515      132462 000207              RTS     PC              ;OTHERWISE, RETURN
21516
21517      ; PREPARE TO DO SIZING
21518
21519      132464 013701 000004      11:   MOV     ERRVEC, R1          ;STORE TIMEOUT VECTOR
21520      132470 012737 132641 000004      MOV     011, ERRVEC      ;POINT NEW TO PROGRAM
21521      132476 012737 000340 000006      MOV     0340, ERRVEC+1    ;AT PRIORITY 7
21522      132504 005037 001160      CLR     $TMP0          ;CLEAR Q22BE COUNTER
21523      132510 012702 170000      MOV     0170000, R2         ;FIRST POSSIBLE ADDRESS
21524      132514 012703 000510      MOV     0510, R3          ;VECTOR FOR IT
21525      132520 000404      RR      31              ;TRY THOSE VALUES
21526
21527      ; NOW DO ACTUAL SIZING
21528
21529      132522 062702 000020      21:   ADD     020, R2          ;GET CSR FOR NEXT Q22BE
  
```

01

SEQ 0416

21530	132526	062703	000004		ADD	04,R3			
21531	132532	005712			TST	(R2)			;GET VECTOR FOR NEXT ONE
21532									;TRY TO ACCESS CSR
21533									
21534									; IF NO TIMEOUT, STORE EXISTING ADDRESSES TO REGISTERS
21535	132534	005737	001160		TST	\$TMP0			;FIRST Q22BE FOUND?
21536	132540	001010			BNE	4\$			;IF SECOND, BRANCH
21537	132542	012705	002644		MOV	0CSR1,R5			;START WITH CSR1 FOR 1ST
21538	132546	010237	002662		MOV	R2,SIMGOA			;SIMULTANEOUS GO
21539	132552	062737	000016	002662	ADD	016,SIMGOA			;ADDRESS
21540	132560	000402			BR	5\$			;BRANCH TO INITIALISE
21541	132562	012705	002664		MOV	0CSR12,R5			;START WITH CSR12 FOR 2ND
21542	132566	012704	000004		MOV	04,R4			;INITIALISE 5 REGISTERS
21543	132572	010215			MOV	R2,(R5)			;INITIALISE CSR1
21544	132574	011565	000002		MOV	(R5),2(R5)			;STORE TO NEXT ONE
21545	132600	005725			TST	(R5)			;GET NEXT ADDRESS
21546	132602	062715	000002		ADD	02,(R5)			;GET ADDRESS, POINT NEXT
21547	132606	077406			SQB	R4,6\$			;DO FOR NEXT 4 REGISTERS
21548	132610	010365	000002		MOV	R3,2(R5)			;STORE INTERRUPT VECTOR
21549	132614	010365	000004		MOV	R3,4(R5)			;AND PRIORITY
21550	132620	062765	000002	000004	ADD	02,4(R5)			
21551	132626	005237	001160		INC	\$TMP0			;COUNT Q22BE'S
21552	132632	022737	000002	001160	CMP	02,\$TMP0			;TWO FOUND?
21553	132640	001406			BEQ	9\$			;IF SO, STOP SIZING
21554	132642	000402			BR	8\$			;OTHERWISE, CONTINUE SIZING
21555									
21556									; ON TIMEOUT TRY TO LOOK AT NEXT ADDRESS RANGE
21557									
21558	132644	005726			TST	(SP)			;RESTORE STACK FROM
21559	132646	005726			TST	(SP)			;TIMEOUT
21560	132650	022702	170160		CMP	0170160,R2			;AT THE LAST POSSIBLE?
21561	132654	001322			BNE	2\$			;IF NOT, BRANCH
21562	132656	005737	002644		TST	CSR1			;IF FOUND?
21563	132662	001402			BEQ	10\$			;IF NONE, BRANCH
21564	132664	104401	132676		TYPE	,ONQ22			;TYPE FOUND
21565	132670	010137	000004		MOV	R1,FRRVEC			;RESTORE TIMEOUT VECTOR
21566	132674	000207			RTS	PC			;RETURN
21567									
21568	132676	006412	031121	041062	ONQ22:	.ASCIZ	<12><15>/Q22BE USED DURING TESTING/		
21569	132704	020105	051525	042105					
21570	132712	042040	051125	047111					
21571	132720	020107	042524	052123					
21572	132726	047111	000107						
21573									
21574									
21575									
21576									
21577									
21578									
21579	132732	005013			Q22INT:	CLR	(R3)		;CLEAR TRANSFER TYPE IN CSR1
21580	132734	052710	000001		BIS	0BIT00,(R0)			;ZERO DONE
21581	132740	011063	000002		MOV	(R0),2(R3)			;SET PRIORITY IN CSR2
21582	132744	042710	000001		BIC	0BIT00,(R0)			;PREPARE TO SET DONE
21583	132750	000207			RTS	PC			
21584									
21585									

.SBTTL DMATRQ DATO CYCLE THRU Q22BE

CORDDABO RDJ11 B CLUSTER MAC111 30(1046) 05 APR 84 16:45 PAGE 418
COKDAB.P11 05-APR 84 16:45 DMATRNDATO CYCLE THRU Q22BE

SEQ 0417

```

21586 ;THIS ROUTINE PERFORMS DATO FROM A LOCATION TEMP THRU THE FIRST
21587 ;FOUND Q22BE STARTING AT LOCATION @CSR1. RO HAS 0 IF ONLY 1 TRANSFER IS
21588 ;TO BE PERFORMED. OTHERWISE 16 BLOCK MODE TRANSFERS ARE TO BE PERFORMED.
21589 ;IN THE LATTER CASE ADDRESS AND WORD COUNT HAS TO BE LOADED BEFORE.
21590
21591 132752 012777 012525 047674 DMATRND: MOV @12525,@DATA ;DATA USED
21592 132760 005700 TST RO ;DO 1 WORD?
21593 132762 001404 BEQ 1$ ;IF YES, BRANCH
21594 132764 012777 001001 047654 MOV @BIT09!BIT00,@CSR2 ;BLOCK MODE, GO
21595 132772 000414 BR 2$ ;BRANCH TO DO IT
21596 132774 012777 001601 047642 1$: MOV @1601,@CSR1 ;RESET LATENCY COUNT,DATO
21597 133002 012777 002710 047640 MOV @TEMP,@BA ;LOAD DMA ADDRESS
21598 133010 012777 177777 047634 MOV @177777,@WC ;DO 1 WORD
21599 133016 012777 000001 047622 MOV @BIT00,@CSR2 ;DO IT
21600 133024 105777 047616 2$: TSTB @CSR2 ;DMA DONE?
21601 133030 100375 BPL 2$ ;WAIT TILL DONE
21602 133032 000207 3$: RTS PC ;RETURN FROM SUBROUTINE
21603
21604 ;SBTTL DMARD DATI THRU Q22BE
21605 ;THIS ROUTINE PERFORMS DATI CYCLE THRU Q22BE IN EITHER BLOCK MODE OR A SINGLE
21606 ;TRANSFER MODE. MEMORY LOCATION USED IS TEMP. RO IS ZERO FOR SINGLE TRANSFER
21607
21608 133034 012777 002710 047606 DMARD: MOV @TEMP,@BA ;LOAD DMA ADDRESS
21609 133042 005700 TST RO ;DO 1 WORD?
21610 133044 001412 BEQ 1$ ;IF YES, BRANCH
21611 133046 012777 001507 047570 MOV @1507,@CSR1 ;16 DATIB
21612 133054 012777 177770 047570 MOV @177770,@WC ;DO 8 WORD
21613 133062 012777 001001 047556 MOV @BIT09!BIT00,@CSR2 ;BLOCK MODE GO
21614 133070 000411 BR 2$ ;GO CHECK
21615 133072 012777 001407 047544 1$: MOV @1407,@CSR1 ;RESET LATENCY COUNT,DATI
21616 ;LOAD NEW DATA TO DATA R.
21617 133100 012777 177777 047544 MOV @177777,@WC ;DO 1 WORD
21618 133106 012777 000001 047532 MOV @BIT00,@CSR2 ;DO IT
21619 133114 105777 047526 2$: TSTB @CSR2 ;DMA DONE?
21620 133120 100375 BPL 2$ ;WAIT TILL DONE
21621 133122 000207 3$: RTS PC ;RETURN FROM SUBROUTINE
21622
21623
21624
21625 133124 011637 001122 TOUT: MOV (SP),@BDADR ;STORE TRAPPED PC
21626 133130 104130 ERROR @130 ;UNEXPECTED TRAP
21627 133132 000002 RTI
21628
21629
21630 ;MMU GLOBAL SUBROUTINES
21631 ;
21632 ;
21633 ;ROUTINE TO INITIALIZE MEMORY MANAGEMENT
21634 ;
21635 133134 010046 MMU: MOV RO,(SP) ;SAVE CONTENTS OF REGISTERS
21636 133136 010146 MOV R1,(SP)
21637 133140 010246 MOV R2,(SP)
21638 133142 012700 177600 MOV @177600,RO
21639 133146 004737 133234 JSR PC,PDR ;INIT I AND D USER PDR'S
21640 133152 004737 133256 JSR PC,PAR ;INIT I USER PARS
21641 133156 004737 133256 JSR PC,PAR ;INIT D USER PARS

```

21642	133162	012700	172200		MOV	0172200,RO	
21643	133166	004737	133234		JSR	PC,PDR	;INIT I AND D SUP PDR'S
21644	133172	004737	133256		JSR	PC,PAR	;INIT I SUP PAR'S
21645	133176	004737	133256		JSR	PC,PAR	;INIT D SUP PAR'S
21646	133202	004737	133234		JSR	PC,PDR	;INIT I AND D KER PDR'S
21647	133206	004737	133256		JSR	PC,PAR	;INIT I KER PAR'S
21648	133212	004737	133256		JSR	PC,PAR	;INIT D KER PAR'S
21649	133216	012737	000027	172516	MOV	027,00172516	;INIT MMR3
21650	133224	012602			MOV	(SP)+,R2	;RESTORE REGISTERS
21651	133226	012601			MOV	(SP)+,R1	
21652	133230	012600			MOV	(SP)+,RO	
21653	133232	000207			RTS	PC	;RETURN
21654							
21655							
21656							
21657	133234	005002					
21658	133236	012720	077406		PDR:	CLR R2	;INIT CNTR
21659	133242	062702	000001		PDR1:	MOV 077406,(RO)+	;INIT PDR
21660	133246	022702	000020			ADD 01,R2	;INCREMENT CNTR
21661	133252	001371				CMP 016,,R2	;ARE WE DONE?
21662	133254	000207				BNE PDR1	;BRANCH IF NOT
21663						RTS PC	;RETURN
21664							
21665							
21666	133256	005001					
21667	133260	010120			PAR:	CLR R1	;SETUP TO INIT PAR
21668	133262	062701	000200		PAR1:	MOV R1,(RO)+	;INIT PAR
21669	133266	022701	001600			ADD 0200,R1	;GET READY FOR NEXT PAR
21670	133272	001372				CMP 01600,R1	;REACHED A PAR?
21671	133274	012720	177600			BNE PAR1	;BRANCH IF NOT
21672	133300	000207				MOV 0177600,(RO)+	;INIT PAR?
21673						RTS PC	;RETURN
21674							
21675							
21676	133302	005205					
21677	133304	000002			ADDTRP:	INC R5	;INCREMENT TIME OUT FLAG
21678						RTI	;RETURN
21679							
21680							
21681	133306	023727	002776	000001	MMUTRP:	CMP FLAG,01	;ARE WE EXPECTING AN ABORT
21682	133314	001401				BEQ 1\$	;YES GO ON
21683	133316	104002				ERROR	;NO GO TO ERROR
21684	133320	010046			1\$:	MOV RO,(SP)	;SAVE CONTENTS OF REG 0
21685	133322	013700	177776			MOV 00177776,RO	;SAVE A COPY OF PSW
21686	133326	072027	177764			ASH 0 14,RO	;LOOK AT BITS<15:14>
21687	133332	020027	000002			CMP RO,0	;WAS PS<15:14>=10
21688	133336	001001				BNE OK	;NO GO ON
21689	133340	000411				BR NOTOK	;YES CHANGE BITS TO 00
21690	133342	013700	177776		OK:	MOV 00177776,RO	;SAVE A COPY OF PSW
21691	133346	072027	000002			ASH 0 14,RO	;LOOK AT BITS<13:12>
21692	133352	072027	177764			ASH 0 14,RO	
21693	133356	020027	000002			CMP RO,0	;WAS PS<13:12>=10
21694	133362	001002				BNE OK1	;NO GO ON
21695	133364	005066	000004		NOTOK:	CLR 4(SP)	;CLEAR ILLEGAL MODE FROM OLD PSW
21696	133370	013737	177574	003004	OK1:	MOV 00177574,SAVMR0	;SAVE A COPY OF MMR0
21697	133376	013737	177574	003006		MOV 00177574,SAVMR1	;SAVE A COPY OF MMR1

CORDABO RDJ11 R CLOSTER MAC111 30(1046) 05 APR 84 16:45 PAGE 420
CORDAB.P11 05-APR 84 16:45 QMARD DATA THRU Q22BF

SEQ 0419

21698	133404	015737	177576	003010	MOV	W0177576,SAVMR2	;SAVE A COPY OF MMR2
21699	133412	005037	177572		CLR	W0177572	;CLEAR ABORT BITS AND TURN MMU OFF
21700	133416	005037	002776		CLR	FLAG	;CLEAR MMU ABORT FLAG
21701	133420	012600			MOV	(SP)+,R0	;RESTORE ORIGINAL CONTENTS OF REG 0
21702	133424	000002			RTI		;RETURN

