

DRV11J

DRV11J DIAG TEST PRT1
CVDRC00

COPYRIGHT (c) 1979-83
AH-F7580-MC
FICHE 01 OF 01

JUL 1984
digital
Made In USA

Table with multiple columns and rows of data, likely a diagnostic test report. The text is extremely faint and illegible due to the low contrast of the scan. The table appears to be organized into several vertical columns, each containing multiple rows of data points or test results.

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

.REM #

IDENTIFICATION

PRODUCT CODE: AC-F756C-MC
PRODUCT NAME: CVDRCCO DRV11J DIAG TST PRT1
DATE CREATED: AUGUST 1983
AUTHOR: BILL HEAVEY
MAINTAINER: DIAGNOSTIC ENGINEERING

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

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

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

COPYRIGHT (C) 1979 DIGITAL EQUIPMENT CORPORATION

52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92

TABLE OF CONTENTS

- 1.0 ABSTRACT
- 2.0 REQUIREMENTS
 - 2.1 EQUIPMENT
 - 2.2 STORAGE
- 3.0 LOADING PROCEDURE
- 4.0 STARTING PROCEDURE
 - 4.1 PROGRAM START
- 5.0 SOFTWARE SWITCH REGISTER
 - 5.1 OPTIONS
 - 5.2 CONTROL
- 6.0 ERROR REPORTING
 - 6.1 ERROR COMMENT
 - 6.2 ERROR DATA
- 7.0 MISCELLANEOUS
 - 7.1 DRV11J BUS ADDRESS MODIFICATION
 - 7.2 XXDP/APT NOTES
 - 7.3 POWER FAIL
 - 7.4 MULTIPLE DRV11J INTERFACE TESTING
 - 7.5 RESTRICTIONS
- 8.0 EXECUTION TIME
- 9.0 PROGRAM TEST DESCRIPTIONS
- 10.0 LISTING

151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207

USE STANDARD PROCEDURE FOR PDP-11 ABSOLUTE
BINARY FORMATTED PAPERTAPES OR XXDP MEDIA
(FILES WITH .BIC OR .BIN EXTENSIONS ONLY).

4.0 STARTING PROCEDURE

1. MAKE SURE THE DRV11-J CABLE IS INSERTED WITH 1/2 A TWIST ON THE I/O CONNECTORS OF THE DRV11-J OPTION. THIS WILL CONNECT PORT A TO PORT C AND PORT B TO PORT D.
 2. MAKE SURE THE DEVICE BUS ADDRESSES AGREE WITH THE DEFAULT VALUES DEFINED IN SECTION 7.1. IF NOT, CHANGE LOCATION(S) AS DESIRED VIA THE 'ADDRESS/' ODT COMMAND.
 3. THE PROGRAM SHOULD ALWAYS BE STARTED AT 200. STARTING AT 200(200G OR .R CVDRCC UNDER XXDP+), THE PROGRAM INITIALIZES ITSELF, PRINTS ITS ID(FIRST TIME ONLY) AND THEN PRINTS THAT THE DRV11-J CABLE IS REQUIRED(FIRST TIME ONLY) AND THEN PRINTS: SWR=XXXXXX NEW=
- WHERE XXXXXX REPRESENTS THE CURRENT VALUE OF THE SOFTWARE SWITCH REGISTER. IF NO CHANGES ARE REQUIRED IN THE SWITCH REGISTER THEN JUST HIT CARRIAGE RETURN. IF CHANGES ARE REQUIRED, THEN A NEW VALUE MAY BE TYPED FOLLOWED BY A CARRIAGE RETURN. REFER TO SECTION 5.0 FOR SWITCH REGISTER OPTIONS.
4. IF THE FOLLOWING ERROR OCCURS RIGHT AFTER STARTUP IT IS POSSIBLE THAT THE DRV11J LOOPBACK CABLE MAY NOT BE INSTALLED OR WAS NOT INSTALLED PROPERLY:

```

REG READ/WRITE ER
ERRPC  TSTNUM  BUSADR  EXPCT  RCVD
002224 000002 *164160 100700 000700
*BUSADR MAY BE DIFFERENT DEPENDING ON THE CSR OF THE DRV11J.

```

5.0 SOFTWARE SWITCH REGISTER

5.1 OPTIONS

THE PROGRAM SWITCH DEFAULT MODE IS SWR = 000000
IF USING A VIDEO TERMINAL ,BIT 15 = 1(HALT ON ERROR)
MAY BE HELPFUL IN KEEPING THE ERROR ON THE SCREEN.

SWITCH	OCTAL	FUNCTION
SW15=1	100000	HALT ON ERROR
SW14=1	040000	LOOP ON TEST
SW13=1	020000	INHIBIT ERROR TYPEOUTS
SW11=1	004000	INHIBIT ITERATIONS
SW09=1	001000	LOOP ON ERROR
SW08=1	0004XX	LOOP ON TEST IN SWR <7-0>

208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224

5.2 CONTROL

1. THE SOFTWARE SWITCH REGISTER 'SWREG' (LOC. 176) CAN BE CHANGED BY USING THE ODT FACILITIES.
2. THE SOFTWARE SWITCH REGISTER CAN BE CHANGED UNDER PROGRAM CONTROL BY TYPING THE 'CONTROL & G' KEYS. THIS KEYBOARD OPERATION WILL PRINT OUT THE CURRENT CONTENTS AND ACCEPT NEW OCTAL SWITCH REGISTER DATA TERMINATED WITH A CARRIAGE RETURN.
3. ONCE THE ODT MODE HAS BEEN ENTERED BECAUSE OF AN ERROR CONDITION WITH BIT15 SET (HALT ON ERROR), STEP #2 ABOVE IS OF NO VALUE, SO RESORT TO STEP #1 TO ALTER THE SOFTWARE SWITCH REGISTER IF DESIRED BEFORE TYPING 'P' (CONTINUE).

226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270

6.0 ERROR REPORTING

6.1 ERROR COMMENT

ALL ERRORS ARE ACCOMPANIED WITH AN ENGLISH LANGUAGE DESCRIPTIVE COMMENT AS TO THE TYPE OF FAILURE. FURTHER QUALIFICATION OF THE ERROR CAN BE OBTAINED IF NEEDED FROM THE COMMENT AT THE ERROR PC OR FROM THE TEST ITSELF.

TO CONSERVE MEMORY SPACE FOR THE 4K PROGRAM MEMORY REQUIREMENT, REGISTER 5 (R5) IS RESERVED FOR THE \$BDDAT LOCATION (1126).

EXAMPLE: CMP \$GDDAT,(R5) IS THE SAME AS CMP \$GDDAT,\$BDDAT.

6.2.1 ERROR DATA

ERROR TITLE HEADING

ERRPC	TSTNUM	BUSADR	EXPCT	RCVD
XXXXXX	XXXXXX	XXXXXX	XXXXXX	XXXXXX

ERRPC	TSTNUM	BUSADR	EXPCT	RCVD

LISTING ADDRESS WHERE THE ERROR WAS DETECTED
TEST NUMBER WHERE THE ERROR OCCURRED
DRV11J BUS REG ADDRESS OF CONCERNED OPERATION
DATA THAT WAS EXPECTED
DATA THAT WAS RECEIVED

6.2.2 ERROR TITLES

REG TIMEOUT ER	;REGISTER TIMEOUT ERROR
REG READ/WRITE ER	;REGISTER READ/WRITE ERROR
IRR REG ER	;INTERRUPT REQUEST REGISTER ERROR
ACR REG ER	;AUTOCLEAR REGISTER ERROR
IMR REG ER	;INTERRUPT MASK REGISTER ERROR
ISR REG ER	;INTERRUPT SERVICE REGISTER ERROR
CHIP STAT ER	;CHIP STATUS ERROR

272 7.0 MISCELLANEOUS
273 -----
274
275 7.1 DRV11-J BUS ADDRESS MODIFICATION
276
277 MODIFY LOCATION '\$BASE' (ADDR: 1244) IF BASE BUS ADDRESS
278 IS NOT 164160.
279
280
281 7.2 XXDP/APT NOTES
282
283 THIS DIAGNOSTIC DOES SUPPORT ACT AND APT ENVIRONMENTS.
284
285 7.3 POWER FAIL
286
287 A POWER FAILURE WILL CAUSE A RESTART MESSAGE ON POWER UP AT
288 WHICH TIME THE PROGRAM IS RESTARTED (ONLY ON SYSTEMS WITH
289 NON-VOLATILE MEMORY AND WITH APPROPRIATE HARDWARE).
290
291 7.4 MULTIPLE DRV11J INTERFACE TESTING
292
293 THIS PROGRAM DOES NOT "AUTO-SIZE" THE NUMBER OF DRV11J'S CONNECTED.
294 THIS DIAGNOSTIC WILL TEST SEQUENTIALLY UP TO 4 DRV11J INTERFACES
295 WITH CONTIGUOUS BUS ADDRESSES. THIS IS ACCOMPLISHED
296 BY THE USER SETTING UP LOCATION 'DEVM' (ADDR: 1246)
297 WITH A BIT MAP INDICATING WHAT INTERFACES ARE TO BE TESTED.
298 I.E. BIT0 = 1 SAYS TEST 1ST DRV11J,
299 BIT1 = 1 SAYS TEST 2ND DRV11J
300 BIT2 = 1 SAYS TEST 3RD DRV11J
301 BIT3 = 1 SAYS TEST 4TH DRV11J
302
303 1ST UNIT = STARTING CSRA 164160 \$DEVM = 1
304 2ND UNIT = STARTING CSRA 164140 \$DEVM = 3
305 3RD UNIT = STARTING CSRA 164120 \$DEVM = 7
306 4TH UNIT = STARTING CSRA 164100 \$DEVM = 17
307
308 7.5 RESTRICTIONS
309
310
311 8.0 EXECUTION TIME
312 -----
313
314 EXECUTION TIME RANGES FROM ABOUT <5 SECONDS ON FIRST
315 PASS WITH NO ITERATIONS TO <20 SECONDS WITH ITERATIONS
316 ENABLED AFTER FIRST PASS,PER DRV11J UNIT CONNECTED.
317 AN "END PASS" MESSAGE INDICATES ALL TESTS HAVE COMPLETED
318 ON ALL SELECTED UNITS.

320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376

9.0 PROGRAM TEST DESCRIPTIONS

GENERAL

THIS DIAGNOSTIC CONTAINS A SERIES OF INDEPENDENT TESTS DESIGNED TO TEST LOGIC FUNCTIONS AND DATA PATHS OF THE DRV11J OPTION. TESTING IS ACCOMPLISHED WITH THE AID OF THE DRV11J LOOP BACK CABLE PROVIDED FOR DIAGNOSTIC TESTING. A COMPLETE LIST OF TESTS IS AVAILABLE IN THE TABLE OF CONTENTS AT THE BEGINNING OF THE LISTING. THE COMMENT FIELD WITHIN EACH TEST CAN BE BENEFICIAL IN TEST UNDERSTANDING.

TESTING WITH THE DRV11J CABLE ALLOWS FOR TESTING OF PORT A WITH PORT C AND THE TESTING OF PORT B WITH PORT D.

9.1 TESTS T1-T22 REGISTER CHECKING

THESE TESTS WILL WRITE/READ PORTS A TO C AND WRITE/READ PORTS B TO D WITH FLOATING ONES AND FLOATING ZEROS, GROWING ONES AND GROWING ZEROS AND DATA PATTERNS. TESTS ARE PERFORMED TO INSURE INTERACTION WITH CONNECTED PORTS AND NO INTERACTION BETWEEN UNCONNECTED PORTS.

9.2 T22-T42 INTERRUPT CONTROL REGISTER CHECKING

TESTS ARE MADE TO THE INTERRUPT CONTROL CHIP REGISTERS IRR, ACR, IMR AND LIMITED TESTING OF ISR REGISTER UNTIL INTERRUPTS ARE PERFORMED. TESTS ARE PERFORMED ON THE REGISTERS WITH FLOATING ONES, FLOATING ZEROS, GROWING ONES AND GROWING ZEROS. CHIP RESET CAPABILITIES ARE TESTED AS WELL AS UNIQUENESS OF REGISTERS WITHIN A CHIP GROUP AND THE UNIQUENESS OF ONE GROUP'S REGISTERS TO ANOTHER GROUP'S REGISTERS.

9.3 TEST 43 - TEST STATUS BITS GINT, S2, S1, S0, GP1, GP2

EXERCISE THE STATUS BITS S2, S1, S0 AND GINT FOR EACH GROUP CHIP BY SETTING SINGLE IRR BITS TO CAUSE THE STATUS BITS TO GO FROM 120 TO 127. EACH IRR BIT IS THEN MASKED TO RETURN STATUS BACK TO ORIGINAL STATUS WITH NO IRR BITS SET.

9.4 T44-T47 POLLED MODE TESTING

THIS TEST WILL WRITE PATTERNS INTO DBRA WITH EITHER LOW TO HIGH OR HIGH TO LOW POLARITIES. AFTER PLACING ALL ONES IN DBRA AND THEN SELECTING ACTIVE LOW, CLEARING DBRA WILL NOW SET IRR BITS 0-7, GROUP 1 AND IRR BITS 0-3, GROUP 2.

377 THE RPLY SIGNALS WILL SET IRR BITS 4-7
378 WHEN WRITING DBRA IN OUTPUT MODE, THIS WILL
379 SET THE RPLY FOR DBRC.(IRR6, GROUP 2)
380 WHEN READING DBRC IN INPUT MODE WILL SET
381 THE RPLY FOR DBRA.(IRR BIT 4, GROUP 2)
382 WHEN WRITING DBRB IN OUTPUT MODE, THIS WILL
383 SET THE RPLY FOR DBRD(IRR BIT 7, GROUP 2)
384 WHEN READING DBRD IN INPUT MODE, THIS WILL
385 SET THE RPLY FOR DBRB(IRR BITS 5, GROUP 2).
386 TEST ALL DATA PATTERNS TO SET IRR BITS
387 GROUP 1 AND GROUP 2 AND FOR THE SETTING OF RPLY
388 BITS IN THE GROUP 2 IRR REGISTER BY WRITING IN
389 OUTPUT MODE AND READING IN INPUT MODE.
390 TEST THAT NO RPLY BITS SET WHEN READING IN OUTPUT
391 MODE AND WHEN WRITING IN INPUT MODE.
392
393 9.5 T50-T51 CSR'S WITH RESET
394 SET UP CSR'S IN OUTPUT MODE AND CLEAR
395 DIRECTION BIT OUT OF EACH CSR EXCEPT FOR
396 CSRA WHICH WILL CLEAR DIR BIT AND I/E BIT.
397
398 10.0 LISTING
399 -----
400 *

```

412      ;;GPA  .HEADER +/CVDRCA  DRV11J DIAG TST PRT1/,1979,+/BILL HEAVEY/
413      ;;JRS  .HEADER +/CVDRCB  DRV11J DIAG TST PRT1/,1979,+/BILL HEAVEY/
414      .TITLE  CVDRCC  DRV11J DIAG TST PRT1
          ;*COPYRIGHT (C) 1979
          ;*DIGITAL EQUIPMENT CORP.
          ;*MAYNARD, MASS. 01754
          ;*
          ;*PROGRAM BY BILL HEAVEY
          ;*
          ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
          ;*PACKAGE (MAINDEC-11-DZQAC-C7), JUL, 1982.
          ;*
          $TN=1
          $SWR=160000      ;;HALT ON ERROR, LOOP ON TEST, INHIBIT ERROR TYP0UT
          ;* EDITED TO PERMIT DRV'S TO RUN ON FALCON (KXT11).      ;;GPA
          ;*              G. PASQUANTONIO, JULY '81                ;;GPA
          ;*
          ;* REVISD TO COMPENSATE FOR LONG CABLES & FILTERS      ;;JRS
          ;*              J. SUTTON, JULY '83                      ;;JRS
          $SWR=165400
          $TN=1
          .SBTTL  OPERATIONAL SWITCH SETTINGS
          ;*
          ;*      SWITCH          USE
          ;*      -----          -----
          ;*      15          HALT ON ERROR
          ;*      14          LOOP ON TEST
          ;*      13          INHIBIT ERROR TYPEOUTS
          ;*      11          INHIBIT ITERATIONS
          ;*      9          LOOP ON ERROR
          ;*      8          LOOP ON TEST IN SWR<7:0>
          .SBTTL  BASIC DEFINITIONS
          ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
          STACK= 1100
          ERROR=  EMT          ;;BASIC DEFINITION OF ERROR CALL
          SCOPE=  IOT          ;;BASIC DEFINITION OF SCOPE CALL
          ;*MISCELLANEOUS DEFINITIONS
          HT= 11          ;;CODE FOR HORIZONTAL TAB
          LF= 12          ;;CODE FOR LINE FEED
          CR= 15          ;;CODE FOR CARRIAGE RETURN
          CRLF= 200        ;;CODE FOR CARRIAGE RETURN-LINE FEED
          PS= 177776      ;;PROCESSOR STATUS WORD
          PSW= PS
          STKLMT= 177774   ;;STACK LIMIT REGISTER
          PIRQ= 177772    ;;PROGRAM INTERRUPT REQUEST REGISTER
          DSWR= 177570    ;;HARDWARE SWITCH REGISTER
          DDISP= 177570   ;;HARDWARE DISPLAY REGISTER
          ;*GENERAL PURPOSE REGISTER DEFINITIONS
          R0= %0          ;;GENERAL REGISTER
          R1= %1          ;;GENERAL REGISTER
          R2= %2          ;;GENERAL REGISTER
          R3= %3          ;;GENERAL REGISTER
          R4= %4          ;;GENERAL REGISTER
          R5= %5          ;;GENERAL REGISTER
          R6= %6          ;;GENERAL REGISTER
          R7= %7          ;;GENERAL REGISTER
          SP= %6          ;;STACK POINTER
000001
160000
415
416
417
418
419
420      165400
421      000001
422
423
001100
104000
000004
000011
000012
000015
000200
177776
177776
177774
177772
177570
177570
000000
000001
000002
000003
000004
000005
000006
000007
000006

```

BASIC DEFINITIONS

```
000007      PC=      #7      ;;PROGRAM COUNTER
; *PRIORITY LEVEL DEFINITIONS
000000      PR0=      0      ;;PRIORITY LEVEL 0
000040      PR1=      40     ;;PRIORITY LEVEL 1
000100      PR2=      100    ;;PRIORITY LEVEL 2
000140      PR3=      140    ;;PRIORITY LEVEL 3
000200      PR4=      200    ;;PRIORITY LEVEL 4
000240      PR5=      240    ;;PRIORITY LEVEL 5
000300      PR6=      300    ;;PRIORITY LEVEL 6
000340      PR7=      340    ;;PRIORITY LEVEL 7
; *"SWITCH REGISTER" SWITCH DEFINITIONS
100000      SW15=     100000
040000      SW14=     40000
020000      SW13=     20000
010000      SW12=     10000
004000      SW11=     4000
002000      SW10=     2000
001000      SW09=     1000
000400      SW08=     400
000200      SW07=     200
000100      SW06=     100
000040      SW05=     40
000020      SW04=     20
000010      SW03=     10
000004      SW02=     4
000002      SW01=     2
000001      SW00=     1
001000      SW9=      SW09
000400      SW8=      SW08
000200      SW7=      SW07
000100      SW6=      SW06
000040      SW5=      SW05
000020      SW4=      SW04
000010      SW3=      SW03
000004      SW2=      SW02
000002      SW1=      SW01
000001      SW0=      SW00
; *DATA BIT DEFINITIONS (BIT00 TO BIT15)
100000      BIT15=    100000
040000      BIT14=    40000
020000      BIT13=    20000
010000      BIT12=    10000
004000      BIT11=    4000
002000      BIT10=    2000
001000      BIT09=    1000
000400      BIT08=    400
000200      BIT07=    200
000100      BIT06=    100
000040      BIT05=    40
000020      BIT04=    20
000010      BIT03=    10
000004      BIT02=    4
000002      BIT01=    2
000001      BIT00=    1
001000      BIT9=     BIT09
000400      BIT8=     BIT08
000200      BIT7=     BIT07
```

BASIC DEFINITIONS

```

000100 BIT6= BIT06
000040 BIT5= BIT05
000020 BIT4= BIT04
000010 BIT3= BIT03
000004 BIT2= BIT02
000002 BIT1= BIT01
000001 BIT0= BIT00
; *BASIC "CPU" TRAP VECTOR ADDRESSES
000004 ERRVEC= 4 ;:TIME OUT AND OTHER ERRORS
000010 RESVEC= 10 ;:RESERVED AND ILLEGAL INSTRUCTIONS
000014 TBITVEC=14 ;:"T" BIT
000014 TRTVEC= 14 ;:TRACE TRAP
000014 BPTVEC= 14 ;:BREAKPOINT TRAP (BPT)
000020 IOTVEC= 20 ;:INPUT/OUTPUT TRAP (IOT) **SCOPE**
000024 PWRVEC= 24 ;:POWER FAIL
000030 EMTVEC= 30 ;:EMULATOR TRAP (EMT) **ERROR**
000034 TRAPVEC=34 ;:"TRAP" TRAP
000060 TKVEC= 60 ;:TTY KEYBOARD VECTOR
000064 TPVEC= 64 ;:TTY PRINTER VECTOR
000240 PIRQVEC=240 ;:PROGRAM INTERRUPT REQUEST VECTOR
424 164160 ABASE= 164160 ;:BASE ADDRESS
425 000001 ADEVM= 1 ;:DEFAULT TO ONE DRV11J
426 100000 RDY= BIT15
427 000400 DIR= BIT8
428 001000 IE= BIT9
;CHIP COMMAND SUMMARY
431 000020 CIRM= 20 ;:CLEAR IRR AND IMR
432 000030 CSIRM= 30 ;:CLEAR SINGLE IRR AND IMR BIT
433
434 000040 CIMR= 40 ;:CLEAR IMR
435 000050 CSIMR= 50 ;:CLEAR SINGLE IMR BIT
436 000060 SIMR= 60 ;:SET ALL IMR BITS
437 000070 SSIMR= 70 ;:SET SINGLE IMR BITS
438
439 000100 CIRR= 100 ;:CLEAR IRR
440 000110 CSIRR= 110 ;:CLEAR SINGLE IRR BITS
441 000120 SIRR= 120 ;:SET ALL IRR BITS
442 000130 SSIRR= 130 ;:SET SINGLE IRR BITS
443
444 000140 CHPISR= 140 ;:CLEAR HIGHEST PRIORITY ISR BIT
445 000160 CISR= 160 ;:CLEAR ISR
446 000170 CSISR= 170 ;:CLEAR SINGLE ISR BIT
447
448 000200 LMD04= 200 ;:LOAD MODE BITS M0-M4
449 000240 LMD57= 240 ;:LOAD MODE BITS M5-M7
450
451 ;CHIP MODE BIT PRESELECTION
452 000240 MISR= 240
453 000244 MIMR= 244
454 000250 MIRR= 250
455 000254 MACR= 254
456
457 ;CHIP WRITE PRESELECTION
458 000300 PACR= 300 ;:PRESELECT AUTO CLEAR REG. FOR WRITING
459 000260 PIMR= 260 ;:PRESELECT IMR REG. FOR WRITING
460 000340 PVMA= 340 ;:PRESELECT VECTOR MEMORY ADDRESS

```

BASIC DEFINITIONS

```

461
462          000000          .SBTTL TRAP CATCHER
                                .=-0
                                ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
                                ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                                ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
                                .=-174
000174 000174 000000          DISPREG: .WORD 0          ;;SOFTWARE DISPLAY REGISTER
000176 000000          SWREG: .WORD 0          ;;SOFTWARE SWITCH REGISTER
000200 000137 001402          .SBTTL STARTING ADDRESS(ES)
463          000100          JMP @#START ;;JUMP TO STARTING ADDRESS OF PROGRAM
464 000100 000104 000200 000002          .=-100
465          000104 000200 000002          .WORD 104,200,2          ;IF 'B EVENT' ON Q BUS IS CONNECTED
                                ;IGNORE IT'S INTERRUPT - JUST DO A RTI

```

ACT11 HOOKS

467

.SBTTL ACT11 HOOKS

HOOKS REQUIRED BY ACT11

000106
000046
013130
000052
000052
000000
000106
001000

\$SVPC=
.=46
\$ENDAD
.=52
.WORD 0
.= \$SVPC
.=1000

;SAVE PC
;;1)SET LOC.46 TO ADDRESS OF \$ENDAD IN .SEOP
;;2)SET LOC.52 TO ZERO
;; RESTORE PC

468
469
470
471
472

;LONGEST TEST TIME
;1ST PASS RUN TIME
;ADDITIONAL RUN TIME

.SBTTL APT PARAMETER BLOCK

SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT

001000
000024
000200
000044
000044
001000
001000

.\$X=. ;;SAVE CURRENT LOCATION
.=24 ;;SET POWER FAIL TO POINT TO START OF PROGRAM
200 ;;FOR APT START UP
.=44 ;;POINT TO APT INDIRECT ADDRESS PNTR.
\$APTHDR ;;POINT TO APT HEADER BLOCK
.=.\$X ;;RESET LOCATION COUNTER

SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC

INTERFACE SPEC.

001000
001000 000000
001002 001170
001004 000006
001006 000024
001010 000024
001012 000031

\$APTHD:
\$HIBTS: .WORD 0 ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
\$MBADR: .WORD \$MAIL ;;ADDRESS OF APT MAILBOX (BITS 0-15)
\$TSTM: .WORD 6. ;;RUN TIM OF LONGEST TEST
\$PASTM: .WORD 20. ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
\$UNITM: .WORD 20. ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
.WORD \$ETEND-\$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)

COMMON TAGS

473

001100 001100
 001100 000000
 001102 000
 001103 000
 001104 000000
 001106 000000
 001110 000000
 001112 000000
 001114 000
 001115 001
 001116 000000
 001120 000000
 001122 000000
 001124 000000
 001126 000000
 001130 000000
 001132 000000
 001134 000
 001135 000
 001136 000000
 001140 177570
 001142 177570
 001144 177560
 001146 177562
 001150 177564
 001152 177566
 001154 000
 001155 002
 001156 012
 001157 000
 001160 000000
 001162 000000
 001164 077
 001165 015
 001166 012

000

.SBTTL COMMON TAGS

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

.-1100
 \$CMTAG: .-1100
 \$TSTNM: .WORD 0
 \$ERFLG: .BYTE 0
 \$ICNT: .WORD 0
 \$LPADR: .WORD 0
 \$LPERR: .WORD 0
 \$ERTTL: .WORD 0
 \$ITEMB: .BYTE 0
 \$ERMAX: .BYTE 1
 \$ERRPC: .WORD 0
 \$GDADR: .WORD 0
 \$BDADR: .WORD 0
 \$GDDAT: .WORD 0
 \$BDDAT: .WORD 0
 \$AUTOB: .BYTE 0
 \$INTAG: .BYTE 0
 \$SWR: .WORD DSWR
 \$DISPLAY: .WORD DDISP
 \$TKS: 177560
 \$TKB: 177562
 \$TPS: 177564
 \$TPB: 177566
 \$NULL: .BYTE 0
 \$FILLS: .BYTE 2
 \$FILLC: .BYTE 12
 \$TPFLG: .BYTE 0
 \$TIMES: 0
 \$ESCAPE: 0
 \$QUES: .ASCII /?/
 \$CRLF: .ASCII <15>
 \$LF: .ASCII <12>

:::START OF COMMON TAGS
 :::CONTAINS THE TEST NUMBER
 :::CONTAINS ERROR FLAG
 :::CONTAINS SUBTEST ITERATION COUNT
 :::CONTAINS SCOPE LOOP ADDRESS
 :::CONTAINS SCOPE RETURN FOR ERRORS
 :::CONTAINS TOTAL ERRORS DETECTED
 :::CONTAINS ITEM CONTROL BYTE
 :::CONTAINS MAX. ERRORS PER TEST
 :::CONTAINS PC OF LAST ERROR INSTRUCTION
 :::CONTAINS ADDRESS OF 'GOOD' DATA
 :::CONTAINS ADDRESS OF 'BAD' DATA
 :::CONTAINS 'GOOD' DATA
 :::CONTAINS 'BAD' DATA
 :::RESERVED--NOT TO BE USED
 :::AUTOMATIC MODE INDICATOR
 :::INTERRUPT MODE INDICATOR
 :::ADDRESS OF SWITCH REGISTER
 :::ADDRESS OF DISPLAY REGISTER
 :::TTY KBD STATUS
 :::TTY KBD BUFFER
 :::TTY PRINTER STATUS REG. ADDRESS
 :::TTY PRINTER BUFFER REG. ADDRESS
 :::CONTAINS NULL CHARACTER FOR FILLS
 :::CONTAINS # OF FILLER CHARACTERS REQUIRED
 :::INSERT FILL CHARS. AFTER A "LINE FEED"
 :::"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
 :::MAX. NUMBER OF ITERATIONS
 :::ESCAPE ON ERROR ADDRESS
 :::QUESTION MARK
 :::CARRIAGE RETURN
 :::LINE FEED

.SBTTL APT MAILBOX-ETABLE

.EVEN
 \$MAIL: .WORD
 \$MSGTY: .WORD AMSTGY
 \$FATAL: .WORD AFATAL
 \$TESTN: .WORD ATESTN
 \$PASS: .WORD APASS
 \$DEVCT: .WORD ADEVCT
 \$UNIT: .WORD AUNIT
 \$MSGAD: .WORD AMSGAD
 \$MSGLG: .WORD AMSLG
 \$ETABLE:
 \$ENV: .BYTE AENV
 \$ENVM: .BYTE AENVM
 \$SWREG: .WORD ASWREG

001170 000000
 001172 000000
 001174 000000
 001176 000000
 001200 000000
 001202 000000
 001204 000000
 001206 000000
 001210 000
 001211 000
 001212 000000

:::APT MAILBOX
 :::MESSAGE TYPE CODE
 :::FATAL ERROR NUMBER
 :::TEST NUMBER
 :::PASS COUNT
 :::DEVICE COUNT
 :::I/O UNIT NUMBER
 :::MESSAGE ADDRESS
 :::MESSAGE LENGTH
 :::APT ENVIRONMENT TABLE
 :::ENVIRONMENT BYTE
 :::ENVIRONMENT MODE BITS
 :::APT SWITCH REGISTER

APT MAILBOX-ETABLE

```

001214 000000      $USWR: .WORD  AUSWR  ;;USER SWITCHES
001216 000000      $CPUOP: .WORD ACPUOP ;;CPU TYPE,OPTIONS
                          :*
                          :*
                          :*
                          :*
                          :*
                          :*
                          :*
                          :*
                          :*
                          :*
001220      000      $MAMS1: .BYTE  AMAMS1 ;;HIGH ADDRESS,M.S. BYTE
001221      000      $MTYP1: .BYTE  AMTYP1 ;;MEM. TYPE,BLK#1
                          :*
                          :*
                          :*
                          :*
                          :*
001222 000000      $MADR1: .WORD  AMADR1 ;;HIGH ADDRESS,BLK#1
                          :*
                          :*
                          :*
001224      000      $MAMS2: .BYTE  AMAMS2 ;;HIGH ADDRESS,M.S. BYTE
001225      000      $MTYP2: .BYTE  AMTYP2 ;;MEM. TYPE,BLK#2
001226 000000      $MADR2: .WORD  AMADR2 ;;MEM.LAST ADDRESS,BLK#2
001230      000      $MAMS3: .BYTE  AMAMS3 ;;HIGH ADDRESS,M.S.BYTE
001231      000      $MTYP3: .BYTE  AMTYP3 ;;MEM. TYPE,BLK#3
001232 000000      $MADR3: .WORD  AMADR3 ;;MEM.LAST ADDRESS,BLK#3
001234      000      $MAMS4: .BYTE  AMAMS4 ;;HIGH ADDRESS,M.S.BYTE
001235      000      $MTYP4: .BYTE  AMTYP4 ;;MEM. TYPE,BLK#4
001236 000000      $MADR4: .WORD  AMADR4 ;;MEM.LAST ADDRESS,BLK#4
001240 000000      $VECT1: .WORD  AVECT1 ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
001242 000000      $VECT2: .WORD  AVECT2 ;;INTERRUPT VECTOR#2BUS PRIORITY#2
001244 164160      $BASE: .WORD  ABASE  ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
001246 000001      $DEVM: .WORD  ADEVM  ;;DEVICE MAP
001250 000000      $CDW1: .WORD  ACDW1  ;;CONTROLLER DESCRIPTION WORD#1
001252
.MEXIT

