

DRV11

DRV11 TEST
CVKAFEO

AH-8207E-MC
FICHE 1 OF 1

MAY 1983
COPYRIGHT © 77-83
MADE IN USA



Microfiche grid containing multiple frames of data, including text and graphical elements.



1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54

.REM 8

IDENTIFICATION

PRODUCT CODE: AC-8206E-MC
PRODUCT NAME: CVKAFEO DRV11 TEST
PRODUCT DATE: JUNE 1982
MAINTAINER : DIAGNOSTIC ENGINEERING
AUTHOR : J. CARMODY

COPYRIGHT (C) 1977, 1983 BY
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS
ALL RIGHTS RESERVED

THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY
BE USED AND COPIED ONLY IN ACCORDANCE WITH THE TERMS
OF SUCH LICENSE AND WITH THE INCLUSION OF THE ABOVE
COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER COPIES
THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAIL-
ABLE TO ANY OTHER PERSON. NO TITLE TO AND OWNERSHIP
OF THE SOFTWARE IS HEREBY TRANSFERRED.

THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE
WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COM-
MITMENT BY DIGITAL EQUIPMENT CORPORATION.

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

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DEC
DECUS

PDP
DECTAPE

UNIBUS
VAX

MASSBUS

D I G I T A L

55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110

1. ABSTRACT

THIS IS A LOGIC TEST OF THE DRV11. TO ALLOW TESTING OF THE DATA LINES AND INTERRUPTS, A SPECIAL MAINTENANCE CABLE (BC08R) IS USED BY DEFAULT. ALSO, A SPECIAL TEST MODULE IS REQUIRED BY OPTION TO TEST THE NEWDATA RDY AND DATATRANS SIGNALS.

NOTE: THE SPECIAL TEST MODULE IS FOR USE BY IN HOUSE MANUFACTURING ONLY.
SEE SECTION 5.2

THIS TEST WILL OPERATE ON ONE DRV11. SPECIAL OPERATIONAL PROCEDURES ARE REQUIRED TO OPERATE ON OTHER THAN THE PRIMARY DRV11. SEE SEC. 5.4

2. REQUIREMENTS

2.1 EQUIPMENT

LSI-11

DRV11

TEST CABLE (BC08R) (BY OPTION)

TEST MODULE (BY OPTION)
(FOR IN HOUSE MANUFACTURING ONLY)

2.2 STORAGE

2.2.1 PROGRAM STORAGE - 4K

3. LOADING PROCEDURE

3.1 METHOD

UNDER XXDP+ OR
ABSOLUTE LOADER

4. STARTING PROCEDURE

200 - NORMAL ENTRY TO TEST ONE DEVICE

TO LOAD AND EXECUTE

1. UNDER XXDP+ OR LOAD PROGRAM WITH THE ABSOLUTE LOADER.
2. IF ANY PROGRAM OPTIONS ARE REQUIRED, SET THE APPROPRIATE BIT IN THE SOFTWARE SWITCH REGISTER AT LOCATION 422. (REF. SECTION 5.1)
3. START PROGRAM AT 200.

111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166

4. PROGRAM WILL PRINT "END OF PASS" FOLLOWING EACH PASS.

4.1 CONTROL SWITCH SETTING

THIS PROGRAM CONTAINS A SOFTWARE SWITCH REGISTER FOR OPTION SELECTION. FOR IT TO OPERATE THE OPERATOR MUST SELECT THE APPROPRIATE OPTION BY SETTING OR RESETTING THE RESPECTIVE BIT IN THE WORD.

TO DO THIS , THE LSI-11 MUST BE IN ODT MODE.

4.2 STARTING ADDRESS OR ADDRESSES

200 = START OF TEST--FOR NORMAL TESTING

5. OPERATING PROCEDURE

1. THE PROGRAM WILL CYCLE CONTINUOUSLY UNLESS HALTED BY THE OPERATOR, OR SOME ERROR CONDITION.
2. TO HALT THE PROGRAM, DEPRESS THE BREAK KEY. ODT WILL DISPLAY THE PC AT WHICH IT WAS HALTED.
3. IF NEW OPTIONS ARE TO BE SELECTED IN THE SWR, THEY MUST BE SET AT THIS TIME.
4. CONTINUE THE PROGRAM VIA A "P" OR A "G" COMMAND.

5.1 SOFTWARE SWITCH SETTINGS

BIT15 - CONTINUE ON ERROR	(100000)
BIT14 - LOOP ON CURRENT ERROR	(040000)
BIT13 - NOT USED	(020000)
BIT12 - NOT USED	(010000)
BIT11 - NOT USED	(004000)
BIT10 - LOOP ON CURRENT TEST	(002000)
BIT9 - RUN TEST MODULE	(001000)
BIT8 - INHIBIT WRAP CABLE	(000400)
BIT7 - NOT USED	(000200)
BIT6 - NOT USED	(000100)
BIT5 - NOT USED	(000040)
BIT4 - NOT USED	(000020)
BIT3 - NOT USED	(000010)
BIT2 - NOT USED	(000004)
BIT1 - NOT USED	(000002)
BIT0 - NOT USED	(000001)

5.2 SELECTION OF TEST OPTIONS

1. TO TEST NEWDATA RDY AND DATATRANS SIGNALS, THE SPECIAL WRAP MODULE MUST BE INSTALLED. THE OPERATOR MUST ALSO SET BIT9 IN THE SWITCH REGISTER (LOC. 422).
NOTE: THE SPECIAL MODULE IS FOR USE BY IN HOUSE MANUFACTURING ONLY.
2. THIS TEST WILL RUN WITH THE WRAP CABLE BY DEFAULT. TO INHIBIT TESTING WITH THE WRAP CABLE, THE OPERATOR MUST SET BIT8 IN THE SWITCH REGISTER (LOC. 422).

167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222

5.3 WRAP CABLE

THE WRAP CABLE IS REQUIRED TO TEST TRANSFER OF DATA INTO AND OUT OF THE INPUT BUFFER, AND THE DEVICE INTERRUPTS.

NOTE !!!!!!! THIS DIAGNOSTIC IS APPROXIMATELY 95% EFFECTIVE WHEN RUN WITH THE WRAP CABLE, AND APPROXIMATELY 60-70% EFFECTIVE WHEN RUN WITHOUT IT.

5.4 TESTING OTHER DRV11 MODULES

TO TEST A DRV11 NOT ADDRESSED AS 167770, OR VECTORED AT 300, THE OPERATOR MUST SUPPLY THE NEW ADDRESSES AND VECTORS TO THE PROGRAM BY DEPOSITING THEM AT THE LOCATIONS TAGGED BY 'RCSR' IN THE BEGINNING OF THE LISTING. THE ORDER IS AS FOLLOWS:

RCSR: CSR ADDRESS
OUTPUT BUFFER ADDRESS
INPUT BUFFER ADDRESS
HIGH BYTE ADDR. OF OUTPUT BUFFER OR
(OUTPUT BUFFER ADDR -1)
'A' INTERRUPT VECTOR ADDRESS
'A' ADDRESS + 2
'B' INTERRUPT VECTOR ADDRESS
'B' ADDRESS + 2

5.5 EXECUTION TIME

TYPICAL RUN TIMES (ONE PASS)
QUICK VERIFY 1 SEC.
WITH WRAP CABLE 10 SEC.

6. ERRORS

6.1 ERROR REPORTING

ALL ERROR REPORTS WILL BE DONE VIA A HALT WITHIN THE PROGRAM. THIS WILL CAUSE ODT TO DISPLAY THE PC+2 OF THE ERROR HALT. AT THIS TIME THE OPERATOR MUST REFERENCE THE LISTING TO DETERMINE THE ERROR DESCRIPTION. THE NUMBER AT TAG \$FATAL IN THE APT MAILBOX CONTAINS THE ERROR NUMBER AND MAY BE USED TO REFERENCE THE DESCRIPTION IN THE TABLE OF CONTENTS.

6.2 ERROR RECOVERY

IN ORDER TO CONTINUE, THE OPERATOR MUST ISSUE A 'P' TO CONTINUE THE PROGRAM, OR MAY SET THE ERROR LOOP SWITCH PRIOR TO CONTINUING.