```

21703      ;FPP COMMON SUBROUTINE'S
21704 133426 012600 WLDTRP: MOV      (SP)+,R0      ;SAVE PC
21705 133430 012605      MOV      (SP)+,R5      ;SAVE STATUS AND RESTORE STACK
21706 133432 104003      ERROR    .5
21707 133434 000110      JMP      (R0)          ;GO BACK INLINE
21708
21709
21710
21711 133436 000000 TRPELG: .WORD    0
21712 133440 000207 ERRFP:  RTS      R7
21713 133442 000207 ERR:    RTS      R7
21714
21715
21716
21717
21718
21719      ;SUBROUTINE DATA VERIFICATION
21720
21721      ; CALLED BY      JSR      R7,DATVER
21722
21723      ;INPUT:          (R4)=EXPECTED DATA
21724                      (R1)=RECEIVED DATA
21725
21726      ;THIS ROUTINE VERIFIES THAT THE 4 CONSECUTIVE WORDS STARTING WITH (R4) ARE
21727      ;EQUAL TO THE FOUR WORDS ADDRESSED BY (R1). THE CONTENTS OF R4, AND R1 ARE NOT
21728      ;DISTURBED.
21729      ;LOCATION "COUNT", IF NOT EQUAL TO 0 SIGNIFIES DATA ERROR
21730      ;IF THE STATUS IS FLOATING MODE, THE LAST TWO BYTES OF RECEIVED
21731      ;ARE SIMPLY CHECKED FOR ZEROS
21732
21733
21734 133444 010446 DATVER: MOV      R4,-(SP)      ;SAVE R4
21735 133446 010146      MOV      R1,-(SP)      ;SAVE R1
21736 133450 012737 000003 003060      MOV      #7,COUNT      ;SET UP ITERATION COUNT
21737 133456 000137 133474      JMP      DAT1
21738
21739 133462 010446 DATVER: MOV      R4,-(SP)      ;SAVE R4
21740 133464 010146      MOV      R1,-(SP)      ;SAVE R1
21741 133466 012737 000005 003060      MOV      #5,COUNT      ;SET UP ITERATION COUNT
21742 133474 005337 003060 DAT1:  DEC      COUNT
21743 133500 001402      BEQ      #3          ;BRANCH IF DONE
21744 133502 001421      CMP      (R4)+,(R1)+
21745 133504 001773      BEQ      DAT1
21746 133506 012601 2$:      MOV      (SP)+,R1      ;RESTORE R1
21747 133510 012604      MOV      (SP)+,R4      ;RESTORE R4
21748 133512 000207      RTS      R7          ;GO BACK TO CALLING ROUTINE
21749      ;IF DATA ERROR, COUNT NE 0

```

```

21750
21751
21752
21753
21754
21755
21756
21757
21758
21759
21760
21761 133514
21762 133514 032737 000100 000052
21763 133522 001026
21764 133524 005037 001102
21765 133530 005037 001164
21766 133534 005237 001206
21767 133540 042737 100000 001206
21768 133546 005327
21769 133550 000001
21770 133552 003022
21771 133554 012737
21772 133556 000001
21773 133560 133550
21774 133562 104401 133627
21775 133566 013746 001206
21776 133572 104405
21777 133574 104401 133624
21778 133600 013700 000042
21779 133604 001405
21780 133606 000005
21781 133610 004710
21782 133612 000240
21783 133614 000240
21784 133616 000240
21785 133620
21786 133620 000137
21787 133622 004562
21788 133624 377 377 000
21789 133627 015 042415 042116
21790 133634 050040 051501 020123
21791 133642 000043
21792
21793
21794
21795
21796
21797
21798
21799
21800
21801
21802
21803
21804
21805

;SBTTL END OF PASS ROUTINE

;*****
;*INCREMENT THE PASS NUMBER ($PASS)
;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
;*TYPE "END PASS XXXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
;*IF THERES A MONITOR GO TO IT
;*IF THERE ISN'T JUMP TO LOOP

$EOP:
    BIT    @BIT06,@052
    BNE    $GET42
    CLR    $TSTNM                ;;ZERO THE TEST NUMBER
    CLR    $TIMES                ;;ZERO THE NUMBER OF ITERATIONS
    INC    $PASS                ;;INCREMENT THE PASS NUMBER
    BIC    @100000,$PASS        ;;DON'T ALLOW A NEG. NUMBER
    DEC    (PC)+                ;;LOOP?
$EOPCT: .WORD    1
    BGT    $DOAGN                ;;YES
    MOV    (PC)+,@(PC)+        ;;RESTORE COUNTER
$ENDCT: .WORD    1
    $EOPCT
    TYPE    , $ENDMG            ;;TYPE "END PASS "
    MOV     $PASS, -(SP)        ;;SAVE $PASS FOR TYPEOUT
    TYPS    , $ENDMG            ;;GO TYPE - DECIMAL ASCII WITH SIGN
    TYPE    , $ENULL            ;;TYPE A NULL CHARACTER
$GET42: MOV     @042,RO          ;;GET MONITOR ADDRESS
    BEQ     $DOAGN              ;;BRANCH IF NO MONITOR
    RESET   $DOAGN              ;;CLEAR THE WORLD
$ENDAD: JSR     PC,(RO)         ;;GO TO MONITOR
    NOP     ;;SAVE ROOM
    NOP     ;;FOR
    NOP     ;;ACT11
$DOAGN:
    JMP     @(PC)+              ;;RETURN
$RTNAD: .WORD    LOOP
$ENULL: .BYTE    1, 1, 0        ;;NULL CHARACTER STRING
$ENDMG: .ASCIIZ  <15><1>>END PASS 0/

;SBTTL SCOPE HANDLER ROUTINE

;*****
;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7;0>)
;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15;08>
;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
;*SW14=1      LOOP ON TEST
;*SW11=1      INHIBIT ITERATIONS
;*SW04=1      LOOP ON ERROR
;*SW08=1      LOOP ON TEST IN SWR<5;0>
;*CALL
;+          SCOPE              ;;SCOPE=101

```

```

21806 133644          $SCOPE:
21807 133644 104407    CRSWR
21808 133646 032777 040000 045264 1$: BIT 0BIT14,0SWR    ;;TEST FOR CHANGE IN SOFT-SWR
21809 133654 001117    BNE $OVER    ;;LOOP ON PRESENT TEST?
21810          ;;YES IF SW14=1
21811 133656 000416    ;####START OF CODE FOR THE XOR TESTER####
21812          $XTSTR: BR 6$    ;;IF RUNNING ON THE "XOR" TESTER CHANGE
21813          MOV 00ERRVEC, (SP)    ;;THIS INSTRUCTION TO A "NOP" (NOP 240)
21814 133664 012737 133704 000004    MOV 05$,00ERRVEC    ;;SAVE THE CONTENTS OF THE ERROR VECTOR
21815 133672 005737 177060    TST 00177060    ;;SET FOR TIMEOUT
21816 133676 012637 000004    MOV (SP)+,00ERRVEC    ;;TIME OUT ON XOR?
21817 133702 000466    BR $SVLAD    ;;RESTORE THE ERROR VECTOR
21818 133704 022626    5$: CMP (SP)+,(SP)+    ;;GO TO THE NEXT TEST
21819 133706 012637 000004    MOV (SP)+,00ERRVEC    ;;CLEAR THE STACK AFTER A TIME OUT
21820 133712 000426    BR 7$    ;;RESTORE THE ERROR VECTOR
21821 133714          6$:;####END OF CODE FOR THE XOR TESTER####
21822 133714 032777 000400 045216    BIT 0BIT08,0SWR    ;;LOOP ON SPEC. TEST?
21823 133722 001407    BEQ 2$    ;;BR IF NO
21824 133724 017746 045210    MOV 0SWR, (SP)    ;;SET DESIRED TEST NUM. FROM SWR
21825 133730 042716 000300    BIC 0$SWRMK,(SP)    ;;STRIP AWAY UNDESIRED BITS
21826 133734 122637 001102    CMPB (SP)+,$TSTNM    ;;ON THE RIGHT TEST?
21827 133740 001465    BEQ $OVER    ;;BR IF YES
21828 133742 105737 001103    2$: TSTB $ERFLG    ;;HAS AN ERROR OCCURRED?
21829 133746 001421    BEQ 3$    ;;BR IF NO
21830 133750 123737 001115 001103    CMPB $ERMAX,$ERFLG    ;;MAX. ERRORS FOR THIS TEST OCCURRED?
21831 133756 101015    BHI 3$    ;;BR IF NO
21832 133760 032777 001000 045152    BIT 0BIT09,0SWR    ;;LOOP ON ERROR?
21833 133766 001404    BEQ 4$    ;;BR IF NO
21834 133770 013737 001110 001106    7$: MOV $LPERR,$LPADR    ;;SET LOOP ADDRESS TO LAST SCOPE
21835 133776 000446    BR $OVER
21836 134000 105037 001103    4$: CLRB $ERFLG    ;;ZERO THE ERROR FLAG
21837 134004 005037 001164    CLR $TIMES    ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
21838 134010 000415    BR 1$    ;;ESCAPE TO THE NEXT TEST
21839 134012 032777 004000 045120    3$: BIT 0BIT11,0SWR    ;;INHIBIT ITERATIONS?
21840 134020 001011    BNE 1$    ;;BR IF YES
21841 134022 005737 001206    TST $PASS    ;;IF FIRST PASS OF PROGRAM
21842 134026 001406    BEQ 1$    ;;INHIBIT ITERATIONS
21843 134030 005237 001104    INC $ICNT    ;;INCREMENT ITERATION COUNT
21844 134034 023737 001164 001104    CMP $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
21845 134042 002024    BGE $OVER    ;;BR IF MORE ITERATION REQUIRED
21846 134044 012737 000001 001104    1$: MOV 01,$ICNT    ;;REINITIALIZE THE ITERATION COUNTER
21847 134052 013737 134130 001164    MOV $MXCNT,$TIMES    ;;SET NUMBER OF ITERATIONS TO DO
21848 134060 105237 001102    $SVLAD: INCB $TSTNM    ;;COUNT TEST NUMBERS
21849 134064 113737 001102 001204    MOVH $TSTNM,$TSTIN    ;;SET TEST NUMBER IN APT MAILBOX
21850 134072 011637 001106    MOV (SP),$LPADR    ;;SAVE SCOPE LOOP ADDRESS
21851 134076 011637 001110    MOV (SP),$LPERR    ;;SAVE ERROR LOOP ADDRESS
21852 134102 005037 001166    CLR $ESCAPE    ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
21853 134106 112737 000001 001115    MOVH 01,$ERMAX    ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
21854 134114 013777 001102 045020    $OVER: MOV $TSTNM,0DISPLAY    ;;DISPLAY TEST NUMBER
21855 134122 013716 001106    MOV $LPADR,(SP)    ;;JUDGE RETURN ADDRESS
21856 134126 000002    RTI    ;;FIXES P5
21857 134130 000001    $MXCNT: 1    ;;MAX. NUMBER OF ITERATIONS
21858          $SPTR: ERROR HANDLER ROUTINE
21859
21860
21861          ;*****
          ;+THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,

```

```

21862 ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
21863 ;*AND GO TO ERTYPE ON ERROR
21864 ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
21865 ;*SW15=1 HALT ON ERROR
21866 ;*SW13=1 INHIBIT ERROR TYPEOUTS
21867 ;*SW10=1 BELL ON ERROR
21868 ;*SW09=1 LOOP ON ERROR
21869 ;*CALL
21870 ;*
21871 ;* ERROR *N ;:ERROR=EMT AND N=ERROR ITEM NUMBER
21872 $ERROR:
21873 134132 005737 004060 TST UQUIET ;:TEST FOR USER QUIET MODE
21874 134136 001403 BEQ 9$ ;:BRANCH IF FIELD-SERVICE MODE
21875 134140 005000 CLR RO ;:IN CASE RO HAS A 03 IN IT (FC)
21876 134142 004737 134354 JSR PC,ABORT ;:TEST FOR ABORT CONDITION
21877 134146 9$:
21878 134146 104407 CKSWR ;:TEST FOR CHANGE IN SOFT SWR
21879 134150 052737 001000 177520 EIS ;:ENABLE HALT ON BREAK
21880 134156 125237 001103 7$: INCB $ERFLG ;:SET THE ERROR FLAG
21881 134162 001775 BEQ 7$ ;:DON'T LET THE FLAG GO TO ZERO
21882 134164 013777 001102 044750 MOV $TSTNM,$DISPLAY ;:DISPLAY TEST NUMBER AND ERROR FLAG
21883 134172 032777 002000 044740 BIT $BIT10,$SWR ;:BELL ON ERROR?
21884 134200 001402 BEQ 1$ ;:NO SKIP
21885 134202 104401 001170 TYPE , $BELL ;:RING BELL
21886 134206 005237 001112 1$: INC $ERTTL ;:COUNT THE NUMBER OF ERRORS
21887 134212 011637 001116 MOV (SP), $ERRPC ;:GET ADDRESS OF ERROR INSTRUCTION
21888 134216 162737 000002 001116 SUB #2, $ERRPC
21889 134224 117737 044666 001114 MOVB $ERRPC, $ITEMB ;:STRIP AND SAVE THE ERROR ITEM CODE
21890 134232 032777 020000 044700 BIT $BIT13,$SWR ;:SKIP TYPEOUT IF SET
21891 134240 001004 BNE 20$ ;:SKIP TYPEOUTS
21892 134242 004737 131740 JSR PC,ERTYPE ;:GO TO USER ERROR ROUTINE
21893 134246 104401 001175 TYPE , $CRLF
21894 134252 20$:
21895 134252 122737 000001 001220 CMPB $APTENV, $ENV ;:RUNNING IN APT MODE
21896 134260 001007 BNE 2$ ;:NO, SKIP APT ERROR REPORT
21897 134262 113737 001114 134274 MOVB $ITEMB, 21$ ;:SET ITEM NUMBER AS ERROR NUMBER
21898 134270 004737 134532 JSR PC, $AT14 ;:REPORT FATAL ERROR TO APT
21899 134274 000 21$: , BYTE 0
21900 134276 000 22$: , BYTE 0
21901 134276 000777 044634 22$: BR 22$ ;:APT ERROR LOOP
21902 134300 005777 2$: TST $SWR ;:HALT ON ERROR
21903 134304 100002 BPL 3$ ;:SKIP IF CONTINUE
21904 134306 000000 HALT ;:HALT ON ERROR!
21905 134310 104407 CKSWR ;:TEST FOR CHANGE IN SOFT SWR
21906 134312 032777 001000 044620 3$: BIT $BIT09,$SWR ;:LOOP ON ERROR SWITCH SET?
21907 134320 001402 BEQ 4$ ;:BR IF NO
21908 134322 013716 001110 MOV $ERRR, (SP) ;:JUDGE RETURN FOR LOOPING
21909 134326 005737 001166 4$: TST $ESCAPE ;:CHECK FOR AN ESCAPE ADDRESS
21910 134332 001402 BEQ 5$ ;:BR IF NONE
21911 134334 013716 001166 MOV $ESCAPE, (SP) ;:JUDGE RETURN ADDRESS FOR ESCAPE
21912 134340 5$:
21913 134340 022737 133610 000042 6$: CMP $ENDAD, $042 ;:ACT 11 AUTO ACCEPT?
21914 134346 001001 BNE 6$ ;:BRANCH IF NO
21915 134350 000000 HALT ;:END
21916 134352 6$:
21917 134352 000002 RTI ;:RETURN

```

