

VSV11

VSV11 CSS DIAG
CVVSABO

AH-E769B-MC
FICHE 1 OF 2

APR 1982
COPYRIGHT © 81-82
MADE IN USA



The main body of the document is a large grid of approximately 15 columns and 25 rows of small, dense text. Each cell in the grid contains technical data, likely related to a diagnostic or maintenance procedure. The text is too small to be legible in this image, but it appears to be organized into a structured table format.

VSV11

VSV11 CSS DIAG
CVVSABO

AH-E769B-MC
FICHE 2 OF 2

APR 1982
COPYRIGHT © 81-82
MADE IN USA



R

ITEM	DESCRIPTION	QTY	UNIT
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%

5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58

IDENTIFICATION

PRODUCT CODE: AC-E768B-MC
PRODUCT NAME: CVVSABO VSV11/VS11 DIAGNOSTIC
MAINTAINER: CSS PERIPHERALS & GRAPHICS GROUP
AUTHORS: DON MACOMBER
 GUS PASQUANTONIO
 BILL WEISKE
DATE: 9-JUL-1981
REVISION HISTORY:
 BILL WEISKE 27-OCT-81
 TEST 14 - BYPASS IF LSI AND .GT. 124K OF MEMORY.
 TEST 17 - REPLACE CLRMEM WITH SETMEM = 0.
 TEST 34 - INCREASE DELAY (WAS PROBLEM @50HZ)

COPYRIGHT (C) 1982
DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED UNDER A LICENCE FOR USE ONLY ON A SINGLE COMPUTER SYSTEM AND MAY BE COPIED ONLY WITH THE INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE, OR ANY OTHER COPIES THEREOF, MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE ITEMS. TITLE TO AND OWNERSHIP OF THE SOFTWARE SHALL AT ALL TIMES REMAIN IN DEC.

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

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

60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116

1.0 PROGRAM ABSTRACT

 THE VSV11 DIAGNOSTIC PROGRAM PROVIDES A SERIES OF TESTS
 DESIGNED TO VERIFY THE INTEGRITY AND OPERABILITY OF THE
 VSV11/VS11 Q-BUS VIDEO IMAGE PROCESSING SYSTEM.
 SOME OF THE TEST SEQUENCES ARE VISUAL ONLY. HOWEVER ALL
 TESTS WILL RUN WITH OR WITHOUT A DISPLAY MONITOR
 CONNECTED TO THE SYSTEM. THE TESTS DESIGNATED
 'DPU-ONLY' WILL RUN WITHOUT THE 'DBUS' CONNECTED
 TO THE DISPLAY PROCESSOR MODULE (M7064).
 THE PROGRAM WILL SUPPORT UP TO 16 VSV11 SYSTEMS.
 HOWEVER, MULTI-UNIT CONFIGURATIONS ARE TESTED ONE AT
 A TIME BY THE PROGRAM.
 ANY LOGIC ERRORS ENCOUNTERED ARE REPORTED ON THE SYSTEM
 CONSOLE DEVICE.

1.1 LIST OF HARDWARE TESTS

 DPU ONLY TESTS:

TEST 1	STATIC RESET
TEST 2	RESET ONES
TEST 3	SOFT INIT ONES
TEST 4	REGISTERS UNIQUE ADDRESS
TEST 5	INCREMENTING REGISTERS
TEST 6	DPU START-STOP
TEST 7	DPU OPCODES
TEST 8	INCREMENTING HISTOGRAM BASE ADDRESS
TEST 9	INCREMENTING CHARACTER BASE ADDRESS
TEST 10	DJMS/DPOP
TEST 11	MAIN MEM MGT ACCESS (OVER 28K)
TEST 12	AUX MEM MGT ACCESS (OVER 28K)
TEST 13	STOP INTERRUPT
TEST 14	DPU TIME-OUT INTERRUPT
TEST 15	ERROR CODES
TEST 16	ADDRESS RELOCATE
TEST 17	CHARACTERS
TEST 18	ABSOLUTE POINTS
TEST 19	LONG VECTORS
TEST 20	RELATIVE POINTS
TEST 21	SHORT VECTORS
TEST 22	RUN-LENGTH

IMAGE MEMORY & SYNC GENERATOR TESTS & DISPLAYS:

TEST 23	CURSOR REGISTERS/SWITCH/MATCH
TEST 24	GRAPH-HISTOGRAM X
TEST 25	GRAPH-HISTOGRAM Y
TEST 26	BIT MAP (1)
TEST 27	BIT MAP (0)
TEST 28	IMAGE MEMORY CLEAR-SET
TEST 29	IMAGE MEMORY INTERLACE
TEST 30	IMAGE MEMORY PATTERNS

117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172

TEST 31 SPARE
TEST 32 SYSTEM VERIFICATION DISPLAY

STAND ALONE TESTS & ROUTINES:

TEST 33 JOY-STICK VERIFICATION (STAND ALONE)
TEST 34 SELECTED DISPLAYS (STAND ALONE)
TEST 35 SYSTEM CONFIGURATION (STAND ALONE)

2.0 STRUCTURE OF PROGRAM

THIS PROGRAM IS STRUCTURED TO RUN UNDER THE DIAGNOSTIC SUPERVISOR (REV D) AND THE XXDP+ MONITOR. A BRIEF OVERVIEW OF OPERATING INSTRUCTIONS IS PROVIDED IN SECTION 3.1 OF THIS DOCUMENT. REFER TO CHAPTER 5 OF THE XXDP+ USERS MANUAL FOR FURTHER DETAILS.

2.1 HARDWARE REQUIREMENTS

PDP-11/LSI-11 PROCESSOR WITH 28K OR MORE OF MEMORY
CONSOLE DEVICE (LA30, LA36, VT50, VT100, ETC.)
XXDP+ LOAD DEVICE (RX, RP, RL, TM, DT, ETC.)
1 TO 8 VSV11 SYSTEMS, EACH CONSISTING OF:
M7064 DISPLAY PROCESSOR
M7062 IMAGE MEMORY (1 TO 4 CHANNELS)
M7061 SYNC GENERATOR WITH CURSOR CONTROL
DISPLAY MONITOR, COLOR OR MONOCHROME (OPTIONAL).
DW11 UNIBUS TO LSI-11 BUS CONVERTER (OPTIONAL, VS11 SYSTEMS ONLY).

2.2 RELATED DOCUMENTS AND STANDARDS

XXDP+ USERS MANUAL (CHQUSA)
VSV11 OPTION DESCRIPTION (YM-C183C)
VSV11 DIAGNOSTIC LISTING (SEC 7 OF THIS DOCUMENT).

3.0 LOADING AND STARTING PROCEDURES

THIS PROGRAM IS LOADED AND STARTED FROM ANY XXDP+ MEDIA USING THE STANDARD XXDP+ OPERATING PROCEDURES.

AT START UP, THE SUPERVISOR WILL IDENTIFY ITSELF AND THE NAME OF THIS PROGRAM ON THE CONSOLE DEVICE, AND THEN DISPLAY A COMMAND MODE PROMPT (DR>) WHICH INDICATES READY TO ACCEPT ANY OF THE FOLLOWING COMMANDS.

174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230

3.1 -----
SUPERVISOR COMMAND SUMMARY

THIS SECTION PRESENTS A BRIEF OVERVIEW OF THE COMMANDS NECESSARY TO CONTROL THE OPERATION OF THIS DIAGNOSTIC UNDER THE REV D DIAGNOSTIC SUPERVISOR.

STA(RT)	INITIAL START-UP -- BUILD P-TABLES.
RES(TART)	RESTART -- USE EXISTING P-TABLES.
CON(TINUE)	CONTINUE AFTER <^C> OR ERROR HALT.
PRO(CCEED)	CONTINUE AFTER ERROR HALT.
EXI(T)	RETURN TO XXDP+ MONITOR.

THE FOLLOWING SWITCHES APPLY TO THE ABOVE:

/TESTS: <TEST NUMBERS TO RUN>
/PASS: <NUMBER OF PASSES TO RUN>
/FLAGS: <SEE FLAG LIST BELOW>
/EOP: <NUMBER OF PASSES 'TIL END-OF-PASS>

ADDITIONAL COMMANDS AVAILABLE ARE:

DRO(P)/UNIT:N	REMOVE UNIT N FROM TEST LIST
ADD/UNIT:N	ADD UNIT N (PREVIOUSLY DROPPED).
DIS(PLAY)/UNIT:N	PRINT UNITS P-TABLE AND STATUS.
PRI(NT)	PRINT STATISTICS (PER-UNIT STATUS).
ZFL(AGS)	CLEAR ALL FLAGS.
FLA(GS)	PRINT CURRENT FLAG SETTINGS.

THE GENERALIZED COMMAND STRING FORMAT IS:

COM(MAND)/SWITCH:VALUE/SWITCH:VALUE ... <CR>

3.2 -----
RUN TIME OPTION FLAGS

THE FOLLOWING FLAGS ARE USED IN LIEU OF THE OLD HARDWARE SWITCH REGISTER TO FURTHER DEFINE PROGRAM BEHAVIOUR:

HOE	HALT ON ERROR
LOE	LOOP ON ERROR
IER	INHIBIT ALL ERROR REPORTS
IBE	INHIBIT BASIC ERROR REPORTS
IXR	INHIBIT EXTENDED ERROR REPORTS
PRI	SEND ALL REPORTS TO LINE PRINTER.
PNT	PRINT TEST NUMBERS (AND I.D'S) AS EXECUTED
BOE	GOOD OLD 'BELL-ON-ERROR'
UAM	RUN IN 'UNATTENDED' MODE (NO MANUAL)
ISR	INHIBIT STATISTICS (PER-UNIT STATUS EACH PASS)
IDU	INHIBIT 'AUTO-DROP' (EXCEPT FOR NON-EXISTENT REG.)
ADR	EXECUTE 'AUTO-DROP' (USER SUPPLIED CODE)
LOT	LOOP ON TEST
EVL	EVALUATE ERRORS (NOT IMPLEMENTED)

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
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285

FLAG SETTINGS ARE ALTERED BY USING THE /FLAGS: SWITCH
IN ANY COMMAND STRING. I.E. STA/FLAG:IER:BOE <CR>
WILL START THE PROGRAM, RUN ALL TEST IN ORDER, INHIBIT
ERROR TYPEOUT, RING BELL ON ERROR.

3.3 -----
INITIAL START-UP -- BUILD P-TABLES

WHENEVER THE PROGRAM IS STARTED, VIA THE STA(RT) COMMAND,
THE SUPERVISOR REQUESTS THE FOLLOWING P-TABLES PARAMETER
CHANGES:

- CHANGE HW (L) ?
- #UNITS (D) ?
- UNIT 0
- DEVICE ADDRESS (D) 172010 ?
- 1ST INTERRUPT VECTOR (D) 320 ?
- INTERRUPT PRIORITY (D) 4 ?
- LUT INSTALLED (L) N ?
- FREQUENCY = 50HZ (L) N ?

IN ADDITION, ON A START, RESTART OR CONTINUE THE SUPERVISOR
REQUESTS CHANGES TO THE SOFTWARE OPERATING PARAMETERS, AS
FOLLOWS:

- CHANGE SW?
- RUN DPU TESTS ONLY (L) N ?
- LONG MEMORY TESTS (L) N ?
- INHIBIT ITERATIONS (L) N ?
- MANUFACTURING MODE (L) N ?
- PER TEST ERROR LIMIT (D) 25 ?
- PER UNIT ERROR LIMIT (D) 200 ?

IF 'RUN DPU TESTS ONLY?' IS ANSWERED 'YES', NO COMMUNICATION
WITH THE IMAGE MEMORIES OR SYNC CHANNELS IS MADE; THE PROGRAM
CAN BE RUN WITH THE 'DBUS' CABLE DISCONNECTED. IF 'LONG
MEMORY TESTS' IS SELECTED, 15 DIFFERENT PIXEL DATA VALUES ARE
USED DURING TEST 28 (IMAGE MEMORY CLEAR-SET) RATHER THAN THE
STANDARD 5 VALUES. INHIBITING ITERATIONS CAUSES THE PROGRAM
TO RUN FASTER, EQUIVALENT TO THE FIRST PASS AFTER STARTUP.
IF 'MANUFACTURING MODE' IS SELECTED, PREDEFINED PARAMETERS ARE
USED FOR THE IMAGE MEMORY AND SYNC CHANNEL CONFIGURATION DATA.
THE ERROR LIMITS DEFINE THE NUMBER OF ERRORS ALLOWED TO OCCUR
BEFORE A UNIT IS DROPPED (UNLESS THE /FLAG:IDU FLAG IS SET TO
INHIBIT DROPPING OF UNITS). FOR EXAMPLE, IF 25 ERRORS OCCUR
IN TEST 3 WHILE TESTING UNIT 2, UNIT 2 WILL BE DROPPED. OR,
IF UNIT 2 ACCUMULATES 200 ERRORS OVER THE COURSE OF SEVERAL TESTS
OR PASSES, IT WILL BE DROPPED.

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

4.0 -----
ERROR REPORTING

LOGIC ERRORS ARE REPORTED BY THE DIAGNOSTIC SUPERVISOR AS THEY OCCUR, HOWEVER, IT MUST BE NOTED THAT CAREFUL OBSERVATION OF THE VARIOUS VISUALS FRAMES IS ALSO REQUIRED TO VERIFY TOTAL SYSTEM PERFORMANCE.

EACH ERROR SIGNATURE CONSISTS OF ONE OR MORE LINES OF TEXT, DESCRIBING THE ERROR, AND INCLUDES EXPECTED VS RECEIVED DATA WHERE APPLICABLE. THE FIRST LINE IS THE SUPERVISORS HEADER (BASIC), WHICH INCLUDES PROGRAM NAME, ERROR TYPE (HRD, SFT, DEV FATAL, OR SYS FATAL), ERROR NUMBER, TEST AND SUB-TEST NUMBERS, AND PC. THE HEADER WILL NORMALLY BE FOLLOWED BY ONE OR MORE (EXTENDED) LINES FURTHER IDENTIFYING THE ERROR.

EXAMPLES:

CVVSA DEV FTL ERR 00002 ON UNIT 00 TST 000 SUB 001 PC:-----
'BUS-INIT' DIDN'T INITIALIZE DPU
UNIT 0 DROPPED
DR>

CVVSA HRD ERR 00801 ON UNIT 00 TST 008 SUB 001 PC:-----
LONG VECTOR FAILURE
DXR,DYR EXP'D: 1776,0000
DXR,DYR REC'D: 0000,0000
ORIGIN: 0000,0000 DX,DY:041776,000000

NOTE THAT THE ERROR NUMBER IS IN THE FORMAT 'TTTEE' WHERE EE IS THE E'TH ERROR CALL WITHIN TEST TTT. WHEN ERRORS OCCUR OUTSIDE THE CONFINES OF A GIVEN TEST (INITIALIZE, OR SOME GLOBAL SUBROUTINE), TTT = 0.

I.E. ERR 00002 = 2ND ERROR IN PROGRAM INITIALIZATION.
ERR 00801 = 1ST ERROR IN TEST 8.

4.1 -----
ERROR HALTS

ERROR HALTS ARE CONDITIONED BY THE HALT-ON-ERROR SWITCH (/FLAG:HOE). HALT IN THIS CONTEXT MEANS RETURN TO COMMAND MODE. THERE ARE NO OTHER PROGRAM HALTS. PRO(CEED) TO RESUME FROM THE POINT OF THE ERROR CALL. CON(TINUE) TO RESTART AT THE BEGINING OF THE TEST IN IN WHICH THE ERROR OCCURED.

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

4.2 -----
DROPPING OF UNITS

UNDER SOME CONDITIONS OF FATAL OR EXCESSIVE ERRORS, A UNIT WILL BE 'DROPPED' FROM THE LIST OF DEVICES BEING TESTED. FOR EXAMPLE, IF A DEVICE REGISTER IS NON-EXISTENT, THE UNIT IS DROPPED BEFORE ACTUAL TESTING COMMENCES. IN ADDITION, A UNIT WILL BE DROPPED IF 25 OR MORE ERRORS OCCUR WITHIN ANY ONE TEST (UNLESS THE /FLAGS:IDU SWITCH IS SET). THE DROPPED UNIT IS REPORTED AND APPEARS IN THE PER-UNIT STATISTICS REPORT. IN SOME CASES, THE ERROR COUNT IS NOT TESTED UNTIL THE END OF A TEST LOOP, SO MORE THAN 25 ERRORS CAN OCCUR BEFORE THE UNIT IS DROPPED.

5.0 -----
PERFORMANCE AND PROGRESS REPORTS

NORMAL PROGRESS THROUGH A GIVEN TEST SEQUENCE IS INDICATED BY THE PRINTING OF EACH TEST'S NUMBER AND TITLE ON THE SYSTEM CONSOLE DEVICE (IF THE /FLAGS:PNT SWITCH IS USED), FOLLOWED BY AN END-PASS WHEN ALL TESTS HAVE BEEN EXECUTED ON ALL UNITS. IN ADDITION, IF THE /FLAGS:ISR SWITCH IS NOT SET, AND IF MULTIPLE UNITS ARE BEING TESTED, THE PER-UNIT STATUS IS PRINTED AT THE END OF EACH PASS.

IF RUNNING WITH ERRORS INHIBITED (/FLA:IER), AN ERROR COUNT IS KEPT FOR EACH TEST, AND DISPLAYED AT THE END OF THAT TEST. THERE ARE NO OTHER PROGRESS REPORTS.

5.1 -----
EXECUTION TIMES

EXECUTION TIMES WILL VARY SOMEWHAT, DEPENDING ON CPU TYPE AND OPERATING MODE. THE FOLLOWING ARE TYPICAL EXECUTION TIMES OBSERVED ON A PDP-11/34 SYSTEM:

	1ST PASS	SUBSEQUENT PASSES
NORMAL MODE	4.5 MIN	8.6 MIN
DPU ONLY MODE	0.4 MIN	2.3 MIN
LONG MEMORY TEST MODE	5.6 MIN	9.7 MIN

386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424

6.0 -----
TEST SUMMARIES

THERE ARE THREE CATAGORIES OF TESTS, AS FOLLOWS:

1. DPU ONLY TESTS (SEC. 6.1)
2. IMAGE MEMORY & SYNC GENERATOR TESTS & DISPLAYS (SEC. 6.2)
3. STAND-ALONE TESTS (SEC. 6.3)

THE FOLLOWING PARAGRAPHS PROVIDE A BRIEF DESCRIPTION OF THE TESTING SEQUENCE. REFER TO THE PROGRAM LISTING FOR DETAILS. DURING PROGRAM INITIALIZATION, CERTAIN MINIMAL TESTING IS DONE IN ORDER TO:

- A. VERIFY THAT THE DEVICE ADDRESS FOR THIS UNIT IS VALID.
- B. VERIFY THAT THIS UNIT IS PROPERLY INITIALIZED FOLLOWING A BUS-RESET.

THE FOLLOWING ERRORS MAY OCCUR OUTSIDE THE CONFINES OF THE NORMAL TEST SEQUENCE:

ERR 00000 UNEXPECTED DPU ERROR INTERRUPT.
ERR 00001 NON-EXISTANT DEVICE REGISTER. ABORT.
ERR 00002 BUS-INIT DIDN'T INITIALIZE DPU.

THE FOLLOWING NOTES APPLY TO ALL TEST SEQUENCES:

1. UNLESS OTHERWISE NOTED, TESTING CONTINUES AFTER ANY ERROR.
2. ALL UNSOLICITED KBD INPUTS ARE IGNORED EXCEPT FOR:
 <^O> SUPPRESS TTY OUTPUT
 <^C> HALT (RETURN TO COMMAND MODE)

426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478

6.1 THE DPU TESTS

THE TESTS IN THIS SECTION ARE THE 'DPU ONLY' TESTS.
THEY ARE RUN WHENEVER THE PROGRAM IS RUN AND THE TEST
SWITCH DOES NOT EXCLUDE THEM. WHEN IN THE 'DPU ONLY' MODE
THESE ARE THE ONLY TESTS THAT MAY (WILL) BE RUN.

6.1.1 TEST 1. STATIC RESET

EXECUTE A RESET INSTRUCTION AND CHECK EACH APPLICABLE REGISTER
FOR ITS CORRECT STATIC STATE.

6.1.2 TEST 2. RESET ONES

SET ALL REGISTERS TO ONES, CHECKING THAT ONES GOT SET, THEN
EXECUTE A RESET INSTRUCTION AND CHECK EACH REGISTER FOR ITS
CORRECT STATIC STATE.

6.1.3 TEST 3. SOFT INIT ONES

SET ALL REGISTERS TO ONES, CHECKING THAT ONES GOT SET, THEN
EXECUTE A SOFTWARE INITIALIZE AND CHECK EACH REGISTER FOR ITS
CORRECT STATIC STATE.

6.1.4 TEST 4. REGISTER UNIQUE ADDRESS

ALL REGISTERS ARE SET TO THEIR NORMAL STATIC STATE VIA SOFT INIT.
EACH APPLICABLE REGISTER, IN TURN, IS SET TO ITS CAPACITY FOR ONES.
THEN ALL OF THE OTHER REGISTERS ARE CHECKED THAT THEY REMAINED STATIC.

6.1.5 TEST 5. INCREMENTING REGISTERS

INCREMENT AND CHECK EACH REGISTER TO THE LIMIT OF ITS CAPABILITY.

480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534

6.1.6 -----
TEST 6. DPU START-STOP

TEST 1 VERIFIES THAT THE DISPLAY PROCESSER (DPU) CAN EXECUTE START, STOP, RESUME, NOP, AND JUMP INSTRUCTIONS.

6.1.7 -----
TEST 7. DPU OPCODES

ALL OPCODES WHICH TRANSFER PIXEL INTENSITY DATA TO THE DPU (100000 THRU 130000) ARE EXECUTED AND THE DSR IS TESTED TO VERIFY THAT THE PIXEL DATA WAS RECEIVED AND RETURNED BY THE DPU. ALL POSSIBLE VALUES OF PIXEL DATA ARE PASSED WITH EACH OPCODE. THEN ALL REMAINING OPCODES EXCEPT BIT MAPS (134000 AND 136000), JUMP (160000), AND STOP (172000) ARE EXECUTED WITH BITS <9:0> OF THE OPCODE WORD SET TO ZERO AND WITH NO FOLLOWING DATA WORD. IN THIS CONFIGURATION, EACH OPCODE (INCLUDING THOSE UN-DEFINED) SHOULD APPEAR AS A 'NOP' AS FAR AS FINAL REGISTER CONTENT IS CONCERNED.

6.1.8 -----
TEST 8. INCREMENTING HISTOGRAM BASE ADDRESS

SET BASE HISTOGRAM AND INCREMENT THE BASE ADDRESS REGISTER TO ITS CAPACITY.

6.1.9 -----
TEST 9. INCREMENTING CHARACTER BASE ADDRESS

SET BASE CHARACTER AND INCREMENT THE BASE ADDRESS REGISTER TO ITS CAPACITY.

6.1.10 -----
TEST 10. DJMS/DPOP

CHECK THE OPERATION OF THE JUMP TO SUBROUTINE INSTUCTION, INCLUDING THE ENABLE BIT AND THE PCSAVE REGISTER.

6.1.11 -----
TEST 11. MAIN MEMORY MGT ACCESS (OVER 28K)

CHECK MAIN MEMORY MANAGEMENT ACCESS FOR ALL AVAILABLE MEMORY OVER 28K.

536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583

6.1.12 TEST 12. AUX MEMORY MGT ACCESS (OVER 28K)

CHECK AUX MEMORY MANAGEMENT ACCESS FOR ALL AVAILABLE MEMORY OVER 28K.

6.1.13 TEST 13. STOP INTERRUPT

THIS TEST VERIFIES THAT THE STOP INTERRUPT LOGIC IN THE DPU FUNCTIONS CORRECTLY. VARIOUS COMBINATIONS OF STOP INSTRUCTIONS WITH INTERRUPT ENABLED AND DISABLED ARE EXECUTED.

6.1.14 TEST 14. DPU TIME-OUT INTERUPT

IN THIS TEST, A NON-EXISTANT MEMORY ADDRESS IS PASSED TO THE DPU PC. IT SHOULD RESPOND WITH A TIME-OUT INTERRUPT BACK TO THE CPU.

6.1.15 TEST 15. ERROR CODES

GENERATE EACH ERROR CONDITION POSSIBLE (TO SOFTWARE) AND CHECK FOR APPROPRIATE ERROR CODE GENERATION.

6.1.16 TEST 16. ADDRESS RELOCATE

THIS TEST VERIFIES THAT THE DPU CAN CALCULATE AN 18 BIT PC FROM AN INITIAL 16 BIT PC, PLUS THE CONTENTS OF THE 12 BIT RELOCATE REGISTER. SINCE THE 2 HIGH ORDER ADDRESS BITS ARE INVISIBLE, THE TEST RELIES ON THE FACT THAT ANY CALCULATED PC WHICH EXCEEDS 18 BITS IN LENGTH WILL BE TRUNCATED TO 18 BITS (I.E. WRAP AROUND TO 000000). THE TEST USES ALL COMBINATIONS OF RELOCATE FACTOR AND INITIAL PC THAT YIELD A FINAL PC WITHIN THE TEST BODY.

585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641

6.1.17 -----
TEST 17. CHARACTERS

IN THE VSV11, THERE IS NO HARDWARE CHARACTER GENERATOR. INSTEAD, CHARACTER MODE CAUSES A SUBROUTINE CALL TO A SUBPICTURE DISPLAY FILE WHICH CONTAINS THE CODE TO DRAW THE DESIRED CHARACTER OR SYMBOL. THE CHARACTER SUB-PIX ADDRESS IS OBTAINED BY USING THE ASCII CHAR CODE TO INDEX INTO AN ADDRESS TABLE WHICH CONTAINS THE STARTING ADDRESSES OF THE CHAR SET. THE STARTING ADDRESS OF THE ADDRESS TABLE IS SET WITH THE 'SET CHAR BASE' INSTRUCTION, AND EACH CHAR SUB-PIX IS TERMINATED WITH A 'DPOP' INSTRUCTION. THIS TEST USES 1 COMMON SUBROUTINE AS A PSEUDO-CHARACTER AND SETS A UNIQUE BASE ADDRESS FOR EACH CHARACTER CODE SUCH THAT: $BASE + 2(CODE) = SUBROUTINE ADDRESS$. THE FULL ASCII SET (000 - 177) IS TESTED IN THIS MANNER.

IF THE PROGRAM IS NOT RUNNING IN DPU-ONLY MODE, THE FULL CHARACTER SET IS THEN DISPLAYED ON THE SCREEN TWICE: IN THE TOP DISPLAY, THE CHARACTERS ARE SITUATED ON EVEN-NUMBERED SCAN LINES, WHILE IN THE BOTTOM DISPLAY THEY ARE ON ODD-NUMBERED SCAN LINES.

6.1.18 -----
TEST 18. ABSOLUTE POINTS

THIS TEST VERIFIES THAT THE DPU CAN EXECUTE ABSOLUTE POINT MODE INSTRUCTIONS UTILIZING THE FULL RANGE OF X/Y.
(1) PLOT ALL X POINTS, HOLDING Y AT 0.
(2) PLOT ALL Y POINTS, HOLDING X AT 0.
(3) PLOT ALL POINTS WHERE $X = Y$.

THE X AND Y POSITION REGISTERS ARE TESTED AS EACH POINT IS EXECUTED. BY DEFAULT, PIXEL DATA ARE TRANSFERRED TO THE IMAGE MEMORY, AND DISPLAYED ON THE MONITOR (IF THERE IS ONE).

THE IMAGE MEMORY AND VIDEO DISPLAY MAY BE INHIBITED BY RESPONDING <YES> TO THE 'RUN DPU TESTS ONLY' QUERY AT START TIME.

6.1.19 -----
TEST 19. LONG VECTORS

GENERATES LONG VECTORS WITH POSITIVE DELTA X/Y FROM $X, Y = 0, 0$ TO ALL POINTS WHERE $X = Y$ (FANS DIAGONALLY FROM BOTTOM TO LEFT EDGE). THEN A SIMILAR PATTERN USING NEGATIVE DELTA X/Y STARTING AT $X, Y = MAX, MAX$. THE X AND Y POSITION REGISTERS ARE TESTED AS EACH VECTOR IS DRAWN.

VIDEO DISPLAY IS INHIBITED IF 'DPU ONLY' WAS SPECIFIED.

643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696

6.1.20 -----
TEST 20. RELATIVE POINTS

VERIFY THAT THE DPU CAN GENERATE RELATIVE POINTS FROM
X,Y = 1000,1000 USING THE FULL RANGE OF DELTA X/Y
(-76 TO +76)

- (1) PLOT ALL X POINTS, HOLDING Y AT -76.
- (2) PLOT ALL Y POINTS, HOLDING X AT -76.
- (3) PLOT ALL POINTS WHERE X = Y.

THE X AND Y POSITION IS VERIFIED AS EACH POINT IS DRAWN.

VIDEO DISPLAY IS INHIBITED IN 'DPU ONLY' MODE AS BEFORE.

6.1.21 -----
TEST 21. SHORT VECTORS

SIMILAR TO TEST 10 ABOVE, EXCEPT USING SHORT VECTORS.

6.1.22 -----
TEST 22. RUN LENGTH

CHECK ALL ASPECTS OF THE RUN LENGTH MODE.

IF RUNNING 'DPU ONLY' MODE, YOU WILL SEE 'END-PASS'
AT THIS POINT. ALL OF THE REMAINING TESTS REQUIRE
THE SERVICES OF THE IMAGE MEMORY.

6.2 -----
IMAGE MEMORY & SYNC GENERATOR TESTS & DISPLAYS

THE TESTS & DISPLAYS IN THIS SECTION REQUIRE THE SERVICES
OF THE IMAGE MEMORY AND SYNC GENERATOR MODULES.
IF THE PROGRAM IS BEING RUN IN
THE 'DPU ONLY' MODE, THESE TESTS MAY NOT (WILL NOT) BE RUN.
OTHERWISE, THEY WILL BE RUN IN SEQUENCE AS LONG AS THEY
ARE NOT INHIBITED BY THE TEST SWITCH.

6.1.23 -----
TEST 23. CURSOR REGISTERS/SWITCH/MATCH

THE CURSOR POSITION REGISTERS, JOYSTICK STATUS ENABLES,
THE 'SOFT' SWITCH, SWITCH INTERRUPT AND MATCH INTERRUPT
ARE TESTED FOR EACH AVAILABLE SYNC GENERATOR CHANNEL.

698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745

6.2.1 -----
TEST 24. GRAPH/HISTOGRAM X

GRAPH PLOT ALL X POINTS BETWEEN 200 AND 1600 (IN 30 UNIT INCREMENTS) USING ALL VALUES OF Y INCREMENT.

I.E. X 200, Y INCR 2
X 230, Y INCR 4

⋮

X 1600, Y INCR 76

THEN REPLOT EACH POINT IN HISTOGRAM MODE TO A RELATIVE BASE LINE WHERE BASE = X-20. THE X AND Y POSITION REGISTERS ARE TESTED AS EACH POINT IS PLOTTED (IN BOTH MODES).

IF RUNNING 'DPU ONLY' MODE, THIS AND ALL FOLLOWING TESTS ARE NOT RUN.

6.2.2 -----
TEST 25. GRAPH/HISTOGRAM Y

SAME AS TEST 11, EXCEPT PLOT IN Y.

6.2.3 -----
TEST 26. BIT MAP (1)

VERIFIES THAT BIT MAP MODE 1 FUNCTIONS CORRECTLY FOR ALL POSSIBLE RUN LENGTHS (0 TO 511.) AND IN BOTH 4 BIT AND 8 BIT PIXEL MODES. A 256 WORD BUFFER IS FILLED WITH THE VALUE 000377, WHICH WILL YIELD A 2 ON, 2 OFF PATTERN OF 4 BIT PIXELS AND/OR A 1 ON, 1 OFF PATTERN OF 8 BIT PIXELS.

THE X AND Y POSITION IS VERIFIED AT THE END OF EACH BIT MAP OPERATION.

6.2.4 -----
TEST 27. BIT MAP (0)

VERIFIES THAT BIT MAP MODE 0 FUNCTIONS CORRECTLY IN ALL IT'S VARIATIONS.

747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795

6.2.5 -----
TEST 28. IMAGE MEMORY CLEAR/SET

NOTE: THIS TEST AND THE ONE FOLLOWING REQUIRE THE SERVICES OF THE PIXEL-READ-BACK FUNCTION.

VERIFIES THAT THE IMAGE MEMORY "CLEAR TO ZERO" AND "SET TO DATA" INSTRUCTIONS FUNCTION CORRECTLY.
(1) THE IMAGE MEMORY IS "CLEARED" AND SELECTED PIXELS ARE READ-BACK AND VERIFIED.
(2) THE IMAGE MEMORY IS "SET TO ALL 1'S" AND SELECTED PIXELS READ-BACK AND VERIFIED.

IN "SHORT MODE", TEST ALL X ADDRESSES FOR Y = 0 AND 2 (EVEN VS ODD), THEN TEST ALL Y ADDRESSES FOR X = 0. IN "LONG MODE", TEST ALL PIXEL ADDRESSES.

NOTE: PART 2 OF THIS TEST FILLS THE IMAGE MEMORY WITH ALL 1'S. THE VIEWING AREA SHOULD BE ALL WHITE -- ANY BLANK SPOTS ARE EITHER BAD IMAGE MEMORY LOCATIONS OR BURNED SPOTS ON THE CRT PHOSPHOR.

6.2.6 -----
TEST 29. INTERLACE

CHECK FOR MEMORY INTERLACE BY WRITING A HORIZONTAL, EVEN NUMBERED LINE, AND CHECKING THAT BACK THE "LINE" ABOVE THE ONE WRITTEN READS BACK AS ZERO.

6.2.7 -----
TEST 30. IMAGE MEMORY PATTERNS

VERIFIES THAT A GIVEN PIXEL CAN BE SET TO ALL POSSIBLE BIT COMBINATIONS, AND THAT THE READ-BACK FUNCTION CAN CORRECTLY READ THAT DATA.

"SHORT" AND "LONG" MODE OPTIONS APPLY AS BEFORE.