7. IMPORTANT TAGS

223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259

FOLLOWING IS A LIST OF IMPORTANT TAGS WITHIN THE LISTING

TAG	COMMENT
--- \$MAIL	START OF THE PROGRAM MAILBOX. MANY CLUES TO PROBLEMS CAN BE FOUND HERE
\$FATAL	ERROR NUMBER. USE THE TABLE OF CONTENTS TO LOCATE THE ERROR INFORMATION AND/OR CODE
\$TESTN	CURRENT TEST NUMBER
\$PASS	PASS COUNT OF THE PROGRAM WHEN ERROR WAS DETECTED OR PROGRAM HALTED
\$SWREG	SOFTWARE SWITCH REGISTER
RCSR	START OF UNIT UNDER TEST ADDRESSES

8. LISTING
&

.TITLE CVKAFE
.ENABLE ABS
.NLIST MD,MC,CND
.LIST ME

000001
000001

X=1
N=1

260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286

```
:REVISION 1. B  
:          2. CHANGED TEST REQUIRING WRAP CABLE TO BE  
:          INHIBITED BY BIT 8 OF THE SWR  
:          2. INITIALIZED A & B INTERRUPT VECTOR LOCATIONS  
:          WITH TRAP CATCHER  
:REVISION 1. C  
:          1. DOCUMENTATION CHANGE ONLY  
:REVISION 1. D  
:          1. ADDED CODE TO PRINT OUT PROGRAMS TITLE  
:REVISION 1. E  
:          1. INCORPORATED PATCHES D1,D2 AND D3  
:          D1 -- "INCORRECT PROGRAM COUNTER ADDRESSING AT  
:          LOC. 1360 IN TEST 1, CAUSED WRONG ADDRESS  
:          TO BE LOADED INTO LOC. 4. THIS RESULTS  
:          IN PROGRAM HALTING AND THE FOLLOWING  
:          MESSAGE TO BE DISPLAYED TO CONSOLE:  
:          'ILLEGAL WORD 1'"  
:          D2 -- "PROGRAM OVERPRINTS END PASS REPORT AND  
:          PROGRAM TITLE."  
:          D3 -- "RELEASED VERSION OF PATCH D2 ON THE MEDIA  
:          IS INCORRECT CAUSING CPU TO HALT AT 314"  
:          2. MADE PROGRAM APT COMPATIBLE
```

```

287 ;GENERAL REGISTER LOGIC TEST
288
289 104000 HLT=104000
290 167770 CSR=167770
291 001200 STKPTR=1200
292 ;REGISTER DEFINITIONS
293 000000 R0=%0
294 000001 R1=%1
295 000002 R2=%2
296 000003 R3=%3
297 000004 R4=%4
298 000005 R5=%5
299 000006 SP=%6
300 000007 PC=%7
301
302 ;SWITCHES
303
304 001000 SW9=1000
305 002000 SW10=2000
306 004000 SW11=4000
307 020000 SW13=20000
308 040000 SW14=40000
309
310 .SBTTL BASIC DEFINITIONS
311
312 ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
313 001100 STACK= 1100
314 .EQUIV EMT,ERROR ;:BASIC DEFINITION OF ERROR CALL
315 .EQUIV IOT,SCOPE ;:BASIC DEFINITION OF SCOPE CALL
316
317 ;*MISCELLANEOUS DEFINITIONS
318 000011 HT= 11 ;:CODE FOR HORIZONTAL TAB
319 000012 LF= 12 ;:CODE FOR LINE FEED
320 000015 CR= 15 ;:CODE FOR CARRIAGE RETURN
321 000200 CRLF= 200 ;:CODE FOR CARRIAGE RETURN-LINE FEED
322 177776 PS= 177776 ;:PROCESSOR STATUS WORD
323 .EQUIV PS,PSW
324 177774 STKLMT= 177774 ;:STACK LIMIT REGISTER
325 177772 PIRQ= 177772 ;:PROGRAM INTERRUPT REQUEST REGISTER
326 177570 DSWR= 177570 ;:HARDWARE SWITCH REGISTER
327 177570 DDISP= 177570 ;:HARDWARE DISPLAY REGISTER
328
329 ;*GENERAL PURPOSE REGISTER DEFINITIONS
330 000000 R0= %0 ;:GENERAL REGISTER
331 000001 R1= %1 ;:GENERAL REGISTER
332 000002 R2= %2 ;:GENERAL REGISTER
333 000003 R3= %3 ;:GENERAL REGISTER
334 000004 R4= %4 ;:GENERAL REGISTER
335 000005 R5= %5 ;:GENERAL REGISTER
336 000006 R6= %6 ;:GENERAL REGISTER
337 000007 R7= %7 ;:GENERAL REGISTER
338 000006 SP= %6 ;:STACK POINTER
339 000007 PC= %7 ;:PROGRAM COUNTER
340
341 ;*PRIORITY LEVEL DEFINITIONS
342 000000 PRO= 0 ;:PRIORITY LEVEL 0

```


343	000040	PR1=	40	::PRIORITY LEVEL	1
344	000100	PR2=	100	::PRIORITY LEVEL	2
345	000140	PR3=	140	::PRIORITY LEVEL	3
346	000200	PR4=	200	::PRIORITY LEVEL	4
347	000240	PR5=	240	::PRIORITY LEVEL	5
348	000300	PR6=	300	::PRIORITY LEVEL	6
349	000340	PR7=	340	::PRIORITY LEVEL	7

351 :*'SWITCH REGISTER' SWITCH DEFINITIONS

352	100000	SW15=	100000
353	040000	SW14=	40000
354	020000	SW13=	20000
355	010000	SW12=	10000
356	004000	SW11=	4000
357	002000	SW10=	2000
358	001000	SW09=	1000
359	000400	SW08=	400
360	000200	SW07=	200
361	000100	SW06=	100
362	000040	SW05=	40
363	000020	SW04=	20
364	000010	SW03=	10
365	000004	SW02=	4
366	000002	SW01=	2
367	000001	SW00=	1
368		.EQUIV	SW09,SW9
369		.EQUIV	SW08,SW8
370		.EQUIV	SW07,SW7
371		.EQUIV	SW06,SW6
372		.EQUIV	SW05,SW5
373		.EQUIV	SW04,SW4
374		.EQUIV	SW03,SW3
375		.EQUIV	SW02,SW2
376		.EQUIV	SW01,SW1
377		.EQUIV	SW00,SW0

379 :*DATA BIT DEFINITIONS (BIT00 TO BIT15)

380	100000	BIT15=	100000
381	040000	BIT14=	40000
382	020000	BIT13=	20000
383	010000	BIT12=	10000
384	004000	BIT11=	4000
385	002000	BIT10=	2000
386	001000	BIT09=	1000
387	000400	BIT08=	400
388	000200	BIT07=	200
389	000100	BIT06=	100
390	000040	BIT05=	40
391	000020	BIT04=	20
392	000010	BIT03=	10
393	000004	BIT02=	4
394	000002	BIT01=	2
395	000001	BIT00=	1
396		.EQUIV	BIT09,BIT9
397		.EQUIV	BIT08,BIT8
398		.EQUIV	BIT07,BIT7

```

399 .EQUIV BIT06,BIT6
400 .EQUIV BIT05,BIT5
401 .EQUIV BIT04,BIT4
402 .EQUIV BIT03,BIT3
403 .EQUIV BIT02,BIT2
404 .EQUIV BIT01,BIT1
405 .EQUIV BIT00,BIT0
  
```

```

407 ;*BASIC "CPU" TRAP VECTOR ADDRESSES
408 ERRVEC= 4 ;: TIME OUT AND OTHER ERRORS
409 RESVEC= 10 ;: RESERVED AND ILLEGAL INSTRUCTIONS
410 TBITVEC=14 ;: "T" BIT
411 TRTVEC= 14 ;: TRACE TRAP
412 BPTVEC= 14 ;: BREAKPOINT TRAP (BPT)
413 IOTVEC= 20 ;: INPUT/OUTPUT TRAP (IOT) **SCOPE**
414 PWRVEC= 24 ;: POWER FAIL
415 EMTVEC= 30 ;: EMULATOR TRAP (EMT) **ERROR**
416 TRAPVEC=34 ;: "TRAP" TRAP
417 TKVEC= 60 ;: TTY KEYBOARD VECTOR
418 TPVEC= 64 ;: TTY PRINTER VECTOR
419 PIRQVEC=240 ;: PROGRAM INTERRUPT REQUEST VECTOR
  
```

```

420 .ENABLE ABS
421 .=0
422 000000 .+2
423 000002 000000 HALT
424 000004 000006 .+2
425 000006 000000 HALT
426 000010 000012 .+2
427 000012 000000 HALT
428 000014 000016 .+2
429 000016 000000 HALT
430 000020 000022 .+2
431 000022 000000 HALT
432 000024 000026 .+2
433 000026 000000 HALT
434 000030 000032 .+2
435 000032 000000 HALT
436 000034 000036 .+2
437 000036 000000 HALT
438 000040 000042 .+2
439 000042 000000 HALT
440 000044 000046 .+2
441 000046 000000 HALT
442 000050 000052 .+2
443 000052 000000 HALT
444 000054 000056 .+2
445 000056 000000 HALT
446 000060 000062 .+2
447 000062 000000 HALT
448 000064 000066 .+2
449 000066 000000 HALT
450 .=100
451 000100 000102 102
452 000102 000002 RTI
453 000104 000106 .+2
454 000106 000000 HALT
  
```

; RTI FOR POSSIBLE CLOCK INTERRUPT


```

455 000110 000112 .+2
456 000112 000000 HALT
457 000114 000116 .+2
458 000116 000000 HALT
459 000120 000122 .+2
460 000122 000000 HALT
461 000124 000126 .+2
462 000126 000000 HALT
463 000130 000132 .+2
464 000132 000000 HALT
465 000134 000136 .+2
466 000136 000000 HALT
467 000140 000142 .+2
468 000142 000000 HALT
469 000144 000146 .+2
470 000146 000000 HALT
471 000150 000152 .+2
472 000152 000000 HALT
473 000154 000156 .+2
474 000156 000000 HALT
475 000160 000162 .+2
476 000162 000000 HALT
477 000164 000166 .+2
478 000166 000000 HALT
479 000170 000172 .+2
480 000172 000000 HALT
481 000200 000200 .=200
482 000200 005067 000202 CLR $PASS ; CLEAR PASS COUNT
483 000204 005067 000172 CLR $FATAL
484 000210 005067 000170 CLR $TESTN
485 000214 000137 001246 JMP @#START1 ; INITIAL START
486 000300 000300 .=300 ; DEVICE INTERRUPT VECTORS
487 000300 000302 .+2
488 000302 000000 HALT
489 000304 000306 .+2
490 000306 000000 HALT
491 000400 000400 .=400
  