```

21918
21919
21920
21921
21922
21923
21924 134354 005737 004056 ABORT: TST UFDFLG ;TEST FOR USER FRIENDLY MODE
21925 134360 001454 BEQ NOABRT ;IF NOT USED THEN CONTINUE NORMAL OPERATION
21926 134362 020027 000032 CMP R0,#32 ;IS IT A 12?
21927 134366 001443 BEQ ABORTZ ;JUST GO BACK TO CHAIN IF IT IS (NO ERROR)
21928 134370 020027 000003 CMP R0,#3 ;IS A 10?
21929 134374 001404 BEQ ABORTC ;BR TO LOAD PC ON XXDP+ STACK (NO ERROR)
21930 134376 005737 004060 TST UQUIET ;TEST FOR USER QUIET MODE
21931 134402 001443 BEQ NOABRT ;IF FIELD SERVICE MODE, CONTINUE NORMAL OPERATION
21932
21933 134404 000422 BR ABORTE ;BECAUSE FIELD SERVICE MODE DOES NOT QUIT ON ERROR
21934
21935
21936
21937 134406 013737 004052 000030 ABORTC MOV SAV30,30 ;RESTORE EMT LOCATION (30)
21938 134414 013737 004054 000032 MOV SAV32,32 ;RESTORE EMT PRIORITY LOCATION (32)
21939 134422 104043 FMT #43 ;GET XXDP STACK LOC. INTO R0 FROM MONITOR
21940 134424 005720 1$; TST (R0); ;FIND END OF STACK
21941 134426 001376 BNE 1$
21942 134430 112760 000057 177777 MOVBL #1,(R0) ;LOAD SLASH OVER ZERO
21943 134436 112720 000136 MOVBL #1,(R0); ;LOAD UPARROW
21944 134442 112720 000103 MOVBL #1,C,(R0); ;LOAD C
21945 134446 105010 CLRB (R0) ;MAKE NEW END TO STACK
21946 134450 000412 BR ABORTZ ;NOW LEAVE
21947 134452 013737 004052 000030 ABORTE: MOV SAV30,30 ;RESTORE EMT LOCATION (30)
21948 134460 013737 004054 000032 MOV SAV32,32 ;RESTORE EMT PRIORITY LOCATION (32)
21949 134466 104042 FMT #42 ;GET DCA LOCATION INTO R0 FROM MONITOR
21950 134470 012760 177777 000042 MOV #1,42(R0) ;SET A 1 INTO LOCATION DRSERR IN MONITOR
21951 134476 013700 000042 ABORTZ: MOV @042,R0 ;AND PUT THE MONITOR RETURN ADDRESS IN R0
21952 134502 005037 000042 CLR @042 ;CLEAR MONITOR RETURN FLAG
21953 134506 000137 133610 JMP $ENDAD ;RETURN TO MONITOR DO NOT PUSH STACK HERE
21954 134512 000207 NOABRT: RTS PC ;IF NOTUED RETURN TO MAINLINE
21955
21956
21957
21958
21959
21960 134514 112737 000001 134760 ;*****
21961 134522 112737 000001 134756 $AT1: MOVBL #1,$FFLG ;TO REPORT FATAL ERROR
21962 134530 000403 $AT3: MOVBL #1,$MFLG ;TO TYPE A MESSAGE
21963 134532 112737 000001 134760 $AT4: MOVBL #1,$FFLG ;TO ONLY REPORT FATAL ERROR
21964 134540 $ATYC:
21965 134540 010046 MOV R0,(SP) ;PUSH R0 ON STACK
21966 134542 010146 MOV R1,(SP) ;PUSH R1 ON STACK
21967 134544 105737 134756 TSTB $MFLG ;SHOULD TYPE A MESSAGE?
21968 134550 001450 BEQ 5$ ;IF NOT: BR
21969 134552 122737 000001 001220 CMPB @APTEENV,$ENV ;OPERATING UNDER APT?
21970 134560 001031 BNE 5$ ;IF NOT: BR
21971 134562 132737 000107 001221 BITB @APTSPOOL,$ENVM ;SHOULD SPOOL MESSAGES?
21972 134570 001450 BEQ 5$ ;IF NOT: BR
21973 134572 017600 000004 MOV @4(SP),R0 ;GET MESSAGE ADDR.

```



```

21974 134576 062766 000002 000004      ADD    02,4(SP)          ;;BUMP RETURN ADDR.
21975 134604 005737 001200      1$:    TST    $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
21976 134610 001375              BNE     1$                  ;;IF NOT: WAIT
21977 134612 010037 001214      MOV     PO,$MSGAD        ;;PUT ADDR IN MAILBOX
21978 134616 105720      2$:    TSTB   (R0),             ;;FIND END OF MESSAGE
21979 134620 001376              BNE     2$
21980 134622 163700 001214      SUB     $MSGAD,R0        ;;SUB START OF MESSAGE
21981 134626 006200              ASR     R0                  ;;GET MESSAGE LGTH IN WORDS
21982 134630 010037 001216      MOV     R0,$MSGLGT        ;;PUT LENGTH IN MAILBOX
21983 134634 012737 000004 001200      MOV     04,$MSGTYPE        ;;TELL APT TO TAKE MSG.
21984 134642 000413              BR      5$
21985 134644 017637 000004 134670 3$:    MOV     04(SP),1$        ;;PUT MSG ADDR IN JSR LINKAGE
21986 134652 062766 000002 000004      ADD     02,4(SP)          ;;BUMP RETURN ADDRESS
21987 134660 013746 177776      MOV     177776,(SP)        ;;PUSH 177776 ON STACK
21988 134664 004737 134762      JSR     PC,$TYPE        ;;CALL TYPE MACRO
21989 134670 000000      4$:    .WORD    0
21990 134672              5$:
21991 134672 105737 134760      10$:   TSTB   $FFLG          ;;SHOULD REPORT FATAL ERROR?
21992 134676 001416              BEQ     12$                ;;IF NOT: BR
21993 134700 005737 001220      TST     $ENV            ;;RUNNING UNDER APT?
21994 134704 001413              BEQ     12$                ;;IF NOT: BR
21995 134706 005737 001200      11$:   TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
21996 134712 001375              BNE     11$                ;;IF NOT: WAIT
21997 134714 017637 000004 001202      MOV     04(SP),$FATAL        ;;GET ERROR #
21998 134722 062766 000002 000004      ADD     02,4(SP)          ;;BUMP RETURN ADDR.
21999 134730 005237 001200      INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
22000 134734 105037 134760      12$:   CLRB   $FFLG          ;;CLEAR FATAL FLAG
22001 134740 105037 134757      CLRB   $LFLG          ;;CLEAR LOG FLAG
22002 134744 105037 134756      CLRB   $MFLG          ;;CLEAR MESSAGE FLAG
22003 134750 012600      MOV     (SP)+,R1        ;;POP STACK INTO R1
22004 134752 012600      MOV     (SP)+,R0        ;;POP STACK INTO R0
22005 134754 000207      RTS     PC              ;;RETURN
22006 134756      000      $MFLG: .BYTE    0          ;;MESSAGE FLAG
22007 134757      000      $LFLG: .BYTE    0          ;;LOG FLAG
22008 134760      000      $FFLG: .BYTE    0          ;;FATAL FLAG
22009      134762      .EVEN
22010      000200      APTSIZE=200
22011      000001      APTENV=001
22012      000100      APTSPool=100
22013      000040      APTCSUP=040
22014      .SBTTL TYPE ROUTINE
22015
22016      ;*****
22017      ;*ROUTINE TO TYPE ASCII MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
22018      ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
22019      ;*NOTE1:      $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
22020      ;*NOTE2:      $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
22021      ;*NOTE3:      $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
22022      ;*
22023      ;*CALL :
22024      ;*1) USING A TRAP INSTRUCTION
22025      ;*      TYPE    ,MESADR          ;;MESADR IS FIRST ADDRESS OF AN ASCII STRING
22026      ;*OR
22027      ;*      TYPE
22028      ;*      MESADR
22029      ;*

```

```

22030
22031 134762 105737 001157 ;TYPE: TSTB $TPFLG ;;IS THERE A TERMINAL?
22032 134766 100002 BPL 1$ ;;BR IF YES
22033 134770 000000 HALT ;;HALT HERE IF NO TERMINAL
22034 134772 000430 BR 3$ ;;LEAVE
22035 134774 010046 1$: MOV R0, -(SP) ;;SAVE R0
22036 134776 017600 000002 MOV @2(SP), R0 ;;GET ADDRESS OF ASCIZ STRING
22037 135002 122737 000001 001220 CMPB @APTENV, $ENV ;;RUNNING IN APT MODE
22038 135010 001011 BNE 62$ ;;NO, GO CHECK FOR APT CONSOLE
22039 135012 132737 000100 001221 BITB @APTPOOL, $ENVM ;;SPOOL MESSAGE TO APT
22040 135020 001405 BEQ 62$ ;;NO, GO CHECK FOR CONSOLE
22041 135022 010037 135032 MOV R0, 61$ ;;SETUP MESSAGE ADDRESS FOR APT
22042 135026 004737 134522 JSR PC, $ATY3 ;;SPOOL MESSAGE TO APT
22043 135032 000000 61$: .WORD 0 ;;MESSAGE ADDRESS
22044 135034 132737 000040 001221 62$: BITB @APTCUP, $ENVM ;;APT CONSOLE SUPPRESSED
22045 135042 001003 BNE 60$ ;;YES, SKIP TYPE OUT
22046 135044 112046 2$: MOVB (R0)+, -(SP) ;;PUSH CHARACTER TO BE TYPED ONTO STACK
22047 135046 001005 BNE 4$ ;;BR IF IT ISN'T THE TERMINATOR
22048 135050 005726 TST (SP)+ ;;IF TERMINATOR POP IT OFF THE STACK
22049 135052 012600 60$: MOV (SP)+, R0 ;;RESTORE R0
22050 135054 062716 000002 3$: ADD @2, (SP) ;;ADJUST RETURN PC
22051 135060 000002 RTI ;;RETURN
22052 135062 122716 000011 4$: CMPB @HT, (SP) ;;BRANCH IF <HT>
22053 135066 001430 BEQ 8$
22054 135070 122716 000200 CMPB @CRLF, (SP) ;;BRANCH IF NOT <CRLF>
22055 135074 001006 BNE 5$
22056 135076 005726 TST (SP)+ ;;POP <CR><LF> EQUIV
22057 135100 104401 TYPE ;;TYPE A CR AND LF
22058 135102 001175 $CRLF
22059 135104 105037 135312 CLRB $CHARCNT ;;CLEAR CHARACTER COUNT
22060 135110 000755 BR 2$ ;;GET NEXT CHARACTER
22061 135112 004737 135174 5$: JSR PC, $TYPEC ;;GO TYPE THIS CHARACTER
22062 135116 123726 001156 6$: CMPB $FILLC, (SP)+ ;;IS IT TIME FOR FILLER CHARS.?
22063 135122 001350 BNE 2$ ;;IF NO GO GET NEXT CHAR.
22064 135124 013746 001154 MOV $NULL, -(SP) ;;GET # OF FILLER CHARS. NEEDED
22065 135130 105366 000001 7$: DECB 1(SP) ;;AND THE NULL CHAR.
22066 135134 002770 BLT 6$ ;;DOES A NULL NEED TO BE TYPED?
22067 135136 004737 135174 JSR PC, $TYPEC ;;BR IF NO--GO POP THE NULL OFF OF STACK
22068 135142 105337 135312 DECB $CHARCNT ;;GO TYPE A NULL
22069 135146 000770 BR 7$ ;;DO NOT COUNT AS A COUNT
22070
22071
22072 ;HORIZONTAL TAB PROCESSOR
22073
22074 135150 112716 000040 8$: MOVB @ , (SP) ;;REPLACE TAB WITH SPACE
22075 135154 004737 135174 9$: JSR PC, $TYPEC ;;TYPE A SPACE
22076 135160 132737 000007 135312 BITB @7, $CHARCNT ;;BRANCH IF NOT AT
22077 135164 001372 BNE 9$ ;;TAB STOP
22078 135170 005726 TST (SP)+ ;;POP SPACE OFF STACK
22079 135172 000724 BR 2$ ;;GET NEXT CHARACTER
22080 135174
22081 135174 105777 043744 $TYPEC: TSTB @TKS ;;CHAR IN BYRD BUFFER? ;MJD001
22082 135200 100022 BPL 10$ ;;BR IF NOT ;MJD001
22083 135202 017746 043740 MOV @TKB, (SP) ;;GET CHAR ;MJD001
22084 135206 042716 177600 BIC @177600, (SP) ;;STRIP EXTRANEOUS BITS ;MJD001
22085 135212 122716 000023 CMPB @XOFF, (SP) ;;WAS CHAR XOFF ;MJD001

```


N1

COKDABO KDJ11-B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 428
 COKDAB.P11 05-APR-84 16:45 TYPE ROUTINE

SEQ 0427