```

ERROR POINTER TABLE

```

.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

```

```

001252
474
475 001252 017022
476 001254 017154
477 001256 017276
478 001260 000000
479
480
481 001262 017041
482 001264 017154
483 001266 017276
484 001270 000000
485
486
487 001272 017063
488 001274 017154
489 001276 017276
490 001300 000000
491
492
493 001302 017076
494 001304 017154
495 001306 017276
496 001310 000000
497
498
499 001312 017111
500 001314 017154
501 001316 017276
502 001320 000000
503
504
505 001322 017124
506 001324 017154
507 001326 017276
508 001330 000000
509
510
511
512 001332 017137
513 001334 017154
514 001336 017276
515 001340 000000
516
517
518
519

```

```

$ERRTB:
;ERROR 1
EM1      ;REG TIMEOUT ER
DH1      ;ERRPC TSTNUM  BUSADR  EXPCT  RCVD
DT1      ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT(R5)
0

;ERROR 2
EM2      ;REG READ/WRITE ER
DH1      ;ERRPC TSTNUM  BUSADR  EXPCT  RCVD
DT1      ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT(R5)
0

;ERROR 3
EM3      ;IRR REG ER
DH1      ;ERRPC TSTNUM  BUSADR  EXPCT  RCVD
DT1      ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT(R5)
0

;ERROR 4
EM4      ;ACR REG ER
DH1      ;ERRPC TSTNUM  BUSADR  EXPCT  RCVD
DT1      ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT(R5)
0

;ERROR 5
EM5      ;IMR REG ERROR
DH1      ;ERRPC TSTNUM  BUSADR  EXPCT  RCVD
DT1      ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT(R5)
0

;ERROR 6
EM6      ;ISR REG ERROR
DH1      ;ERRPC TSTNUM  BUSADR  EXPCT  RCVD
DT1      ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT(R5)
0

;ERROR 7
EM7      ;CHIP STAT ER
DH1      ;ERRPC TSTNUM  BUSADR  EXPCT  RCVD
DT1      ;$ERRPC TSTNUM  $BDADR  $GDDAT  $BDDAT(R5)
0

```

; BUS REGISTER ADDRESS POINTERS

ERROR POINTER TABLE

520
 521 001342 164160
 522 001344 164162
 523 001346 164164
 524 001350 164166
 525 001352 164170
 526 001354 164172
 527 001356 164174
 528 001360 164176
 529
 530
 531
 532
 533 001362 000000
 534 001364 000001
 535 001366 000000
 536 001370 000000
 537 001372 000000
 538 001374 000000
 539 001376 000000
 540 001400 000000

DRCSA: ABASE
 DRDBA: ABASE+2
 DRCSB: ABASE+4
 DRDBB: ABASE+6
 DRCSC: ABASE+10
 DRDBC: ABASE+12
 DRCSB: ABASE+14
 DRDBD: ABASE+16

;COMMON PROGRAM LOCATION(S)

TSTNUM: 0 ;CONTAINS TEST NUMBER ON ERROR
 DMAP: 1
 INTFLG: .WORD 0
 XXDP: .WORD 0
 IMRLOC: .WORD 0
 ISRLOC: .WORD 0
 IRRLOC: .WORD 0
 ACRLOC: .WORD 0

T1 TEST THAT ALL REGISTERS ARE ADDRESSABLE

```

613 002204 000771          BR      1$          ;TRY NEXT
614 002206 022626          2$:    CMP      (SP)+,(SP)+ ;FIX STACK SINCE NO RTI
615 002210 104001          ERROR+ 1          ;BUS ADRS INDICATED DID NOT RESPOND
616 002212 012737 000006 000004 3$:    MOV      @ERRVEC+2,@ERRVEC ;RESTORE LOC 4
617
618

```

```

;*****
;*TEST 2      TEST CSRA W/R DIR BIT,INT. CHIP RESET STATUS
;*****

```

```

002220 000004
619 002222 013700 001342    TST2:  SCOPE
620 002226 013701 001352    MOV      DRCSA,R0      ;GET CSR ADDRESS
621 002232 005010          MOV      DRCSC,R1     ;STORE CSRC ADDRESS
622 002234 005011          CLR      (R0)         ;INIT CSRA
623 002236 010037 001122    CLR      (R1)         ;INIT CSRC
624          MOV      R0,$BDADR ;STORE CSR ADDRESS
625 002242 012737 100700 001124  MOV      @RDY!DIR!BIT7!BIT6,$GDDAT ;SET UP EXPECTED DATA
626 002250 112760 000001 000001  MOVVB   @BIT0,1(R0)   ;SET DIRECTION BIT
627 002256 011015          MOV      (R0),(R5)   ;GET CSR DATA
628 002260 042715 000007          BIC      @7,(R5)     ;CLEAR UNDEFINED BITS
629 002264 023715 001124          CMP      $GDDAT,(R5)
630 002270 001401          BEQ      100$
631 002272 104002          ERROR+ 2          ;CSRA ERROR
632 002274 010137 001122 100$:  MOV      R1,$BDADR   ;STORE CSRC ADDRESS
633 002300 012737 000200 001124  MOV      @BIT7,$GDDAT ;STORE EXPECTED
634 002306 011115          MOV      (R1),(R5)   ;STORE CSRC
635 002310 042715 000007          BIC      @7,(R5)     ;CLEAR UNDEFINED BITS
636 002314 023715 001124          CMP      $GDDAT,(R5)
637 002320 001401          BEQ      1$
638 002322 104002          ERROR+ 2          ;CSRC ERROR
639 002324 010037 001122 1$:    MOV      R0,$BDADR   ;CSRA ADDRESS
640 002330 012737 100300 001124  MOV      @RDY!BIT7!BIT6,$GDDAT
641 002336 105060 000001          CLR      1(R0)       ;CLEAR CSRA DIR BIT
642 002342 011015          MOV      (R0),(R5)   ;READ CSR
643 002344 042715 000007          BIC      @7,(R5)     ;CLEAR UNDEFINED BITS
644 002350 023715 001124          CMP      $GDDAT,(R5)
645 002354 001401          BEQ      2$          ;;BR IF EQUAL
646 002356 104002          ERROR+ 2
647 002360 012737 000300 001124 2$:    MOV      @BIT7!BIT6,$GDDAT
648 002366 112761 000001 000001  MOVVB   @BIT0,1(R1)   ;CSRC TO OUTPUT MODE
649 002374 011015          MOV      (R0),(R5)   ;READ CSRA
650 002376 042715 000007          BIC      @7,(R5)     ;CLEAR UNDEFINED BITS
651 002402 023715 001124          CMP      $GDDAT,(R5) ;CHECK IF RDY BIT CLEARED
652 002406 001401          BEQ      TST3        ;;BR IF EQUAL
653 002410 104002          ERROR+ 2          ;CSRA REG ERROR
654
655

```

```

;*****
;*TEST 3      TEST CSRA INT. ENABLE BIT
;*****

```

```

002412 000004
656 002414 004737 013164    TST3:  SCOPE
657 002420 013700 001342    JSR     PC,CLRCR      ;CLEAR CSR REGISTER
658 002424 013701 001352    MOV      DRCSA,R0     ;GET CSRA ADDRESS
659 002430 112761 000001 000001  MOV      DRCSC,R1     ;GET CSRC ADDRESS
660 002436 010037 001122    MOVVB   @BIT0,1(R1)   ;SET DIRECTION BIT CSRC
661 002442 012737 001300 001124  MOV      R0,$BDADR   ;STORE CSRA ADDRESS
662 002450 012710 001000    MOV      @BIT9!BIT7!BIT6,$GDDAT
663 002454 011015          MOV      @IE,(R0)    ;SET INTERRUPT ENABLE BIT
          MOV      (R0),(R5)

```

T3 TEST CSRA INT. ENABLE BIT

```

664 002456 042715 000007      BIC    #7,(R5)      ;CLEAR UNDEFINED BITS
665 002462 023715 001124      CMP    $GDDAT,(R5)
666 002466 001401              BEQ    1$
667 002470 104002              ERROR+ 2           ;INT. ENABLE ERROR,CSRA
668 002472 012737 000300 001124 1$:  MOV    #BIT7!BIT6,$GDDAT
669 002500 105060 000001              CLRB  1(R0)       ;CLEAR I/E BIT
670 002504 011015              MOV    (R0),(R5)
671 002506 042715 000007      BIC    #7,(R5)      ;CLEAR UNDEFINED BITS
672 002512 023715 001124      CMP    $GDDAT,(R5)
673 002516 001401              BEQ    TST4        ;;BR IF EQUAL
674 002520 104002              ERROR+ 2           ;CSRA ERROR
675
676

```

```

;*****
;*TEST 4      TEST CSRA I/E,DIR BIT
;*****

```

```

TST4:  SCOPE
677 002522 000004              JSR    PC,CLRCR    ;CLEAR CSR REGISTERS
678 002524 004737 013164      MOV    DRCSA,R0    ;GET CSRA ADDRESS
679 002530 013700 001342      MOV    R0,$BDADR   ;SAVE CSRA ADDRESS
680 002534 010037 001122      MOV    #101700,$GDDAT ;EXPECTED I/E,DIR
681 002540 012737 101700 001124  MOV    #BIT1!BIT0,1(R0)
682 002546 112760 000003 000001      MOVB   ;SET I/E AND DIR BIT
683 002554 011015              MOV    (R0),(R5)
684 002556 042715 000007      BIC    #7,(R5)      ;CLEAR UNDEFINED BITS
685 002562 023715 001124      CMP    $GDDAT,(R5)
686 002566 001401              BEQ    1$
687 002570 104002              ERROR+ 2           ;CSRA ERROR
688 002572 012737 100300 001124 1$:  MOV    #RDY!BIT7!BIT6,$GDDAT
689 002600 005010              CLR    (R0)
690 002602 011015              MOV    (R0),(R5)
691 002604 042715 000007      BIC    #7,(R5)      ;CLEAR UNDEFINED BITS
692 002610 023715 001124      CMP    $GDDAT,(R5)
693 002614 001401              BEQ    TST5        ;;BR IF EQUAL
694 002616 104002              ERROR+ 2           ;CSRA ERROR
695
696
697

```

```

;*****
;*TEST 5      TEST CSRB W/R DIR BIT,INT CHIP RESET STATUS
;*****

```

```

TST5:  SCOPE
698 002620 000004              JSR    PC,CLRCR    ;CLEAR CSR REGISTERS
699 002622 004737 013164      JSR    DRCSB,R0    ;GET CSR ADDRESS
700 002626 013700 001346      MOV    DRCSB,R0    ;GET CSR ADDRESS
701 002632 013701 001356      MOV    DRCSB,R1    ;GET CSR ADDRESS
702 002636 010037 001122      MOV    R0,$BDADR   ;STORE CSR ADDRESS
703 002642 012737 100400 001124  MOV    #RDY!DIR,$GDDAT ;SET UP EXPECTED DATA
704 002650 112760 000001 000001      MOVB   ;SET DIRECTION BIT
705 002656 011015              MOV    (R0),(R5)   ;GET CSRB DATA
706 002660 023715 001124      CMP    $GDDAT,(R5)
707 002664 001401              BEQ    100$
708 002666 104002              ERROR+ 2           ;CSRB ERROR
709 002670 010137 001122 100$:  MOV    R1,$BDADR   ;STORE CSRD ADDRESS
710 002674 005037 001124      CLR    $GDDAT      ;STORE EXPECTED
711 002700 011115              MOV    (R1),(R5)   ;READ CSRD
712 002702 023715 001124      CMP    $GDDAT,(R5)
713 002706 001401              BEQ    1$
714 002710 104002              ERROR+ 2           ;CSRD ERROR
714 002712 010037 001122 1$:  MOV    R0,$BDADR   ;STORE CSRB ADDRESS

```

T5 TEST CSRB W/R DIR BIT,INT CHIP RESET STATUS

```

715 002716 012737 100000 001124      MOV    #RDY,$GDDAT      ;STORE EXPECTED
716 002724 105060 000001              CLR    1(R0)            ;CLEAR CSR DIR BIT
717 002730 011015              MOV    (R0),(R5)        ;READ CSR
718 002732 023715 001124      CMP    $GDDAT,(R5)
719 002736 001401              BEQ    2$               ;;BR IF EQUAL
720 002740 104002              ERROR+ 2
721 002742 005037 001124      CLR    $GDDAT          ;STORE EXPECTED
722 002746 112761 000001 000001 2$:  MOV    #BIT0,1(R1)     ;CSR TO OUTPUT MODE
723 002754 011015              MOV    (R0),(R5)        ;READ CSRB
724 002756 023715 001124      CMP    $GDDAT,(R5)     ;RDY BIT CLEARED
725 002762 001401              BEQ    TST6            ;;BR IF EQUAL
726 002764 104002              ERROR+ 2               ;CSRB REG ERROR
727
728

```

;*TEST 6 TEST CSRC W/R DIR BIT,INT CHIP RESET STATUS

```

002766 00C004
729 002770 004737 013164      TST6: SCOPE
730 002774 013700 001352      JSR    PC,CLRCSR       ;CLEAR CSR REGISTERS
731 003000 013701 001342      MOV    DRCSA,R1        ;GET CSRA ADDRESS
732 003004 010037 001122      MOV    RO,$BDADR       ;STORE CSR ADDRESS
733 003010 012737 100600 001124  MOV    #RDY!DIR!BIT7,$GDDAT
734 003016 112760 000001 000001  MOV    #BIT0,1(R0)     ;SET DIRECTION BIT
735 003024 011015              MOV    (R0),(R5)        ;GET CSR DATA
736 003026 042715 000007              BIC    #7,(R5)         ;CLEAR UNDEFINED BITS
737 003032 023715 001124      CMP    $GDDAT,(R5)
738 003036 001401              BEQ    100$            ;CHECK CSRA
739 003040 104002              ERROR+ 2              ;CSRA ERROR
740 003042 010137 001122 100$:  MOV    R1,$BDADR       ;STORE CSR ADDRESS
741 003046 012737 000300 001124  MOV    #BIT7!BIT6,$GDDAT
742 003054 011115              MOV    (R1),(R5)        ;READ CSRA
743 003056 042715 000007              BIC    #7,(R5)         ;CLEAR UNDEFINED BITS
744 003062 023715 001124      CMP    $GDDAT,(R5)
745 003066 001401              BEQ    1$             ;CHECK CSRA
746 003070 104002              ERROR+ 2              ;CSRA ERROR
747 003072 010037 001122 1$:  MOV    RO,$BDADR       ;STORE CSR ADDRESS
748 003076 012737 100200 001124  MOV    #RDY!BIT7,$GDDAT ;STORE EXPECTED DATA
749 003104 105060 000001              CLR    1(R0)            ;CLEAR CSR DIR BIT
750 003110 011015              MOV    (R0),(R5)        ;READ CSR
751 003112 042715 000007              BIC    #7,(R5)         ;CLEAR UNDEFINED BITS
752 003116 023715 001124      CMP    $GDDAT,(R5)
753 003122 001401              BEQ    2$             ;;BR IF EQUAL
754 003124 104002              ERROR+ 2
755 003126 012737 000200 001124 2$:  MOV    #BIT7,$GDDAT    ;EXPECTED DATA
756 003134 112761 000001 000001  MOV    #BIT0,1(R1)     ;CSRA TO OUTPUT MODE
757 003142 011015              MOV    (R0),(R5)        ;READ CSRC
758 003144 042715 000007              BIC    #7,(R5)         ;CLEAR UNDEFINED BITS
759 003150 023715 001124      CMP    $GDDAT,(R5)     ;RDY BIT CLEARED
760 003154 001401              BEQ    TST7            ;;BR IF EQUAL
761 003156 104002              ERROR+ 2               ;CSRC REG ERROR
762
763

```

;*TEST 7 TEST CSRD W/R DIR BIT,INT CHIP RESET STATUS

```

003160 000004
764 003162 004737 013164      TST7: SCOPE
765 003166 013700 001356      JSR    PC,CLRCSR       ;CLEAR CSR REGISTERS
                          MOV    DRCSA,R1        ;GET CSR ADDRESS

```


T7 TEST CSRD W/R DIR BIT,INT CHIP RESET STATUS

```

766 003172 013701 001346      MOV      DRCSB,R1      ;CSRB ADDRESS
767 003176 010037 001122      MOV      R0,$BDADR    ;STORE CSRB ADDRESS
768 003202 012737 100400 001124  MOV      #RDY!DIR,$GDDAT ;SET UP EXPECTED DATA
769 003210 112760 000001 000001  MOVVB   #BIT0,1(R0)   ;SET DIRECTION BIT
770 003216 011015                MOV      (R0),(R5)    ;GET CSRB DATA
771 003220 023715 001124      CMP      $GDDAT,(R5)
772 003224 001401                BEQ      100$
773 003226 104002                ERROR+  2
774 003230 010137 001122      100$:  MOV      R1,$BDADR    ;STORE CSRB ADDRESS
775 003234 005037 001124      CLR      $GDDAT      ;STORE EXPECTED
776 003240 011115                MOV      (R1),(R5)
777 003242 023715 001124      CMP      $GDDAT,(R5)
778 003246 001401                BEQ      1$
779 003250 104002                ERROR+  2
780 003252 010037 001122      1$:   MOV      R0,$BDADR    ;CSRB ERROR
781 003256 012737 100000 001124  MOV      #RDY,$GDDAT ;STORE CSRD ADDRESS
782 003264 105060 000001                CLRB    1(R0)        ;STORE EXPECTED
783 003270 011015                MOV      (R0),(R5)   ;CLEAR CSRB DIR BIT
784 003272 023715 001124      CMP      $GDDAT,(R5) ;READ CSRB
785 003276 001401                BEQ      2$          ;:BR IF EQUAL
786 003300 104002                ERROR+  2
787 003302 005037 001124      2$:   CLR      $GDDAT      ;EXPECTED
788 003306 112761 000001 000001  MOVVB   #BIT0,1(R1)   ;CSRD TO OUTPUT MODE
789 003314 011015                MOV      (R0),(R5)   ;READ CSRB
790 003316 023715 001124      CMP      $GDDAT,(R5) ;RDY BIT CLEARED
791 003322 001401                BEQ      TST10       ;:BR IF EQUAL
792 003324 104002                ERROR+  2            ;CSRD REG ERROR
793
794

```

```

;*****
;*TEST 10      TEST DBRA W/R IN OUTPUT MODE
;*****

```

```

TST10:  SCOPE
795 003330 004737 013164      JSR      PC,CLRCSR    ;CLEAR ALL CSRS
796 003334 012703 013272      MOV      #BEGPAT,R3  ;GET DATA PATTERN TABLE
797 003340 012737 003370 001110  MOV      #1,$LPERR   ;SET UP SCOPE ADDRESS
798 003346 013700 001342      MOV      DRCSA,R0    ;GET CSRA
799 003352 013701 001344      MOV      DRDBA,R1    ;GET DBRA ADDRESS
800 003356 010137 001122      MOV      R1,$BDADR   ;STORE DBRA ADDRESS
801 003362 112760 000001 000001  MOVVB   #BIT0,1(R0)   ;SET CSRA IN OUTPUT MODE
802 003370 011337 001124      1$:   MOV      (R3),$GDDAT ;SAVE EXPECTED DATA
803 003374 005011                CLR      (R1)        ;CLEAR DBRA
804 003376 051311                BIS     (R3),(R1)    ;WRITE INTO DBRA
805 003400 011115                MOV     (R1),(R5)    ;READ DBRA
806 003402 023715 001124      CMP     $GDDAT,(R5)  ;CHECK W/R DBRA
807 003406 001401                BEQ     2$          ;NEXT PAT. IF EQUAL
808 003410 104002                ERROR+  2            ;DBRA W/R ERROR
809 003412 005723      2$:   TST     (R3)+       ;INC FOR NEXT PATTERN
810 003414 020327 013572      CMP     R3,#ENDDAT   ;CHECK FOR END
811 003420 001363                BNE    1$          ;DO NEXT PATTERN
812
813

```

```

;*****
;*TEST 11      TEST DBRB W/R IN OUTPUT MODE
;*****

```

```

TST11:  SCOPE
814 003424 004737 013164      JSR      PC,CLRCSR    ;CLEAR ALL CSRS
815 003430 012703 013272      MOV      #BEGPAT,R3  ;GET DATA PATTERN TABLE
816 003434 012737 003464 001110  MOV      #1,$LPERR   ;SET UP SCOPE ADDRESS

```

T11 TEST DBRB W/R IN OUTPUT MODE

```

817 003442 013700 001346      MOV      DRCSB,R0      ;GET CSRB
818 003446 013701 001350      MOV      DRDBB,R1     ;GET DBRB ADDRESS
819 003452 010137 001122      MOV      R1,$BDADR    ;STORE DBRB ADDRESS
820 003456 112760 000001 000001  MOVB     #BIT0,1(R0)   ;SET CSRB IN OUTPUT MODE
821 003464 011337 001124      MOV      (R3),$GDDAT  ;SAVE EXPECTED DATA
822 003470 005011              CLR      (R1)         ;CLEAR DBRB
823 003472 051311              BIS      (R3),(R1)    ;WRITE INTO DBRB
824 003474 011115              MOV      (R1),(R5)    ;READ DBRB
825 003476 023715 001124      CMP      $GDDAT,(R5)  ;CHECK W/R DBRB
826 003502 001401              BEQ      2$           ;NEXT PAT. IF EQUAL
827 003504 104002              ERROR+   2           ;DBRB W/R ERROR
828 003506 005723              TST      (R3)+        ;INC FOR NEXT PATTERN
829 003510 020327 013572      CMP      R3,#ENDDAT  ;CHECK FOR END
830 003514 001363              BNE      1$           ;DO NEXT PATTERN

```

831
832
833

```

;*****
;*TEST 12      TEST DBRC W/R IN OUTPUT MODE
;*****

```

```

003516 000004
834 003520 004737 013164      JSR      PC,CLRCSR    ;CLEAR ALL CSRS
835 003524 012703 013272      MOV      #BEGPAT,R3  ;GET DATA PATTERN TABLE
836 003530 012737 003560 001110  MOV      #1$,$LPERR   ;SET UP SCOPE ADDRESS
837 003536 013700 001352      MOV      DRCSB,R0    ;GET CSRB
838 003542 013701 001354      MOV      DRDBC,R1    ;GET DBRC ADDRESS
839 003546 010137 001122      MOV      R1,$BDADR  ;STORE DBRC ADDRESS
840 003552 112760 000001 000001  MOVB     #BIT0,1(R0)   ;SET CSRB IN OUTPUT MODE
841 003560 011337 001124      MOV      (R3),$GDDAT ;SAVE EXPECTED DATA
842 003564 005011              CLR      (R1)         ;CLEAR DBRC
843 003566 051311              BIS      (R3),(R1)    ;WRITE INTO DBRC
844 003570 011115              MOV      (R1),(R5)    ;READ DBRC
845 003572 023715 001124      CMP      $GDDAT,(R5)  ;CHECK W/R DBRC
846 003576 001401              BEQ      2$           ;NEXT PAT. IF EQUAL
847 003600 104002              ERROR+   2           ;DBRC W/R ERROR
848 003602 005723              TST      (R3)+        ;INC FOR NEXT PATTERN
849 003604 020327 013572      CMP      R3,#ENDDAT  ;CHECK FOR END
850 003610 001363              BNE      1$           ;DO NEXT PATTERN

```

851
852
853

```

;*****
;*TEST 13      TEST DBRD W/R IN OUTPUT MODE
;*****

```

```

003612 000004
854 003614 004737 013164      JSR      PC,CLRCSR    ;CLEAR ALL CSRS
855 003620 012703 013272      MOV      #BEGPAT,R3  ;GET DATA PATTERN TABLE
856 003624 012737 003654 001110  MOV      #1$,$LPERR   ;SET UP SCOPE ADDRESS
857 003632 013700 001356      MOV      DRCSB,R0    ;GET CSRB
858 003636 013701 001360      MOV      DRDBD,R1    ;GET DBRD ADDRESS
859 003642 010137 001122      MOV      R1,$BDADR  ;STORE DBRD ADDRESS
860 003646 112760 000001 000001  MOVB     #BIT0,1(R0)   ;SET CSRB IN OUTPUT MODE
861 003654 011337 001124      MOV      (R3),$GDDAT ;SAVE EXPECTED DATA
862 003660 005011              CLR      (R1)         ;CLEAR DBRD
863 003662 051311              BIS      (R3),(R1)    ;WRITE INTO DBRD
864 003664 011115              MOV      (R1),(R5)    ;READ DBRD
865 003666 023715 001124      CMP      $GDDAT,(R5)  ;CHECK W/R DBRD
866 003672 001401              BEQ      2$           ;NEXT PAT. IF EQUAL
867 003674 104002              ERROR+   2           ;DBRD W/R ERROR

```

N2

T13 TEST DBRD W/R IN OUTPUT MODE

868 003676 005723
869 003700 020327 013572
870 003704 001363

2\$:

TST (R3)+
CMP R3,#ENDDAT
BNE 1\$

:INC FOR NEXT PATTERN
:CHECK FOR END
:DO NEXT PATTERN

T13 TEST DBRD W/R IN OUTPUT MODE

872
873
874

: *TEST 14 TEST CSR UNIQUENESS, CSRS (A-B), (C-D)
: *****

003706 000004
875 003710 004737 013164
876
877
878
879
880 003714 112760 000003 000001
881 003722 112763 000001 000001
882 003730 010037 001122
883 003734 012737 101700 001124
884 003742 011015
885 003744 042715 000007
886 003750 023715 001124
887 003754 001401
888 003756 104002
889 003760 010137 001122
890 003764 005037 001124
891 003770 011115
892 003772 023715 001124
893 003776 001401
894 004000 104002
895 004002 010337 001122
896 004006 012737 100400 001124
897 004014 011315
898 004016 023715 001124
899 004022 001401
900 004024 104002
901 004026 010237 001122
902 004032 012737 000200 001124
903 004040 011215
904 004042 042715 000007
905 004046 023715 001124
906 004052 001401
907 004054 104002
908
909

TST14: SCOPE
JSR PC, CLRCSR ;CLEAR ALL CSRS
;CSRA = R0
;CSRB = R1
;CSRC = R2
;CSRD = R3
;CSRA OUTPUT MODE, I/E
;CSRD OUTPUT MODE
MOVB #3, 1(R0)
MOVB #BIT0, 1(R3)
MOV R0, %BDADR
MOV #101700, %GDDAT ;EXPECTED DATA
MOV (R0), (R5)
BIC #7, (R5) ;CLEAR UNDEFINED BITS
CMP %GDDAT, (R5)
BEQ 1\$
ERROR* 2 ;CSRA ERROR
1\$: MOV R1, %BDADR ;CHECK CSRB
CLR %GDDAT
MOV (R1), (R5)
CMP %GDDAT, (R5)
BEQ 2\$
ERROR* 2 ;CSRB ERROR
2\$: MOV R3, %BDADR
MOV #100400, %GDDAT
MOV (R3), (R5)
CMP %GDDAT, (R5)
BEQ 3\$
ERROR* 2 ;CSRD ERROR
3\$: MOV R2, %BDADR ;CHECK CSRC
MOV #BIT7, %GDDAT ;EXPECTED
MOV (R2), (R5)
BIC #7, (R5)
CMP %GDDAT, (R5)
BEQ TST15 ;;BR IF EQUAL
ERROR* 2 ;CSRC ERROR

: *TEST 15 TEST CSR UNIQUENESS, CSRS (A-D), (C-B)
: *****

004056 000004
910 004060 004737 013164
911
912
913
914
915 004064 112760 000003 000001
916 004072 112761 000001 000001
917 004100 010037 001122
918 004104 012737 101700 001124
919 004112 011015
920 004114 042715 000007
921 004120 023715 001124
922 004124 001400

TST15: SCOPE
JSR PC, CLRCSR ;CLR ALL CSRS
;CSRA = R0
;CSRB = R1
;CSRC = R2
;CSRD = R3
;CSRA OUTPUT, I/E
;CSRB OUTPUT MODE
;CSRA ADDRESS
MOVB #3, 1(R0)
MOVB #BIT0, 1(R1)
MOV R0, %BDADR
MOV #101700, %GDDAT
MOV (R0), (R5)
BIC #7, (R5) ;CLEAR UNDEFINED BITS
CMP %GDDAT, (R5)
BEQ 1\$

T15 TEST CSR UNIQUENESS,CSRS (A-D),(C B)

```

923 004126 010137 001122 1$: MOV R1,$BDADR ;CHECK CSRB
924 004132 012737 100400 001124 MOV #100400,$GDDAT
925 004140 011115 MOV (R1),(R5)
926 004142 023715 001124 CMP $GDDAT,(R5)
927 004146 001401 BEQ 2$
928 004150 104002 ERROR. 2 ;CSRB ERROR
929 004152 010237 001122 2$: MOV R2,$BDADR ;CHECK CSRC
930 004156 012737 000200 001124 MOV #BIT7,$GDDAT
931 004164 011215 MOV (R2),(R5)
932 004166 042715 000007 BIC #7,(R5)
933 004172 023715 001124 CMP $GDDAT,(R5)
934 004176 001401 BEQ 3$
935 004200 104002 ERROR. 2 ;CSRC ERROR
936 004202 010337 001122 3$: MOV R3,$BDADR
937 004206 005037 001124 CLR $GDDAT
938 004212 011315 MOV (R3),(R5)
939 004214 023715 001124 CMP $GDDAT,(R5)
940 004220 001401 BEQ TST16 ;:BR IF EQUAL
941 004222 104002 ERROR. 2 ;CSRD ERROR
942
943
944

```

```

;:*****
;:TEST 16 TEST PORT A TO PORT C INTERACTION
;:*****

```

```

004224 000004
945 004226 004737 013164 TST16: SCOPE
946 004232 012703 013272 JSR PC,CLRCSR ;CLEAR ALL CSRS
947 004236 012737 004266 001110 MOV #BEGPAT,R3 ;GET DATA PATTERN TABLE
948 004244 013700 001342 MOV #1,$LPERR ;SET UP SCOPE ADDRESS
949 004250 013701 001352 MOV DRCSA,R0 ;GET PORT A, CSRA ADDRESS
950 004254 013702 001346 MOV DRCSB,R1 ;GET PORT C, CSRC ADDRESS
951 004260 112762 000001 000001 MOV DRCSB,R2 ;CSRB ADDRESS
952 004266 010037 001122 1$: MOVB #BIT0,1(R2) ;CSRB IN OUTPUT MODE
953 004272 005062 000002 MOV R0,$BDADR ;STORE CSRA ADDRESS
954 004276 105061 000001 CLR 2(R2) ;CLEAR DBRB
955 004302 112760 000001 000001 CLR 1(R1) ;PORT C,CSRC,INPUT MODE
956 004310 011360 000002 MOVB #BIT0,1(R0) ;SET CSRA IN OUTPUT MODE
957 004314 016104 000002 MOV (R3),2(R0) ;WRITE INTO DBRA
958 004320 012737 100700 001124 MOV 2(R1),R4 ;READ DBRC
959 ;RDY!DIR!BIT7!BIT6,$GDDAT
960 004326 011015 MOV (R0),(R5) ;CSRA DIR SHOULD STAY SET
961 004330 042715 000007 BIC #7,(R5) ;READ CSRA
962 004334 023715 001124 CMP $GDDAT,(R5) ;CLEAR UNDEFINED BITS
963 004340 001401 BEQ 100$
964 004342 104002 ERROR. 2 ;CSRA ERROR
965 004344 012737 000200 001124 100$: MOV #BIT7,$GDDAT
966 004352 010137 001122 MOV R1,$BDADR ;CSRC ADDRESS
967 004356 011115 MOV (R1),(R5) ;READ CSRC
968 004360 042715 000007 BIC #7,(R5) ;CLEAR UNDEFINED BITS
969 004364 023715 001124 CMP $GDDAT,(R5)
970 004370 001401 BEQ 101$
971 004372 104002 ERROR. 2 ;CSRC ERROR
972 004374 013737 001354 001122 101$: MOV DRDBC,$BDADR ;STORE DBRC ADDRESS
973 004402 011337 001124 MOV (R3),$GDDAT ;SAVE EXPECTED
974 004406 010415 MOV R4,(R5) ;DBRC CONTENTS
975 004410 023715 001124 CMP $GDDAT,(R5) ;CHECK PORT C,DBRC
976 004414 001401 BEQ 2$ ;BEQ TO NEXT SUBTEST