```

.SBTTL APT MAILBOX-ETABLE

```

492
493
494
495
496 000400
497 000400 000000
498 000402 000000
499 000404 000000
500 000406 000000
501 000410 000000
502 000412 000000
503 000414 000000
504 000416 000000
505 000420
506 000420 000
507 000421 000
508 000422 000000
509 000424 000000
510 000426 000000
  
```

```

*****
.EVEN
$MAIL: ;: APT MAILBOX
$MSGTY: .WORD AMSGTY ;: MESSAGE TYPE CODE
$FATAL: .WORD AFATAL ;: FATAL ERROR NUMBER
$TESTN: .WORD ATESTN ;: TEST NUMBER
$PASS: .WORD APASS ;: PASS COUNT
$DEVCT: .WORD ADEVCT ;: DEVICE COUNT
$UNIT: .WORD AUNIT ;: I/O UNIT NUMBER
$MSGAD: .WORD AMSGAD ;: MESSAGE ADDRESS
$MSGLG: .WORD AMSGLG ;: MESSAGE LENGTH
$ETABLE: ;: APT ENVIRONMENT TABLE
$ENV: .BYTE AENV ;: ENVIRONMENT BYTE
$ENVM: .BYTE AENVM ;: ENVIRONMENT MODE BITS
$SWREG: .WORD ASWREG ;: APT SWITCH REGISTER
$USWR: .WORD AUSWR ;: USER SWITCHES
$CPUOP: .WORD ACPUOP ;: CPU TYPE,OPTIONS
  
```

```
511          :*          BITS 15-11=CPU TYPE
512          :*          11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
513          :*          11/70=06,PDQ=07,Q=10
514          :*          BIT 10=REAL TIME CLOCK
515          :*          BIT 9=FLOATING POINT PROCESSOR
516          :*          BIT 8=MEMORY MANAGEMENT
517 000430    $ETEND:
518          .MEXIT
519          .SBTTL  APT PARAMETER BLOCK
520
521          :*****
522          :SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
523          :*****
524          .SX=.      ;;SAVE CURRENT LOCATION
525          =24      ;;SET POWER FAIL TO POINT TO START OF PROGRAM
526 000024    200      ;;FOR APT START UP
527          =44      ;;POINT TO APT INDIRECT ADDRESS PNTR.
528 000044    $APTHDR ;;POINT TO APT HEADER BLOCK
529          =.SX     ;;RESET LOCATION COUNTER
530          :*****
531          :SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
532          :INTERFACE SPEC.
533
534 000430    $APTHD:
535 000430    $HIBTS: .WORD 0      ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
536 000432    $MADR: .WORD $MAIL  ;;ADDRESS OF APT MAILBOX (BITS 0-15)
537 000434    $STMT: .WORD 10     ;;RUN TIM OF LONGEST TEST
538 000436    $PASTM: .WORD 10   ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
539 000440    $UNITM: .WORD 0    ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
540 000442    .WORD $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
541          =1200
542          DEVCNT=$DEVCT
543          ERRNUM=$FATAL
544          OPTION=$CPUOP
545
546          ;THIS TABLE CONTAINS INITIAL REGISTER AND VECTOR ADDRESSES
547
548 001200    167770  RCSR:  CSR
549 001202    167772          CSR+2
550 001204    167774          CSR+4
551 001206    167773          CSR+3
552 001210    000300  RCSR1: 300
553 001212    000302          302
554 001214    000304          304
555 001216    000306          306
556
557          ;THIS TABLE CONTAINS REGISTER AND VECTOR ADDRESSES OF THE DR11-C UNDER TEST
558
559 001220    167770  DRCSR: 167770      ;ADDRESS OF DR11-C STATUS REGISTER
560 001222    167772  DROBUF: 167772    ;ADDRESS OF DR OUTPUT BUFFER REG.
561 001224    167774  DRIBUF: 167774    ;ADDRESS OF DR INPUT BUFFER REG.
562 001226    167773  DRBHIO: 167773    ;HIGH BYTE OF OUTPUT BUFFER REG.
563
564 001230    000300  DRVECA: 300      ;INTERRUPT VECTOR OF UNIT UNDER TEST
565 001232    000302  DRLVLA: 302
566 001234    000304  DRVECB: 304      ;INTERRUPT VECTOR
```

```

567 001236 000306      DRLVLB: 306
568 001240 000000      XORFLG: 0
569
570 001242 000000      COUNT: 0          ;COUNT LOCATION
571 001244 000240      PL: 240          ;PRIORITY LEVEL
572
573 001246 012706 001200  START1: MOV #STKPTR,SP
574 001252 000137 001256      JMP @#START
575 ;*****
576
577 ;+          ROUTINE TO PRINT DIAGNOSTIC'S TITLE IF ALLOWED BY APT MODE
578
579 ; ALL CODE ADDED FOR TITLE PRINTOUT IS NOTED
580 ; WITH A ;+ IN THE COMMENT SECTION.
581 ;*****
582 001256 132767 000040 177135  START: BITB #40, $ENVM          ;+ WILL APT ALLOW PRINTING?
583 001264 001013          BNE AROUND                    ;+ NO BRANCH AROUND
584 001266 012700 004616      MOV #TITLE1,R0              ;+ GET STARTING ADDRESS OF TITLE MSG
585 001272 105737 177564      LOOP1: TSTB @#TPS           ;+ CHECK IF TERMINAL READY
586 001276 100375          BPL LOOP1                    ;+ LOOP IF NOT
587 001300 112037 177566      MOVB (R0)+,@#TPB           ;+ PRINT A CHARACTER
588 001304 001372          BNE LOOP1                    ;+ BRANCH BACK IF NOT DONE PRINTING
589 001306 105737 177564      LOOP: TSTB @#TPS
590 001312 100375          BPL LOOP
591 ;*****
592
593 ; INITIALIZE ADDRESS AND VECTORS
594
595 001314 012700 001200      AROUND: MOV #RCSR, R0        ; GET ADDRESS OF UNIT UNDER TEST
596 001320 012701 005000      MOV #5000, R1
597 001324 077101          1$: SOB R1,1$
598 001326 012701 001220      MOV #DRCSR,R1
599
600 001332 012737 004672 000024  MOV #PFAIL,@#24
601 001340 012021          MOV (R0)+,(R1)+             ;LOAD INITIAL TEST ADDRESSES
602 001342 012021          MOV (R0)+,(R1)+
603 001344 012021          MOV (R0)+,(R1)+
604 001346 012021          MOV (R0)+,(R1)+
605 001350 012021          MOV (R0)+,(R1)+
606 001352 012021          MOV (R0)+,(R1)+
607 001354 012021          MOV (R0)+,(R1)+
608
609 ;DOES RESET CLEAR REGISTER?
610 001356          TST1:
611 001356 012767 000001 177020  LP1: MOV #1,$TESTN          ; MOVE TEST NUMBER TO MAILBOX
612 001364 016705 177630      MOV DRCSR,R5              ;GET ADDRESS OF STATUS REGISTER
613 001370 106427 000200      MTPS #200                 ;TURN OFF INTERRUPTS
614 001374 012737 001456 000004  MOV #ERR1,@#4              ;SET TIME OUT TRAP VECTOR
615 001402 012777 177777 177612  MOV #-1,@DROBUF           ;PRESET OUTPUT BUFFER
616 001410 000005          RESET                      ;CLEAR DATA REGISTER
617 001412 017700 177604      MOV @DROBUF,R0            ;GET RESULT OF RESET
618 001416 001450          BEQ CON
619 001420 032767 040000 176774  BIT #BIT14,$SWREG         ; CHECK FOR LOOP ON ERROR
620 001426 001356          BNE LP1                    ; GC TO LOOP ERROR
621 001430 012767 000001 176744  MOV #1,$FATAL
622 001436 012767 000001 176734  MOV #1,$MSGTY             ; MOVE ERROR NUM TO MAILBOX