```

22086 135216 001012          BNE      102$          ;;BR IF NOT          ;MJD001
22087 135220          101$:      TSIB      @TKS          ;;WAIT FOR CHAR      ;MJD001
22088 135220 105777 043720      BPL      101$          ;MJD001
22089 135224 100375          MOV      @TKB,(SP)      ;;GET CHAR          ;MJD001
22090 135226 117716 043714      BIC      #177600,(SP)  ;;STRIP IT          ;MJD001
22091 135232 042716 177600      CMP      #XON,(SP)    ;;WAS IT XON?        ;MJD001
22092 135236 122716 000021      BNE      101$          ;;BR IF NOT          ;MJD001
22093 135242 001366          102$:      TST      (SP)+      ;;FIX STACK          ;MJD001
22094 135244          10$:      TSIB      @TPS          ;;WAIT UNTIL PRINTER IS READY ;MJD001
22095 135244 005726          BPL      10$          ;MJD001
22096 135246          MOV      2(SP),@TPB          ;;LOAD CHAR TO BE TYPED INTO DATA REG.
22097 135246 105777 043676      CMP      @CR,2(SP)    ;;IS CHARACTER A CARRIAGE RETURN?
22098 135252 100375          BNE      1$          ;;BRANCH IF NO
22099 135254 116677 000002 043670 1$:      CLRB      $CHARCNT      ;;YES--CLEAR CHARACTER COUNT
22100 135262 122766 000015 000002      BR      $TYPEX      ;;EXIT
22101 135270 001003          CMP      @LF,2(SP)    ;;IS CHARACTER A LINE FEED?
22102 135272 105037 135312      BEQ      $TYPEX      ;;BRANCH IF YES
22103 135276 000406          INCB      (PC)+      ;;COUNT THE CHARACTER
22104 135300 122766 000012 000002      $CHARCNT: WORD 0      ;;CHARACTER COUNT STORAGE
22105 135306 001402          $TYPEX: RTS      PC
22106 135310 105227          ;SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
22107 135312 000000          ;*****
22108 135314 000207          ;THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
22109          ;OCTAL (ASCII) NUMBER AND TYPE IT.
22110          ;$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
22111          ;CALL:
22112          ;*      MOV      NUM, -(SP)          ;;NUMBER TO BE TYPED
22113          ;*      TYPOS          ;;CALL FOR TYPEOUT
22114          ;*      .BYTE  N          ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
22115          ;*      .BYTE  M          ;;M=1 OR 0
22116          ;*          ;;1=TYPE LEADING ZEROS
22117          ;*          ;;0=SUPPRESS LEADING ZEROS
22118          ;*
22119          ;*      $TYPON- -ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
22120          ;*      $TYPOS OR $TYPOC
22121          ;*CALL:
22122          ;*      MOV      NUM, -(SP)          ;;NUMBER TO BE TYPED
22123          ;*      TYPON          ;;CALL FOR TYPEOUT
22124          ;*
22125          ;*      $TYPOC -ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
22126          ;*CALL:
22127          ;*      MOV      NUM, -(SP)          ;;NUMBER TO BE TYPED
22128          ;*      TYPOC          ;;CALL FOR TYPEOUT
22129          ;*
22130          ;*      $TYPOS: MOV      @SP, (SP)      ;;PICKUP THE MODE
22131          ;*      MOV      1(SP), $OFL      ;;LOAD ZERO FILL SWITCH
22132          ;*      MOV      (SP)+, $OMODE+1      ;;NUMBER OF DIGITS TO TYPE
22133          ;*      ADD      #2, (SP)          ;;ADJUST RETURN ADDRESS
22134          ;*      BR      $TYPON
22135          ;*      $TYPOC: MOV      @1, $OFL      ;;SET THE ZERO FILL SWITCH
22136          ;*      MOV      @6, $OMODE+1      ;;SET FOR SIX-DIGIT
22137          ;*
22138 135316 017646 000000 135541      $TYPOS: MOV      @SP, (SP)      ;;PICKUP THE MODE
22139 135322 115637 000001 135541      MOV      1(SP), $OFL      ;;LOAD ZERO FILL SWITCH
22140 135330 112637 135543      MOV      (SP)+, $OMODE+1      ;;NUMBER OF DIGITS TO TYPE
22141 135334 062716 000002      ADD      #2, (SP)      ;;ADJUST RETURN ADDRESS
22142 135340 000406          BR      $TYPON
22143 135342 112737 000001 135541      $TYPOC: MOV      @1, $OFL      ;;SET THE ZERO FILL SWITCH
22144 135350 112737 000006 135543      MOV      @6, $OMODE+1      ;;SET FOR SIX-DIGIT

```

```

22142 135356 112737 000005 135540 $TYPON: MOVB 05,$OCNT ;;SET THE ITERATION COUNT
22143 135364 010346 MOV R3,-(SP) ;;SAVE R3
22144 135366 010446 MOV R4,-(SP) ;;SAVE R4
22145 135370 010546 MOV R5,(SP) ;;SAVE R5
22146 135372 113704 135543 MOVB $OMODE+1,R4 ;;GET THE NUMBER OF DIGITS TO TYPE
22147 135376 005406 NEG R4
22148 135400 062704 000006 ADD 06,R4 ;;SUBTRACT IT FOR MAX. ALLOWED
22149 135404 110437 135542 MOVB R4,$OMODE ;;SAVE IT FOR USE
22150 135410 113704 135541 MOVB $OFILL,R4 ;;GET THE ZERO FILL SWITCH
22151 135414 016605 000012 MOV 12(SP),R5 ;;PICKUP THE INPUT NUMBER
22152 135420 005003 CLR R3 ;;CLEAR THE OUTPUT WORD
22153 135422 006105 1$: ROL R5 ;;ROTATE MSB INTO "C"
22154 135424 000404 BR 3$ ;;GO DO MSB
22155 135426 006105 2$: ROL R5 ;;FORM THIS DIGIT
22156 135430 006105 ROL R5
22157 135432 006105 ROL R5
22158 135434 010503 MOV R5,R3
22159 135436 006103 3$: ROL R3 ;;GET LSB OF THIS DIGIT
22160 135440 105337 135542 DECB $OMODE ;;TYPE THIS DIGIT?
22161 135444 100016 BPL 7$ ;;BR IF NO
22162 135446 042703 177770 BIC 0177770,R3 ;;GET RID OF JUNK
22163 135452 001002 BNE 4$ ;;TEST FOR 0
22164 135454 005704 TST R4 ;;SUPPRESS THIS 0?
22165 135456 001403 BEQ 5$ ;;BR IF YES
22166 135460 005204 4$: INC R4 ;;DON'T SUPPRESS ANYMORE 0'S
22167 135462 052703 000060 BIS 0'0,R3 ;;MAKE THIS DIGIT ASCII
22168 135466 052703 000040 5$: BIS 0',R3 ;;MAKE ASCII IF NOT ALREADY
22169 135472 110337 135536 MOVB R3,R4 ;;SAVE FOR TYPING
22170 135476 104401 135536 TYPE ,R4 ;;GO TYPE THIS DIGIT
22171 135502 105337 135540 7$: DECB $OCNT ;;COUNT BY 1
22172 135506 003347 RGT 2$ ;;BR IF MORE TO DO
22173 135510 002402 BLT 6$ ;;BR IF DONE
22174 135512 005204 INC R4 ;;INSURE LAST DIGIT ISN'T A BLANK
22175 135514 000744 BR 2$ ;;GO DO THE LAST DIGIT
22176 135516 012605 6$: MOV (SP)+,R5 ;;RESTORE R5
22177 135520 012604 MOV (SP)+,R4 ;;RESTORE R4
22178 135522 012603 MOV (SP)+,R3 ;;RESTORE R3
22179 135524 016666 000002 000004 MOV 2(SP),4(SP) ;;SET THE STACK FOR RETURNING
22180 135532 012616 MOV (SP)+,(SP)
22181 135534 000002 RTI ;;RETURN
22182 135536 000 8$: .BYTE 0 ;;STORAGE FOR ASCII DIGIT
22183 135537 000 .BYTE 0 ;;TERMINATOR FOR TYPE ROUTINE
22184 135540 000 $OCNT: .BYTE 0 ;;OCTAL DIGIT COUNTER
22185 135541 000 $OFILL: .BYTE 0 ;;ZERO FILL SWITCH
22186 135542 000000 $OMODE: .WORD 0 ;;NUMBER OF DIGITS TO TYPE
22187 .SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
22188
22189 *****
22190 ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5 DIGIT
22191 ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
22192 ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
22193 ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
22194 ;*REPLACED WITH SPACES.
22195 ;*CALL:
22196 ;* MOV NUM,-(SP) ;*PUT THE BINARY NUMBER ON THE STACK
22197 ;* TYPDS ;*GO TO THE ROUTINE

```

```

22198
22199 135544
22200 135544 010046
22201 135546 010146
22202 135550 010246
22203 135552 010346
22204 135554 010546
22205 135556 012746 020200
22206 135562 016605 000020
22207 135566 100004
22208 135570 005405
22209 135572 112766 000055 000001
22210 135600 005000
22211 135602 012703 135760
22212 135606 112723 000040
22213 135612 005002
22214 135614 016001 135750
22215 135620 160105
22216 135622 002402
22217 135624 005202
22218 135626 000774
22219 135630 060105
22220 135632 005702
22221 135634 001002
22222 135636 105716
22223 135640 100407
22224 135642 106316
22225 135644 103003
22226 135646 116663 000001 177777
22227 135654 052702 000060
22228 135660 052702 000040
22229 135664 110223
22230 135666 005720
22231 135670 020027 000010
22232 135674 002746
22233 135676 003002
22234 135700 010502
22235 135702 000764
22236 135704 105726
22237 135706 100003
22238 135710 116663 177777 177776
22239 135716 105013
22240 135720 012605
22241 135722 012603
22242 135724 012602
22243 135726 012601
22244 135730 012600
22245 135732 104401 135760
22246 135736 016666 000002 000004
22247 135744 012616
22248 135746 000002
22249 135750 023420
22250 135752 001750
22251 135754 000144
22252 135756 000017
22253 135760 000004

$TYPDS:
MOV R0,-(SP) ;;PUSH R0 ON STACK
MOV R1,-(SP) ;;PUSH R1 ON STACK
MOV R2,-(SP) ;;PUSH R2 ON STACK
MOV R3,-(SP) ;;PUSH R3 ON STACK
MOV R5,-(SP) ;;PUSH R5 ON STACK
MOV 020200,-(SP) ;;SET BLANK SWITCH AND SIGN
MOV 20(SP),R5 ;;GET THE INPUT NUMBER
BPL 1$ ;;BR IF INPUT IS POS.
NEG R5 ;;MAKE THE BINARY NUMBER POS.
MOVB 0'-,1(SP) ;;MAKE THE ASCII NUMBER NEG.
1$: CLR R0 ;;ZERO THE CONSTANTS INDEX
MOV 0$DBLK,R3 ;;SETUP THE OUTPUT POINTER
MOVB 0',(R3) ;;SET THE FIRST CHARACTER TO A BLANK
2$: CLR R2 ;;CLEAR THE BCD NUMBER
MOV 0$DTBL(R0),R1 ;;GET THE CONSTANT
3$: SUB R1,R5 ;;FORM THIS BCD DIGIT
BLT 4$ ;;BR IF DONE
INC R2 ;;INCREASE THE BCD DIGIT BY 1
BR 3$
4$: ADD R1,R5 ;;ADD BACK THE CONSTANT
TST R2 ;;CHECK IF BCD DIGIT=0
BNE 5$ ;;FALL THROUGH IF 0
TSTB (SP) ;;STILL DOING LEADING 0'S?
BMI 7$ ;;BR IF YES
5$: ASLB (SP) ;;MSD?
BCC 6$ ;;BR IF NO
MOVB 1(SP),-1(R3) ;;YES--SET THE SIGN
6$: BIS 0'0,R2 ;;MAKE THE BCD DIGIT ASCII
7$: RIS 0',R2 ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
MOVB R2,(R3) ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
TST (R0) ;;JUST INCREMENTING
CMP R0,010 ;;CHECK THE TABLE INDEX
BLT 2$ ;;GO DO THE NEXT DIGIT
BGT 8$ ;;GO TO EXIT
MOV R5,R2 ;;GET THE LSD
BR 6$ ;;GO CHANGE TO ASCII
8$: TSTB (SP) ;;WAS THE LSD THE FIRST NON-ZERO?
BPL 9$ ;;BR IF NO
MOVB -1(SP),-2(R3) ;;YES--SET THE SIGN FOR TYPING
9$: CLRB (R3) ;;SET THE TERMINATOR
MOV (SP),R5 ;;POP STACK INTO R5
MOV (SP),R3 ;;POP STACK INTO R3
MOV (SP),R2 ;;POP STACK INTO R2
MOV (SP),R1 ;;POP STACK INTO R1
MOV (SP),R0 ;;POP STACK INTO R0
TYPE , $DBLK ;;NOW TYPE THE NUMBER
MOV 2(SP),4(SP) ;;ADJUST THE STACK
MOV (SP),-1(SP)
RTI ;;RETURN TO USER

$DTBL: 10000.
1000.
100.
10.
$DBLK: ,BLKW 4

```

DP

CORRADO K0111 R CLUSTER MAY11 30(1046) 05-APR-84 16:45 PAGE 431
COKDAB.P11 05 APR 84 16:45 TTY INPUT ROUTINE

SEQ 0430