```

T16 TEST PORT A TO PORT C INTERACTION

```

977 004416 104002          ERROR* 2          ;DBRC REG ERROR
978 004420 005137 001124 2$:  COM      $GDDAT      ;SET UP TO WRITE COM DATA FROM
979                                     ;PORT C TO PORT A
980 004424 013737 001344 001122  MOV     DRDBA,$BDADR  ;STORE DBRA ADDRESS
981 004432 013761 001124 000002 3$:  MOV     $GDDAT,2(R1)  ;WRITE PORT C, INPUT MODE
982 004440 105060 000001          CLR     1(R0)        ;MAKE PORT A INPUT MODE
983 004444 112761 000001 000001  MOV     @BIT0,1(R1)  ;MAKE PORT C OUTPUT MODE
984 004452 016015 000002          MOV     2(R0),(R5)   ;READ PORT A, DBRA
985 004456 023715 001124          CMP     $GDDAT,(R5)  ;PORT A =PORT C?
986 004462 001401          BEQ     4$          ;CHECK DBRB FOR NO DATA CHANGE
987 004464 104002          ERROR* 2          ;PORT A, DBRA REG ERROR
988 004466 013737 001350 001122 4$:  MOV     DRDBB,$BDADR  ;STORE DBRB ADDRESS
989 004474 005037 001124          CLR     $GDDAT      ;CONTENTS SHOULD STAY ZERO
990 004500 016215 000002          MOV     2(R2),(R5)  ;READ DBRB FOR CLEAR
991 004504 023715 001124          CMP     $GDDAT,(R5)  ;CHECK FOR DBRB CLEAR
992 004510 001401          BEQ     5$          ;YES, CONTINUE
993 004512 104002          ERROR* 2          ;DBRB INTERACTION ERROR
994 004514 005723          TST     (R3)+        ;INC FOR NEXT PATTERN
995 004516 020327 013572 5$:  CMP     R3,@ENDDAT  ;CHECK FOR END
996 004522 001261          BNE     1$          ;DO NEXT PATTERN
997
998
999

```

```

;*****
;*TEST 17 TEST PORT B TO PORT D INTERACTION
;*****

```

```

004524 000004
1000 004526 004737 013164          TST17: SCOPE
1001 004532 012703 013272          JSR     PC,CLRCR     ;CLEAR ALL CSRS
1002 004536 012737 004566 001110  MOV     @BEGPAT,R3   ;GET DATA PATTERN TABLE
1003 004544 013700 001346          MOV     @1$, $LPERR  ;SET UP SCOPE ADDRESS
1004 004550 013701 001356          MOV     DRCSB,R0     ;GET PORT B, CSRB ADDRESS
1005 004554 013702 001342          MOV     DRCSA,R2     ;GET PORT D, CSRD ADDRESS
1006 004560 112762 000001 000001  MOV     @BIT0,1(R2)  ;STORE CSRA ADDRESS
1007 004566 010037 001122 1$:  MOV     R0,$BDADR    ;CSRA IN OUTPUT MODE
1008 004572 005062 000002          CLR     2(R2)        ;STORE CSRB ADDRESS
1009 004576 105061 000001          CLR     1(R1)        ;CLEAR DBRA
1010 004602 112760 000001 000001  MOV     @BIT0,1(R0)  ;PORT D, CSRD, INPUT MODE
1011 004610 011360 000002          MOV     (R3),2(R0)  ;SET CSRB IN OUTPUT MODE
1012 004614 016104 000002          MOV     2(R1),R4     ;WRITE INTO DBRB
1013 004620 012737 100400 001124  MOV     @RDY!DIR,$GDDAT ;DBRD CONTENTS
1014 004626 011015          MOV     (R0),(R5)   ;SAVE EXPECTED
1015 004630 023715 001124          CMP     $GDDAT,(R5) ;READ CSRB
1016 004634 001401          BEQ     100$        ;DBRD CONTENTS
1017 004636 104002          ERROR* 2          ;CSRB ERROR
1018 004640 005037 001124 100$: CLR     $GDDAT      ;SAVE EXPECTED
1019 004644 010137 001122          MOV     R1,$BDADR   ;SAVE CSRD ADDRESS
1020 004650 011115          MOV     (R1),(R5)
1021 004652 023715 001124          CMP     $GDDAT,(R5)
1022 004656 001401          BEQ     101$        ;DBRD CONTENTS
1023 004660 104002          ERROR* 2          ;CSRD ERROR
1024 004662 013737 001360 001122 101$: MOV     DRDBD,$BDADR  ;SAVE DBRD ADDRESS
1025 004670 011337 001124          MOV     (R3),$GDDAT ;SAVE EXPECTED
1026 004674 010415          MOV     R4,(R5)     ;DBRD CONTENTS
1027 004676 023715 001124          CMP     $GDDAT,(R5) ;CHECK PORT D, DBRD
1028 004702 001401          BEQ     2$          ;BEQ TO NEXT SUBTEST
1029 004704 104002          ERROR* 2          ;DBRD REG ERROR
1030 004706 005137 001124 2$:  COM      $GDDAT      ;SET UP TO WRITE COM DATA FROM

```

T17 TEST PORT B TO PORT D INTERACTION

```

1031
1032 004712 013737 001350 001122      MOV      DRDBB,$BDADR      ;PORT D TO PORT B
1033 004720 013761 001124 000002 3$:  MOV      $GDDAT,2(R1)    ;STORE DBRB ADDRESS
1034 004726 105060 000001      CLR      1(R0)           ;WRITE PORT D,INPUT MODE
1035 004732 112761 000001 000001  MOV      @BIT0,1(R1)     ;MAKE PORT B INPUT MODE
1036 004740 016015 000002      MOV      2(R0),(R5)      ;MAKE PORT D OUTPUT MODE
1037 004744 023715 001124      CMP      $GDDAT,(R5)     ;READ PORT B,DBRB
1038 004750 001401      BEQ      4$             ;PORT B =PORT D?
1039 004752 104002      ERROR+  2              ;CHECK DBRA FOR NO DATA CHANGE
1040 004754 013737 001344 001122 4$:  MOV      DRDBA,$BDADR    ;STORE DBRA ADDRESS
1041 004762 005037 001124      CLR      $GDDAT         ;CONTENTS = 0
1042 004766 016215 000002      MOV      2(R2),(R5)     ;READ DBRA FOR CLEAR
1043 004772 023715 001124      CMP      $GDDAT,(R5)    ;DBRA CLEAR?
1044 004776 001401      BEQ      5$             ;YES,CONTINUE
1045 005000 104002      ERROR+  2              ;DBRA INTERACTION ERROR
1046 005002 005723 013572 5$:  TST      (R3)+          ;INC FOR NEXT PATTERN
1047 005004 02C327      CMP      R3,@ENDDAT     ;CHECK FOR END
1048 005010 001266      BNE      1$             ;DO NEXT PATTERN
1049
1050
1051

```

```

*****
;*TEST 20 TEST PORT C TO PORT A INTERACTION
*****

```

```

1052 005012 000004      TST20: SCOPE
1053 005014 004737 013164      JSR      PC,CLRCSR      ;CLEAR ALL CSRS
1054 005020 012703 013272      MOV      @BEGPAT,R3     ;GET DATA PATTERN TABLE
1055 005024 012737 005054 001110  MOV      @1$,$LPERR     ;SET UP SCOPE ADDRESS
1056 005032 013700 001352      MOV      DRCSC,R0       ;GET PORT C, CSRC ADDRESS
1057 005036 013701 001342      MOV      DRCSA,R1       ;GET PORT A, CSRA ADDRESS
1058 005042 013702 001356      MOV      DRCSD,R2       ;STORE CSRD ADDRESS
1059 005046 112762 000001 000001  MOV      @BIT0,1(R2)     ;CSRD IN OUTPUT MODE
1060 005054 010037 001122 1$:  MOV      R0,$BDADR      ;STORE CSRC ADDRESS
1061 005060 005062 000002      CLR      2(R2)         ;CLEAR DBRD
1062 005064 105061 000001      CLR      1(R1)         ;PORT A,CSRA,INPUT MODE
1063 005070 112760 000001 000001  MOV      @BIT0,1(R0)     ;SET CSRC IN OUTPUT MODE
1064 005076 011360 000002      MOV      (R3),2(R0)     ;WRITE INTO DBRC
1065 005102 016104 000002      MOV      2(R1),R4       ;READ DBRA
1066 005106 012737 100600 001124  MOV      @RDY!DIR!BIT7,$GDDAT ;CSRC DIR SHOULD BE SET
1067 005114 011015      MOV      (R0),(R5)     ;READ CSRC
1068 005116 042715 000007      BIC      @7,(R5)       ;CLEAR UNDEFINED BITS
1069 005122 023715 001124      CMP      $GDDAT,(R5)
1070 005126 001401      BEQ      100$          ;CSRC ERROR
1071 005130 104002      ERROR+  2
1072 005132 012737 000300 001124 100$: MOV      @BIT7!BIT6,$GDDAT ;CSRA ADDRESS
1073 005140 010137 001122      MOV      R1,$BDADR     ;READ CSRA
1074 005144 011115      MOV      (R1),(R5)     ;CLEAR UNDEFINED BITS
1075 005146 042715 000007      BIC      @7,(R5)
1076 005152 023715 001124      CMP      $GDDAT,(R5)
1077 005156 001401      BEQ      101$          ;CSRA ERROR
1078 005160 104002      ERROR+  2
1079 005162 013737 001344 001122 101$: MOV      DRDBA,$BDADR    ;STORE DBRA ADDRESS
1080 005170 011337 001124      MOV      (R3),$GDDAT   ;SAVE EXPECTED
1081 005174 010415      MOV      R4,(R5)       ;DBRA CONTENTS
1082 005176 023715 001124      CMP      $GDDAT,(R5)   ;CHECK PORT A,DBRA
1083 005202 001401      BEQ      2$            ;BEQ TO NEXT SUBTEST
1084 005204 104002      ERROR+  2              ;DBRA REG ERROR

```

T20 TEST PORT C TO PORT A INTERACTION

```

1085 005206 005137 001124      2$:  COM      $GDDAT      ;SET UP TO WRITE COM DATA FROM
1086                                     ;PORT A TO PORT C
1087 005212 013737 001354 001122  MOV      DRDBC,$BDADR  ;STORE DBRC ADDRESS
1088 005220 013761 001124 000002  3$:  MOV      $GDDAT,2(R1) ;WRITE PORT A,INPUT MODE
1089 005226 105060 000001          CLR      1(R0)        ;MAKE PORT C INPUT MODE
1090 005232 112761 000001 000001  MOV      #BIT0,1(R1)  ;MAKE PORT A OUTPUT MODE
1091 005240 016015 000002          MOV      2(R0),(R5)   ;READ PORT C,DBRC
1092 005244 023715 001124          CMP      $GDDAT,(R5) ;PORT C =PORT A?
1093 005250 001401          BEQ      4$          ;CHECK DBRD FOR NO DATA CHANGE
1094 005252 104002          ERROR+  2          ;PORT C,DBRC REG ERROR
1095 005254 013737 001360 001122  4$:  MOV      DRDBD,$BDADR ;STORE DBRD ADDRESS
1096 005262 005037 001124          CLR      $GDDAT      ;DBRD = 0 EXPECTED
1097 005266 016215 000002          MOV      2(R2),(R5)  ;READ DBRD FOR CLEAR
1098 005272 023715 001124          CMP      $GDDAT,(R5) ;IS DBRD CLEAR?
1099 005276 001401          BEQ      5$          ;YES,CONTINUE
1100 005300 104002          ERROR+  2          ;DBRD INTERACTION ERROR
1101 005302 005723          5$:  TST      (R3)+       ;INC FOR NEXT PATTERN
1102 005304 020327 013572          CMP      R3,#ENDDAT ;CHECK FOR END
1103 005310 001261          BNE      1$          ;DO NEXT PATTERN
1104
1105

```

```

;*****
;*TEST 21      TEST PORT D TO PORT B INTERACTION
;*****

```

```

005312 000004
1106 005314 004737 013164      TST21: SCOPE
1107 005320 012703 013272      JSR      PC,CLRCSR   ;CLEAR ALL CSRS
1108 005324 012737 005354 001110  MOV      #BEGPAT,R3 ;GET DATA PATTERN TABLE
1109 005332 013700 001356          MOV      #1,$LPERR  ;SET UP SCOPE ADDRESS
1110 005336 013701 001346          MOV      DRCS0,R0   ;GET PORT D, CSRD ADDRESS
1111 005342 013702 001352          MOV      DRCSB,R1  ;GET PORT B, CSRB ADDRESS
1112 005346 112762 000001 000001  MOV      DRCS0,R2   ;STORE CSRD ADDRESS
1113 005354 010037 001122          MOV      #BIT0,1(R2);CSRD IN OUTPUT MODE
1114 005360 005062 000002          1$:  MOV      R0,$BDADR  ;STORE CSRD ADDRESS
1115 005364 105061 000001          CLR      2(R2)      ;CLEAR DBRC
1116 005370 112760 000001 000001  CLR      1(R1)      ;PORT B,CSRB,INPUT MODE
1117 005376 011360 000002          MOV      #BIT0,1(R0);SET CSRD IN OUTPUT MODE
1118 005402 016104 000002          MOV      (R3),2(R0) ;WRITE INTO DBRD
1119 005406 012737 100400 001124  MOV      #RDY!DIR,$GDDAT ;SAVE EXPECTED
1120 005414 011015          MOV      2(R1),R4   ;READ DBRB
1121 005416 023715 001124          CMP      $GDDAT,(R5);READ CSRD
1122 005422 001401          BEQ      100$
1123 005424 104002          ERROR+  2          ;CSRD ERROR
1124 005426 005037 001124          100$: CLR      $GDDAT     ;SAVE EXPECTED
1125 005432 010137 001122          MOV      R1,$BDADR  ;SAVE CSRB ADDRESS
1126 005436 011115          MOV      (R1),(R5)
1127 005440 023715 001124          CMP      $GDDAT,(R5)
1128 005444 001401          BEQ      101$
1129 005446 104002          ERROR+  2          ;CSRB ERROR
1130 005450 013737 001350 001122  101$: MOV      DRDBB,$BDADR ;SAVE DBRB ADDRESS
1131 005456 011337 001124          MOV      (R3),$GDDAT ;SAVE EXPECTED
1132 005462 010415          MOV      R4,(R5)    ;DBRB CONTENTS
1133 005464 023715 001124          CMP      $GDDAT,(R5);CHECK PORT B,DBRB
1134 005470 001401          BEQ      2$          ;BEQ TO NEXT SUBTEST
1135 005472 104002          ERROR+  2          ;DBRB REG ERROR
1136 005474 005137 001124          2$:  COM      $GDDAT   ;SET UP TO WRITE COM DATA FROM
1137                                     ;PORT B TO PORT D
1138 005500 013737 001360 001122  MOV      DRDBD,$BDADR ;STORE DBRD ADDRESS

```


T21 TEST PORT D TO PORT B INTERACTION

```

1139 005506 013761 001124 000002 3$: MOV $GDDAT,2(R1) ;WRITE PORT B,INPUT MODE
1140 005514 105060 000001 CLR 1(R0) ;MAKE PORT D INPUT MODE
1141 005520 112761 000001 000001 MOV 0BITC,1(R1) ;MAKE PORT B OUTPUT MODE
1142 005526 016015 000002 MOV 2(R0),(R5) ;READ PORT D,DBRD
1143 005532 023715 001124 CMP $GDDAT,(R5) ;PORT B -PORT D?
1144 005536 001401 BEQ 4$ ;CHECK DBRC FOR NO DATA CHANGE
1145 005540 104002 ERROR+ 2 ;PORT D,DBRD REG ERROR
1146 005542 013737 001354 001122 4$: MOV DRDBC,$BDADR ;STORE DBRC ADDRESS
1147 005550 005037 001124 CLR $GDDAT ;EXPECT DBRC = 0
1148 005554 016215 000002 MOV 2(R2),(R5) ;READ DBRC
1149 005560 023715 001124 CMP $GDDAT,(R5) ;IS DBRC = 0
1150 005564 001401 BEQ 5$ ;YES,CONTINUE
1151 005566 104002 ERROR+ 2 ;DBRC INTERACTION ERROR
1152 005570 005723 5$: TST (R3)+ ;INC FOR NEXT PATTERN
1153 005572 020327 013572 CMP R3,$ENDDAT ;CHECK FOR END
1154 005576 001266 BNE 1$ ;DO NEXT PATTERN
1155
1156

```

```

;*****
;*TEST 22 TEST GROUP 1,2 IMR,IRR,ACR WITH CHIP RESET
;*****

```

```

005600 000004
1157 005602 004737 013164 TST22: SCOPE
1158 005606 012702 000002 JSR PC,CLRCR ;CLEAR ALL CSRS
1159 005612 013700 001342 MOV #2,R2 ;TWO GROUPS TO TEST
1160 005616 013701 001346 MOV DRCSA,R0 ;STORE CSRA
1161 005622 004737 013234 MOV DRCSB,R1 ;STORE CSRB
1162 005626 010137 001122 111$: JSR PC,CLRIRR ;CLEAR IRR REGISTERS
1163 005632 012737 000377 001124 MOV R1,$BDADR ;STORE ADDRESS
1164 005640 112710 000244 MOV #377,$GDDAT ;STORE EXPECTED DATA
1165 005644 111115 MOVB #MIMR,(R0) ;LOAD MODE BITS FOR IMR
1166 005646 023715 001124 MOVB (R1),(R5) ;READ IMR REGISTER
1167 005652 001401 CMP $GDDAT,(R5)
1168 005654 104005 BEQ 1$
1169 005656 005037 001124 1$: CLR $GDDAT ;IMR REG ERROR
1170 005662 112710 000250 MOVB #MIRR,(R0) ;STORE EXPECTED
1171 005666 111115 MOVB (R1),(R5) ;LOAD MODE BITS FOR IRR
1172 005670 023715 001124 CMP $GDDAT,(R5) ;READ IRR REGISTER
1173 005674 001401 BEQ 2$
1174 005676 104003 ERROR+ 3 ;IRR REG ERROR
1175 005700 112710 000254 2$: MOVB #MACR,(R0) ;LOAD MODE BITS FOR ACR
1176 005704 111115 MOVB (R1),(R5) ;READ ACR REGISTER
1177 005706 023715 001124 CMP $GDDAT,(R5)
1178 005712 001401 BEQ 3$
1179 005714 104004 ERROR+ 4 ;ACR REG ERROR
1180 005716 005302 3$: DEC R2 ;FINISHED BOTH GROUPS?
1181 005720 001405 BEQ TST23 ;;BR IF EQUAL
1182 005722 013700 001352 MOV DRCSA,R0
1183 005726 013701 001356 MOV DRCSB,R1
1184 005732 000735 BR 111$
1185
1186

```

```

;*****
;*TEST 23 TEST GROUPS 1 AND 2 ACR UNIQUENESS
;*****

```

```

005734 000004
1187 005736 004737 013164 TST23: SCOPE
1188 005742 013700 001342 JSR PC,CLRCR ;CLEAR ALL CSRS
1189 005746 013701 001346 MOV DRCSA,R0 ;CSRA ADDRESS
MOV DRCSB,R1 ;CSRB ADDRESS

```

T23 TEST GROUPS 1 AND 2 ACR UNIQUENESS

```

1190 005752 012703 000002      MOV      #2,R3      ;COUNTER FOR TESTING TWO GROUPS
1191 005756 004737 013234      JSR      PC,CLRIRR  ;CLEAR IRR REGS WITH CHIP RESET
1192 005762 010137 001122      MOV      R1,$BDADR  ;STORE ADDRESS
1193 005766 012737 000252      MOV      #252,$GDDAT ;STORE EXPECTED
1194 005774 112710 000254      MOV      #MACR,(R0) ;LOAD MODE BITS FOR ACR
1195 006000 112710 000300      MOV      #PACR,(R0) ;PRESELECT ACR FOR WRITING
1196 006004 113711 001124      MOV      $GDDAT,(R1) ;WRITE INTO DATA PORT
1197 006010 111115      MOV      (R1),(R5)  ;STORE DATA FOR COMPARE
1198 006012 023715 001124      CMP      $GDDAT,(R5) ;CHECK ACR RESULTS
1199 006016 001401      BEQ      1$        ;
1200 006020 104004      ERROR+ 4          ;ACR REG ERROR
1201 006022 112710 000244      MOV      #MIMR,(R0) ;CHANGE TO IMR REGISTER
1202 006026 012737 000377      MOV      #377,$GDDAT ;STORE EXPECTED
1203 006034 111115      MOV      (R1),(R5)  ;READ IMR
1204 006036 023715 001124      CMP      $GDDAT,(R5) ;SHOULD STILL BE ALL 1'S
1205 006042 001401      BEQ      2$        ;
1206 006044 104005      ERROR+ 5          ;IMR REG ERROR
1207 006046 112710 000240      MOV      #MISR,(R0) ;LOAD MODE BITS FOR ISR
1208 006052 005037 001124      CLR      $GDDAT    ;STORE EXPECTED
1209 006056 111115      MOV      (R1),(R5)  ;READ ISR
1210 006060 023715 001124      CMP      $GDDAT,(R5) ;ISR SHOULD BE CLEARED
1211 006064 001401      BEQ      3$        ;
1212 006066 104006      ERROR+ 6          ;ISR REG ERROR
1213 006070 112710 000250      MOV      #MIRR,(R0) ;LOAD MODE BITS FOR IRR
1214 006074 111115      MOV      (R1),(R5)  ;READ IRR
1215 006076 023715 001124      CMP      $GDDAT,(R5) ;IRR SHOULD BE CLEARED
1216 006102 001401      BEQ      4$        ;
1217 006104 104003      ERROR+ 3          ;IRR REG ERROR
1218 006106 105010      CLR      (R0)      ;CHIP RESET GROUP 1
1219 006110 112710 000254      MOV      #MACR,(R0) ;LOAD MODE BITS FOR ACR
1220 006114 111115      MOV      (R1),(R5)  ;READ ACR
1221 006116 023715 001124      CMP      $GDDAT,(R5) ;ACR SHOULD BE CLEARED
1222 006122 001401      BEQ      5$        ;
1223 006124 104004      ERROR+ 4          ;ACR REG ERROR
1224 006126 005303      DEC      R3        ;TEST GROUPS 1 AND 2
1225 006130 001405      BEQ      TST24     ;;BR IF BOTH GROUPS TESTED
1226 006132 013700 001352      MOV      DRCSA,R0  ;GROUP 2 CONTROL PORT
1227 006136 013701 001356      MOV      DRCSB,R1  ;GROUP 2 DATA PORT
1228 006142 000707      BR       111$     ;TEST GROUP 2 ACR UNIQUENESS
1229
1230

```

```

;*****
;*TEST 24 TEST GROUPS 1 AND 2 IMR UNIQUENESS
;*****

```

```

006144 000004
1231
1232 006146 004737 013234      JSR      PC,CLRCSR  ;CLEAR ALL CSRS
1233 006152 013700 001342      JSR      PC,CLRIRR  ;; *** VDRCA1 CHANGES THIS *** ;:GPA
1234 006156 013701 001346      MOV      DRCSA,R0  ;CSRA ADDRESS
1235 006162 012703 000002      MOV      DRCSB,R1  ;CSRB ADDRESS
1236 006166 010137 001122      MOV      #2,R3    ;COUNTER FOR TWO GROUP TESTING
1237 006172 012737 000252      MOV      R1,$BDADR  ;STORE ADDRESS
1238 006200 112710 000244      MOV      #252,$GDDAT ;STORE EXPECTED
1239 006204 112710 000260      MOV      #MIMR,(R0) ;LOAD MODE BITS FOR IMR
1240 006210 113711 001124      MOV      #PIMR,(R0) ;PRESELECT IMR FOR WRITING
1241 006214 111115      MOV      $GDDAT,(R1) ;WRITE INTO DATA PORT
1242 006216 023715 001124      MOV      (R1),(R5)  ;STORE DATA FOR COMPARE
1243 006222 001401      CMP      $GDDAT,(R5) ;CHECK IMR RESULTS
                        BEQ      1$        ;

```

T24 TEST GROUPS 1 AND 2 IMR UNIQUENESS

```

1244 006224 104005          ERROR+ 5          ;IMR ERROR
1245 006226 112710 000254 1$:  MOVB  #MACR,(R0)  ;CHANGE TO ACR REGISTER
1246 006232 005037 001124    CLR  $GDDAT      ;STORE EXPECTED
1247 006236 111115          MOVB  (R1),(R5)   ;READ ACR
1248 006240 023715 001124    CMP  $GDDAT,(R5) ;SHOULD STILL BE CLEARED
1249 006244 001401          BEQ  2$
1250 006246 104004          ERROR+ 4          ;ACR REG ERROR
1251 006250 112710 000240 2$:  MOVB  #MISR,(R0)  ;LOAD MODE BITS FOR ISR
1252 006254 111115          MOVB  (R1),(R5)   ;READ ISR
1253 006256 023715 001124    CMP  $GDDAT,(R5) ;ISR SHOULD BE CLEARED
1254 006262 001401          BEQ  3$
1255 006264 104006          ERROR+ 6          ;ISR REG ERROR
1256 006266 112710 000250 3$:  MOVB  #MIRR,(R0)  ;LOAD MODE BITS FOR IRR
1257 006272 111115          MOVB  (R1),(R5)   ;READ IRR
1258 006274 023715 001124    CMP  $GDDAT,(R5) ;IRR SHOULD BE CLEARED
1259 006300 001401          BEQ  4$
1260 006302 104003          ERROR+ 3          ;IRR REG ERROR
1261 006304 105010          CLRB  (R0)        ;CHIP RESET GROUP 1
1262 006306 112710 000244 4$:  MOVB  #MIMR,(R0)  ;LOAD MODE BITS FOR IMR
1263 006312 012737 000377 001124    MOV  #377,$GDDAT ;STORE EXPECTED
1264 006320 111115          MOVB  (R1),(R5)   ;READ IMR
1265 006322 023715 001124    CMP  $GDDAT,(R5) ;IMR SHOULD BE ALL ONES
1266 006326 001401          BEQ  5$
1267 006330 104005          ERROR+ 5          ;IMR REG ERROR
1268 006332 005303          DEC  R3           ;DO GROUPS 1 AND GROUP 2
1269 006334 001405          BEQ  TST25        ;BR IF BOTH GROUPS TESTED
1270 006336 013700 001352    MOV  DRCSA,R0     ;SET UP TO TEST GROUP 2
1271 006342 013701 001356    MOV  DRCSA,R1     ;GROUP 2 DATA PORT
1272 006346 000707          BR   111$        ;DO GROUP 2 IMR UNIQUENESS
1273
1274

```

```

;*****
;*TEST 25 TEST GROUPS 1 AND 2 IRR UNIQUENESS
;*****

```

```

TST25: SCOPE
1275 006350 000004          JSR  PC,CLRCR     ;CLEAR ALL CSRS
1276 006352 004737 013164    MOV  DRCSA,R0     ;CSRA ADDRESS
1277 006356 013700 001342    MOV  DRCSB,R1     ;CSRB ADDRESS
1278 006362 013701 001346    MOV  #2,R3        ;COUNTER FOR TESTING TWO GROUPS
1279 006366 012703 000002    JSR  PC,CLRIRR    ;CLEAR IRR REGS WITH CHIP RESET
1280 006372 004737 013234 111$: MOV  R1,$BADR     ;STORE ADDRESS
1281 006376 010137 001122    MOV  #200,$GDDAT ;STORE EXPECTED
1282 006402 012737 000200 001124    MOVB #MIRR,(R0)  ;LOAD MODE BITS FOR IRR
1283 006410 112710 000250    MOVB #137,(R0)   ;SET SINGLE IRR BIT7
1284 006414 112710 000137    MOVB (R1),(R5)   ;STORE DATA FOR COMPARE
1285 006420 111115          CMP  $GDDAT,(R5) ;CHECK IRR RESULTS
1286 006422 023715 001124    BEQ  1$
1287 006426 001401          ERROR+ 1$
1288 006430 104003          ERROR+ 3          ;IRR ERROR
1289 006432 112710 000244 1$:  MOVB  #MIMR,(R0)  ;CHANGE TO IMR REGISTER
1290 006436 012737 000377 001124    MOV  #377,$GDDAT ;STORE EXPECTED
1291 006444 111115          MOVB  (R1),(R5)   ;READ IMR
1292 006446 023715 001124    CMP  $GDDAT,(R5) ;SHOULD STILL BE ALL 1'S
1293 006452 001401          BEQ  2$
1294 006454 104005          ERROR+ 5          ;IMR REG ERROR
1295 006456 112710 000240 2$:  MOVB  #MISR,(R0)  ;LOAD MODE BITS FOR ISR
1296 006462 005037 001124    CLR  $GDDAT      ;STORE EXPECTED
1297 006466 111115          MOVB  (R1),(R5)   ;READ ISR
1298 006470 023715 001124    CMP  $GDDAT,(R5) ;ISR SHOULD BE CLEARED

```

T25 TEST GROUPS 1 AND 2 IRR UNIQUENESS

```

1298 006474 001401          BEQ      3$
1299 006476 104006          ERROR+  6          ;ISR REG ERROR
1300 006500 112710 000254    3$:  MOVB   #MACR,(R0)  ;LOAD MODE BITS FOR ACR
1301 006504 111115          MOVB   (R1),(R5)    ;READ ACR
1302 006506 023715 001124    CMP    $GDDAT,(R5) ;ACR SHOULD BE CLEARED
1303 006512 001401          BEQ      4$
1304 006514 104004          ERROR+  4          ;ACR REG ERROR
1305 006516 105010          4$:  CLRB   (R0)        ;CHIP RESET GROUP 1
1306 006520 112710 000250    MOVB   #MIRR,(R0)  ;LOAD MODE BITS FOR IRR
1307 006524 111115          MOVB   (R1),(R5)    ;READ IRR
1308 006526 023715 001124    CMP    $GDDAT,(R5) ;IRR SHOULD BE CLEARED
1309 006532 001401          BEQ      5$
1310 006534 104003          ERROR+  3          ;IRR REG ERROR
1311 006536 005303          5$:  DEC    R3          ;TEST GROUPS 1 AND 2
1312 006540 001405          BEQ    TST26       ;:BR IF BOTH GROUPS TESTED
1313 006542 013700 001352    MOV    DRCSA,R0    ;GROUP 2 CONTROL PORT
1314 006546 013701 001356    MOV    DRCSA,R1    ;GROUP 2 DATA PORT
1315 006552 000711          BR     111$       ;TEST GROUP 2 ACR UNIQUENESS
1316
1317
1318

```

```

;*****
;*TEST 26      TEST GROUPS 1,2 ACR WITH PATTERNS
;*****

```

```

006554 000004
1319 006556 004737 013164    TST26: SCOPE
1320 006562 012702 000002    JSR    PC,CLRCSR   ;CLEAR ALL CSRS
1321 006566 012737 006614 001110  MOV    #2,R2       ;TEST TWO GROUPS
1322 006574 013700 001342    MOV    #1,$LPERR   ;SET UP LOOP RETURN
1323 006600 013701 001346    MOV    DRCSA,R0    ;STORE CSRA ADDRESS
1324 006604 012703 013272    MOV    DRCSB,R1    ;STORE CSRB ADDRESS
1325 006610 010137 001122    111$: MOV    #BEGPAT,R3 ;SET UP PATTERN TABLE
1326 006614 112710 000254    MOV    R1,$BDADR   ;STORE ADDRESS
1327 006620 112710 000300    1$:  MOVB   #MACR,(R0) ;LOAD MODE BITS FOR ACR
1328 006624 111337 001124    MOVB   #PACR,(R0) ;PRESELECT ACR FOR WRITING
1329 006630 111311          MOVB   (R3),$GDDAT ;STORE EXPECTED
1330 006632 111115          MOVB   (R3),(R1)   ;WRITE INTO ACR
1331 006634 023715 001124    MOVB   (R1),(R5)   ;READ OUT OF ACR
1332 006640 001401          CMP    $GDDAT,(R5)
1333 006642 104004          BEQ    2$
1334 006644 005723          ERROR+  4          ;ACR REGISTER ERROR
1335 006646 020327 013572    2$:  TST    (R3)+     ;INC FOR NEXT PATTERN
1336 006652 001360          CMP    R3,#ENDDAT ;CHECK FOR TABLE END
1337 006654 005302          BNE    1$         ;W/R NEXT PATTERN
1338 006656 001405          DEC    R2         ;FINISHED BOTH GROUPS?
1339 006660 013700 001352    BEQ    TST27       ;:BR IF EQUAL
1340 006664 013701 001356    MOV    DRCSA,R0
1341 006670 000745          MOV    DRCSA,R1
1342
1343          BR     111$       ;DO NEXT GROUP