```


DATA REG DID NOT CLEAR

```

623 001444 005767 176752      TST    $SWREG      ; CHECK FOR HALT ON ERROR
624 001450 100401      BMI    1$         ; CONTINUE IF SET
625 001452 000000      HALT                    ;<DATA REG DID NOT CLEAR>
626 001454                      1$:
627 001454 000431      BR     CON
628 001456                      ERR1:
629 001456 032767 040000 176736  BIT    #BIT14,$SWREG ; CHECK FOR LOOP ON ERROR
630 001464 001337      BNE    LP1        ; GO TO LOOP ERROR
631 001466 012767 000002 176706  MOV    #2,$FATAL
632 001474 012767 000001 176676  MOV    #1,$MSGTY   ; MOVE ERROR NUM TO MAILBOX
633 001502 005767 176714      TST    $SWREG      ; CHECK FOR HALT ON ERROR
634 001506 100401      BMI    1$         ; CONTINUE IF SET
635 001510 000000      HALT                    ;<TIME WHEN ADDRESSING PLU>
636 001512                      1$:
637 001512 032767 002000 176702  BIT    #BIT10,$SWREG ; CHECK FOR LOOP ON TEST
638 001520 001316      BNE    TST1       ; GO TO LOOP ON TEST
639 001522 000407      BR     TST2
640 001524 012706 001200      5$:  MOV    #STKPTR,SP   ; RESET STACK POINTER
641 001530 012737 000006 000004  MOV    #6,@#4      ; RESTORE TIME OUT TRAP
642 001536 000765      BR     1$
643 001540 000772      BR     -12
644                      ; TEST 'NEWDATA RDY' AND 'DATATRANS' SIGNALS IN PLU
645                      ; NOTE***** THE PLU TEST MODULE MUST BE INSTALLED
646                      ; TO EXECUTE THIS TEST
647
648 001542                      TST2:
649 001542 012767 000002 176634  MOV    #2,$TESTN   ; MOVE TEST NUMBER TO MAILBOX
650 001550 032767 001000 176644  BIT    #BIT9,$SWREG
651 001556 001505      BEQ    TST3       ; SKIP TEST IF NOT SELECTED
652 001560 012706 001200      MOV    #STKPTR,SP ; SET UP STACK POINTER
653 001564 000005      RESET                ; CLEAR EVERYTHING
654                      ; THIS RESET SHOULD INITIALIZE THE
655                      ; SIGNAL LATCHES IN THE TEST MODULE
656 001566 012777 031460 177426  MOV    #31460,@DROBUF ; PRIME THE LATCHES
657 001574 000240      NOP
658 001576 000240      NOP
659 001600 017700 177420      MOV    @DRIBUF,R0  ; GET DATA
660 001604 032700 000001      BIT    #BIT0,R0    ; CHECK DATA TRANS SIG
661 001610 001016      BNE    T2CON      ; CONTINUE IF PRESENT
662 001612 032767 040000 176602  BIT    #BIT14,$SWREG ; CHECK FOR LOOP ON ERROR
663 001620 001350      BNE    TST2       ; GO TO LOOP ERROR
664 001622 012767 000003 176552  MOV    #3,$FATAL
665 001630 012767 000001 176542  MOV    #1,$MSGTY   ; MOVE ERROR NUM TO MAILBOX
666 001636 005767 176560      MOV    $SWREG      ; CHECK FOR HALT ON ERROR
667 001642 100401      TST    1$         ; CONTINUE IF SET
668 001644 000000      HALT                    ;<NO DATA TRANS SIGNAL>
669 001646                      1$:
670 001646 032700 000002      T2CON: BIT    #BIT1,R0    ; CHECK NEW DATA RDY SIGNAL
671 001652 001016      BNE    T2CN1      ; CONTINUE IF OK
672 001654 032767 040000 176540  BIT    #BIT14,$SWREG ; CHECK FOR LOOP ON ERROR
673 001662 001371      BNE    T2CON      ; GO TO LOOP ERROR
674 001664 012767 000004 176510  MOV    #4,$FATAL
675 001672 012767 000001 176500  MOV    #1,$MSGTY   ; MOVE ERROR NUM TO MAILBOX
676 001700 005767 176516      TST    $SWREG      ; CHECK FOR HALT ON ERROR
677 001704 100401      BMI    1$         ; CONTINUE IF SET
678 001706 000000      HALT                    ;<NO NEW DATA RDY SIGNAL>

```

```

679 001710 1$:
680 001710 T2CN1: RESET ; CLEAR EVERYTHING
681 001712 000005 NOP
682 001714 000240 NOP
683 001716 017700 177302 MOV @DRIBUF,RO ; CHECK SIGNAL LATCHES
684 001722 005700 TST RO ; SHOULD BE CLEAR
685 001724 001416 BEQ 1$ ; CONTINUE IF CLEAR
686 001726 032767 040000 176466 BIT #BIT14,$SWREG ; CHECK FOR LOOP ON ERROR
687 001734 001365 BNE T2CN1 ; GO TO LOOP ERROR
688 001736 012767 000005 176436 MOV #5,$FATAL
689 001744 012767 000001 176426 MOV #1,$MSGTY ; MOVE ERROR NUM TO MAILBOX
690 001752 005767 176444 TST $SWREG ; CHECK FOR HALT ON ERROR
691 001756 100401 BMI 1$ ; CONTINUE IF SET
692 001760 000000 HALT ;<SIGNALS DID NOT CLEAR>
693 001762 1$:
694 001762 032767 002000 176432 BIT #BIT10,$SWREG ; CHECK FOR LOOP ON TEST
695 001770 001264 BNE TST2 ; GO TO LOOP ON TEST
696 001772 TST3:
697 001772 012767 000003 176404 MOV #3,$TESTN ; MOVE TEST NUMBER TO MAILBOX
698 002000 012777 177777 177214 MOV #-1,@DROBUF ; ALL ONES TO REGISTER
699 002006 017700 177210 MOV @DROBUF,RO
700 002012 022700 177777 CMP #-1,RO
701 002016 001416 BEQ 1$
702 002020 032767 040000 176374 BIT #BIT14,$SWREG ; CHECK FOR LOOP ON ERROR
703 002026 001361 BNE TST3 ; GO TO LOOP ERROR
704 002030 012767 000006 176344 MOV #6,$FATAL
705 002036 012767 000001 176334 MOV #1,$MSGTY ; MOVE ERROR NUM TO MAILBOX
706 002044 005767 176352 TST $SWREG ; CHECK FOR HALT ON ERROR
707 002050 100401 BMI 1$ ; CONTINUE IF SET
708 002052 000000 HALT ;<REGISTER WILL NOT HOLD ALL ONES>
709 002054 1$:
710 002054 032767 002000 176340 BIT #BIT10,$SWREG ; CHECK FOR LOOP ON TEST
711 002062 001343 BNE TST3 ; GO TO LOOP ON TEST
712
713 ;WRAP CABLE MUST BE INSTALLED TO EXECUTE THIS TEST
714 002064 TST4:
715 002064 012767 000004 176312 MOV #4,$TESTN ; MOVE TEST NUMBER TO MAILBOX
716 002072 032767 000400 176322 BIT #BIT8,$SWREG
717 002100 001031 BNE TST5 ; SKIP TEST IF NOT SELECTED
718 002102 012777 177777 177112 MOV #-1,@DROBUF
719 002110 000005 RESET ; SET DATA TO ALL ONES
720 002112 005777 177106 TST @DRIBUF ; REGISTER SHOULD CLEAR
721 002116 001416 BEQ 1$
722 002120 032767 040000 176274 BIT #BIT14,$SWREG ; CHECK FOR LOOP ON ERROR
723 002126 001356 BNE TST4 ; GO TO LOOP ERROR
724 002130 012767 000007 176244 MOV #7,$FATAL
725 002136 012767 000001 176234 MOV #1,$MSGTY ; MOVE ERROR NUM TO MAILBOX
726 002144 005767 176252 TST $SWREG ; CHECK FOR HALT ON ERROR
727 002150 100401 BMI 1$ ; CONTINUE IF SET
728 002152 000000 HALT ;<REGISTER DID NOT CLEAR BY RESET>
729 002154 1$:
730 002154 032767 002000 176240 BIT #BIT10,$SWREG ; CHECK FOR LOOP ON TEST
731 002162 001340 BNE TST4 ; GO TO LOOP ON TEST
732
733 TST5:
734 002164 012767 000005 176212 MOV #5,$TESTN ; MOVE TEST NUMBER TO MAILBOX