```

22254 .SBTTL TTY INPUT ROUTINE
22255
22256 ;;*****
22257 .ENABL LSB
22258
22259 ;;*****
22260 ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
22261 ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
22262 ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
22263 ;*WHEN OPERATING IN TTY FLAG MODE.
22264 135770 022737 000176 001140 $CKSWR: CMP    $SWREG,SWR    ;;IS THE SOFT-SWR SELECTED?
22265 135776 001074          BNE     15$      ;;BRANCH IF NO
22266 136000 105777 043140          TSTB    $TKS      ;;CHAR THERE?
22267 136004 100071          BPL     15$      ;;IF NO, DON'T WAIT AROUND
22268 136006 117746 043134          MOVB    $TKB,-(SP)    ;;SAVE THE CHAR
22269 136012 042716 177600          BIC     $C177,(SP)    ;;STRIP-OFF THE ASCII
22270 136016 022726 000007          CMP     $7,(SP)+    ;;IS IT A CONTROL G?
22271 136022 001062          BNE     15$      ;;NO, RETURN TO USER
22272 136024 123727 001134 000001          CMPB    $AUTOB,$1    ;;ARE WE RUNNING IN AUTO-MODE?
22273 136032 001456          BEQ     15$      ;;BRANCH IF YES
22274
22275 136034 104401 136525          TYPE    , $CNTLG    ;;ECHO THE CONTROL-G (+G)
22276 136040 104401 136532          $GTSWR: TYPE    , $MSWR    ;;TYPE CURRENT CONTENTS
22277 136044 013746 000176          MOV     SWREG,-(SP)    ;;SAVE SWREG FOR TYPEOUT
22278 136050 104402          TYP0C    ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
22279 136052 104401 136543          TYPE    , $MNEW    ;;PROMPT FOR NEW SWR
22280 136056 005046          19$: CLR     -(SP)    ;;CLEAR COUNTER
22281 136060 005046          CLR     -(SP)    ;;THE NEW SWR
22282 136062 105777 043056          7$: TSTB    $TKS      ;;CHAR THERE?
22283 136066 100375          BPL     7$      ;;IF NOT TRY AGAIN
22284
22285 136070 117746 043052          MOVB    $TKB,-(SP)    ;;PICK UP CHAR
22286 136074 042716 177600          BIC     $C177,(SP)    ;;MAKE IT 7-BIT ASCII
22287
22288
22289
22290 136100 021627 000025          9$: CMP     (SP), $25    ;;IS IT A CONTROL U?
22291 136104 001005          BNE     10$      ;;BRANCH IF NOT
22292 136106 104401 136520          TYPE    , $CNTLU    ;;YES, ECHO CONTROL U (+U)
22293 136112 062706 000006          20$: ADD     $6,SP    ;;IGNORE PREVIOUS INPUT
22294 136116 000757          BR      19$      ;;LET'S TRY IT AGAIN
22295
22296
22297 136120 021627 000015          10$: CMP     (SP), $15    ;;IS IT A <CR>?
22298 136124 001022          BNE     16$      ;;BRANCH IF NO
22299 136126 005766 000004          TST     4(SP)    ;;YES, IS IT THE FIRST CHAR?
22300 136132 001403          BEQ     11$      ;;BRANCH IF YES
22301 136134 016677 000002 042776          MOV     2(SP), $SWR    ;;SAVE NEW SWR
22302 136142 062706 000006          11$: ADD     $6,SP    ;;CLEAR UP STACK
22303 136146 104401 001175          14$: TYPE    , $CRLF    ;;ECHO <CR> AND <LF>
22304 136152 123727 001135 000001          CMPB    $INTAG,$1    ;;RE-ENABLE TTY KBD INTERRUPTS?
22305 136160 001003          BNE     15$      ;;BRANCH IF NOT
22306 136162 012777 000100 042754          MOV     $100,$TKS    ;;RE-ENABLE TTY KBD INTERRUPTS
22307 136170 000002          15$: RTI      ;;RETURN
22308 136172 004757 135174          16$: JSR     PC, $TYPEC    ;;ECHO CHAR
22309 136176 021627 000060          CMP     (SP), $60    ;;CHAR = 0?

```

```

22310 136202 002420          BLT      18$          ;;BRANCH IF YES
22311 136204 021627 000067    CMP      (SP),067      ;;CHAR > 7?
22312 136210 003015          BGT      18$          ;;BRANCH IF YES
22313 136212 042726 000060    BIC      060,(SP)+    ;;STRIP-OFF ASCII
22314 136216 005766 000002    TST      2(SP)        ;;IS THIS THE FIRST CHAR
22315 136222 001403          BEQ      17$          ;;BRANCH IF YES
22316 136224 006316          ASL      (SP)          ;;NO, SHIFT PRESENT
22317 136226 006316          ASL      (SP)          ;;CHAR OVER TO MAKE
22318 136230 006316          ASL      (SP)          ;;ROOM FOR NEW ONE.
22319 136232 005266 000002    17$: INC      2(SP)        ;;KEEP COUNT OF CHAR
22320 136236 056616 177776    BIS      -2(SP),(SP)    ;;SET IN NEW CHAR
22321 136240 000707          BR       7$          ;;GET THE NEXT ONE
22322 136244 104401 001174    18$: TYPE    , $QUES    ;;TYPE ?<CR><LF>
22323 136250 000720          BR       20$          ;;SIMULATE CONTROL-U
22324          .DSABL  LSB
22325
22326
22327          ;;*****
22328          ;;THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
22329          ;;CALL:
22330          ;;RDCHR          ;;INPUT A SINGLE CHARACTER FROM THE TTY
22331          ;;RETURN HERE    ;;CHARACTER IS ON THE STACK
22332          ;;              ;;WITH PARITY BIT STRIPPED OFF
22333          ;;
22334
22335 136252 011646          $RDCHR: MOV      (SP),-(SP)    ;;PUSH DOWN THE PC
22336 136254 016666 000004 000002    MOV      4(SP),2(SP)    ;;SAVE THE PS
22337 136262 105777 042656    1$: TSTB     0$TKS        ;;WAIT FOR
22338 136266 100375          BPL      1$          ;;A CHARACTER
22339 136270 117766 042652 000004    MOVB     0$TKB,4(SP)    ;;READ THE TTY
22340 136276 042766 177600 000004    BIC      01C<177>,4(SP)    ;;GET RID OF JUNK IF ANY
22341 136304 026627 000004 000023    CMP      4(SP),023    ;;IS IT A CONTROL-S?
22342 136312 001013          BNE      3$          ;;BRANCH IF NO
22343 136314 105777 042624    2$: TSTB     0$TKS        ;;WAIT FOR A CHARACTER
22344 136320 100375          BPL      2$          ;;LOOP UNTIL ITS THERE
22345 136322 117746 042620    MOVB     0$TKB,-(SP)    ;;GET CHARACTER
22346 136326 042716 177600    BIC      01C177, (SP)    ;;MAKE IT 7-BIT ASCII
22347 136332 022627 000021    CMP      (SP)+,021    ;;IS IT A CONTROL-Q?
22348 136336 001366          BNE      2$          ;;IF NOT DISCARD IT
22349 136340 000750          BR       1$          ;;YES, RESUME
22350 136342 026627 000004 000021    3$: CMP      4(SP),0$XON    ;;IS IT A RANDOM XON?
22351 136350 001744          BEQ      1$          ;;BRANCH IF YES
22352 136352 026627 000004 000140    CMP      4(SP),0140    ;;IS IT UPPER CASE?
22353 136360 002407          BLT      4$          ;;BRANCH IF YES
22354 136362 026627 000004 000175    CMP      4(SP),0175    ;;IS IT A SPECIAL CHAR?
22355 136370 003003          BGT      4$          ;;BRANCH IF YES
22356 136372 042766 000040 000004    BIC      040,4(SP)    ;;MAKE IT UPPER CASE
22357 136400 000002    4$: RTI          ;;GO BACK TO USER
22358
22359          ;;*****
22360          ;;THIS ROUTINE WILL INPUT A STRING FROM THE TTY
22361          ;;CALL:
22362          ;;RDLIN          ;;INPUT A STRING FROM THE TTY
22363          ;;RETURN HERE    ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
22364          ;;              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
22365 136402 010346          $RDLIN: MOV      R3, -(SP)    ;;SAVE R3

```

```

22366 136404 012703 136510 1$: MOV 0$TTYIN,R3 ;;GET ADDRESS
22367 136410 022703 136520 2$: CMP 0$TTYIN+8.,R3 ;;BUFFER FULL?
22368 136414 101405 BLOS 4$ ;;BR IF YES
22369 136416 104410 RDCHR ;;GO READ ONE CHARACTER FROM THE TTY
22370 136420 112613 MOV8 (SP)+,(R3) ;;GET CHARACTER
22371 136422 122713 000177 10$: CMPB 0177,(R3) ;;IS IT A RUBOUT
22372 136426 001003 BNE 3$ ;;SKIP IF NOT
22373 136430 104401 001174 4$: TYPE , $QUES ;;TYPE A '?'
22374 136434 000763 BR 1$ ;;CLEAR THE BUFFER AND LOOP
22375 136436 111337 136506 3$: MOV8 (R3),9$ ;;ECHO THE CHARACTER
22376 136442 104401 136506 TYPE ,9$
22377 136446 122723 000015 CMPB 015,(R3)+ ;;CHECK FOR RETURN
22378 136452 001356 BNE 2$ ;;LOOP IF NOT RETURN
22379 136454 105063 177777 CLR8 -1(R3) ;;CLEAR RETURN (THE 15)
22380 136460 104401 001176 TYPE , $LF ;;TYPE A LINE FEED
22381 136464 012603 MOV (SP)+,R3 ;;RESTORE R3
22382 136466 011646 MOV (SP),-(SP) ;;ADJUST THE STACK AND PUT ADDRESS OF THE
22383 136470 016666 000004 000002 MOV 4(SP),2(SP) ;; FIRST ASCII CHARACTER ON IT
22384 136476 012766 136510 000004 MOV 0$TTYIN,4(SP)
22385 136504 000002 RTI ;;RETURN
22386 136506 000 9$: .BYTE 0 ;;STORAGE FOR ASCII CHAR. TO TYPE
22387 136507 000 .BYTE 0 ;;TERMINATOR
22388 136510 000010 $TTYIN: .BLKB 8. ;;RESERVE 8 BYTES FOR TTY INPUT
22389 136520 052536 005015 000 $CNTLU: .ASCIZ /?U/<15><12> ;;CONTROL "U"
22390 136525 136 006507 000012 $CNTLG: .ASCIZ /?G/<15><12> ;;CONTROL "G"
22391 136532 005015 053523 020122 $MSWR: .ASCIZ <15><12>/SWR = /
22392 136540 020075 000
22393 136543 040 047040 053505 $MNEW: .ASCIZ / NEW = /
22394 136550 036440 000040
22395
22396 .SBTTL READ AN OCTAL NUMBER FROM THE TTY
22397
22398 ;*****
22399 ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
22400 ;*CHANGE IT TO BINARY,
22401 ;*CALL:
22402 ;* RDOCT ;;READ AN OCTAL NUMBER
22403 ;* RETURN HERE ;;LOW ORDER BITS ARE ON TOP OF THE STACK
22404 ;* ;;HIGH ORDER BITS ARE IN $HI OCT
22405 136554 011646 $RDOCT: MOV (SP),-(SP) ;;PROVIDE SPACE FOR THE
22406 136556 016666 000004 000002 MOV 4(SP),2(SP) ;;INPUT NUMBER
22407 136564 010046 MOV R0,-(SP) ;;PUSH R0 ON STACK
22408 136566 010146 MOV R1,-(SP) ;;PUSH R1 ON STACK
22409 136570 010246 MOV R2,-(SP) ;;PUSH R2 ON STACK
22410 136572 104411 1$: RDLIN ;;READ AN ASCII LINE
22411 136574 012600 MOV (SP)+,R0 ;;GET ADDRESS OF 1ST CHARACTER
22412 136576 005001 CLR R1 ;;CLEAR DATA WORD
22413 136600 005002 CLR R2
22414 136602 112046 2$: MOV8 (R0)+,-(SP) ;;PICKUP THIS CHARACTER
22415 136604 001412 DEQ 3$ ;;IF ZERO GET OUT
22416 136606 006301 ASL R1 ;;*2
22417 136610 006102 ROL R1 ;;*4
22418 136612 006301 ASL R1 ;;*8
22419 136614 006102 ROL R2
22420 136616 006301 ASL R1
22421 136620 006102 ROL R2

```