```

```

;*****
;*TEST 27      TEST GROUPS 1,2 IMR WITH PATTERNS
;*****

```

```

006672 000004
1344 006674 004737 013164    TST27: SCOPE
1345 006700 012702 000002    JSR    PC,CLRCSR   ;CLEAR ALL CSRS
1346 006704 012737 006732 001110  MOV    #2,R2       ;TEST TWO GROUPS
1347 006712 013700 001342    MOV    #1,$LPERR   ;SET UP LOOP RETURN
1348 006716 013701 001346    MOV    DRCSA,R0    ;STORE CSRA ADDRESS
1348 006716 013701 001346    MOV    DRCSB,R1    ;STORE CSRB ADDRESS

```

T27 TEST GROUPS 1,2 IMR WITH PATTERNS

```

1349 006722 012703 013272      111$:  MOV      #BEGPAT,R3      ;SET UP PATTERN TABLE
1350 006726 010137 001122      MOV      R1,$BDADR        ;STORE ADDRESS
1351 006732 112710 000244      1$:   MOVVB   #MIMR,(R0)       ;LOAD MODE BITS FOR IMR
1352 006736 112710 000260      MOVVB   #PIMR,(R0)       ;PRESELECT IMR FOR WRITING
1353 006742 111337 001124      MOVVB   (R3),$GDDAT      ;STORE EXPECTED
1354 006746 111311              MOVVB   (R3),(R1)        ;WRITE INTO IMR
1355 006750 111115              MOVVB   (R1),(R5)        ;READ OUT OF IMR
1356 006752 023715 001124      CMP     $GDDAT,(R5)
1357 006756 001401              BEQ     2$
1358 006760 104005              ERROR+  5                ;IMR REGISTER ERROR
1359 006762 005723              2$:   TST     (R3)+          ;INC FOR NEXT PATTERN
1360 006764 020327 013572      CMP     R3,#ENDDAT       ;CHECK FOR TABLE END
1361 006770 001360              BNE    1$                ;W/R NEXT PATTERN
1362 006772 005302              DEC     R2                ;FINISHED BOTH GROUPS?
1363 006774 001405              BEQ     TST30             ;BR IF EQUAL
1364 006776 013700 001352      MOV     DRCSB,R0
1365 007002 013701 001356      MOV     DRCSA,R1
1366 007006 000745              BR     111$              ;DO NEXT GROUP
1367
1368

```

```

;*****
;*TEST 30      TEST GROUP 1,2 CLEAR IMR INSTR.
;*****

```

```

007010 000004
1369 007012 004737 013164      TST30: SCOPE
1370 007016 012704 000002      JSR     PC,CLRCR        ;CLEAR ALL CSRS
1371 007022 013700 001342      MOV     #2,R4           ;COUNTER FOR TWO GROUPS
1372 007026 013701 001346      MOV     DRCSA,R0        ;CSRA ADDRESS
1373 007032 010137 001122      MOV     DRCSB,R1        ;CSRB ADDRESS
1374 007036 005037 001124      111$:  MOV     R1,$BDADR      ;STORE ADDRESS
1375 007042 112710 000244      CLR     $GDDAT          ;EXPECTED DATA
1376 007046 112710 000040      MOVVB   #MIMR,(R0)     ;LOAD MODE BITS TO READ IMR
1377 007052 111115              MOVVB   #CIMR,(R0)     ;CLEAR IMR COMMAND
1378 007054 023715 001124      MOVVB   (R1),(R5)      ;READ DATA PORT
1379 007060 001401              CMP     $GDDAT,(R5)
1380 007062 104005              BEQ     1$
1381 007064 005304              ERROR+  5                ;ERROR ,IMR SHOULD BE CLEARED
1382 007066 001405              1$:   DEC     R4             ;FINISHED BOTH GROUPS?
1383 007070 013700 001352      BEQ     TST31           ;BR IF BOTH GROUPS TESTED
1384 007074 013701 001356      MOV     DRCSB,R0
1385 007100 000754              MOV     DRCSA,R1
1386
1387

```

```

;*****
;*TEST 31      TEST GROUP 1,2 SET IMR INSTR.
;*****

```

```

007102 000004
1388 007104 004737 013164      TST31: SCOPE
1389 007110 012704 000002      JSR     PC,CLRCR        ;CLEAR ALL CSRS
1390 007114 013700 001342      MOV     #2,R4           ;COUNTER FOR TWO GROUPS
1391 007120 013701 001346      MOV     DRCSA,R0        ;CSRA ADDRESS
1392 007124 010137 001122      MOV     DRCSB,R1        ;CSRB ADDRESS
1393 007130 012737 000377 001124  111$:  MOV     R1,$BDADR      ;STORE ADDRESS
1394 007136 112710 000244      MOV     #377,$GDDAT     ;EXPECTED DATA
1395 007142 112710 000040      MOVVB   #MIMR,(R0)     ;LOAD MODE BITS TO READ IMR
1396 007146 112710 000060      MOVVB   #CIMR,(R0)     ;CLEAR IMR
1397 007152 111115              MOVVB   #SIMR,(R0)     ;SET IMR COMMAND
1398 007154 023715 001124      MOVVB   (R1),(R5)      ;READ DATA PORT
1399 007160 001401              CMP     $GDDAT,(R5)

```

T31 TEST GROUP 1,2 SET IMR INSTR.

1400 007162 104005
1401 007164 005304
1402 007166 001405
1403 007170 013700 001352
1404 007174 013701 001356
1405 007200 000751
1406
1407

1\$: ERROR+ 5 ;ERROR ,IMR SHOULD BE SET
DEC R4 ;FINISHED BOTH GROUPS?
BEQ TST32 ;:BR IF BOTH GROUPS TESTED
MOV DRCSA,R0 ;SETUP FOR GROUP 2
MOV DRCSB,R1
BR 111\$;DO IMR TEST WITH GROUP 2

::*****
;*TEST 32 TEST GROUP 1,2 CLEAR SINGLE IMR BIT INSTR.
::*****

007202 000004
1408 007204 004737 013164
1409 007210 012704 000002
1410 007214 012737 007252 001110
1411 007222 013700 001342
1412 007226 013701 001346
1413 007232 010137 001122
1414 007236 012703 013436
1415 007242 112710 000244
1416 007246 012702 000050
1417 007252 111337 001124
1418 007256 112710 000060
1419 007262 110210
1420 007264 111115
1421 007266 023715 001124
1422 007272 001401
1423 007274 104005
1424 007276 005723
1425 007300 020327 013456
1426 007304 001402
1427 007306 005202
1428 007310 000760
1429 007312 005304
1430 007314 001405
1431 007316 013700 001352
1432 007322 013701 001356
1433 007326 000741
1434
1435
1436

TST32: SCOPE
JSR PC,CLRCR ;CLEAR ALL CSRS
MOV #2,R4 ;GROUP COUNTER
MOV #1,\$LPERR ;SCOPE RETURN ADDRESS
MOV DRCSA,R0 ;CSRA ADDRESS
MOV DRCSB,R1 ;CSRB ADDRESS
111\$: MOV R1,\$BDADR ;STORE ADDRESS
MOV #BGCHP4,R3 ;GOOD DATA PATTERN TABLE
MOVB #MIMR,(R0) ;LOAD MODE BITS FOR IMR
MOV #CSIMR,R2 ;CLEAR SINGLE IMR BIT VALUE
1\$: MOVB (R3),\$GDDAT ;STORE EXPECTED
MOVB #SIMR,(R0) ;SET ALL IMR BITS
MOVB R2,(R0) ;CLEAR SINGLE IMR BIT
MOVB (R1),(R5) ;READ DATA PORT
CMP \$GDDAT,(R5)
BEQ 2\$
ERROR+ 5 ;IMR REG ERROR
2\$: TST (R3)+ ;INC EXPECTED DATA TABLE
CMP R3,#EDCHP4 ;CHECK FOR END
BEQ 3\$
INC R2 ;SET UP TO CLEAR NEXT IMR BIT
BR 1\$;CLEAR NEXT IMR BIT
3\$: DEC R4 ;DO GROUPS 1 AND 2
BEQ TST33 ;:BR IF BOTH GROUPS TESTED
MOV DRCSA,R0 ;CSRA ADDRESS
MOV DRCSB,R1 ;CSRB ADDRESS
BR 111\$;DO GROUP 2

::*****
;*TEST 33 TEST GROUP 1,2 SET SINGLE IMR BIT INSTR.
::*****

007330 000004
1437 007332 004737 013164
1438 007336 012704 000002
1439 007342 012737 007400 001110
1440 007350 013700 001342
1441 007354 013701 001346
1442 007360 010137 001122
1443 007364 012703 013374
1444 007370 112710 000244
1445 007374 012702 000070
1446 007400 111337 001124
1447 007404 112710 000040
1448 007410 110210
1449 007412 111115
1450 007414 023715 001124

TST33: SCOPE
JSR PC,CLRCR ;CLEAR ALL CSRS
MOV #2,R4 ;GROUP COUNTER
MOV #1,\$LPERR ;SCOPE RETURN ADDRESS
MOV DRCSA,R0 ;CSRA ADDRESS
MOV DRCSB,R1 ;CSRB ADDRESS
111\$: MOV R1,\$BDADR ;STORE ADDRESS
MOV #BGCHP3,R3 ;GOOD DATA PATTERN TABLE
MOVB #MIMR,(R0) ;LOAD MODE BITS FOR IMR
MOV #SSIMR,R2 ;SET SINGLE IMR BIT VALUE
1\$: MOVB (R3),\$GDDAT ;STORE EXPECTED
MOVB #CIMR,(R0) ;CLEAR ALL IMR BITS
MOVB R2,(R0) ;SET SINGLE IMR BIT
MOVB (R1),(R5) ;READ DATA PORT
CMP \$GDDAT,(R5)

T33 TEST GROUP 1,2 SET SINGLE IMR BIT INSTR.

```

1451 007420 001401      BEQ      2$
1452 007422 104005      ERROR+  5          ;IMR REG ERROR
1453 007424 005723      2$:      TST      (R3)+    ;INC EXPECTED DATA TABLE
1454 007426 020327 013414  CMP      R3,#EDCHP3 ;CHECK FOR END
1455 007432 001402      BEQ      3$
1456 007434 005202      INC      R2          ;SET UP TO SET NEXT IMR BIT
1457 007436 000760      BR       1$          ;SET NEXT IMR BIT
1458 007440 005304      3$:      DEC      R4          ;DO GROUPS 1 AND 2
1459 007442 001405      BEQ      TST34       ;:BR IF BOTH GROUPS TESTED
1460 007444 013700 001352  MOV      DRCSC,R0    ;CSRC ADDRESS
1461 007450 013701 001356  MOV      DRCSD,R1    ;CSRD ADDRESS
1462 007454 000741      BR       111$       ;DO GROUP 2

```

```

;*****
;*TEST 34      TEST GROUP 1,2 SET IRR INSTR.
;*****

```

```

007456 000004
1465 007460 004737 013164  T34:  SCOPE
1466 007464 012704 000002      JSR      PC,CLRCSR   ;CLEAR ALL CSRS
1467 007470 013700 001342      MOV      #2,R4       ;COUNTER FOR TWO GROUPS
1468 007474 013701 001346      MOV      DRCSA,R0    ;CSRA ADDRESS
1469 007500 004737 013234      MOV      DRCSE,R1    ;CSRE ADDRESS
1470 007504 010137 001122      JSR      PC,CLRIRR   ;CLEAR IRR REGS WITH CHIP RESET
1471 007510 012737 000377 001124  111$:  MOV      R1,#BDADR   ;STORE ADDRESS
1472 007516 112710 000250      MOV      #377,$GDDAT ;EXPECTED DATA
1473 007522 112710 000120      MOV      #MIRR,(R0)  ;LOAD MODE BITS TO READ IRR
1474 007526 111115      MOV      #SIRR,(R0)  ;SET IRR COMMAND
1475 007530 023715 001124      MOV      (R1),(R5)   ;READ DATA PORT
1476 007534 001401      CMP      $GDDAT,(R5)
1477 007536 104003      BEQ      1$
1478 007540 005304      1$:      ERROR+  3          ;ERROR ,IRR SHOULD BE SET
1479 007542 001405      DEC      R4          ;FINISHED BOTH GROUPS?
1480 007544 013700 001352      BEQ      TST35       ;:BR IF BOTH GROUPS TESTED
1481 007550 013701 001356  MOV      DRCSC,R0    ;SETUP FOR GROUP 2
1482 007554 000753      MOV      DRCSD,R1
1483      BR       111$       ;DO IRR TEST WITH GROUP 2
1484
1485

```

```

;*****
;*TEST 35      TEST GROUP 1,2 CLEAR IRR INSTR.
;*****

```

```

007556 000004
1486 007560 004737 013164  TST35: SCOPE
1487 007564 012704 000002      JSR      PC,CLRCSR   ;CLEAR ALL CSRS
1488 007570 013700 001342      MOV      #2,R4       ;COUNTER FOR TWO GROUPS
1489 007574 013701 001346      MOV      DRCSE,R0    ;CSRE ADDRESS
1490 007600 004737 013234      MOV      DRCSE,R1    ;CSRE ADDRESS
1491 007604 010137 001122      JSR      PC,CLRIRR   ;CLEAR IRR REGS WITH CHIP RESET
1492 007610 005037 001124      MOV      R1,#BDADR   ;STORE ADDRESS
1493 007614 112710 000250      CLR      $GDDAT      ;EXPECTED DATA
1494 007620 112710 000120      MOV      #MIRR,(R0)  ;LOAD MODE BITS TO READ IRR
1495 007624 112710 000100      MOV      #SIRR,(R0)  ;SET IRR BITS
1496 007630 111115      MOV      #CIRR,(R0)  ;CLEAR IRR COMMAND
1497 007632 023715 001124      MOV      (R1),(R5)   ;READ DATA PORT
1498 007636 001401      CMP      $GDDAT,(R5)
1499 007640 104003      BEQ      1$
1500 007642 005304      1$:      ERROR+  3          ;ERROR ,IRR SHOULD BE CLEARED
1501 007644 001405      DEC      R4          ;FINISHED BOTH GROUPS?
1501 007644 001405      BEQ      TST36       ;:BR IF BOTH GROUPS TESTED

```

T35 TEST GROUP 1,2 CLEAR IRR INSTR.

1502 007646 013700 001352
1503 007652 013701 001356
1504 007656 000752
1505
1506

MOV DRCSC,R0 ;SETUP FOR GROUP 2
MOV DRCSD,R1
BR 111\$;DO IRR TEST WITH GROUP 2

; *TEST 36 TEST GROUP 1,2 CLEAR SINGLE IRR BIT INSTR.

007660 000004
1507 007662 004737 013164
1508 007666 012704 000002
1509 007672 012737 007734 001110
1510 007700 013700 001342
1511 007704 013701 001346
1512 007710 004737 013234
1513 007714 010137 001122
1514 007720 012703 013436
1515 007724 112710 000250
1516 007730 012702 000110
1517 007734 111337 001124
1518 007740 112710 000120
1519 007744 110210
1520 007746 111115
1521 007750 023715 001124
1522 007754 001401
1523 007756 104003
1524 007760 005723
1525 007762 020327 013456
1526 007766 001402
1527 007770 005202
1528 007772 000760
1529 007774 005304
1530 007776 001405
1531 010000 013700 001352
1532 010004 013701 001356
1533 010010 000741
1534
1535

TST36: SCOPE
JSR PC,CLRCSR ;CLEAR ALL CSRS
MOV #2,R4 ;GROUP COUNTER
MOV #1\$, \$LPERR ;SCOPE RETURN ADDRESS
MOV DRCSA,R0 ;CSRA ADDRESS
MOV DRCSE,R1 ;CSRE ADDRESS
JSR PC,CLRIRR ;CLEAR IRR REGS WITH CHIP RESET
111\$: MOV R1,\$BDADR ;STORE ADDRESS
MOV #BGCHP4,R3 ;GOOD DATA PATTERN TABLE
MOVB #MIRR,(R0) ;LOAD MODE BITS FOR IRR
MOV #CSIRR,R2 ;CLEAR SINGLE IRR BIT VALUE
1\$: MOVB (R3),\$GDDAT ;STORE EXPECTED
MOVB #SIRR,(R0) ;SET ALL IRR BITS
MOVB R2,(R0) ;CLEAR SINGLE IRR BIT
MOVB (R1),(R5) ;READ DATA PORT
CMP \$GDDAT,(R5)
BEQ 2\$
ERROR+ 3 ;IRR REG ERROR
2\$: TST (R3)+ ;INC EXPECTED DATA TABLE
CMP R3,#EDCHP4 ;CHECK FOR END
BEQ 3\$
INC R2 ;SET UP TO CLEAR NEXT IRR BIT
BR 1\$;CLEAR NEXT IRR BIT
3\$: DEC R4 ;DO GROUPS 1 AND 2
BEQ TST37 ;;BR IF BOTH GROUPS TESTED
MOV DRCSC,R0 ;CSRC ADDRESS
MOV DRCSD,R1 ;CSRD ADDRESS
BR 111\$;DO GROUP 2

; *TEST 37 TEST GROUP 1,2 SET SINGLE IRR BIT INSTR.

010012 000004
1536 010014 004737 013164
1537 010020 012704 000002
1538 010024 012737 010066 001110
1539 010032 013700 001342
1540 010036 013701 001346
1541 010042 004737 013234
1542 010046 010137 001122
1543 010052 012703 013374
1544 010056 112710 000250
1545 010062 012702 000130
1546 010066 111337 001124
1547 010072 112710 000100
1548 010076 110210
1549 010100 111115
1550 010102 023715 001124
1551 010106 001401
1552 010110 104003

TST37: SCOPE
JSR PC,CLRCSR ;CLEAR ALL CSRS
MOV #2,R4 ;GROUP COUNTER
MOV #1\$, \$LPERR ;SCOPE RETURN ADDRESS
MOV DRCSA,R0 ;CSRA ADDRESS
MOV DRCSE,R1 ;CSRE ADDRESS
JSR PC,CLRIRR ;CLEAR IRR REGS WITH CHIP RESET
111\$: MOV R1,\$BDADR ;STORE ADDRESS
MOV #BGCHP3,R3 ;GOOD DATA PATTERN TABLE
MOVB #MIRR,(R0) ;LOAD MODE BITS FOR IRR
MOV #SSIRR,R2 ;SET SINGLE IRR BIT VALUE
1\$: MOVB (R3),\$GDDAT ;STORE EXPECTED
MOVB #CIRR,(R0) ;CLEAR ALL IRR BITS
MOVB R2,(R0) ;SET SINGLE IRR BIT
MOVB (R1),(R5) ;READ DATA PORT
CMP \$GDDAT,(R5)
BEQ 2\$
ERROR+ 3 ;IRR REG ERROR

T37 TEST GROUP 1,2 SET SINGLE IRR BIT INSTR.

```

1553 010112 005723          2$: TST      (R3),          ;INC EXPECTED DATA TABLE
1554 010114 020327 013414  CMP      R3,#EDCMP3      ;CHECK FOR END
1555 010120 001402          BEQ      3$              ;
1556 010122 005202          INC      R2              ;SET UP TO SET NEXT IRR BIT
1557 010124 000760          BR       1$              ;SET NEXT IRR BIT
1558 010126 005304          3$: DEC      R4              ;DO GROUPS 1 AND 2
1559 010130 001405          BEQ      TST40          ;:BR IF BOTH GROUPS TESTED
1560 010132 013700 001352  MOV      DRCSC,R0        ;CSRC ADDRESS
1561 010136 013701 001356  MOV      DRCSD,R1        ;CSRD ADDRESS
1562 010142 000741          BR       111$          ;DO GROUP 2
1563
1564

```

```

;*****
;*TEST 40      TEST GROUP 1,2 CLEAR IRR-IMR INSTR.
;*****

```

```

TST40: SCOPE
1565 010144 000004          JSR      PC,CLRCSR      ;CLEAR ALL CSRS
1566 010146 004737 013164  MOV      #2,R4          ;COUNTER FOR TWO GROUPS
1567 010152 012704 000002  MOV      DRCSA,R0        ;CSRA ADDRESS
1568 010156 013700 001342  MOV      DRCSE,R1        ;CSRE ADDRESS
1569 010162 013701 001346  JSR      PC,CLRIRR      ;CLEAR IRR REGS WITH CHIP RESET
1570 010166 004737 013234  111$: MOV      R1,#BDADR      ;STORE ADDRESS
1571 010172 010137 001122  CLR      #GDDAT         ;EXPECTED DATA
1572 010176 005037 001124  MOVB    #MIRR,(R0)      ;LOAD MODE BITS TO READ IRR
1573 010202 112710 000250  MOVB    #SIRR,(R0)      ;SET IRR BITS
1574 010206 112710 000120  MOVB    #SIMR,(R0)      ;SET IMR BITS
1575 010212 112710 000060  MOVB    #CIRMR,(R0)     ;CLEAR IRR-IMR COMMAND
1576 010216 112710 000020  MOVB    (R1),(R5)       ;READ DATA PORT FOR IRR
1577 010222 111115          MOVB    (R1),(R5)
1578 010224 023715 001124  CMP     #GDDAT,(R5)
1579 010230 001401          BEQ     1$              ;
1580 010232 104003          ERROR.  3              ;ERROR ,IRR SHOULD BE CLEARED
1581 010234 112710 000244  1$:  MOVB    #MIMR,(R0)   ;LOAD MODE BITS TO READ IMR
1582 010240 111115          MOVB    (R1),(R5)       ;READ IMR REG
1583 010242 023715 001124  CMP     #GDDAT,(R5)
1584 010246 001401          BEQ     2$              ;
1585 010250 104005          ERROR.  5              ;IRR-IMR COMMAND DID NOT CLEAR IMR
1586 010252 005304          2$: DEC     R4              ;FINISHED BOTH GROUPS?
1587 010254 001405          BEQ     TST41          ;:BR IF BOTH GROUPS TESTED
1588 010256 013700 001352  MOV     DRCSC,R0        ;SETUP FOR GROUP 2
1589 010262 013701 001356  MOV     DRCSD,R1
1590 010266 000741          BR     111$          ;DO IRR-IMR TEST WITH GROUP 2
1591

```

```

;*****
;*TEST 41      TEST GROUP 1,2 CLEAR SINGLE IRR-IMR BIT INSTR.
;*****

```

```

TST41: SCOPE
1592 010270 000004          JSR      PC,CLRCSR      ;CLEAR ALL CSRS
1593 010272 004737 013164  MOV      #2,R4          ;GROUP COUNTER
1594 010302 012737 010340 001110  MOV      #1,#LPERR      ;SCOPE RETURN ADDRESS
1595 010310 013700 001342  MOV      DRCSA,R0        ;CSRA ADDRESS
1596 010314 013701 001346  MOV      DRCSE,R1        ;CSRE ADDRESS
1597 010320 004737 013234  JSR      PC,CLRIRR      ;CLEAR IRR REGS WITH CHIP RESET
1598 010324 010137 001122  111$: MOV      R1,#BDADR      ;STORE ADDRESS
1599 010330 012703 013436  MOV      #BGCHP4,R3     ;GOOD DATA PATTERN TABLE
1600 010334 012702 000030  MOV      #CSIRMR,R2     ;CLEAR SINGLE IRR-IMR BIT VALUE
1601 010340 111337 001124  1$:  MOVB    (R3),#GDDAT   ;STORE EXPECTED
1602 010344 112710 000250  MOVB    #MIRR,(R0)      ;LOAD MODE BITS TO READ IRR
1603 010350 112710 000120  MOVB    #SIRR,(R0)      ;SET ALL IRR BITS

```

T41 TEST GROUP 1,2 CLEAR SINGLE IRR-IMR BIT INSTR.

```

1604 010354 112710 000060      MOVB    #SIMR,(R0)      ;SET ALL IMR BITS
1605 010360 110210              MOVB    R2,(R0)        ;CLEAR SINGLE IRR-IMR BIT
1606 010362 111115              MOVB    (R1),(R5)      ;READ DATA PORT FOR IRR
1607 010364 023715 001124      CMP     $GDDAT,(R5)
1608 010370 001401              BEQ     2$
1609 010372 104003              ERROR.  3              ;IRR REG ERROR
1610 010374 112710 000244      2$:    MOVB    #MIMR,(R0)  ;SET UP TO READ IMR
1611 010400 111115              MOVB    (R1),(R5)      ;READ IMR REGISTER
1612 010402 023715 001124      CMP     $GDDAT,(R5)
1613 010406 001401              BEQ     3$
1614 010410 104005              ERROR.  5              ;IMR REG ERROR
1615 010412 005723              TST     (R3),          ;INC EXPECTED DATA TABLE
1616 010414 020327 013456      CMP     R3,#EDCHP4     ;CHECK FOR END
1617 010420 001402              BEQ     4$
1618 010422 005202              INC     R2              ;SET UP TO CLEAR NEXT IRR-IMR BIT
1619 010424 000745              BR      1$              ;CLEAR NEXT IRR-IMR BIT
1620 010426 005304              4$:    DEC     R4              ;DO GROUPS 1 AND 2
1621 010430 001405              BEQ     TST42           ;BR IF BOTH GROUPS TESTED
1622 010432 013700 001352      MOV     DRCSC,R0        ;CSRC ADDRESS
1623 010436 013701 001356      MOV     DRCSD,R1        ;CSRD ADDRESS
1624 010442 000730              BR      111$           ;DO GROUP 2
1625
1626

```

```

;*****
;*TEST 42      TEST GROUPS 1,2 FOR GROUP UNIQUENESS
;*****

```

```

1627 010444 000004 013164      TST42: SCOPE
1628 010446 004737              JSR     PC,CLRCSR      ;CLEAR ALL CSRS
1629                                ;CSRA = R0
1630                                ;CSRB = R1
1631                                ;CSRC = R2
1632                                ;CSRD = R3
1632 010452 012704 000002      MOV     #2,R4          ;COUNTER FOR TESTING TWO GROUPS
1633 010456 004737 013234      JSR     PC,CLRIRR     ;CLEAR IRR REGS WITH CHIP RESET
1634 010462 010337 001122      111$:  MOV     R3,#BDADR     ;STORE ADDRESS
1635 010466 112710 000300      MOVB   #PACR,(R0)     ;PRESELECT ACR FOR WRITING
1636 010472 112711 000377      MOVB   #377,(R1)      ;WRITE INTO DATA PORT FOR ACR
1637 010476 112710 000120      MOVB   #SIIR,(R0)     ;SET IRR TO ALL 1'S
1638 010502 112710 000040      MOVB   #CIMR,(R0)     ;CLEAR IMR
1639 010506 112712 000254      MOVB   #MACR,(R2)     ;LOAD MODE TO READ ACR
1640 010512 005037 001124      CLR     $GDDAT        ;EXPECTED
1641 010516 111315              MOVB   (R3),(R5)      ;STORE DATA FOR COMPARE
1642 010520 023715 001124      CMP     $GDDAT,(R5)   ;CHECK ACR RESULTS
1643 010524 001401              BEQ     1$
1644 010526 104004              ERROR.  4              ;ERROR,OTHER GROUP ACR SHOULD BE CLEARED
1645 010530 112712 000244      1$:    MOVB   #MIMR,(R2)   ;CHANGE TO IMR REGISTER
1646 010534 012737 000377 001124  MOV     #377,$GDDAT   ;STORE EXPECTED
1647 010542 111315              MOVB   (R3),(R5)      ;READ IMR
1648 010544 023715 001124      CMP     $GDDAT,(R5)   ;SHOULD BE ALL 1'S
1649 010550 001401              BEQ     2$
1650 010552 104005              ERROR.  5              ;ERROR,OTHER GROUP IMR SHOULD BE SET
1651 010554 112712 000240      2$:    MOVB   #MISR,(R2)   ;LOAD MODE BITS FOR ISR
1652 010560 005037 001124      CLR     $GDDAT        ;STORE EXPECTED
1653 010564 111315              MOVB   (R3),(R5)      ;READ ISR
1654 010566 023715 001124      CMP     $GDDAT,(R5)   ;ISR SHOULD BE CLEARED
1655 010572 001401              BEQ     3$
1656 010574 104006              ERROR.  6              ;ISR REG ERROR
1657 010576 112712 000250      3$:    MOVB   #MIRR,(R2)   ;LOAD MODE BITS FOR IRR

```

T42 TEST GROUPS 1,2 FOR GROUP UNIQUENESS

```

1658 010602 111315          MOV# (R3),(R5)      ;READ IRR
1659 010604 023715 001124  CMP  $GDDAT,(R5)   ;IRR SHOULD BE CLEARED
1660 010610 001401          BEQ  4$
1661 010612 104003          ERROR+ 3             ;ERROR,OTHER GROUP IRR SHOULD BE CLEARED
1662 010614 005304          4$: DEC  R4           ;TEST GROUPS 1 AND 2
1663 010616 001415          BEQ  TST43         ;:BR IF BOTH GROUPS TESTED
1664 010620 004737 013164  JSR  PC,CLRCSR     ;CLEAR ALL CSRS
1665 010624 013700 001352  MOV  DRCSC,R0      ;GROUP 2 CONTROL PORT
1666 010630 013701 001356  MOV  DRCSD,R1      ;GROUP 2 DATA PORT
1667 010634 013702 001342  MOV  DRCSA,R2      ;GROUP 1 CONTROL PORT
1668 010640 013703 001346  MOV  DRCSB,R3      ;GROUP 1 DATA PORT
1669 010644 004737 013234  JSR  PC,CLRIRR     ;CLEAR IRR REGS WITH CHIP RESET
1670 010650 000704          BR   111$         ;TEST GROUP 2 ACR UNIQUENESS
1671
1672

```

```

;*****
;TEST 43 TEST STATUS BITS GINT,S2,S1,S0,GP1,2
;*****

```

```

010652 000004
1673 010654 004737 013164  TST43: SCOPE
1674 010660 013700 001342  JSR  PC,CLRCSR     ;CLEAR ALL CSRS
1675 010664 013701 001346  MOV  DRCSA,R0
1676 010670 012703 000002  MOV  DRCSB,R1
1677 010674 012704 000120  111$: MOV  @2,R3         ;DO TWO GROUPS
1678 010700 005077 170436  MOV  @120,R4       ;EXPECTED STATUS BITS
1679 010704 005077 170442  CLR  @DRCSA         ;INIT CSRA
1680 010710 012702 013274  CLR  @DRCSB         ;INIT CSRC
1681 010714 112760 000001 000001  MOV  @BGPAT1,R2    ;EXPECTED IRR PATTERN
1682 010722 112777 000204 170412  MOV# @BIT0,1(R0)   ;CSR TO OUTPUT MODE
1683 010730 112777 000204 170414  MOV# @204,@DRCSA   ;POLLED MODE FOR CSRA,GROUP 1
1684 010736 112710 000020  MOV# @204,@DRCSB   ;POLLED MODE FOR CSRC,GROUP 2
1685 010742 012737 000070 001372  MOV# @CIRMR,(R0)   ;CLEAR IMR + IRR
1686 010750 012737 000130 001376  MOV  @SSIMR,IMRLOC ;STORE CODE FOR SINGLE IMR
1687 010756 010037 001122  MOV  @SSIRR,IRRLOC ;STORE CODE FOR SINGLE IRR
1688 010762 010437 001124  112$: MOV  R0,$BDADR   ;CSR CHIP COMMAND ADDRESS
1689 010766 113710 001376  MOV  R4,$GDDAT     ;EXPECTED DATA
1690 010772 111015  MOV# IRRLOC,(R0)   ;SET SINGLE IRR BIT
1691 010774 042715 000007  MOV# (R0),(R5)    ;CHECK STATUS
1692 011000 023715 001124  BIC  @7,(R5)       ;: *** VDRCA1 ADDS THIS *** ;:GPA
1693 011004 001401  BEQ  1$
1694 011006 104007  ERROR+ 7           ;CHIP STATUS ERROR
1695 011010 005037 001124  1$: CLR  $GDDAT
1696 011014 010137 001122  MOV  R1,$BDADR     ;CSR CHIP DATA ADDRESS
1697 011020 111237 001124  MOV# (R2),$GDDAT
1698 011024 112710 000250  MOV# @MIRR,(R0)   ;READ IRR
1699 011030 111115  MOV# (R1),(R5)
1700 011032 023715 001124  CMP  $GDDAT,(R5)  ;CHECK IRR
1701 011036 001401  BEQ  2$
1702 011040 104003  ERROR+ 3           ;IRR ERROR
1703 011042 010037 001122  2$: MOV  R0,$BDADR
1704 011046 113710 001372  MOV# IMRLOC,(R0)  ;SET IMR BIT
1705 011052 012737 000320 001124  MOV  @320,$GDDAT  ;CSR EXPECTED DATA
1706 011060 111015  MOV# (R0),(R5)
1707 011062 042715 000007  BIC  @7,(R5)       ;CLEAR UNDEFINED BITS
1708 011066 023715 001124  CMP  $GDDAT,(R5)
1709 011072 001401  BEQ  3$
1710 011074 104007  ERROR+ 7           ;CHIP STATUS ERROR
1711 011076 010137 001122  3$: MOV  R1,$BDADR