6.2.8 -----
TEST 31. SPARE

RUNS AS A BLANK TEST

797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842

6.2.9 -----
TEST 32. SYSTEM VERIFICATION DISPLAY

THIS TEST PROVIDES ONE TEST PATTERN IN WHICH ALL GRAPHICS FUNCTIONS ARE EXERCISED:

1. CHARACTERS AT BOTTOM CENTER.
2. EDGE CLIPPING AT LOWER LEFT AND UPPER RIGHT.
3. THREE OCTAGONS USING LONG VECTORS (LARGE ONE), SHORT VECTORS (2 SMALL ONES), AND SOME RELATIVE POINTS INSIDE THE SMALL ONES.
4. GRAPH-HISTOGRAM Y AT UPPER LEFT.
5. GRAPH-HISTOGRAM X AT LOWER RIGHT.
6. BIT MAPS -- MODE 1 AT BOTTOM, MODE 0 AT TOP.
7. A CIRCULAR VECTOR SWEEP AT CENTER, SHOWING 16 COLOR/INTENSITY LEVELS.
8. A PERIMETER OUTLINE OF THE DISPLAY.
9. STATIC FOR 2 SECONDS, THEN BLINKING FOR 2 SECONDS.

6.3 -----
STAND-ALONE TESTS & ROUTINES

THE TESTS AND ROUTINES IN THIS SECTION ARE STAND-ALONE. THEY ARE NOT INCLUDED IN THE NORMAL TEST SEQUENCE AND MUST BE RUN BY UNIQUE START COMMAND

IE. START/TEST:NN<CR> OR RESTART/TEST:NN<CR>

6.3.1 -----
TEST 33. JOY-STICK VERIFICATION

THIS IS A TEST FRAME FOR THE JOY-STICK, AND INCLUDES 3 NESTED BOXES(CO-ORDINATES LABELED), AND AN IN-LINE READ-OUT OF THE CURRENT JOY-STICK POSITION. A "SWITCH" INTERRUPT IS MARKED BY AN 'X' AT THE CURRENT CO-ORDS. A "MATCH" INTERRUPT (ON THE BOXES ONLY) IS MARKED BY AN X AS ABOVE, AND BY THE WORD 'MATCH' DISPLAYED AT UPPER RIGHT.

844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
900
901

6.3.2 TEST 34. SELECTED DISPLAYS

THIS IS A PATTERN GENERATOR FOR COLOR MONITOR CONVERGENCE SET-UP. IT INCLUDES THE FOLLOWING PATTERNS SELECTABLE BY THE USER AT WILL.

- 0 'TYPE THIS' (ON CONSOLE)
- 1 TURN ON BLINKING
- 2 TURN OFF BLINKING
- 3 COLOR BARS
- 4 7 X 7 DOTS
- 5 7X7 CROSSHATCH
- 6 PERIMETER OUTLINE
- 7 BASIC COLOR ID
- 8 GUNS ID
- 9 ALL WHITE SCREEN
- 10 ALTERNATING WHITE SCREEN & PERIMETER OUTLINE
- 11 ALL RED SCREEN
- 12 ALL BLUE SCREEN
- 13 ALL GREEN SCREEN

NOTE: SELECTIONS 4 THROUGH 6 ALSO INCLUDE A PERIMETER OUTLINE OF THE SCREEN, IN WHICH EACH SIDE OF THE OUTLINE INCLUDES 3 'DOTS'. SELECTIONS 3, 7 AND 8 INCLUDE A SOLID WHITE PERIMETER OUTLINE.

6.3.3 TEST 35. SYSTEM CONFIGURATION TYPE-OUT

PRINT ON THE CONSOLE THE CONFIGURATION OF THE VSV11 SYSTEM.

7.0 PROGRAM LISTING

PROGRAM LISTING FOLLOWS:

.NLIST BEX
.DSABL GBL

z

```

PROGRAM HEADER
903          .SBTTL  PROGRAM HEADER
904
905          .MCALL  SVC
906 000000   SVC          ; INITIALIZE SUPERVISOR MACROS
907
908          :XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
909          : IF STRUCTURED MACROS ARE TO BE USED, YOU MUST ADD
910          : .REQ    SPMAC.SML (OR THE CURRENT EQUIVALENT)
911          : .MCALL  STRUCT
912          :
913          :
914          000001   SVCGBL= 1          ; LIST GLOBAL TAGS AT RIGHT MARGIN.
915          000001   SVCTST= 1          ;          DITTO  TEST TAGS.
916          000001   SVCSUB= 1         ;          DITTO  SUBTEST TAGS.
917          000001   SVCTAG= 1         ;          DITTO  ANY OTHER TAGS.
918          000001   SVCINS= 1         ;          DITTO  INSTRUCTIONS AND DATA.
919
920          : THESE SYMBOLS CONTROL THE LISTING FIELD OF ALL SVC MACRO
921          : EXPANSIONS. YOU MAY CHANGE THEM AT ANY TIME OR PLACE.
922
923          :          1 = RIGHT-JUSTIFY (MAKES IT EASY TO DISTINGUISH
924          :          SVC'S MACRO CODE FROM YOUR OWN).
925          :          0 = LEFT-JUSTIFY (ALIGN IN A NORMAL FASHION).
926          :          -1 = DON'T LIST THE EXPANSIONS AT ALL.
927          :XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
928
929          .ENABL  ABS,AMA
930          002000   .=          2000
931
932          :++
933          : THE PROGRAM HEADER IS THE INTERFACE BETWEEN
934          : THE DIAGNOSTIC PROGRAM AND THE SUPERVISOR.
935          :--
936          174561   PRGSIZ= <L$LAST>/16. ; PROGRAM SIZE IN 1/8 K UNITS (OCTAL).
937          000000   SVCGBL= 0          ; ALIGN THE HEADER STUFF.
938          000000   SVCINS= 0
939
940          002000   POINTER BGNSW,BGNSFT,BGNAU,BGNDU,BGNRPT
941          002000   HEADER  CVVSA,A,0,655.,0
          L$NAME::          ;DIAGNOSTIC NAME
          .ASCII  /C/
          .ASCII  /V/
          .ASCII  /V/
          .ASCII  /S/
          .ASCII  /A/
          .BYTE   0
          .BYTE   0
          .BYTE   0
          L$REV::          ;REVISION LEVEL
          .ASCII  /A/
          L$DEPO::          ;0
          .ASCII  /0/
          L$UNIT::          ;NUMBER OF UNITS
          .WORD   0
          L$TIML::          ;LONGEST TEST TIME
          .WORD   655.
          L$HPCP::          ;POINTER TO H.W. QUES.
          002000   103
          002001   126
          002002   126
          002003   123
          002004   101
          002005   000
          002006   000
          002007   000
          002010   101
          002011   060
          002012   000000
          002014   001217
          002016
    
```


PROGRAM HEADER

002016	002240			
002020		L\$SPCP::	.WORD	L\$HARD ;PTR. TO S.W. QUES.
002020	002522			
002022		L\$HPTP::	.WORD	L\$SOFT ;PTR. TO DEF. H.W. PTABLE
002022	002224			
002024		L\$SPTP::	.WORD	L\$HW ;PTR. TO S.W. PTABLE
002024	002502			
002026		L\$LADP::	.WORD	L\$SW ;DIAG. END ADDRESS
002026	113404			
002030		L\$STA::	.WORD	L\$LAST ;RESERVED FOR APT STATS
002030	000000			
002032		L\$CO::	.WORD	0
002032	000000			
002034		L\$DTYP::	.WORD	0 ;DIAGNOSTIC TYPE
002034	000000			
002036		L\$APT::	.WORD	0 ;APT EXPANSION
002036	000000			
002040		L\$DTP::	.WORD	0 ;PTR. TO DISPATCH TABLE
002040	107114			
002042		L\$DISPATCH	.WORD	L\$DISPATCH ;DIAGNOSTIC RUN PRIORITY
002042	000000			
002044		L\$ENVI::	.WORD	0 ;FLAGS DESCRIBE HOW IT WAS SETUP
002044	000000			
002046		L\$EXP1::	.WORD	0 ;EXPANSION WORD
002046	000000			
002050		L\$MREV::	.WORD	0 ;SVC REV AND EDIT #
002050	003			
002051	003			
002052		L\$EF::	.BYTE	C\$REVISION ;DIAG. EVENT FLAGS
002052	000000			
002054	000000			
002056		L\$SPC::	.WORD	0
002056	000000			
002060		L\$DEVP::	.WORD	0 ; POINTER TO DEVICE TYPE LIST
002060	002162			
002062		L\$REPP::	.WORD	L\$DVTYP ;PTR. TO REPORT CODE
002062	033272			
002064		L\$EXP4::	.WORD	L\$RPT
002064	000000			
002066		L\$EXP5::	.WORD	0
002066	000000			
002070		L\$AUT::	.WORD	0 ;PTR. TO ADD UNIT CODE
002070	033012			
002072		L\$DUT::	.WORD	L\$AU ;PTR. TO DROP UNIT CODE
002072	033110			
002074		L\$LUN::	.WORD	L\$DU ;LUN FOR EXERCISERS TO FILL
002074	000000			
002076		L\$DESP::	.WORD	0 ;PTR. TO DIAG. DESCRIPTION
002076	002122			
002100		L\$LOAD::	.WORD	L\$DESC ;GENERATE SPECIAL AUTOLOAD EMT
002100	104035			
002102		L\$SETP::	EMT	E\$LOAD ;PTR. TO ERRTABL
002102	000000			
002104		L\$ICP::	.WORD	0 ;PTR. TO INIT CODE
002104	032014			
002106		L\$CCP::	.WORD	L\$INIT ;PTR. TO CLEAN-UP CODE
002106	033220			
			.WORD	L\$CLEAN

002110				LSACP::					:PTR. TO AUTO CODE
002110	033216			.WORD	LSAUTO				
002112				LSPRT::					:PTR. TO PROTECT TABLE
002112	032004			.WORD	LSPROT				
002114				LSTEST::					:TEST NUMBER
002114	000000			.WORD	0				
002116				LSDLY::					:DELAY COUNT
002116	000000			.WORD	0				
002120				LSHIME::					:PTR. TO HIGH MEM
002120	000000			.WORD	0				
942	002122			DESCRIPT	<**** VSV11-VS11 DIAGNOSTIC ****>				
	002122			LSDESC::					
	002122	052	052	.ASCIZ	/**** VSV11-VS11 DIAGNOSTIC ****/				
				.EVEN					
943	002162			DEV TYP	<VSV11-VS11, AND DISPLAY MONITOR>				
	002162			LSDV TYP::					
	002162	126	123	.ASCIZ	/VSV11-VS11, AND DISPLAY MONITOR/				
				.EVEN					
944				SVCGBL=	1				: SHOVE EVERYTHING BACK TO THE RIGHT.
945	000001			SVCINS=	1				
946	000001			SVCGBL=-	1				: KILL ALL SVC STUFF.
947	177777			SVCTST=-	1				
948	177777			SVCSUB=-	1				
949	177777			SVCTAG=-	1				
950	177777			SVCINS=-	1				
951	177777								


```

972          .SBTTL  HARDWARE PARAMETER CODING SECTION
973
974          :++
975          : THE HARDWARE PARAMETER CODING SECTION CONTAINS MACROS
976          : THAT ARE USED BY THE SUPERVISOR TO BUILD P-TABLES.  THE
977          : MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE
978          : INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES.  THE
979          : MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS
980          : WITH THE OPERATOR.
981          :--
982 002236          BGNHRD
983
984 002240          GPRMA  HPM1,0,0,160000,177776,YES ;GET 1ST REGISTER ADDRESS.
985 002250          GPRMA  HPM2,2,0,0,776,YES      ;GET 1ST VECTOR ADDRESS.
986 002260          GPRMD  HPM3,4,0,340,0,7,YES    ;GET INTERRUPT PRIORITY.
987 002272          GPRML  HPM4,6,-1,YES          ;ASK IF LUT AVAILABLE
988 002300          GPRML  HPM5,10,-1,YES         ;ASK IF FREQUENCY = 50 HZ.
989 002306          EXIT HRD
990 002310          104    105    126  HPM1:  .ASCIZ  /DEVICE ADDRESS      /
991 002340          061    123    124  HPM2:  .ASCIZ  /1ST INTERRUPT VECTOR /
992 002370          111    116    124  HPM3:  .ASCIZ  /INTERRUPT PRIORITY /
993 002420          114    125    124  HPM4:  .ASCIZ  /LUT INSTALLED      /
994 002450          106    122    105  HPM5:  .ASCIZ  /FREQUENCY = 50HZ      /
995          .EVEN
996 002500          ENDHRD
  
```

```

998          .SBTTL  SOFTWARE P-TABLE
999
1000         :++
1001         : THE SOFTWARE P-TABLE CONTAINS THE VALUES OF THE PROGRAM
1002         : PARAMETERS THAT CAN BE CHANGED BY THE OPERATOR.
1003         :--
1004 002500          BGNSW  SFPTBL
1005 002502 000000  DPUMOD:: .WORD 0          : DPU/MEM TEST MODE...
1006                                     : ... 0 = RUN ALL...
1007                                     : ...NZ = RUN DPU ONLY.
1008 002504 000000  IMTMOD:: .WORD 0         : IMAGE MEMORY TEST MODE...
1009                                     : ... 0 = SHORT TEST...
1010                                     : ...NZ = LONG TEST.
1011 002506 000000  TIDFLG:: .WORD 0         : TEST ID'S...
1012                                     : ... 0 = TYPE TEST ID AT EACH TTET START...
1013                                     : ...NZ = DON'T.
1014 002510 000000  NOITS:: .WORD 0          : INHIBIT ITERATION OPTION.
1015                                     : ... 0 = ITERATE.
1016                                     : ...NZ = INHIBIT ITERATE.
1017 002512 000000  MFGFLG:: .WORD 0         : MANUFACTURING TEST MODE...
1018                                     : ... 0 = USER/FIELD SVC...
1019                                     : ...NZ = MANUFACTURING.
1020 002514 000031  LERRMAX::          .WORD 25. : LOCAL (PER TEST) ERROR LIMIT
1021 002516 000310  GERRMAX::          .WORD 200. : GLOBAL (PER UNIT) ERROR LIMIT
1022 002520          ENDSW
1023
1024
    
```



```

1026          .SBTTL  SOFTWARE PARAMETER CODING SECTION
1027
1028          :++
1029          : THE SOFTWARE PARAMETER CODING SECTION CONTAINS MACROS
1030          : THAT ARE USED BY THE SUPERVISOR TO BUILD P-TABLES.  THE
1031          : MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE
1032          : INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES.  THE
1033          : MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS
1034          : WITH THE OPERATOR.
1035          :--
1036 002520          BGNSFT
1037 002522          GPRML  SPM1,0,-1,YES          ; GET OPTIONAL DPU ONLY FLAG.
1038 002530          GPRML  SPM2,2,-1,YES          ; GET MEM TEST MODE.
1039          :GPRML  SPM3,4,-1,YES          ; GET TEST ID CONTROL.
1040 002536          GPRML  SPM4,6,-1,YES          ; GET ITERATION CONTROL.
1041 002544          GPRML  SPM5,10,-1,YES         ; GET MANUFACTURING MODE.
1042 002552          GPRMD  SPM6,12,D,7777,0,7777,YES ; GET LOCAL ERROR LIMIT
1043 002564          GPRMD  SPM7,14,D,7777,0,7777,YES ; GET GLOBAL ERROR LIMIT
1044 002576          EXIT SFT
1045 002600          122      125      116  SPM1:  .ASCIZ  /RUN DPU TESTS ONLY      /
1046 002630          114      117      116  SPM2:  .ASCIZ  /LONG MEMORY TESTS      /
1047          :SPM3:  .ASCIZ  /INHIBIT TEST ID'S      /
1048 002660          111      116      110  SPM4:  .ASCIZ  /INHIBIT ITERATIONS      /
1049 002710          115      101      116  SPM5:  .ASCIZ  /MANUFACTURING MODE      /
1050 002740          120      105      122  SPM6:  .ASCIZ  /PER TEST ERROR LIMIT      /
1051 002770          120      105      122  SPM7:  .ASCIZ  /PER UNIT ERROR LIMIT      /
1052 003020          ENDSFT          ; UNUSED.
    
```

1054
1055
1056
1057
1058
1059
1060
1061 003020

.SBTTL GLOBAL EQUATES SECTION

:++
: THE GLOBAL EQUATES SECTION CONTAINS PROGRAM EQUATES THAT
: ARE USED IN MORE THAN ONE TEST.
:--

EQUALS ; GET STANDARD EQUATES.

: BIT DIFINITIONS

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
000100	BIT6== BIT06
000040	BIT5== BIT05
000020	BIT4== BIT04
000010	BIT3== BIT03
000004	BIT2== BIT02
000002	BIT1== BIT01
000001	BIT0== BIT00

: EVENT FLAG DEFINITIONS
: EF32:EF17 RESERVED FOR SUPERVISOR TO PROGRAM COMMUNICATION

000040	EF.START== 32.	: START COMMAND WAS ISSUED
000037	EF.RESTART== 31.	: RESTART COMMAND WAS ISSUED
000036	EF.CONTINUE== 30.	: CONTINUE COMMAND WAS ISSUED
000035	EF.NEW== 29.	: A NEW PASS HAS BEEN STARTED
000034	EF.PWR== 28.	: A POWER-FAIL/POWER-UP OCCURRED

: PRIORITY LEVEL DEFINITIONS

000340	PRI07== 340
000300	PRI06== 300
000240	PRI05== 240
000200	PRI04== 200
000140	PRI03== 140
000100	PRI02== 100

```
000040      PRI01== 40
000000      PRI00== 0
           .
           .: OPERATOR FLAG BITS
           .
000004      EVL==      4
000010      LOT==     10
000020      ADR==     20
000040      IDU==     40
000100      ISR==    100
000200      UAM==    200
000400      BOE==    400
001000      PNT==   1000
002000      PRI==   2000
004000      IXE==   4000
010000      IBE==  10000
020000      IER==  20000
040000      LOE==  40000
100000      HOE== 100000
```

1062

```
1064      .SBTTL MEMORY MANAGEMENT DEFINITIONS
1065
1066      ;*KT11 VECTOR ADDRESS
1067
1068      000250      MMVEC= 250
1069
1070      ;*KT11 STATUS REGISTER ADDRESSES
1071
1072      177572      SR0= 177572
1073      177574      SR1= 177574
1074      177576      SR2= 177576
1075      172516      SR3= 172516
1076
1077      ;*KERNEL 'I' PAGE DESCRIPTOR REGISTERS
1078
1079      172300      KIPDR0= 172300
1080      172302      KIPDR1= 172302
1081      172304      KIPDR2= 172304
1082      172306      KIPDR3= 172306
1083      172310      KIPDR4= 172310
1084      172312      KIPDR5= 172312
1085      172314      KIPDR6= 172314
1086      172316      KIPDR7= 172316
1087
1088      ;*KERNEL 'I' PAGE ADDRESS REGISTERS
1089
1090      172340      KIPAR0= 172340
1091      172342      KIPAR1= 172342
1092      172344      KIPAR2= 172344
1093      172346      KIPAR3= 172346
1094      172350      KIPAR4= 172350
1095      172352      KIPAR5= 172352
1096      172354      KIPAR6= 172354
1097      172356      KIPAR7= 172356
```

```

1099      .SBTTL  VSV-11 INSTRUCTION EQUATES.
1100      ;
1101      ;PLOTING INSTRUCTIONS.
1102      ;
1103      100000  CHAR== 100000      ; CHARACTER MODE.
1104      104000  SVEC== 104000    ; SHORT VECTOR MODE.
1105      110000  LVEC== 110000    ; LONG VECTOR MODE.
1106      114000  APNT== 114000    ; ABSOLUTE POINT MODE.
1107      120000  GHX== 120000    ; GRAPH/HISTOGRAM X MODE.
1108      124000  GHY== 124000    ; GRAPH/HISTOGRAM Y MODE.
1109      130000  RPNT== 130000    ; RELATIVE POINT MODE.
1110      ;
1111      ;PIXEL DATA FIELD <10:2> FOR THE ABOVE OPCODES.
1112      ;
1113      002000  L0== BIT10        ; LEVEL 0 (BLACK) IS JUST THE ENABLE.
1114      003774  L255== L0!<255.*BIT2> ; LEVEL 255 IS THE MAX POSSIBLE.
1115      003774  ALL== L255
1116      003774  ALLB== 3774      ; ALL INTENSITIES (WITH BLINK).
1117      003374  ALLNB== 3374    ; ALL INTENSITIES (WITHOUT BLINK).
1118      ;
1119      ;
1120      ;COLOR DEFINITIONS:
1121      003774  WHITE == ^03774 ; ALL
1122      002400  GRN1 == ^0<400!2000> ; BIT8!L0
1123      003000  GRN2 == ^0<1000!2000> ; BIT9!L0
1124      003400  GRN3 == GRN1!GRN2 ; BIT8!BIT9
1125      002200  REDG == ^0<200!2000> ; BIT7!L0
1126      002100  BLUEG == ^0<100!2000> ; BIT6!L0
1127      003600  YELLOW == GRN3!REDG
1128      003500  EGGBL == GRN3!BLUEG
1129      003300  VIOLET == GRN2!REDG!BLUEG
1130      003200  GOLD == GRN2!REDG
1131      002300  MAGEN == REDG!BLUEG
1132      ;
1133      ; COLOR LOOK-UP-RAM DEFINITIONS.
1134      ;
1135      010000  LREAD== 1*BIT12    ; CSR -- READ DATA FROM ADDR <7:0>.
1136      002000  LDI== 1*BIT10    ; INTERRUPT ON READ/WRITE DONE.
1137      ;
1138      010000  LSET== 1*BIT12    ; DATA REG -- SET ALL ADDRESSES TO...
1139      ;...DATA PATTERN IN <11:0>...
1140      ;...BLU<11:8>, GRN<7:4>, RED<3:0>.
1141      ;
1142      000100  GCOFF== 1*BIT6    ; MAINT REG -- GAMMA CORRECT OFF.
1143      000040  CVC== 1*BIT5     ; COMP VIDEO > MDAC (READ-ONLY).
1144      000020  REDC== 1*BIT4    ; RED DAC > MDAC (READ-ONLY).
1145      000010  GRNC== 1*BIT3    ; GRN DAC > MDAC (READ-ONLY).
1146      000004  BLUC== 1*BIT2   ; BLU DAC > MDAC (READ-ONLY).
1147      000000  MDV0== 0         ; SET MAINT DAC TO 0.0 V.
1148      000001  MDV1== 1         ; 0.5 V.
1149      000002  MDV2== 2         ; 1.0 V.
1150      000003  MDV3== 3         ; 1.35 V.
1151      ;
1152      ; BIT MAP MODE DEFINITIONS.
1153      ;
1154      136000  BM14== 136000     ; MODE 1 WITH 4 BIT PIXELS.
1155      137000  BM18== 137000     ; MODE 1 WITH 8 BIT PIXELS.

```


1156				: PIXEL COUNT IN <8:0>.
1157				
1158	134000	BM04==	134000	: MODE 0 WITH 4 BIT PIXELS.
1159	135000	BM08==	135000	: MODE 0 WITH 8 BIT PIXELS.
1160	000000	M32==	0	: 32 SQUARE PIXEL ARRAY.
1161	000001	M64==	1	: 64 SQUARE.
1162	000002	M128==	2	: 128 SQUARE.
1163	000003	M256==	3	: 256 SQUARE.
1164	000010	EX2==	10	: EXPAND BY 2, NO SMOOTHING.
1165	000020	EX4==	20	: EXPAND BY 4, NO SMOOTHING.
1166	000050	EXSM2==	50	: EXPAND AND SMOOTH BY 2.
1167	000060	EXSM4==	60	: EXPAND AND SMOOTH BY 4.
1168	000100	R256==	100	: 256 X 256 RESOLUTION.
1169	000400	RND8==	400	: ROUND FOR 8-BIT-WIDE IMAGE MEMORY
1170	000200	ODDSKP==	200	: SKIP ODD-NUMBERED LINES ON SCREEN (FOR NON-INTERLACED) ;aaaI
1171				
1172	144000	RNLN==	144000	: OP-CODE FOR RUN-LENGTH MODE
1173	000100	DBLPX==		: DOUBLE THE PIXEL COUNT
1174	000040	YDOWN==	BIT5	: DISPLAY LINES TOP-TO-BOTTOM
1175	000400	RLSKIP==	BIT8	: SKIP LINES, WITHOUT THROWING THE DATA AWAY
1176				: (FOR RUN-LENGTH MODE ONLY)
1177				: SKIPS EVEN LINES IF START ON EVEN, & VICE-VERSA
1178	162000	IMREAD==	162000	

```

1180 ;CONTROL AND STATUS INSTRUCTIONS.
1181 ;
1182 146040 CUOFF== 146040 ;CURSOR (JOY-STICK) VIDEO OFF.
1183 146060 CUON== 146060 ;CURSOR VIDEO ON.
1184 146100 CURD== 146100 ;READ JSX,JSY => DXR,DYR.
1185 146016 CUIM== 146016 ;INTERRUPT ON MATCH (SWITCH DISABLED).
1186 146013 CUIS== 146013 ;INTERRUPT ON SWITCH (MATCH DISABLED).
1187 146012 CUIOFF== 146012 ;INTERRUPTS (BOTH) OFF.
1188 175000 CUWT== 175000 ;CURSOR WRITE.
1189 175000 LDCP== 175000 ;LOAD CURSOR POSITION (=CUWT)
1190 146000 LDJSS== 146000 ;LOAD JOYSTICK STATUS DISPLAY INSTRUCTION
1191 175400 LDECC== 175400 ;LOAD EXTENDED CURSOR CONTROL
1192 000002 SWCHON== BIT1 ;SOFT SWITCH ON.
1193 175401 BLINK== 175401 ;BLINK.
1194 175400 NBLINK==175400 ;NO BLINK.
1195
1196 000002 JSWE == BIT1 ;JOYSTICK SWITCH-ENABLE WRITE ENABLE
1197 000001 JSWD == BIT0 ;JOYSTICK SWITCH-ENABLE WRITE DATA
1198 000010 JMWE == BIT3 ;CURSOR MATCH-ENABLE WRITE ENABLE
1199 000004 JMWD == BIT2 ;CURSOR MATCH-ENABLE WRITE DATA
1200 000040 JCWE == BIT5 ;CROSSHAIR-INTENSITY WRITE-ENABLE
1201 000020 JCWD == BIT4 ;CROSSHAIR-INTENSITY WRITE DATA
1202
1203
1204 150000 SETHB== 150000 ;SET GRAPH/HISTO BASE LINE.
1205 152000 SETCB== 152000 ;SET CHARACTER BASE ADDRESS.
1206
1207 160000 DJMP== 160000 ;DISPLAY JUMP.
1208 160001 DJMS== 160001 ;DISPLAY JUMP-TO-SUBROUTINE
1209 164000 DNOP== 164000 ;DISPLAY NO-OP.
1210 165000 DPOP== 165000 ;DPOP = RTS FROM CHARACTER SUB-PIX.
1211 164001 SYNC== DNOP+1 ;PROCEED IN SYNC = DNOP + N <8:0>.
1212 ;WAITS FOR N OCCURRENCES OF VERTICAL...
1213 ;...SYNC START, THEN PROCEEDS...
1214 ;...(16.6MS @ 60HZ OR 20MS @ 50HZ).
1215 000001 AUXSEG== BIT0 ;SPECIFY AUX. SEGMENT IN DISPATCH ADDRESSES. 0001
1216
1217 170200 SWTCH== 170200 ;SWITCH READ/WRITE STATUS.
1218 170140 CLRMEM== 170140 ;CLEAR IMAGE MEMORY TO 0'S.
1219 170100 SETMEM== 170100 ;SET IMAGE MEMORY TO CURRENT PIX DATA.
1220
1221 172000 STOP== 172000 ;DISPLAY STOP.
1222 171000 SIOFF== 171000 ;STOP INTERRUPT DISABLE.
1223 171400 SION== 171400 ;STOP INTERRUPT ENABLE.
1224 173000 STOPN== STOP!SIOFF ;STOP, DO NOT INTERRUPT.
1225 173400 STOPI== STOP!SION ;STOP AND INTERRUPT.
1226
1227 174100 GXI== 174100 ;SET X INCREMENT <5:0> FOR GRAPH/HST Y.
1228 174100 GYI== GXI ;SET Y INCREMENT <5:0> FOR GRAPH/HST X.
1229
1230 000001 HCPY== BIT0 ;ENABLE HARD-COPY.
1231 000010 SWE== BIT3 ;ENABLE SWITCH RD/WRT.
1232 176000 PROTEC== 176000 ;PROTECT MEMORY (OFF).
1233 176050 READ== 176040!SWE ;READ MEM (DISPLAY), SWITCH ENABLED.
1234 176034 WRT== 176024!SWE ;WRITE MEM (1'S AND 0'S), SWITCH ENABLED
1235 176036 WRT1== 176026!SWE ;WRITE MEM (1'S ONLY), SWITCH ENABLED.
1236 176074 RDWRT== READ!WRT ;READ, WRITE ALL DURING RETRACE.

```

1237	176076	RDWRT1== READ!WRT1	:READ, WRITE 1'S DURING RETRACE.
1238	176051	HCOPY== READ!HCPY	:READ MEM => HARD COPY DEVICE.
1239			
1240	000000	CH0== 0	:CHANNEL SELECT BITS...
1241	000400	CH1== BIT8	:...FOR MEMORY CONTROL...
1242	001000	CH2== BIT9	:...INSTRUCTIONS (176XXX).
1243	001400	CH3== BIT8!BIT9	
1244			
1245	000000	JS0== CH0	:JOY-STICK UNIT SELECT BITS...
1246	000400	JS1== CH1	:...FOR CURSOR CONTROL...
1247	001000	JS2== CH2	:...INSTRUCTIONS (146XXX).
1248	001400	JS3== CH3	

```

1250 ;OTHER USEFUL DEFINITIONS.
1251 .
1252 040000 I== BIT14 ;INTENSIFY VECTOR OR POINT.
1253 020000 MSX== BIT13 ;MINUS SIGNS FOR...
1254 000100 MSY== BIT6 ;...SHORT FORM VECTOR DATA.
1255 020000 MXY== BIT13 ;MINUS SIGN FOR LONG FORM DATA.
1256 020000 M== BIT13 ;ANOTHER MINUS SIGN
1257 040000 HST== BIT14 ;USE GHX/GHY DATA AS HISTOGRAM.
1258 020000 CGRPH== BIT13 ;USE GHX/GHY DATA AS CONNECTED GRAPH
1259 060000 FHST== HST!BIT13 ;USE GHX/GHY DATA AS FILLED HISTOGRAM (BAR-GRAPH)
1260 010000 GHIINH== BIT12 ; INHIBIT ERROR DURING PLOT.
1261
1262 001776 MAXX== 511.*2 ;MAXIMUM X ADDRESS (512 X 512 X 2).
1263 001776 MAXY== MAXX ;SAME FOR Y.
1264 000776 HAFX== <MAXX/2>-1 ;HALF MAX X (CENTER SCREEN).
1265 000776 HAFY== HAFX ;SAME FOR Y.
1266
1267 001776 MAXY50== 511.*2 ;MAX VISIBLE Y ON 50HZ SYSTEM.
1268 000776 HAFY50== <MAXY50/2>-1 ;HALF VISIBLE Y.
1269
1270 001676 MAXY60== <511.-32.>*2 ;MAX VISIBLE Y ON 60HZ SYSTEM.
1271 000736 HAFY60== <MAXY60/2>-1 ;HALF VISIBLE Y.
1272
1273 000002 DX== 2 ;DELTA X (1 PIXEL UNIT).
1274 000002 DYI== DX ;DELTA Y, INTERLACED MODE.
1275 000004 DYNI== DX*2 ;DELTA Y, NON-INTERLACED MODE.
1276
1277 ;DSR REGISTER SELECT CODES:
1278 000000 SELDSR== 0 ;SELECT REAL DSR
1279 000001 SELPCS== 1 ;SELECT PCSAVE
1280 000002 SELFLG== 2 ;SELECT FLAGS
1281 000003 SELCSR== 3 ;SELECT CSR
1282 000004 SELMRR== 4 ;SELECT MAIN RELOC
1283 000004 SELMPM== SELMRR ;SELECT MAIN PROTECT
1284 000006 SELXRR== 6 ;SELECT AUX. RELOC
1285 000006 SELXPM== SELXRR ;SELECT AUX. PROTECT
1286
1287 000005 SELHBA== 5 ;SELECT H-BASE
1288 000007 SELCBA== 7 ;SELECT C-BASE
1289 000010 WE== BIT3 ;DSR WRITE-ENABLE
1290 000014 SETMRR==SELMRR!WE ;SET MAIN-SEG RELOCATION REGISTER TO 0...
;...OR SETMR!<ADDR/2> TO RELOCATE.
1291
1292 000015 SETMPM==15 ;SET MAIN-SEG PROTECTION MASK
1293 000016 SETXRR==SELXRR!WE ;SET AUX.-SEG RELOCATION REGISTER
1294 000017 SETXPM==17 ;SET AUX.-SEG PROTECTION MASK
1295 000013 WRTCSR==SELCSR!WE ;WRITE THE CSR REGISTER
1296 000012 CLFLGS==SELFLG!WE ;CLEAR THE FLAGS REGISTER
1297 000010 SETDSR==SELDSR!WE ; SET REAL DSR.
1298 000013 SETCSR==WRTCSR ; SET THE CSR REGISTER.
1299 000012 SETFLG==CLFLGS ; CLEAR THE FLAGS REGISTER.
1300
1301 ;'WRITE' FUNCTIONS FOR DXR (BITS 15-14):
1302 000000 PIXRBK== 0*BIT14 ;PIXEL READBACK -- <15:14>=00
1303 042000 WRTJSS== 1*BIT14!BIT10 ;WRITE JOYSTICK STATUS REGISTER -- <15:14>=01
1304 100000 RSTPOS== 2*BIT14 ;RESTORE TRUE X-Y POSITION TO DXR,DYR -- <15:14>=10 ;0001
1305 140000 WRTCPX== 3*BIT14 ;WRITE CURSOR POSITION X-COORDINATE -- <15:14>=11
1306 040000 GTJSSW== BIT14 ;GET JOYSTICK SWITCH INTERRUPT COORDINATES
    
```

```

1307
1308 ;'WRITE' FUNCTIONS FOR DYR (BITS 15-14):
1309 ; PIXEL READBCK IS THE SAME - 00
1310 040000 WRTMSR== 1*BIT14 ;WRITE MEMORY STATUS REGISTER BA -- <15:14>=01
1311 100000 SINIT== 2*BIT14 ;PERFORM 'SOFT-INIT' -- <15:14>=10
1312 140000 WRTCPY== 3*BIT14 ;WRITE CURSOR POSITION Y-COORDINATE -- <15:14>=11
1313
1314 ;CSR BIT DEFINITIONS:
1315 000040 FORCSI== BIT5 ;FORCE STOP INTERRUPT
1316 000100 ENECHK== BIT6 ;ENABLE ERROR CHECKING
1317 000200 CHPROT== BIT7 ;CHANNEL PROTECT
1318 ;@@@I
1319 ;FLAGS REGISTER BIT POSITIONS:
1320 000001 PINTRA== BIT0 ;PENDING INTERRUPT, CHANNEL A
1321 000002 PINTRB== BIT1 ;PENDING INTERRUPT, CHANNEL B
1322 000004 PVEC2== BIT2 ;VECTOR BIT 2 FOR INTERRUPT
1323 000010 JSLCKO== BIT3 ;JOYSTICK INTERRUPT LOCKOUT
1324
1325 ; ERROR CODE DEFINITIONS.
1326
1327 100000 ERR== 100000 ; ERROR (COMPOSITE).
1328 104003 NXME== ERR!4000!SELCSR ; NONEXISTANT MEMORY ERROR.
1329 114003 MPE== ERR!14000!SELCSR ; MEMORY PROTECTION ERROR.
1330 124003 RSVGOP== ERR!24000!SELCSR ; RESERVED OPCODE, RESERVED OPERATION.
1331 120003 SEQERR== ERR!20000!SELCSR ; SEQUENCE ERROR.
1332 140003 SYNCTO== ERR!40000!SELCSR ; SYNC TIMEOUT FROM IMAGE MEMORY.
1333 144003 DAVTO== ERR!44000!SELCSR ; DATA AVAILABLE TIMEOUT (PIXEL READBCK).
1334 150003 DRDYTO== ERR!50000!SELCSR ; DATA READY TIMEOUT (JOYSTICK STATUS).
1335
1336 ;DXR, DYR BITS
1337
1338 010000 CHIE ==BIT12 ; CROSSHAIR INTENSITY ENABLE STATUS (1=ON)
1339 010000 JSMIES ==BIT12 ; JOYSTICK MATCH INTERRUPT ENABLE STATE
1340 004000 JSSIES ==BIT11 ; JOYSTICK SWITCH INTERRUPT ENABLE STATE
1341 042000 WJSS ==BIT14!BIT10 ; WRITE JOYSTICK STATUS REGISTER
1342 160000 WECC ==160000 ; WRITE EXTENDED CURSOR CONTROL
1343
1344
1345 ;
1346 ;SOME HANDY -11 OPDEF'S.
1347 ;
1348 000403 SKP3= BR+3 ;SKIP NEXT 3 WORDS.
1349 000402 SKP2= BR+2 ;SKIP NEXT 2 WORDS.
1350 000401 SKP1= BR+1 ;SKIP NEXT WORD.
1351 000400 SKP0= BR+0 ;SAME AS A NOP.
1352
1353 ;
1354 ; SOME GENERAL EQUATES.
1355 ;
1356
1357 000004 ERRVEC==4 ; POINTER TO ERROR VECTOR FOR BUS TIME OUT.