```


REGISTER DID NOT CLEAR BY RESET

SEQ 0015

```

735 002172 012777 052525 177022 MOV #52525,@DROBUF ;LOAD TEST DATA INTO BUFFER
736 002200 017700 177016 MOV @DROBUF,R0 ;COPY DATA FROM BUFFER TO R0
737 002204 022700 052525 CMP #52525,R0 ;COMPARE DATA
738 002210 001416 BEQ 1$ ;BR IF DATA IS CORRECT
739 002212 032767 040000 176202 BIT #BIT14,$SWREG ;CHECK FOR LOOP ON ERROR
740 002220 001361 BNE TST5 ;GO TO LOOP ERROR
741 002222 012767 000010 176152 MOV #10,$FATAL
742 002230 012767 000001 176142 MOV #1,$MSGTY ;MOVE ERROR NUM TO MAILBOX
743 002236 005767 176160 TST $SWREG ;CHECK FOR HALT ON ERROR
744 002242 100401 BMI 1$ ;CONTINUE IF SET
745 002244 000000 HALT ;<INCORRECT DATA IN REG>
746 002246 1$:
747 002246 032767 002000 176146 BIT #BIT10,$SWREG ;CHECK FOR LOOP ON TEST
748 002254 001343 BNE TST5 ;GO TO LOOP ON TEST
749
750 002256 TST6:
751 002256 012767 000006 176120 MOV #6,$TESTN ;MOVE TEST NUMBER TO MAILBOX
752 002264 012777 125252 176730 MOV #125252,@DROBUF ;LOAD TEST DATA INTO BUFFER
753 002272 017700 176724 MOV @DROBUF,R0 ;COPY DATA FROM BUFFER TO R0
754 002276 022700 125252 CMP #125252,R0 ;COMPARE DATA
755 002302 001416 BEQ 1$ ;BR IF DATA IS CORRECT
756 002304 032767 040000 176110 BIT #BIT14,$SWREG ;CHECK FOR LOOP ON ERROR
757 002312 001361 BNE TST6 ;GO TO LOOP ERROR
758 002314 012767 000011 176060 MOV #11,$FATAL
759 002322 012767 000001 176050 MOV #1,$MSGTY ;MOVE ERROR NUM TO MAILBOX
760 002330 005767 176066 TST $SWREG ;CHECK FOR HALT ON ERROR
761 002334 100401 BMI 1$ ;CONTINUE IF SET
762 002336 000000 HALT ;<INCORRECT DATA IN REG>
763 002340 1$:
764 002340 032767 002000 176054 BIT #BIT10,$SWREG ;CHECK FOR LOOP ON TEST
765 002346 001343 BNE TST6 ;GO TO LOOP ON TEST
766
767 ;TEST RELIABILITY OF DR11-C OUTPUT BUFFER REGISTER
768 002350 TST7:
769 002350 012767 000007 176026 MOV #7,$TESTN ;MOVE TEST NUMBER TO MAILBOX
770 002356 010502 BUFTST: MOV R5,R2 ;GET ADDRESS OF DRCSR
771 002360 005722 TST (R2)+ ;R2=ADDRESS OF OUTPUT BUFFER REG.
772 002362 005003 CLR R3 ;INITIALIZE DATA REGISTER
773 002364 010312 LP7: MOV R3,(R2) ;SEND THE DATA
774 002366 021203 CMP (R2),R3 ;CHECK THE RECEIVED DATA
775 002370 001004 BNE 5$ ;ERROR IF NOT THE SAME
776 002372 005203 INC R3 ;INCREMENT THE DATA
777 002374 105703 TSTB R3 ;CHECK FOR END OF DATA
778 002376 001417 BEQ 1$ ;CONTINUE IF END
779 002400 000771 BR LP7 ;GO TO SEND DATA IF NOT
780 002402 5$:
781 002402 032767 040000 176012 BIT #BIT14,$SWREG ;CHECK FOR LOOP ON ERROR
782 002410 001365 BNE LP7 ;GO TO LOOP ERROR
783 002412 012767 000012 175762 MOV #12,$FATAL
784 002420 012767 000001 175752 MOV #1,$MSGTY ;MOVE ERROR NUM TO MAILBOX
785 002426 005767 175770 TST $SWREG ;CHECK FOR HALT ON ERROR
786 002432 100401 BMI 1$ ;CONTINUE IF SET
787 002434 000000 HALT ;<DATA INCORRECT IN REG>
788 002436 1$:
789 002436 032767 002000 175756 BIT #BIT10,$SWREG ;CHECK FOR LOOP ON TEST
790 002444 001341 BNE TST7 ;GO TO LOOP ON TEST
  
```



```

;TEST THAT BYTE REFERENCE TO DROBUF AFFECT PROPER BYTE ONLY
791
792
793 002446          012767 000010 175730  TST10:
794 002446          012777 177777 176540  TAG:
795 002454          105077 176534          MOV #10,$TESTN      ; MOVE TEST NUMBER TO MAILBOX
796 002462          017700 176530          MOV #-1,@DROBUF    ; SET ALL ONES IN BUFFER
797 002466          022700 177400          CLR @DROBUF        ; CLEAR LOW BYTE
798 002472          001416          MOV @DROBUF,R0     ; COPY DATA
799 002476          032767 040000 175714  BEQ #177400,R0    ; VERIFY THAT LOW BYTE IS CLEAR
800 002500          001362          BIT #BIT14,$SWREG ; CHECK FOR LOOP ON ERROR
801 002510          000013 175664          BNE TAG           ; GO TO LOOP ERROR
802 002516          000001 175654          MOV #13,$FATAL
803 002524          100401          MOV #1,$MSGTY     ; MOVE ERROR NUM TO MAILBOX
804 002530          000000          TST $SWREG        ; CHECK FOR HALT ON ERROR
805 002532          002000 175660 1$:          BMI #1            ; CONTINUE IF SET
806 002534          001341          BIT #BIT10,$SWREG ; CHECK FOR LOOP ON TEST
807 002542          000011 175632          BNE TST10        ; GO TO LOOP ON TEST
808 002544          012767 000011 175632  TST11:
809 002552          012777 177777 176442  MOV #11,$TESTN   ; MOVE TEST NUMBER TO MAILBOX
810 002560          105077 176442          MOV #-1,@DROBUF  ; SET ALL ONES IN BUFFER
811 002564          017700 176432          CLR @DRBHI0     ; CLEAR HIGH BYTE
812 002570          022700 000377          MOV @DROBUF,R0  ; VERIFY THAT HIGH BYTE IS CLEAR
813 002574          001416          CMP #377,R0
814 002576          032767 040000 175616  BEQ #1           ; CHECK FOR LOOP ON ERROR
815 002604          001357          BIT #BIT14,$SWREG ; CHECK FOR LOOP ON ERROR
816 002606          012767 000014 175566  BNE TST11        ; GO TO LOOP ERROR
817 002614          012767 000001 175556  MOV #14,$FATAL
818 002622          005767 175574          MOV #1,$MSGTY   ; MOVE ERROR NUM TO MAILBOX
819 002626          100401          TST $SWREG       ; CHECK FOR HALT ON ERROR
820 002630          000000          BMI #1           ; CONTINUE IF SET
821 002632          032767 002000 175562 1$:          HALT             ; <HIGH BYTE FAILED TO CLEAR>
822 002632          001341          BIT #BIT10,$SWREG ; CHECK FOR LOOP ON TEST
823 002640          000012 175534          BNE TST11        ; GO TO LOOP ON TEST
824 002642          012767 000012 175534  TST12:
825 002642          000012 175534          MOV #12,$TESTN  ; MOVE TEST NUMBER TO MAILBOX

```

CVKAFE MACY11 30(1046) 22-NOV-82 10:43 PAGE 18
 CVKAFE.P11 22-NOV-82 10:37 ERROR 14

HIGH BYTE FAILED TO CLEAR

SEQ 0017

```

830 002650 005067 000110 CLR T12DAT ;CLEAR DATA LOCATION
831 002654 012704 002764 MOV #T12DAT,R4 ;STORE ADDRESS OF DATA LOCATION
832 002660 005077 176336 CLR @DROBUF ;CLEAR OUTPUT BUFFER
833 002664 105077 176336 T12CON: CLRB @DRBHIO ;CLEAR HIGH BYTE
834 002670 105277 176332 3$: INCB @DRBHIO ;INCREMENT HIGH BYTE
835 002674 105264 000001 INCB 1(R4) ;INCREMENT COMPARISON DATA
836 002700 027714 176316 CMP @DROBUF,(R4) ;COMPARE DATA
837 002704 001004 BNE 6$ ;BRANCH ON ERROR
838 002706 105764 000001 4$: TSTB 1(R4) ;FINISHED?
839 002712 001417 BEQ 1$ ;YES
840 002714 000765 BR 3$ ;CONTINUE TESTING
841 002716
842 002716 032767 040000 175476 6$: BIT #BIT14,$SWREG ; CHECK FOR LOOP ON ERROR
843 002724 001346 BNE TST12 ; GO TO LOOP ERROR
844 002726 012767 000015 175446 MOV #15,$FATAL
845 002734 012767 000001 175436 MOV #1,$MSGTY ; MOVE ERROR NUM TO MAILBOX
846 002742 005767 175454 TST $SWREG ; CHECK FOR HALT ON ERROR
847 002746 100401 BMI 1$ ; CONTINUE IF SET
848 002750 000000 HALT ;<DATA INCORRECT IN REG>
849 002752
850 002752 032767 002000 175442 1$: BIT #BIT10,$SWREG ; CHECK FOR LOOP ON TEST
851 002760 001330 BNE TST12 ; GO TO LOOP ON TEST
852 002762 000401 BR TST13
853 002764 000000 T12DAT: .WORD 0
;CONTROL STATUS REGISTER (DRCSR) TESTS.
854
855 002766 TST13:
856 002766 012767 000013 175410 MOV #13,$TESTN ; MOVE TEST NUMBER TO MAILBOX
857 002774 005015 CLR (R5) ;CLEAR STATUS REGISTER
858 002776 011500 MOV (R5),R0 ;COPY DATA
859 003000 032700 000143 BIT #143,R0 ;VERIFY THAT IE AND CSR BITS ARE CLEAR
860 003004 001416 BEQ T13CON ;IF YES, CONTINUE
861 003006 032767 040000 175406 BIT #BIT14,$SWREG ; CHECK FOR LOOP ON ERROR
862 003014 001364 BNE TST13 ; GO TO LOOP ERROR
863 003016 012767 000016 175356 MOV #16,$FATAL
864 003024 012767 000001 175346 MOV #1,$MSGTY ; MOVE ERROR NUM TO MAILBOX
865 003032 005767 175364 TST $SWREG ; CHECK FOR HALT ON ERROR
866 003036 100401 BMI 1$ ; CONTINUE IF SET
867 003040 000000 HALT ;<STATUS REG DID NOT CLEAR>
868 003042
869 003042 012715 000140 1$: T13CON: MOV #140,@R5 ;INTERRUPT ENABLE FOR A+B
870 003046 011500 MOV @R5,R0
871 003050 032700 000140 BIT #140,R0 ;INTERRUPT ENABLE BITS SET?
872 003054 001016 BNE 1$ ;CONTINUE IF YES
873 003056 032767 040000 175336 BIT #BIT14,$SWREG ; CHECK FOR LOOP ON ERROR
874 003064 001366 BNE T13CON ; GO TO LOOP ERROR
875 003066 012767 000017 175306 MOV #17,$FATAL
876 003074 012767 000001 175276 MOV #1,$MSGTY ; MOVE ERROR NUM TO MAILBOX
877 003102 005767 175314 TST $SWREG ; CHECK FOR HALT ON ERROR
878 003106 100401 BMI 1$ ; CONTINUE IF SET
879 003110 000000 HALT ;<ENABLE BITS NOT ON>
880 003112
881 003112 032767 002000 175302 1$: BIT #BIT10,$SWREG ; CHECK FOR LOOP ON TEST
882 003120 001322 BNE TST13 ; GO TO LOOP ON TEST
883
884 003122 TST14:
885 003122 012767 000014 175254 MOV #14,$TESTN ; MOVE TEST NUMBER TO MAILBOX