```

T43 TEST STATUS BITS GINT,S2,S1,S0,GP1,2

```

1712 011102 011237 001124      MOV      (R2),%GDDAT      ;EXPECTED DATA
1713 011106 112710 000244      MOV      #MIMR,(R0)      ;READ IMR BITS
1714 011112 111115      MOV      (R1),(R5)      ;SAVE IMR READ
1715 011114 023715 001124      CMP      %GDDAT,(R5)
1716 011120 001401      BEQ      4$
1717 011122 104005      ERROR+  5                ;IMR ERROR
1718 011124 005722      4$:  TST      (R2)+        ;NEXT EXPECTED FOR IMR + IRR
1719 011126 020227 013314      CMP      R2,%EDCHP1      ;CHECK FOR END
1720 011132 001407      BEQ      5$
1721 011134 005237 001376      INC      IRRLOC          ;NEXT IRR BIT
1722 011140 005237 001372      INC      IMRLOC          ;NEXT IMR BIT
1723      ;:GPA  INC      R4        ;INDEX EXPECTED STATUS
1724 011144 000240      NOP
1725 011146 000137 010756      JMP      112$           ;DO NEXT STATUS CHECK
1726 011152 005303      5$:  DEC      R3            ;FINISHED BOTH GROUPS?
1727 011154 001406      BEQ      TST44          ;:BR IF EQUAL
1728 011156 013700 001352      MOV      DRCSC,R0
1729 011162 013701 001356      MOV      DRCSD,R1
1730 011166 000137 010674      JMP      111$           ;DO NEXT GROUP

```

1731
1732
1733

```

;*****
;*TEST 44      TEST POLLED MODE;CSRS A,B=OUT C,D=IN,ACTIVE LOW
;*****

```

```

1734 011172 000004 013164      TST44: SCOPE
1735 011174 004737 013164      JSR      PC,CLRCSR      ;CLEAR ALL CSRS
1736      ;R0 = CSRA-GROUP 1 CONTROL
1737      ;R1 = CSRB-GROUP 1 DATA
1738      ;R2 = CSRC-GROUP 2 CONTROL
1739      ;R3 = CSRD-GROUP 2 DATA
1739 011200 012737 011252 001110      MOV      #1$,%LPERR      ;SET FOR SCOPE RETURN
1740 011206 012704 013272      MOV      #BEGPAT,R4      ;START OF PATTERN TABLE
1741 011212 112760 000001 000001      MOV      #BIT0,1(R0)      ;SET CSRA TO OUTPUT MODE
1742 011220 112761 000001 000001      MOV      #BIT0,1(R1)      ;SET CSRB TO OUTPUT MODE
1743 011226 105010      CLR      (R0)            ;CHIP RESET GROUP 1 CSRA
1744 011230 105012      CLR      (R2)            ;CHIP RESET GROUP 2 CSRC
1745 011232 112710 000204      MOV      #204,(R0)        ;LOAD MODE BITS FOR POLLED MODE,GR 1
1746 011236 112712 000204      MOV      #204,(R2)        ;LOAD MODE BITS FOR POLLED MODE,GR2
1747 011242 112710 000250      MOV      #MIRR,(R0)       ;LOAD BITS TO READ IRR GROUP 1
1748 011246 112712 000250      MOV      #MIRR,(R2)       ;LOAD BITS TO READ IRR GROUP 2
1749 011252 011460 000002      1$:  MOV      (R4),2(R0)       ;SET PATTERN DBRA FROM H TO L
1750 011256 012760 177777 000002      MOV      #-1,2(R0)        ;FORCE ALL BITS IN DBRA HIGH
1751 011264 112710 000020      MOV      #CIRMR,(R0)      ;CLEAR IMR+IRR GROUP 1
1752 011270 112712 000020      MOV      #CIRMR,(R2)      ;CLEAR IMR+IRR GROUP 2
1753 011274 005114      COM      (R4)            ;COMPLEMENT TABLE DATA
1754 011276 011460 000002      MOV      (R4),2(R0)       ;XMIT CSR
1755 011302 012760 177777 000002      MOV      #-1,2(R0)        ;FORCE ALL BITS HIGH AGAIN
1756 011310 005114      COM      (R4)            ;RESTORE TABLE DATA
1757 011312 010137 001122      2$:  MOV      R1,%BDADR        ; GROUP 1 DATA PORT
1758 011316 111437 001124      MOV      (R4),%GDDAT
1759 011322 111115      MOV      (R1),(R5)        ;READ IRR, GROUP 1
1760 011324 023715 001124      CMP      %GDDAT,(R5)
1761 011330 001401      BEQ      3$
1762 011332 104003      ERROR+  3                ;IRR ERROR, GROUP 1
1763 011334 010337 001122      3$:  MOV      R3,%BDADR        ;GROUP 2 DATA PORT
1764 011340 116437 000001 001124      MOV      1(R4),%GDDAT     ;BUILD EXPECTED DATA
1765 011346 042737 000360 001124      BIC      #360,%GDDAT     ;SAVE BITS 0-3

```

T44 TEST POLLED MODE;CSRS A,B=OUT C,D-IN,ACTIVE LOW

```

1766 011354 052737 000100 001124      BIS      #100,$GDDAT      ;EXPECTED 0-3,URPLY 6(C)
1767 011362 111315      MOVB     (R3),(R5)      ;READ IRR BITS,GROUP 2
1768 011364 023715 001124      CMP      $GDDAT,(R5)
1769 011370 001401      BEQ      4$
1770 011372 104003      ERROR+  3              ;IRR ERROR,GROUP 2
1771 011374 005762 000002      TST      2(R2)          ;READ DBRC FOR RPLY 4(A)
1772 011400 052737 000020 001124      BIS      #20,$GDDAT    ;STORE EXPECTED
1773 011406 111315      MOVB     (R3),(R5)      ;READ IRR BITS,GROUP 2
1774 011410 023715 001124      CMP      $GDDAT,(R5)
1775 011414 001401      BEQ      5$
1776 011416 104003      ERROR+  3              ;IRR ERROR,GROUP 2
1777 011420 005061 000002      CLR      2(R1)          ;CLEAR DBRB FOR RPY 7(D)
1778 011424 052737 000200 001124      BIS      #200,$GDDAT   ;EXPECTED
1779 011432 111315      MOVB     (R3),(R5)      ;READ IRR BITS GROUP 2
1780 011434 023715 001124      CMP      $GDDAT,(R5)
1781 011440 001401      BEQ      6$
1782 011442 104003      ERROR+  3              ;IRR ERROR,GROUP 2
1783 011444 005763 000002      TST      2(R3)          ;READ DBRD FOR RPLY 5(B)
1784 011450 052737 000040 001124      BIS      #40,$GDDAT
1785 011456 111315      MOVB     (R3),(R5)      ;READ IRR BITS GROUP2
1786 011460 023715 001124      CMP      $GDDAT,(R5)
1787 011464 001401      BEQ      7$
1788 011466 104003      ERROR+  3              ;GET NEXT PATTERN
1789 011470 005724      TST      (R4)+          ;IRR ERROR,GROUP 2
1790 011472 020427 013572      CMP      R4,#ENDDAT    ;INDEX DATA TABLE
1791 011476 001265      BNE      1$            ;CHECK FOR END
1792
1793
1794

```

```

;*****
;*TEST 45      TEST GROUPS 1,2 IN POLLED MODE,NO REPLY
;*****

```

```

1795 011500 000004      TST45:  SCOPE
1796 011502 004737 013164      JSR      PC,CLRCSR     ;CLEAR ALL CSRS
1797
1798
1799
1800 011506 012704 000002      MOV      #2,R4         ;R0 = CSRA-GROUP 1 CONTROL
1801
1802
1803
1804
1805 011512 112760 000001 000001 111$:  MOVB     #BIT0,1(R0)    ;R1 = CSRB-GROUP 1 DATA
1806 011520 112761 000001 000001      MOVB     #BIT0,1(R1)    ;R2 = CSRC-GROUP 2 CONTROL
1807 011526 012760 177777 000002      MOV      #-1,2(R0)     ;R3 = CSRD-GROUP 2 DATA
1808 011534 105010      CLRB    (R0)           ;TWO PASSES
1809 011536 105012      CLRB    (R2)           ;FIRST PASS CSRA,CSRB = OUTPUT
1810 011540 112710 000204      MOVB     #204,(R0)      ;CSRC,CSRD = INPUT
1811 011544 112712 000204      MOVB     #204,(R2)      ;SEC PASS CSRC,CSRD = OUTPUT
1812 011550 112710 000020      MOVB     #CIRMR,(R0)    ;CSRA,CSRB = INPUT
1813 011554 112712 000020      MOVB     #CIRMR,(R2)
1814 011560 005760 000002      TST      2(R0)          ;SET CSR TO OUTPUT MODE
1815 011564 012737 000320 001124      MOV      #320,$GDDAT   ;SET CSR TO OUTPUT MODE
1816 011572 010037 001122      MOV      R0,#BDADR     ;SET ALL ONES DBR FOR H TO L
1817 011576 111015      MOVB     (R0),(R5)      ;CHIP RESET GROUP CSR
1818 011600 042715 000007      BIC      #7,(R5)        ;CHIP RESET GROUP CSR
1819 011604 023715 001124      CMP      $GDDAT,(R5)    ;LOAD MODE BITS FOR POLLED MODE
                          ;LOAD MODE BITS FOR POLLED MODE
                          ;CLEAR IMR+IRR
                          ;CLEAR IMR+IRR
                          ;READ DBR IN OUTPUT MODE,NO REPLY
                          ;STORE EXPECTED
                          ;STORE ADDRESS
                          ;READ STATUS
                          ;CLEAR UNDEFINED BITS
                          ;CHECK STATUS

```

T45 TEST GROUPS 1,2 IN POLLED MODE,NO REPLY

```

1820 011610 001401 BEQ 1$
1821 011612 104007 ERROR+ 7 ;CHIP STATUS ERROR
1822 011614 010237 001122 1$: MOV R2,$BDADR ;STORE ADDRESS
1823 011620 111215 MOV (R2),(R5) ;READ STATUS
1824 011622 042715 000007 BIC #7,(R5) ;CLEAR UNDEFINED BITS
1825 011626 023715 001124 CMP $GDDAT,(R5)
1826 011632 001401 BEQ 2$
1827 011634 104007 ERROR+ 7 ; CHIP STATUS ERROR
1828 011636 012762 000000 000002 2$: MOV #0,2(R2) ;WRITE ONLY,DBR INPUT MODE FOR NO REPLY
1829 011644 005761 000002 TST 2(R1) ;READ DBR OUTPUT MODE,NO REPLY
1830 011650 012763 000000 000002 MOV #0,2(R3) ;WRITE ONLY, DBR INPUT MODE,NO REPLY
1831 011656 010137 001122 MOV R1,$BDADR ; CHIP DATA PORT
1832 011662 112710 000250 MOV #MIRR,(R0) ;LOAD BITS TO READ IRR
1833 011666 005037 001124 CLR $GDDAT ;IRR SHOULD BE ZERO
1834 011672 111115 MOV (R1),(R5) ;READ IRR
1835 011674 023715 001124 CMP $GDDAT,(R5)
1836 011700 001412 BEQ 3$
1837 011702 005737 017344 TST KXTFLAG ; ON KXT11, READ-BEFORE-WRITE...::GPA
1838 011706 001406 BEQ 100$ ;...AT 2$ UPSETS EXPECTED IRR. ::GPA
1839 011710 012737 000300 001124 MOV #300,$GDDAT ; TRY THE ALTERNATE VALUE. ::GPA
1840 011716 023715 001124 CMP $GDDAT,(R5) ::GPA
1841 011722 001401 BEQ 3$ ::GPA
1842 011724 104003 ERROR+ 3 ;IRR ERROR+ ::GPA
1843 011726 005037 001124 CLR $GDDAT ;IRR ERROR+ ::GPA
1844 011732 010337 001122 MOV R3,$BDADR ;CHIP DATA PORT
1845 011736 112712 000250 MOV #MIRR,(R2) ;LOAD MODE BITS TO READ IRR
1846 011742 111315 MOV (R3),(R5) ;READ IRR BITS
1847 011744 023715 001124 CMP $GDDAT,(R5)
1848 011750 001412 BEQ 4$
1849 011752 005737 017344 TST KXTFLAG ; ON KXT11, DITTO. ::GPA
1850 011756 001406 BEQ 101$ ;TRY THE ALTERNATE VALUE. ::GPA
1851 011760 012737 000060 001124 MOV #60,$GDDAT ; TRY THE ALTERNATE VALUE. ::GPA
1852 011766 023715 001124 CMP $GDDAT,(R5) ::GPA
1853 011772 001401 BEQ 4$ ;IRR ERROR+ ::GPA
1854 011774 104003 ERROR+ 3 ;IRR ERROR+ ::GPA
1855 011776 005304 DEC R4 ;FINISHED TWO PASSES
1856 012000 001413 BEQ TST46 ;;BR IF EQUAL
1857 012002 004737 013164 JSR PC,CLRCR ;CLEAR ALL CSRS
1858 012006 013700 001352 MOV DRCS,R0 ;SET UP FOR NEXT PASS
1859 012012 013701 001356 MOV DRCS,R1
1860 012016 013702 001342 MOV DRCS,R2
1861 012022 013703 001346 MOV DRCS,R3
1862 012026 000631 BR 111$ ;DO NEXT PASS
1863
1864

```

```

;*****
;*TEST 46 TEST POLLED MODE;CSRS C,D=OUT A,B=IN,ACTIVE LOW
;*****

```

```

1865 012030 000004 TST46: SCOPE
1866 012032 004737 013164 JSR PC,CLRCR ;CLEAR ALL CSRS
;R0 = CSRA-GROUP 1 CONTROL
;R1 = CSRB-GROUP 1 DATA
;R2 = CSRC-GROUP 2 CONTROL
;R3 = CSRD-GROUP 2 DATA
1867
1868
1869
1870 012036 012737 012110 001110 MOV #1,$LPERR ;SCOPE LOOP RETURN
1871 012044 012704 013272 MOV #BEGPAT,R4 ;PATTERN TABLE
1872 012050 112762 000001 000001 MOV #BIT0,1(R2) ;SET CSRC TO OUTPUT MODE
1873 012056 112763 000001 000001 MOV #BIT0,1(R3) ;SET CSRD TO OUTPUT MODE

```

T46 TEST POLLED MODE;CSRS C,D=OUT A,B=IN,ACTIVE LOW

```

1874 012064 105010          CLRB      (R0)          ;CHIP RESET GROUP 1 CSRA
1875 012066 105012          CLRB      (R2)          ;CHIP RESET GROUP 2 CSRC
1876 012070 112710 000204    MOVB      #204,(R0)     ;LOAD BITS FOR POLLED MODE,GR 1
1877 012074 112712 000204    MOVB      #204,(R2)     ;LOAD BITS FOR POLLED MODE,GR2
1878 012100 112710 000250    MOVB      #MIRR,(R0)    ;LOAD BITS TO READ IRR GRP 1    ;;JRS
1879 012104 112712 000250    MOVB      #MIRR,(R2)    ;LOAD BITS TO READ IRR GRP 2    ;;JRS
1880 012110 011462 000002    MOV      (R4),2(R2)     ;SET PATTERN DBRC FOR H TO L
1881 012114 012762 177777 000002 1$:  MOV      #-1,2(R2)     ;FORCE ALL BITS IN DRDBC HIGH    ;;JRS
1882 012122 112710 000020    MOVB      #CIRMR,(R0)   ;CLEAR IMR+IRR GROUP 1
1883 012126 112712 000020    MOVB      #CIRMR,(R2)   ;CLEAR IMR+IRR GROUP 2
1884 012132 005114          COM      (R4)          ;COMPLEMENT TABLE DATA    ;;JRS
1885 012134 011462 000002    MOV      (R4),2(R2)     ;XMIT CSR    ;;JRS
1886 012140 012762 177777 000002 2$:  MOV      #-1,2(R2)     ;FORCE ALL BITS HIGH    ;;JRS
1887 012146 005114          COM      (R4)          ;RESTORE TABLE DATA    ;;JRS
1888 012150 010137 001122    MOV      R1,$BDADR      ;GROUP 1 DATA PORT
1889 012154 111437 001124    MOVB      (R4),$GDDAT
1890 012160 111115          MOVB      (R1),(R5)     ;READ IRR,GROUP 1
1891 012162 023715 001124    CMP      $GDDAT,(R5)
1892 012166 001401          BEQ      3$
1893 012170 104003          ERROR+   3             ;IRR ERROR,GROUP 1
1894 012172 010337 001122 3$:  MOV      R3,$BDADR      ;GROUP 2 DATA PORT
1895 012176 116437 000001 001124    MOVB      1(R4),$GDDAT
1896 012204 042737 000360 001124    BIC      #360,$GDDAT    ;SAVE BITS 0-3
1897 012212 052737 000020 001124    BIS      #20,$GDDAT    ;EXPECTED 0-3,URPLY 4(A)
1898 012220 111315          MOVB      (R3),(R5)     ;READ IRR BITS,GROUP 2
1899 012222 023715 001124    CMP      $GDDAT,(R5)
1900 012226 001401          BEQ      4$
1901 012230 104003          ERROR+   3             ;IRR ERROR,GROUP 2
1902 012232 005760 000002 4$:  TST      2(R0)          ;READ DBRA FOR RPLY 6(C)
1903 012236 052737 000100 001124    BIS      #100,$GDDAT    ;STORE EXPECTED
1904 012244 111315          MOVB      (R3),(R5)     ;READ IRR BITS,GROUP 2
1905 012246 023715 001124    CMP      $GDDAT,(R5)
1906 012252 001401          BEQ      5$
1907 012254 104003          ERROR+   3             ;IRR ERROR,GROUP 2
1908 012256 005063 000002 5$:  CLR      2(R3)          ;CLEAR DBRD FOR RPY 5(B)
1909 012262 052737 000040 001124    BIS      #40,$GDDAT    ;EXPECTED
1910 012270 111315          MOVB      (R3),(R5)     ;READ IRR BITS GROUP 2
1911 012272 023715 001124    CMP      $GDDAT,(R5)
1912 012276 001401          BEQ      6$
1913 012300 104003          ERROR+   3             ;IRR ERROR,GROUP 2
1914 012302 005761 000002 6$:  TST      2(R1)          ;READ DBRB FOR RPLY 7(D)
1915 012306 052737 000200 001124    BIS      #200,$GDDAT
1916 012314 111315          MOVB      (R3),(R5)     ;READ IRR BITS GROUP2
1917 012316 023715 001124    CMP      $GDDAT,(R5)
1918 012322 001401          BEQ      7$
1919 012324 104003          ERROR+   3             ;IRR ERROR,GROUP 2
1920 012326 005724 7$:  TST      (R4),         ;INDEX DATA TABLE
1921 012330 020427 013572    CMP      R4,#ENDDAT    ;CHECK FOR END
1922 012334 001265          BNE      1$            ;DO NEXT PATTERN

```

1923
1924
1925
1926

;;*****
;*TEST 47 TEST IRR BITS WITH DATA PAT.,POLLED MODE,ACT. HIGH
;;*****

```

TST47: SCOPE
1927 012340 004737 013164    JSR      PC,CLRCSR     ;CLEAR ALL CSRS

```

T47 TEST IRR BITS WITH DATA PAT.,POLLED MODE,ACT. HIGH

```

1928 012344 012703 013272          MOV      #BEGPAT,R3      ;GET DATA PATTERN TABLE
1929 012350 012737 000001 001110  MOV      #1,$LPERR      ;SET UP SCOPE ADDRESS
1930 012356 013700 001342          MOV      DRCSA,R0
1931 012362 013701 001346          MOV      DRCSB,R1
1932 012366 013702 001356          MOV      DRCSB,R2
1933 012372 112760 000001 000001  MOVB     #BIT0,1(R0)    ;CSRA OUTPUT MODE
1934 012400 112761 000001 000001  MOVB     #BIT0,1(R1)    ;CSRB OUTPUT MODE
1935 012406 112710 000224          MOVB     #224,(R0)      ;LOAD MODE BITS FOR POLLED MODE,GP1
1936 012412 112760 000224 000010  MOVB     #224,10(R0)    ;LOAD MODE BITS FOR POLLED MODE,GP2
1937 012420 005060 000002          1$: CLR      2(R0)          ;CLEAR DBRA
1938 012424 112710 000020          MOVB     #CIRMR,(R0)    ;CLEAR IMR + IRR GROUP 1
1939 012430 112760 000020 000010  MOVB     #CIRMR,10(R0)  ;CLEAR IMR + IRR GROUP 2
1940 012436 011360 000002          MOV      (R3),2(R0)     ;STORE PATTERN INTO DBRA
1941 012442 010137 001122          MOV      R1,$BDADR      ;CSRB ADDRESS
1942 012446 112710 000250          MOVB     #MIRR,(R0)     ;LOAD BITS TO READ IRR1
1943 012452 111337 001124          MOVB     (R3),$GDDAT
1944 012456 111115          MOVB     (R1),(R5)      ;READ IRR,GP1
1945 012460 023715 001124          CMP      $GDDAT,(R5)
1946 012464 001401          BEQ      2$
1947 012466 104003          ERROR+  3              ;IRR ERROR,GP1
1948 012470 010237 001122          2$: MOV      R2,$BDADR    ;CSRD ADDRESS
1949 012474 112760 000250 000010  MOVB     #MIRR,10(R0)   ;SET MODE TO READ IRR BITS GROUP 2
1950 012502 116337 000001 001124  MOVB     1(R3),$GDDAT   ;BUILD EXPECTED DATA
1951 012510 042737 000360 001124  BIC      #360,$GDDAT    ;BITS 0-3 EXPECTED
1952 012516 052737 000100 001124  BIS      #100,$GDDAT    ;"A" REPLY EXPECTED
1953 012524 111215          MOVB     (R2),(R5)      ;READ IRR2
1954 012526 023715 001124          CMP      $GDDAT,(R5)
1955 012532 001401          BEQ      3$
1956 012534 104003          ERROR+  3              ;IRR GROUP 2
1957 012536 005723          3$: TST      (R3)        ;INDEX DATA TABLE
1958 012540 020327 013572          CMP      R3,$ENDDAT     ;CHECK FOR PATTERN END
1959 012544 001325          BNE      1$            ;DO NEXT PATTERN

```

```

1960
1961
1962
;*****
; *TEST 50      TEST CSRA AND CSRB WITH RESET
;*****

```

```

012546 000004
012550 012737 000001 001160  TST50: SCOPE
1963 012556 004737 013164          MOV      #1,$TIMES     ;;DO 1 ITERATION
1964 012562 013700 001342          JSR      PC,CLRCR      ;CLEAR ALL CSRS
1965 012566 013701 001346          MOV      DRCSA,R0
1966 012572 112760 001001 000001  MOV      DRCSB,R1
1967 012600 112761 000001 000001  MOVB     #IE!BIT0,1(R0) ;CSRA TO OUTPUT MODE
1968 012606 010037 001122          MOVB     #BIT0,1(R1)   ;SET CSRB TO OUTPUT MODE
1969 012612 012737 100300 001124  MOV      R0,$BDADR     ;STORE CSRA ADDRESS
1970 012620 000005          MOV      #100300,$GDDAT ;RDY + CHIP STATUS EXP'D
1971 012622 011015          RESET
1972 012624 042715 000007          MOV      (R0),(R5)     ;INITIALIZE
1973 012630 023715 001124          BIC      #7,(R5)       ;STORE REC'D
1974 012634 001401          CMP      $GDDAT,(R5)   ;CLEAR UNDEFINED BITS
1975 012636 104002          BEQ      1$           ;MAKE COMPARE
1976 012640 012737 100000 001124  1$: ERROR+  2          ;CSRA ERROR ON RESET
1977 012646 010137 001122          MOV      #100000,$GDDAT ;STORE EXPECTED
1978 012652 011115          MOV      R1,$BDADR     ;CHECK CSRB
1979 012654 023715 001124          MOV      (R1),(R5)     ;STORE IN $BDDAT
1980 012660 001401          CMP      $GDDAT,(R5)   ;MAKE COMPARE
                                BEQ      TST51          ;;BR IF EQUAL

```


T50 TEST CSRA AND CSRB WITH RESET

1981 012662 104002
1982
1983
1984

ERROR+ 2 ;CSRB ERROR WITH RESET

: TEST 51 TEST CSRC AND CSRD WITH RESET

012664 000004
012666 012737 000001 001160
1985 012674 004737 013164
1986 012700 013702 001352
1987 012704 013703 001356
1988 012710 112762 000001 000001
1989 012716 112763 000001 000001
1990 012724 010237 001122
1991 012730 012737 100200 001124
1992 012736 000005
1993 012740 011215
1994 012742 042715 000007
1995 012746 023715 001124
1996 012752 001401
1997 012754 104002
1998 012756 010337 001122
1999 012762 012737 100000 001124
2000 012770 011315
2001 012772 023715 001124
2002 012776 001401
2003 013000 104002
2004 013002
2005

TST51: SCOPE
MOV #1,\$TIMES ;DO 1 ITERATION
JSR PC,CLRCSR
MOV DRCSC,R2
MOV DRCSD,R3
MOVB #BIT0,1(R2) ;SET CSRC TO OUTPUT MODE
MOVB #BIT0,1(R3) ;SET CSRD TO OUTPUT MODE
MOV R2,\$BDADR ;STORE ADDRESS
MOV #100200,\$GDDAT ;RDY + CHIP STATUS
RESET ;INITIALIZE
MOV (R2),(R5) ;STORE CSRC
BIC #7,(R5) ;CLEAR UNDEFINED BITS
CMP \$GDDAT,(R5) ;MAKE COMPARE
BEQ 1\$
ERROR+ 2 ;CSRC ERROR WITH RESET
1\$: MOV R3,\$BDADR ;CHECK CSRD
MOV #100000,\$GDDAT ;STORE EXPECTED
MOV (R3),(R5) ;READ CSRD
CMP \$GDDAT,(R5) ;MAKE COMPARE
BEQ 2\$
2\$: ERROR+ 2 ;CSRD ERROR ON RESET

T51 TEST CSRC AND CSRD WITH RESET

2007
 2008
 2009
 2010
 2011 013002 013701 001342
 2012 013006 000241
 2013 013010 006037 001364
 2014 013014 001412
 2015 013016 162701 000020
 2016 013022 005237 001202
 2017 013026 032737 000001 001364
 2018 013034 001764
 2019 013036 000137 002100
 2020

```

;DON'T REPORT "EOP" UNTIL ALL SELECTED DRV11-J'S HAVE BEEN TESTED.
NXDEV:  MOV    DRCSA,R1      ;INIT TO SETUP NEXT DRV11J ADDRESSES
NXDEV1: CLC                ;CLEAR CARRY FOR DEVICE MAP
        ROR    DMAP         ;LOOK FOR NEXT DRV11J
        BEQ    $EOP        ;BR IF ALL TESTED
        SUB    #20,R1       ;NEXT DRV11-J STARTS -20
        INC    $UNIT       ;UPDATE UNIT NUMBER
        BIT    #1,DMAP      ;IS UNIT SELECTED?
        BEQ    NXDEV1       ;NEXT,IF NOT SELECTED
        JMP    NEXPAS      ;TEST NEXT DRV11-J
  
```

```

.SBTTL  END OF PASS ROUTINE
;*****
;*INCREMENT THE PASS NUMBER ($PASS)
;*TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
;*IF THERES A MONITOR GO TO IT
;*IF THERE ISN'T JUMP TO START1
$EOP:
  
```

013042
 013042 000240
 013044 005037 001102
 013050 005037 001160
 013054 005237 001176
 013060 042737 100000 001176
 013066 005327
 013070 000001
 013072 003022
 013074 012737
 013076 000001
 013100 013070
 013102 104401 013147
 013106 013746 001176
 013112 104405
 013114 104401 013144
 013120 013700 000042
 013124 001405
 013126 000005
 013130 004710
 013132 000240
 013134 000240
 013136 000240
 013140
 013140 000137
 013142 002034
 013144 377 377 000
 013147 015 012 105
 013152 116 104 040
 013155 120 101 123
 013160 123 040 043
 013163 000

```

        NOP
        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
        TYPE    , $ENDMG   ;;TYPE "END PASS #"
        MOV    $PASS,-(SP) ;;SAVE $PASS FOR TYPEOUT
        TYPDS   ;;GO TYPE--DECIMAL ASCII WITH SIGN
        TYPE    , $ENULL   ;;TYPE A NULL CHARACTER
$GET42: MOV    @#42,R0     ;;GET MONITOR ADDRESS
        BEQ    $DOAGN     ;;BRANCH IF NO MONITOR
        RESET   ;;CLEAR THE WORLD
$ENDAD: JSR    PC,(R0)    ;;GO TO MONITOR
        NOP      ;;SAVE ROOM
        NOP      ;;FOR
        NOP      ;;ACT11
$DOAGN: JMP    @(PC)+     ;;RETURN
$RTNAD: .WORD  START1
$ENULL: .BYTE  -1,-1,0   ;;NULL CHARACTER STRING
$ENDMG: .ASCIZ <15><12>/END PASS #/
  
```

PROGRAM SUBROUTINES

2022
 2023
 2024
 2025
 2026
 2027
 2028
 2029
 2030 013164 012705 001126
 2031 013170 013700 001342
 2032 013174 013701 001346
 2033 013200 013702 001352
 2034 013204 013703 001356
 2035 013210 005037 001124
 2036 013214 005015
 2037 013216 005010
 2038 013220 105061 000001
 2039 013224 005012
 2040 013226 105063 000001
 2041 013232 000207
 2042
 2043
 2044
 2045 013234 010046
 2046 013236 013700 001342
 2047 013242 105060 000011
 2048 013246 112760 000001 000001
 2049 013254 005060 000002
 2050 013260 105010
 2051 013262 105060 000010
 2052 013266 012600
 2053 013270 000207
 2054

.SBTTL PROGRAM SUBROUTINES

```

;*****
;CLEAR ALL CONTROL/STATUS REGISTERS
;INIT REGISTERS R0-R4 WITH CSRA-CSRD
;AND STORE $BDDAT INTO R5.
;*****
CLRCSR: MOV    $BDDAT,R5      ;BAD DATA STORAGE IN R5
        MOV    DRCSA,R0     ;CSRA ADDRESS TO R0
        MOV    DRCSB,R1     ;CSRB ADDRESS TO R1
        MOV    DRCSA,R2     ;CSRC ADDRESS TO R2
        MOV    DRCSA,R3     ;CSRD ADDRESS TO R3
        CLR    $GDDAT       ;CLEAR EXPECTED
        CLR    (R5)         ;CLEAR REC'D
        CLR    (R0)         ;CLEAR CSRA;CHIP RESET GROUP 1
        CLRB   1(R1)        ;CLEAR HIGH BYTE CSRB
        CLR    (R2)         ;CLEAR CSRC;CHIP RESET GROUP 2
        CLRB   1(R3)        ;CLEAR HIGH BYTE CSRD
        RTS    PC

;CLEAR IRR REGISTERS, GROUP 1, GROUP 2 WITH CHIP RESET
CLRIRR: MOV    R0,-(SP)
        MOV    DRCSA,R0     ;START OF CSR ADDRESS
        CLRB   11(R0)       ;CSRC TO INPUT MODE
        MOVB   $BIT0,1(R0)  ;CSRA TO OUTPUT MODE
        CLR    2(R0)        ;CLEAR DBRA
        CLRB   (R0)         ;CHIP RESET OF GROUP1
        CLRB   10(R0)       ;CHIP RESET OF GROUP 2
        MOV    (SP)+,R0     ;RESTORE REGISTER
        RTS    PC          ;EXIT
    
```