```


1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399

000000

```
.SBTTL SPECIAL MACROS AND OPDEFS.  
EN=0 ; INITIALIZE ERROR NUMBER  
: MACRO TO XCT MEMORY SETUP FUNCTIONS (STAT C INSTRUCTIONS)  
: VAILD FUNCTIONS ARE:  
: PROTEC, READ, WRT, WRT1, RDWRT, RDWRT1, HCOPY.  
: .MACRO SETUP FUNC,N1,N2,N3,N4  
: .IIF NB <N1>, FUNC!CH'N1  
: .IIF NB <N2>, FUNC!CH'N2  
: .IIF NB <N3>, FUNC!CH'N3  
: .IIF NB <N4>, FUNC!CH'N4  
: .ENDM SETUP  
: MACROS FOR ASSEMBLING VECTOR DATA.  
: CALLING ARGUMENTS ARE:  
: A = INTENSIFY BIT, 'I' OR 'U'.  
: B = DELTA X, +/-0 THRU 1776 (76 IF SHORT TYPE).  
: C = DELTA Y, DITTO  
: .MACRO SXY A,B,C ;ASSEMBLE SHORT (1 WORD) DATA.  
: .I = 40000  
: .DX = B*200  
: .DY = C  
: .IIF IDN <U> <A>, .I = 0  
: .IIF LT B, .DX = <-B*200>!20000  
: .IIF LT C, .DY = <-C>!100  
: .WORD .I!.DX!.DY  
: .ENDM SXY  
: .MACRO LXY A,B,C ;ASSEMBLE LONG (2 WORDS) DATA.  
: .I = 40000  
: .DX = B  
: .DY = C  
: .IIF IDN <U> <A>, .I = 0  
: .IIF LT B, .DX = <-B>!20000  
: .IIF LT C, .DY = <-C>!20000  
: .WORD .I!.DX, .DY  
: .ENDM LXY
```

1401
 1402
 1403
 1404
 1405
 1406
 1407
 1408
 1409
 1410
 1411
 1412
 1413
 1414
 1415
 1416
 1417
 1418
 1419
 1420
 1421
 1422
 1423
 1424
 1425
 1426
 1427
 1428
 1429
 1430
 1431
 1432
 1433
 1434
 1435
 1436
 1437
 1438
 1439
 1440
 1441
 1442
 1443
 1444
 1445
 1446
 1447
 1448
 1449
 1450
 1451
 1452
 1453
 1454
 1455
 1456
 1457

```

: MACROS TO STANDARDIZE THE INITIALIZATION OF EACH TEST.
: PROVIDES AN ASCIZ TITLE STRING, AND POINTER FOR
: SUBSEQUENT PRINTING (IF REQ'D) VIA 'PRINTF TNAM,TNUM' CALL.
: ALSO INITS A SEQUENTIAL ERROR NUMBER SEQUENCE, AND
: SETS AN ITERATION COUNT FOR THE CURRENT TEST.
:
: .MACRO BEGIN.TEST TEXT,LK,?TAG1,?TAG2
.RADIX 10
    TSTITLE \TN+1,<TEXT>,<LK>
.RADIX
.NLIST
.LIST MC
.LIST
    BGNTST
    TN=T$TESTNUM           ;TEST NUMBER
    EN=TN*100.             ;INIT ERROR NUMBER SEQUENCE.
    .IF B <LK>, MOV #1,LOOPK ;DEFAULT = 1.
    .IF NB <LK>, MOV #LK,LOOPK
    MOV #TN,TNUM           ;SET TEST NUMBER...
    MOV #TAG1,TNAM        ;...AND NAME POINTER.
    BR TAG2               ;SKIP OVER THE ASCII.
:TAG1: .ASCIZ \N%ATEST %D2%A: TEXT\
TAG1: .ASCIZ \A TEXT\
    .EVEN
TAG2:
    .ENDM BEGIN.TEST

:
: THIS MACRO IS USED BY MACRO 'BEGIN.TEST' MACRO, ABOVE.
:
: .MACRO TSTITLE A,ASCII,I
.SBTTL *** TEST 'A' ASCII
.NLIST
.LIST ME
.LIST

:*****
: *
: * BEGIN TEST 'A' - ASCII
: *
:*****

: (ITERATION COUNT = I)

.NLIST
.NLIST ME
.LIST
    .ENDM TSTITLE

:
: MACROS TO STANDARDIZE THE END OF EACH TEST.
:
: .MACRO END.TEST
.NLIST
.RADIX 10
    
```

1458
 1459
 1460
 1461
 1462
 1463
 1464
 1465
 1466
 1467
 1468
 1469
 1470
 1471
 1472
 1473
 1474
 1475
 1476
 1477
 1478
 1479
 1480
 1481
 1482
 1483
 1484
 1485
 1486
 1487
 1488
 1489
 1490
 1491
 1492
 1493
 1494
 1495
 1496
 1497
 1498
 1499
 1500
 1501
 1502
 1503
 1504
 1505
 1506
 1507
 1508
 1509
 1510
 1511
 1512
 1513
 1514

```

.NLIST MC
.LIST ME
.LIST
    ENDTN  \TN
.NLIST
.RADIX
.NLIST ME
.LIST
    ENDTST
    .ENDM  END.TEST

:
: THIS MACRO IS USED BY 'END.TEST' MACRO, ABOVE.
:
    .MACRO  ENDTN  A
:
:*****
: *
: *   END TEST 'A'
: *
:*****
    .ENDM  ENDTN

:
: MACROS TO DEFINE SEQUENTIAL ERROR NUMBERS FOR
: SUBSEQUENT SUPERVISOR ERROR CALLS.
: REQUIRES PRIOR USE OF 'BEGIN.TEST' MACRO.
:
    .MACRO  SFERR  ADDR,PNTR          ; SYSTEM FATAL.
    EN=EN+1
    JSR    PC,INCERK
    ERRSF  EN,ADDR,PNTR
    JSR    PC,CKDROP
    .ENDM  SFERR

    .MACRO  DFERR  ADDR,PNTR,CK      ; DEVICE FATAL.
    EN=EN+1
    JSR    PC,INCERK
    ERRDF  EN,ADDR,PNTR
    JSR    PC,CKDROP
    .IF    NB,CK
    CKLOOP
    .ENDC
    .ENDM  DFERR

    .MACRO  SFTERR ADDR,PNTR,CK      ; SOFT ERROR.
    EN=EN+1
    JSR    PC,INCERK
    ERRSOF EN,ADDR,PNTR
    JSR    PC,CKEMAX
    .IF    NB,CK
    CKLOOP
    .ENDC
    .ENDM  SFTERR

    .MACRO  HRDERR ADDR,PNTR,CK      ; HARD ERROR.

```

```

1515 .NLIST
1516 .NLIST ME
1517 .LIST
1518 EN=EN+1
1519 JSR PC,INCERK
1520 ERRHRD EN,ADDR,PNTR
1521 JSR PC,CKEMAX
1522 .IF NB,CK
1523 CKLOOP
1524 .ENDC
1525 .ENDM HRDERR
1526
1527
1528
1529 : MACRO - DO SOME COMMON STUFF AT THE BEGINNING OF A TEST.
1530 :
1531 : .MACRO COMBEG
1532 .NLIST
1533 .LIST ME
1534 .LIST
1535 JSR PC,TSTGO ; TITLE.
1536 JSR PC,DRESET ; DO SOFT INIT.
1537 .NLIST ME
1538 .ENDM COMBEG
1539
1540 :
1541 : MACRO - DO SOME COMMON STUFF AT THE END OF A TEST.
1542 :
1543 : .MACRO COMEND TAG,?L,?X
1544 .NLIST
1545 .LIST ME
1546 .LIST
1547 JSR PC,LOOP ; REPEAT 'TIL LOOPER EXPIRES.
1548 BCS L
1549 BR X
1550
1551 L: JMP TAG
1552
1553 X: JSR PC,TSTEND ; PRINT ERROR SUMMARY, IF REQ'D.
1554 .NLIST
1555 .NLIST ME
1556 .LIST
1557 .ENDM COMEND
1558
1559 :
1560 : MACRO TO CONTROL ITERATION LOOPS.
1561 :
1562 : .MACRO LOOPTO TAG
1563 .NLIST
1564 .LIST ME
1565 .LIST
1566 JSR PC,LOOP
1567 BCS TAG
1568 .NLIST
1569 .NLIST ME
1570 .LIST
1571 .ENDM LOOPTO
    
```

1572
 1573
 1574
 1575
 1576
 1577
 1578
 1579
 1580
 1581
 1582
 1583
 1584
 1585
 1586
 1587
 1588
 1589
 1590
 1591
 1592
 1593
 1594
 1595
 1596
 1597
 1598
 1599
 1600
 1601
 1602
 1603
 1604
 1605
 1606
 1607
 1608
 1609
 1610
 1611
 1612
 1613
 1614
 1615
 1616
 1617
 1618
 1619
 1620
 1621
 1622
 1623
 1624
 1625
 1626
 1627
 1628

```

: MACRO - WRITE FROM SPEC'D DATA SOURCE INTO SPEC'D DSR ACCESS REGISTER.
: ONLY DSR, FLG, CSR, MRR, MPM, XRR, XPM ARE VALID ARGUMENTS FOR 'DSRREG'.
: RO=WORK.
:
: .MACRO WTDSRA DATSRC,DSRREG
.NLIST
.LIST MEB
.LIST
: .NTYPE .NT,DATSRC
: .IF EQ,.NT&^077-^027
MOV DATSRC&^C17!SET'DSRREG,@DSR
: .IFF
MOV #SET'DSRREG',RO ; WRITE DATSRC INTO DSRREG .
BIS DATSRC,RO
MOV RO,@DSR
: .ENDC
.NLIST
.NLIST MEB
.LIST
: .ENDM WTDSRA

:
: MACRO - READ FROM SPEC'D DSR ACCESS REGISTER INTO SPEC'D DATA DESTINATION.
: ONLY DSR, PCS, FLG, CSR, MRR, MPM, XRR, XPM, HBA, CBA ARE VALID
: 'DSRREG' ARGUMENTS.
: RO=WORK.
:
: .MACRO RDDSRA DSRREG,DATDST
.NLIST
.LIST ME
.LIST
MOV #SEL'DSRREG',@DSR ; READ DSRREG INTO DATDST .
MOV @DSR,DATDST
.NLIST
.NLIST ME
.LIST
: .ENDM RDDSRA

:
: MACRO - IF R1=R2, BRANCH TO OK, ELSE CALL HARDWARE ERROR.
:
: .MACRO IFERROR ARG1,ARG2,?OK$,CK
.NLIST
.LIST ME
.LIST
CMP R1,R2 ; OK?
BEQ OK$ ; YES.
HRDERR ARG1,ARG2,CK ; NO.
.NLIST
.LIST ME
.LIST
OK$:
.NLIST ME
: .ENDM IFERROR
    
```

1629
 1630
 1631
 1632
 1633
 1634
 1635
 1636
 1637
 1638
 1639
 1640
 1641
 1642
 1643
 1644
 1645
 1646
 1647
 1648
 1649
 1650
 1651
 1652
 1653
 1654
 1655
 1656
 1657
 1658
 1659
 1660
 1661
 1662
 1663
 1664
 1665
 1666
 1667
 1668
 1669
 1670
 1671
 1672
 1673
 1674
 1675
 1676
 1677
 1678
 1679
 1680
 1681
 1682
 1683
 1684
 1685

```

: MACRO - IF R1=R2 AND R3=R4, BRANCH TO OK, ELSE CALL HARDWARE ERROR.
:
: .MACRO IFERRX2 ARG1,ARG2,?OK$,?ERR,CK
.NLIST
.LIST ME
.LIST
        CMP      R1,R2          : 1ST ITEM OK?
        BNE      ERR           : NO.
        CMP      R3,R4          : 2ND ITEM OK?
        BEQ      OK$           : YES.
ERR:     HRDERR ARG1,ARG2,CK   : NO.
.NLIST
.LIST ME
.LIST
OK$:
.NLIST ME
        .ENDM IFERRX2

:
: MACRO - SOFTWARE INITIALIZE THE DPU. THIS DIFFERS FROM 'DPINIT'
: IN TWO WAYS. FIRST, IT IS ASSUMED THAT THE DPU IS STOPPED AND
: NOT HUNG UP. SECOND, NO CHECKING IS DONE BY THIS CODE.
:
: .MACRO INITDP
.NLIST
.LIST ME
.LIST
        JSR      PC,DPINIT      ; GO DO SOFT INIT, CHECK THE STATE.
.NLIST
.NLIST ME
.LIST
        .ENDM INITDP

:
: .MACRO DPSTART ADRS
.NLIST
.LIST ME
.LIST
        MOV      ADRS,@DPC      ; START THE DPU.
        JSR      PC,WAITF       ; WAIT FOR DISPLAY STOP.
.NLIST
.NLIST ME
.LIST
        .ENDM DPSTART

:
: .MACRO DPCONT
.NLIST
.LIST ME
.LIST
        BIS      #BIT0,@DPC     ; CONTINUE THE DPU.
        JSR      PC,WAITF       ; WAIT FOR DISPLAY STOP.
.NLIST
.NLIST ME
.LIST
    
```



```

1686             .ENDM   DPCONT
1687
1688 ;MACRO TO DO A 3-OPERAND BIT-SET:
1689             .MACRO   BISW3  S1,S2,DST
1690             .NLIST
1691             .LIST  MEB
1692             .LIST
1693             .NTYPE   .NT,DST
1694             .IF      EQ, .NT&^070
1695             MOV      S2,DST
1696             BIS      S1,DST
1697             .IFF
1698             MOV      S2,-(SP)
1699             BIS      S1,(SP)
1700             MOV      (SP)+,DST
1701             .ENDC
1702             .NLIST
1703             .LIST  MEB
1704             .LIST
1705             .ENDM   BISW3
1706
1707 ;MACRO TO DO A 3-OPERAND BIT-CLEAR:
1708             .MACRO   BICW3  S1,S2,DST
1709             .NLIST
1710             .LIST  MEB
1711             .LIST
1712             .NTYPE   .NT,DST
1713             .IF      EQ, .NT&^070
1714             MOV      S2,DST
1715             BIC      S1,DST
1716             .IFF
1717             MOV      S2,-(SP)
1718             BIC      S1,(SP)
1719             MOV      (SP)+,DST
1720             .ENDC
1721             .NLIST
1722             .LIST  MEB
1723             .LIST
1724             .ENDM   BICW3
1725
1726 ;MACRO TO SET UP THE INTERRUPT MASK AND PRIME THE INTERRUPT FLAG:
1727             .MACRO   INTSET  IEX
1728             .NLIST
1729             .LIST  MEB
1730             .LIST  ME
1731             .LIST
1732             .IF      NB,IEX
1733             MOV      #^0177400!IOKCKIN!IOK'IEX,INTMASK      ; PRIME FLAG, EXPECT IEX
1734             .IFF
1735             MOV      #^0177400!IOKCKIN,INTMASK              ; PRIME FLAG, NO INTR. EXPECTED.
1736             .ENDC
1737             .NLIST
1738             .LIST  MEB
1739             .LIST
1740             .ENDM   INTSET
1741
1742
    
```

1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756

```

: MACRO TO FORCE IN-LINE RETURN FROM REAL OR IMAGINED INTERUPT.
:
: .MACRO FORRTI
.NLIST
.LIST ME
.LIST
      MOV      #.+6,(SP)      ; FORCE RTI RETURN TO -
      RTI                               ; - NEXT LOC.
.NLIST
.NLIST ME
.LIST
      .ENDM  FORRTI

```

```
1758      .SBTTL  USER PATCHABLE FLAGS
1759
1760      ;
1761      ; THE FOLLOWING LOCATIONS MAY BE PATCHED BY THE USER
1762      ; TO OBTAIN THE RESULTS DESCRIBED FOR EACH.
1763      ;
1764
1765 003020 000000  FORCER: 0      ; FORCE TYPE ALL HARD ERRORS (THE ONES CALLED -
1766                                     ; - BY THE MACRO 'IFERROR'). AN ERROR NEED NOT -
1767                                     ; - EXIST, JUST ASSUME AND TYPE THE MESSAGE.
1768
1769
1770
1771 003022 001700  MFGMO: 1700  ; MANUFACTURING MEMORY 0 MASTER SPECIFICATION.
1772                                     ; IN MANUFACTURING MODE, MEMORY 0 IS COMPARED
1773                                     ; WITH THIS VALUE WHEN SYSTEM CONFIGURATION IS
1774                                     ; EXECUTED. DISCREPANCY IS REPORTED.
1775                                     ; (CURRENTLY SHOWN FOR 4 BIT MEMORY.)
1776
1777 003024 120000  MFGSO: 120000 ; MANUFACTURING SYNC CHAN 0 MASTER SPECIFICATION.
1778                                     ; IN MANUFACTURING MODE, SYNC CHAN 0 IS COMPARED
1779                                     ; WITH THIS VALUE WHEN SYSTEM CONFIGURATION IS
1780                                     ; EXECUTED. DISCREPANCY IS REPORTED.
1781                                     ; (CURRENTLY SHOWN FOR INTERLACED SYNC CHAN.)
```

1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794 003026 000000
1795 003030 000000
1796
1797
1798
1799
1800 003032 172010
1801 003034
1802 003034 172012
1803 003036 172014
1804 003040 172016
1805
1806 003042 172020
1807 003044 172022
1808 003046 172024
1809
1810 003050 000200
1811 003052 000320
1812 003054 000324
1813 003056 000330
1814 003060 000334
1815 003062 000340
1816
1817 003064 040000
1818
1819
1820
1821
1822
1823
1824 003066 000004
1825 003070 000100
1826 003072 001700
1827 003074 176077
1828
1829
1830
1831
1832
1833
1834

```
.SBTTL GLOBAL DATA SECTION

:++
: THE GLOBAL DATA SECTION CONTAINS DATA THAT ARE USED
: IN MORE THAN ONE TEST.
:--

:
: THE FOLLOWING DATA ARE SET FOR EACH UNIT AT INIT TIME.
: SINGLE UNIT DEFAULTS (LISTED) ARE IN THE DEFAULT P-TABLE.
:
UNITN:: 0          ;UNIT # UNDER TEST.
QVP:: 0           ;QUICK VERIFY FLAG.

:
: DEVICE REGISTER ADDRESSES *** KEEP IN THE ORDER GIVEN ***
:
DPC:: 172010      ;DISPLAY PC
DRR::
DSR:: 172012      ;DISPLAY STATUS REGISTER (ALSO REL REG).
DXR:: 172014      ;DISPLAY X POSITION REGISTER.
DYSR:: 172016     ;DISPLAY Y POSITION REGISTER.

LSR:: 172020      ; LOOK-UP-RAM STATUS.
LDR:: 172022      ;          DATA.
LMR:: 172024      ;          AND MAINT REGISTERS.

DPRI:: PRI04      ;INTERRUPT PRIORITY.
STPV:: 320        ;STOP VECTOR
JSMV:: 324        ;JOY-STICK MATCH VECTOR.
TOTV:: 330        ;TIME-OUT VECTOR
JSSV:: 334        ;JOY-STICK SWITCH (DEMAND) VECTOR.
LUTV:: 340        ; LUT READ/WRITE DONE VECTOR.

OPTI:: I          ;OPTIONAL 'INTENSIFY' BIT FOR DPU...
: ...IF DPUMOD = 0, OPTI = I(INTENSIFY).
: ...OTHERWISE, OPTI = 0 (UNINTENSIFY).

:
: AND THESE ARE SET BY THE IMAGE MEMORY SIZER.
:
PDBITS:: 4        ;NUMBER OF PIXEL BITS (2, 4, 6, OR 8).
PDI:: 1*BIT6      ;PIXEL DATA INCREMENT (MIN INTENSITY).
PDF:: 17*BIT6     ;PIXEL DATA FIELD (MAX INTENSITY).
PDM:: ^C<17*BIT6> ;PIXEL DATA FIELD MASK.

:
: STANDARD CONFIGURATIONS
:
: BITS  PDI  PDF
: 2     0400 1400
: 4     0100 1700
: 6     0020 1760
: 8     0004 1774
```

```

1836
1837      ; STORAGE FOR DEVICE REGISTERS
1838
1839 003076 000000 100000 000000 DUMMY: 0,100000,0,0 ;DUMMY DEVICE REGISTERS...
1840 003106 000000 000000 000000      0,0,0,0,0,0,0,0 ;...FOR MULTI-UNIT CHECKOUT.
1841
1842
1843 003126 000000 IDPC:: 0 ;SAVED DPC ON ANY INTERRUPT.
1844 003130 000000 IDSR:: 0 ;DITTO DSR
1845 003132 000000 IDXR:: 0 ;DITTO DXR
1846 003134 000000 IDYR:: 0 ;DITTO DYR
1847 003136 000000 ILSR:: 0 ;DITTO LUT CSR
1848 003140 000000 ILDR:: 0 ;DITTO LUT DATA
1849 003142 000000 ILMR:: 0 ;DITTO LUT MAINT.
1850
1851      ;SAVED "INTERNAL" REGISTERS ON AN INTERRUPT (IN ORDER OF SELECT CODE):
1852 003144 000000 ISDSR:: 0 ;SAVED "SELECTED" DSR
1853 003146 000000 IPCSAV:: 0 ;SAVED PCSAVE
1854 003150 000000 IFLAGS:: 0 ;SAVED FLAGS
1855 003152 000000 ICSR:: 0 ;SAVED CSR
1856 003154 000000 IMAIN:: 0 ;SAVED MAIN SEG. MMGT.
1857 003156 000000 IHBASE:: 0 ;SAVE HISTOGRAM BASE
1858 003160 000000 IAUX:: 0 ;SAVED AUX. SEG MMGT.
1859 003162 000000 ICBASE:: 0 ;SAVED CHARACTER BASE
1860
1861 003164 000000 SDPC:: 0 ;SAVED DPC AT ANY OTHER TIME.
1862 003166 000000 SDSR:: 0 ;DITTO DSR
1863 003170 000000 SDXR:: 0 ;DITTO DXR
1864 003172 000000 SDYR:: 0 ;DITTO DYR
1865
1866 003174 000000 ;SAVED INTERNAL REGISTERS AT OTHER TIMES
1867 003176 000000 SSSDR:: 0 ;SAVED "SELECTED" DSR
1868 003200 000000 SPCSAV:: 0 ;SAVED PCSAVE
1869 003202 000000 SFLAGS:: 0 ;SAVED FLAGS
1870 003204 000000 SCSR:: 0 ;SAVED CSR
1871 003206 000000 SMAIN:: 0 ;SAVED MAIN SEG. MMGT.
1872 003210 000000 SHBASE:: 0 ;SAVE HISTOGRAM BASE
1873 003212 000000 SAUX:: 0 ;SAVED AUX. SEG MMGT.
1874 SCBASE:: 0 ;SAVED CHARACTER BASE
1875
1876      ; AND FINALLY SOME MISCELLANEOUS REGISTERS.
1877
1877 003214 000000 SCFLG:: 0 ; SYS CONFIG FLAG (0=1ST TIME, NZ=SUBSEQUENT).
1878 003216 000000 LUTAV:: 0 ; LUT AVAILABLE FLAG (NZ = USE LUT).
1879 003220 000000 SHADLY:: 0 ; SHADING DELAY COUNTER (OPTIONAL).
1880 003222 110224 HUE:: NTSC8+2 ; CURRENT COLOR POINTER (OPTIONAL).
1881
1882      ;STIK:: .BYTE -1, -1 ; 8 JOY-STICK AVAILABLE FLAGS...
1883      ; .BYTE -1, -1 ;... -1 = I DUNNO...
1884      ; .BYTE -1, -1 ;... 0 = NO...
1885      ; .BYTE -1, -1 ;... +1 = YES.
1886
1887 003224 000000 DUFLG:: 0 ;'DROPPED UNIT' FLAG. INHIBITS DPU...
1888      ;...CODE IN 'CLEAN-UP'.
1889 003226 000000 NODEV:: 0 ;FLAG TO SAY NO DEVICE.
1890
1891 003230 000001 INTLAC:: 1 ; INTERLACED MODE
1892 003232 000000 TEMP1:: 0 ;SOME TEMP LOCATIONS.
    
```

```

GLOBAL DATA SECTION

1893 003234 000000      TEMP2:: 0
1894 003236 000000      XXCOMM:: 0
1895 003240 000000      FREE:: 0
1896 003242 000000      FRESIZ:: 0
1897 003244 000000      KTFLG:: 0
1898
1899
1900 003246 000074      HZ:: 60.
1901 003250 000000      YMAX:: 0
1902 003252 000000      MEMFLG:: 0
1903 003254 000000      SYCFLG:: 0
1904 003256
1905 003256
1906 003256 000000      CTAB::
1907 003260 000000      MEMTAB::
1908 003262 000000      CTABM:: 0
1909 003264 000000      0
1910 003266 177777      0
1911 003270
1912 003270 000000      SYCTAB::
1913 003272 000000      CTABS:: 0
1914 003274 000000      0
1915 003276 000000      0
1916 003300 177777      0
1917 003302
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928 003302
1929 003502 000000      ERTABL: .BLKW 64.
                                ERTABE: 0

                                ; XXDP+ COMM BLOCK POINTER.
                                ; 1ST FREE MEMORY ADDRESS...
                                ; ...AND SIZE (IN WORDS).
                                ; KT11, MEM AVAIL FLAG -
                                ; - 0 = <24K OR NO KT -
                                ; - NZ = >24K AND KT.
                                ; LINE FREQUENCY.
                                ; MAX Y ACCORDING TO LINE FREQUENCY.
                                ; MEM FLAGS (1'S IN BITS 0-3 REFLECT MEMS AVAIL).
                                ; SYNC CHAN FLAGS (1'S IN BITS 0-3 REFLECT SYNC CHANS AVAIL)
                                ; CONFIGURATION TABLES.
                                ; MEMORY CONFIG WORK.
                                ; END OF MEM TABLE.
                                ; SYNC CHAN CONFIG WORK.
                                ; END OF SYNC CHAN TABLE.
                                ; END OF CONFIG WORK AREA.

                                ;ERROR STATISTICS TABLE (1 WORD PER UNIT), 64 UNITS MAX:
                                :
                                :
                                : 0 = UNIT NOT TESTED
                                : 100000 = UNIT ONLINE, NO ERRORS
                                : 10XXXX = UNIT ONLINE, ENCOUNTERED XXXX ERRORS
                                : 160000 = UNIT DROPPED, NON-EXISTENT DEVICE REGISTER
                                : 160001 = UNIT DROPPED, NOT IDLE AT START
                                : 14XXXX = UNIT DROPPED, ENCOUNTERED XXXX ERRORS

```

1931
 1932
 1933
 1934
 1935
 1936
 1937
 1938
 1939 003504 040 040
 1940 003543 045 101
 1941 003564 045 101
 1942 003622 045 101
 1943 003657 045 101
 1944 003715 045 101
 1945 003752 045 116
 1946 003756 040 040
 1947 004014 040 040
 1948 004066 045 116
 1949 004072 040 040
 1950 004135 040 040
 1951 004206 045 116
 1952 004212 040 040
 1953 004251 045 116
 1954 004255 040 040
 1955 004315 045 116
 1956 004321 040 040
 1957 004356 040 040
 1958 004400 045 101
 1959 004434 040 040
 1960 004503 040 040
 1961 004534 040 040
 1962 004566 040 040
 1963 004612 040 040
 1964 004642 040 040
 1965 004676 040 040
 1966 004724 045 116
 1967 004746 040 040
 1968 004773 045 116
 1969 005047 040 040
 1970 005073 040 040
 1971 005121 045 116
 1972 005175 040 040
 1973 005221 040 040
 1974 005250 045 116
 1975 005314 040 040
 1976 005344 040 040
 1977 005374 045 116
 1978 005431 040 040
 1979 005453 045 116
 1980 005475 040 040
 1981 005521 045 116
 1982 005556 040 040
 1983 005574 040 040
 1984 005641 040 040
 1985 005677 045 101
 1986 005733 045 116
 1987 006010 045 116

.SBTTL GLOBAL TEXT SECTION

:+
 : THE GLOBAL TEXT SECTION CONTAINS FORMAT STATEMENTS,
 : MESSAGES, AND ASCII INFORMATION THAT ARE USED IN
 : MORE THAN ONE TEST.
 :--

116 NXR: .ASCIZ / NON-EXISTANT DEVICE REGISTER/
 040 NXR: .ASCIZ /%A ADDRESS: %06/
 040 PCSX: .ASCII /%A DPC,DSR EXP'D: %06%,%06%/

040 .ASCIZ /%A DPC,DSR REC'D: %06%,%06/
 040 XYX: .ASCII /%A DXR,DYR EXP'D: %06%,%06%/

040 .ASCIZ /%A DXR,DYR REC'D: %06%,%06/
 045 FUSI: .ASCII /%N%/

125 USI: .ASCIZ / UNEXPECTED 'STOP' INTERRUPT/
 042 NSI: .ASCIZ / 'STOP' INTERRUPT EXPECTED, NOT RECEIVED/
 045 FUTO: .ASCII /%N%/

125 UTO: .ASCIZ / UNEXPECTED DPU 'ERROR' INTERRUPT/
 104 NTO: .ASCIZ / DPU 'ERROR' INTERRUPT EXP'D, NOT REC'D/
 045 FUMI: .ASCII /%N%/

125 UMI: .ASCIZ / UNEXPECTED 'MATCH' INTERRUPT/
 045 FUSWI: .ASCII /%N%/

125 USWI: .ASCIZ / UNEXPECTED 'SWITCH' INTERRUPT/
 045 FNOINTR: .ASCII /%N%/

116 NOINTR: .ASCIZ / NO INTERRUPT WAS GENERATED/
 111 IFAULT: .ASCIZ / INTERRUPT FAULT/
 040 INTX: .ASCIZ /%A CPU PC: %06% DPC: %06/
 042 NOINIT: .ASCIZ / 'BUS-INIT' DIDN'T INITIALIZE DISPLAY/
 123 SSF: .ASCIZ / START-NOP-STOP FAILURE/
 122 RJF: .ASCIZ / RESUME-NOP-JUMP FAILURE/
 112 JSF: .ASCIZ / JUMP SELF FAILURE/
 105 ESF: .ASCIZ / EXTERNAL STOP FAILURE/
 120 PDXI: .ASCIZ / PIXEL DATA XFER INCORRECT/
 116 OPCF: .ASCIZ / NULL OPCODE FAILURE/
 045 OPCFX: .ASCIZ /%N% OPCODE: %06/
 122 RELF: .ASCIZ / RELOCATION FAILURE/
 045 RELFX: .ASCIZ /%N% INITIAL DPC: %06%, RELOC(DSR): %06%/

101 APF: .ASCIZ / ABS POINT FAILURE/
 114 LVF: .ASCIZ / LONG VECTOR FAILURE/
 045 LONGX: .ASCIZ /%N% ORIGIN: %04%,%04% DX,DY: %06%,%06/
 122 RPF: .ASCIZ / REL POINT FAILURE/
 123 SVF: .ASCIZ / SHORT VECTOR FAILURE/
 045 SHRTX: .ASCIZ /%N% ORIGIN: %04%,%04% DXY: %06/
 107 GHXF: .ASCIZ / GRAPH-HISTO X FAILURE/
 107 GHYF: .ASCIZ / GRAPH-HISTO Y FAILURE/
 045 GHXYX: .ASCIZ /%N% AMPL: %06% INCR: %02/
 102 BMF: .ASCIZ / BIT MAP FAILURE/
 045 BMX: .ASCIZ /%N% OPCODE: %06/
 103 CHR: .ASCIZ / CHAR MODE FAILURE/
 045 CHRFX: .ASCIZ /%N% BASE: %06% CHAR: %03/
 124 PRBHNG: .ASCII / TEST ABORTED/
 120 PRBH: .ASCIZ / PIXEL-READ-BACK FAILS OR HANGS DPU/
 111 IMDI: .ASCIZ / IMAGE MEMORY DATA INCORRECT/
 040 IMDIX: .ASCII /%A ADDRESS (X,Y): %04%,%04/
 045 .ASCIZ /%N% FIELD: %04% EXP'D: %04% REC'D: %04/
 045 IMDIX2: .ASCIZ /%N% SUSPECT RAM GROUP %01%, IMAGE MEMORY CHANNEL: %01/