```

22422 136622 042716 177770          BIC    0(C7,(SP)    ;;STRIP THE ASCII JUNK
22423 136626 062601          ADD    (SP)+,R1    ;;ADD IN THIS DIGIT
22424 136630 000764          BR      2$         ;;LOOP
22425 136632 005726          3$: TST    (SP)+    ;;CLEAN TERMINATOR FROM STACK
22426 136634 010166 000012      MOV    R1,12(SP)  ;;SAVE THE RESULT
22427 136640 010237 136654      MOV    R2,$HIOCT
22428 136644 012602          MOV    (SP)+,R2    ;;POP STACK INTO R2
22429 136646 012601          MOV    (SP)+,R1    ;;POP STACK INTO R1
22430 136650 012600          MOV    (SP)+,R0    ;;POP STACK INTO R0
22431 136652 000002          RTI                ;;RETURN
22432 136654 000000          $HIOCT: .WORD    0    ;;HIGH ORDER BITS GO HERE
22433                      .SBTTL  TRAP DECODER
22434
22435                      ;;*****
22436                      ;;THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
22437                      ;;AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
22438                      ;;OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
22439                      ;;GO TO THAT ROUTINE.
22440
22441 136656 010046          $TRAP: MOV    R0,-(SP)    ;;SAVE R0
22442 136660 016600 000002      MOV    2(SP),R0    ;;GET TRAP ADDRESS
22443 136664 005740          TST    -(R0)        ;;BACKUP BY 2
22444 136666 111000          MOVB   (R0),R0        ;;GET RIGHT BYTE OF TRAP
22445 136670 006300          ASL    R0            ;;POSITION FOR INDEXING
22446 136672 016000 136712      MOV    $TRPAD(R0),R0  ;;INDEX TO TABLE
22447 136676 000200          RTS    R0            ;;GO TO ROUTINE
22448
22449
22450                      ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
22451
22452 136700 011646          $TRAP2: MOV    (SP),-(SP)  ;;MOVE THE PC DOWN
22453 136702 016666 000004 000002  MOV    4(SP),2(SP)  ;;MOVE THE PSW DOWN
22454 136710 000002          RTI                ;;RESTORE THE PSW
22455
22456                      .SBTTL  TRAP TABLE
22457
22458                      ;;THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
22459                      ;;BY THE "TRAP" INSTRUCTION.
22460
22461                      ; ROUTINE
22462                      ;
22463 136712 136700          $TRPAD: .WORD    $TRAP2
22464 136714 134762          $TYPE    ;;CALL+TYPE    TRAP+1(104401)  TTY TYPEOUT ROUTINE
22465 136716 135342          $TYP0C   ;;CALL+TYP0C   TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
22466 136720 135316          $TYP0S   ;;CALL+TYP0S   TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
22467 136722 135356          $TYP0N   ;;CALL+TYP0N   TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
22468 136724 135544          $TYPDS   ;;CALL+TYPDS   TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
22469
22470 136726 136040          $GT5WR    ;;CALL+GT5WR    TRAP+6(104406)  GET SOFT SWR SETTING
22471
22472 136730 135770          $CK5WR    ;;CALL+CK5WR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT SWR
22473 136732 136252          $RDCHR    ;;CALL+RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
22474 136734 136402          $RDLIN    ;;CALL+RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
22475 136736 136554          $RDOCT    ;;CALL+RDOCT    TRAP+12(104412) READ AN OCTAL NUMBER FROM TTY
22476                      .SBTTL  POWER DOWN AND UP ROUTINES
22477

```