PROGRAM SUBROUTINES

```

2056
2057      .SBTTL  PATTERNS FOR REGISTER R/W
2058      ;
2059      ;PATTERNS USED FOR LOADING/READING REGISTERS
2060
2061 013272 000000      BEGPAT: 0                ;GROWING 1
2062 013274 000001      BGPAT1: 1
2063 013276 000003              3
2064 013300 000007              7
2065 013302 000017              17
2066 013304 000037              37
2067 013306 000077              77
2068 013310 000177              177
2069 013312 000377              377
2070 013314 000777      EDCHP1: 777
2071 013316 001777              1777
2072 013320 003777              3777
2073 013322 007777              7777
2074 013324 017777              17777
2075 013326 037777              37777
2076 013330 077777              77777
2077 013332 177777              177777
2078 013334 177776      BGCHP2: 177776          ;GROWING 0
2079 013336 177774              177774
2080 013340 177770              177770
2081 013342 177760              177760
2082 013344 177740              177740
2083 013346 177700              177700
2084 013350 177600              177600
2085 013352 177400      EDCHP2: 177400
2086 013354 177000              177000
2087 013356 176000              176000
2088 013360 174000              174000
2089 013362 170000              170000
2090 013364 160000              160000
2091 013366 140000              140000
2092 013370 100000              100000
2093
2094 013372 000000      BGCHP3: 000000
2095 013374 000001              1                ;WALKING 1
2096 013376 000002              2
2097 013400 000004              4
2098 013402 000010              10
2099 013404 000020              20
2100 013406 000040              40
2101 013410 000100              100
2102 013412 000200              200
2103 013414 000400      EDCHP3: 400
2104 013416 001000              1000
2105 013420 002000              2000
2106 013422 004000              4000
2107 013424 010000              10000
2108 013426 020000              20000
2109 013430 040000              40000
2110 013432 100000              100000
2111 013434 177777              177777          ;WALKING 0
2112 013436 177776      BGCHP4: 177776

```

PATTERNS FOR REGISTER R/W

2113	013440	177775	177775
2114	013442	177773	177773
2115	013444	177767	177767
2116	013446	177757	177757
2117	013450	177737	177737
2118	013452	177677	177677
2119	013454	177577	177577
2120	013456	177377	EDCHP4: 177377
2121	013460	176777	176777
2122	013462	175777	175777
2123	013464	173777	173777
2124	013466	167777	167777
2125	013470	157777	157777
2126	013472	137777	137777
2127	013474	077777	077777
2128	013476	177777	177777
2129	013500	000000	ENDPAT: 000000
2130			
2131			;DATA PATTERNS
2132	013502	155555	PATDAT: 155555
2133	013504	133333	133333
2134	013506	066666	066666
2135	013510	125252	125252
2136	013512	052525	052525
2137	013514	177777	177777
2138	013516	000000	000000
2139	013520	107070	107070
2140	013522	070707	070707
2141	013524	144444	144444
2142	013526	033333	033333
2143	013530	011111	011111
2144	013532	022222	022222
2145	013534	044444	044444
2146	013536	111111	111111
2147	013540	166666	166666
2148	013542	010421	010421
2149	013544	021042	021042
2150	013546	031463	031463
2151	013550	042104	042104
2152	013552	063146	063146
2153	013554	073567	073567
2154	013556	104210	104210
2155	013560	114631	114631
2156	013562	135673	135673
2157	013564	146314	146314
2158	013566	156735	156735
2159	013570	167356	167356
2160	013572	000000	ENDDAT: 000000
2161			

SYSMAC ROUTINES

2163
2164
2165

.SBTTL SYSMAC ROUTINES

```

.SBTTL TYPE ROUTINE
*****
;ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
;THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
;NOTE1:      $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
;NOTE2:      $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
;NOTE3:      $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
;
;CALL:
;1) USING A TRAP INSTRUCTION
;      TYPE      ,MESADR      ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
;OR
;      TYPE
;      MESADR
;
;TYPE:  TSTB      $TPFLG      ;;IS THERE A TERMINAL?
        BPL        1$          ;;BR IF YES
        HALT      ;;HALT HERE IF NO TERMINAL
        BR        3$          ;;LEAVE
1$:     MOV        RO,-(SP)     ;;SAVE RO
        MOV        @2(SP),RO   ;;GET ADDRESS OF ASCIZ STRING
        CMPB      @APTENV,$ENV ;;RUNNING IN APT MODE
        BNE      62$          ;;NO,GO CHECK FOR APT CONSOLE
        BITB      @APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
        BEQ      62$          ;;NO,GO CHECK FOR CONSOLE
        MOV        RO,61$      ;;SETUP MESSAGE ADDRESS FOR APT
        JSR      PC,$ATY3     ;;SPOOL MESSAGE TO APT
        .WORD     0           ;;MESSAGE ADDRESS
61$:   BITB      @APTCSUP,$ENVM ;;APT CONSOLE SUPPRESSED
        BNE      60$          ;;YES,SKIP TYPE OUT
        MOVB      (RO)+,-(SP)  ;;PUSH CHARACTER TO BE TYPED ONTO STACK
        BNE      4$          ;;BR IF IT ISN'T THE TERMINATOR
        TST      (SP)+        ;;IF TERMINATOR POP IT OFF THE STACK
60$:   MOV        (SP)+,RO     ;;RESTORE RO
        ADD      @2,(SP)      ;;ADJUST RETURN PC
        RTI          ;;RETURN
4$:    CMPB      @HT,(SP)     ;;BRANCH IF <HT>
        BEQ      8$          ;;BRANCH IF NOT <CRLF>
        CMPB      @CRLF,(SP)
        BNE      5$          ;;POP <CR><LF> EQUIV
        TST      (SP)+        ;;TYPE A CR AND LF
        TYPE
        $CRLF
        CLRB      $CHARCNT    ;;CLEAR CHARACTER COUNT
        BR        2$          ;;GET NEXT CHARACTER
        JSR      PC,$TYPEC     ;;GO TYPE THIS CHARACTER
5$:    CMPB      $FILLC,(SP)+  ;;IS IT TIME FOR FILLER CHARS.?
        BNE      2$          ;;IF NO GO GET NEXT CHAR.
        MOV        $NULL,-(SP) ;;GET # OF FILLER CHARS. NEEDED
        ;;AND THE NULL CHAR.
        DECB      1(SP)       ;;DOES A NULL NEED TO BE TYPED?
        BLT      6$          ;;BR IF NO--GO POP THE NULL OFF OF STACK
        JSR      PC,$TYPEC     ;;GO TYPE A NULL
        DECB      $CHARCNT    ;;DO NOT COUNT AS A COUNT
        BR        7$          ;;LOOP
6$:    MOV
7$:

```

TYPE ROUTINE

```

;HORIZONTAL TAB PROCESSOR
013762 112716 000040      8$:  MOVB  #' (SP)      ;;REPLACE TAB WITH SPACE
013766 004737 014006      9$:  JSR   PC,$TYPEC     ;;TYPE A SPACE
013772 132737 000007 014124  BITB  #7,$CHARCNT    ;;BRANCH IF NOT AT
014000 001372                BNE   9$             ;;TAB STOP
014002 005726                TST   (SP)          ;;POP SPACE OFF STACK
014004 000724                BR    2$             ;;GET NEXT CHARACTER
014006
$TYPEC:
014006 105777 165132      TSTB  @TKS           ;;CHAR IN KYBD BUFFER?
014012 100022                BPL   10$           ;;BR IF NOT
014014 017746 165126      MOV   @TKB, -(SP)    ;;GET CHAR
014020 042716 177600      BIC   #177600,(SP)  ;;STRIP EXTRANEIOUS BITS
014024 122716 000023      CMPB  #XOFF,(SP)    ;;WAS CHAR XOFF
014030 001012                BNE   102$          ;;BR IF NOT
014032
101$:
014032 105777 165106      TSTB  @TKS           ;;WAIT FOR CHAR
014036 100375                BPL   101$          ;;BR IF NOT
014040 117716 165102      MOVB  @TKB,(SP)     ;;GET CHAR
014044 042716 177600      BIC   #177600,(SP)  ;;STRIP IT
014050 122716 000021      CMPB  #XON,(SP)    ;;WAS IT XON?
014054 001366                BNE   101$          ;;BR IF NOT
014056
102$:
014056 005726                TST   (SP)          ;;FIX STACK
014060
10$:
014060 105777 165064      TSTB  @TPS           ;;WAIT UNTIL PRINTER IS READY
014064 100375                BPL   10$           ;;BR IF NOT
014066 116677 000002 165056  MOVB  2(SP),@TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
014074 122766 000015 000002  CMPB  #CR,2(SP)     ;;IS CHARACTER A CARRIAGE RETURN?
014102 001003                BNE   1$            ;;BRANCH IF NO
014104 105037 014124      CLRB  $CHARCNT      ;;YES--CLEAR CHARACTER COUNT
014110 000406                BR    $TYPEX        ;;EXIT
014112 122766 000012 000002 1$:  CMPB  #LF,2(SP)     ;;IS CHARACTER A LINE FEED?
014120 001402                BEQ   $TYPEX        ;;BRANCH IF YES
014122 105227                INCB  (PC)          ;;COUNT THE CHARACTER
014124 000000                $CHARCNT: .WORD 0  ;;CHARACTER COUNT STORAGE
014126 000207                $TYPEX: RTS PC
2166
.SBTTL APT COMMUNICATIONS ROUTINE
;*****
014130 112737 000001 014374  $ATY1: MOVB  #1,$FFLG  ;;TO REPORT FATAL ERROR
014136 112737 000001 014372  $ATY3: MOVB  #1,$MFLG  ;;TO TYPE A MESSAGE
014144 000403                BR    $ATYC
014146 112737 000001 014374  $ATY4: MOVB  #1,$FFLG  ;;TO ONLY REPORT FATAL ERROR
014154
$ATYC:
014154 010046                MOV   R0, -(SP)     ;;PUSH R0 ON STACK
014156 010146                MOV   R1, -(SP)     ;;PUSH R1 ON STACK
014160 105737 014372      TSTB  $MFLG         ;;SHOULD TYPE A MESSAGE?
014164 001450                BEQ   5$            ;;IF NOT: BR
014166 122737 000001 001210  CMPB  #APTENV,$ENV  ;;OPERATING UNDER APT?
014174 001031                BNE   3$            ;;IF NOT: BR
014176 132737 000100 001211  BITB  #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
014204 001425                BEQ   3$            ;;IF NOT: BR
014206 017600 000004      MOV   @4(SP),R0     ;;GET MESSAGE ADDR.
014212 062766 000002 000004  ADD   #2,4(SP)      ;;BUMP RETURN ADDR.
014220 005737 001170      1$:  TST   $MSGTYPE     ;;SEE IF DONE W/ LAST XMISSION?
014224 001375                BNE   1$            ;;IF NOT: WAIT
014226 010037 001204      MOV   R0,$MSGAD     ;;PUT ADDR IN MAILBOX
014232 105720      2$:  TSTB  (R0)         ;;FIND END OF MESSAGE

```

APT COMMUNICATIONS ROUTINE

```

014234 001376          BNE      2$
014236 163700 001204  SUB      $MSGAD,RO      ;;SUB START OF MESSAGE
014242 006200          ASR      RO              ;;GET MESSAGE LNTH IN WORDS
014244 010037 001206  MOV      RO,$MSGLGT     ;;PUT LENGTH IN MAILBOX
014250 012737 000004 001170  MOV      @4,$MSGTYPE    ;;TELL APT TO TAKE MSG.
014256 000413          BR       5$
014260 017637 000004 014304 3$:  MOV      @4(SP),4$      ;;PUT MSG ADDR IN JSR LINKAGE
014266 062766 000002 000004  ADD      @2,4(SP)       ;;BUMP RETURN ADDRESS
014274 013746 177776  MOV      177776,-(SP)   ;;PUSH 177776 ON STACK
014300 004737 013574  JSR      PC,$TYPE      ;;CALL TYPE MACRO
014304 000000          4$:  .WORD    0
014306          5$:
014306 105737 014374 10$:  TSTB    $FFLG          ;;SHOULD REPORT FATAL ERROR?
014312 001416          BEQ      12$          ;;IF NOT: BR
014314 005737 001210  TST     $ENV           ;;RUNNING UNDER APT?
014320 001413          BEQ      12$          ;;IF NOT: BR
014322 005737 001170 11$:  TST     $MSGTYPE       ;;FINISHED LAST MESSAGE?
014326 001375          BNE      11$          ;;IF NOT: WAIT
014330 017637 000004 001172  MOV      @4(SP),$FATAL  ;;GET ERROR #
014336 062766 000002 000004  ADD      @2,4(SP)       ;;BUMP RETURN ADDR.
014344 005237 001170  INC     $MSGTYPE       ;;TELL APT TO TAKE ERROR
014350 105037 014374 12$:  CLRB   $FFLG          ;;CLEAR FATAL FLAG
014354 105037 014373  CLRB   $LFLG          ;;CLEAR LOG FLAG
014360 105037 014372  CLRB   $MFLG          ;;CLEAR MESSAGE FLAG
014364 012601          MOV     (SP)+,R1        ;;POP STACK INTO R1
014366 012600          MOV     (SP)+,RO        ;;POP STACK INTO RO
014370 000207          RTS     PC          ;;RETURN
014372 000          $MFLG: .BYTE 0      ;;MESSG. FLAG
014373 000          $LFLG: .BYTE 0      ;;LOG FLAG
014374 000          $FFLG: .BYTE 0      ;;FATAL FLAG

```

```

000200
000001
000100
000040

```

2167

```

APTSIZE=200
APTENV=001
APTSPool=100
APTCsup=040
.SBTTL BINARY TO OCTAL (ASCII) AND TYPE
*****
*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
*OCTAL (ASCII) NUMBER AND TYPE IT.
* $TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
*CALL:
*   MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
*   TYPOS    ;;CALL FOR TYPEOUT
*   .BYTE   N              ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
*   .BYTE   M              ;;M=1 OR 0
*                               ;;1=TYPE LEADING ZEROS
*                               ;;0=SUPPRESS LEADING ZEROS
*
* $TYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
* $TYPOS OR $TYPOC
*CALL:
*   MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
*   TYPON    ;;CALL FOR TYPEOUT
*
* $TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
*CALL:
*   MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED

```


BINARY TO OCTAL (ASCII) AND TYPE

```

014376 017646 000000          ;* TYPOC          ;;CALL FOR TYPEOUT
014402 116637 000001 014621 $TYPOS: MOV @ (SP), -(SP) ;;PICKUP THE MODE
014410 112637 014623          MOV 1(SP), $OFILL ;;LOAD ZERO FILL SWITCH
014414 062716 000002          MOV (SP), $OMODE+1 ;;NUMBER OF DIGITS TO TYPE
014420 000406                ADD @2, (SP) ;;ADJUST RETURN ADDRESS
014422 112737 000001 014621 $TYPOC: MOV @1, $OFILL ;;SET THE ZERO FILL SWITCH
014430 112737 000006 014623          MOV @6, $OMODE+1 ;;SET FOR SIX(6) DIGITS
014436 112737 000005 014620 $TYPON: MOV @5, $OCNT ;;SET THE ITERATION COUNT
014444 010346                MOV R3, -(SP) ;;SAVE R3
014446 010446                MOV R4, -(SP) ;;SAVE R4
014450 010546                MOV R5, -(SP) ;;SAVE R5
014452 113704 014623          MOV $OMODE+1, R4 ;;GET THE NUMBER OF DIGITS TO TYPE
014456 005404                NEG R4
014460 062704 000006          ADD @6, R4 ;;SUBTRACT IT FOR MAX. ALLOWED
014464 110437 014622          MOV R4, $OMODE ;;SAVE IT FOR USE
014470 113704 014621          MOV $OFILL, R4 ;;GET THE ZERO FILL SWITCH
014474 016605 000012          MOV 12(SP), R5 ;;PICKUP THE INPUT NUMBER
014500 005003                CLR R3 ;;CLEAR THE OUTPUT WORD
014502 006105                1$: ROL R5 ;;ROTATE MSB INTO "C"
014504 000404                BR 3$ ;;GO DO MSB
014506 006105                2$: ROL R5 ;;FORM THIS DIGIT
014510 006105                ROL R5
014512 006105                ROL R5
014514 010503                MOV R5, R3
014516 006103                3$: ROL R3 ;;GET LSB OF THIS DIGIT
014520 105337 014622          DECB $OMODE ;;TYPE THIS DIGIT?
014524 100016                BPL 7$ ;;BR IF NO
014526 042703 177770          BIC @177770, R3 ;;GET RID OF JUNK
014532 001002                BNE 4$ ;;TEST FOR 0
014534 005704                TST R4 ;;SUPPRESS THIS 0?
014536 001403                BEQ 5$ ;;BR IF YES
014540 005204                4$: INC R4 ;;DON'T SUPPRESS ANYMORE 0'S
014542 052703 000060          BIS @'0, R3 ;;MAKE THIS DIGIT ASCII
014546 052703 000040          5$: BIS @' ,R3 ;;MAKE ASCII IF NOT ALREADY
014552 110337 014616          MOV R3, 8$ ;;SAVE FOR TYPING
014556 104401 014616          TYPE ,8$ ;;GO TYPE THIS DIGIT
014562 105337 014620          7$: DECB $OCNT ;;COUNT BY 1
014566 003347                BGT 2$ ;;BR IF MORE TO DO
014570 002402                BLT 6$ ;;BR IF DONE
014572 005204                INC R4 ;;INSURE LAST DIGIT ISN'T A BLANK
014574 000744                BR 2$ ;;GO DO THE LAST DIGIT
014576 012605                6$: MOV (SP), R5 ;;RESTORE R5
014600 012604                MOV (SP), R4 ;;RESTORE R4
014602 012603                MOV (SP), R3 ;;RESTORE R3
014604 016666 000002 000004          MOV 2(SP), 4(SP) ;;SET THE STACK FOR RETURNING
014612 012616                MOV (SP), (SP)
014614 000002                RTI ;;RETURN
014616 000                8$: .BYTE 0 ;;STORAGE FOR ASCII DIGIT
014617 000                .BYTE 0 ;;TERMINATOR FOR TYPE ROUTINE
014620 000                $OCNT: .BYTE 0 ;;OCTAL DIGIT COUNTER
014621 000                $OFILL: .BYTE 0 ;;ZERO FILL SWITCH
014622 000000                $OMODE: .WORD 0 ;;NUMBER OF DIGITS TO TYPE
2168 .SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
*****
;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE

```

CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

```

; *NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
; *BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
; *REPLACED WITH SPACES.
; *CALL:
; *      MOV      NUM, -(SP)      ;;PUT THE BINARY NUMBER ON THE STACK
; *      TYPDS   ;;GO TO THE ROUTINE
; *TYPDS:
014624      010046      MOV      R0, -(SP)      ;;PUSH R0 ON STACK
014624      010146      MOV      R1, -(SP)      ;;PUSH R1 ON STACK
014630      010246      MOV      R2, -(SP)      ;;PUSH R2 ON STACK
014632      010346      MOV      R3, -(SP)      ;;PUSH R3 ON STACK
014634      010546      MOV      R5, -(SP)      ;;PUSH R5 ON STACK
014636      012746      020200      MOV      #20200, -(SP)  ;;SET BLANK SWITCH AND SIGN
014642      016605      000020      MOV      20(SP), R5    ;;GET THE INPUT NUMBER
014646      100004      BPL      1$           ;;BR IF INPUT IS POS.
014650      005405      NEG      R5           ;;MAKE THE BINARY NUMBER POS.
014652      112766      000055      000001      MOVB     #'-.1(SP)    ;;MAKE THE ASCII NUMBER NEG.
014660      005000      1$:      CLR      R0           ;;ZERO THE CONSTANTS INDEX
014662      012703      015040      MOV      #DBLK, R3    ;;SETUP THE OUTPUT POINTER
014666      112723      000040      MOVB     #' ,(R3)+    ;;SET THE FIRST CHARACTER TO A BLANK
014672      005002      2$:      CLR      R2           ;;CLEAR THE BCD NUMBER
014674      016001      015030      MOV      $DTBL(R0), R1 ;;GET THE CONSTANT
014700      160105      3$:      SUB      R1, R5       ;;FORM THIS BCD DIGIT
014702      002402      BLT      4$           ;;BR IF DONE
014704      005202      INC      R2           ;;INCREASE THE BCD DIGIT BY 1
014706      000774      BR       3$
014710      060105      4$:      ADD      R1, R5       ;;ADD BACK THE CONSTANT
014712      005702      TST      R2           ;;CHECK IF BCD DIGIT=0
014714      001002      BNE      5$           ;;FALL THROUGH IF 0
014716      105716      TSTB     (SP)         ;;STILL DOING LEADING 0'S?
014720      100407      BMI      7$           ;;BR IF YES
014722      106316      5$:      ASLB     (SP)         ;;MSD?
014724      103003      BCC      6$           ;;BR IF NO
014726      116663      000001      177777      MOVB     1(SP), -1(R3) ;;YES--SET THE SIGN
014734      052702      000060      6$:      BIS      #'0, R2      ;;MAKE THE BCD DIGIT ASCII
014740      052702      000040      7$:      BIS      #' ,R2      ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
014744      110223      MOVB     R2, (R3)+    ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
014746      005720      TST      (R0)+       ;;JUST INCREMENTING
014750      020027      000010      CMP      R0, #10     ;;CHECK THE TABLE INDEX
014754      002746      BLT      2$           ;;GO DO THE NEXT DIGIT
014756      003002      BGT      8$           ;;GO TO EXIT
014760      010502      MOV      R5, R2      ;;GET THE LSD
014762      000764      BR       6$           ;;GO CHANGE TO ASCII
014764      105726      8$:      TSTB     (SP)+       ;;WAS THE LSD THE FIRST NON-ZERO?
014766      100003      BPL      9$           ;;BR IF NO
014770      116663      177777      177776      MOVB     -1(SP), -2(R3) ;;YES--SET THE SIGN FOR TYPING
014776      105013      9$:      CLRB     (R3)        ;;SET THE TERMINATOR
015000      012605      MOV      (SP)+, R5    ;;POP STACK INTO R5
015002      012603      MOV      (SP)+, R3    ;;POP STACK INTO R3
015004      012602      MOV      (SP)+, R2    ;;POP STACK INTO R2
015006      012601      MOV      (SP)+, R1    ;;POP STACK INTO R1
015010      012600      MOV      (SP)+, R0    ;;POP STACK INTO R0
015012      104401      015040      TYPE     , $DBLK     ;;NOW TYPE THE NUMBER
015016      016666      000002      000004      MOV      2(SP), 4(SP) ;;ADJUST THE STACK
015024      012616      MOV      (SP)+, (SP)
015026      000002      RTI
015030      023420      $DTBL: 10000.      ;;RETURN TO USER

```

CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

G5

```

015032 001750          1000.
015034 000144          100.
015036 000012          10.
015040
2169 $DBLK: .BLKW 4
      .SBTTL ERROR HANDLER ROUTINE
      ;*****
      ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
      ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
      ;*AND GO TO SWRCK ON ERROR
      ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
      ;*SW15=1      HALT ON ERROR
      ;*SW13=1      INHIBIT ERROR TYPEOUTS
      ;*SW09=1      LOOP ON ERROR
      ;*CALL
      ;*          ERROR  N          ;;ERROR=EMT AND N=ERROR ITEM NUMBER
$ERROR:
015050
015050 104407          CKSWR          ;;TEST FOR CHANGE IN SOFT-SWR
015052 104407          CKSWR          ;GO LOOK FOR SWR CHANGE
015054 105237 001103  7$:  INCB          $ERFLG          ;;SET THE ERROR FLAG
015060 001775          BEQ          7$          ;;DON'T LET THE FLAG GO TO ZERO
015062 013777 001102 164052  MOV          $TSTNM,$DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
015070 005237 001112          INC          $ERTTL          ;;INC THE ERROR COUNT
015074 011637 001116          MOV          (SP),$ERRPC          ;;GET ADDRESS OF ERROR INSTRUCTION
015100 162737 000002 001116  SUB          #2,$ERRPC
015106 117737 164004 001114  MOVB         @,$ERRPC,$ITEMB ;;STRIP AND SAVE THE ERROR ITEM CODE
015114 032777 020000 164016  BIT          @BIT13,$SWR    ;;SKIP TIMEOUT IF SET
015122 001004          BNE          20$          ;;SKIP TIMEOUTS
015124 004737 015224          JSR          PC,$SWRCK          ;;GO TO USER ERROR ROUTINE
015130 104401 001165          TYPE          , $CRLF
015134
015134 122737 000001 001210  20$:  CMPB         @APTENV,$ENV    ;;RUNNING IN APT MODE
015142 001007          BNE          2$          ;;NO,SKIP APT ERROR REPORT
015144 113737 001114 015156  MOVB         $ITEMB,21$    ;;SET ITEM NUMBER AS ERROR NUMBER
015152 004737 014146          JSR          PC,$ATY4          ;;REPORT FATAL ERROR TO APT
015156 000
015157 000          21$:  .BYTE          0
015160 000777          .BYTE          0
015162 005777 163752          22$:  BR          22$          ;;APT ERROR LOOP
015166 100002          2$:  TST          @SWR          ;;HALT ON ERROR
015170 000000          BPL          3$          ;;SKIP IF CONTINUE
015172 104407          HALT          ;;HALT ON ERROR!
015174 032777 001000 163736  3$:  CKSWR          ;;TEST FOR CHANGE IN SOFT-SWR
015202 001402          BIT          @BIT09,$SWR    ;;LOOP ON ERROR SWITCH SET?
015204 013716 001110          BEQ          4$          ;;BR IF NO
015210 005737 001162          MOV          $LPERR,(SP)    ;;FUJGE RETURN FOR LOOPING
015214 001402          TST          $ESCAPE          ;;CHECK FOR AN ESCAPE ADDRESS
015216 013716 001162          BEQ          5$          ;;BR IF NONE
015222          MOV          $ESCAPE,(S') ;;FUJGE RETURN ADDRESS FOR ESCAPE
015222 000002          5$:  RTI          ;;RETURN
2170 ;*****
2171 ;GO TYPE ERROR
2172 ;GO UPDATE SOFTWARE SWR IF 'CNTRL/G'
2173 ;*****
2174 015224 113737 001102 001362 SWRCK: MOVB         $TSTNM,$STNUM ;;SET UP TEST # ON ER
2175 015232 004737 015242          JSR          PC,$ERRIYP    ;;GO TYPE ERROR
2176 015236 104407          CKSWR          ;;GO LOOK FOR SWR CHANGE
2177 015240 000207          RTS          PC          ;;RETURN TO ERROR HANDLER

```

ERROR MESSAGE TYPEOUT ROUTINE

2178

```

.SBTTL ERROR MESSAGE TYPEOUT ROUTINE
;*****
;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
$ERRTYP:
015242 015242 104401 001165      TYPE      ,$CRLF      ;; "CARRIAGE RETURN" & "LINE FEED"
015246 015246 010046              MOV      RO,-(SP)    ;; SAVE RO
015250 015250 005000              CLR      RO          ;; PICKUP THE ITEM INDEX
015252 015252 153700 001114      BISB     @#$ITEMB,RO
015256 015256 001004              BNE     1$          ;; IF ITEM NUMBER IS ZERO, JUST
                                ;; TYPE THE PC OF THE ERROR
015260 015260 013746 001116      MOV      $ERRPC,-(SP)
                                ;; SAVE $ERRPC FOR TYPEOUT
                                ;; ERROR ADDRESS
015264 015264 104402              TYPOC     ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
015266 015266 000426              BR       6$          ;; GET OUT
015270 015270 005300      1$:    DEC      RO          ;; ADJUST THE INDEX SO THAT IT WILL
015272 015272 006300              ASL      RO          ;; WORK FOR THE ERROR TABLE
015274 015274 006300              ASL      RO
015276 015276 006300              ASL      RO
015300 015300 062700 001252      ADD      @#$ERRTB,RO ;; FORM TABLE POINTER
015304 015304 012037 015314      MOV      (RO)+,2$   ;; PICKUP "ERROR MESSAGE" POINTER
015310 015310 001404              BEQ     3$          ;; SKIP TYPEOUT IF NO POINTER
015312 015312 104401              TYPE     ;; TYPE THE "ERROR MESSAGE"
015314 015314 000000      2$:    .WORD    0      ;; "ERROR MESSAGE" POINTER GOES HERE
015316 015316 104401 001165      TYPE     ,$CRLF     ;; "CARRIAGE RETURN" & "LINE FEED"
015322 015322 012037 015332      3$:    MOV      (RO)+,4$ ;; PICKUP "DATA HEADER" POINTER
015326 015326 001404              BEQ     5$          ;; SKIP TYPEOUT IF 0
015330 015330 104401              TYPE     ;; TYPE THE "DATA HEADER"
015332 015332 000000      4$:    .WORD    0      ;; "DATA HEADER" POINTER GOES HERE
015334 015334 104401 001165      TYPE     ,$CRLF     ;; "CARRIAGE RETURN" & "LINE FEED"
015340 015340 011000      5$:    MOV      (RO),RO  ;; PICKUP "DATA TABLE" POINTER
015342 015342 001004              BNE     7$          ;; GO TYPE THE DATA
015344 015344 012600      6$:    MOV      (SP)+,RO  ;; RESTORE RO
015346 015346 104401 001165      TYPE     ,$CRLF     ;; "CARRIAGE RETURN" & "LINE FEED"
015352 015352 000207              RTS      PC        ;; RETURN
015354 015354              7$:
015354 015354 013046      MOV      @$(RO)+,-(SP) ;; SAVE @$(RO)+ FOR TYPEOUT
015356 015356 104402              TYPOC     ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
015360 015360 005710              TST      (RO)      ;; IS THERE ANOTHER NUMBER?
015362 015362 001770              BEQ     6$          ;; BR IF NO
015364 015364 104401 015372      TYPE     ,8$       ;; TYPE TWO(2) SPACES
015370 015370 000771              BR       7$        ;; LOOP
015372 015372 040 040 000 8$:    .ASCIZ  / /          ;; TWO(2) SPACES
                                .EVEN

```

2179

```

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

```

SCOPE HANDLER ROUTINE

```

015376          $SCOPE:
015376 104407          CKSWR          ;;TEST FOR CHANGE IN SOFT-SWR
015400 032777 040000 163532 1$: BIT    @BIT14,@SWR      ;;LOOP ON PRESENT TEST?
015406 001114          BNE    $OVER          ;;YES IF SW14=1
                                ;#####START OF CODE FOR THE XOR TESTER#####
015410 000416          $XTSTR: BR    6$          ;;IF RUNNING ON THE "XOR" TESTER CHANGE
                                ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
                                ;;SAVE THE CONTENTS OF THE ERROR VECTOR
015412 013746 000004          MOV    @ERRVEC,-(SP)      ;;SET FOR TIMEOUT
015416 012737 015436 000004          MOV    @5,@ERRVEC      ;;TIME OUT ON XOR?
015424 005737 177060          TST    @177060          ;;RESTORE THE ERROR VECTOR
015430 012637 000004          MOV    (SP)+,@ERRVEC      ;;GO TO THE NEXT TEST
015434 000463          BR    $SVLAD          ;;CLEAR THE STACK AFTER A TIME OUT
015436 022626          5$: CMP    (SP)+,(SP)+      ;;RESTORE THE ERROR VECTOR
015440 012637 000004          MOV    (SP)+,@ERRVEC      ;;LOOP ON THE PRESENT TEST
015444 000423          BR    7$
015446          6$:;#####END OF CODE FOR THE XOR TESTER#####
015446 032777 000400 163464          BIT    @BIT08,@SWR      ;;LOOP ON SPEC. TEST?
015454 001404          BEQ    2$          ;;BR IF NO
015456 127737 163456 001102          CMPB  @SWR,$TSTNM      ;;ON THE RIGHT TEST? SWR<7:0>
015464 001465          BEQ    $OVER          ;;BR IF YES
015466 105737 001103          2$: TSTB  $ERFLG      ;;HAS AN ERROR OCCURRED?
015472 001421          BEQ    3$          ;;BR IF NO
015474 123737 001115 001103          CMPB  $ERMAX,$ERFLG    ;;MAX. ERRORS FOR THIS TEST OCCURRED?
015502 101015          BHI    3$          ;;BR IF NO
015504 032777 001000 163426          BIT    @BIT09,@SWR      ;;LOOP ON ERROR?
015512 001404          BEQ    4$          ;;BR IF NO
015514 013737 001110 001106          7$: MOV    $LPERR,$LPADR    ;;SET LOOP ADDRESS TO LAST SCOPE
015522 000446          BR    $OVER
015524 105037 001103          4$: CLRB  $ERFLG          ;;ZERO THE ERROR FLAG
015530 005037 001160          CLR    $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
015534 000415          BR    1$          ;;ESCAPE TO THE NEXT TEST
015536 032777 004000 163374          3$: BIT    @BIT11,@SWR      ;;INHIBIT ITERATIONS?
015544 001011          BNE    1$          ;;BR IF YES
015546 005737 001176          TST    $PASS          ;;IF FIRST PASS OF PROGRAM
015552 001406          BEQ    1$          ;;INHIBIT ITERATIONS
015554 005237 001104          INC    $ICNT          ;;INCREMENT ITERATION COUNT
015560 023737 001160 001104          CMP    $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
015566 002024          BGE    $OVER          ;;BR IF MORE ITERATION REQUIRED
015570 012737 000001 001104          1$: MOV    @1,$ICNT      ;;REINITIALIZE THE ITERATION COUNTER
015576 013737 015654 001160          MOV    $MXCNT,$TIMES  ;;SET NUMBER OF ITERATIONS TO DO
015604 105237 001102          $SVLAD: INCB  $TSTNM      ;;COUNT TEST NUMBERS
015610 113737 001102 001174          MOVB  $TSTNM,$TESTN    ;;SET TEST NUMBER IN APT MAILBOX
015616 011637 001106          MOV    (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
015622 011637 001110          MOV    (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
015626 005037 001162          CLR    $ESCAPE        ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
015632 112737 000001 001115          MOVB  @1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
015640 013777 001102 163274          $OVER: MOV    $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
015646 013716 001106          MOV    $LPADR,(SP)    ;;FUDGE RETURN ADDRESS
015652 000002          RTI                ;;FIXES PS
015654 000062          $MXCNT: 50.          ;;MAX. NUMBER OF ITERATIONS