1988	006101	045	116	045	IMDIXC:	.ASCII	/%NZA ADDRESS (X,Y): %04ZA,%04/
1989	006137	045	116	045		.ASCIZ	/%NZA FIELD: %04ZA EXP'D: %04ZA REC'D: %04/
1990	006214	045	116	045		.ASCIZ	/%NZA SUSPECT RAM GROUP %01ZA, IMAGE MEMORY CHANNEL: %01/
1991	006306	040	040	114	NLRI:	.ASCIZ	/ LUT READ INTERRUPT EXPECTED, NOT RECEIVED/
1992	006362	040	040	114	NLWI:	.ASCIZ	/ LUT WRITE INTERRUPT EXPECTED, NOT RECEIVED/
1993	006437	040	040	125	ULRI:	.ASCIZ	/ UNEXPECTED LUT READ INTERRUPT/
1994	006477	040	040	125	ULWI:	.ASCIZ	/ UNEXPECTED LUT WRITE INTERRUPT/
1995	006540	045	101	040	LINTX:	.ASCIZ	/%A CSR: %06/
1996	006555	040	040	114	LAIER:	.ASCIZ	/ LUT AUTO-INCR INCORRECT ON READ/
1997	006617	040	040	114	LAI EW:	.ASCIZ	/ LUT AUTO-INCR INCORRECT ON WRITE/
1998	006662	040	040	114	LDINC:	.ASCIZ	/ LUT DATA INCORRECT/
1999	006707	045	101	040	LRWX:	.ASCIZ	/%A CSR: %06ZA EXP'D: %06ZA REC'D: %06/
2000	006760	040	040	042	NSINIT:	.ASCIZ	/ 'SOFT-INIT' DIDN'T INITIALIZE THE DPU/
2001	007030	040	040	042	BRINIT:	.ASCIZ	/ 'BUS-RESET' DIDN'T INITIALIZE THE DPU/
2002	007100	040	040	104	IDPCE:	.ASCIZ	# DPC READ/WRITE ERROR#
2003	007127	040	040	104	IDSRE:	.ASCIZ	# DSR READ/WRITE ERROR#
2004	007156	040	040	103	ICSRE:	.ASCIZ	# CSR READ/WRITE ERROR#
2005	007205	040	040	115	IMRRE:	.ASCIZ	# MRR READ/WRITE ERROR#
2006	007234	040	040	115	IMPWPE:	.ASCIZ	# MPM WRITE/MWP (HBASE) READ ERROR#
2007	007277	040	040	115	IMPME:	.ASCIZ	# MPM READ/WRITE ERROR#
2008	007326	040	040	130	IXRRE:	.ASCIZ	# XRR READ/WRITE ERROR#
2009	007355	040	040	130	IXPME:	.ASCIZ	# XPM READ/WRITE ERROR#
2010	007404	040	040	130	IXPWPE:	.ASCIZ	# XPM WRITE/XWP (HBASE) READ ERROR#
2011	007447	040	040	110	IHBAE:	.ASCIZ	# HBA READ/WRITE ERROR#
2012	007476	040	040	103	ICBAE:	.ASCIZ	# CBA READ/WRITE ERROR#
2013	007525	040	040	104	JDPDCE:	.ASCIZ	# DJMS/DPOP FAILURE (DPC)#
2014	007557	040	040	104	JPPCSE:	.ASCIZ	# DJMS/DPOP FAILURE (PCS)#
2015	007611	040	040	104	DPDSE:	.ASCIZ	/ DPC(W),DSR(R) DUAL ADRS FAULT/
2016	007651	040	040	104	DSDPE:	.ASCIZ	/ DSR(W),DPC(R) DUAL ADRS FAULT/
2017	007711	040	040	104	DSDXE:	.ASCIZ	/ DSR(W),DXR(R) DUAL ADRS FAULT/
2018	007751	040	040	104	DSDYE:	.ASCIZ	/ DSR(W),DYR(R) DUAL ADRS FAULT/
2019	010011	040	040	104	DSPCE:	.ASCIZ	/ DSR(W),PCS(R) DUAL ADRS FAULT/
2020	010051	040	040	104	DSFLE:	.ASCIZ	/ DSR(W),FLG(R) DUAL ADRS FAULT/
2021	010111	040	040	104	DSCSE:	.ASCIZ	/ DSR(W),CSR(R) DUAL ADRS FAULT/
2022	010151	040	040	104	DSMRE:	.ASCIZ	/ DSR(W),MRR(R) DUAL ADRS FAULT/
2023	010211	040	040	104	DSMPE:	.ASCIZ	/ DSR(W),MPM(R) DUAL ADRS FAULT/
2024	010251	040	040	104	DSXRE:	.ASCIZ	/ DSR(W),XRR(R) DUAL ADRS FAULT/
2025	010311	040	040	104	DSXPE:	.ASCIZ	/ DSR(W),XPM(R) DUAL ADRS FAULT/
2026	010351	040	040	103	CSDSE:	.ASCIZ	/ CSR(W),DSR(R) DUAL ADRS FAULT/
2027	010411	040	040	103	CSPCE:	.ASCIZ	/ CSR(W),PCS(R) DUAL ADRS FAULT/
2028	010451	040	040	103	CSFLE:	.ASCIZ	/ CSR(W),FLG(R) DUAL ADRS FAULT/
2029	010511	040	040	103	CSMRE:	.ASCIZ	/ CSR(W),MRR(R) DUAL ADRS FAULT/
2030	010551	040	040	103	CSMPE:	.ASCIZ	/ CSR(W),MPM(R) DUAL ADRS FAULT/
2031	010611	040	040	103	CSXRE:	.ASCIZ	/ CSR(W),XRR(R) DUAL ADRS FAULT/
2032	010651	040	040	103	CSXPE:	.ASCIZ	/ CSR(W),XPM(R) DUAL ADRS FAULT/
2033	010711	040	040	115	MRDSE:	.ASCIZ	/ MRR(W),DSR(R) DUAL ADRS FAULT/
2034	010751	040	040	115	MRPCE:	.ASCIZ	/ MRR(W),PCS(R) DUAL ADRS FAULT/
2035	011011	040	040	115	MRFLE:	.ASCIZ	/ MRR(W),FLG(R) DUAL ADRS FAULT/
2036	011051	040	040	115	MRCSE:	.ASCIZ	/ MRR(W),CSR(R) DUAL ADRS FAULT/
2037	011111	040	040	115	MRMPE:	.ASCIZ	/ MRR(W),MPM(R) DUAL ADRS FAULT/
2038	011151	040	040	115	MRXRE:	.ASCIZ	/ MRR(W),XRR(R) DUAL ADRS FAULT/
2039	011211	040	040	115	MRXPE:	.ASCIZ	/ MRR(W),XPM(R) DUAL ADRS FAULT/
2040	011251	040	040	115	MPDSE:	.ASCIZ	/ MPM(W),DSR(R) DUAL ADRS FAULT/
2041	011311	040	040	115	MPPCE:	.ASCIZ	/ MPM(W),PCS(R) DUAL ADRS FAULT/
2042	011351	040	040	115	MPFLE:	.ASCIZ	/ MPM(W),FLG(R) DUAL ADRS FAULT/
2043	011411	040	040	115	MPCSE:	.ASCIZ	/ MPM(W),CSR(R) DUAL ADRS FAULT/
2044	011451	040	040	115	MPMRE:	.ASCIZ	/ MPM(W),MRR(R) DUAL ADRS FAULT/

2045	011511	040	040	115	MPXRE: .ASCIZ / MPM(W),XRR(R) DUAL ADRS FAULT/
2046	011551	040	040	115	MPXPE: .ASCIZ / MPM(W),XPM(R) DUAL ADRS FAULT/
2047	011611	040	040	130	XRDSE: .ASCIZ / XRR(W),DSR(R) DUAL ADRS FAULT/
2048	011651	040	040	130	XRPCE: .ASCIZ / XRR(W),PCS(R) DUAL ADRS FAULT/
2049	011711	040	040	130	XRFLE: .ASCIZ / XRR(W),FLG(R) DUAL ADRS FAULT/
2050	011751	040	040	130	XRCSE: .ASCIZ / XRR(W),CSR(R) DUAL ADRS FAULT/
2051	012011	040	040	130	XRMRE: .ASCIZ / XRR(W),MRR(R) DUAL ADRS FAULT/
2052	012051	040	040	130	XRMPE: .ASCIZ / XRR(W),MPM(R) DUAL ADRS FAULT/
2053	012111	040	040	130	XRXPE: .ASCIZ / XRR(W),XPM(R) DUAL ADRS FAULT/
2054	012151	040	040	130	XPDSE: .ASCIZ / XPM(W),DSR(R) DUAL ADRS FAULT/
2055	012211	040	040	130	XPPCE: .ASCIZ / XPM(W),PCS(R) DUAL ADRS FAULT/
2056	012251	040	040	130	XPFLE: .ASCIZ / XPM(W),FLG(R) DUAL ADRS FAULT/
2057	012311	040	040	130	XPCSE: .ASCIZ / XPM(W),CSR(R) DUAL ADRS FAULT/
2058	012351	040	040	130	XPMRE: .ASCIZ / XPM(W),MRR(R) DUAL ADRS FAULT/
2059	012411	040	040	130	XPMPE: .ASCIZ / XPM(W),MPM(R) DUAL ADRS FAULT/
2060	012451	040	040	130	XPXRE: .ASCIZ / XPM(W),XRR(R) DUAL ADRS FAULT/
2061	012511	040	040	115	MPMACE: .ASCIZ / MAIN MEM MGT ACCESS FAULT/
2062	012545	040	040	101	XPMACE: .ASCIZ / AUX MEM MGT ACCESS FAULT/
2063	012600	040	040	115	MPMNXE: .ASCIZ / MAIN MEM MGT NON-EXIST MEM FAULT/
2064	012643	045	101	040	DPCERA: .ASCIZ /%A (DPC) EXP'D: %06%A, REC'D: %06/
2065	012706	045	101	040	DSRERA: .ASCIZ /%A (DSR) EXP'D: %06%A, REC'D: %06/
2066	012751	045	101	040	DXRERA: .ASCIZ /%A (DXR) EXP'D: %06%A, REC'D: %06/
2067	013014	045	101	040	DYRERA: .ASCIZ /%A (DYR) EXP'D: %06%A, REC'D: %06/
2068	013057	045	101	040	PCSERA: .ASCIZ /%A (PCS) EXP'D: %06%A, REC'D: %06/
2069	013122	045	101	040	FLGERA: .ASCIZ /%A (FLG) EXP'D: %06%A, REC'D: %06/
2070	013165	045	101	040	CSRERA: .ASCIZ /%A (CSR) EXP'D: %06%A, REC'D: %06/
2071	013230	045	101	040	MRRERA: .ASCIZ /%A (MRR) EXP'D: %06%A, REC'D: %06/
2072	013273	045	101	040	MPMERA: .ASCIZ /%A (MPM) EXP'D: %06%A, REC'D: %06/
2073	013336	045	101	040	XRRERA: .ASCIZ /%A (XRR) EXP'D: %06%A, REC'D: %06/
2074	013401	045	101	040	XPMERA: .ASCIZ /%A (XPM) EXP'D: %06%A, REC'D: %06/
2075	013444	045	101	040	HBAERA: .ASCIZ /%A (HBA) EXP'D: %06%A, REC'D: %06/
2076	013507	045	101	040	CBAERA: .ASCIZ /%A (CBA) EXP'D: %06%A, REC'D: %06/
2077	013552	040	040	104	ECE: .ASCIZ / DIDN'T GET EXPECTED ERROR CODE/
2078					:MORE ERROR MESSAGES --
2079					
2080	013613	040	040	103	CRWRZ: .ASCIZ '' CURSOR REGISTER WRITE/READ ZEROS ERROR''
2081	013664	040	040	103	CRWRO: .ASCIZ '' CURSOR REGISTER WRITE/READ ONES ERROR''
2082	013734	040	040	103	CRDAX: .ASCIZ '' CURSOR REGISTER DUAL ADDRESS - WRITE Y CHANGED X''
2083	014017	040	040	103	CRDAY: .ASCIZ '' CURSOR REGISTER DUAL ADDRESS - WRITE X CHANGED Y''
2084	014102	040	040	103	CRWRE: .ASCIZ '' CURSOR REGISTER WRITE/READ ERROR''
2085	014145	040	040	112	JSEWRZ: .ASCIZ '' JOYSTICK ENABLES WRT/RD 0'S ERROR''
2086	014211	040	040	112	JSEWRO: .ASCIZ '' JOYSTICK ENABLES WRT/RD 1'S ERROR''
2087	014255	040	040	112	CEZWE: .ASCIZ '' JOYSTICK ENABLES ZERO-THEN-WRITE ERROR''
2088	014326	040	040	112	CESWE: .ASCIZ '' JOYSTICK ENABLES SET-THEN-WRITE ERROR''
2089	014376	040	040	112	JENOCL: .ASCIZ '' J.S. CHANNEL DECODE ERROR - ENABLES DIDN'T CLEAR''
2090	014461	040	040	122	CSRGSNE: .ASCIZ '' REGISTER ERROR AFTER J.S. SWITCH SET, NO ENABLE''
2091	014543	040	040	122	CSRGENS: .ASCIZ '' REGISTER ERROR AFTER J.S. SW ENABLE SET, NO SWITCH''
2092	014630	040	040	111	CSIFC: .ASCIZ '' INTERNAL REG. ERROR AFTER CLEARING J.S. SW INTR''
2093	014712	040	040	105	NOSWPI: .ASCIZ '' ERROR IN FLAGS REG. -- SHOULD SHOW PENDING SW INTR''
2094	014777	040	040	106	NOSWI: .ASCIZ '' FAULTY J.S. SWITCH INTERRUPT''
2095	015036	040	040	111	SWIBV: .ASCIZ '' INCORRECT VECTOR ON J.S. SWITCH INTERRUPT''
2096	015112	040	040	123	SWNOHLT: .ASCIZ '' SWITCH INTR REQ DIDN'T HALT DISPLAY PROCESSING - BAD DPC''
2097	015205	040	040	106	SWNOCU: .ASCIZ '' FAULTY CURSOR RETRIEVAL ON J.S. SW INTR REQ''
2098					
2099	015263	040	040	115	NOCMI: .ASCIZ '' MATCH INTERRUPT FAULTY OR NOT RECEIVED''
2100	015334	040	040	116	NOMST: .ASCIZ '' NO STOP ON MATCH INTERRUPT''
2101	015371	040	040	115	CMNOMI: .ASCIZ '' MATCH BUT NO MATCH INTERRUPT''

```

2102 015430 040 040 111 CMIVB: .ASCIZ  " INCORRECT VECTOR ON CURSOR MATCH INTERRUPT"
2103 015505 040 040 127 MATPCS: .ASCIZ  " WRONG DPC OR DSR FOR MATCH STOP"
2104 015547 040 040 127 MATXYE: .ASCIZ  " WRONG DXR OR DYR FOR MATCH STOP"
2105 015611 040 040 127 MATPOS: .ASCIZ  " WRONG DXR OR DYR AFTER POSITION-RESTORE"
2106 015663 040 040 111 MATIRE: .ASCIZ  " INTERNAL REGISTER INCORRECT FOR MATCH"
2107 015733 040 040 111 MEREG: .ASCIZ  " INTERNAL REGISTER INCORRECT AFTER SETUP FOR MATCH"
2108 016017 040 040 106 MATFLE: .ASCIZ  " FLAGS REGISTER INCORRECT FOR MATCH"
2109 016064 040 040 125 MATUI: .ASCIZ  " UNEXPECTED INTERRUPT ON RESUME AFTER MATCH"
2110 016141 040 040 104 MENDPCS: .ASCIZ  " DPC OR DSR WRONG ON AFTER RESUME FOLLOWING MATCH"
2111 016224 040 040 103 CMPE: .ASCIZ  " CURSOR MATCHED AT WRONG POSITION"
2112 016267 040 040 116 NOMAT: .ASCIZ  " NO CURSOR MATCH DETECTED"
2113 016322 045 116 045 EXPGT2: .ASCIZ  "/%N% EXP'D: %06%, %06%N% REC'D: %06%, %06/"
2114 016377 045 116 045 FCOORD: .ASCII  "/%N% CURSOR (X,Y) = %06%, %06/"
2115 016437 045 116 045 .ASCIZ  "/%N% REC'D DXR, DYR = %06%, %06/"
2116 016500 045 116 045 FSYCHAN: .ASCIZ  "%N% SYNC GEN/JOYSTICK CHANNEL = %01"
2117
2118
2119 016546 000      NUL: .ASCIZ  //
2120 016547 045 116 000 NULCR: .ASCIZ  "/%N/"
2121 016552 040 040 122 RLXIE: .ASCIZ  "/ RUN-LENGTH X-INCREMENT FAILURE/"
2122 016613 040 040 122 RLX2IE: .ASCIZ  "/ RUN-LENGTH X-DOUBLE INCREMENT FAILURE/"
2123 016663 040 040 122 RLYUE: .ASCIZ  "/ RUN-LENGTH Y-UP FAILURE/"
2124 016715 040 040 122 RLYDE: .ASCIZ  "/ RUN-LENGTH Y-DOWN FAILURE/"
2125 016751 040 040 122 RLY2UE: .ASCIZ  "/ RUN-LENGTH Y-SKIP UP FAILURE/"
2126 017010 040 040 122 RLY2DE: .ASCIZ  "/ RUN-LENGTH Y-SKIP DOWN FAILURE/"
2127 017051 045 101 040 EXPGOT: .ASCIZ  "/%A EXP'D: %06%, REC'D: %06/"
2128 017105 045 101 040 DUAD12: .ASCIZ  "/%A REG(W) WRITTEN TO: %06% REG(R) READ; EXP'D: %06%, REC'D: %06/"
2129 017207 040 040 115 INLE: .ASCIZ  "/ MEM INTERLACE ERROR/"
2130 017235 045 101 040 INLEXF: .ASCIZ  "/%A PIXEL READBACK EXP'D: %06%, REC'D: %06%, CHAN: %01/"
2131 017326 040 040 120 SCPRE: .ASCIZ  "/ PIXEL READBACK ERROR/"
2132 017355 040 040 123 SCIDE: .ASCIZ  "/ SYNC CHAN INTERLACE DIFFERENCE/"
2133 017416 040 040 103 SCME: .ASCIZ  "/ CONFIG DOESN'T MATCH MFG. MASTER/"
2134 017461 045 101 040 SCMFG: .ASCIZ  "/%A MFG MEM 0: %06%, MFG SYNC CHAN 0: %06/"
2135 017534 045 116 045 SCM: .ASCII  "/%N% MEMORYS:/"
2136 017552 045 101 040 .ASCII  "/%A 0 = %06/"
2137 017565 045 101 054 .ASCII  "/%A, 1 = %06/"
2138 017601 045 101 054 .ASCII  "/%A, 2 = %06/"
2139 017615 045 101 054 .ASCII  "/%A, 3 = %06/"
2140 017632 045 116 045 SCS: .ASCIZ  "/%N% SYNC CHANS:/"
2141 017653 045 101 040 .ASCII  "/%A 0 = %06/"
2142 017666 045 101 054 .ASCII  "/%A, 1 = %06/"
2143 017702 045 101 054 .ASCII  "/%A, 2 = %06/"
2144 017716 045 101 054 .ASCII  "/%A, 3 = %06%N/"
2145 017735 045 116 045 NOMAN: .ASCIZ  "/%N%*** MANUAL INTERVENTION NOT ALLOWED -- ABORTING ***%N/"
2146 .EVEN
    
```

```

2148 .SBTTL GLOBAL ERROR REPORT SECTION
2149
2150 :++
2151 : THE GLOBAL ERROR REPORT SECTION CONTAINS THE PRINTB AND PRINTX
2152 : CALLS THAT ARE USED IN MORE THAN ONE TEST.
2153 : ASCII TEXT STRINGS ARE FOUND IN THE GLOBAL TEXT SECTION.
2154 :--
2155
2156 020032 BGNMSG NXRERR ;NON-EXISTANT DEVICE REGISTER.
2157 020032 PRINTX #NXRX,NODEV ;NODEV = NEXM ADDRESS.
2158 020056 004737 022276 JSR PC,EXTEND ; PRINT EXTENSION IF REQUIRED.
2159 020062 ENDMSG
2160
2161 020064 BGNMSG PCSERR ;BAD DPC AND/OR DSR SIGNATURE.
2162 020064 PRINTX #PCSX,R2,R3,@DPC,@DSR ;R2,R3 = EXP PC,SR
2163 020120 004737 022276 JSR PC,EXTEND ; PRINT EXTENSION IF REQUIRED.
2164 020124 ENDMSG
2165
2166 020126 BGNMSG XYERR ;BAD DXR AND/OR DYR SIGNATURE.
2167 020126 PRINTX #XYX,R4,R5,@DXR,@DYR ;R4,R5 = EXP X,Y
2168 020162 004737 022276 JSR PC,EXTEND ; PRINT EXTENSION IF REQUIRED.
2169 020166 ENDMSG
2170
2171 020170 BGNMSG RLXYE ;BAD DXR AND/OR DYR SIGNATURE.
2172 020170 PRINTX #XYX,R1,R3,R2,R4 ; R1,R3 = EXP, R2,R4 = REC.
2173 020220 004737 022276 JSR PC,EXTEND ; PRINT EXTENSION IF REQUIRED.
2174 020224 ENDMSG
2175
2176 020226 BGNMSG INTERR ;DISPLAY INTERRUPT ERROR SIGNATURE.
2177 020226 PRINTX #INTX,R1,@DPC ;R1 = CPU PC
2178 020254 PRINTX #NULCR ; PRINT A NEWLINE
2179 020274 122737 000377 026107 CMPB #^0377,INTFLAG ;WAS ANY INTERRUPT GENERATED?
2180 020302 001011 BNE 1$ ;BR IF YES
2181 020304 PRINTX #FNOINTR ; NO -- REPORT THE FACT
2182 020324 000465 BR 20$
2183 020326 132737 000001 026107 1$: BITB #I0KSTP,INTFLAG ; WAS IT AN UNEXPECTED 'STOP' INTERRUPT?
2184 020334 001411 BEQ 2$ ; BR IF NO.
2185 020336 PRINTX #FUSI ; YES -- REPORT IT
2186 020356 000450 BR 20$
2187 020360 132737 000002 026107 2$: BITB #I0KJSM,INTFLAG ; WAS IT AN UNEXPECTED 'MATCH' INTERRUPT?
2188 020366 001411 BEQ 3$ ; BR IF NO.
2189 020370 PRINTX #FUMI ; REPORT IF YES.
2190 020410 000433 BR 20$
2191 020412 132737 000010 026107 3$: BITB #I0KJSS,INTFLAG ; WAS IT AN UNEXPECTED 'SWITCH' INTERRUPT?
2192 020420 001411 BEQ 4$ ; BR IF NO
2193 020422 PRINTX #FUSWI ; REPORT IF YES
2194 020442 000416 BR 20$
2195 020444 132737 000004 026107 4$: BITB #I0KERR,INTFLAG ; WAS IT AN UNEXPECTED 'ERROR' INTERRUPT?
2196 020452 001412 BEQ 20$
2197 020454 PRINTX #FUTO ; REPORT IT.
2198 020474 004737 023516 JSR PC,ERICHK ;CHECK THE ERROR CODE
2199 020500 004737 022276 20$: JSR PC,EXTEND ; PRINT EXTENSION IF REQUIRED.
2200 020504 ENDMSG
2201
2202 020506 BGNMSG IMDERR ;IMAGE MEM DATA ERROR.
2203 020506 010137 072506 MOV R1,IMCRAM ;TRANSLATE X ADDR...
2204 020512 006237 072506 ASR IMCRAM ;...TO RAM NUMBER.

```

2205	020516		PRINTX	#IMDIX,R1,R2,PDF,R3,R4	
2206	020552		PRINTX	#IMDIX2,IMCRAM,IMC.CH	
2207	020602	004737 022276	JSR	PC,EXTEND	; PRINT EXTENSION IF REQUIRED.
2208	020606		ENDMSG		
2209					
2210	020610		BGNMSG	LINTER	; LUT INTERRUPT ERROR.
2211	020610		PRINTX	#LINTX,@LSR	; SHOW CSR.
2212	020634	004737 022276	JSR	PC,EXTEND	; PRINT EXTENSION IF REQUIRED.
2213	020640		ENDMSG		
2214					
2215	020642		BGNMSG	LRWERR	; LUT READ/WRITE ERROR.
2216	020642		PRINTX	#LRWX,@LSR,R1,R2	; CSR, EXP'D, REC'D.
2217	020672	004737 022276	JSR	PC,EXTEND	; PRINT EXTENSION IF REQUIRED.
2218	020676		ENDMSG		
2219					
2220	020700		BGNMSG	EXPREC	
2221	020700		PRINTX	#EXPGOT,R1,R2	; R1=EXP'D, R2=REC'D.
2222	020724	004737 022276	JSR	PC,EXTEND	; PRINT EXTENSION IF REQUIRED.
2223	020730		ENDMSG		
2224					
2225	020732		BGNMSG	DPCER	
2226	020732		PRINTX	#DPCERA,R1,R2	; R1=EXPECTED, R2=RECEIVED.
2227	020756	004737 022276	JSR	PC,EXTEND	; PRINT EXTENSION IF REQ'D.
2228	020762		ENDMSG		
2229					
2230	020764		BGNMSG	DSRER	
2231	020764		PRINTX	#DSRERA,R1,R2	; R1=EXPECTED, R2=RECEIVED.
2232	021010	004737 022276	JSR	PC,EXTEND	; PRINT EXTENSION IF REQ'D.
2233	021014		ENDMSG		
2234					
2235	021016		BGNMSG	DXRER	
2236	021016		PRINTX	#DXRERA,R1,R2	; R1=EXPECTED, R2=RECEIVED.
2237	021042	004737 022276	JSR	PC,EXTEND	; PRINT EXTENSION IF REQ'D.
2238	021046		ENDMSG		
2239					
2240	021050		BGNMSG	DYRER	
2241	021050		PRINTX	#DYRERA,R1,R2	; R1=EXPECTED, R2=RECEIVED.
2242	021074	004737 022276	JSR	PC,EXTEND	; PRINT EXTENSION IF REQ'D.
2243	021100		ENDMSG		
2244					
2245	021102		BGNMSG	PCSER	
2246	021102		PRINTX	#PCSERA,R1,R2	; R1=EXPECTED, R2=RECEIVED.
2247	021126	004737 022276	JSR	PC,EXTEND	; PRINT EXTENSION IF REQ'D.
2248	021132		ENDMSG		
2249					
2250	021134		BGNMSG	FLGER	
2251	021134		PRINTX	#FLGERA,R1,R2	; R1=EXPECTED, R2=RECEIVED.
2252	021160	004737 022276	JSR	PC,EXTEND	; PRINT EXTENSION IF REQ'D.
2253	021164		ENDMSG		
2254					
2255	021166		BGNMSG	CSRER	
2256	021166		PRINTX	#CSRERA,R1,R2	; R1=EXPECTED, R2=RECEIVED.
2257	021212	004737 022276	JSR	PC,EXTEND	; PRINT EXTENSION IF REQ'D.
2258	021216		ENDMSG		
2259					
2260	021220		BGNMSG	MRRER	
2261	021220		PRINTX	#MRRERA,R1,R2	; R1=EXPECTED, R2=RECEIVED.

```

2262 021244 004737 022276          JSR      PC,EXTEND          ; PRINT EXTENSION IF REQ'D.
2263 021250          ENDMSG
2264
2265 021252          BGNMSG  MPMER
2266 021252          PRINTX #MPMERA,R1,R2          ; R1=EXPECTED, R2=RECEIVED.
2267 021276 004737 022276          JSR      PC,EXTEND          ; PRINT EXTENSION IF REQ'D.
2268 021302          ENDMSG
2269
2270 021304          BGNMSG  XRRER
2271 021304          PRINTX #XRRERA,R1,R2          ; R1=EXPECTED, R2=RECEIVED.
2272 021330 004737 022276          JSR      PC,EXTEND          ; PRINT EXTENSION IF REQ'D.
2273 021334          ENDMSG
2274
2275 021336          BGNMSG  XPMER
2276 021336          PRINTX #XPMERA,R1,R2          ; R1=EXPECTED, R2=RECEIVED.
2277 021362 004737 022276          JSR      PC,EXTEND          ; PRINT EXTENSION IF REQ'D.
2278 021366          ENDMSG
2279
2280 021370          BGNMSG  HBAER
2281 021370          PRINTX #HBAERA,R1,R2          ; R1=EXPECTED, R2=RECEIVED.
2282 021414 004737 022276          JSR      PC,EXTEND          ; PRINT EXTENSION IF REQ'D.
2283 021420          ENDMSG
2284
2285 021422          BGNMSG  CBAER
2286 021422          PRINTX #CBAERA,R1,R2          ; R1=EXPECTED, R2=RECEIVED.
2287 021446 004737 022276          JSR      PC,EXTEND          ; PRINT EXTENSION IF REQ'D.
2288 021452          ENDMSG
2289
2290 021454          BGNMSG  DUADRE
2291 021454          PRINTX #DUAD12,R3,R1,R2          ; R3=WRITTEN, R1=EXP'D, R2=REC'D.
2292 021502 004737 022276          JSR      PC,EXTEND          ; PRINT EXTENSION IF REQ'D.
2293 021506          ENDMSG
2294
2295 021510          BGNMSG  SYSCON
2296 021510 005737 002512          TST     MFGFLG              ; MFG MODE?
2297 021514 001414          BEQ     1$                  ; NO.
2298 021516          PRINTX #SCMFG,MFGMO,MFGSO
2299 021546          1$: PRINTX #SCM,MENTAB,MENTAB+2,MENTAB+4,MENTAB+6
2300 021606          PRINTX #SCS,SYCTAB,SYCTAB+2,SYCTAB+4,SYCTAB+6
2301 021646          ENDMSG
2302
2303 021650          BGNMSG  MPMER
2304 021650          PRINTX #MRRERA,R3,R4
2305 021674          PRINTX #NULCR
2306 021714          PRINTX #DPCERA,R1,R2
2307 021740          PRINTX #NULCR          ;PRINT BLANK LINE
2308 021760          ENDMSG
2309
2310
2311 021762          BGNMSG  XPMER
2312 021762          PRINTX #XRRERA,R3,R4
2313 022006          PRINTX #NULCR
2314 022026          PRINTX #DPCERA,R1,R2
2315 022052          PRINTX #NULCR          ;PRINT BLANK LINE
2316 022072          ENDMSG
2317
2318
    
```

```

GLOBAL ERROR REPORT SECTION

2319 022074          BGNMSG  EXPRC2
2320 022074          PRINTX  #EXPGT2,R1,R3,R2,R4      ;R1,R3 = EXP'D, R2,R4 = REC'D
2321 022124 004737 022276      JSR    PC,EXTEND      ; PRINT EXTENSION, IF REQ'D.
2322 022130          ENDMSG
2323
2324 022132          BGNMSG  PNTCOOR
2325 022132          PRINTX  #FCOORD,CSMF4X,CSMF4Y,SDXR,SDYR
2326 022172 004737 022276      JSR    PC,EXTEND
2327 022176          ENDMSG
2328
2329 022200          BGNMSG  PNTSYCH
2330 022200          PRINTX  #FSYCHAN,CSCHAN
2331 022224          ENDMSG
2332
2333 022226          BGNMSG  INLEX
2334 022226          PRINTX  #INLEXF,R1,R2,R5
2335 022254          PRINTX  #NULCR      ;PRINT BLANK LINE
2336 022274          ENDMSG
2337
2338          : THIS ROUTINE APPENDS A UNIQUE EXTENSION (IF REQUIRED)
2339          : TO ANY OF THE ABOVE ERROR SIGNATURES.
2340
2341 022276 005727          EXTEND: TST      (PC)+
2342 022300 000000          EXTA:   0              ; 0 = NO EXTENSION.
2343 022302 001402          BEQ     1$
2344 022304 004777 177770      JSR    PC,@EXTA      ; APPEND EXTENSION TEXT.
2345 022310          1$:   PRINTX  #NULCR      ; PRINT A BLANK LINE
2346 022330 000207          RTS     PC

```

```

2348 .SBTTL INTERNAL REGISTER CHECKER, ERROR CODE HANDLER, DPINIT
2349
2350 ;ROUTINES TO CHECK CONTENTS OF INTERNAL REGISTERS.
2351 ; UPON ENTRY, THE EXPECTED DATA SHOULD BE IN LOCATION 'GDDAT'
2352
2353 ;MACRO TO ASSEMBLE REGISTER CHECK FRONT-END:
2354 .MACRO CHECK RGS,RNAM
2355 MOV #SEL'RGS,-(SP) ;GET REGISTER-SELECT CODE ONTO THE STACK.
2356 MOV #99$,-(SP) ;AND THE ADDRESS OF THE REGISTER-NAME MESSAGE.
2357 JMP IREGCK ;THEN GO TO COMMON HANDLER.
2358 99$: .ASCIZ /%A 'RNAM'/
2359 .EVEN
2360 .ENDM CHECK
2361
2362 022332 DSRCHK: CHECK DSR,<DSR> ;CHECK THE DSR
2363 022360 PCSCHK: CHECK PCS,<PCSAVE> ;...PCSAVE
2364 022410 FLGCHK: CHECK FLG,<FLAGS> ;...FLAGS
2365 022440 CSRCHK: CHECK CSR,<CSR> ;...CSR
2366 022466 MMCHK: CHECK MRR,<MAIN MEM MGMT> ;...MAIN-SEG MEM. MGMT.
2367 022526 HBCHK: CHECK HBA,<HISTOGRAM BASE> ;...HISTOGRAM BASE
2368 022566 XMCHK: CHECK XRR,<AUX MEM MGMT> ;...AUX-SEG MEM. MGMT.
2369 022624 CBCHK: CHECK CBA,<CHARACTER BASE> ;...CHARACTER BASE
2370
2371
2372 022664 040 040 111 RGE: .ASCIZ / INCORRECT DATA IN INTERNAL REGISTER .../
2373 022736 045 116 045 GDBAD: .ASCIZ /%NZA EXP'D: %06ZA REC'D: %06/
2374 022775 045 116 045 TINERR: .ASCIZ /%NZA TEST: /
2375 023012 045 116 045 SIMSG: .ASCIZ /%NZA ... AFTER DOING SOFT INIT./
2376 .EVEN
2377
2378 023054 000000 LOOPFL:: 0 ;FLAG OR DATA FOR TEST-LOOP-DEPENDENT STUFF.
2379 023056 000000 GDDAT:: 0 ;STORAGE FOR EXPECTED DATA.
2380 023060 000000 BADDAT: 0 ;STORAGE FOR ACTUAL DATA.
2381 023062 000000 REGNAM: 0 ;ADDRESS OF REGISTER NAME STRING.
2382 023064 000000 SIFLAG: 0 ;FLAG TO SAY DOING SOFT INIT.
2383
2384 ;COME HERE WITH (SP)=ADDRESS OF NAME STRING, 2(SP)=SELECT CODE.
2385 023066 012637 023062 IREGCK: MOV (SP)+,REGNAM ;GET STRING ADDRESS.
2386 023072 012677 157736 MOV (SP)+,@DSR ;SELECT THE REGISTER.
2387 023076 017737 157732 023060 MOV @DSR,BADDAT ;GET THE DATA.
2388 023104 012777 000000 157722 MOV #SELDJR,@DSR ;SELECT THE REAL DSR
2389 023112 023737 023056 023060 CMP GDDAT,BADDAT ;COMPARE EXPECTED W/ ACTUAL.
2390 023120 001410 BEQ 1$ ;OK IF EQUAL; JUST EXIT.
2391 023122 HRDERR RGE,REGERR ;IF NOT, SAY SO & PRINT DATA.
2392 023142 000207 1$: RTS PC
2393
2394 023144 BGNMSG REGERR ;PRINT THE REGISTER NAME.
2395 023144 PRINTX REGNAM ;AND THE DATA.
2396 023164 PRINTX #GDBAD,GDDAT,BADDAT
2397 023214 PRINTX #TINERR
2398 023234 PRINTX TNAM ;PRINT TEST NAME
2399 023254 005737 023064 TST SIFLAG ;SOFT INIT?
2400 023260 001410 BEQ 2$ ;BR IF NO
2401 023262 PRINTX #SIMSG ;PRINT IF YES
2402 023302 2$: PRINTX #NULCR ;PRINT BLANK LINE
2403 023322 ENDMMSG
2404