```

ENABLE BITS NOT ON

```

886 003130 012715 000140      MOV      #140,@R5      ;SET INTERRUPT ENABLE FLOPS
887 003134 000005              RESET              ;CLEAR THOSE FLOPS
888 003136 011500      MOV      @R5,R0      ;COPY CONTENTS OF DRCSR TO R0
889 003140 032700 000140      BIT      #140,R0     ;TEST INTERRUPT ENABLE BITS
890 003144 001416      BEQ      1$         ;BR IF CLEARED
891 003146 032767 040000 175246  BIT      #BIT14,$SWREG ; CHECK FOR LOOP ON ERROR
892 003154 001362      BNE      TST14      ; GO TO LOOP ERROR
893 003156 012767 000020 175216  MOV      #20,$FATAL
894 003164 012767 000001 175206  MOV      #1,$MSGTY   ; MOVE ERROR NUM TO MAILBOX
895 003172 005767 175224      TST      $SWREG     ; CHECK FOR HALT ON ERROR
896 003176 100401      BMI     1$         ; CONTINUE IF SET
897 003200 000000      HALT              ;<INTERRUPT ENABLE DID NOT CLEAR>
898 003202              1$:
899 003202 032767 002000 175212  BIT      #BIT10,$SWREG ; CHECK FOR LOOP ON TEST
900 003210 001344      BNE      TST14      ; GO TO LOOP ON TEST
901
902 003212              TST15:
903 003212 012767 000015 175164  MOV      #15,$TESTN  ; MOVE TEST NUMBER TO MAILBOX
904 003220 052715 000001      BIS      #1,@R5     ; SHOULD SET REQ A ALSO
905 003224 032715 000201      BIT      #201,@R5   ; VERIFY THAT REQ A IS SET
906 003230 001402      BEQ      5$         ; FLAG ERROR MESSAGE IF NO
907 003232 005015      CLR     @R5        ; CLEAR STATUS REGISTER
908 003234 000416      BR      1$         ; GO TO NEXT TEST
909 003236              5$:
910 003236 032767 040000 175156  BIT      #BIT14,$SWREG ; CHECK FOR LOOP ON ERROR
911 003244 001362      BNE      TST15      ; GO TO LOOP ERROR
912 003246 012767 000021 175126  MOV      #21,$FATAL
913 003254 012767 000001 175116  MOV      #1,$MSGTY   ; MOVE ERROR NUM TO MAILBOX
914 003262 005767 175134      TST      $SWREG     ; CHECK FOR HALT ON ERROR
915 003266 100401      BMI     1$         ; CONTINUE IF SET
916 003270 000000      HALT              ;<A REQ DID NOT SET>
917 003272              1$:
918

```


A REQ DID NOT SET

SEQ 0019

```

919 003272          TST16:
920 003272 012767 000016 175104  MOV #16,$TESTN      ; MOVE TEST NUMBER TO MAILBOX
921 003300 052715 000002          BIS #2,@R5          ; SHOULD SET REQ B
922 003304 032715 100002          BIT #100002,@R5    ; VERIFY THAT REQ B IS SET
923 003310 001402          BEQ 5$             ; FLAG ERROR MESSAGE IF NO
924 003312 005015          CLR @R5            ; CLEAR STATUS REGISTER
925 003314 000416          BR 1$             ; GO TO NEXT TEST
926 003316
927 003316 032767 040000 175076  5$: BIT #BIT14,$SWREG   ; CHECK FOR LOOP ON ERROR
928 003324 001362          BNE TST16         ; GO TO LOOP ERROR
929 003326 012767 000022 175046  MOV #22,$FATAL
930 003334 012767 000001 175036  MOV #1,$MSGTY      ; MOVE ERROR NUM TO MAILBOX
931 003342 005767 175054          TST $SWREG         ; CHECK FOR HALT ON ERROR
932 003346 100401          BMI 1$            ; CONTINUE IF SET
933 003350 000000          HALT              ; <B REQ DID NOT SET>
934 003352
935
936 003352          TST17:
937 003352 012767 000017 175024  MOV #17,$TESTN    ; MOVE TEST NUMBER TO MAILBOX
938 003360 106427 000340          MTPS #340         ; LOCK OUT INTERRUPTS
939 003364 052715 177777          BIS #-1,@R5       ; LOAD ALL ONES IN STATUS REGISTER
940 003370 022715 100343          CMP #100343,(R5) ; VERIFY THAT ALL WRITE BITS ARE SET IN DRCSR
941 003374 001416          BEQ T17CON        ; BR IF SET
942 003376 032767 040000 175016  BIT #BIT14,$SWREG ; CHECK FOR LOOP ON ERROR
943 003404 001362          BNE TST17         ; GO TO LOOP ERROR
944 003406 012767 000023 174766  MOV #23,$FATAL
945 003414 012767 000001 174756  MOV #1,$MSGTY      ; MOVE ERROR NUM TO MAILBOX
946 003422 005767 174774          TST $SWREG         ; CHECK FOR HALT ON ERROR
947 003426 100401          BMI 1$            ; CONTINUE IF SET
948 003430 000000          HALT              ; <INCORRECT DATA IN REG>
949 003432
950 003432 042715 000003          T17CON: BIC #3,@R5        ; CLEAR CSR BITS
951 003436 032715 000140          BIT #140,@R5     ; TEST INTERRUPT ENABLE BITS
952 003442 001016          BNE 1$            ; CONTINUE IF STILL SET
953 003444 032767 040000 174750  BIT #BIT14,$SWREG ; CHECK FOR LOOP ON ERROR
954 003452 001367          BNE T17CON        ; GO TO LOOP ERROR
955 003454 012767 000024 174720  MOV #24,$FATAL
956 003462 012767 000001 174710  MOV #1,$MSGTY      ; MOVE ERROR NUM TO MAILBOX
957 003470 005767 174726          TST $SWREG         ; CHECK FOR HALT ON ERROR
958 003474 100401          BMI 1$            ; CONTINUE IF SET
959 003476 000000          HALT              ; <WRONG BITS SET>
960 003500
961 003500 032767 002000 174714  1$: BIT #BIT10,$SWREG ; CHECK FOR LOOP ON TEST
962 003506 001321          BNE TST17         ; GO TO LOOP ON TEST
963
964 003510          TST20:
965 003510 012767 000020 174666  MOV #20,$TESTN    ; MOVE TEST NUMBER TO MAILBOX
966 003516 106427 000340          MTPS #340         ; LOCK OUT INTERRUPTS
967 003522 052715 000003          BIS #3,@R5        ; SET CSR BITS
968 003526 000005          RESET           ; SHOULD CLEAR INTERRUPT ENABLE FLOPS
969 003530 032715 000140          BIT #140,@R5     ; VERIFY THAT FLOPS ARE CLEARED
970 003534 001416          BEQ 1$            ; BR IF YES
971 003536 032767 040000 174656  BIT #BIT14,$SWREG ; CHECK FOR LOOP ON ERROR
972 003544 001361          BNE TST20         ; GO TO LOOP ERROR
973 003546 012767 000025 174626  MOV #25,$FATAL
974 003554 012767 000001 174616  MOV #1,$MSGTY      ; MOVE ERROR NUM TO MAILBOX
  