```

2180

```

.SBTTL TTY INPUT ROUTINE
;*****
.ENABL LSB
;*****
;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL

```

TTY INPUT ROUTINE

```

; *WHEN OPERATING IN TTY FLAG MODE.
015656 022737 000176 001140 $CKSWR: CMP @SWREG,SWR ;; IS THE SOFT-SWR SELECTED?
015664 001074 BNE 15$ ;; BRANCH IF NO
015666 105777 163252 TSTB @TKS ;; CHAR THERE?
015672 100071 BPL 15$ ;; IF NO, DON'T WAIT AROUND
015674 117746 163246 MOVB @TKB,-(SP) ;; SAVE THE CHAR
015700 042716 177600 BIC @C177,(SP) ;; STRIP-OFF THE ASCII
015704 022726 000007 CMP @7,(SP) ;; IS IT A CONTROL G?
015710 001062 BNE 15$ ;; NO, RETURN TO USER
015712 123727 001134 000001 CMPB $AUTOB,#1 ;; ARE WE RUNNING IN AUTO-MODE?
015720 001456 BEQ 15$ ;; BRANCH IF YES
015722 104401 016413 TYPE , $CNTLG ;; ECHO THE CONTROL-G (+G)
015726 104401 016420 $GTSWR: TYPE , $MSWR ;; TYPE CURRENT CONTENTS
015732 013746 000176 MOV SWREG,-(SP) ;; SAVE SWREG FOR TYPEOUT
015736 104402 TYPOC ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
015740 104401 016431 TYPE , $MNEW ;; PROMPT FOR NEW SWR
015744 005046 19$: CLR -(SP) ;; CLEAR COUNTER
015746 005046 CLR -(SP) ;; THE NEW SWR
015750 105777 163170 7$: TSTB @TKS ;; CHAR THERE?
015754 100375 BPL 7$ ;; IF NOT TRY AGAIN
015756 117746 163164 MOVB @TKB,-(SP) ;; PICK UP CHAR
015762 042716 177600 BIC @C177,(SP) ;; MAKE IT 7-BIT ASCII
015766 021627 000025 9$: CMP (SP),#25 ;; IS IT A CONTROL-U?
015772 001005 BNE 10$ ;; BRANCH IF NOT
015774 104401 016406 TYPE , $CNTLU ;; YES, ECHO CONTROL-U (+U)
016000 062706 000006 20$: ADD @6,SP ;; IGNORE PREVIOUS INPUT
016004 000757 BR 19$ ;; LET'S TRY IT AGAIN
016006 021627 000015 10$: CMP (SP),#15 ;; IS IT A <CR>?
016012 001022 BNE 16$ ;; BRANCH IF NO
016014 005766 000004 TST 4(SP) ;; YES, IS IT THE FIRST CHAR?
016020 001403 BEQ 11$ ;; BRANCH IF YES
016022 016677 000002 163110 MOV 2(SP),@SWR ;; SAVE NEW SWR
016030 062706 000006 11$: ADD @6,SP ;; CLEAR UP STACK
016034 104401 001165 14$: TYPE , $CRLF ;; ECHO <CR> AND <LF>
016040 123727 001135 000001 CMPB $INTAG,#1 ;; RE-ENABLE TTY KBD INTERRUPTS?
016046 001003 BNE 15$ ;; BRANCH IF NOT
016050 012777 000100 163066 MOV @100,@TKS ;; RE-ENABLE TTY KBD INTERRUPTS
016056 000002 15$: RTI ;; RETURN
016060 004737 014006 16$: JSR PC,$TYPEC ;; ECHO CHAR
016064 021627 000060 CMP (SP),#60 ;; CHAR < 0?
016070 002420 BLT 18$ ;; BRANCH IF YES
016072 021627 000067 CMP (SP),#67 ;; CHAR > 7?
016076 003015 BGT 18$ ;; BRANCH IF YES
016100 042726 000060 BIC @60,(SP) ;; STRIP-OFF ASCII
016104 005766 000002 TST 2(SP) ;; IS THIS THE FIRST CHAR
016110 001403 BEQ 17$ ;; BRANCH IF YES
016112 006316 ASL (SP) ;; NO, SHIFT PRESENT
016114 006316 ASL (SP) ;; CHAR OVER TO MAKE
016116 006316 ASL (SP) ;; ROOM FOR NEW ONE.
016120 005266 000002 17$: INC 2(SP) ;; KEEP COUNT OF CHAR
016124 056616 177776 BIS -2(SP),(SP) ;; SET IN NEW CHAR
016130 000707 BR 7$ ;; GET THE NEXT ONE
016132 104401 001164 18$: TYPE , $QUES ;; TYPE ?<CR><LF>
016136 000720 BR 20$ ;; SIMULATE CONTROL-U
.DSABL LSB
; *****
; *THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY

```

TTY INPUT ROUTINE

```

;*CALL:
;* RDCHR
;* RETURN HERE
;*
;:INPUT A SINGLE CHARACTER FROM THE TTY
;:CHARACTER IS ON THE STACK
;:WITH PARITY BIT STRIPPED OFF

016140 011646
016142 016666 000004 000002
016150 105777 162770
016154 100375
016156 117766 162764 000004
016164 042766 177600 000004
016172 026627 000004 000023
016200 001013
016202 105777 162736
016206 100375
016210 117746 162732
016214 042716 177600
016220 022627 000021
016224 001366
016226 000750
016230 026627 000004 000021
016236 001744
016240 026627 000004 000140
016246 002407
016250 026627 000004 000175
016256 003003
016260 042766 000040 000004
016266 000002

$RDCHR: MOV (SP),-(SP) ;:PUSH DOWN THE PC
MOV 4(SP),2(SP) ;:SAVE THE PS
1$: TSTB @TKS ;:WAIT FOR
BPL 1$ ;:A CHARACTER
MOVB @TKB,4(SP) ;:READ THE TTY
BIC @C<177>,4(SP) ;:GET RID OF JUNK IF ANY
CMP 4(SP),@23 ;:IS IT A CONTROL-S?
BNE 3$ ;:BRANCH IF NO
2$: TSTB @TKS ;:WAIT FOR A CHARACTER
BPL 2$ ;:LOOP UNTIL ITS THERE
MOVB @TKB,-(SP) ;:GET CHARACTER
BIC @C177,(SP) ;:MAKE IT 7-BIT ASCII
CMP (SP),@21 ;:IS IT A CONTROL-Q?
BNE 2$ ;:IF NOT DISCARD IT
BR 1$ ;:YES, RESUME
3$: CMP 4(SP),@XON ;:IS IT A RANDOM XON? ;RAN001
BEQ 1$ ;:BRANCH IF YES ;RAN001
CMP 4(SP),@140 ;:IS IT UPPER CASE?
BLT 4$ ;:BRANCH IF YES
CMP 4(SP),@175 ;:IS IT A SPECIAL CHAR?
BGT 4$ ;:BRANCH IF YES
BIC @40,4(SP) ;:MAKE IT UPPER CASE
4$: RTI ;:GO BACK TO USER

;:*****
;:THIS ROUTINE WILL INPUT A STRING FROM THE TTY
;*CALL:
;* RDLIN
;* RETURN HERE
;:INPUT A STRING FROM THE TTY
;:ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
;:TERMINATOR WILL BE A BYTE OF ALL O'S

016270 010346
016272 012703 016376
016276 022703 016406
016302 101405
016304 104410
016306 112613
016310 122713 000177
016314 001003
016316 104401 001164
016322 000763
016324 111337 016374
016330 104401 016374
016334 122723 000015
016340 001356
016342 105063 177777
016346 104401 001166
016352 012603
016354 011646
016356 016666 000004 000002
016364 012766 016376 000004
016372 000002
016374 000
016375 000

$RDLIN: MOV R3,-(SP) ;:SAVE R3
1$: MOV @TTYIN,R3 ;:GET ADDRESS
2$: CMP @TTYIN+8.,R3 ;:BUFFER FULL?
BLOS 4$ ;:BR IF YES
RDCHR ;:GO READ ONE CHARACTER FROM THE TTY
MOVB (SP),R3 ;:GET CHARACTER
10$: CMPB @177,R3 ;:IS IT A RUBOUT
BNE 3$ ;:SKIP IF NOT
4$: TYPE ,QUES ;:TYPE A '?'
BR 1$ ;:CLEAR THE BUFFER AND LOOP
3$: MOVB (R3),9$ ;:ECHO THE CHARACTER
TYPE ,9$
CMPB @15,R3 ;:CHECK FOR RETURN
BNE 2$ ;:LOOP IF NOT RETURN
CLRB -1(R3) ;:CLEAR RETURN (THE 15)
TYPE ,LF ;:TYPE A LINE FEED
MOV (SP),R3 ;:RESTORE R3
MOV (SP),-(SP) ;:ADJUST THE STACK AND PUT ADDRESS OF THE
MOV 4(SP),2(SP) ;: FIRST ASCII CHARACTER ON IT
RTI ;:RETURN
9$: .BYTE 0 ;:STORAGE FOR ASCII CHAR. TO TYPE
.BYTE 0 ;:TERMINATOR

```

TTY INPUT ROUTINE

```

016376          $TTYIN: .BLKB  8.          ;;RESERVE 8 BYTES FOR TTY INPUT
016406      136      125      015  $CNTLU: .ASCIZ  /+U/<15><12>  ;;CONTROL "U"
016411          012      000
016413      136      107      015  $CNTLG: .ASCIZ  /+G/<15><12>  ;;CONTROL "G"
016416          012      000
016420      015      012      123  $MSWR:  .ASCIZ  <15><12>/SWR = /
016423      127      122      040
016426      075      040      000
016431      040      040      116  $MNEW:  .ASCIZ  / NEW = /
016434      105      127      040
016437      075      040      000

```

2181

.SBTTL POWER DOWN AND UP ROUTINES

;POWER DOWN ROUTINE

```

016442      012737      016606      000024  $PWRDN: MOV      @ $ILLUP,@PWRVEC  ;;SET FOR FAST UP
016450      012737      000340      000026      MOV      @340,@PWRVEC+2  ;;PRIO:7
016456      01C046      MOV      R0,-(SP)  ;;PUSH R0 ON STACK
016460      010146      MOV      R1,-(SP)  ;;PUSH R1 ON STACK
016462      010246      MOV      R2,-(SP)  ;;PUSH R2 ON STACK
016464      010346      MOV      R3,-(SP)  ;;PUSH R3 ON STACK
016466      010446      MOV      R4,-(SP)  ;;PUSH R4 ON STACK
016470      010546      MOV      R5,-(SP)  ;;PUSH R5 ON STACK
016472      017746      162442      MOV      @SWR,-(SP)  ;;PUSH @SWR ON STACK
016476      010637      016612      MOV      SP,$SAVR6  ;;SAVE SP
016502      012737      016514      000024      MOV      @ $PWRUP,@PWRVEC  ;;SET UP VECTOR
016510      000000      HALT
016512      000776      BR      -2  ;;HANG UP

```

;POWER UP ROUTINE

```

016514      012737      016606      000024  $PWRUP: MOV      @ $ILLUP,@PWRVEC  ;;SET FOR FAST DOWN
016522      013706      016612      MOV      $SAVR6,SP  ;;GET SP
016526      005037      016612      CLR      $SAVR6  ;;WAIT LOOP FOR THE TTY
016532      005237      016612      1$: INC      $SAVR6  ;;WAIT FOR THE INC
016536      001375      BNE     1$  ;;OF WORD
016540      012677      162374      MOV      (SP)+,@SWR  ;;POP STACK INTO @SWR
016544      012605      MOV      (SP)+,R5  ;;POP STACK INTO R5
016546      012604      MOV      (SP)+,R4  ;;POP STACK INTO R4
016550      012603      MOV      (SP)+,R3  ;;POP STACK INTO R3
016552      012602      MOV      (SP)+,R2  ;;POP STACK INTO R2
016554      012601      MOV      (SP)+,R1  ;;POP STACK INTO R1
016556      012600      MOV      (SP)+,R0  ;;POP STACK INTO R0
016560      012737      016442      000024      MOV      @ $PWRDN,@PWRVEC  ;;SET UP THE POWER DOWN VECTOR
016566      012737      000340      000026      MOV      @340,@PWRVEC+2  ;;PRIO:7
016574      104401      TYPE  ;;REPORT THE POWER FAILURE
016576      016614      $PWRMG: .WORD  PWRMSG  ;;POWER FAIL MESSAGE POINTER
016600      012716      MOV      (PC)+,(SP)  ;;RESTART AT START1
016602      002034      $PWRAD: .WORD  START1  ;;RESTART ADDRESS
016604      000002      RTI
016606      000000      $ILLUP: HALT  ;;THE POWER UP SEQUENCE WAS STARTED
016610      000776      BR      -2  ;; BEFORE THE POWER DOWN WAS COMPLETE
016612      000000      $SAVR6: 0  ;;PUT THE SP HERE

```

2182

```

016614      015      012      122  PWRMSG: .ASCIZ  <15><12>/RESTARTED FROM PWR FAIL/
016617      105      123      124
016622      101      122      124
016625      105      104      040
016630      106      122      117
016633      115      040      120

```


POWER DOWN AND UP ROUTINES

016636 127 122 040
 016641 106 101 111
 016644 114 000
 2183
 2184

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

016646 010046
 016650 016600 000002
 016654 005740
 016656 111000
 016660 006300
 016662 016000 016702
 016666 00C200

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

016670 011646
 016672 016666 000004 000002
 016700 000002

```
;:THIS IS USE TO HANDLE THE "GETPRI" MACRO
$TRAP2: MOV (SP),-(SP) ;:MOVE THE PC DOWN
        MOV 4(SP),2(SP) ;:MOVE THE PSW DOWN
        RTI ;:RESTORE THE PSW
```

```
.SBTTL TRAP TABLE
;THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
;BY THE "TRAP" INSTRUCTION.
; ROUTINE
```

016702 016670
 016704 013574
 016706 014422
 016710 014376
 016712 014436
 016714 014624
 016716 015726
 016720 015656
 016722 016140
 016724 016270

```
$TRPAD: .WORD $TRAP2
        $TYPE ;:CALL=TYPE TRAP+1(104401) TTY TYPEOUT ROUTINE
        $TYPOC ;:CALL=TYPOC TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
        $TYPOS ;:CALL=TYPOS TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
        $TYPON ;:CALL=TYPON TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
        $TYPDS ;:CALL=TYPDS TRAP+5(104405) TYPE DECIMAL NUMBER (WITH SIGN)
        $GTSWR ;:CALL=GTSWR TRAP+6(104406) GET SOFT-SWR SETTING
        $CKSWR ;:CALL=CKSWR TRAP+7(104407) TEST FOR CHANGE IN SOFT-SWR
        $RDCHR ;:CALL=RDCHR TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
        $RDLIN ;:CALL=RDLIN TRAP+11(104411) TTY TYPEIN STRING ROUTINE
```

2185
 2186
 2187
 2188
 2189

016726 015 012 103
 016731 126 104 122
 016734 103 103 040
 016737 104 122 126
 016742 061 061 112
 016745 040 104 111
 016750 101 107 040
 016753 124 105 123
 016756 124 040 120
 016761 101 122 124
 016764 040 061 040
 016767 040 015 012
 016772 000
 2190 016773 015 012 104
 016776 122 126 061
 017001 061 112 040
 017004 103 101 102

```
.SBTTL ASCII MESSAGES
;:GPA TITLED: .ASCIZ <15><12>/CVDRCA DRV11J DIAG TEST PART 1 /<15><12>
;:JRS TITLED: .ASCIZ <15><12>/CVDRCB DRV11J DIAG TEST PART 1 /<15><12> ;:GPA
TITLED: .ASCIZ <15><12>/CVDRCC DRV11J DIAG TEST PART 1 /<15><12> ;:JRS
```

```
TLCABL: .ASCIZ <15><12>/DRV11J CABLE REQ'D/<15><12>
```


ASCII MESSAGES

017246	122	040	040
017251	101	104	122
017254	123	040	040
017257	040	040	105
017262	130	120	103
017265	124	040	040
017270	040	122	103
017273	126	104	000

2201

2202

2203	017276	001116	001362	001122	DT1:	.EVEN
	017304	001124	001126	000000		\$ERRPC, TSTNUM, \$BDADR, \$GDDAT, \$BDDAT, 0

2204	017312	001116	001362	001122	DT2:	\$ERRPC, TSTNUM, \$BDADR, \$GDADR, \$GDDAT, \$BDDAT, 0
------	--------	--------	--------	--------	------	--

	017320	001120	001124	001126		
	017326	000000				

FALCON (KXT-11) UPGRADE ROUTINES.

::GPA

```

2206 .SBTTL FALCON (KXT-11) UPGRADE ROUTINES. ::GPA
2207 :
2208 : THE FOLLOWING ROUTINES HAVE BEEN ADDED TO ALLOW DIAGNOSTIC(S)
2209 : TO RUN ON A FALCON (KXT-11) BASED SYSTEM.
2210 : TO DETERMINE WHETHER WE'RE A FALCON OR NOT, WE'LL SIZE THE 1ST 3/4 OF
2211 : THE I/O PAGE (28K TO 31K). FALCON HAS 2KW LOCAL RAM AT 28K(.4) TO 30K
2212 : AND A MACRO-ODT AT 30K TO 31K. CONSEQUENTLY, ALL I/O DEVICES MUST
2213 : BE PLACED BETWEEN 174000 AND 177776. ADDITIONALLY, WE'LL STRAP THE
2214 : EMT AND TRAP SERVICE LEVEL TO PRI6, AND SET THE HALT VECTOR SO THAT
2215 : WE CAN STOP THE SUCKER !!
2216 :
2217 : TO MINIMIZE THE IMPACT OF THESE CHANGES ON FINAL PROGRAM SIZE, THE
2218 : BULK OF THIS CODE IS PLACED IN THE FLOATING VECTOR SPACE (400-776).
2219 : IF THE CPU AT HAND IS A FALCON (KXT11), IT STAYS THERE (NO HARM DONE).
2220 : OTHERWISE, THE AREA IS RESTORED TO ITS ORIGINAL "TRAP-CATCHER" STATE.
2221 :
2222 FALCON: INC 0-1 ; ONCE-ONLY !!! ::GPA
2223 BNE 1$ ; ::GPA
2224 CALL KXTCHK ; EXECUTE FALCON CHECK ::GPA
2225 1$: TST (PC) ; TEST FALCON FLAG... ::GPA
2226 KXTFLAG: 0 ;...NZ = FALCON... ::GPA
2227 RETURN ;...AND RETURN TO CALLER... ::GPA
2228
2229 $SVPC= ; ::GPA
2230 = 400 ; RESTORE FROM 374:376 AT END ::GPA
2231 KXTCHK: CLR KXTFLAG ; ASSUME NOT FALCON. ::GPA
2232 MOV @04,-(SP) ; SAVE ERROR VECTOR. ::GPA
2233 MOV @2$,@04 ; SET A TRAP CATCHER. ::GPA
2234 MOV @160010,R0 ; FALCON RAM STARTS AT 28K.4. ::GPA
2235 1$: TST (R0) ; ::GPA
2236 240 ; ::GPA
2237 CMP R0,@174000 ; SIZE TO 31K. ::GPA
2238 BLO 1$ ; ::GPA
2239 MOV R0,KXTFLAG ; MUST BE FALCON, SET THE FLAG ::GPA
2240 MOV @40,R0 ; GET PRI1 BIT... ::GPA
2241 BIC R0,@06 ;...AND LOWER BUS-ERROR... ::GPA
2242 BIC R0,@016 ;...BPT... ::GPA
2243 BIC R0,@022 ;...IOT... ::GPA
2244 BIC R0,@032 ;...EMT... ::GPA
2245 BIC R0,@036 ;...AND TRAP SERVICE TO PRI6 ::GPA
2246 MOV @170000,@0140 ; ENABLE "BREAK" HALT. ::GPA
2247 MOV (SP)+,@04 ; RESTORE ERROR VECTOR... ::GPA
2248 RETURN ;...AND RETURN. ::GPA
2249
2250 2$: MOV @3$,(SP) ; TRAP -- NOT A FALCON... ::GPA
2251 RTI ;...CONTINUE. ::GPA
2252 3$: MOV (SP)+,@04 ; RESET ERROR VECTOR ::GPA
2253 MOV @402,R0 ; SET-UP TO RESTORE FLOATING... ::GPA
2254 MOV @0376,R1 ;...VECTORS (400 - 776). ::GPA
2255 MOV SP,R2 ; SAVE STACK POINTER IN R2 ::GPA
2256 MOV @6$,R4
2257 4$: MOV -(R4),-(SP) ; PUSH THE RESTORE CODE... ::GPA
2258 CMP R4,@5$ ;...ONTO THE STACK. ::GPA
2259 BHI 4$ ; ::GPA
2260 MOV SP,PC ; AND EXECUTE IT. ::GPA

```

FALCON (KXT-11) UPGRADE ROUTINES.

::GPA

```

2262
2263
2264
2265 000546 010060 177776
2266 000552 010110
2267 000554 022020
2268 000556 020027 000776
2269 000562 101771
2270 000564 010206
2271 000566 000207
2272 000570
2273
2274
2275
2276
2277
2278 000104
2282
2283 017350
2284 017350
2285 000001

;
; THIS CODE IS RELOCATED TO AND EXECUTED IN THE STACK AREA.
;
5$:
MOV R0,-2(R0) ; RESTORE ..2...
MOV R1,(R0) ; ...HALT (OR IOT).
CMP (R0),.(R0).
CMP R0,#776
BLOS 5$ ; LOOP 'TIL DONE
MOV R2,SP ; THEN RESTORE SP...
RETURN ; ...AND RETURN TO CALLER

6$:
;
; IF FALCON, THIS AREA IS FREE FOR ANY PROGRAM UNIQUE
; CHANGES OR DATA STRUCTURES.
; IF USED, YOU'D BETTER PROTECT IT !!!
;
$FREE= <1000-.>/2 ; FREE WORDS LEFT.

LASTAD= .=$SVPC
.END

```

SYMBOL TABLE

ABASE = 164160	BGPAT1 013274	DSWR = 177570	PVMA = 000340	TST11 003422
ACDW1 = 000000	BIT0 = 000001	DT1 017276	PWRMSG 016614	TST12 003516
ACDW2 = 000000	BIT00 = 000001	DT2 017312	PWRVEC = 000024	TST13 003612
ACPUOP = 000000	BIT01 = 000002	EDCHP1 013314	RDCHR = 104410	TST14 003706
ACRLOC 001400	BIT02 = 000004	EDCHP2 013352	RDLIN = 104411	TST15 004056
ADDW0 = 000000	BIT03 = 000010	EDCHP3 013414	RDY = 100000	TST16 004224
ADDW1 = 000000	BIT04 = 000020	EDCHP4 013456	RESVEC = 000010	TST17 004524
ADDW10 = 000000	BIT05 = 000040	EMTVEC = 000030	R6 = 000006	TST2 002220
ADDW11 = 000000	BIT06 = 000100	EM1 017022	R7 = 000007	TST20 005012
ADDW12 = 000000	BIT07 = 000200	EM2 017041	SCOPE = 000004	TST21 005312
ADDW13 = 000000	BIT08 = 000400	EM3 017063	SIMR = 000060	TST22 005600
ADDW14 = 000000	BIT09 = 001000	EM4 017076	SIRR = 000120	TST23 005734
ADDW15 = 000000	BIT10 = 000002	EM5 017111	SSIMR = 000070	TST24 006144
ADDW2 = 000000	BIT11 = 002000	EM6 017124	SSIRR = 000130	TST25 006350
ADDW3 = 000000	BIT12 = 004000	EM7 017137	STACK = 001100	TST26 006554
ADDW4 = 000000	BIT13 = 010000	ENDDAT 013572	START 001402	TST27 006672
ADDW5 = 000000	BIT14 = 020000	ENDPAT 013500	START1 002034	TST3 002412
ADDW6 = 000000	BIT15 = 040000	ERROR = 104000	STKLMT = 177774	TST30 007010
ADDW7 = 000000	BIT2 = 000004	ERRVEC = 000004	SWR 001140	TST31 007102
ADDW8 = 000000	BIT3 = 000010	FALCON 017330	SWRCK 015224	TST32 007202
ADDW9 = 000000	BIT4 = 000020	GTSWR = 104406	SWREG 000176	TST33 007330
ADEVCT = 000000	BIT5 = 000040	HT = 000011	SWO = 000001	TST34 007456
ADEVN = 000001	BIT6 = 000100	IE = 001000	SW00 = 000001	TST35 007556
AENV = 000000	BIT7 = 000200	IMRLOC 001372	SW01 = 000002	TST36 007660
AENVN = 000000	BIT8 = 000400	INTFLG 001366	SW02 = 000004	TST37 010012
AFATAL = 000000	BIT9 = 001000	IOTVEC = 000020	SW03 = 000010	TST4 002522
AMADR1 = 000000	BIT10 = 000004	IRRLOC 001376	SW04 = 000020	TST40 010144
AMADR2 = 000000	BIT11 = 000010	ISRLOC 001374	SW05 = 000040	TST41 010270
AMADR3 = 000000	BPTVEC = 000014	KXTCHK 000400	SW06 = 000100	TST42 010444
AMADR4 = 000000	CHPISR = 000140	KXTFLA 017344	SW07 = 000200	TST43 010652
AMAMS1 = 000000	CIMR = 000040	LASTAD = 017350	SW08 = 000400	TST44 011172
AMAMS2 = 000000	CIRMR = 000020	LF = 000012	SW09 = 001000	TST45 011500
AMAMS3 = 000000	CIRR = 000100	LMD04 = 000200	SW1 = 000002	TST46 012030
AMAMS4 = 000000	CISR = 000160	LMD57 = 000240	SW10 = 002000	TST47 012336
AMSGAD = 000000	CKSWR = 104407	MACR = 000254	SW11 = 004000	TST5 002620
AMSGLG = 000000	CLRCR = 013164	MIMR = 000244	SW12 = 010000	TST50 012546
AMSGTY = 000000	CLRIRR = 013234	MIRR = 000250	SW13 = 020000	TST51 012664
AMTYP1 = 000000	CR = 000015	MISR = 000240	SW14 = 040000	TST6 002766
AMTYP2 = 000000	CRLF = 000200	NEXPAS 002100	SW15 = 100000	TST7 003160
AMTYP3 = 000000	CSIMR = 000050	NEXPA1 002104	SW2 = 000004	TYPDS = 104405
AMTYP4 = 000000	CSIRMR = 000030	NXDEV 013002	SW3 = 000010	TYPE = 104401
APASS = 000000	CSIRR = 000110	NXDEV1 013006	SW4 = 000020	TYPOC = 104402
APRIOR = 000000	CSISR = 000170	PACR = 000300	SW5 = 000040	TYPON = 104404
APTCSU = 000040	DDISP = 177570	PATDAT 013502	SW6 = 000100	TYPOS = 104403
APTENV = 000001	DH1 017154	PIMR = 000260	SW7 = 000200	XXDP 001370
APTSIZ = 000200	DH2 017221	PIRQ = 177772	SW8 = 000400	\$APTHD 001000
APTSP0 = 000100	DIR = 000400	PIRQVE = 000240	SW9 = 001000	\$ATYC 014154
ASWREG = 000000	DISPLA 001142	PRO = 000000	TBITVE = 000014	\$ATY1 014130
ATESTN = 000000	DISPRE 000174	PR1 = 000040	TITLED 016726	\$ATY3 014136
AUNIT = 000000	DMAP 001364	PR2 = 000100	TKVEC = 000060	\$ATY4 014146
AUSWR = 000000	DRCSA 001342	PR3 = 000140	TLCABL 016773	\$AUTOB 001134
AVECT1 = 000000	DRCSB 001346	PR4 = 000200	TPVEC = 000064	\$BASE 001244
AVECT2 = 000000	DRCSC 001352	PR5 = 000240	TRAPVE = 000034	\$BDADR 001122
BEGPAT 013272	DRCSD 001356	PR6 = 000300	TRTVEC = 000014	\$BDDAT 001126
BGCHP2 013334	DRDBA 001344	PR7 = 000340	TSTNUM 001362	\$CDW1 001250
BGCHP3 013374	DRDBB 001350	PS = 177776	TST1 002142	\$CHARC 014124
BGCHP4 013436	DRDBC 001354	PSW = 177776	TST10 003326	\$CKSWR 015656
	DRDBD 001360			

SYMBOL TABLE

\$CMTAG 001100	\$ESCAP 001162	\$MAIL 001170	\$PWRDN 016442	\$TRAP 016646
\$CM3 = 000000	\$ETABL 001210	\$MAMS1 001220	\$PWRMG 016576	\$TRAP2 016670
\$CNTLG 016413	\$ETEND 001252	\$MAMS2 001224	\$PWRUP 016514	\$TRP = 000012
\$CNTLU 016406	\$FATAL 001172	\$MAMS3 001230	\$QUES 001164	\$TRPAD 016702
\$CPUOP 001216	\$FFLG 014374	\$MAMS4 001234	\$RDCHR 016140	\$TSTM 001004
\$CRLF 001165	\$FILLC 001156	\$MBADR 001002	\$RDLIN 016270	\$TSTNM 001102
\$DBLK 015040	\$FILLS 001155	\$MFLG 014372	\$RDSZ = 000010	\$TTYIN 016376
\$DEVCT 001200	\$FREE = 000104	\$MNEW 016431	\$RTNAD 013142	\$TYPDS 014624
\$DEVN 001246	\$GDADR 001120	\$MSGAD 001204	\$SAVR6 016612	\$TYPE 013574
\$DOAGN 013140	\$GDDAT 001124	\$MSGLG 001206	\$SCOPE 015376	\$TYPEC 014006
\$DTBL 015030	\$GET42 013120	\$MSGTY 001170	\$SETUP = 000117	\$TYPEX 014126
\$ENDAD 013130	\$GTSWR 015726	\$MSWR 016420	\$STUP = 177777	\$TYPOC 014422
\$ENDCT 013076	\$HD = 000003	\$MTYP1 001221	\$SVLAD 015604	\$TYPON 014436
\$ENDMG 013147	\$HIBTS 001000	\$MTYP2 001225	\$SVPC = 017350	\$TYPOS 014376
\$ENULL 013144	\$ICNT 001104	\$MTYP3 001231	\$SWR = 165400	\$UNIT 001202
\$ENV 001210	\$ILLUP 016606	\$MTYP4 001235	\$SWREG 001212	\$UNITM 001010
\$ENVM 001211	\$INTAG 001135	\$MXCNT 015654	\$SWRMK = 000000	\$USWR 001214
\$EOP 013042	\$ITEMB 001114	\$NULL 001154	\$TESTN 001174	\$VECT1 001240
\$EOPCT 013070	\$LF 001166	\$NWTST = 000001	\$TIMES 001160	\$VECT2 001242
\$ERFLG 001103	\$LFLG 014373	\$OCNT 014620	\$TKB 001146	\$XOFF = 000023
\$ERMAX 001115	\$LPADR 001106	\$OMODE 014622	\$TKS 001144	\$XON = 000021
\$ERROR 015050	\$LPERR 001110	\$OVER 015640	\$TN = 000052	\$XTSTR 015410
\$ERRPC 001116	\$MADR1 001222	\$PASS 001176	\$TPB 001152	\$GET4 = 000000
\$ERRTB 001252	\$MADR2 001226	\$PASTM 001006	\$TPFLG 001157	\$OFILL 014621
\$ERRTY 015242	\$MADR3 001232	\$PWRAD 016602	\$TPS 001150	.\$X = 001000
\$ERTTL 001112	\$MADR4 001236			