```



```

2405 023324
2406 023324
2407 023344
2408 023364
2409 023404 000207
2410
2411
2412
2413
2414
2415
2416
2417 023406 012737 100000 023056 IIRCNC: MOV #CHAR,GDDAT ;SEE IF FLAGS HAS 'CHAR' MODE ONLY.
2418 023414 004737 022410 JSR PC,FLGCHK
2419 023420 000414 BR IIRNFC ;BYPASS CSR CHECK.
2420 023422 012737 100000 023056 IIRCHK: MOV #CHAR,GDDAT ;SEE IF FLAGS HAS 'CHAR' MODE ONLY.
2421 023430 004737 022410 JSR PC,FLGCHK
2422 023434 004737 023524 IIRCNC: JSR PC,ERRCHK ;SPECIFICALLY CHECK FOR AN ERROR CODE.
2423 023440 012737 000003 023056 MOV #SELCSR,GDDAT ;WE SHOULD ONLY SEE SELECT CODE IN CSR.
2424 023446 004737 022440 JSR PC,CSRCHK
2425 023452 012737 000001 023056 IIRNFC: MOV #1,GDDAT ;EXPECT TO SEE 'EMPTY' STATUS IN PCSAVE.
2426 023460 004737 022360 JSR PC,PCCHK ;...GO CHECK IT.
2427 023464 005037 023056 CLR GDDAT ;REMAINING REGISTERS S/B CLEAR.
2428 023470 004737 022466 JSR PC,MMCHK ;...MAIN MEM MGMT
2429 023474 004737 022526 JSR PC,HBCHK ;...HISTOGRAM BASE
2430 023500 004737 022566 JSR PC,XMCHK ;...AUX MEM MGMT
2431 023504 004737 022624 JSR PC,CBCHK ;...CHARACTER BASE.
2432 023510 005037 023064 CLR SIFLAG ;CLEAR SOFT-INIT FLAG.
2433 023514 000207 RTS PC
2434
2435
2436 023516 012746 000001 ;ERRCHK: CHECKS THE CSR FOR AN ERROR CODE & REPORTS IT.
2437 023522 000401 ERICLK: MOV #1,-(SP) ; SET FLAG TO INHIBIT USE OF 'DFERR'
2438 023524 005046 BR ERRCK1
2439 023526 012777 000003 157300 ERRCHK: CLR -(SP) ; CLEAR FLAG
2440 023534 017746 157274 ERRCK1: MOV #SELCSR,@DSR ;SELECT THE CSR.
2441 023540 012777 000000 157266 MOV @DSR,-(SP) ;READ IT.
2442 023546 011637 023060 MOV #SELDSCR,@DSR ;SELECT REAL DSR SO PEOPLE WON'T GET MIXED UP..
2443 023552 042716 177760 MOV (SP),BADDAT ;STORE CSR CONTENTS.
2444 023556 022627 000003 BIC #^C17,(SP) ;EXTRACT THE SELECT!WE FIELD.
2445 023562 001424 CMP (SP)+,#SELCSR ;WE SHOULD SEE THE CSR REG SELECT CODE.
2446 023564 005716 BEQ 1$
2447 023566 001011 TST (SP) ; SHOULD WE DO DFERR?
2448 023570 BNE 20$ ; BR IF NO
2449 023610 000466 DFERR NOCSR ; IF NOT, SAY WE CAN'T READ IT.
2450 023612 BR 2$ ; GO TO EXIT
2451 023632 000455 20$: PRINTX #FNOCNR ; PRINT MESSAGE 'CAN'T READ CSR'
BR 2$ ; GO TO EXIT
2452
2453 023634 005737 023060 1$: TST BADDAT ;TEST THE ERROR BIT.
2454 023640 100052 BPL 2$ ;OK IF CLEAR; JUST EXIT.
2455 023642 000337 023060 SWAB BADDAT ;IF SET, GET ERROR CODE
2456 023646 006237 023060 ASR BADDAT
2457 023652 006237 023060 ASR BADDAT ;...INTO BITS<4:1>.
2458 023656 042737 177741 023060 BIC #^C36,BADDAT ;ZAP OUT JUNK.
2459 023664 013746 023060 MOV BADDAT,-(SP) ;SAVE ERROR CODE * 2 ON STACK
2460 023670 062716 024266 ADD #ERCTBL,(SP) ;GET TO THE MESSAGE ADDRESS TABLE.
2461 023674 013637 024264 MOV @(SP)+,ERCNAM ;GET THE ADDRESS OF THE REG NAME MESSAGE.

```

```

2462 023700 005716          TST      (SP)          ; USE 'DFERR'?
2463 023702 001011          BNE      11$          ; BR IF NO
2464 023704          HRDERR  ERCM,ERCOD    ;REPORT THE BAD NEWS.
2465 023724 000420          BR       2$
2466 023726          11$:  PRINTX  #FERCM    ; PRINT 'CSR ERROR CODE DETECTED ='
2467 023746          PRINTX  ERCNAM      ; ... AND THE ERROR NAME.
2468 023766 005726          2$:  TST      (SP)+
2469 023770 000207          RTS      PC
2470
2471 023772          045      116      045  FNOC SR: .ASCII  /%N%A  ***/
2472 024003          040      040      103  NOCSR: .ASCIZ  / CAN'T READ THE CSR/
2473 024030          045      116      045  FERCM: .ASCII  /%N%A  ***/
2474 024041          040      040      103  ERCM:  .ASCIZ  / CSR ERROR CODE REC'D = /
2475          .EVEN
2476 024074          BGNMSG  ERCOD
2477 024074          PRINTX  ERCNAM
2478 024114 023727 023060 000026  CMP      BADDAT,#13+2 ; IS IT 'DBUS DATA R/W ERROR'?
2479 024122 001014          BNE      1$          ;BR IF NO
2480 024124          PRINTX  #GDBAD,@DXR,@DZR ; YES -- DXR = EXP'D, DZR = REC'D
2481 024154          1$:  PRINTX  #TINERR
2482 024174          PRINTX  TNAM
2483 024214 005737 023064          TST      SIFLAG      ;SOFT INIT?
2484 024220 001410          BEQ      2$          ;BR IF NO
2485 024222          PRINTX  #SIMSG    ;PRINT IF YES
2486 024242          2$:  PRINTX  #NULCR    ;PRINT BLANK LINE
2487 024262          ENDMMSG
2488 024264 000000          ERCNAM: 0
2489 024266 024326 024365 024430  ERCTBL: 77$,1$,2$,3$,4$,5$,6$,7$
2490 024306 024735 025004 025067 10$,11$,12$,13$,14$,15$,16$,17$
2491
2492          ;ERROR CODE EXPLANATIONS:
2493 024326          045      101      056  77$:  .ASCIZ  /%A... 0: (ILLEGAL ERROR CODE)/
2494 024365          045      101      056  1$:   .ASCIZ  /%A... 1: NXM TIMEOUT ON DMA X'FER/
2495 024430          045      101      056  2$:   .ASCIZ  /%A... 2: (ILLEGAL ERROR CODE)/
2496 024467          045      101      056  3$:   .ASCIZ  /%A... 3: MEMORY PROTECTION ERROR/
2497 024531          045      101      056  4$:   .ASCIZ  /%A... 4: SEQUENCE ERROR/
2498 024562          045      101      056  5$:   .ASCIZ  /%A... 5: RESERVED OPERATION/
2499 024617          045      101      056  6$:   .ASCIZ  /%A... 6: VECTOR ENDPOINT MISMATCH/
2500 024662          045      101      056  7$:   .ASCIZ  /%A... 7: U-PROC EXECUTED UNUSED MICROWORD/
2501 024735          045      101      056  10$:  .ASCIZ  /%A... 10: IMAGE MEMORY 'SYNC' TIMEOUT/
2502 025004          045      101      056  11$:  .ASCIZ  /%A... 11: PIXEL READBACK 'DATA AVAILABLE' TIMEOUT/
2503 025067          045      101      056  12$:  .ASCIZ  /%A... 12: J.S. STATUS 'DATA READY' TIMEOUT/
2504 025143          045      101      056  13$:  .ASCIZ  /%A... 13: DBUS DATA READ-WRITE ERROR/
2505 025211          045      101      056  14$:  .ASCIZ  /%A... 14: DBUS SIGNAL HUNG/
2506 025245          045      101      056  15$:  .ASCIZ  /%A... 15: VBUS SIGNAL HUNG/
2507 025301          045      101      056  16$:  .ASCIZ  /%A... 16: (ILLEGAL ERROR CODE)/
2508 025341          045      101      056  17$:  .ASCIZ  /%A... 17: (ILLEGAL ERROR CODE)/
2509          .EVEN
2510
2511          ;ROUTINE TO DO A 'SOFT' INIT ON THE DPU:
2512          .ENABL  LSB
2513 025402 004737 031412  DPINIT: JSR      PC,SAVERS ; SAVE R0-R5.
2514 025406 012737 000001 023064  MOV      #1,SIFLAG ; SAY DOING SOFT INIT
2515 025414 012777 000000 155412  MOV      #0,@DSR ;SELECT THE REAL DSR
2516 025422 005777 155406          TST      @DSR ;ARE WE STOPPED?
2517 025426 100423          BMI      11$ ;BR IF YES
2518 025430 004737 026756          JSR      PC,RELEAS ;SEE IF WE'RE HUNG.
    
```



```

2577      .SBTTL  GLOBAL SUBROUTINES SECTION
2578
2579      :++
2580      : THE GLOBAL SUBROUTINES SECTION CONTAINS THE SUBROUTINES
2581      : THAT ARE USED IN MORE THAN ONE TEST.
2582      :--
2583
2584      :
2585      : DEFAULT DISPLAY INTERRUPT HANDLERS.
2586      : IF DISPLAY TIME-OUT, REPORT DEV FATAL, AND ABORT PASS.
2587      : OTHERWISE, SAVE DPU REGISTERS AND DISMISS.
2588      :
2589
2590      : BIT DEFINITIONS FOR "INTMASK" AND "INTFLAG" BYTES:
2591      :
2592      :           IOKCKIN=BIT7      ; DON'T CHECK FOR BAD INTERRUPTS -- TEST WILL.
2593      :           IOKSTP=BIT0       ; EXPECT "STOP" INTERRUPT
2594      :           IOKJSM=BIT1       ; EXPECT JOYSTICK "MATCH" INTERRUPT.
2595      :           IOKERR=BIT2       ; EXPECT "ERROR" INTERRUPT.
2596      :           IOKJSS=BIT3       ; EXPECT "JOYSTICK SWITCH" INTERRUPT.
2597      :
2598      : INTERRUPT MASK -- SAYS EXPECTING INTERRUPTS
2599      INTMASK: 026106      000      .BYTE 0
2600
2601      : INTERRUPT FLAG -- SAYS WE GOT ONE (IF POSITIVE)
2602      INTFLAG: 026107      000      .BYTE 0
2603
2604      : SAVED INTERRUPT VECTOR:
2605      INTVEC: 026110      000000    .WORD 0
2606      : SAVE CPU PC
2607      INTCPC: 026112      000000    .WORD 0
2608
2609      : SUBROUTINE TO ENABLE INTERRUPTS:
2610      ENAINT: MOV      RO, -(SP)      ; SAVE R0
2611      MOV      STPV, RO      ; GET POINTER TO VECTORS
2612      MOV      #DSTP, (RO)+      ; SET UP STOP VECTOR
2613      MOV      #PRI07, (RO)+      ; SET UP MATCH VECTOR
2614      MOV      #DJM, (RO)+
2615      MOV      #DT0, (RO)+      ; SET UP ERROR VECTOR
2616      MOV      #PRI07, (RO)+
2617      MOV      #DJS, (RO)+      ; SET UP JOYSTICK SWITCH VECTOR
2618      MOV      #PRI07, (RO)
2619      MOV      (SP)+, RO      ; RESTORE R0
2620      MOV      (SP), -(SP)
2621      MOV      #0, 2(SP)      ; SET CPU TO LEVEL 0
2622      RTI
2623
2624      : SUBROUTINE TO DISABLE INTERRUPTS (RAISE PRIORITY TO LEVEL 7)
2625      DSBINT: MOV      (SP), -(SP)
2626      MOV      #PRI07, 2(SP)
2627      RTI
2628
2629      : ERROR (TIME-OUT) INTERRUPT SERVICE:
2630      DTO:  MOV      TOTV, INTVEC      ; SAVE THE VECTOR
2631      CLRB     INTFLAG      ; CLEAR FLAG TO SAY WE GOT INTERRUPT
2632      BITB     #IOKERR, INTMASK      ; EXPECTING ERROR INTERRUPT?
2633      BNE     DPU SAV      ; BR IF YES; NO ERROR.
    
```

```

2634 026232 152737 000004 026107      BISB  #IOKERR,INTFLAG ;NO. SET THE ERROR FLAG.
2635 026240 000447                BR    DPUSAV      ;GO FINISH UP.
2636
2637 026242 013737 003052 026110  DSTP:  MOV    STPV,INTVEC  ;SAVE THE VECTOR
2638 026250 105037 026107                CLRB  INTFLAG      ;CLEAR FLAG TO SAY WE GOT INTERRUPT
2639 026254 132737 000001 026106      BITB  #IOKSTP,INTMASK ;EXPECTING STOP INTERRUPT?
2640 026262 001036                BNE   DPUSAV      ;BR IF YES
2641 026264 152737 000001 026107      BISB  #IOKSTP,INTFLAG ;NO. SET THE ERROR FLAG.
2642 026272 000432                BR    DPUSAV      ;GO FINISH UP.
2643
2644 026274 013737 003054 026110  DJM:   MOV    JSMV,INTVEC ;SAVE THE VECTOR
2645 026302 105037 026107                CLRB  INTFLAG      ;CLEAR FLAG TO SAY WE GOT INTERRUPT
2646 026306 132737 000002 026106      BITB  #IOKJSM,INTMASK ;EXPECTING MATCH INTERRUPT?
2647 026314 001021                BNE   DPUSAV      ;BR IF YES
2648 026316 152737 000002 026107      BISB  #IOKJSM,INTFLAG ;NO. SET THE ERROR FLAG.
2649 026324 000415                BR    DPUSAV      ;GO FINISH UP.
2650
2651 026326 013737 003060 026110  DJS:   MOV    JSSV,INTVEC ;SAVE THE VECTOR
2652 026334 105037 026107                CLRB  INTFLAG      ;CLEAR FLAG TO SAY WE GOT INTERRUPT
2653 026340 132737 000010 026106      BITB  #IOKJSS,INTMASK ;EXPECTING SWITCH INTERRUPT?
2654 026346 001004                BNE   DPUSAV      ;BR IF YES
2655 026350 152737 000010 026107      BISB  #IOKJSS,INTFLAG ;NO. SET THE ERROR FLAG.
2656 026356 000400                BR    DPUSAV      ;GO FINISH UP.
2657
2658
2659 026360 011637 026112      DPUSAV: MOV    (SP),INTCPC ;SAVE CPU PC.
2660 026364 010146                MOV   R1,-(SP)    ;SAVE R1
2661 026366 013701 003034                MOV   DSR,R1      ;GET DSR ADDRESS INTO R1
2662 026372 017737 154434 003126      MOV   @DPC,IDPC   ;SAVE DISPLAY PC...
2663 026400 011137 003130                MOV   (R1),IDSR   ;...STATUS...
2664 026404 017737 154426 003132      MOV   @DXR,IDX   ;...X POSITION...
2665 026412 017737 154422 003134      MOV   @DYR,IDY   ;...Y POSITION...
2666 026420 012711 000001                MOV   #SELPCS,(R1) ;SAVE PCSAVE...
2667 026424 011137 003146                MOV   (R1),IPCSAV ;
2668 026430 012711 000002                MOV   #SELFLG,(R1) ;SAVE FLAGS...
2669 026434 011137 003150                MOV   (R1),IFLAGS ;
2670 026440 012711 000003                MOV   #SELCSR,(R1) ;SAVE CSR...
2671 026444 011137 003152                MOV   (R1),ICSR  ;
2672 026450 012711 000004                MOV   #SELMRR,(R1) ;SAVE MAIN MEM-MGMT..
2673 026454 011137 003154                MOV   (R1),IMAIN ;
2674 026460 012711 000005                MOV   #SELMBA,(R1) ;SAVE HISTO. BASE...
2675 026464 011137 003156                MOV   (R1),IHBASE ;
2676 026470 012711 000006                MOV   #SELXRR,(R1) ;SAVE AUX. MEM-MGMT..
2677 026474 011137 003160                MOV   (R1),IAUX  ;
2678 026500 012711 000007                MOV   #SELCBA,(R1) ;SAVE CHAR. BASE...
2679 026504 011137 003162                MOV   (R1),ICBASE ;
2680 026510 012711 000000                MOV   #SELDSR,(R1) ;END UP AT DSR...
2681 026514 011137 003144                MOV   (R1),ISDSR ;...AND SAVE IT.
2682                                BITB  #IOKERR,INTFLAG ; DID WE GET AN ERROR INTERRUPT?
2683 026520 105737 026107                TSTB  INTFLAG     ; ANY ERROR FLAGS SET?
2684 026524 001424                BEQ   1$          ; BR IF NO
2685 026526 105737 026106                TSTB  INTMASK     ; YES. IS CHECKING TURNED OFF?
2686 026532 100421                BMI   1$          ; BR IF YES
2687 026534 013746 022300                MOV   EXTA,-(SP)  ; SAVE ERROR EXTENSION ADDRESS.
2688 026540 012737 023324 022300      MOV   #PTINER,EXTA ; SET EXTENSION TO PRINT TEST IN ERROR.
2689 026546 013701 026112                MOV   INTCPC,R1  ; GET CPU PC.
2690 026552 004737 030216                JSR   PC,INCERK  ; INCREMENT THE ERROR COUNTS
    
```

```

2691 026556          ERRDF 0,IFAU,INTERR ;REPORT ERROR...
2692 026566 012637 022300      MOV  (SP)+,EXTA ;RESTORE EXTA
2693 026572 004737 030254      JSR  PC,CKEMAX
2694
2695 026576 012601          1$:  MOV  (SP)+,R1 ;RESTORE R1
2696 026600 000002          RTI ;...AND DISMISS.
2697
2698          : AND A SIMILAR ONE FOR THE LUT INTERRUPTS.
2699          :
2700 026602 000137          LDUN: JMP  @(PC)+
2701 026604 026606          LUTSAV
2702 026606 017737 154230 003136 LUTSAV: MOV  @LSR,ILSR ; SAVE LUT REGISTERS.
2703 026614 017737 154224 003140      MOV  @LDR,ILDR
2704 026622 017737 154220 003142      MOV  @LMR,ILMR
2705 026630 000002          RTI ; AND DISMISS.
    
```


2764	027014	010001		FILL1:	MOV	R0,R1	:	FILL WITH (R0).
2765	027016	000402			SKP2			
2766	027020	010001		FILL2:	MOV	R0,R1		
2767	027022	005101			COM	R1	:	FILL WITH ALTERNATE WORDS.
2768	027024	013702	003240		MOV	FREE,R2	:	POINTER.
2769	027030	013703	003242		MOV	FRESIZ,R3	:	WORD COUNT.
2770	027034	042703	100003		BIC	#100003,R3	:	INSURE WC IS EVEN.
2771	027040	010022		1\$:	MOV	R0,(R2)+	:	1ST WORD...
2772	027042	010122			MOV	R1,(R2)+	:	...NEXT WORD.
2773	027044	162703	000002		SUB	#2,R3		
2774	027050	003373			BGT	1\$		
2775	027052	000207			RTS	PC		


```

2832
2833      : SUBROUTINE TO BLAST THE LUT RAM WITH COLOR OR GREY-SCALE DATA.
2834
2835 027054 012701 107222 LUMIN: MOV #LUMTBL,R1 ; GREY-SCALE TABLE ADDRESS.
2836 027060 012746 000001      MOV #1,-(SP) ; LUT WORDS / SHADE.
2837 027064 000404      SKP2+2
2838 027066 012701 110222 NTSC: MOV #NTSC8,R1 ; BASIC 8 COLOR TABLE ADDRESS.
2839 027072 012746 000040      MOV #32,-(SP) ; LUT WORDS / COLOR.
2840 027076 000404      SKP2+2
2841 027100 012701 110242 EXPR: MOV #EXPRM,R1 ; EXPERIMENTAL 16 COLOR TABLE.
2842 027104 012746 000020      MOV #16,-(SP) ; LUT WORDS / COLOR.
2843
2844 027110 005737 003216      TST LUTAV
2845 027114 001424      BEQ 3$ ; RETURN IF NO LUT INSTALLED.
2846 027116 005077 153720      CLR @LSR ; ADDR 0, NO INTERRUPTS.
2847 027122 052777 000100 153716      BIS #GCOFF,@LMR ; GAMMA CORRECTION OFF.
2848 027130 011600      1$: MOV (SP),R0 ; SET ITERATION K.
2849 027132 012177 153706      MOV (R1)+,@LDR ; BLAST THE RAM.
2850 027136 005777 153700      2$: TST @LSR ; WAIT FOR READY.
2851 027142 100375      BPL -4
2852 027144 105777 153672      TSTB @LSR ; LAST ADDRESS DONE ??
2853 027150 001406      BEQ 3$ ; EXIT IF SO.
2854 027152 005300      DEC R0
2855 027154 001765      BEQ 1$ ; GET NEXT COLOR.
2856 027156 016177 177776 153660      MOV -2(R1),@LDR ; REPEAT THIS COLOR.
2857 027164 000764      BR 2$
2858 027166 005726      3$: TST (SP)+ ; FIX THE STACK...
2859 027170 000207      RTS PC ; ...AND RETURN.
2860
2861      :
2862      : BLAST 16 SHADES OF THE BASIC COLOR IN (R1).
2863      : LOCATION "SHADLY" SET BY CALLER TO PRODUCE A PLEASING
2864      : VISUAL EFFECT.
2865 027172 005737 003216 SHAD16: TST LUTAV
2866 027176 001433      BEQ 3$ ; RETURN IF NO LUT INSTALLED.
2867 027200      BREAK
2868 027202 005077 153634      CLR @LSR ; ADDR 0, NO INTERRUPTS.
2869 027206 052777 000100 153632      BIS #GCOFF,@LMR ; GAMMA CORRECT OFF.
2870 027214 042701 177356      BIC #^C421,R1 ; SET DIMMEST SHADE THIS COLOR.
2871 027220 005002      CLR R2 ; 1ST SHADE IS ALWAYS BLACK.
2872 027222 012703 000020      1$: MOV #16.,R3 ; 16 LUT WORDS / SHADE.
2873 027226 010277 153612      2$: MOV R2,@LDR ; WRITE DATA.
2874 027232 005777 153604      TST @LSR
2875 027236 100375      BPL -4
2876 027240 013700 003220      MOV SHADLY,R0 ; DELAY.
2877 027244 005300      DEC R0
2878 027246 001376      BNE -2
2879 027250 105777 153566      TSTB @LSR ; LUT RAM FULL ??
2880 027254 001404      BEQ 3$ ; EXIT IF SO.
2881 027256 005303      DEC R3 ; THIS SHADE DONE ??
2882 027260 001362      BNE 2$ ; NOT YET, LOOP.
2883 027262 060102      ADD R1,R2 ; YES, INCREMENT THE SHADE...
2884 027264 000756      BR 1$ ; ...AND CONTINUE.
2885 027266 000207      3$: RTS PC
    
```

```

2887
2888
2889
2890
2891
2892
2893 027270 020027 001000
2894 027274 103003
2895 027276 005400
2896 027300 062700 000100
2897 027304 042700 177600
2898 027310 000207
2899
2900 027312 004737 027270
2901 027316 000300
2902 027320 006200
2903 027322 000207
2904
2905
2906
2907
2908
2909
2910 027324 012737 030060 003232
2911 027332 013737 003232 003234
2912 027340 042701 170000
2913 027344 010102
2914 027346 000302
2915 027350 006202
2916 027352 042702 177770
2917 027356 150237 003232
2918 027362 010102
2919 027364 012700 000006
2920 027370 006202
2921 027372 005300
2922 027374 001375
2923 027376 042702 177770
2924 027402 150237 003233
2925 027406 010102
2926 027410 012700 000003
2927 027414 006202
2928 027416 005300
2929 027420 001375
2930 027422 042702 177770
2931 027426 150237 003234
2932 027432 042701 177770
2933 027436 150137 003235
2934 027442 013701 003232
2935 027446 013702 003234
2936 027452 000207

: SUBROUTINES TO TRANSLATE AN ABSOLUTE X OR Y TO SHORT FORM
: DELTA X OR Y. USED IN SVEC AND RPNT TESTS.
: INPUT R0 = ABS CO-ORD, IN THE RANGE (1000 +/-76).
: OUTPUT R0 = DX OR DY CORRECTLY POSITIONED AND SIGNED.
SDY:  CMP      R0,#1000
      BHIS     .+10           ;SKP3 IF POS DELTA.
      NEG      R0           ;MAKE NEG DELTA.
      ADD      #100,R0       ;SET SIGN BIT.
      BIC      #^C177,R0    ;R0 = +/-DY IN <6:0>.
      RTS      PC

SDX:  JSR      PC,SDY       ;MAKE A DY...
      SWAB     R0           ;...MOVE TO HI BYTE.
      ASR      R0           ;R0 = +/-DX IN <13:7>.
      RTS      PC

: SUBROUTINE TO CONVERT 12 BIT OCTAL TO ASCII FOR DISPLAY.
: R1 = OCTAL X OR Y CO-ORD AT ENTRY.
: R1,,R2 = 4 DIGIT ASCII EQUIVALENT AT EXIT.
: R0, R1, AND R2 NOT SAVED AT ENTRY.
CNVRT: MOV      #'00,TEMP1   ;ASCII 00
      MOV      TEMP1,TEMP2   ;HERE TOO.
      BIC      #^C7777,R1    ;STRIP ANY CRAP BITS.
      MOV      R1,R2
      SWAB     R2
      ASR      R2
      BIC      #^C7,R2
      BISB     R2,TEMP1      ;1ST (HI ORDER) DIGIT.
      MOV      R1,R2
      MOV      #6,R0
1$:   ASR      R2           ;SHIFT OFF 6 BITS.
      DEC      R0
      BNE     1$
      BIC      #^C7,R2
      BISB     R2,TEMP1+1    ;2ND DIGIT.
      MOV      R1,R2
      MOV      #3,R0
2$:   ASR      R2           ;SHIFT OFF 3 BITS.
      DEC      R0
      BNE     2$
      BIC      #^C7,R2
      BISB     R2,TEMP2      ;3RD DIGIT.
      BIC      #^C7,R1
      BISB     R1,TEMP2+1    ;4TH (AND LAST) DIGIT.
      MOV      TEMP1,R1     ;HI HALF => R1.
      MOV      TEMP2,R2     ;LO HALF => R2.
      RTS      PC          ;AND EXIT.
    
```

```

2938
2939
2940
2941
2942
2943 027454
2944 027456 012727 005670
2945 027462 005670
2946 027464 005777 153344
2947 027470 100407
2948 027472 004737 027530
2949 027476 005337 027462
2950 027502 001370
2951 027504 000241
2952 027506 000401
2953 027510 000261
2954 027512 000207
2955
2956
2957
2958
2959
2960
2961
2962
2963
2964 027514 012746 023420
2965 027520 000402
2966 027522 012746 011610
2967 027526 000402
2968 027530 012746 000001
2969 027534 011646
2970 027536 010066 000002
2971 027542 013700 027564
2972 027546 005300
2973 027550 001376
2974 027552 005316
2975 027554 001372
2976 027556 005726
2977 027560 012600
2978 027562 000207
2979 027564 000010

: SUBROUTINE TO WAIT FOR DISPLAY STOP FLAG.
: RETURN WITH 'C' = 1 AS SOON AS STOP OCCURS.
: OTHERWISE, 'C' = 0 AND RETURN AFTER APPROX 300 MSEC.
WAITF: BREAK ; DO A SUPVSR BREAK FIRST.
        MOV #3000.,(PC)+ ; 300 MSEC TIMER.
1$:     3000.
2$:     TST @DSR
        BMI 3$ ; EXIT ON STOP FLAG.
        JSR PC,US100 ; OTHERWISE, WAIT 100 USEC...
        DEC 1$
        BNE 2$ ;...AND TRY AGAIN.
        CLC ; C = 0, DPU STILL RUNNING...
        SKP1 ;...OR HUNG-UP AFTER 300 MSEC.
3$:     SEC ; C = 1, DPU IS STOPPED.
        RTS PC

: SUBROUTINES TO PAUSE (WAIT) IN INCREMENTS OF 100 USEC.
: BASED ON AVERAGE TIME (CPU DEPENDANT) TO EXECUTE
: DEC RO
: BNE -2
: DELAY CONSTANT (K100US) SHOULD BE ADJUSTED AS FOLLOWS:
: FOR SLOW CPU'S (11/03 THRU 10), K100US = 8.
: FOR FAST CPU'S (11/34 THRU 70), K100US = 32.
PAUSE1: MOV #10000.,-(SP) ; 1 SEC TIMER.
        SKP2
PAUS.5: MOV #5000.,-(SP) ; 1/2 SEC TIMER.
        SKP2
US100:  MOV #1, -(SP) ; 100 USEC TIMER.
        MOV (SP), -(SP) ; SAVE THE COUNT
        MOV RO, 2(SP) ; SAVE RO
1$:     MOV K100US, RO
        DEC RO
        BNE -2 ; 100 USEC (MORE OR LESS) LOOP.
        DEC (SP)
        BNE 1$ ; VARIABLE LOOP.
        TST (SP)+ ; FIX STACK...
        MOV (SP)+, RO ; RESTORE RO
        RTS PC ;...AND RETURN.
K100US: 8. ; 100 USEC TIMER.
    
```

```

2981
2982
2983
2984
2985
2986
2987
2988
2989
2990
2991 027566 012737 027620 000004
2992 027574 012737 000200 000006
2993 027602 005003
2994 027604 005711
2995
2996 027606 020102
2997 027610 001407
2998 027612 062701 000002
2999 027616 000772
3000
3001 027620 005103
3002 027622 012716 027630
3003 027626 000002
3004 027630
3005 027636 005703
3006 027640 001401
3007 027642 000261
3008 027644 000207
3009
3010
3011
3012
3013
3014
3015
3016 027646
3017 027646 005737 002510
3018 027652 001006
3019 027654 005737 003030
3020 027660 100403
3021 027662 005337 027700
3022 027666 001002
3023 027670 000241
3024 027672 000401
3025 027674 000261
3026 027676 000207
3027 027700 000000

: ROUTINE TO TEST FOR A NEXM IN THE RANGE (R1) THRU (R2).
: ON RETURN, IF 'C' = 1, (R1) = NEXM ADDRESS.
: 'C' = 0, ALL ADDRESSES OK.
:
: CALL:  MOV ADR1,R1
:        MOV ADR2,R2
:        JSR PC,NXM
:        RETURN          ;TEST 'C' AND PROCEED.
NXM:    MOV #2$,@#4      ; SET BUSERR VECTOR.
        MOV #PRI04,@#6
        CLR R3          ; FLAG.
1$:     TST (R1)        ; TEST THE ADDRESS(ES).
        ; IF ANY TRAP, CONTINUE AT 2$.
        ; OTHERWISE, CONTINUE HERE.
        ; BR IF FINISHED (NO NEXM'S).
        ; SET NEXT ADDRESS...
        ; ...AND CONTINUE.
        CMP R1,R2
        BEQ 3$
        ADD #2,R1
        BR 1$
2$:     COM R3          ; GOT ONE, SET FLAG...
        MOV #3$, (SP)
        RTI            ; ...AND DISMISS INTERRUPT...
3$:     CLRVEC #4      ; ...AND GIVE BACK THE VECTOR.
        TST R3         ; DID WE CATCH ONE ??
        BEQ .+4        ; NO, 'C' = 0, SKIP NEXT.
        SEC           ; YES, 'C' = 1, (R1) = NEXM ADDR.
        RTS PC
:
: SUBROUTINE TO EXECUTE TEST ITERATIONS.
: EXIT WITH 'C' SET IF LOOPS ALLOWED AND LOOP COUNT NON-ZERO.
: LOOP COUNTER IS SET BY 'BEGIN.TEST' MACRO.
:
: CALL:  LOOPTO ARG
:
LOOP:   TST NOITS      ; ITERATIONS INHIBITED?
        BNE 1$        ; YES.
        TST QVP       ; NO.
        BMI 1$        ; LOOPS DISALLOWED IN QUICK PASS.
        DEC LOOPK     ; BUMP LOOP COUNTER.
        BNE 2$
1$:     CLC           ; LOOP DISALLOWED, OR DONE.
        SKP1
2$:     SEC           ; LOOP ENABLED.
        RTS PC
LOOPK:  0             ; LOOP (ITERATION) COUNTER.
    
```

```

3029
3030
3031
3032
3033
3034
3035
3036
3037
3038 027702 005037 023064
3039 027706 005037 030130
3040 027712 005037 022300
3041 027716 105037 026106
3042 027722 013700 003026
3043 027726 006300
3044 027730 005737 003226
3045 027734 001430
3046 027736 100010
3047 027740 052760 160000 003302
3048 027746
3049 027756 000407
3050 027760 052760 160001 003302 3$:
3051 027766
3052 027776 012737 177777 003224 2$:
3053 030004
3054 030012
3055 030014 000422
3056
3057 030016
3058 030020 032700 001000
3059
3060
3061 030024 001412
3062 030026
3063 030052 005237 030070
3064 030056 004737 026114
3065 030062 000207
3066 030064 000000
3067 030066 000000
3068 030070 000000
3069
3070
3071
3072
3073 030072
3074 030074 030027 020000
3075 030100 001412
3076 030102
3077 030126 000207
3078
3079 030130 000000
3080 030132 045 101 040
3081 030151 105 122 122
3082
3083
3084
3085