```

RESET DID NOT CLEAR BITS

SEQ 0020

```

975 003562 005767 174634          TST  $SWREG      ; CHECK FOR HALT ON ERROR
976 003566 100401          BMI  1$         ; CONTINUE IF SET
977 003570 000000          HALT                ;<RESET DID NOT CLEAR BITS>
978 003572          1$:
979 003572 032767 002000 174622    BIT  #BIT10,$SWREG ; CHECK FOR LOOP ON TEST
980 003600 001343          BNE  TST20      ; GO TO LOOP ON TEST
981
982          ;NOTE: THE WRAP CABLE MUST BE INSTALLED TO EXECUTE
983          ;TESTS 21-27
984 0036J2          TST21:
985 003602 012767 000021 174574    MOV  #21,$TESTN  ; MOVE TEST NUMBER TO MAILBOX
986 003610 032767 000400 174604    BIT  #BIT8,$SWREG
987 003616 001402          BEQ  LP21       ; DO TESTS IF NOT INHIBITED
988 003620 000167 000710          JMP  TST999     ; IF INHIBITED
989 003624 005015          LP21: CLR  @R5        ; CLEAR STATUS REGISTER
990 003626 005215          INC  @R5        ; SET CSRO
991 003630 105715          TSTB @R5       ; CHECK FOR REQ A FLAG
992 003632 100416          BMI  1$         ; BR IF SET
993 003634 032767 040000 174560    BIT  #BIT14,$SWREG ; CHECK FOR LOOP ON ERROR
994 003642 001357          BNE  TST21     ; GO TO LOOP ERROR
995 003644 012767 000026 174530    MOV  #26,$FATAL
996 003652 012767 000001 174520    MOV  #1,$MSGTY  ; MOVE ERROR NUM TO MAILBOX
997 003660 005767 174536          TST  $SWREG    ; CHECK FOR HALT ON ERROR
998 003664 100401          BMI  1$         ; CONTINUE IF SET
999 003666 000000          HALT                ;<BIT0 DID NOT SET BIT7>
1000 003670          1$:
1001 003670 032767 002000 174524    BIT  #BIT10,$SWREG ; CHECK FOR LOOP ON TEST
1002 003676 001341          BNE  TST21     ; GO TO LOOP ON TEST
1003

```

BIT0 DID NOT SET BIT7

SEQ 0021

```

1004 003700          TST22:
1005 003700 012767 000022 174476  MOV #22,$TESTN      ; MOVE TEST NUMBER TO MAILBOX
1006 003706 012715 000002          MOV #2,@R5          ; SET CSR1
1007 003712 005715          TST @R5              ; CHECK FOR REQ B FLAG
1008 003714 100416          BMI 1$              ; BR IF SET
1009 003716 032767 040000 174476  BIT #BIT14,$SWREG  ; CHECK FOR LOOP ON ERROR
1010 003724 001365          BNE TST22          ; GO TO LOOP ERROR
1011 003726 012767 000027 174446  MOV #27,$FATAL
1012 003734 012767 000001 174436  MOV #1,$MSGTY      ; MOVE ERROR NUM TO MAILBOX
1013 003742 005767 174454          TST $SWREG         ; CHECK FOR HALT ON ERROR
1014 003746 100401          BMI 1$              ; CONTINUE IF SET
1015 003750 000000          HALT                ; <BIT1 DID NOT SET BIT15>
1016 003752
1017 003752 032767 002000 174442  1$: BIT #BIT10,$SWREG ; CHECK FOR LOOP ON TEST
1018 003760 001347          BNE TST22          ; GO TO LOOP ON TEST
1019 003762
1020 003762 012767 000023 174414  TST23: MOV #23,$TESTN      ; MOVE TEST NUMBER TO MAILBOX
1021 003770 012715 000003          MOV #3,@R5          ; CSRO AND CSR1
1022 003774 011500          MOV (R5),R0         ; COPY DATA
1023 003776 022700 100203          CMP #100203,R0      ; COMPARE DATA
1024 004002 001416          BEQ 1$              ; BR IF GOOD DATA IS RECEIVED
1025 004004 032767 040000 174410  BIT #BIT14,$SWREG  ; CHECK FOR LOOP ON ERROR
1026 004012 001363          BNE TST23          ; GO TO LOOP ERROR
1027 004014 012767 000030 174360  MOV #30,$FATAL
1028 004022 012767 000001 174350  MOV #1,$MSGTY      ; MOVE ERROR NUM TO MAILBOX
1029 004030 005767 174366          TST $SWREG         ; CHECK FOR HALT ON ERROR
1030 004034 100401          BMI 1$              ; CONTINUE IF SET
1031 004036 000000          HALT                ; <INCORRECT BITS SET IN REG>
1032 004040
1033 004040 032767 002000 174354  1$: BIT #BIT10,$SWREG ; CHECK FOR LOOP ON TEST
1034 004046 001345          BNE TST23          ; GO TO LOOP ON TEST
1035
1036
1037 004050          ; CAN WE RAISE INTERRUPT "A"
1038 004050 012767 000024 174326  TST24: MOV #24,$TESTN      ; MOVE TEST NUMBER TO MAILBOX
1039 004056 106427 000340          MTPS #340          ; LOCK OUT INTERRUPTS
1040 004062 012706 001200          MOV #STKPTR,SP      ; INITIALIZE STACK POINTER
1041 004066 012777 004110 175134  MOV #4,@DRVECA      ; INTERRUPT RETURN POINTER
1042 004074 012715 000101          MOV #101,@R5        ; INTERRUPT ENABLE AND CSRO
1043 004100 106427 000000          MTPS #0             ; ALLOW INTERRUPTS
1044 004104 000240          NOP
1045 004106 000402          BR 5$              ; NO INTERRUPT
1046
1047 004110 005015          4$: CLR @R5          ; CLEAR INTERRUPT ENABLE
1048 004112 000416          BR 1$              ; GO TO NEXT TEST
1049 004114
1050 004114 032767 040000 174300  5$: BIT #BIT14,$SWREG ; CHECK FOR LOOP ON ERROR
1051 004122 001352          BNE TST24          ; GO TO LOOP ERROR
1052 004124 012767 000031 174250  MOV #31,$FATAL
1053 004132 012767 000001 174240  MOV #1,$MSGTY      ; MOVE ERROR NUM TO MAILBOX
1054 004140 005767 174256          TST $SWREG         ; CHECK FOR HALT ON ERROR
1055 004144 100401          BMI 1$              ; CONTINUE IF SET
1056 004146 000000          HALT                ; <NO A INTERRUPT>
1057 004150
1058 004150 032767 002000 174244  1$: BIT #BIT10,$SWREG ; CHECK FOR LOOP ON TEST
1059 004156 001334          BNE TST24          ; GO TO LOOP ON TEST

```



```

1060
1061
1062 004160
1063 004160 012767 000025 174216
1064 004166 012706 001200
1065 004172 106427 000340
1066 004176 012777 004220 175030
1067 004204 012715 000042
1068 004210 106427 000000
1069 004214 000240
1070 004216 000402
1071 004220 005015
1072 004222 000416
1073 004224
1074 004224 032767 040000 174170
1075 004232 001352
1076 004234 012767 000032 174140
1077 004242 012767 000001 174130
1078 004250 005767 174146
1079 004254 100401
1080 004256 000000
1081 004260
1082 004260 032767 002000 174134
1083 004266 001334
1084
1085
1086 004270
1087 004270 012767 000026 174106
1088 004276 017702 174730
1089 004302 016777 174736 174722
1090 004310 012706 001200
1091 004314 012777 004370 174706
1092 004322 012715 000101
1093 004326 106427 000000
1094 004332 000240
1095 004334
1096 004334 032767 040000 174060
1097 004342 001352
1098 004344 012767 000033 174030
1099 004352 012767 000001 174020
1100 004360 005767 174036
1101 004364 100401
1102 004366 000000
1103 004370
1104 004370 032767 002000 174024
1105 004376 001341
1106 004400 005015
1107 004402 010277 174624
1108
1109 004406
1110 004406 012767 000027 173770
1111 004414 005000
1112 004416 010077 174600
1113 004422 027700 174576
1114 004426 001020
1115 004430 005200

:RAISE INTERRUPT 'B'
TST25:
MOV #25,$TESTN ; MOVE TEST NUMBER TO MAILBOX
MOV #STKPTR,SP ; INITIALIZE STACK POINTER
MTPS #340 ; LOCK OUT INTERRUPTS
MOV #4$,@DRVECB ; INTERRUPT RETURN POINTER
MOV #42,@R5 ; IE AND CSR1
MTPS #0 ; ALLOW INTERRUPTS
NOP
BR 5$ ; NO INTERRUPT
4$: CLR @R5 ; CLEAR INTERRUPT ENABLE
BR 1$ ; GO TO NEXT TEST
5$: BIT #BIT14,$SWREG ; CHECK FOR LOOP ON ERROR
BNE TST25 ; GO TO LOOP ERROR
MOV #32,$FATAL
MOV #1,$MSGTY ; MOVE ERROR NUM TO MAILBOX
TST $SWREG ; CHECK FOR HALT ON ERROR
BMI 1$ ; CONTINUE IF SET
HALT ; <NO B INTERRUPT>
1$: BIT #BIT10,$SWREG ; CHECK FOR LOOP ON TEST
BNE TST25 ; GO TO LOOP ON TEST

:TEST FOR INTERRUPT FROM DEVICE
TST26:
MOV #26,$TESTN ; MOVE TEST NUMBER TO MAILBOX
MOV @DRLVLA,R2 ; SAVE INTERRUPT PSW
LP26: MOV PL,@DRLVLA ; LOCK OUT SUCCESSIVE INTERRUPTS
MOV #STKPTR,SP ; INITIALIZE STACK POINTER
MOV #1$,@DRVECA ; INTERRUPT RETURN POINTER
MOV #101,@R5 ; SET INTERRUPT ENABLE-AND CSRO
MTPS #0 ; ALLOW INTERRUPTS
NOP
5$: BIT #BIT14,$SWREG ; CHECK FOR LOOP ON ERROR
BNE TST26 ; GO TO LOOP ERROR
MOV #33,$FATAL
MOV #1,$MSGTY ; MOVE ERROR NUM TO MAILBOX
TST $SWREG ; CHECK FOR HALT ON ERROR
BMI 1$ ; CONTINUE IF SET
HALT ; <NO DEVICE INTERRUPT>
1$: BIT #BIT10,$SWREG ; CHECK FOR LOOP ON TEST
BNE LP26 ; GO TO LOOP ON TEST
CLR @R5 ; CLEAR INTERRUPT ENABLE
MOV R2,@DRLVLA ; RESTORE INTERRUPT PSW

:PLU WRAP TEST
TST27:
MOV #27,$TESTN ; MOVE TEST NUMBER TO MAILBOX
CLR R0 ; SET UP STARTING DATA
WLOOP: MOV R0,@DROBUF ; SEND DATA
CMP @DRIBUF,R0 ; CHECK THE DATA
BNE 5$ ; ERROR IF NOT RIGHT
INC R0 ; CHANGE DATA
  