```

22478
22479
22480 136740 012737 137100 000024 $PWRDN: MOV $ILLUP,$PWRVEC ;;SET FOR FAST UP
22481 136746 012737 000340 000026 MOV $340,$PWRVEC+2 ;;PRIO:7
22482 136754 010046 MOV R0,-(SP) ;;PUSH R0 ON STACK
22483 136756 010146 MOV R1,-(SP) ;;PUSH R1 ON STACK
22484 136760 010246 MOV R2,-(SP) ;;PUSH R2 ON STACK
22485 136762 010346 MOV R3,-(SP) ;;PUSH R3 ON STACK
22486 136764 010446 MOV R4,-(SP) ;;PUSH R4 ON STACK
22487 136766 010546 MOV R5,-(SP) ;;PUSH R5 ON STACK
22488 136770 017746 042144 MOV @SWR,-(SP) ;;PUSH @SWR ON STACK
22489 136774 010637 137104 MOV SP,$SAVR6 ;;SAVE SP
22490 137000 012737 137012 000024 MOV $PWRUP,$PWRVEC ;;SET UP VECTOR
22491 137006 000000 HALT
22492 137010 000776 BR -2 ;;HANG UP
22493
22494
22495
22496 137012 012737 137100 000024 $PWRUP: MOV $ILLUP,$PWRVEC ;;SET FOR FAST DOWN
22497 137020 013706 137104 MOV $SAVR6,SP ;;GET SP
22498 137024 005037 137104 CLR $SAVR6 ;;WAIT LOOP FOR THE TTY
22499 137030 005237 137104 1$: INC $SAVR6 ;;WAIT FOR THE INC
22500 137034 001375 BNE 1$ ;;OF WORD
22501 137036 012677 042076 MOV (SP)+,@SWR ;;POP STACK INTO @SWR
22502 137042 012605 MOV (SP)+,R5 ;;POP STACK INTO R5
22503 137044 012604 MOV (SP)+,R4 ;;POP STACK INTO R4
22504 137046 012603 MOV (SP)+,R3 ;;POP STACK INTO R3
22505 137050 012602 MOV (SP)+,R2 ;;POP STACK INTO R2
22506 137052 012601 MOV (SP)+,R1 ;;POP STACK INTO R1
22507 137054 012600 MOV (SP)+,R0 ;;POP STACK INTO R0
22508 137056 012737 136740 000024 MOV $PWRDN,$PWRVEC ;;SET UP THE POWER DOWN VECTOR
22509 137064 012737 000340 000026 MOV $340,$PWRVEC+2 ;;PRIO:7
22510 137072 104401 TYPE ;;REPORT THE POWER FAILURE
22511 137074 137106 $PWRMG: .WORD $POWER ;;POWER FAIL MESSAGE POINTER
22512 137076 000002 RTI
22513 137100 000000 $ILLUP: HALT ;;THE POWER UP SEQUENCE WAS STARTED
22514 137102 000776 BR -2 ;; BEFORE THE POWER DOWN WAS COMPLETE
22515 137104 000000 $SAVR6: 0 ;;PUT THE SP HERE
22516 137106 005015 047520 042527 $POWER: .ASCIIZ <15><12>"POWER"
22517 137114 000122 .EVEN
22518
22519 000001 .END

```


ABASE = 000000	ALR5TS 021716	BIT08 = 000400	CSR2 002646	DT52 131640
ABOEXT 105774	AMADR1 = 000000	BIT09 = 001000	CSR22 002666	DT64 131652
ABORT 134354	AMADR2 = 000000	BIT1 = 000002	CURADD 110524	DT65 131664
ABORTC 134406	AMADR3 = 000000	BIT10 = 002000	CURDAT 110516	DT7 131464
ABORTE 134452	AMADR4 = 000000	BIT11 = 004000	C121A 023102	DT75 131674
ABORTI 104522	AMAMS1 = 000000	BIT12 = 010000	C121B 023122	DVDSUB 066232
ABORTR 104556	AMAMS2 = 000000	BIT13 = 020000	C121C 023142	DVFSUB 065270
ABORT7 134476	AMAMS3 = 000000	BIT14 = 040000	DAPABO 104300	D1 052432
ABORTO 043636	AMAMS4 = 000000	BIT15 = 100000	DATA 002654	D2 052444
ABORT7 044024	AMSGAD = 000000	BIT2 = 000004	DATA2 002674	D3 052446
ABROUT 105652	AMSGLG = 000000	BIT3 = 000010	DATBO 120452	D4 052460
ACDW1 = 000000	AMSGTY = 000000	BIT4 = 000020	DATI 121002	D5 052474
ACDW2 = 000000	AMTYP1 = 000000	BIT5 = 000040	DATVER 133462	D6 052504
ACPUOP = 000000	AMTYP2 = 000000	BIT6 = 000100	DATVER 133444	D7 052510
ACTCHS 002704	AMTYP3 = 000000	BIT7 = 000200	DAT1 133474	EMTOA 030304
AC0 = 0000000	AMTYP4 = 000000	BIT8 = 000400	DCOUNT 110500	EMTOB 030312
AC1 = 0000001	APASS = 000000	BIT9 = 001000	DDISP = 177570	EMTSV 003764
AC2 = 0000002	APRIOR = 000000	BNO 043566	DH1 130360	EMTVEC = 000030
AC3 = 0000003	APTCSU = 000040	BN1 043756	DH105 131321	EM1 122614
AC4 = 0000004	APTENV = 000001	BPTOA 030500	DH115 131353	EM10 123066
AC5 = 0000005	APTSIZ = 000200	BPTOB 030506	DH24 130644	EM100 126402
AC6 = 0000006	APTSPD = 000100	BPTVEC = 000014	DH27 130710	EM101 126440
AC7 = 0000007	ASWREG = 000000	BTER 051202	DH4 130405	EM102 126473
ADDLSB 110510	ATESTN = 000000	BTEXP 003040	DH41 130753	EM103 126555
ADDI 052020	AUNIT = 000000	BTGO 051002	DH43 131024	EM104 126630
ADDTRP 133302	AUSWR = 000000	BTRES 003050	DH47 131124	EM105 126700
ADDW0 = 000000	AVECT1 = 000000	BTST 051446	DH5 130472	EM106 126753
ADDW1 = 000000	AVECT2 = 000000	BTSTE 051474	DH65 131211	EM107 127031
ADDW10 = 000000	A126 025066	BYPARR 103134	DH7 130571	EM11 123107
ADDW11 = 000000	BA 002650	CCR = 177746	DH72 131255	EM110 127066
ADDW12 = 000000	BACDAT 102644	CCRTBL 105612	DISPLA 001142	EM111 127130
ADDW13 = 000000	BA2 002670	CHECK 015160	DISPRE 000174	EM112 127206
ADDW14 = 000000	BCR = 177524	CHECK1 015236	DMAPAR 117670	EM113 127271
ADDW15 = 000000	BCSR = 177520	CHECK2 015314	DMARD 133034	EM114 127355
ADDW2 = 000000	BDR = 177524	CHECK7 053222	DMATRN 132752	EM115 127376
ADDW3 = 000000	BFA 052052	CHECK10 053442	DPAREN 117730	EM116 127446
ADDW4 = 000000	BFAC1 051640	CHECK26 056224	DSWR = 177570	EM117 127527
ADDW5 = 000000	BFAC2 051660	CHECK27 056532	DT1 131430	EM12 123131
ADDW6 = 000000	BFAC3 051700	CHECK30 057042	DT105 131706	EM120 127572
ADDW7 = 000000	BFAC4 051720	CHECK32 057434	DT115 131716	EM121 127640
ADDW8 = 000000	BFAC5 051742	CHECK33 057710	DT130 131732	EM122 127736
ADDW9 = 000000	BFAE 052112	CHECK7 053226	DT14 131476	EM123 127764
ADEVCT = 000000	BFB 052110	CHK10 053454	DT17 131510	EM124 130016
ADEVN = 000000	BGNTLP 110026	CHK7 053234	DT24 131522	EM125 130110
AENVN = 000000	BIT0 = 000001	CH10 053446	DT27 131532	EM126 130147
AFATAL = 000000	BIT00 = 000001	CKSWR = 104407	DT35 131544	EM127 130204
ALCTR 003112	BIT01 = 000002	CMPTN 064252	DT4 131436	EM13 123157
ALR0TS 021350	BIT02 = 000004	COUNT 003060	DT41 131556	EM130 130246
ALR1TS 021426	BIT03 = 000010	CPE REG = 177766	DT43 131566	EM131 130273
ALR2TS 021504	BIT04 = 000020	CR = 000015	DT47 131602	EM132 130323
ALR3TS 021552	BIT05 = 000040	CRUF = 000200	DT5 131450	EM14 123206
ALR4TS 021640	BIT06 = 000100	CSR1 002644	DT50 131614	EM15 123231
	BIT07 = 000200	CSR12 002664	DT51 131626	EM16 123265

EM17	123332	EM76	126300	FIN6	053054	KDPA2	172364	MAC3	051076
EM2	122650	EM77	126350	FIN7	053274	KDPA3	172366	MAC4	051112
EM20	123376	ENDHRT	102634	FLAG	002776	KDPA4	172370	MAC5	051126
EM21	123431	ENDLUP	110444	FLO	003022	KDPA5	172372	MADCC	015760
EM22	123474	ENDMOV	110526	FLOAT	003012	KDPA6	172374	MAIREG	177750
EM23	123533	ENDTAG	111630	FMPARR	102306	KDPA7	172376	MALCC	016606
EM24	123607	ENDTLP	110414	FPVEC	000244	KDPDR0	172320	MARCC	016666
EM25	123652	ERR	133442	FRSTST	004564	KDPDR1	172322	MASK	003120
EM26	123712	ERRFP	133440	FSTADD	110520	KDPDR2	172324	MA11	010632
EM27	123761	ERROR	104000	FWDSEQ	110506	KDPDR3	172326	MA55	011536
EM3	122662	ERROUT	101700	GOODAD	002770	KDPDR4	172330	M8B11	010730
EM30	124021	ERRVEC	000004	GPR0TS	004726	KDPDR5	172332	M8B22	011212
EM31	124073	ERTYPE	131740	GPR1TS	005004	KDPDR6	172334	M8CCC	015502
EM32	124120	EXBAD	110460	GPR2TS	005062	KDPDR7	172336	M8C00	010434
EM33	124161	EXBAD2	111602	GPR3TS	005140	KIPAR0	172340	M8C11	011010
EM34	124223	EXITST	110470	GPR4TS	005216	KIPAR1	172342	M8C22	011300
EM35	124263	EXPBDT	105632	GPR5TS	005274	KIPAR2	172344	M8I00	010354
EM36	124311	EXPDAT	110502	GPR6TS	005352	KIPAR3	172346	M8PTC	030434
EM37	124355	EXPIR1	034164	GTSWR	104406	KIPAR4	172350	M8SCC	015554
EM4	122674	EXPTBL	102672	HITMIS	177752	KIPAR5	172352	M8T	032544
EM40	124417	EXPWDT	105622	HMPARR	102460	KIPAR6	172354	M8TA	032566
EM41	124464	FINNOP	021070	HOP10	065502	KIPAR7	172356	M8TB	032570
EM42	124550	FIN1	052174	HOP11	066444	KIPDR0	172300	M8TCC	015432
EM43	124576	FIN10	053514	HOP12	067452	KIPDR1	172302	M8TD	032600
EM44	124627	FIN11	053610	HOP13	070450	KIPDR2	172304	M8TEL	051476
EM45	124667	FIN116	022560	HOP14	071510	KIPDR3	172306	M8TF	032610
EM46	124751	FIN117	022674	HOP15	073150	KIPDR4	172310	M8TO	032622
EM47	125041	FIN120	023016	HOP16	073450	KIPDR5	172312	M8TOA	032652
EM5	122734	FIN121	023414	HOP17	074102	KIPDR6	172314	M8TOB	032654
EM50	125126	FIN122	023664	HOP18	076102	KIPDR7	172316	M8TOC	032666
EM51	125151	FIN125	025026	HOP19	076504	KMCR	177734	M8TOD	032712
EM52	125203	FIN126	026066	HOP20	100074	LASTCH	112426	M8TOE	032714
EM53	125241	FIN127	026526	HOP21	101130	LCDSUB	076372	M8TOF	032726
EM54	125275	FIN13	053774	HOP22	101556	LCFSUB	075770	M8T1	050760
EM55	125333	FIN130	027416	HOP44	064350	LDPARS	132320	M8T2	050762
EM56	125367	FIN14	054124	HT	000011	LDPDRS	132350	M8T2A	050774
EM57	125413	FIN15	054312	ILAOA	030600	LEDS	116672	M8T8	051204
EM6	122767	FIN16	054456	ILBOB	030606	LF	000012	M8TRA	051234
EM60	125445	FIN17	054640	ILL	052674	LKS	177546	M8T8B	051256
EM61	125513	FIN2	052270	ILLBOA	030674	LKSEL	002702	M8T8C	051300
EM62	125542	FIN20	055012	ILLBOB	030702	LKSINT	132444	M8T8D	051322
EM63	125613	FIN21	055202	ILLOP1	052530	LOOP	004562	M8T8E	051346
EM64	125644	FIN22	055346	ILLOP2	052604	LOOPIN	003114	M8T8F	051372
EM65	125703	FIN23	055466	INITMM	132156	LOST	052160	M8T8FG	051406
EM66	125753	FIN24	055642	INQ22	121324	LOWADD	002766	M8T8I	051432
EM67	126012	FIN26	056322	INTRR	104730	LSTADD	110522	M866	011612
EM7	123032	FIN27	056630	INTRPC	104706	LXPSUB	077654	M8B44	011456
EM70	126047	FIN30	057140	IOTOA	030206	MACCC	016034	MCLRD	074136
EM71	126100	FIN31	057260	IOTOB	030214	MACE	051142	MCLRI	074214
EM72	126134	FIN32	057532	IOTVEC	000020	MACO	051032	MCMPD	064050
EM73	126172	FIN33	060006	IOXXX	033340	MACOA	051036	MCTSCC	015702
EM74	126216	FIN4	052610	KDPA0	172360	MAC1	051046	MDAO	010230
EM75	126247	FIN5	052700	KDPA1	172362	MAC2	051062	MDCCC	015622

122

SEQ 0437

MDDSUB	072704	MITO	032144	MJSR4B	013772	MMODD	071510	MSCF	101130
MDFSUB	071234	MITOA	032174	MJSR5	014166	MMODF	070450	MSDF	073450
MDIVD	065502	MITOB	032176	MJSR5A	014200	MMRL5	012274	MSEB	177744
MDIVF	064350	MITOC	032210	MJSR5B	014224	MMR0	177572	MSFD	073150
MDMO	010204	MITOD	032234	MJSR6	014102	MMR1	177574	MSFDI	074102
MDM27	010314	MITOE	032236	MJSR6A	014114	MMR2	177576	MSOB	017316
MDSO	010260	MITOF	032250	MJSR6B	014140	MMR3	172516	MSPAA	007462
MEMK	117310	MJ	012542	MJSR7	014250	MMU	133134	MSPAU	027416
MEMQ	117340	MJP	013374	MJSR7A	014262	MMULD	067452	MSPB	005554
MEMTO	030240	MJP17	013404	MJSR7E	014306	MMULF	066444	MSPBB	007546
MET	032262	MJP27	013426	MJU1	012614	MMUTRP	133306	MSPC	005600
META	032320	MJP27A	013440	MJU1A	012626	MMVCC	015366	MSPD	005630
METB	032330	MJP37	013512	MJU2	012556	MMVEC	000250	MSPEO	005666
METD	032340	MJP37A	013524	MJU2A	012570	MM11	010522	MSPF	005730
METF	032350	MJP67	013450	MJU2B	012600	MM22	011070	MSPG	005772
METO	032370	MJP67A	013462	MJU3	012640	MNCCCC	016116	MSPH	006030
METOA	032434	MJP67B	013476	MJU3A	012652	MNGOP	061746	MSPI	006074
METOB	032444	MJP77	013500	MJU3B	012662	MNNRM1	060722	MSPJ	006134
METOC	032456	MJP77E	013534	MJU4	012734	MNNRM2	061142	MSPK	006254
METOD	032502	MJRA	014312	MJU4A	012746	MNNRM3	061274	MSPM	006322
METOE	032512	MJR27	014346	MJU4B	012760	MNNRM4	061416	MSPN	006400
METOF	032524	MJR27A	014402	MJU5	012674	MNRM	062600	MSPQ	006446
MFA	051500	MJR27B	014422	MJU5A	012706	MODE1	045654	MSPR	006552
MFAU	051476	MJR37	014504	MJU5B	012720	MODE2	047234	MSPQ	006650
MFSRCM	060010	MJR37A	014540	MJU6	012774	MODGAR	071500	MSPR	006756
MIALL	033140	MJR6A	014500	MJU6A	013006	MRLB1	012022	MSPS	007044
MIALLA	033164	MJR6B	014502	MJU7	013026	MRLCC	016370	MSPT	007112
MIALLB	033166	MJR67	014424	MJU7E	013040	MRL0	011750	MSPU	007130
MIALLD	033176	MJR67A	014460	MJ2	012612	MRL2	012102	MSPV	007230
MIALLF	033206	MJR77	014560	MJ5	012772	MRL3	012160	MSPVO	007174
MIL	032740	MJR77A	014614	MJ7	013022	MRL4	012234	MSPX	007274
MILA	032764	MJSI	033220	MLCD	076102	MRL6	012336	MSPY	007344
MILAO	030532	MJSIA	033252	MLCF	075264	MRL7	012374	MSPZ	007412
MILB	032766	MJSIB	033254	MLDC	063502	MRRB1	012514	MSP0	005532
MILD	032776	MJSIC	033266	MLDC2	075164	MRRCC	016450	MSTB3	007616
MILF	033006	MJSID	033312	MLDDM2	060060	MRR0	012472	MSTO	030722
MILLRO	030632	MJSIE	033314	MLDDM3	060146	MRT	031344	MSTOE	030776
MILLO	030044	MJSIF	033326	MLDDM4	060262	MRTA	031366	MSTOEE	031004
MILLOA	030110	MJSR	013534	MLDDM5	060366	MRTB	031370	MST4	007670
MILLOB	030116	MJSRA	013652	MLDDM6	060456	MRTF	031400	MST4B	007726
MILO	033020	MJSRB	014016	MLDDM7	060542	MRTF	031410	MST5	007766
MILUA	033052	MJSRC	014164	MLDM27	060630	MRT0	031422	MST5B	010020
MILUB	033054	MJSR1	013654	MLDSUB	070236	MRTOA	031452	MST6	010062
MILOC	033066	MJSR1A	013666	MLFSUB	067240	MRTOB	031454	MST7	010130
MILUD	033112	MJSR1B	013712	MLS1	074250	MRTOC	031466	MSW37	012436
MILOE	033114	MJSR2	013572	MLS2	074342	MRTOD	031512	MSXP	101272
MILOF	033126	MJSR2A	013604	MLS3	074434	MRTOE	031514	MSXT	017120
MIOT	032066	MJSR2B	013630	MLS4	074544	MRTOF	031526	MSXICC	016756
MIOTA	032110	MJSR3	014020	MLS5	074640	MRTS	014620	MS11	010670
MIOTB	032112	MJSR3A	014032	MLS6	074750	MSB	062456	MS22	011142
MIOTD	032122	MJSR3B	014056	MLS7	075050	MSBCC	016234	MS33	011322