```

```

: PRINT THE NUMBER AND NAME OF EACH TEST AS WE GO ALONG.
: INCREMENT 'TESTK' TO INDICATE THE NUMBER OF TESTS
: IN THE CURRENT RUN SEQUENCE.
: CLEAR THE ERROR COUNTER AND SIGNATURE EXTENSION FLAGS.
:
: *** NOTE: REQUIRES PRIOR USE OF THE 'BEGIN.TEST' MACRO
: TO SET NAME AND NUMBER POINTERS.
:
TSTGO: CLR SIFLAG ; CLEAR 'SOFT INIT' FLAG
        CLR ERRK ; CLEAR LOCAL ERROR COUNTER.
        CLR EXTA ; CLEAR ERROR EXTENSION FLAG.
        CLRB INTMASK ; CLEAR INTERRUPT MASK (CHECK ERROR)
        MOV UNITN,R0 ; GET THE UNIT NUMBER,
        ASL R0 ; ... AND MAKE IT A WORD OFFSET.
        TST NODEV ; DID STARTUP FIND THE DEVICE?
        BEQ 4$ ; BR IF YES
        BPL 3$ ; BR IF NOT IDLE
        BIS #160000,ERTABL(R0) ; FLAG ERROR IN THE ERROR TABLE
        ERRDF 1,NXR,NXRERR ; NO DEVICE HERE -- PRINT IT
        BR 2$
3$: BIS #160001,ERTABL(R0) ; FLAG ERROR IN THE ERROR TABLE
    ERRDF 2,NOINIT ; DEVICE NOT IDLE
2$: MOV #-1,DUFLG ; DROP THE UNIT
    DODU UNITN
    DOCLN ; ABORT THE PASS
    BR 5$
4$: RFLAGS R0 ; GET THE OPERATOR FLAGS.
    BIT #PNT,R0 ; PRINT THE TEST NUMBERS?
    TST TIDFLG ; WANNA TYPE THE TEST ID?
    BNE 1$ ; NAH!
    BEQ 1$ ; BR IF NO
    PRINTF TNAM,TNUM ; PRINT NUMBER AND TITLE.
1$: INC TESTK ; BUMP TEST COUNTER.
    JSR PC,ENAINTE ; ENABLE INTERRUPTS
5$: RTS PC
TNUM: 0
TNAM: 0
TESTK: 0 ; NUMBER OF TESTS RUN THIS PASS.
:
: AT END OF EACH TEST, PRINT THE NUMBER OF ERRORS RECEIVED
: IF NORMAL ERROR REPORTING IS DISABLED (FLA:IER).
:
TSTEND: RFLAGS R0
        BIT R0,#IER ; BR IF 'IER' NOT SET.
        BEQ 1$ ; PRINT ERROR COUNT.
1$: PRINTF #ESUM,ERRK
    RTS PC
ERRK: 0 ; LOCAL ERROR COUNT.
ESUM: .ASCIZ /%A %D%A ERRORS/
EMAXDU: .ASCIZ /ERROR LIMIT REACHED -- DROPPING UNIT/
        .EVEN
: ROUTINES TO INCREMENT LOCAL ERROR COUNT AND CHECK FOR LIMIT:
:

```

```

3086 030216 005237 030130      INCERK: INC      ERRK      : INCREMENT LOCAL ERROR COUNT
3087 030222 010046              MOV      RO,-(SP) : SAVE RO
3088 030224 013700 003026      MOV      UNITN,RO : GET UNIT NUMBER,
3089 030230 006300              ASL      RO      : ... AND MAKE IT A WORD OFFSET.
3090 030232 062700 003302      ADD      #ERTABL,RO : RO GETS ADDRESS OF ERROR TABLE ENTRY.
3091 030236 005210              INC      (RO)     : INCREMENT THE DEVICE ERROR COUNT
3092 030240 032710 007777      BIT      #7777,(RO) : DID WE OVERFLOW THE FIELD?
3093 030244 001001              BNE     1$       : BR IF NO.
3094 030246 005310              DEC      (RO)     : YES -- BACK IT UP TO 7777.
3095 030250 012600 1$:      MOV      (SP)+,RO : RESTORE RO
3096 030252 000207              RTS      PC      : RETURN TO CALLER.
3097
3098 030254 010046      CKEMAX: MOV      RO,-(SP) : SAVE RO
3099 030256 013700 003026      MOV      UNITN,RO : GET UNIT NUMBER
3100 030262 006300              ASL      RO      : ... AND MAKE IT A WORD OFFSET
3101 030264 016000 003302      MOV      ERTABL(RO),RO : GET ERROR TABLE ENTRY
3102 030270 042700 170000      BIC      #170000,RO : EXTRACT ERROR COUNT FIELD
3103 030274 020037 002516      CMP      RO,GERRMAX : IS GLOBAL LIMIT EXCEEDED FOR THIS UNIT?
3104 030300 103004              BHIS    1$       : BR IF YES
3105 030302 023737 030130 002514      CMP      ERRK,LERRMAX : IS LOCAL LIMIT EXCEEDED FOR THIS TEST?
3106 030310 103417              BLO     2$       : BR IF NO
3107 030312              1$:      RFLAGS   RO      : GET OPERATOR FLAGS
3108 030314 032700 000040      BIT      #IDU,RO  : IS DROPPING INHIBITED?
3109 030320 001013              BNE     2$       : BR IF YES.
3110 030322 012737 177777 003224      MOV      #-1,DUFLG : NO -- DROP THE UNIT
3111 030330              ERRDF   4,EMAXDU
3112 030340              DODU    UNITN
3113 030346              DOCLN
3114 030350 012600 2$:      MOV      (SP)+,RO : RESTORE RO
3115 030352 000207              RTS      PC      : RETURN TO CALLER
3116
3117 030354 010046      CKDROP: MOV      RO,-(SP)
3118 030356              RFLAGS   RO
3119 030360 032700 000040      BIT      #IDU,RO
3120 030364 001011              BNE     1$
3121 030366 012600              MOV      (SP)+,RO
3122 030370 012737 177777 003224      MOV      #-1,DUFLG
3123 030376              DODU    UNITN
3124 030404              DOCLN      :ABORT THE PASS
3125 030406 000401              BR      2$
3126 030410 012600 1$:      MOV      (SP)+,RO
3127 030412 000207 2$:      RTS      PC
    
```

```

3129
3130      ; SUBROUTINE - CONFIGURE VSV11 SYSTEM.
3131      ;
3132      CONFIG:
3133      CONMEM: JSR      PC,DPINIT
3134      MOV      #40000,R0      ; PROTECT ALL MEMORIES
3135      10$:    MOV      R0,@DYR
3136      JSR      PC,WAITF
3137      ADD      #CH1,R0
3138      BIT      #CH3,R0
3139      BNE      10$
3140
3141      MOV      #0,@DXR      ; TRY A PIXEL READ.
3142      JSR      PC,WAITF
3143      BCS      1$          ; BR IF UNSUCCESSFUL.
3144      DFERR    PRBH        ; PIXEL READBACK HANGS UP.
3145      1$:    MOV      #144003,GDDAT
3146      JSR      PC,CSRCHK
3147      CLR      R5          ; INIT MEM TABLE PTR.
3148      2$:    MOV      #13,@DSR      ; CLEAR CSR.
3149      CLR      CTABM(R5)    ; CLEAR TABLE ENTRY.
3150      MOV      R5,R1      ; ENABLE MEM R/W.
3151      ASR      R1
3152      SWAB    R1
3153      BIS      #40060,R1
3154      MOV      R1,@DYR
3155      JSR      PC,WAITF
3156      MOV      #0,@DXR      ; PIXEL READ.
3157      JSR      PC,WAITF
3158      MOV      #3,@DSR
3159      MOV      @DSR,R2      ; IS MEM THERE?
3160      BMI      5$          ; NO.
3161      MOV      #0,@DSR      ; YES. GET PIXEL DATA.
3162      MOV      @DSR,R2
3163      BIS      #176003,R2  ; INVERT PIXEL DATA.
3164      COM     R2
3165      MOV      R2,CTABM(R5) ; PUT PIXEL DATA IN MEM TABLE.
3166      MOV      #8.,R3      ; GOT AN EVEN NUMBER OF PIXEL BITS?
3167      CLR      R4
3168      ASR      R2
3169      3$:    ASR      R2
3170      BIT      #1,R2
3171      BEQ     4$
3172      INC     R4
3173      4$:    DEC     R3
3174      BNE     3$
3175      BIT      #1,R4
3176      BEQ     5$
3177      HRDERR  SCPRE,SYSCON ; NO.
3178      5$:    BIC     #60,R1    ; YES. RE-PROTECT THE MEM.
3179      MOV      R1,@DYR
3180      JSR      PC,WAITF
3181      ADD      #2,R5      ; NEXT MEM.
3182      CMP     R5,#10      ; ALL MEMS CHECKED?
3183      BNE     2$          ; NO.
3184      6$:    MOV      #BIT3,MEMFLG ; YES. SET ONE FLAG TO INDICATE MEMS AVAIL.
3185      MOV      #MEMTAB,R1
    
```

```

3186 030740 005721          7$:   TST      (R1)+      :
3187 030742 001403          BEQ      8$          :
3188 030744 052737 000020 003252  BIS      #BIT4, MEMFLG :
3189 030752 006237 003252  8$:   ASR      MEMFLG    :
3190 030756 103370          BCC      7$          :
3191 030760 004737 025402  CONSYC: JSR      PC, DPINIT :
3192 030764 005003          CLR      R3          :
3193 030766 005005          CLR      R5          : INIT SYNC CHAN TABLE PTR.
3194 030770 012777 000013 152036 1$:   MOV      #13, @DSR   : CLEAR CSR.
3195 030776 005065 003270  CLR      CTABS(R5)  : CLEAR TABLE ENTRY.
3196 031002 010501          MOV      R5, R1     : READ CURSOR STATUS.
3197 031004 006201          ASR      R1          :
3198 031006 000301          SWAB    R1          :
3199 031010 052701 042100  BIS      #42100, R1 :
3200 031014 010177 152016  MOV      R1, @DXR   :
3201 031020 004737 027454  JSR      PC, WAITF  :
3202 031024 012777 000003 152002  MOV      #3, @DSR   : GOT A DATA READY ERROR?
3203 031032 017702 151776  MOV      @DSR, R2   :
3204 031036 042702 003777  BIC      #3777, R2  :
3205 031042 020227 150000  CMP      R2, #150000 :
3206 031046 001412          BEQ      2$          : YES. (NO CHAN HERE).
3207 031050 052765 100000 003270  BIS      #BIT15, CTABS(R5) : NO. SET CHAN EXISTS IN TABLE.
3208 031056 017702 151756  MOV      @DYR, R2   : GET INTERLACE BIT.
3209 031062 042702 157777  BIC      #157777, R2 :
3210 031066 050265 003270  BIS      R2, CTABS(R5) : SET IT IN TABLE.
3211 031072 050203          BIS      R2, R3     : REMEMBER IT IN R3.
3212 031074 062705 000002 2$:   ADD      #2, R5     : NEXT SYNC CHAN.
3213 031100 020527 000010  CMP      R5, #10   : ALL SYNC CHANS CHECKED?
3214 031104 001331          BNE      1$          : NO.
3215 031106 012705 003270  MOV      #CTABS, R5 : YES. ALL INTERLACE BITS THE SAME?
3216 031112 010337 003230  MOV      R3, INTLAC : REMEMBER INTERLACE/NON-INTERLACE STATUS
3217 031116 052703 100000  BIS      #BIT15, R3 : (R3 HAS LAST INTERLACE BIT FOUND).
3218 031122 005715          3$:   TST      (R5)      : GOT AN ENTRY IN TABLE?
3219 031124 001402          BEQ      4$          : NO.
3220 031126 021503          CMP      (R5), R3   : YES. INTERLACE BIT SAME?
3221 031130 001006          BNE      5$          : NO.
3222 031132 062705 000002 4$:   ADD      #2, R5     : YES. NEXT TABLE ENTRY.
3223 031136 020527 003300  CMP      R5, #CTABS+10 : DONE?
3224 031142 001367          BNE      3$          : NO.
3225 031144 000410          BR      6$          : YES.
3226
3227 031146          5$:   HRDERR  SCIDE, SYSCON :
3228
3229 031166 005737 002512  6$:   TST      MFGFLG    : MANUFACTURING FLAG SET?
3230 031172 001420          BEQ      9$          : NO.
3231 031174 023737 003256 003022  CMP      MEMTAB, MFGMO : YES. NEW MEM 0 = MFG MEM 0 MASTER?
3232 031202 001004          BNE      7$          : NO.
3233 031204 023737 003270 003024  CMP      SYCTAB, MFGSO : NEW SYNC CHAN 0 = MFG MASTER SYNC CHAN 0?
3234 031212 001410          BEQ      9$          : YES.
3235 031214          7$:   HRDERR  SCME, SYSCON : NO.
3236
3237 031234 012737 000010 003254 9$:   MOV      #BIT3, SYCFLG : SET ONE FLAG TO INDICATE SYNC CHANS AVAIL.
3238 031242 012701 003270  MCV      #SYCTAB, R1 :
3239 031246 005721          10$:  TST      (R1)+      :
3240 031250 001403          BEQ      11$         :
3241 031252 052737 000020 003254 11$:  BIS      #BIT4, SYCFLG :
3242 031260 006237 003254  ASR      SYCFLG    :
    
```


3243	031264	103370	BCC	10\$:
3244	031266	000207	RTS	PC	
3245					
3246					
3247					

```
3249
3250      ; SUBROUTINE - ENABLE MEM MGT.
3251
3252 031270 005737 003244      KTON:   TST      KTFLG      ; GOT KT?
3253 031274 001403              :      BEQ      1$          ; NO.
3254 031276 012737 000001 177572 :      MOV     #1,SRO     ; YES. ENABLE KT11.
3255 031304 000207              1$:      RTS      PC
3256
3257
3258
3259      ; SUBROUTINE - DISABLE MEM MGT.
3260
3261
3262 031306      KTOFF:  ;TST      KTFLG      ; GOT KT11?
3263              ;BEQ      1$          ; NO.
3264 031306 000240              NOP
3265 031310 000240              NOP
3266 031312 012737 000000 177572 :      MOV     #0,SRO     ; DISABLE KT.
3267 031320 000207              1$:      RTS      PC
```

```

3269
3270      ; SUBROUTINE - FILL ALL MEMORY OVER 28K WITH "STOPN".
3271      ; (NOTE - ENABLE KT BEFORE CALLING THIS).
3272      ;
3273 031322 005737 003244      KTBKGD: TST      KTFLG      ; GOT KT?
3274 031326 001430          BEQ      9$        ; NO. GET OUT.
3275 031330 004737 031270      JSR      PC,KTON    ; YES. ENABLE KT.
3276 031334 012700 173000      MOV      #STOPN,R0  ; FILL ALL MEM (>28K) WITH BACKGROUND.
3277 031340 012737 001600 172354  MOV      #1600,@#KIPAR6
3278 031346 012701 140000      1$:      MOV      #140000,R1
3279 031352 012702 150000      MOV      #150000,R2
3280 031356 010021          2$:      MOV      R0,(R1)+
3281 031360 020102          CMP      R1,R2
3282 031362 001375          BNE      2$
3283 031364 023737 172354 003244  CMP      @#KIPAR6,KTFLG
3284 031372 001404          BEQ      3$
3285 031374 062737 000200 172354  ADD      #200,@#KIPAR6
3286 031402 000761          BR       1$
3287
3288 031404 004737 031306      3$:      JSR      PC,KTOFF  ; DISABLE KT.
3289 031410 000207          9$:      RTS      PC
    
```

```

3291
3292      ; SUBROUTINE TO SAVE R0-R5.
3293
3294 031412 010037 031462   SAVERS: MOV    R0,SAVED
3295 031416 012700 031464   MOV    #SAVED+2,R0
3296 031422 010120         MOV    R1,(R0)+
3297 031424 010220         MOV    R2,(R0)+
3298 031426 010320         MOV    R3,(R0)+
3299 031430 010420         MOV    R4,(R0)+
3300 031432 010520         MOV    R5,(R0)+
3301 031434 000207         RTS    PC
3302
3303      ; SUBROUTINE TO RESTORE R0-R5.
3304
3305
3306 031436 012700 031464   RESTRS: MOV    #SAVED+2,R0
3307 031442 012001         MOV    (R0)+,R1
3308 031444 012002         MOV    (R0)+,R2
3309 031446 012003         MOV    (R0)+,R3
3310 031450 012004         MOV    (R0)+,R4
3311 031452 012005         MOV    (R0)+,R5
3312 031454 013700 031462   MOV    SAVED,R0
3313 031460 000207         RTS    PC
3314
3315
3316
3317 031462 000000   SAVED:  0
3318 031464 000000   0
3319 031466 000000   0
3320 031470 000000   0
3321 031472 000000   0
3322 031474 000000   0
    
```

```

3324
3325      ; SUBROUTINE TO SET-UP VARIOUS ENVIRONMENTAL PARAMETERS.
3326
3327 031476      ENVIRN: MEMORY R0
3328 031500      MOV R0,FREE ; GET 1ST FREE ADDRESS...
3329 031504 010037 003240      ADD #2,FREE
3330 031512 011037 003242      MOV (R0),FRESIZ ;...AND WORD COUNT.
3331 031516 162737 000004 003242      SUB #4,FRESIZ
3332 031524 013702 002012      MOV L$UNIT,R2 ; GET NUMBER OF UNITS
3333 031530 162737 000007 003242 10$: SUB #7,FRESIZ ; TAKE AWAY 7 WORDS PER UNIT
3334 031536 005302      DEC R2
3335 031540 001373      BNE 10$
3336
3337 031542 012737 000010 027564      MOV #8.,K100US ; ASSUME LSI FOR 100US TIMER.
3338 031550      READBUS
3339 031552      BCOMPLETE 1$
3340 031554 012737 000040 027564      MOV #32.,K100US ; NOT LSI -- CHANGE TIMER.
3341 031562      ;MOV #60.,HZ ; ASSUME 60 HZ SYSTEM.
3342      ;
3343      ;
3344      ;
3345      ;
3346      ;
3347      ;
3348      ;
3349      ;
3350 031562 004737 031656      ;
3351 031566 000240 000240      1$: ;MOV #MAXY60,YMAX ;
3352      ;
3353      ;
3354      ;
3355      ;
3356      ;
3357      ;
3358      ;
3359      ;
3360      ;
3361      ;
3362      ;
3363      ;
3364 031572 104042      ;
3365 031574 010037 003236      ;
3366 031600 042700 007777      ;
3367 031604 162700 027000      ;
3368 031610 062700 026050      ;
3369 031614 010037 000206      ;
3370 031620 012737 000137 000204      ;
3371 031626 013700 003236      ;
3372 031632 042700 007777      ;
3373 031636 062700 003726      ;
3374 031642 010037 000212      ;
3375 031646 012737 000137 000210      ;
3376 031654 000207      ;
3377      ;
3378      ;

```

```

*****
: NOW THE FOLLOWING CODE PROVIDES A KEY RESTART FOR
: THE DIAGNOSTIC SUPERVISOR (DRS), AND FOR XXDP+ MONITOR.
: IF EITHER ONE CHANGES, THIS PROBABLY WILL NOT WORK !!
: BUT FOR NOW -- IT'S CONVENIENT FOR DEBUGGING PURPOSES.
:
: THE FOLLOWING OFFSETS APPLY TO DRS REV D.
XX.ENTRY= 25570 ; DRS (REV D) ENTRY FROM XXDP+.
SUPVSR= 26050 ; DRS (REV D) SUPERVISOR RESTART.
XX.EXIT= 26072 ; DRS (REV D) EXIT TO XXDP+.
XXDP= 3726 ; XXDP+ RESTART ADDRESS (XX3726).
:
KLUDGE: EMT 42 ; XXDP+ COMM BLOCK POINTER => R0
MOV R0,XXCOMM ; SAVE IT.
BIC #7777,R0 ; MASK TO 2K PAGE...
SUB #27000,R0 ;...MINUS DRS SIZE (5.75K)...
ADD #SUPVSR,R0 ;...PLUS DRS RESTART OFFSET...
MOV R0,#206 ;... = DRS RESTART ADDRESS.
MOV #137,#204 ; LOC 204 = JMP DRS-RESTART.
MOV XXCOMM,R0 ; GET POINTER AGAIN.
BIC #7777,R0 ; MASK TO 2K PAGE...
ADD #XXDP,R0 ;...PLUS RESTART OFFSET...
MOV R0,#212 ;... = XXDP+ RESTART ADDRESS.
MOV #137,#210 ; LOC 210 = JMP XXDP+
RTS PC
*****

```

```

3380
3381      ; SUBROUTINE - IF OVER 28K & KT11 AVAIL, INIT KT11.
3382      ;
3383      ; KTINIT: CLR      KTFLG
3384      ;          CMP      LSHIME,#1577      ; GOT ENOUGH MEMORY (>28K)?
3385      ;          BLOS     9$                ; NO.
3386      ;          MOV      @#ERRVEC,R0      ; SAVE OLD ERR VEC PTR.
3387      ;          MOV      #2$,@#ERRVEC    ; SET ERR VEC PTR.
3388      ;          TST      @#SRO          ; GOT KT11?
3389      ;          NOP
3390      ;          MOV      LSHIME,KTFLG    ; (TRAP IF NO).
3391      ;          BIC      #177,KTFLG      ; YES. SET KT FLAG.
3392      ;          MOV      R0,@#ERRVEC    ; RESTORE OLD ERR VEC PTR.
3393      ;          CLR      R0              ; R0 = AR DATA.
3394      ;          MOV      #KIPAR,R1       ; R1 = KI REGS PTR.
3395      ;          MOV      #77406,-40(R1) ; SET DESCRIPTOR REG.
3396      ;          MOV      R0,(R1)+      ; SET KIPAR REG.
3397      ;          ADD      #200,R0        ; BUMP AR DATA BY '4K'.
3398      ;          CMP      R0,#2000      ; AT 'I/O'?
3399      ;          BNE     1$              ; NO.
3400      ;          MOV      #177600,-(R1)  ; YES. SET KTPAR7 FOR I/O.
3401      ;          BR      9$
3402
3403      ; 2$: FORRTI
3404      ;          MOV      #.+6,(SP)      ; FORCE RTI RETURN TO -
3405      ;          RTI
3406      ;          MOV      R0,@#ERRVEC    ; - NEXT LOC.
3407      ;          MOV      R0,@#ERRVEC    ; RESTORE OLD ERR VEC PTR.
3408
3409      ; 9$: RTS      PC
    
```

```

3408 .SBTTL INITIALIZE SECTION
3409
3410 :++
3411 : THE INITIALIZE SECTION CONTAINS THE CODING THAT IS PERFORMED
3412 : AT THE BEGINNING OF EACH PASS.
3413 :--
3414 032004 BGNPROT
3415 032004 177777 177777 177777 -1, -1, -1 ; NO DEVICE PROTECTION REQUIRED.
3416 032014 ENDPROT
3417
3418 032014 BGNINIT
3419
3420 : IF "START" OR "RESTART", SET QUICK-PASS FLAG AND BUS-INIT.
3421 : IF "CONTINUE", NOTHING IS REQUIRED.
3422 :
3423 032014 005037 023064 CLR SIFLAG ;CLEAR "SOFT INIT" FLAG
3424 032020 READEF #EF.CONTINUE
3425 032026 BNCOMPLETE 1$
3426 032030 023737 003026 002012 CMP UNITN,LSUNIT ;UNIT IN RANGE?
3427 032036 103056 BHIS 4$ ;BR IF NO.
3428 032040 005737 003224 TST DUFLG ;DROPPED UNIT?
3429 032044 100460 BMI NXTU ;BR IF YES
3430 032046 013701 003026 MOV UNITN,R1
3431 032052 006301 ASL R1
3432 032054 005761 003302 TST ERTABL(R1)
3433 032060 001504 BEQ SETU
3434 032062 032761 040000 003302 BIT #BIT14,ERTABL(R1) ; DROPPED?
3435 032070 001046 BNE NXTU
3436 032072 EXIT INIT ;DO NOTHING IF "CONTINUE".
3437 032076 1$: READEF #EF.NEW ;TAKE NEXT UNIT IF NOT NEW PASS.
3438 032104 BNCOMPLETE NXTU
3439 032106 READEF #EF.START
3440 032114 BCOMPLETE 2$
3441 032116 READEF #EF.RESTART
3442 032124 BNCOMPLETE 31$
3443 032126 2$: ; 1ST PASS, BUS-INIT...
3444 032126 BRESET ; BUS RESET.
3445 032130 012737 177777 003030 MOV #-1,QVP ;...QUICK VERIFY...
3446 032136 005037 030070 CLR TESTK ;...CLEAR TOTAL TEST COUNT.
3447 032142 004737 031476 JSR PC,ENVIRN ; SET ENVIRONMENT.
3448 032146 012700 003302 MOV #ERTABL,R0
3449 032152 005020 30$: CLR (R0)+ ;CLEAR THE ERROR TABLE
3450 032154 020027 003502 CMP R0,#ERTABE
3451 032160 103774 BLO 30$
3452 032162 000404 BR 4$
3453 032164 005037 003030 31$: CLR QVP
3454 032170 000137 032240 JMV PASRPT ;GO REPORT THE STATUS
3455
3456 032174 4$:
3457 032174 012737 177777 003026 NEWPAS: MOV #-1,UNITN ;INIT UNIT NUMBER...
3458 032202 005037 032726 CLR DEVCNT ;CLEAR COUNT OF DEVICES RUNNING
3459 032206 NXTU: BREAK
3460 032210 005237 003026 INC UNITN ;...AND SET NEXT UNIT NUMBER.
3461 032214 023737 003026 002012 CMP UNITN,LSUNIT
3462 032222 103423 BLO SETU
3463 032224 012737 177777 003224 MOV #-1,DUFLG
3464 032232 000401 BR 11$

```

3465	032234				DOCLN		:ABORT, NO MORE UNITS.
3466	032236	000240			NOP		
3467	032240				11\$: PASRPT:		
3468	032240	023727	002012	000001	CMP	LSUNIT,#1	:HOW MANY UNITS SELECTED?
3469	032246	101752			BLOS	NEWPAS	:BR IF ONLY 1
3470	032250	005737	032726		TST	DEV CNT	:ARE ANY STILL RUNNING?
3471	032254	001747			BEQ	NEWPAS	:BR IF NO
3472	032256				RFLAGS	RO	
3473	032260	032700	000100		BIT	#ISR,RO	:SHOULD WE PRINT STATISTICS
3474	032264	001343			BNE	NEWPAS	:BR IF NO
3475							
3476	032266				DORPT		
3477	032270	000741			BR	NEWPAS	
3478	032272				10\$: SETU:		
3479							
3480	032272				GPHARD	UNITN,RO	:GET UNIT N P-TABLE POINTER.
3481	032300				BNCOMPLETE	NXTU	:BR IF UNIT NOT AVAILABLE.
3482	032302	005037	003224		CLR	DUFLG	:CLEAR 'DROPPED' FLAG.
3483	032306	005237	032726		INC	DEV CNT	
3484	032312	012001			MOV	(R0)+,R1	: GET 1ST REGISTER ADDRESS.
3485	032314	012702	003032		MOV	#DPC,R2	
3486	032320	010122			20\$: MOV	R1,(R2)+	
3487	032322	062701	000002		ADD	#2,R1	
3488	032326	020227	003046		CMP	R2,#LMR	
3489	032332	101772			BLOS	20\$	
3490							
3491							
3492	032334	012001			MOV	(R0)+,R1	: GET 1ST VECTOR ADDRESS.
3493	032336	011002			MOV	(R0),R2	: GET INTERRUPT PRIORITY
3494	032340	012037	003050		MOV	(R0)+,DPRI	: SET INTERRUPT PRIORITY.
3495	032344	010137	003052		MOV	R1,STPV	: SET STOP VECTOR POINTER...
3496	032350	012721	026242		MOV	#DSTP,(R1)+	:...VECTOR...
3497	032354	010221			MOV	R2,(R1)+	:...AND PRIORITY.
3498	032356	010137	003054		MOV	R1,JSMV	: DITTO JOY-STICK MATCH.
3499	032362	012721	026274		MOV	#DJM,(R1)+	
3500	032366	010221			MOV	R2,(R1)+	
3501	032370	010137	003056		MOV	R1,TOTV	: DITTO TIME-OUT.
3502	032374	012721	026210		MOV	#DIO,(R1)+	
3503	032400	010221			MOV	R2,(R1)+	
3504	032402	010137	003060		MOV	R1,JSSV	: DITTO JOY-STICK SWITCH.
3505	032406	012721	026326		MOV	#DJS,(R1)+	
3506	032412	010221			MOV	R2,(R1)+	
3507	032414	010137	003062		MOV	R1,LUTV	: DITTO LUT DONE.
3508	032420	012721	026602		MOV	#LDUN,(R1)+	
3509	032424	010221			MOV	R2,(R1)+	
3510	032426	012037	003216		MOV	(R0)+,LUTAV	: GET 'LUT AVAILABLE' ENTRY
3511	032432	012737	001676	003250	MOV	#MAXY60,YMAX	: ASSUME 60HZ FREQUENCY.
3512	032440	012737	000074	003246	MOV	#60.,HZ	
3513	032446	005720			TST	(R0)+	: WHAT IS SELECTED FREQUENCY?
3514	032450	001406			BEQ	21\$: BR IF 60HZ
3515	032452	012737	000062	003246	MOV	#50.,HZ	: SAY 50 HZ
3516	032460	012737	001776	003250	MOV	#MAXY50,YMAX	: AND SET UP FOR 512 VISIBLE LINES
3517	032466				21\$: CLR	OPTI	
3518	032466	005037	005064		TST	DPUMOD	: DPU ONLY?
3519	032472	005737	002502		BNE	1\$: SKIP NEXT IF DPU-ONLY.
3520	032476	001003			MOV	#I,OPTI	:SET OPTIONAL I(NTENSIFY) BIT.
3521	032500	012737	040000	003064			

3522 032506
3523
3524

1\$:
:
:
:

TST QVP
BEQ 5\$

:1ST PASS ??
:NO, SKIP THE PASS 1 STUFF.

```

3526
3527
3528
3529
3530 032506 013701 003026
3531 032512 006301
3532 032514 052761 100000 003302
3533 032522 005037 022300
3534 032526 023727 002012 000001
3535 032534 101416
3536 032536
3537 032540 032700 001000
3538 032544 001412
3539 032546
3540 032572
3541 032572 005037 003226
3542 032576 013701 003032
3543 032602 013702 003040
3544 032606 004737 027566
3545 032612 103003
3546 032614 010137 003226
3547 032620 000416
3548 032622 012777 000000 150204 2$:
3549 032630 005777 150200
3550 032634 100413
3551 032636 013701 003026
3552 032642 006301
3553 032644 005261 003302
3554 032650 012737 000001 003226
3555 032656 012737 177777 003224 3$:
3556
3557 032664 005737 003216 4$:
3558 032670 001412
3559 032672 013701 003042
3560 032676 013702 003046
3561 032702 004737 027566
3562 032706 103003
3563 032710 012737 000000 003216
3564
3565
3566
3567 032716
3568 032724
3569
3570 032726 000000
3571 032730 045 116 045
3572
3573
3574
3575
3576
3577
3578
3579
3580 032776 005737 003020
3581 033002 001402
3582 033004 062716 000004

:
: 1ST PASS, CHECK THAT DEVICE ADDRESSES ARE VALID, AND
: THAT THE DISPLAY STATUS IS PROPERLY INITIALIZED.
:
:
: MOV UNITN,R1
: ASL R1
: BIS #BIT15,ERTABL(R1) ; SAY DEVICE RUNNING
: CLR EXTA ; CLEAR ERROR EXTENSION FLAG.
: CMP L$UNIT,#1 ; ARE WE TESTING MULTIPLE UNITS?
: BLOS 10$ ; BR IF NO.
: RFLAGS R0 ; YES -- GET OPERATOR FLAGS.
: BIT #PNT,R0 ; SHOULD WE PRINT UNIT #?
: BEQ 10$ ; BR IF NOT.
: PRINTF #PUNIT,UNITN ; PRINT THE UNIT #
:
10$:
: CLR NODEV
: MOV DPC,R1
: MOV DYR,R2
: JSR PC,NXM ;TEST ALL 4 DPU REGISTERS...
: BCC 2$ ;...AND BR IF ALL OK.
: MOV R1,NODEV ;FLAG DEVICE AS NON-EXISTENT
: BR 3$ ;DROP THIS UNIT.
: MOV #0,@DSR ;SELECT REAL DSR
: TST @DSR ;IS UNIT STOPPED?
: BMI 4$ ;BR IF DISPLAY INITIALIZED OK.
: MOV UNITN,R1
: ASL R1
: INC ERTABL(R1)
: MOV #1,NODEV ;FLAG AS NOT IDLE
: MC #-1,DUFLG
:
4$:
: TST LUTAV ; NOW, SEE IF LUT INSTALLED.
: BEQ 5$ ; BR IF NO
: MOV LSR,R1
: MOV LMR,R2
: JSR PC,NXM ; TEST ALL LUT REGISTERS...
: BCC 5$ ;...BR IF IT'S THERE.
: MOV #0,LUTAV ; CLEAR 'LUT AVAILABLE' FLAG.
:
: FINALLY, SET CPU PRIORITY AND WE'RE DONE.
5$:
: SETPRI #PRI00 ;ENABLE INTERRUPTS.
: ENDINIT
:
DEVcnt: 0 ;COUNT OF RUNNING DEVICES
PUNIT: .ASCIZ /%N%N%***** TESTING UNIT %D2%A *****/
: .EVEN
:
: SUBROUTINE - IF USER PATCHABLE FLAG 'FORCER' IS SET (NON-ZERO),
: THEN FORCE RETURN TO THE NEXT ERROR CALL (HRDERR) IN SEQUENCE.
:
FORCET: TST FORCER ; FORCE ERROR TYP0UT?
: BEQ 1$ ; NO.
: ADD #4,(SP) ; YES. FORCE RTN TO THE HRDERR CALL.