. ABS. 017350 000
 000000 001

ERRORS DETECTED: 0

VIRTUAL MEMORY USED: 48544 WORDS (190 PAGES)
 DYNAMIC MEMORY: 19748 WORDS (75 PAGES)
 ELAPSED TIME: 00:03:02
 CVDRCC, CVDRCC / -SP/CRF=SYSMAC.MLB/ML, CVDRCC.P11

SYMBOL CROSS REFERENCE

CREF V01

SYMBOL	VALUE	REFERENCES								
ABASE	= 164160	#8-424	10-473	10-473	11-521	11-522	11-523	11-524	11-525	11-526
		11-527	11-528	12-553						
ACDW1	= 000000	10-473	10-473							
ACDW2	= 000000	10-473								
ACPUOP	= 000000	10-473	10-473							
ACRLOC	= 001400	#11-540								
ADDW0	= 000000	10-473								
ADDW1	= 000000	10-473								
ADDW10	= 000000	10-473								
ADDW11	= 000000	10-473								
ADDW12	= 000000	10-473								
ADDW13	= 000000	10-473								
ADDW14	= 000000	10-473								
ADDW15	= 000000	10-473								
ADDW2	= 000000	10-473								
ADDW3	= 000000	10-473								
ADDW4	= 000000	10-473								
ADDW5	= 000000	10-473								
ADDW6	= 000000	10-473								
ADDW7	= 000000	10-473								
ADDW8	= 000000	10-473								
ADDW9	= 000000	10-473								
ADEVCT	= 000000	10-473	10-473							
ADEVN	= 000001	#8-425	10-473	10-473						
AENV	= 000000	10-473	10-473							
AENVN	= 000000	10-473	10-473							
AFATAL	= 000000	10-473	10-473							
AMADR1	= 000000	10-473	10-473							
AMADR2	= 000000	10-473	10-473							
AMADR3	= 000000	10-473	10-473							
AMADR4	= 000000	10-473	10-473							
AMAMS1	= 000000	10-473	10-473							
AMAMS2	= 000000	10-473	10-473							
AMAMS3	= 000000	10-473	10-473							
AMAMS4	= 000000	10-473	10-473							
AMSGAD	= 000000	10-473	10-473							
AMSGLG	= 000000	10-473	10-473							
AMSGTY	= 000000	10-473	10-473							
AMTYP1	= 000000	10-473	10-473							
AMTYP2	= 000000	10-473	10-473							
AMTYP3	= 000000	10-473	10-473							
AMTYP4	= 000000	10-473	10-473							
APASS	= 000000	10-473	10-473							
APRIOR	= 000000	10-473								
APTCU	= 000040	17-2165	#17-2166							
APTENV	= 000001	17-2165	17-2166	#17-2166	17-2169					
APTSIZ	= 000200	12-544	#17-2166							
APTSPO	= 000100	17-2165	17-2166	#17-2166						
ASWREG	= 000000	10-473	10-473							
ATESTN	= 000000	10-473	10-473							
AUNIT	= 000000	10-473	10-473							
AUSWR	= 000000	10-473	10-473							

SYMBOL CROSS REFERENCE

CREF V01

SYMBOL	VALUE	REFERENCES
AVECT1	= 000000	10-473 10-473
AVECT2	= 000000	10-473 10-473
BEGPAT	013272	12-796 12-815 12-835 12-855 13-946 13-1001 13-1053 13-1107 13-1324
		13-1349 13-1740 13-1871 13-1928 #16-2061
BGCHP2	013334	#16-2078
BGCHP3	013374	13-1443 13-1543 #16-2095
BGCHP4	013436	13-1414 13-1514 13-1599 #16-2112
BGPAT1	013274	13-1680 #16-2062
BIT0	= 000001	#8-423 12-626 12-648 12-659 12-681 12-703 12-722 12-734 12-756
		12-769 12-788 12-801 12-820 12-840 12-860 13-881 13-916 13-951
		13-955 13-983 13-1006 13-1010 13-1035 13-1058 13-1062 13-1090 13-1112
		13-1116 13-1141 13-1681 13-1741 13-1742 13-1805 13-1806 13-1872 13-1873
		13-1933 13-1934 13-1966 13-1967 13-1988 13-1989 15-2048
BIT00	= 000001	#8-423 8-423
BIT01	= 000002	#8-423 8-423
BIT02	= 000004	#8-423 8-423
BIT03	= 000010	#8-423 8-423
BIT04	= 000020	#8-423 8-423
BIT05	= 000040	#8-423 8-423
BIT06	= 000100	#8-423 8-423
BIT07	= 000200	#8-423 8-423
BIT08	= 000400	#8-423 8-423 17-2179
BIT09	= 001000	#8-423 8-423 17-2169 17-2179
BIT1	= 000002	#8-423 12-681
BIT10	= 002000	#8-423
BIT11	= 004000	#8-423 17-2179
BIT12	= 010000	#8-423
BIT13	= 020000	#8-423 17-2169
BIT14	= 040000	#8-423 17-2179
BIT15	= 100000	#8-423 8-426
BIT2	= 000004	#8-423
BIT3	= 000010	#8-423
BIT4	= 000020	#8-423
BIT5	= 000040	#8-423
BIT6	= 000100	#8-423 12-625 12-640 12-647 12-661 12-668 12-688 12-741 13-958
		13-1072
BIT7	= 000200	#8-423 12-625 12-633 12-640 12-647 12-661 12-668 12-688 12-733
		12-741 12-748 12-755 13-902 13-930 13-958 13-965 13-1065 13-1072
BIT8	= 000400	#8-423 8-427
BIT9	= 001000	#8-423 8-428 12-661
BPTVEC	= 000014	#8-423
CHPISR	= 000140	#8-444
CIMR	= 000040	#8-434 13-1376 13-1395 13-1447 13-1638
CIRMR	= 000020	#8-431 13-1575 13-1684 13-1751 13-1752 13-1812 13-1813 13-1882 13-1883
		13-1938 13-1939
CIRR	= 000100	#8-439 13-1495 13-1547
CISR	= 000160	#8-445
CKSMR	= 104407	17-2169 17-2169 17-2169 17-2176 17-2179 #17-2184
CLRCSR	013164	12-656 12-677 12-698 12-729 12-764 12-795 12-814 12-834 12-854
		13-875 13-910 13-945 13-1000 13-1052 13-1106 13-1157 13-1187 13-1275
		13-1319 13-1344 13-1369 13-1388 13-1408 13-1437 13-1465 13-1486 13-1507
		13-1536 13-1565 13-1592 13-1627 13-1664 13-1673 13-1734 13-1795 13-1857

SYMBOL CROSS REFERENCE

CREF V01

SYMBOL	VALUE	REFERENCES
CLRIAR	013234	13-1865 13-1927 13-1963 13-1985 #15-2030 13-1161 13-1191 13-1232 13-1279 13-1469 13-1490 13-1512 13-1541 13-1569 13-1597 13-1633 13-1669 #15-2045
CR	= 000015	#8-423 17-2165 17-2165
CRLF	= 000200	#8-423 17-2165 17-2165
CSIMR	= 000050	#8-435 13-1416
CSIRMR	= 000030	#8-432 13-1600
CSIRR	= 000110	#8-440 13-1516
CSISR	= 000170	#8-446
DDISP	= 177570	#8-423 10-473 12-544
DM1	017154	11-476 11-482 11-488 11-494 11-500 11-506 11-513 #17-2199
DM2	017221	#17-2200
DIR	= 000400	#8-427 12-625 12-702 12-733 12-768 13-958 13-1013 13-1065 13-1119
DISPLA	001142	#10-473 *12-544 *12-544 17-2169 17-2179
DISPRE	000174	#8-462 12-544
DMAP	001364	#11-534 *12-583 *12-584 12-587 *14-2013 14-2017
DRCSA	001342	#11-521 *12-586 12-590 12-606 12-619 12-657 12-678 12-731 12-798 13-948 13-1005 13-1056 13-1159 13-1188 13-1233 13-1276 13-1322 13-1347 13-1371 13-1390 13-1411 13-1440 13-1467 13-1488 13-1510 13-1539 13-1567 13-1595 13-1667 13-1674 13-1678 13-1682 13-1860 13-1930 13-1964 14-2011 15-2031 15-2046
DRCSE	001346	#11-523 12-699 12-766 12-817 13-950 13-1003 13-1110 13-1160 13-1189 13-1234 13-1277 13-1323 13-1348 13-1372 13-1391 13-1412 13-1441 13-1468 13-1489 13-1511 13-1540 13-1568 13-1596 13-1668 13-1675 13-1861 13-1931 13-1965 15-2032
DRCSC	001352	#11-525 12-620 12-658 12-730 12-837 13-949 13-1055 13-1111 13-1182 13-1226 13-1270 13-1313 13-1339 13-1364 13-1383 13-1403 13-1431 13-1460 13-1480 13-1502 13-1531 13-1560 13-1587 13-1622 13-1665 13-1679 13-1683 13-1728 13-1858 13-1986 15-2033
DRCSD	001356	#11-527 12-700 12-765 12-857 13-1004 13-1057 13-1109 13-1183 13-1227 13-1271 13-1314 13-1340 13-1365 13-1384 13-1404 13-1432 13-1461 13-1481 13-1503 13-1532 13-1561 13-1588 13-1623 13-1666 13-1729 13-1859 13-1932 13-1987 15-2034
DRDBA	001344	#11-522 12-799 13-980 13-1040 13-1079
DRDBB	001350	#11-524 12-818 13-988 13-1032 13-1130
DRDBC	001354	#11-526 12-838 13-972 13-1087 13-1146
DRDBD	001360	#11-528 12-593 12-858 13-1024 13-1095 13-1138
DSWR	= 177570	#8-423 10-473 12-544
DT1	017276	11-477 11-483 11-489 11-495 11-501 11-507 11-514 #17-2203
DT2	017312	#17-2204
EDCHP1	013314	13-1719 #16-2070
EDCHP2	013352	#16-2085
EDCHP3	013414	13-1454 13-1554 #16-2103
EDCHP4	013456	13-1425 13-1525 13-1616 #16-2120
EMTVEC	= 000030	#8-423 *12-544 *12-544
EM1	017022	11-475 #17-2192
EM2	017041	11-481 #17-2193
EM3	017063	11-487 #17-2194
EM4	017076	11-493 #17-2195
EM5	017111	11-499 #17-2196
EM6	017124	11-505 #17-2197
EM7	017137	11-512 #17-2198

SYMBOL CROSS REFERENCE

CREF V01

SYMBOL	VALUE	REFERENCES
ENDDAT	013572	12-810 12-829 12-849 12-869 13-995 13-1047 13-1102 13-1153 13-1335
ENDPAT	013500	13-1360 13-1790 13-1921 13-1958 *16-2160
ERROR	* 104000	*16-2129 *8-423 12-615 12-631 12-638 12-646 12-653 12-667 12-674 12-687
		12-694 12-707 12-713 12-720 12-726 12-739 12-746 12-754 12-761
		12-773 12-779 12-786 12-792 12-808 12-827 12-847 12-867 13-888
		13-894 13-900 13-907 13-928 13-935 13-941 13-964 13-971 13-977
		13-987 13-993 13-1017 13-1023 13-1029 13-1039 13-1045 13-1071 13-1078
		13-1084 13-1094 13-1100 13-1123 13-1129 13-1135 13-1145 13-1151 13-1168
		13-1174 13-1179 13-1200 13-1206 13-1212 13-1217 13-1223 13-1244 13-1250
		13-1255 13-1260 13-1267 13-1287 13-1293 13-1299 13-1304 13-1310 13-1333
		13-1358 13-1380 13-1400 13-1423 13-1452 13-1477 13-1499 13-1523 13-1552
		13-1579 13-1584 13-1609 13-1614 13-1644 13-1650 13-1656 13-1661 13-1694
		13-1702 13-1710 13-1717 13-1762 13-1770 13-1776 13-1782 13-1788 13-1821
		13-1827 13-1842 13-1854 13-1893 13-1901 13-1907 13-1913 13-1919 13-1947
		13-1956 13-1975 13-1981 13-1997 13-2003
ERRVEC	* 000004	*8-423 12-544 *12-544 *12-544 *12-605 12-616 *12-616 17-2179 *17-2179
FALCON	017330	*17-2179 12-548 *18-2222
GNS	* *****	8-462 8-462 17-2184 17-2184 17-2184 17-2184 17-2184 17-2184 17-2184 17-2184
		17-2184 17-2184 17-2184
GTSWR	* 104406	12-581 *17-2184
MT	* 000011	*8-423 17-2165 17-2165
IE	* 001000	*8-428 12-662 13-1966
IMRLOC	001372	*11-537 *13-1685 13-1704 *13-1722
INTFLG	001366	*11-535
IOTVEC	* 000020	*8-423 *12-544 *12-544
IRRLOC	001376	*11-539 *13-1686 13-1689 *13-1721
ISRLOC	001374	*11-538
KXTCHK	000400	18-2224 *18-2231
KXTFLA	017344	13-1837 13-1849 *18-2226 *18-2231 *18-2239
LASTAD	* 017350	*19-2284
LF	* 000012	*8-423 17-2165 17-2165
LMD04	* 000200	*8-448
LMD57	* 000240	*8-449
MACR	* 000254	*8-455 13-1175 13-1194 13-1219 13-1245 13-1300 13-1326 13-1639
MIMR	* 000244	*8-453 13-1164 13-1201 13-1238 13-1262 13-1288 13-1351 13-1375 13-1394
		13-1415 13-1444 13-1580 13-1610 13-1645 13-1713
MIRR	* 000250	*8-454 13-1170 13-1213 13-1256 13-1282 13-1306 13-1472 13-1493 13-1515
		13-1544 13-1572 13-1602 13-1657 13-1698 13-1747 13-1748 13-1832 13-1845
		13-1878 13-1879 13-1942 13-1949
MISR	* 000240	*8-452 13-1207 13-1251 13-1294 13-1651
NEXPAS	002100	12-588 *12-590 14-2019
NEXPA1	002104	*12-591 12-594
NXDEV	013002	*14-2011
NXDEV1	013006	12-589 *14-2012 14-2018
PACR	* 000300	*8-458 13-1195 13-1327 13-1635
PATDAT	013502	*16-2132
PIMR	* 000260	*8-459 13-1239 13-1352
PIRQ	* 177772	*8-423
PIRQVE	* 000240	*8-423

16

SYMBOL CROSS REFERENCE

CREF V01

SYMBOL	VALUE	REFERENCES
PRO	= 000000	08-423
PR1	= 000040	08-423
PR2	= 000100	08-423
PR3	= 000140	08-423
PR4	= 000200	08-423
PR5	= 000240	08-423
PR6	= 000300	08-423
PR7	= 000340	08-423 12-598
PS	= 177776	08-423 8-423
PSW	= 177776	08-423
PVMA	= 000340	08-460
PWRMSG	016614	17-2181 017-2182
PWRVEC	= 000024	08-423 *12-544 *12-544 *17-2181 *17-2181 *17-2181 *17-2181 *17-2181 *17-2181
RDCHR	= 104410	17-2180 017-2184
RDLIN	= 104411	017-2184
RDY	= 100000	08-426 12-625 12-640 12-688 12-702 12-715 12-733 12-748 12-768
		12-781 13-958 13-1013 13-1065 13-1119
RESVEC	= 000010	08-423
R6	= 000006	08-423 *12-544 *12-544 12-544
R7	= 000007	08-423
SCOPE	= 000004	08-423 12-602 12-618 12-655 12-676 12-697 12-728 12-763 12-794
		12-813 12-833 12-853 13-874 13-909 13-944 13-999 13-1051 13-1105
		13-1156 13-1186 13-1230 13-1274 13-1318 13-1343 13-1368 13-1387 13-1407
		13-1436 13-1464 13-1485 13-1506 13-1535 13-1564 13-1591 13-1626 13-1672
		13-1733 13-1794 13-1864 13-1926 13-1962 13-1984
SIMR	= 000060	08-436 13-1396 13-1418 13-1574 13-1604
SIRR	= 000120	08-441 13-1473 13-1494 13-1518 13-1573 13-1603 13-1637
SSIMR	= 000070	08-437 13-1445 13-1685
SSIRR	= 000130	08-442 13-1545 13-1686
STACK	= 001100	08-423 12-595
START	001402	8-462 012-544
START1	002034	12-580 012-582 14-2020 17-2181
STKLMT	= 177774	08-423
SWR	001140	010-473 12-544 *12-544 12-544 *12-544 *12-544 17-2169 17-2169 17-2169
		17-2179 17-2179 17-2179 17-2179 17-2179 17-2180 17-2180 17-2181 17-2181
SWRCK	015224	17-2169 017-2174
SWREG	000176	08-462 12-544 17-2180 17-2180
SWO	= 000001	08-423
SW00	= 000001	08-423 8-423
SW01	= 000002	08-423 8-423
SW02	= 000004	08-423 8-423
SW03	= 000010	08-423 8-423
SW04	= 000020	08-423 8-423
SW05	= 000040	08-423 8-423
SW06	= 000100	08-423 8-423
SW07	= 000200	08-423 8-423
SW08	= 000400	08-423 8-423
SW09	= 001000	08-423 8-423
SW1	= 000002	08-423
SW10	= 002000	08-423
SW11	= 004000	08-423
SW12	= 010000	08-423

SYMBOL CROSS REFERENCE

CREF V01

SYMBOL	VALUE	REFERENCES
SW13	= 020000	#8-423
SW14	= 040000	#8-423
SW15	= 100000	#8-423
SW2	= 000004	#8-423
SW3	= 000010	#8-423
SW4	= 000020	#8-423
SW5	= 000040	#8-423
SW6	= 000100	#8-423
SW7	= 000200	#8-423
SW8	= 000400	#8-423
SW9	= 001000	#8-423
TBITVE	= 000014	#8-423
TITLED	016726	12-575 #17-2189
TKVEC	= 000060	#8-423
TLCABL	016773	12-576 #17-2190
TPVEC	= 000064	#8-423
TRAPVE	= 000034	#8-423 *12-544 *12-544
TRTVEC	= 000014	#8-423
TSTNUM	001362	#11-533 *17-2174 17-2203 17-2204
TST1	002142	#12-602
TST10	003326	12-791 #12-794
TST11	003422	#12-813
TST12	003516	#12-833
TST13	003612	#12-853
TST14	003706	#13-874
TST15	004056	13-906 #13-909
TST16	004224	13-940 #13-944
TST17	004524	#13-999
TST2	002220	#12-618
TST20	005012	#13-1051
TST21	005312	#13-1105
TST22	005600	#13-1156
TST23	005734	13-1181 #13-1186
TST24	006144	13-1225 #13-1230
TST25	006350	13-1269 #13-1274
TST26	006554	13-1312 #13-1318
TST27	006672	13-1338 #13-1343
TST3	002412	12-652 #12-655
TST30	007010	13-1363 #13-1368
TST31	007102	13-1382 #13-1387
TST32	007202	13-1402 #13-1407
TST33	007330	13-1430 #13-1436
TST34	007456	13-1459 #13-1464
TST35	007556	13-1479 #13-1485
TST36	007660	13-1501 #13-1506
TST37	010012	13-1530 #13-1535
TST4	002522	12-673 #12-676
TST40	010144	13-1559 #13-1564
TST41	010270	13-1586 #13-1591
TST42	010444	13-1621 #13-1626
TST43	010652	13-1663 #13-1672
TST44	011172	13-1727 #13-1733

SYMBOL CROSS REFERENCE

CREF V01

SYMBOL	VALUE	REFERENCES
\$ENDMG	013147	14-2020 #14-2020
\$ENULL	013144	14-2020 #14-2020
\$ENV	001210	#10-473 12-564 17-2165 17-2166 17-2166 17-2169
\$ENVM	001211	#10-473 12-544 17-2165 17-2165 17-2166
\$EOP	013042	14-2014 #14-2020
\$EOPCT	013070	#14-2020 14-2020
\$ERFLG	001103	#10-473 *17-2169 17-2169 17-2169 17-2179 17-2179 17-2179 *17-2179 17-2179
\$ERMAX	001115	#10-473 *12-544 17-2179 *17-2179 17-2179 17-2179
\$ERROR	015050	12-544 #17-2169
\$ERRPC	001116	#10-473 *17-2169 *17-2169 17-2169 17-2169 17-2169 17-2178 17-2203 17-2204
\$ERRTB	001252	#11-473 17-2178
\$ERRTY	015242	17-2175 #17-2178
\$ERTTL	001112	#10-473 *17-2169 17-2169 17-2169
\$ESCAP	001162	#10-473 *12-544 17-2169 17-2169 17-2169 17-2169 *17-2179
\$ETABL	001210	#10-473
\$ETEND	001252	9-472 #10-473
\$FATAL	001172	#10-473 *12-545 *17-2166 17-2166 *17-2166 #17-2166
\$FFLG	014374	*17-2166 *17-2166 17-2166 *17-2166 #17-2166
\$FILLC	001156	#10-473 17-2165 17-2165 17-2165
\$FILLS	001155	#10-473 17-2165 17-2165
\$FREE	000104	#19-2278 19-2279
\$GDADR	001120	#10-473 17-2204
\$GDDAT	001124	#10-473 *12-603 *12-625 12-629 *12-633 12-636 *12-640 12-644 *12-647
		12-651 *12-661 12-665 *12-668 12-672 *12-680 12-685 *12-688 12-692
		*12-702 12-705 *12-709 12-711 *12-715 12-718 *12-721 12-724 *12-733
		12-737 *12-741 12-744 *12-748 12-752 *12-755 12-759 *12-768 12-771
		*12-775 12-777 *12-781 12-784 *12-787 12-790 *12-802 12-806 *12-821
		12-825 *12-841 12-845 *12-861 12-865 *13-883 13-886 *13-890 13-892
		*13-896 13-898 *13-902 13-905 *13-918 13-921 *13-924 13-926 *13-930
		13-933 *13-937 13-939 *13-958 13-962 *13-965 13-969 *13-973 13-975
		*13-978 13-981 13-985 *13-989 13-991 *13-1013 13-1015 *13-1018 13-1021
		*13-1025 13-1027 *13-1030 13-1033 13-1037 *13-1041 13-1043 *13-1065 13-1069
		*13-1072 13-1076 *13-1080 13-1082 *13-1085 13-1088 13-1092 *13-1096 13-1098
		*13-1119 13-1121 *13-1124 13-1127 *13-1131 13-1133 *13-1136 13-1139 13-1143
		*13-1147 13-1149 *13-1163 13-1166 *13-1169 13-1172 *13-1177 13-1193 13-1196
		13-1198 *13-1202 13-1204 *13-1208 13-1210 13-1215 13-1221 *13-1237 13-1240
		13-1242 *13-1246 13-1248 13-1253 13-1258 *13-1263 13-1265 *13-1281 13-1285
		*13-1289 13-1291 *13-1295 13-1297 13-1302 13-1308 *13-1328 13-1331 *13-1353
		13-1356 *13-1374 13-1378 *13-1393 13-1398 *13-1417 13-1421 *13-1446 13-1450
		*13-1471 13-1475 *13-1492 13-1497 *13-1517 13-1521 *13-1546 13-1550 *13-1571
		13-1577 13-1582 *13-1601 13-1607 13-1612 *13-1640 13-1642 *13-1646 13-1648
		*13-1652 13-1654 13-1659 *13-1688 13-1692 *13-1695 *13-1697 13-1700 *13-1705
		13-1708 *13-1712 13-1715 *13-1758 13-1760 *13-1764 *13-1765 *13-1766 13-1768
		*13-1772 13-1774 *13-1778 13-1780 *13-1784 13-1786 *13-1815 13-1819 13-1825
		*13-1833 13-1835 *13-1839 13-1840 *13-1843 13-1847 *13-1851 13-1852 *13-1889
		13-1891 *13-1895 *13-1896 *13-1897 13-1899 *13-1903 13-1905 *13-1909 13-1911
		*13-1915 13-1917 *13-1943 13-1945 *13-1950 *13-1951 *13-1952 13-1954 *13-1969
		13-1973 *13-1976 13-1979 *13-1991 13-1995 *13-1999 13-2001 *15-2035 17-2203
		17-2204
\$GET42	013120	#14-2020
\$GTSWR	015726	#17-2180 17-2184 17-2184

B7

SYMBOL CROSS REFERENCE

CREF V01

SYMBOL	VALUE	REFERENCES
\$MD	* 000003	8-414 8-414 8-414
\$MIBTS	001000	*9-472
\$ICNT	001104	*10-473 *17-2179 17-2179 *17-2179 17-2179 17-2179
\$ILLUP	016606	17-2181 17-2181 *17-2181
\$INTAG	001135	*10-473 17-2180 17-2180 17-2180
\$ITEMB	001114	*10-473 *17-2169 17-2169 17-2169 17-2169 17-2169 17-2178
\$LF	001166	*10-473 17-2165 17-2165 17-2169 17-2169 17-2180 17-2180
\$LFLG	014373	*17-2166 *17-2166
\$LPADR	001106	*10-473 *12-544 *17-2179 *17-2179 17-2179 17-2179 17-2179
\$LPERR	001110	*10-473 *12-544 *12-797 *12-816 *12-836 *12-856 *13-947 *13-1002 *13-1054 *13-1108 *13-1321 *13-1346 *13-1410 *13-1439 *13-1509 *13-1538 *13-1594 *13-1739 *13-1870 *13-1929 17-2169 17-2179 *17-2179 17-2179 17-2179
\$MADR1	001222	*10-473
\$MADR2	001226	*10-473
\$MADR3	001232	*10-473
\$MADR4	001236	*10-473
\$MAIL	001170	9-472 9-472 *10-473 12-544 17-2165 17-2169 17-2179
\$MAMS1	001220	*10-473
\$MAMS2	001224	*10-473
\$MAMS3	001230	*10-473
\$MAMS4	001234	*10-473
\$MBADR	001002	*9-472
\$MFLG	014372	*17-2166 17-2166 *17-2166 *17-2166
\$MNEW	016431	17-2180 *17-2180
\$MSGAD	001204	*10-473 *17-2166 17-2166
\$MSGLG	001206	*10-473 *17-2166
\$MSGTY	001170	*10-473 *12-546 17-2166 *17-2166 17-2166 *17-2166
\$MSWR	016420	17-2180 *17-2180
\$MTYP1	001221	*10-473
\$MTYP2	001225	*10-473
\$MTYP3	001231	*10-473
\$MTYP4	001235	*10-473
\$MXCNT	015654	17-2179 17-2179 17-2179 *17-2179
\$NULL	001154	*10-473 17-2165 17-2165 17-2165
\$NWTST	* 000001	*12-602 12-602 *12-602 *12-618 12-618 *12-618 *12-655 12-655 *12-655 *12-676 12-676 *12-676 *12-697 12-697 *12-697 *12-728 12-728 *12-728 *12-763 12-763 *12-763 *12-794 12-794 *12-794 *12-813 12-813 *12-813 *12-833 12-833 *12-833 *12-853 12-853 *12-853 *13-874 13-874 *13-874 *13-909 13-909 *13-909 *13-944 13-944 *13-944 *13-999 13-999 *13-999 *13-1051 13-1051 *13-1051 *13-1105 13-1105 *13-1105 *13-1156 13-1156 *13-1156 *13-1186 13-1186 *13-1186 *13-1230 13-1230 *13-1230 *13-1274 13-1274 *13-1274 *13-1318 13-1318 *13-1318 *13-1343 13-1343 *13-1343 *13-1368 13-1368 *13-1368 *13-1387 13-1387 *13-1387 *13-1407 13-1407 *13-1407 *13-1436 13-1436 *13-1436 *13-1464 13-1464 *13-1464 *13-1485 13-1485 *13-1485 *13-1506 13-1506 *13-1506 *13-1535 13-1535 *13-1535 *13-1564 13-1564 *13-1564 *13-1591 13-1591 *13-1591 *13-1626 13-1626 *13-1626 *13-1672 13-1672 *13-1672 *13-1733 13-1733 *13-1733 *13-1794 13-1794 *13-1794 *13-1864 13-1864 *13-1864 *13-1926 13-1926 *13-1926 *13-1962 13-1962 *13-1962 *13-1984 13-1984 *13-1984
\$OCNT	014620	*17-2167 *17-2167 *17-2167
\$OMODE	014622	*17-2167 *17-2167 17-2167 *17-2167 *17-2167 *17-2167
\$OVER	015640	17-2179 17-2179 17-2179 *17-2179 *17-2179
\$PASS	001176	*10-473 *12-544 *14-2020 *14-2020 14-2020 14-2020 14-2020 17-2179 17-2179

SYMBOL CROSS REFERENCE

CREF V01

SYMBOL	VALUE	REFERENCES
		13-906 13-909 13-909 @13-909 13-940 13-944 13-944 @13-944 13-999
		13-999 @13-999 13-1051 13-1051 @13-1051 13-1105 13-1105 @13-1105 13-1156
		13-1156 @13-1156 13-1181 13-1186 13-1186 @13-1186 13-1225 13-1230 13-1230
		@13-1230 13-1269 13-1274 13-1274 @13-1274 13-1312 13-1318 @13-1318 13-1318
		13-1338 13-1343 13-1343 @13-1343 13-1363 13-1368 @13-1368 13-1368
		13-1387 13-1387 @13-1387 13-1402 13-1407 @13-1407 13-1430 13-1436
		13-1436 @13-1436 13-1459 13-1464 13-1464 @13-1464 13-1479 13-1485
		@13-1485 13-1501 13-1506 13-1506 @13-1506 13-1530 13-1535 @13-1535 13-1535
		13-1559 13-1564 13-1564 @13-1564 13-1586 13-1591 13-1591 @13-1591 13-1621
		13-1626 13-1626 @13-1626 13-1663 13-1672 13-1672 @13-1672 13-1727 13-1733
		13-1733 @13-1733 13-1794 13-1794 @13-1794 13-1856 13-1864 13-1864 @13-1864 13-1864
		13-1926 13-1926 @13-1926 13-1962 13-1962 @13-1962 13-1980 13-1984 13-1984
		@13-1984
\$TPB	001152	@10-473 17-2165 17-2165 17-2165
\$TPFLG	001157	@10-473 17-2165 17-2165 17-2165
\$TPS	001150	@10-473 17-2165 17-2165 17-2165
\$TRAP	016646	12-544 @17-2184
\$TRAP2	016670	@17-2184 17-2184
\$TRP	= 000012	@17-2184 17-2184 17-2184 17-2184 17-2184 @17-2184 17-2184 17-2184
		17-2184 @17-2184 17-2184 17-2184 17-2184 @17-2184 17-2184 17-2184
		17-2184 17-2184 @17-2184 17-2184 17-2184 17-2184 @17-2184 17-2184
		17-2184 17-2184 17-2184 @17-2184 17-2184 17-2184 17-2184 @17-2184
		17-2184 17-2184 17-2184 17-2184 @17-2184 17-2184 17-2184 17-2184
		@17-2184 17-2184
\$TRPAD	016702	17-2184 @17-2184
\$TSTM	001004	@9-472
\$TSTNM	001102	@10-473 *14-2020 17-2169 17-2169 17-2169 17-2174 17-2179 17-2179 *17-2179
		17-2179 17-2179 17-2179 17-2179
\$TTYIN	016376	17-2180 17-2180 17-2180 @17-2180
\$TYPBN	= *****	17-2184
\$TYPDS	014624	@17-2168 17-2184 17-2184
\$TYPE	013574	@17-2165 17-2166 17-2184 17-2184
\$YPEC	014006	17-2165 17-2165 17-2165 @17-2165 17-2180
\$TYPEX	014126	17-2165 17-2165 @17-2165
\$TYPOC	014422	@17-2167 17-2184 17-2184
\$TYPON	014436	17-2167 @17-2167 17-2184
\$TYPOS	014376	@17-2167 17-2184
\$UNIT	001202	@10-473 *12-582 12-597 *14-2016
\$UNITM	001010	@9-472
\$USMR	001214	@10-473
\$VECT1	001240	@10-473
\$VECT2	001242	@10-473
\$XOFF	= 000023	17-2165 17-2165
\$XON	= 000021	17-2165 17-2165 17-2180
\$XTSTR	015410	@17-2179
\$GET4	= 000000	@14-2020 14-2020
\$OFILL	014621	*17-2167 *17-2167 17-2167 @17-2167
\$OCAT	= *****	17-2169 17-2179
.\$ASTA	= *****	17-2166 17-2166
.\$X	= 001000	@9-472 9-472