MIOTF	032132	MJSR4	013734	MLX9	076504	MSHCCC	016304	MS77	011662
MIOTO	030142	MJSR4A	013746	MMARK	017372	MSCD	100074	MTP	031540

MTPA	031742	NULL	=	000000	PR2	=	000100	SDPAR6	=	172274	SWREG	=	000176
MTPAA	032016	NXMF IN	=	047456	PR3	=	000140	SDPAR7	=	172276	SW0	=	000001
MTPAE	032046	NXMPAR	=	106022	PR4	=	000200	SDPDR0	=	172220	SW00	=	000001
MTPAH	032014	NXMTRP	=	047064	PR5	=	000240	SDPDR1	=	172222	SW01	=	000002
MTPAL	032004	ODDXX	=	033440	PR6	=	000300	SDPDR2	=	172224	SW02	=	000004
MTPB	031572	OK	=	133342	PR7	=	000340	SDPDR3	=	172226	SW03	=	000010
MTPF	031612	OKAY7	=	042576	PS	=	177776	SDPDR4	=	172230	SW04	=	000020
MTPD	031624	OKAY7A	=	042706	PSW	=	177776	SDPDR5	=	172232	SW05	=	000040
MTPOA	031654	OKA7	=	042664	PSWBTS	=	005434	SDPDR6	=	172234	SW06	=	000100
MTPOB	031656	OK1	=	133370	PWRVEC	=	000024	SDPDR7	=	172236	SW07	=	000200
MTPOC	031670	OK7	=	042646	Q22EN	=	002752	SEQ	=	003032	SW08	=	000400
MTPOD	031714	ONQ22	=	132676	Q22INT	=	132732	SFDSUB	=	073344	SW09	=	001000
MTPOE	031716	PAR	=	133256	Q22SI2	=	132452	SIMGOA	=	002662	SW1	=	000002
MTPOF	031730	PARAD1	=	045504	RAMPAR	=	132374	SIPAR0	=	172240	SW10	=	002000
MTPQ	031602	PARAD2	=	047116	RBUF	=	177562	SIPAR1	=	172242	SW11	=	004000
MTPR	031570	PARVA1	=	045536	RCSR	=	177560	SIPAR2	=	172244	SW12	=	010000
MTRPO	030336	PARVA2	=	047150	RCHR	=	104410	SIPAR3	=	172246	SW13	=	020000
MTRY	027560	PARVA3	=	047266	RDLIN	=	104411	SIPAR4	=	172250	SW14	=	040000
MTRYA	027634	PAR1	=	133260	RDOCT	=	104412	SIPAR5	=	172252	SW15	=	100000
MTRYB	027654	PCR	=	177522	RECDAT	=	110504	SIPAR6	=	172254	SW2	=	000004
MTRYM	027706	PDR	=	133234	RECDST	=	003102	SIPAR7	=	172256	SW3	=	000010
MTSO	015366	PDR1	=	133236	RECFEC	=	003062	SIPDR0	=	172200	SW4	=	000020
MTT	031016	PHY1	=	045622	RECS	=	003072	SIPDR1	=	172202	SW5	=	000040
MTTA	031052	PIR	=	177772	RESVEC	=	000010	SIPDR2	=	172204	SW6	=	000100
MTTB	031054	PIRQ	=	177772	RET1	=	015104	SIPDR3	=	172206	SW7	=	000200
MTTD	031064	PIRQNX	=	034042	RET2	=	015172	SIPDR4	=	172210	SW8	=	000400
MTTE	031074	PIRQT	=	121614	RET3	=	015250	SIPDR5	=	172212	SW9	=	001000
MTTR	031200	PIRQVE	=	000240	RITEDA	=	110512	SIPDR6	=	172214	SXPSUB	=	101452
MTTRA	031244	PIRRTN	=	034342	RTSE	=	014672	SIPDR7	=	172216	TAB1	=	003174
MTTRB	031246	PIRTBL	=	034144	RTS1	=	014640	SIXBIT	=	111774	TAB10	=	003324
MTTRC	031260	PIRTEX	=	034044	RTS6	=	014650	SLEND	=	116510	TAB11	=	003334
MTTRD	031316	PIRXXX	=	034006	RXXX	=	033660	SLOC00	=	002762	TAB11A	=	003344
MTTRE	031320	PIR1	=	034044	R6	=	000006	SLOC01	=	002764	TAB12	=	003354
MTTRF	031332	PIR2	=	034164	R7	=	000007	SPAUI	=	027434	TAB13	=	003364
MTTS	031106	PIR2EX	=	034272	SAVBR	=	002710	SPAUI2	=	027456	TAB13B	=	003374
MTTSA	031142	PIR3	=	034272	SAVMR0	=	003004	SPAUI3	=	027476	TAB14	=	003404
MTTSB	031146	PIR3EX	=	034366	SAVMR1	=	003006	SPAUI4	=	027522	TAB15	=	003414
MTTSD	031156	PIR4	=	034366	SAVMR2	=	003010	SPAUI5	=	027536	TAB16	=	003424
MTTSE	031166	PIR5	=	034444	SAVPCR	=	002706	SPAUI6	=	027554	TAB17	=	003434
MTTSQ	031144	PIR5EX	=	034654	SAVPOS	=	003116	SPS	=	003034	TAB18	=	003444
MOVAD	062710	PIR6	=	034654	SAVSUP	=	003000	SPSJ	=	003036	TAB2	=	003204
MXDF1	061526	PIR6EX	=	034756	SAVUSE	=	003002	SR0	=	177572	TAB21	=	003454
MXOR	017236	PITBL1	=	034250	SAV30	=	004052	SR1	=	177574	TAB22	=	003464
MXRCC	017026	PITBL2	=	034640	SAV32	=	004054	SR2	=	177576	TAB23	=	003474
M2	004610	PI1	=	034314	SCDSUB	=	100714	SR3	=	172516	TAB24	=	003504
M3	004624	PI2	=	034326	SCOPE	=	000004	STACK	=	001100	TAB25	=	003514
M4	004644	PI3	=	034330	SDFSUB	=	073700	START	=	003764	TAB26	=	003524
M5	004664	PLFO	=	043514	SDPAR0	=	172260	STBOT	=	001000	TAB27	=	003534
M6	004704	PLF1	=	043706	SDPAR1	=	172262	STKMT	=	177774	TAB28	=	003544
NEWADD	002774	POLY	=	120001	SDPAR2	=	172264	STNOV1	=	107662	TAB29	=	003554
NEWDAT	110514	PROCNT	=	022744	SDPAR3	=	172266	STNOV2	=	110764	TAB29A	=	003564
NOABRT	134512	PRO	=	000000	SDPAR4	=	172270	SUBI	=	051766	TAB3	=	003214
NOTOK	133364	PR1	=	000040	SDPAR5	=	172272	SWR	=	001140	TAB30	=	003574

COKDABO KDJ11-E CLUSTER MAC111 30(1046) 05-APR-84 16:45 PAGE 441
 COKDAB.P11 05-APR-84 16:45 SYMBOL TABLE

SEQ 0439

TAB31	003604	TE117	022560	TSFP2	052176	TSM16	050014	TST52	116272
TAB32	003614	TE117A	022672	TSFP20	054642	TSM6A	037746	TST53	116510
TAB33	003624	TE120	022676	TSFP21	055014	TSM6B	040404	TST54	117016
TAB34	003634	TE120A	022752	TSFP22	055204	TSM6C	041144	TST55	117372
TAB4	003224	TE121	023026	TSFP23	055350	TSM6D	041644	TST56	117640
TAB40	003644	TE122	023414	TSFP24	055470	TSM16A	050110	TST57	117760
TAB41	003654	TE123	023666	TSFP25	055644	TSM16B	050136	TST6	102466
TAB42	003664	TE124	024106	TSFP26	055762	TSM16C	050162	TST60	120142
TAB43	003674	TE125	024340	TSFP27	056324	TSM16D	050226	TST61	121162
TAB45	003704	TE125A	024600	TSFP3	052270	TSM7	042724	TST62	121440
TAB46	003714	TE126	025026	TSFP30	056632	TSM9	043330	TST63	121624
TAB47	003724	TE126A	025074	TSFP31	057142	TSTADD	002772	TST64	121724
TAB47A	003734	TE126B	025550	TSFP32	057262	TSTLOC	003122	TST65	122106
TAB48	003744	TE127	026066	TSFP33	057534	TSTLUP	107744	TST66	122606
TAB49	003754	TE127A	026266	TSFP4	052364	TST1	004562	TST7	102710
TAB5	003234	TE130	026530	TSFP5	052610	TST10	103146	TST10	044374
TAB5A	003244	TE130A	027006	TSFP6	052700	TST11	103372	TST10D1	053464
TAB6	003254	TF114	021046	TSFP7	053054	TST12	103630	TST10D2	053474
TAB6A	003264	TG114	021056	TSF10	053460	TST13	104012	TST10D4	053504
TAB7	003274	THRBIT	112252	TSF13	053770	TST14	104310	TST11	044574
TAB8	003304	TH114	021066	TSF14	054120	TST15	104436	TST11D1	053600
TAB9	003314	TIMDEL	114452	TSF15	054306	TST16	104632	TST12	045232
TAPABO	104426	TIMEOU	052600	TSF16	054452	TST17	105000	TST14	046754
TA114	021006	TIMOUT	002750	TSF17	054634	TST2	101556	TST15	047640
TA116	022462	TKVEC	000060	TSF2	052244	TST20	105172	TST16	050644
TBITVE	000014	TMM16A	050354	TSF20	055006	TST21	105654	TST16A	050636
TB114	021016	TMM16B	050234	TSF21	055176	TST22	106032	TST1822	045706
TC114	021026	TMM16C	050264	TSF22	055342	TST23	106230	TS2600	056242
TD114	021036	TMM16D	050314	TSF23	055462	TST24	106550	TS2601	056252
TEMP	002710	TMM16E	050344	TSF24	055636	TST25	107306	TS2602	056262
TE102	017576	TMM16F	050346	TSF31	057254	TST26	107432	TS2603	056272
TE103	017620	TM16A	050740	TSF6	053050	TST27	110576	TS2604	056302
TE104	017642	TOUT	133124	TSF7	053240	TST3	101704	TS2605	056312
TE105	017664	TPVEC	000064	TSLOOP	111072	TST30	111630	TS2700	056550
TE106	017706	TRAPVE	000034	TSM4	042736	TST31	112002	TS2701	056560
TE107	017730	TRPFLG	133436	TSMB	042756	TST32	112260	TS2702	056570
TE110	017752	TRPOA	030402	TSMC	042766	TST33	112564	TS2703	056600
TE111	017774	TRPOB	030410	TSMU0	014674	TST34	112716	TS2704	056610
TE112	020016	TRTVEC	000014	TSMU1	034772	TST35	113066	TS2705	056620
TE113	020240	TRYMA	027746	TSMU2	035056	TST36	113352	TS3000	057060
TE113A	020460	TRYMB	030000	TSMU3	036256	TST37	113520	TS3001	057070
TE114	020632	TRYMC	030026	TSMU4	036556	TST4	101752	TS3002	057100
TE115	022002	TS031	057166	TSMU5	037506	TST40	114014	TS3003	057110
TE115A	022212	TSEND	111612	TSMU6	037640	TST41	114172	TS3004	057120
TE115B	022222	TSFP1	052112	TSMU7	042344	TST42	114462	TS3005	057130
TE115C	022230	TSFP10	053274	TSMU8	042772	TST43	114522	TS3200	057452
TE115D	022236	TSFP11	053514	TSMU9	043174	TST44	114620	TS3201	057462
TE115F	022244	TSFP12	053610	TS	044074	TST45	114702	TS3203	057502
TE116	022246	TSFP13	053650	TSM11	044456	TST46	115116	TS3204	057512
TE116A	022522	TSFP14	053774	TSM12	044714	TST47	115160	TS3205	057522
TE116B	022526	TSFP15	054126	TSM13	045302	TST5	102320	TS3300	057726
TE116C	022540	TSFP16	054314	TSM14	046350	TST50	115534	TS3301	057736
TE116D	022550	TSFP17	054460	TSM15	047474	TST51	116002	TS3302	057746

COKDABO KDJ11-B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 442
 COKDAB.P11 05-APR-84 16:45 SYMBOL TABLE

SEQ 0440

TS3303	057756	UDPDR2=	177624	\$BELL	001170	\$FFLG	134760	\$RDCHR	136252
TS3304	057766	UDPDR3=	177626	\$CDW1	001260	\$FILLC	001156	\$RDLIN	136402
TS3305	057776	UDPDR4=	177630	\$CDW2	001262	\$FILLS	001155	\$RDOCT	136554
TS6DA	053030	UDPDR5=	177632	\$CHARC	135312	\$GDADR	001120	\$RDSZ =	000010
TS6DA1	053040	UDPDR6=	177634	\$CKSWR	135770	\$GODAT	001124	\$RTNAD	133622
TS7	042614	UDPDR7=	177636	\$CMTAG	001100	\$GET42	133600	\$SAVR6	137104
TS7DA1	053244	UFDFLG	004056	\$CM3 =	000000	\$GTSWR	136040	\$SCOPE	133644
TS7DA2	053254	UFDSET=	000001	\$CM4 =	000002	\$HD =	000001	\$SETUP=	000137
TS7DA4	053264	UIPAR0=	177640	\$CNTLG	136525	\$HIBTS	000232	\$STUP =	177777
TS7FIN	042770	UIPAR1=	177642	\$CNTLU	136520	\$HIQCT	136654	\$SVLAD	134060
TS9FIN	044072	UIPAR2=	177644	\$CPUOP	001226	\$ICNT	001104	\$SVPC =	000232
TYPDS =	104405	UIPAR3=	177646	\$CRLF	001175	\$ILLUP	137100	\$SWR =	167400
TYPE =	104401	UIPAR4=	177650	\$DBLK	135760	\$INTAG	001135	\$SWREG	001222
TYPDC =	104402	UIPAR5=	177652	\$DDW0	001264	\$ITEMB	001114	\$SWRMK=	000300
TYPON =	104404	UIPAR6=	177654	\$DDW1	001266	\$LF	001176	\$TESTN	001204
TYPDS =	104403	UIPAR7=	177656	\$DDW10	001310	\$LFLG	134757	\$TIMES	001164
T10FIN	044456	UIPAR0=	177600	\$DDW11	001312	\$LPADR	001106	\$TKB	001146
T11FIN	044714	UIPAR1=	177602	\$DDW12	001314	\$LPERR	001110	\$TKS	001144
T114	020764	UIPAR2=	177604	\$DDW13	001316	\$MADR1	001232	\$TMP0	001160
T116	022430	UIPAR3=	177606	\$DDW11'	001320	\$MADR2	001236	\$TMP1	001162
T12FIN	045300	UIPAR4=	177610	\$DDW11	001322	\$MADR3	001242	\$TN =	000067
T122A	023512	UIPAR5=	177612	\$DDW2	001270	\$MADR4	001246	\$TPB	001152
T122B	023644	UIPAR6=	177614	\$DDW3	001272	\$MAIL	001200	\$TPFLG	001157
T123A	024050	UIPAR7=	177616	\$DDW4	001274	\$MAMS1	001230	\$TPS	001150
T123B	024060	UNXPIR	034070	\$DDW5	001276	\$MAMS2	001234	\$TRAP	136656
T123C	024066	UQUIET	004060	\$DDW6	001300	\$MAMS3	001240	\$TRAP2	136700
T123D	024074	VIR1	045570	\$DDW7	001302	\$MAMS4	001244	\$TRP =	000013
T123E	024076	VIR2	047202	\$DDW8	001304	\$MBADR	000234	\$TRPAD	136712
T123F	024102	VIR3	047320	\$DDW9	001306	\$MFLG	134756	\$TSTM	000236
T124A	024302	VMKOR	004062	\$DEVCT	001210	\$MNEW	136543	\$TSTNM	001102
T124B	024312	VQBE1	002656	\$DEVM	001256	\$MSGAD	001214	\$TTYIN	136510
T124C	024320	VQBE2	002676	\$DOAGN	133620	\$MSGLG	001216	\$TYPDS	135544
T124D	024326	VQPR1	002660	\$DTBL	135750	\$MSGTY	001200	\$TYPE	134762
T124E	024330	VQPR2	002700	\$ENDAD	133610	\$MSWR	136532	\$TYPEC	135174
T124F	024334	WC	002652	\$ENDCT	133556	\$MTYP1	001231	\$TYPEX	135314
T13FIN	045706	WCP	002672	\$ENDMG	133627	\$MTYP2	001235	\$TYPDC	135342
T14	047000	WLDTRP	133426	\$ENULL	133624	\$MTYP3	001241	\$TYPON	135356
T14FIN	047352	XBUF =	177566	\$ENV	001220	\$MTYP4	001245	\$TYPDS	135316
T15	047710	XCBIT	016526	\$ENVN	001221	\$MXCNT	134130	\$UNIT	001212
T15A	047766	XCSR =	177564	\$EOP	133514	\$NULL	001154	\$UNITM	000242
T15FIN	050014	XME100	017532	\$EOPCT	133550	\$NWTST =	000000	\$USWR	001224
UDPAR0=	177660	XME101	017554	\$ERFLG	001103	\$OCNT	135540	\$VECT1	001250
UDPAR1=	177662	\$APTHD	000232	\$ERMAX	001115	\$OMODE	135542	\$VECT2	001252
UDPAR2=	177664	\$ATYC	134540	\$ERROR	134132	\$OVER	134114	\$XOFF =	000023
UDPAR3=	177666	\$ATY1	134514	\$ERRPC	001116	\$PASS	001206	\$XON =	000021
UDPAR4=	177670	\$ATY3	134522	\$ERRTB	001324	\$PASTM	000240	\$XTSTR	133656
UDPAR5=	177672	\$ATY4	134532	\$ERTIL	001112	\$POWER	137106	\$GET4 =	000000
UDPAR6=	177674	\$AUTOB	001134	\$ESCAP	001166	\$PWRDN	136740	\$OFILL	135541
UDPAR7	177676	\$BASE	001254	\$ETABL	001220	\$PWRMG	137074	.	= 137116
UDPDR0=	177620	\$BDADR	001122	\$ETEND	001324	\$PWRUP	137012	\$X	= 000232
UDPDR1=	177622	\$BDDAT	001126	\$FATAL	001202	\$QUES	001174		

. ABS. 137116 000

ERRORS DETECTED 0

COKDABO KDJ11 B CLUSTER MACY11 30(1046) 05-APR-84 16:45 PAGE 443
COKDAB.P11 05-APR-84 16:45 SYMBOL TABLE

SEQ 0441

COKDAB,COKDAB.PRT/SOL/NL:TOC+SYSMAC,SML,COKDAB.P11
RUN TIME: 101.53.1 SECONDS
RUN TIME RATIO: 214.156+1.3
CORE USED: 43K (86 PAGES)

COKDAB.P11	05-APR....B1
COKDAB.F11	05-APR....C1
COKDAB.P11	05-APR....D1
COKDAB.P11	05-APR....E1
COKDAB.P11	05-APR....F1
COKDAB.P11	05-APR....G1
COKDAB.P11	05-APR....H1
COKDAB.P11	05-APR....I1
COKDAB.P11	05-APR....J1
COKDAB.P11	05-APR....K1
COKDAB.P11	05-APR....L1
COKDAB.P11	05-APR....M1
COKDAB.P11	05-APR....N1

COKDAB.P11	05-APR....B2
COKDAB.P11	05-APR....C2
COKDAB.P11	05-APR....D2
COKDAB.P11	05-APR....E2
COKDAB.P11	05-APR....F2
COKDAB.P11	05-APR....G2
COKDAB.P11	05-APR....H2
COKDAB.P11	05-APR....I2
COKDAB.P11	05-APR....J2
COKDAB.P11	05-APR....K2
COKDAB.P11	05-APR....L2
COKDAB.P11	05-APR....M2
COKDAB.P11	05-APR....N2

COKDAB.P11	05-APR....B3
	C3