```

3583 033010 000207

1\$: RTS PC ;

```

3585          .SBTTL  ADD AND DROP UNITS SECTIONS
3586
3587          :++
3588          : THE ADD-UNIT SECTION CONTAINS THE CODING THAT CAUSES A DEVICE
3589          : TO BE (A) ADDED TO THE TEST LIST FOR THE FIRST TIME,
3590          : OR (B) RE-INSERTED IF IT HAD BEEN PREVIOUSLY DROPPED.
3591          :--
3592 033012      BGNAU
3593 033012      010001      MOV      R0,R1          ; GET UNIT TO BE ADDED (R0)
3594 033014      006301      ASL      R1              ; MAKE IT A WORD INDEX
3595 033016      052761      100000      003302      BIS      #100000,ERTABL(R1) ; SET THE 'ACTIVE' BIT
3596 033024      042761      040000      003302      BIC      #40000,ERTABL(R1) ; CLEAR THE 'DROPPED' BIT
3597 033032      PRINTF      #1$,R0
3598 033054      EXIT      AU
3599 033060      045        116        045      1$: .ASCIZ  /%N% UNIT %D% ADDED/
3600          .EVEN
3601
3602 033106      ENDAU          ; UNUSED.
3603          :++
3604          : THE DROP-UNIT SECTION CONTAINS THE CODING THAT CAUSES A DEVICE
3605          : TO BE REMOVED FROM THE TEST LIST.
3606          :
3607          : SUPVSR DOES THE 'DROPPING'. THIS IS JUST TO TELL THE MAN,
3608          : 'DROPPED' UNITS ARE RE-SELECTED ON OPERATOR 'STA' OR 'ADD',
3609          : COMMAND, OTHERWISE REMAIN INACTIVE. THE 'DISPLAY' COMMAND
3610          : WILL PRINT ALL DROPPED UNITS, AND THE P-TABLES OF THOSE
3611          : WHICH ARE STILL ACTIVE.
3612          : UPON ENTRY, R0 CONTAINS THE UNIT TO BE DROPPED.
3613          :--
3614 033110      BGNDU
3615 033110      012737      177777      003224      MOV      #-1,DUFLG
3616 033116      010001      : MOV      R0,R1
3617          : MOVB     #-1,STIK(R1) ; REMOVE 'JOY-STICK' FLAG.
3618 033120      006301      ASL      R1
3619 033122      052761      140000      003302      BIS      #140000,ERTABL(R1) ; SAY DROPPED
3620 033130      000240      000240      000240      240,240,240 ; ?????????
3621 033136      PRINTF      #1$,R0
3622 033160      EXIT      DU
3623 033164      045        116        045      1$: .ASCIZ  /%N% UNIT %D% DROPPED/
3624          .EVEN
3625 033214      ENDDU
3626          :++
3627          : AUTO-DROP CODE SECTION.
3628          :--
3629 033216      BGNAUTO
3630 033216      ENDAUTO          ; UNUSED.

```

```

3632          .SBTTL  CLEAN-UP AND REPORT CODING SECTIONS
3633
3634          :++
3635          : THE CLEANUP CODING SECTION CONTAINS THE CODING THAT IS
3636          : EXECUTED AT THE END OF EACH PASS (OR SUB-PASS).
3637          : USE TO RETURN DEVICE UNDER TEST TO A NEUTRAL STATE.
3638          :--
3639 033220          BGNCLN
3640 033220 005737 003224      TST      DUFLG      : 'DROPPED' FLAG IS SET ON...
3641 033224 100414          BMI      1$          : ...AND GROSS DPU FAULT...
3642          : ...DON'T TRY TO XCT DPU CODE.
3643
3644 033226 004737 026756          JSR      PC,RELEAS      : CHECK FOR 'HUNG' DPU.
3645 033232 012777 040000 147576      MOV      #GTJSSW,@DXR      : CLEAR PENDING SWITCH INTERRUPT
3646 033240 004737 027454          JSR      PC,WAITF
3647 033244 012777 100000 147566      MOV      #100000,@DYR      : DO SOFT INIT.
3648 033252 004737 027454          JSR      PC,WAITF
3649 033256 005737 003216      1$:     TST      LUTAV
3650 033262 001402          BEQ      2$
3651 033264 005077 147552          CLR      @LSR          : IF LUT, CLEAR STATUS.
3652 033270      2$:     ENDCLN
3653
3654          :++
3655          : THE REPORT CODING SECTION CONTAINS THE
3656          : 'PRINTS' CALLS THAT GENERATE STATISTICAL REPORTS.
3657          :--
3657 033272          BGNRPT
3658 033272          PRINTS #DEVSUM
3659 033312 010246          MOV      R2,-(SP)
3660 033314 010346          MOV      R3,-(SP)
3661 033316 010446          MOV      R4,-(SP)
3662 033320 012704 003302          MOV      #ERTABL,R4      : GET START OF ERROR TABLE.
3663 033324 005003          CLR      R3          : CLEAR UNIT NUMBER
3664 033326 011402      1$:     MOV      (R4),R2      : GET ERROR TABLE ENTRY & TEST IT.
3665 033330 001467          BEQ      4$          : ZERO IF UNIT NOT RUN
3666 033332 100066          BPL      4$
3667 033334 032702 040000          BIT      #BIT14,R2      : WAS UNIT DROPPED?
3668 033340 001015          BNE      2$          : BR IF YES
3669 033342 042702 170000          BIC      #^C7777,R2      : GET ERROR COUNT FIELD
3670 033346          PRINTS #DEVONL,R3,R2      : PRINT
3671 033372 000446          BR      4$
3672 033374 020227 160000      2$:     CMP      R2,#160000      : WAS UNIT NON-EXISTENT?
3673 033400 001012          BNE      3$          : BR IF NO
3674 033402          PRINTS #DEVNXR,R3
3675 033424 000431          BR      4$
3676 033426 020227 160001      3$:     CMP      R2,#160001      : WAS UNIT NOT READY AT STARTUP?
3677 033432 001012          BNE      30$         : BR IF NO.
3678 033434          PRINTS #DEVNRD,R3
3679 033456 000414          BR      4$
3680 033460 042702 170000      30$:    BIC      #^C7777,R2
3681 033464          PRINTS #DEVDRD,R3,R2
3682 033510 062704 000002      4$:     ADD      #2,R4
3683 033514 005203          INC      R3
3684 033516 020427 003502          CMP      R4,#ERTABE
3685 033522 103701          BLO      1$
3686 033524 012604          MOV      (SP)+,R4
3687 033526 012603          MOV      (SP)+,R3
3688 033530 012602          MOV      (SP)+,R2

```

```
3689 033532          ENDRPT          ; UNUSED.
3690
3691
3692 033534      045      116      045  DEVSUM: .ASCIZ  /%N%ADEVICE STATUS SUMMARY:%N/
3693 033571      045      101      040  DEVONL: .ASCIZ  /%A  UNIT %D3%A  ONLINE,  ERRORS = %D%N/
3694 033641      045      101      040  DEVNXR: .ASCIZ  /%A  UNIT %D3%A  DROPPED,  NON-EXISTENT REGISTER%N/
3695 033723      045      101      040  DEVNRD: .ASCIZ  /%A  UNIT %D3%A  DROPPED,  NOT READY AT STARTUP%N/
3696 034004      045      101      040  DEVDRO: .ASCIZ  /%A  UNIT %D3%A  DROPPED,  ERRORS = %D%N/
3697          .EVEN
```

3699		.SBTTL		
3700		.SBTTL	HARDWARE TEST SECTION.	
3701		.SBTTL		
3702		.SBTTL	TEST #	DESCRIPTION
3703		.SBTTL	-----	-----
3704		.SBTTL		
3705	000000	TN=0		
3706		.NLIST	MC,ME	

END TEST 2

```
*****  
*  
* END TEST 3  
*  
*****
```


4009	040134				RDDSR	DSR,R2		
	040134	012777	000000	142672	MOV	#SELDZR,@DSR	:	READ DSR INTO R2 .
	040142	017702	142666		MOV	@DSR,R2		
4010	040146				IFERROR	XRDSE,DUADRE		
	040146	020102			CMP	R1,R2	:	OK?
	040150	001410			BEQ	95\$:	YES.
	040152				HRDERR	XRDSE,DUADRE,	:	NO.
	040172							95\$:
4011	040172	012701	000001		MOV	#1,R1		
4012	040176				RDDSR	PCS,R2		
	040176	012777	000001	142630	MOV	#SELPDS,@DSR	:	READ PCS INTO R2 .
	040204	017702	142624		MOV	@DSR,R2		
4013	040210				IFERROR	XRPCZ,DUADRE		
	040210	020102			CMP	R1,R2	:	OK?
	040212	001410			BEQ	96\$:	YES.
	040214				HRDERR	XRPCZ,DUADRE,	:	NO.
	040234							96\$:
4014	040234	012701	100000		MOV	#100000,R1		
4015	040240				RDDSR	FLG,R2		
	040240	012777	000002	142566	MOV	#SELFLG,@DSR	:	READ FLG INTO R2 .
	040246	017702	142562		MOV	@DSR,R2		
4016	040252				IFERROR	XRFLE,DUADRE		
	040252	020102			CMP	R1,R2	:	OK?
	040254	001410			BEQ	97\$:	YES.
	040256				HRDERR	XRFLE,DUADRE,	:	NO.
	040276							97\$:
4017	040276	012701	000003		MOV	#3,R1		
4018	040302				RDDSR	CSR,R2		
	040302	012777	000003	142524	MOV	#SELCSR,@DSR	:	READ CSR INTO R2 .
	040310	017702	142520		MOV	@DSR,R2		
4019	040314				IFERROR	XRCSE,DUADRE		
	040314	020102			CMP	R1,R2	:	OK?
	040316	001410			BEQ	98\$:	YES.
	040320				HRDERR	XRCSE,DUADRE,	:	NO.
	040340							98\$:
4020	040340	012701	000000		MOV	#0,R1		
4021	040344				RDDSR	MRR,R2		
	040344	012777	000004	142462	MOV	#SELMRR,@DSR	:	READ MRR INTO R2 .
	040352	017702	142456		MOV	@DSR,R2		
4022	040356	042702	170000		BIC	#170000,R2	:	MASK OUT MPM.
4023	040362				IFERROR	XRMRE,DUADRE		
	040362	020102			CMP	R1,R2	:	OK?
	040364	001410			BEQ	99\$:	YES.
	040366				HRDERR	XRMRE,DUADRE,	:	NO.
	040406							99\$:
4024	040406				RDDSR	MPM,R2		
	040406	012777	000004	142420	MOV	#SELMPM,@DSR	:	READ MPM INTO R2 .
	040414	017702	142414		MOV	@DSR,R2		
4025	040420	042702	007777		BIC	#7777,R2	:	MASK OUT MPM.
4026	040424				IFERROR	XRMPE,DUADRE		
	040424	020102			CMP	R1,R2	:	OK?
	040426	001410			BEQ	100\$:	YES.
	040430				HRDERR	XRMPE,DUADRE,	:	NO.
	040450							100\$:
4027	040450				RDDSR	XPM,R2		
	040450	012777	000006	142356	MOV	#SELXPM,@DSR	:	READ XPM INTO R2 .
	040456	017702	142352		MOV	@DSR,R2		

4159
4160 042136

END.TEST

```
*****  
: *  
: * END TEST 5  
: *  
*****
```



```
4333 043406 000137 042654      13$:  JMP      14$           ;USE JMP TO GET BACK FOR LOOP.
4334                          :
4335                          : ON ERROR, SHOW THE FAILING DPU OPCODE.
4336                          :
4337 043412                    20$:  PRINTX #OPCFX,OPC1      ; EXTENSION SHOWS BAD OPCODE.
4338 043436 000207            RTS      PC
4339                          :
4340                          : DISPLAY CODE.
4341                          :
4342 043440 102000 164000 173000 OPC1:  CHAR!LO, DNOP, STOPN    ; DISPLAY CODE.
4343
4344 043446                    END.TEST
```

```
*****
*
*      END TEST 7
*
*****
```


4380
4381 043716

END.TEST

```
*****  
*  
* END TEST 8  
*  
*****
```


:*
:* END TEST 9
:*
:*****

:*
:* END TEST 13
:*
:*****


```
047304 004737 027646 JSR PC,LOOP
047310 103707 BCS 1$
4747 047312 000400 BR 5$ ;...TIL LOOPER EXPIRES.
4748
4749 047314 012777 000014 133512 5$: MOV #SETMRR!0,@DSR ;INSURE MAIN-SEG RELOC IS 0...
4750 047322 012777 000000 133504 MOV #SELDSP,@DSR ;SELECT REAL DSR
4751 047330 004737 030072 JSR PC,TSTEND
4752
4753 047334 END.TEST
```

```
:*****
:
: END TEST 14
:
:*****
```

```
4754 047336 045 116 045 NOTOT: .ASCIZ /%N%A *** MEMORY > 124K. CANNOT TEST TIMEOUT. ***/ ;***B
4755 .EVEN
```


*** TEST 15 ERPOR CODES

SEQ 0130

```

4862 050422           DPSTART #ECRSV2           ;START DPU ON ODD Y VALUE.
      050422 012777 052472 132402  MOV      #ECRSV2,@DPC      ; START THE DPU.
      050430 004737 027454           JSR      PC,WAITF         ; WAIT FOR DISPLAY STOP.
4863 050434 105737 026107  TSTB    INTFLAG          ; DID WE GET THE EXPECTED ERROR INTERRUPT?
4864 050440 001412           BEQ      15$              ; BR IF YES.
4865 050442 012701 050430  MOV      #-12,R1          ; NO. SO GET PC FOR ERROR REPORT,
4866 050446           HRDERR  NTO,INTERR        ; ... AND REPORT THE PROBLEM.
4867 050466           15$:
4868 050466 012701 124103  MOV      #RSVDOP!ENECHK,R1
4869 050472           RDDSRA  CSR,R2
      050472 012777 000003 132334  MOV      #SELCSR,@DSR      ; READ CSR INTO R2 .
      050500 017702 132330  MOV      @DSR,R2
4870 050504           IFERROR  ECE,CSRER
      050504 020102           CMP      R1,R2            ; OK?
      050506 001410           BEQ      69$              ; YES.
      050510           HRDERR  ECE,CSRER,      ; NO.
      050530           69$:
4871 050530           ENDSEG          ; <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<< END SEGMENT.
4872
4873           ; TEST RSVDOP - ODD X VALUE (BIT 0 SET)
4874
4875 050532           BGNSEG          ; <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<< BEGIN SEGMENT.
4876 050534           INITDP
      050534 004737 025402           JSR      PC,DPINIT        ; GO DO SOFT INIT, CHECK THE STATE.
4877 050540 012777 000113 132266  MOV      #ENECHK!WRTCSR,@DSR ;ENABLE RSVD-OP ERROR CHECKING.
4878 050546           INTSET  ERR
      050546 012737 177604 026106  MOV      #^0177400!IOKCKIN!IOKERR,INTMASK ; PRIME FLAG, EXPECT ERR
4879 050554           DPSTART #ECRSV3           ;START DPU ON ODD X VALUE.
      050554 012777 052502 132250  MOV      #ECRSV3,@DPC      ; START THE DPU.
      050562 004737 027454           JSR      PC,WAITF         ; WAIT FOR DISPLAY STOP.
4880 050566 105737 026107  TSTB    INTFLAG          ; DID WE GET THE EXPECTED ERROR INTERRUPT?
4881 050572 001412           BEQ      16$              ; BR IF YES.
4882 050574 012701 050562  MOV      #-12,R1          ; NO. SO GET PC FOR ERROR REPORT,
4883 050600           HRDERR  NTO,INTERR        ; ... AND REPORT THE PROBLEM.
4884 050620           16$:
4885 050620 012701 124103  MOV      #RSVDOP!ENECHK,R1
4886 050624           RDDSRA  CSR,R2
      050624 012777 000003 132202  MOV      #SELCSR,@DSR      ; READ CSR INTO R2 .
      050632 017702 132176  MOV      @DSR,R2
4887 050636           IFERROR  ECE,CSRER
      050636 020102           CMP      R1,R2            ; OK?
      050640 001410           BEQ      70$              ; YES.
      050642           HRDERR  ECE,CSRER,      ; NO.
      050662           70$:
4888 050662           ENDSEG          ; <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<< END SEGMENT.
4889
4890           ; TEST SEQERR - SEQUENCE ERROR (DJMS FROM SUBROUTINE)
4891
4892 050664           BGNSEG          ; <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<< BEGIN SEGMENT.
4893 050666           INITDP
      050666 004737 025402           JSR      PC,DPINIT        ; GO DO SOFT INIT, CHECK THE STATE.
4894 050672           INTSET  ERR
      050672 012737 177604 026106  MOV      #^0177400!IOKCKIN!IOKERR,INTMASK ; PRIME FLAG, EXPECT ERR
4895 050700           DPSTART #ECSEQ1           ;START THE DPU.
      050700 012777 052512 132124  MOV      #ECSEQ1,@DPC      ; START THE DPU.
      050706 004737 027454           JSR      PC,WAITF         ; WAIT FOR DISPLAY STOP.
4896 050712 105737 026107  TSTB    INTFLAG          ; DID WE GET THE EXPECTED ERROR INTERRUPT?

```



```
4970 051476 017702 131332      MOV    @DSR,R2
      051502                     IFERROR ECE,CSRER
      051502 020102              CMP    R1,R2
      051504 001410              BEQ    75$                       ; OK?
      051506                     HRDERR  ECE,CSRER,      ; YES.
      051526                     ; NO.
4971 051526      75$:
      4972      ENDSEG                ; <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<< END SEGMENT.
      4973
      4974
      4975      ; TEST SEQERR - SEQUENCE ERROR (DMA READBACK FROM SUBROUTINE)
      4976 051530      BGNSEG          ; <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<< BEGIN SEGMENT.
      4977 051532      INITDP
      4978 051532 004737 025402      JSR    PC,DPINIT          ; GO DO SOFT INIT, CHECK THE STATE.
      051536      INTSET   ERR
      4979 051536 012737 177604 026106      MOV    #^0177400!IOKCKIN!IOKERR,INTMASK  ; PRIME FLAG, EXPECT ERR
      051544      DPSTART  #ECSEQ6
      051544 012777 052676 131260      MOV    #ECSEQ6,@DPC       ; START THE DPU.
      051552 004737 027454      JSR    PC,WAITF           ; WAIT FOR DISPLAY STOP.
      4980 051556 105737 026107      TSTB  INTFLAG           ; DID WE GET THE EXPECTED ERROR INTERRUPT?
      4981 051562 001412      BEQ    22$                       ; BR IF YES.
      4982 051564 012701 051552      MOV    #-12,R1           ; NO. SO GET PC FOR ERROR REPORT,
      4983 051570      HRDERR  NTO,INTERR      ; ... AND REPORT THE PROBLEM.
      4984 051610      22$:
      4985 051610 012701 120003      MOV    #SEQERR,R1
      4986 051614      RDDSRA  CSR,R2
      051614 012777 000003 131212      MOV    #SELCSR,@DSR      ; READ CSR INTO R2 .
      051622 017702 131206      MOV    @DSR,R2
      4987 051626      IFERROR ECE,CSRER
      051626 020102              CMP    R1,R2
      051630 001410              BEQ    76$                       ; OK?
      051632                     HRDERR  ECE,CSRER,      ; YES.
      051652                     ; NO.
      4988 051652      76$:
      4989      ENDSEG                ; <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<< END SEGMENT.
      4990
      4991      ; TEST SEQERR - SEQUENCE ERROR (ILLEGAL DMA READBACK RESTART)
      4992
      4993 051654      BGNSEG          ; <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<< BEGIN SEGMENT.
      4994 051656      INITDP
      4995 051656 004737 025402      JSR    PC,DPINIT          ; GO DO SOFT INIT, CHECK THE STATE.
      051662      INTSET   ERR
      4996 051662 012737 177604 026106      MOV    #^0177400!IOKCKIN!IOKERR,INTMASK  ; PRIME FLAG, EXPECT ERR
      051670      DPSTART  #ECSEQ7
      051670 012777 052726 131134      MOV    #ECSEQ7,@DPC       ; START THE DPU.
      051676 004737 027454      JSR    PC,WAITF           ; WAIT FOR DISPLAY STOP.
      4997 051702 105737 026107      TSTB  INTFLAG           ; DID WE GET THE EXPECTED ERROR INTERRUPT?
      4998 051706 001412      BEQ    23$                       ; BR IF YES.
      4999 051710 012701 051676      MOV    #-12,R1           ; NO. SO GET PC FOR ERROR REPORT,
      5000 051714      HRDERR  NTO,INTERR      ; ... AND REPORT THE PROBLEM.
      5001 051734      23$:
      5002 051734 012701 120003      MOV    #SEQERR,R1
      5003 051740      RDDSRA  CSR,R2
      051740 012777 000003 131066      MOV    #SELCSR,@DSR      ; READ CSR INTO R2 .
      051746 017702 131062      MOV    @DSR,R2
      5004 051752      IFERROR ECE,CSRER
      051752 020102              CMP    R1,R2
                                      ; OK?
```


5085										
5086	052472	114000	000000	000001	ECRSV2:	APNT,0,1				: RSV DOP - ODD Y VALUE.
5087	052500	173000				STOPN				
5088										
5089	052502	114000	000001	000000	ECRSV3:	APNT,1,0				: RSV DOP - ODD X VALUE.
5090	052510	173000				STOPN				
5091										
5092	052512	160001	052520		ECSEQ1:	DJMS,1\$: SEQERR - SEQUENCE ERROR.
5093	052516	173000				STOPN				
5094	052520	160001	052524		1\$:	DJMS,2\$:DJMS FROM SUBROUTINE.
5095	052524	173000			2\$:	STOPN				
5096	052526	173000				STOPN				
5097										
5098	052530	114000	000000	000000	ECSEQ2:	APNT,0,0				: SEQERR.
5099	052536	152000	052550			SETCB,2\$				
5100	052542	103774				CHAR!ALL				
5101	052544	000000			1\$:	0				
5102	052546	173000				STOPN				
5103	052550	000000			2\$:	0				:DATA AS 1ST WORD -- STILL IN CHAR MODE.
5104	052552	000000	000000	000000		0,0,0,0,0,0,0,0,0				
5105	052572	165000				DPOP				
5106	052574	173000				STOPN				
5107										
5108	052576	114000	000000	000000	ECSEQ3:	APNT,0,0				: SEQERR.
5109	052604	152000	052616			SETCB,2\$				
5110	052610	103774				CHAR!ALL				
5111	052612	000000			1\$:	0				
5112	052614	173000				STOPN				
5113	052616	160001	052624		2\$:	DJMS,3\$:DJMS IN CHAR SUBROUTINE.
5114	052622	173000				STOPN				
5115	052624	173000			3\$:	STOPN				
5116	052626	173000				STOPN				
5117										
5118	052630	170140			ECSEQ4:	CLRMEM				: SEQERR.
5119	052632	114000	000000	000000		APNT,0,0				
5120	052640	160001	052646			DJMS,1\$				
5121	052644	173000				STOPN				
5122	052646	134000			1\$:	BM04				
5123	052650	052650			2\$:	2\$				
5124	052652	173000				STOPN				
5125										
5126	052654	114000	000000	000000	ECSEQ5:	APNT,0,0				: SEQERR.
5127	052662	160001	052670			DJMS,1\$				
5128	052666	173000				STOPN				
5129	052670	136000			1\$:	BM14				
5130	052672	052672			2\$:	2\$				
5131	052674	173000				STOPN				
5132										
5133	052676	114000	000000	000000	ECSEQ6:	APNT,0,0				: SEQERR.
5134	052704	160001	052712			DJMS,1\$				
5135	052710	173000				STOPN				
5136	052712	162000			1\$:	IMREAD				:DMA READBACK FROM SUBROUTINE.
5137	052714	000001				1				
5138	052716	000001				1				
5139	052720	000001				1				
5140	052722	052750				ECSQ7B				
5141	052724	173000				STOPN				

5142
5143 052726 114000 000000 000000 ECSEQ7: APNT,0,0
5144 052734 162001 IMREAD!1 ;ILLEGAL DMA READBACK RESTART.
5145 052736 000001 1
5146 052740 052750 ECSQ7B
5147 052742 173000 STOPN
5148 052744 162101 IMREAD!101
5149 052746 173000 STOPN
5150 052750 ECSQ7B: .BLKW 2
5151
5152 052754 176000 ECSTO: PROTEC!CH0
5153 052756 176400 PROTEC!CH1
5154 052760 177000 PROTEC!CH2
5155 052762 177400 PROTEC!CH3
5156 052764 114000 040000 000000 APNT,I!0,0
5157 052772 173000 STOPN
5158
5159 052774 END.TEST

:*****
: *
: * END TEST 15
: *
:*****


```

5321
5322 054036
5323 054064 000207
5324
5325 054066 054110
5326
5327
5328
5329 054070 164000 164000 164000
5330 054076 152000 054110
5331 054102 160000
5332 054104 000000
5333 054106 173000
5334
5335 054110 054112
5336 054112 173000
5337
5338 054114 152000
5339 054116 054110
5340 054120 165000
5341
5342
5343
5344 054122 176074 176474 177074
5345
5346 054132 116000
5347 054134 170100
5348 054136 152000 110302
5349 054142 117774 001000 001500
5350 054150 100000
5351 054152 040 041 042
5352
5353 054172 114000 001000 001400
5354 054200 100000
5355 054202 060 061 062
5356 054214 114000 001000 001300
5357 054222 100000
5358 054224 072 073 074
5359
5360 054234 114000 001000 001200
5361 054242 100000
5362 054244 101 102 103
5363
5364 054262 114000 001000 001100
5365 054270 100000
5366 054272 116 117 120
5367
5368 054310 114000 001000 001000
5369 054316 100000
5370 054320 133 134 135
5371
5372
5373 054326 117774 001000 000602
5374 054334 100000
5375 054336 040 041 042
5376
5377 054356 114000 001000 000502

```

```

:20$: PRINTX #CHRFX,CT.BAS,R1 ; ERROR EXTENSION.
RTS PC
:
:CHSEG: CT.SUB ;CHARACTER BASE CONTROL, WITH SEGMENT SELECT.
:
: DISPLAY CODE FOR CHAR MODE TEST.
:
CT: DNOP,DNOP,DNOP ;INITIAL BASE ADDR = CT.SUB.
SETCB,CT.SUB ;XFER PC TO THE CHAR...
DJMP ;...DATA ARRAY.
CT.ASC: 0 ;FINAL 'DPOP' SHOULD RETURN HERE
CT.RET: STOPN
CT.SUB: .+2 ;SUBROUTINE ENTRY POWNTER.
STOPN ;STOP HERE IF CHAR ADDRESSES...
;...COME OUT RIGHT.
CT.BAS: SETCB ;ON RESUME, SET NEW BASE ADDR...
CT.SUB ;...AND DPOP TO NEXT CHAR.
DPOP
: DISPLAY FILE FOR CHARACTER DISPLAY:
CHDISP: RDWRT!CH0,RDWRT!CH1,RDWRT!CH2,RDWRT!CH3
:***B CLRMEM
APNT!LO ;***B
SETMEM ;***B
SETCB, ACAT
APNT!ALL,1000,1500
CHAR
.ASCII ? !'#$%&'()*+,-./?
.EVEN
APNT,1000,1400
CHAR
.ASCII /0123456789/
APNT,1000,1300
CHAR
.ASCII /:;<=>?@ /
.EVEN
APNT,1000,1200
CHAR
.ASCII /ABCDEFGHIJKLM /
.EVEN
APNT,1000,1100
CHAR
.ASCII /NOPQRSTUVWXYZ /
.EVEN
APNT,1000,1000
CHAR
.ASCII /[ \]^_ /
.EVEN
APNT!ALL,1000,602
CHAR
.ASCII ? !'#$%&'()*+,-./?
.EVEN
APNT,1000,502

```


5378	054364	100000				CHAR
5379	054366	060	061	062		.ASCII /0123456789/
5380	054400	114000	001000	000402		APNT,1000,402
5381	054406	100000				CHAR
5382	054410	072	073	074		.ASCII /:;<=>?@ /
5383						.EVEN
5384	054420	114000	001000	000302		APNT,1000,302
5385	054426	100000				CHAR
5386	054430	101	102	103		.ASCII /ABCDEFGHIJKLM /
5387						.EVEN
5388	054446	114000	001000	000202		APNT,1000,202
5389	054454	100000				CHAR
5390	054456	116	117	120		.ASCII /NOPQRSTUVWXYZ /
5391						.EVEN
5392	054474	114000	001000	000102		APNT,1000,102
5393	054502	100000				CHAR
5394	054504	133	134	135		.ASCII /[\] ^ _ /
5395						.EVEN
5396	054512	173000				STOPN
5397						
5398	054514					END.TEST

*
* END TEST 17
*

```

5862 057752 114000 000000 000000 X2INC: APNT,0,0 ; X2 INCREMENT.
5863 057760 173000 STOPN
5864 057762 114000 040000 000000 X2INCA: APNT,1!0,0
5865 057770 144100 RNLN!DBLP!X
5866 057772 000777 X2INCB: 000777
5867 057774 173000 STOPN
5868 057776 160000 057762 DJMP,X2INCA
5869
5870 060002 114000 000000 000000 YUP: APNT,0,0 ; Y UP.
5871 060010 144000 000377 RNLN,377
5872 060014 173000 STOPN
5873
5874 060016 114000 000000 000002 YDN: APNT,0,2 ; Y DOWN.
5875 060024 144040 000377 RNLN!YDOWN,377
5876 060030 173000 STOPN
5877
5878 060032 114000 000000 000000 Y2UP: APNT,0,0 ; Y SKIP UP.
5879 060040 144400 000377 RNLN!RLSKIP,377
5880 060044 173000 STOPN
5881
5882 060046 114000 000000 000004 Y2DN: APNT,0,4 ; Y SKIP DOWN.
5883 060054 144440 000377 RNLN!RLSKIP!YDOWN,377
5884 060060 173000 STOPN
5885
5886 060062 END.TEST
    
```

```

:*****
: *
: * END TEST 22
: *
:*****
    
```

5888

```
*****
:
: BEGIN TEST 23 - CURSOR REGISTERS/SWITCH/MATCH
:
:*****
```

: (ITERATION COUNT = 3.)