```

NO DEVICE INTERRUPT

SEQ 0023

```
1116 004432 001434 BEQ 1$ ;NEXT TEST IF END
1117 004434 022700 031460 3$: CMP #31460,R0 ; CHECK FOR TEST MODULE CODE
1118 004440 001411 BEQ 4$
1119 004442 022700 031461 CMP #31461,R0
1120 004446 001406 BEQ 4$
1121 004450 022700 031462 CMP #31462,R0
1122 004454 001403 BEQ 4$
1123 004456 022700 031463 CMP #31463,R0
1124 004462 001355 BNE WLOOP
1125 004464 005200 4$: INC R0
1126 004466 000762 BR 3$ ; RECHECK DATA CODE
1127 004470 5$:
1128 004470 032767 040000 173724 BIT #BIT14,$SWREG ; CHECK FOR LOOP ON ERROR
1129 004476 001347 BNE WLOOP ; GO TO LOOP ERROR
1130 004500 012767 000034 173674 MOV #34,$FATAL
1131 004506 012767 000001 173664 MOV #1,$MSGTY ; MOVE ERROR NUM TO MAILBOX
1132 004514 005767 173702 TST $SWREG ; CHECK FOR HALT ON ERROR
1133 004520 100401 BMI 1$ ; CONTINUE IF SET
1134 004522 000000 HALT ;<WRAP DATA DID NOT COMPARE>
1135 004524 1$:
1136 004524 032767 002000 173670 BIT #BIT10,$SWREG ; CHECK FOR LOOP ON TEST
1137 004532 001325 BNE TST27 ; GO TO LOOP ON TEST
```

```

1138
1139
1140 004534          TST999:
1141 004534 005237 000406      INC      @#$PASS      ; INCREMENT PASS COUNT
1142 004540 132767 000040 173653 BITB     #40,$ENVM    ; WILL APT ALLOW PRINTING?
1143 004546 001010          BNE     ACT          ; NO
1144 004550 012700 004654      MOV     #MSG,R0     ; GET MESSAGE ADDRESS
1145 004554 105737 177564      WAIT:  TSTB    @#TPS  ; CHECK IF TTY READY
1146 004560 100375          BPL     WAIT        ; IF NOT
1147 004562 112037 177566      MOVB   (R0)+,@#TPB ; PRINT THE CHARACTER
1148 004566 001372          BNE     WAIT        ; NEXT IF NOT DONE
1149 004570 013700 000042      ACT:   MOV     @#42,R0 ; CHECK ACT
1150 004574 001405          BEQ    GOAGIN       ; KEEP GOING
1151 004576 000005          RESET
1152 004600 004710          $ENDAD: JSR    PC,(R0) ; ACT HOOKS
1153 004602 000240          NOP
1154 004604 000240          NOP
1155 004606 000240          NOP
1156 004610 000167 174500      GOAGIN: JMP     AROUND ;+ DO ANOTHER PASS
1157                                     ;+ ON PASSES >1 THE TITLE PRINTOUT WILL BE SUPPRESSED
1158
1159 004614          .EVEN
1160                                     STORE:
1161                                     TPS=177564
1162                                     TPB=177566
1162 004614 177777          PASSPT: -1
1163 004616 053103 040513 026506      TITLE1: .ASCIZ .CVKAF-E0  DRV11 DIAGNOSTIC. <15><12>
1164 004624 030105 020040 042040
1165 004632 053122 030461 042040
1166 004640 040511 047107 051517
1167 004646 044524 006503 000012
1168 004654 047105 020104 043117      MSG:   .ASCIZ  .END OF PASS.<15><12>
1169 004662 050040 051501 006523
1170 004670 000012
1171                                     .EVEN

```



```

1172
1173           ;ENTER HERE FOR POWER FAIL
1174
1175 004672 010046          PFAIL:  MOV    %0,-(6)           ;SAVE REGISTER OR STACK
1176 004674 010146          MOV    %1,-(6)           ;WHEN POWERING DOWN
1177 004676 010246          MOV    %2,-(6)
1178 004700 010346          MOV    %3,-(6)
1179 004702 010446          MOV    %4,-(6)
1180 004704 010546          MOV    %5,-(6)
1181 004706 016746 173112  MOV    24,-(6)
1182 004712 010637 004726  MOV    %6,@#SAVR6       ;STORE STACK POSITION
1183 004716 012737 004730 000024  MOV    #RESTAR,@#24
1184 004724 000000          HALT
1185 004726 000000          SAVR6: 0
1186 004730 016706 177772  RESTAR:MOV  SAVR6,%6
1187 004734 012667 173064  MOV    (6)+,%24
1188 004740 012605          MOV    (6)+,%25
1189 004742 012604          MOV    (6)+,%24
1190 004744 012603          MOV    (6)+,%23
1191 004746 012602          MOV    (6)+,%2
1192 004750 012601          MOV    (6)+,%1
1193 004752 012600          MOV    (6)+,%0
1194 004754 000137 001256  JMP    @#START
1195 000001          .END

```

ABASE = 000000	495		
ACDW1 = 000000	495		
ACDW2 = 000000	495		
ACPUOP = 000000	495	510	
ACT = 004570	1143	1149#	
ADDW0 = 000000	495		
ADDW1 = 000000	495		
ADDW10 = 000000	495		
ADDW11 = 000000	495		
ADDW12 = 000000	495		
ADDW13 = 000000	495		
ADDW14 = 000000	495		
ADDW15 = 000000	495		
ADDW2 = 000000	495		
ADDW3 = 000000	495		
ADDW4 = 0C0000	495		
ADDW5 = 000000	495		
ADDW6 = 000000	495		
ADDW7 = 000000	495		
ADDW8 = 000000	495		
ADDW9 = 000000	495		
ADEVCT = 000000	495	501	
ADEVN = 000000	495		
AENV = 000000	495	506	
AENVN = 000000	495	507	
AFATAL = 000000	495	498	
AMADR1 = 000000	495		
AMADR2 = 000000	495		
AMADR3 = 000000	495		
AMADR4 = 000000	495		
AMAMS1 = 000000	495		
AMAMS2 = 000000	495		
AMAMS3 = 000000	495		
AMAMS4 = 000000	495		
AMSGAD = 000000	495	503	
AMSGLG = 000000	495	504	
AMSGTY = 000000	495	497	
AMTYP1 = 000000	495		
AMTYP2 = 000000	495		
AMTYP3 = 000000	495		
AMTYP4 = 000000	495		
APASS = 000000	495	500	
APRIOR = 000000	495		
AROUND = 001314	583	595#	1156
ASWREG = 000000	495	508	
ATESTN = 000000	495	499	
AUNIT = 000000	495	502	
AUSWR = 000000	495	509	
AVECT1 = 000000	495		
AVECT2 = 000000	495		
BIT0 = 000001	405#	660	
BIT00 = 000001	395#	405	
BIT01 = 000002	394#	404	
BIT02 = 000004	393#	403	
BIT03 = 000010	392#	402	
BIT04 = 000020	391#	401	

CVKAFE MACY11 30(1046) 22-NOV-82 10:43 PAGE 35
 CVKAFE.P11 22-NOV-82 10:37 CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0032

.\$POWE	1#	
.\$SRAND	1#	
.\$SRDDE	1#	
.\$SRDOC	1#	
.\$SREAD	1#	
.\$SR2AZ	1#	
.\$SSAVE	1#	
.\$SSB2D	1#	
.\$SSB2O	1#	
.\$SSCOP	1#	
.\$SSIZE	1#	
.\$SSUPR	1#	
.\$STRAP	1#	310#
.\$STYPB	1#	
.\$STYPD	1#	
.\$STYPE	1#	310#
.\$STYPO	1#	
.\$S4OCA	1#	
.\$1170	1#	

. ABS. 004760 000

ERRORS DETECTED: 0

CVKAFE.BIN, CVKAFE.LST/CRF/SOL/NL: TOC=SYSMAC.SML, CVKAFE.P11
 RUN-TIME: 97.5 SECONDS
 RUN-TIME RATIO: 91/18=4.9
 CORE USED: 35K (70 PAGES)