5889
5890
5891
5892
5893
5894
5895
5896
5897
5898
5899
5900
5901
5902
5903
5904
5905
5906
5907
5908
5909
5910
5911
5912
5913
5914
5915
5916
5917
5918
5919
5920
5921
5922
5923
5924
5925 060152 005737 002502
5926 060156 001402
5927 060160 000137 065016

```
: THIS TEST, USING THE LOADABLE CURSOR POSITION REGISTERS AND 'SOFT' SWITCH,  
: PERFORMS THE FOLLOWING CHECKS ON THE CURSOR REGISTERS, JOYSTICK SWITCH, AND  
: CURSOR MATCH AND ASSOCIATED LOGIC FOR EACH AVAILABLE SYNC GENERATOR CHANNEL:
```

- (1) WRITE/READ TEST OF THE CURSOR POSITION REGISTERS USING DATA OF ALL 1'S AND ALL 0'S, USING PROGRAMMED I/O (VIA DXR AND DYR) TO CHECK FOR STUCK 0'S, STUCK 1'S AND DUAL ADDRESSING.
- (2) WRITE/READ TEST OF THE CURSOR POSITION REGISTERS VIA THE DISPLAY FILE INSTRUCTION LDPC (LOAD CURSOR POSITION) AND CURD (CURSOR READ), USING AN INCREMENTING COUNT PATTERN ON THE X POSITION (0 - 3776) AND A DECREMENTING COUNT PATTERN ON THE Y POSITION (3776 - 0).
- (3) WRITE/READ TEST OF THE SWITCH ENABLE, MATCH ENABLE, AND CROSSHAIR INTENSITY ENABLE BITS USING PROGRAMMED I/O TO SET AND CLEAR ALL BITS.
- (4) WRITE/READ TEST OF THE ENABLE BITS VIA DISPLAY FILE INSTRUCTION LDJSS (LOAD JOYSTICK STATUS), USING ALL COMBINATIONS OF LOAD-ENABLES AND DATA.
- (5) SET THE SWITCH ENABLE AND MATCH ENABLE BITS, THEN PERFORM A WRITE TO THE JOYSTICK STATUS REGISTER OF ANOTHER CHANNEL WITH DATA CONFIGURED TO SET THE ENABLE BITS; VERIFY THAT THE ENABLES OF THE CHANNEL UNDER TEST WERE CLEARED. FAILURE HERE INDICATES A FAULTY CHANNEL ADDRESS DECODER IN THE SYNC CHANNEL UNDER TEST.
- (6) TEST OPERATION OF THE JOYSTICK SWITCH INTERRUPT.
- (7) TEST OPERATION OF THE CURSOR MATCH INTERRUPT. FLOATING 1'S AND 0'S PATTERNS ARE USED IN THE CURSOR X AND Y POSITIONS TO VERIFY THE A MATCH OCCURS ON THE SELECTED POINT WHILE DRAWING A VECTOR THROUGH IT, AND THAT A MATCH DOES NOT OCCUR WHILE DRAWING VECTORS AROUND THE SELECTED POINT PARALLEL TO THE X AND Y AXES.

```
TST      DPUMOD      : DPU-MODE ONLY?  
BEQ      CSMTST      : BR IF NOT -- OK TO EXECUTE.  
JMP      CSMXIT      : YES -- JUST GO TO EXIT.
```



```

6025
6026                ;CHECK DUAL ADDRESSING -- VERIFY THAT X NOT CHANGED BY WRITING Y.
6027
6028 060672 012701 103776 CSMT4: MOV    #BIT15!*03776,R1 ; EXPECT CURSOR X TO BE ALL 1'S
6029 060676 012703 100000      MOV    #BIT15!0,R3    ; EXPECT CURSOR Y TO BE 0
6030 060702      BISW3  CSCH.11,#WRTCPX,R5 ; SET UP WRITE CURSOR W/ CHANNEL
      060702 012705 140000      MOV    #WRTCPX,R5
      060706 053705 065146      BIS    CSCH.11,R5

6031
6032 060712 1$:    BGNSEG          ; <<<<<<<<<<<<<<<<<<<<<<<<<BEGIN SEGMENT
6033 060714 013700 065150      MOV    CSRETRY,R0      ; @@@R
6034 060720      BISW3  R5,R1,@DXR      ; LOAD CURSOR X WITH A PATTERN.
      060720 010146      MOV    R1,-(SP)
      060722 050516      BIS    R5,(SP)
      060724 012677 122106      MOV    (SP)+,@DXR
6035 060730 004737 027454      JSR    PC,WAITF
6036 060734      BISW3  R5,R3,@DYR      ; LOAD OPPOSITE PATTERN INTO CURSOR Y.
      060734 010346      MOV    R3,-(SP)
      060736 050516      BIS    R5,(SP)
      060740 012677 122074      MOV    (SP)+,@DYR
6037 060744 004737 065474      JSR    PC,GETCURS      ; GET THE CURSOR INTO R2 & R4.
6038 060750      CMP    R1,R2          ; CHECK DXR (X-CURSOR)
6039 060752 001002      BNE    2$              ; BR IF ERROR
6040 060754 020304      CMP    R3,R4          ; CHECK DYR (Y-CURSOR)
6041 060756 001412      BEQ    3$              ; BR IF OK.
6042 060760      2$:    DEC    R0          ; DECREMENT THE RETRY COUNTER ;@@@R
6043 060762 100356      BPL    10$            ; GO TRY AGAIN IF NOT NEGATIVE ;@@@R
6044 060764      HRDERR CRDAY,RLXYE      ; REPORT THE ERROR.
6045 061004      3$:    ENDSEG          ; >>>>>>>>>>>>>END SEGMENT
6046
6047 061006 032703 003776      BIT    #*03776,R3      ; DID WE WRITE 1'S INTO Y YET?
6048 061012 001004      BNE    4$              ; BR IF YES, SO WE'RE DONE.
6049 061014 010103      MOV    R1,R3          ; NO, SO SWAP PATTERNS.
6050 061016 012701 100000      MOV    #BIT15,R1      ; ... SET UP TO WRITE X TO 0.
6051 061022 000733      BR    1$              ; LOOP BACK TO DO IT.
6052
4$:
  
```

```

6054
6055 ; DO WRITE/READ TEST USING DISPLAY FILE.
6056
6057 061024 005001 CSMT5: CLR R1 ; INIT THE X PATTERN TO 0
6058 061026 012703 003776 MOV #^03776,R3 ; INIT THE Y PATTERN TO ALL 1'S
6059 061032 012746 175000 BISW3 CSCH.6,#LDCP,CSMF1A ; SET UP THE DISPLAY FILE
        061032 053716 065142 MOV #LDCP,-(SP)
        061036 012637 065200 BIS CSCH.6,(SP)
        061042 012637 065200 MOV (SP)+,CSMF1A
6060 061046 012746 146100 BISW3 CSCH.8,#CURD,CSMF1B ; ... BY INSERTING THE CHANNEL.
        061046 053716 065144 MOV #CURD,-(SP)
        061052 053716 065144 BIS CSCH.8,(SP)
        061056 C12637 065206 MOV (SP)+,CSMF1B
6061
6062 061062 010137 065202 1$: MOV R1,CSMF1A+2 ; INSERT X COORD. TO BE WRITTEN
6063 061066 010337 065204 MOV R3,CSMF1A+4 ; ... AND THE Y COORDINATE.
6064 061072 013705 065150 MOV CSRETRY,R5
6065 061076 BGNSEG ; <<<<<<<<<<<<<<<<<<<<<<<<< BEGIN SEGMENT
6066 061100 012777 065200 121724 2$: DPSTART #CSMF1A ; EXECUTE LOAD THEN READ OF CURSOR.
        061100 004737 027454 MOV #CSMF1A,@DPC ; START THE DPU.
        061106 052701 100000 JSR PC,WAITF ; WAIT FOR DISPLAY STOP.
        061112 052703 100000 BIS #BIT15,R1 ; EXPECT THE CURSOR FLAG BIT IN DXR
        061116 004737 065520 BIS #BIT15,R3 ; ... AND DYR.
        061122 020102 JSR PC,RDDXY ; GET MASKED DXR AND DYR INTO R2 AND R4.
        061126 001002 CMP R1,R2 ; CHECK DXR FOR X CURSOR.
        061130 020304 BNE 3$ ; BR IF INCORRECT.
        061132 001412 CMP R3,R4 ; CHECK DYR FOR Y CURSOR.
        061134 005305 BEQ 4$ ; BR IF OK.
        061136 100357 3$: DEC R5
        061140 BPL 2$
        061142 HRDERR CRWRE,RLXYE ; ERROR -- DXR OR DYR WRITE/READ ERROR.
        061144 4$: ENDSEG ; >>>>>>>>>>>>>>>>>>>>>>>>> END SEGMENT
6078
6079 061164 062701 100002 ADD #2!BIT15,R1 ; INCREMENT THE X PATTERN.
6080 061170 162703 100002 SUB #2!BIT15,R3 ; DECREMENT THE Y PATTERN.
6081 061174 032701 003776 BIT #^03776,R1 ; DID WE INCREMENT X AROUND TO 0?
6082 061200 001330 BNE 1$ ; LOOP BACK IF NO.
6083 061202 30$:

```



```
6170 061654                HRDERR  CEZWE,RLXYE      ; REPORT THE ERROR.
6171 061674                43$:
6172 061674 012701 114000  MOV      #BIT15!JSSIES!JSMIES,R1 ; ASSUME SWITCH & MATCH ENABLES ARE 1.
6173 061700 012703 110000  MOV      #BIT15!CHIE,R3           ; ASSUME CROSSHAIR ENABLE IS 1.
6174
6175 061704 032705 000002  BIT      #JSWE,R5                ; ENABLED TO WRITE SWITCH ENABLE?
6176 061710 001405        BEQ      5$                      ; BR IF NO -- LEAVE EXPECTED ALONE.
6177 061712 032705 000001  BIT      #JSWD,R5                ; WRITE TO 0?
6178 061716 001002        BNE      5$                      ; BR IF NO -- LEAVE AT 1.
6179 061720 042701 004000  BIC      #JSSIES,R1              ; YES -- CLEAR EXPECTED STATUS.
6180
6181 061724                5$:
6182 061724 032705 000010  BIT      #JMWE,R5                ; ENABLED TO WRITE MATCH ENABLE?
6183 061730 001405        BEQ      6$                      ; BR IF NO -- LLAVE EXPECTED ALONE.
6184 061732 032705 000004  BIT      #JMWD,R5                ; WRITE TO 0?
6185 061736 001002        BNE      6$                      ; BR IF NC -- LEAVE AT 1.
6186 061740 042701 010000  BIC      #JSMIES,R1              ; YES -- CLEAR EXPECTED STATUS.
6187 061744                6$:
6188 061744 032705 000040  BIT      #JCWE,R5                ; ENABLED TO WRITE CROSSHAIR ENABLE?
6189 061750 001407        BEQ      7$                      ; BR IF NO -- LEAVE EXPECTED ALONE.
6190 061752 032705 000020  BIT      #JCWD,R5                ; WRITE TO 0?
6191 061756 001004        BNE      7$                      ; BR IF NO -- LEAVE AT 1.
6192 061760 042703 010000  BIC      #CHIE,R3                ; YES -- CLEAR EXPECTED STATUS.
6193 061764 013700 065150  MOV      CSRETRY,R0              ; @@@R
6194 061770                7$:
6195 061770                DPCONT ; RESTART, TO SET ALL, THEN DO GENERAL WRITE.
        061770 052777 000001 121034 BIS      #BIT0,@DPC              ; CONTINUE THE DPU.
        061776 004737 027454        JSR      PC,WAITF                 ; WAIT FOR DISPLAY STOP.
6196 062002 004737 027454        JSR      PC,WAITF                 ; WAIT FOR DONE.
6197 062006 004737 065566        JSR      PC,RDJSE                 ; GET DXR, DYR INTO R2, R4.
6198 062012 020102        CMP      R1,R2                   ; CHECK DXR (X-CURSOR)
6199 062014 001002        BNE      72$                     ; BR IF ERROR
6200 062016 020304        CMP      R3,R4                   ; CHECK DYR (Y-CURSOR)
6201 062020 001412        BEQ      73$                     ; BR IF OK.
6202 062022 005300        72$: DEC      R0                       ; DECREMENT THE RETRY COUNTER ;@@@R
6203 062024 100361        BPL      7$                       ; GO TRY AGAIN IF NOT NEGATIVE ;@@@R
6204 062026                HRDERR  CESWE,RLXYE      ; REPORT THE ERROR.
6205
6206 062046                73$: ENDSEG ;>>>>>>>>>>>END SEGMENT
6207 062050 005237 065214        INC      CSMF2B                  ; INCREMENT THE LDJSS WRITE DATA.
6208 062054 032737 000077 065214 BIT      #*077,CSMF2B            ; DONE ALL COMBOS?
6209 062062 001402        BEQ      8$                      ; BR TO EXIT IF YES.
6210 062064 000137 061522        JMP      1$                      ; NO, SO LOOP BACK FOR ANOTHER ROUND.
6211 062070                8$:
```



```
6292  
6293 ;(3) SET SWITCH AND ENABLE, SEE PENDING INTERRUPT  
6294  
6295 062450 CSMT12: BGNSEG ;<<<<<<<<<<<<<<<<<<<BEGIN SEGMENT  
6296 062452 INITDP  
6297 062452 004737 025402 JSR PC,DPINIT ; GO DO SOFT INIT, CHECK THE STATE.  
6298 062456 004737 026176 JSR PC,DSBINT  
062462 012746 042003 BISW3 CSCH.8,#WJSS!JSWE!JSWD,@DXR ; ENABLE SWITCH INTERRUPT.  
062466 053716 065144 MOV #WJSS!JSWE!JSWD,-(SP)  
062472 012677 120340 BIS CSCH.8,(SP)  
6299 062476 004737 027454 MOV (SP)+,@DXR  
6300 062502 JSR PC,WAITF  
062506 012746 160002 BISW3 CSCH.8,#WECC!SWCHON,@DYR ; TURN THE SWITCH ON.  
062506 053716 065144 MOV #WECC!SWCHON,-(SP)  
062512 012677 120322 BIS CSCH.8,(SP)  
6301 062516 004737 027454 MOV (SP)+,@DYR  
6302 062522 012701 100016 JSR PC,WAITF  
6303 062526 RD DSRA #CHAR!JSLCKO!PINTRB!PVEC2,R1 ; EXPECT FLAGS TO SHOW PENDING.  
062526 012777 000002 120300 MOV FLG,R2 ; READ THE FLAGS REGISTER INTO R2.  
062534 017702 120274 MOV #SELF, @DSR ; READ FLG INTO R2 .  
6304 062540 MOV @DSR,R2  
062540 020102 IFERRR NOSWPI,EXPREC  
062542 001410 CMP R1,R2 ; OK?  
062544 BEQ 64$ ; YES.  
062564 HRDERR NOSWPI,EXPREC, ; NO.  
6305 062564 64$:  
6306 062564 1$:  
062564 012746 160000 BISW3 CSCH.8,#WECC,@DYR ; TURN OFF THE SWITCH.  
062570 053716 065144 MOV #WECC,-(SP)  
062574 012677 120240 BIS CSCH.8,(SP)  
6307 062600 004737 027454 MOV (SP)+,@DYR  
6308 062604 JSR PC,WAITF  
062604 012746 042002 BISW3 CSCH.8,#WJSS!JSWE!0,@DXR  
062610 053716 065144 MOV #WJSS!JSWE!0,-(SP)  
062614 012677 120216 BIS CSCH.8,(SP)  
6309 062620 004737 027454 MOV (SP)+,@DXR  
6310 062624 012777 040000 120204 JSR PC,WAITF  
6311 062632 004737 027454 MOV #GTJSSW,@DXR ;DISMISS INTR REQ, IF ANY  
6312 JSR PC,WAITF  
6313 062636 WTDSRA #0,FLG ; CLEAR THE FLAGS AND INTERRUPT.  
062636 012777 000012 120170 MOV #0^C17!SETFLG,@DSR  
6314 062644 004737 023422 JSR PC,IIRCHK ; CHECK THE INTERNAL REGISTERS  
6315 062650 2$:  
6316 062652 004737 026074 ENDSEG ;>>>>>>>>>>>>>END SEGMENT  
JSR PC,DPRESÉT ; DO SOFT INITIALIZE.
```



```

6345
6346                 ;(5) CHECK SWITCH INTERRUPT OPERATION FROM DISPLAY FILE.
6347
6348 063020          CSMT14: BGNSEG                ;<<<<<<<<<<<<<<<BEGIN SEGMENT
6349 063022          INITDP
6350 063022 004737 025402  JSR PC,DPINIT          ; GO DO SOFT INIT, CHECK THE STATE.
6351 063026 004737 026114  JSR PC,ENAINT         ; ENABLE INTERRUPTS.
6352 063032 012746 175000  BISW3 CSCH.6,#LDPC,CSMF3A   ; SET UP DISPLAY FILE TO LOAD CURSOR,
6353 063036 053716 065142  MOV #LDPC,-(SP)
6354 063042 012637 065230  BIS CSCH.6,(SP)
6355 063046 012746 146013  MOV (SP)+,CSMF3A
6356 063052 053716 065144  BISW3 CSCH.8,#CUIS,CSMF3B ; ... AND SET SWITCH ENABLE
6357 063056 012637 065236  MOV #CUIS,-(SP)
6358 063062 012746 175402  BIS CSCH.8,(SP)
6359 063066 053716 065142  MOV (SP)+,CSMF3B
6360 063072 012637 065240  BISW3 CSCH.6,#LDECC!SWCHON,CSMF3C ; ... AND TURN ON SWITCH.
6361 063076 112737 000377  MOV #LDECC!SWCHON,-(SP)
6362 063104 012777 065230 26107  MOVB #^0377,INTFLAG ; PRIME THE INTERRUPT FLAG.
6363 063112 004737 027454 117720  DPSTART #CSMF3A ; START THE DPU.
6364 063116 004737 027454  MOV #CSMF3A,@DPC ; START THE DPU.
6365 063122 017702 117704  JSR PC,WAITF ; WAIT FOR DISPLAY STOP.
6366 063126 105737 026107  JSR PC,WAITF ; WAIT FOR STOP.
6367 063132 001412  MOV @DPC,R2 ; SAVE THE DPC.
6368 063134 012701 063116  TSTB INTFLAG ; TEST FOR GOOD INTERRUPT.
6369 063140 020227 065252  BEQ 1$ ; BR IF GOOD.
6370 063164 103412  MOV #.-16,R1
6371 063166 012701 065244  HRDERR NOSWI,INTERR
6372 063212 013701 065232 1$: CMP R2,#CSMF3D ; DID THE DPC HIT THE STOPN?
6373 063216 052701 100000  BLO 2$ ; BR IF NO -- OK
6374 063222 013703 065234  MOV #CSMF3C+4,R1 ; EXPECT A DPC PRIOR TO STOPN.
6375 063226 052703 100000  HRDERR SWNOHLT,EXPREC
6376 063232 005737 065150 2$: BISW3 #BIT15,CSMF3A+2,R1 ; EXPECT DXR TO HAVE X CURSOR.
6377 063236 001416  MOV CSMF3A+2,R1
6378 063240 004737 065520  BIS #BIT15,R1
6379 063244 020102  MOV #BIT15,CSMF3A+4,R3 ; EXPECT DYR TO HAVE Y CURSOR.
6380 063246 001002  TST CSRETRY
6381 063250 020304  BEQ 3$
6382 063252 001410  JSR PC,RDDXY ; GET MASKED DXR, DYR INTO R2,R4.
6383 063254  IFERRX2 SWNOCU,RLXYE
6384 063274 004737 026176  CMP R1,R2 ; 1ST ITEM OK?
6385 063300 004737 026074  BNE 65$ ; NO.
6386 063302  CMP R3,R4 ; 2ND ITEM OK?
6387 063302  BEQ 64$ ; YES.
6388 063302  HRDERR SWNOCU,RLXYE, ; NO.
6389 063302 65$:
6390 063302 64$:
6391 063302 3$: JSR PC,DSBINT
6392 063302  ENDSEG ;>>>>>>>>>>>>>>>END SEGMENT
6393 063302  JSR PC,DPRESET ; DO SOFT INIT.

```


6420	063626	013705	065150		MOV	CSRETRY,R5	
6421	063632	013701	065326		MOV	CSMF4X,R1	
6422	063636	013703	065330		MOV	CSMF4Y,R3	
6423	063642	052701	100000		BIS	#BIT15,R1	
6424	063646	052703	100000		BIS	#BIT15,R3	
6425	063652				BISW3	CSCH.8,#WECC,@DYR	; CLEAR THE SWITCH
	063652	012746	160000		MOV	#WECC,-(SP)	
	063656	053716	065144		BIS	CSCH.8,(SP)	
	063662	012677	117152		MOV	(SP)+,@DYR	
6426	063666	004737	027454		JSR	PC,WAITF	
6427	063672	112737	000377	026107	MOVB	#^0377,INTFLAG	; PRIME THE INTERRUPT FLAG
6428	063700				DPSTART	#CSMF4D	
	063700	012777	065334	117124	MOV	#CSMF4D,@DPC	; START THE DPU.
	063706	004737	027454		JSR	PL,WAITF	; WAIT FOR DISPLAY STOP.
6429	063712	017702	117114		MOV	@DPC,R2	
6430	063716	105737	026107		TSTB	INTFLAG	; ANY INTERRUPTS?
6431	063722	002414			BLT	403\$	
6432	063724	010146			MOV	R1,-(SP)	
6433	063726	012701	063710		MOV	#-14,R1	
6434	063732				HRDERR	IFAUULT,INTERR	
6435	063752	012601			MOV	(SP)+,R1	
6436	063754			403\$:			
6437	063754	004737	065520		JSR	PC,RDDXY	
6438	063760	020102			CMP	R1,R2	
6439	063762	001002			BNE	44\$	
6440	063764	020304			CMP	R3,R4	
6441	063766	001414			BEQ	45\$	
6442	063770	005305		44\$:	DEC	R5	
6443	063772	100337			BPL	43\$	
6444	063774				HRDERR	CRWRE,RLXYE	
6445	064014	000137	064710		JMP	8\$; GO TRY THE NEXT COORDINATES.
6446	064020			45\$:			
6447	064020	112737	000202	026106	MOVB	#I0KCKIN!I0KJSM,INTMASK	; SAY WE'LL CHECK FOR MATCH
6448	064026	112737	000377	026107	MOVB	#^0377,INTFLAG	; PRIME THE INTERRUPT FLAG.
6449	064034				DPCONT		
	064034	052777	000001	116770	BIS	#BIT0,@DPC	; CONTINUE THE DPU.
	064042	004737	027454		JSR	PC,WAITF	; WAIT FOR DISPLAY STOP.
6450	064046	012701	000100		MOV	#100,R1	; SETUP A WATCHDOG COUNTER.
6451	064052	105737	026107		TSTB	INTFLAG	; DID WE GET AN INTERRUPT?
6452	064056	100002		40\$:	BPL	41\$; BR IF YES.
6453	064060	005301			DEC	R1	
6454	064062	003373			BGT	40\$	
6455	064064	017702	116742		MOV	@DPC,R2	; GET DPC FOR ERROR.
6456	064070	017737	116742	003170	MOV	@DXR,SDXR	
6457	064076	017737	116736	003172	MOV	@DYR,SDYR	
6458	064104	105737	026107		TSTB	INTFLAG	
6459	064110	001417			BEQ	5\$; WAS IT A GOOD INTERRUPT?
6460	064112	002434			BLT	50\$; BR IF NO INTERRUPT OCCURRED.
6461	064114	010146			MOV	R1,-(SP)	
6462	064116	012701	064052		MOV	#40\$,R1	
6463	064122				HRDERR	NOCMI,INTERR	
6464	064142	012601			MOV	(SP)+,R1	
6465	064144	000137	064204		JMP	50\$	
6466							
6467	064150			5\$:			
6468	064150	027727	116656	065372	CMP	@DPC,#CSMF4K+2	; DID WE GET TO THE STOPN (NO MATCH)?
6469	064156	001040			BNE	51\$; BR IF NO

```

6470 064160          HRDERR  NOMST,PNTCOOR
6471 064200 000137 064520      JMP      7$          ; GO SEE IF OTHER THINGS CHECK OUT.
6472
6473 064204          50$:      ; COME HERE IF NO MATCH INTERRUPT OCCURRED.
6474 064204 020227 065370      CMP      R2,#CSMF4K  ; DID WE DO A MATCH STOP?
6475 064210 001011          BNE      501$       ; BR IF NO
6476 064212          HRDERR  CMNOMI,PNTCOOR ; YES
6477 064232 000412          BR       51$        ; GO CHECK OTHER MATCH DATA.
6478 064234          501$:     HRDERR  NOMAT,PNTCOOR ; NO MATCH AT ALL.
6479 064254 000137 064520      JMP      7$          ; GO SEE IF IT MATCHES ELSEWHERE.
6480
6481
6482 064260          51$:      ; COME HERE IF WE THINK WE GOT A MATCH
6483
6484 064260 012701 065370      MOV      #CSMF4K,R1  ; EXPECT DPC TO BE JUST AFTER MATCH.
6485 064264 012703 113774      MOV      #LVEC!ALL,R3 ; EXPECT DSR TO SHOW LVEC & ALL PIX BITS
6486 064270 017702 116536      MOV      @DPC,R2    ; GET ACTUAL DPC.
6487 064274 017704 116534      MOV      @DSR,R4    ; AND THE DSR.
6488 064300          IFERRX2 MATPCS,EXPRC2
6489 064300 020102          CMP      R1,R2      ; 1ST ITEM OK?
6490 064302 001002          BNE      65$        ; NO.
6491 064304 020304          CMP      R3,R4      ; 2ND ITEM OK?
6492 064306 001410          BEQ      64$        ; YES.
6493 064310          65$:      HRDERR  MATPCS,EXPRC2, ; NO.
6494 064330          64$:
6495
6496 064330 013701 065326      MOV      CSMF4X,R1  ; GET EXPECTED MATCH X
6497 064334 013703 065330      MOV      CSMF4Y,R3  ; ... AND THE Y.
6498 064340 017702 116472      MOV      @DXR,R2    ; GET DXR, SHOULD HOLD MATCH X
6499 064344 017704 116470      MOV      @DYR,R4    ; GET DYR, SHOULD HOLD MATCH Y.
6500 064350          IFERRX2 MATXYE,EXPRC2
6501 064350 020102          CMP      R1,R2      ; 1ST ITEM OK?
6502 064352 001002          BNE      67$        ; NO.
6503 064354 020304          CMP      R3,R4      ; 2ND ITEM OK?
6504 064356 001410          BEQ      66$        ; YES.
6505 064360          67$:      HRDERR  MATXYE,EXPRC2, ; NO.
6506 064400          66$:
6507
6508 064400 012777 100000 116430      MOV      #RSTPOS,@DXR ; RESTORE GRAPHIC POSITION TO DXR, DYR
6509 064406 004737 027454          JSR      PC,WAITF   ; WAIT FOR STOP.
6510 064412 062701 000004          ADD      #4,R1      ; X SHOULD BE X(MATCH)+4
6511 064416 017702 116414          MOV      @DXR,R2    ; GET ACTUAL DXR
6512 064422 017704 116412          MOV      @DYR,R4    ; ... AND DYR.
6513 064426          IFERRX2 MATPOS,EXPRC2
6514 064426 020102          CMP      R1,R2      ; 1ST ITEM OK?
6515 064430 001002          BNE      69$        ; NO.
6516 064432 020304          CMP      R3,R4      ; 2ND ITEM OK?
6517 064434 001410          BEQ      68$        ; YES.
6518 064436          69$:      HRDERR  MATPOS,EXPRC2, ; NO.
6519 064456          68$:
6520
6521 064456 012701 110000          6$:      MOV      #LVEC,R1  ; GET OP-CODE EXPECTED FOR FLAGS.
6522 064462          RDDSRA  FLG,R2    ; GET FLAGS.
6523 064462 012777 000002 116344      MOV      #SELFGL,@DSR ; READ FLG INTO R2.
6524 064470 017702 116340          MOV      @DSR,R2
6525 064474          IFERROR MATFLE,FLGER,7$
6526 064474 020102          CMP      R1,R2      ; OK?

```

6506	064476	001410						BEQ	7\$: YES.
	064500							HRDERR	MATFLE,FLGER,		: NO.
	064520										
	064520	112737	000377	026107	7\$:			MOV	#*0377,INTFLAG		: PRIME THE INTERRUPT FLAG.
6507	064526				7\$:			DPSTART	#CSMF4E		: START AGAIN TO SEE IF WE GET BAD MATCH
	064526	012777	065372	116276				MOV	#CSMF4E,@DPC		: START THE DPU.
	064534	004737	027454					JSR	PC,WAITF		: WAIT FOR DISPLAY STOP.
6508	064540	004737	027454					JSR	PC,WAITF		
6509	064544	105737	026107					TSTB	INTFLAG		: DID WE GET AN INTERRUPT?
6510	064550	002421						BLT	17\$: BR IF NO
6511	064552	001446						BEQ	18\$: BR IF GOT 'GOOD' INTERRUPT.
6512	064554	017702	116252					MOV	@DPC,R2		
6513	064560	010146						MOV	R1,-(SP)		
6514	064562	012701	064542					MOV	#.-16.,R1		
6515	064566							HRDERR	MATUI,INTERR		
6516	064606	012601						MOV	(SP)+,R1		
6517	064610	000137	064710					!P	8\$		
6518											
6519	064614				17\$:			MOV	#CSMF4I+2,R1		: GET EXPECT STOP DPC.
6520	064614	012701	065472					MOV	#STOPN!GOLD,R3		: DSR SHOULD HAVE STOP + GOLD COLOR.
6521	064620	012703	173200					MOV	@DPC,R2		: GET THE DPC.
6522	064624	017702	116202					MOV	@DSR,R4		: GET THE DSR
6523	064630	017704	116200					IFERRX2	MENDPCS,EXPRC2		
6524	064634							CMP	R1,R2		: 1ST ITEM OK?
	064636	020102						BNE	71\$: NO.
	064640	001002						CMP	R3,R4		: 2ND ITEM OK?
	064642	020304						BEQ	70\$: YES.
	064644	001410			71\$:			HRDERR	MENDPCS,EXPRC2,		: NO.
	064664				70\$:						
6525	064664	000137	064710					JMP	8\$		
6526											
6527	064670							HRDERR	MATUI,PNTCOOR		
6528											
6529	064710							ENDSEG	; >>>>>>>>>>>>>END SEGMENT		
6530	064712	062737	000002	065022	8\$:			ADD	#2,CSMPX		: BUMP POINTER TO X DATA.
6531	064720	023727	065022	065136				CMP	CSMPX,#CSMPTE		: IS X POINTER AT END OF TABLE?
6532	064726	103002						BHIS	9\$: BR IF YES -- GO DO NEXT Y.
6533	064730	000137	063524					JMP	3\$: LOOP BACK.
6534											
6535	064734	062737	000002	065024	9\$:			ADD	#2,CSMPY		
6536	064742	023727	065024	065136				CMP	CSMPY,#CSMPTE		: IS Y POINTER AT END OF TABLE?
6537	064750	103002						BHIS	10\$: BR IF YES -- WE'RE DONE.
6538	064752	000137	063510					JMP	2\$: NO -- GO DO NEXT Y.
6539	064756				10\$:						
6540											
6541	064756							CSMNEXT:			
6542	064756	005237	065140					INC	CSCHAN		: INCREMENT TO NEXT CHANNEL
6543	064762	023727	065140	000004				CMP	CSCHAN,#4		: DID WE DO ALL AVAILABLE CHANNELS?
6544	064770	002002						BGE	1\$: BR IF YES -- GO TO EXIT.
6545	064772	000137	060210					JMP	CSMLOOP		: NO. LOOP BACK TO TEST NEXT CHANNEL.
6546	064776										
6547	064776				1\$:			CSMEND:			
	064776	004737	027646					COMEND	CSMITR		
	065002	103401						JSR	PC,LOOP		: REPEAT 'TIL LOOPER EXPIRES.
	065004	000402						BCS	64\$		
								BR	65\$		


```

065006 000137 060204      64$: JMP CSMITR
065012 004737 030072      65$: JSR PC,TSTEND ; PRINT ERROR SUMMARY, IF REQ'D.
6548 065016 CSMXIT: EXIT TST
6549
6550
6551
6552 ; TABLE OF MATCH POINTS -- ALL VALUES ARE USED FOR X AND Y:
6553
6554 065022 000000 CSMPIX: .WORD 0
6555 065024 000000 CSMPY: .WORD 0
6556
6557 065026 000000 000002 000004 CSMPXB: .WORD 0,BIT1,BIT2,BIT3,BIT4 ; FIRST, FLOATING 1 ON 0'S, BITS 2
6558 065040 000040 000100 000200 .WORD BIT5,BIT6,BIT7,BIT8,BIT9 ; ... THRU 9.
6559 001776 A1=^01776 ; ALL 1'S FOR POSITION
6560 065052 001776 001774 001772 .WORD A1,A1&^CBIT1,A1&^CBIT2,A1&^CBIT3,A1&^CBIT4 ; THEN, FLOATING 0 ON 1'S
6561 065064 001736 001676 001576 .WORD A1&^CBIT5,A1&^CBIT6,A1&^CBIT7,A1&^CBIT8,A1&^CBIT9
6562 065076 001252 000524 001462 .WORD 1252,524,1462,636 ; ALTERNATING BITS AND PAIRS.
6563 065106 000006 000016 000036 .WORD 6,16,36,76,176,376
6564 065122 001770 001760 001740 .WORD 1770,1760,1740,1700,1600,1400
6565 065136 000000 CSMPTE: .WORD 0
6566
6567 065140 000000 CSCHAN: .WORD 0
6568 065142 000000 CSCH.6: .WORD 0
6569 065144 000000 CSCH.8: .WORD 0
6570 065146 000000 CSCH.11: .WORD 0
6571
6572 065150 000002 CSRETRY: .WORD 2
6573
6574 ; EXTENDED PRINTOUT FOR SYNC CHANNEL:
6575 065152 PRSYCH: PRINTX #FSYCHAN,CSCHAN
6576 065176 000207 RTS PC
6577
6578 ;DISPLAY FILES FOR CURSOR/SWITCH/MATCH TEST --
6579
6580 065200 175000 000000 000000 CSMF1A: .WORD LDPC,0,0 ;LOAD CURSOR POSITION REGISTERS
6581 065206 146100 CSMF1B: .WORD CURD ;READ THEM BACK
6582 065210 173000 .WORD STOPN
6583
6584 065212 146052 CSMF2A: .WORD CUIOFF!CUOFF ; CLEAR ALL ENABLES
6585 065214 146012 CSMF2B: .WORD CUIOFF ; DO GENERAL WRITE
6586 065216 146100 173000 CSMF2C: .WORD CURD,STOPN ; READ CURSOR (ENABLES COME TOO), THEN STOP
6587 065222 146077 CSMF2D: .WORD CUON!^077 ; SET ALL ENABLES
6588 065224 160000 065214 .WORD DJMP,CSMF2B ; THEN GO DO GENERAL WRITE AND READ.
6589
6590 065230 175000 001332 002464 CSMF3A: .WORD LDPC,^01332,^02464 ; LOAD CURSOR WITH SOME DATA
6591 065236 146013 CSMF3B: .WORD CUIS ; ENABLE SWITCH INTERRUPT
6592 065240 175402 CSMF3C: .WORD LDECC!SWCHON ; TURN ON THE SOFT SWITCH
6593 065242 164000 164000 164000 .WORD DNOP,DNOP,DNOP,DNOP ; DO SOME NO-OPS -- SHOULD STOP
6594 065252 173000 CSMF3D: .WORD STOPN ; STOP -- SWITCH DIDN'T WORK
6595
6596 ;DISPLAY FILE FOR CURSOR MATCH TEST:
6597
6598 065254 176074 176474 177074 CSMF4V: .WORD RDWRT!CHO,RDWRT!CH1,RDWRT!CH2,RDWRT!CH3 ; MEMORY SETUP, R/W.
6599 065264 160000 065300 .WORD DJMP,CSMF4A
6600
6601 065270 176034 176434 177034 CSMF4U: .WORD WRT!CHO,WRT!CH1,WRT!CH2,WRT!CH3 ; MEMORY SETUP, WRT-ONLY.
  
```

```

6602 065300 170140          CSMF4A: .WORD CLRMEM          ; CLEAR THE MEMORIES.
6603 065302 160001 065320    .WORD DJMS,CSMF4B          ; GO TURN ON CURSOR AND MATCH ENABLE
6604 065306 160001 065314 173000 .WORD DJMS,CSMF4C,STOPN      ; TURN SWITCH ON, AND STOP
6605 065314 175402 165000    CSMF4C: .WORD LDECC!SWCHON,DPOP ; TURN SWITCH ON, AND RETURN.
6606
6607 065320 146076 165000    CSMF4B: .WORD CUON!CUIM,DPOP   ; ENABLE CURSOR AND MATCH.
6608 065324 175000          CSMF4LC: .WORD LDCP           ; LOAD CURSOR POSITION.
6609 065326 000000          CSMF4X: .WORD 0              ; X
6610 065330 000000          CSMF4Y: .WORD 0              ; Y
6611 065332 165000          .WORD DPOP                   ; RETURN.
6612
6613 065334 114000 160001 065326 CSMF4D: .WORD APNT,DJMS,CSMF4X   ; SET MATCH POSITION
6614 065342 160001 065324          .WORD DJMS,CSMF4LC          ; LOAD CURSOR POSITION.
6615 065346 146152 173000    CSMF4J: .WORD CURD!JSWE!JMWE!JCWE,STOPN ; STOP FOR CHECK.
6616 065352 160001 065320          .WORD DJMS,CSMF4B          ; GO TURN ON MATCH ENABLE
6617 065356 160001 065314          .WORD DJMS,CSMF4C          ; GO TURN ON SWITCH
6618 065362 113774          .WORD LVEC!ALL             ; SET TO WHITE.
6619 065364 040004 000000          .WORD I!4,0               ; DRAW, SHOULD MATCH
6620 065370 173000          CSMF4K: .WORD STOPN          ; ... WITH DPC AT F4K
6621 065372 114000 160001 065326 CSMF4E: .WORD APNT,DJMS,CSMF4X   ; RESTART. SET POINT
6622 065400 113600          .WORD LVEC!YELLOW         ; CHANGE COLOR
6623 065402 000002 000000          .WORD 2,0                 ; ... THEN MOVE TO X+2, Y.
6624 065406 041776 000000 061776 .WORD I!MAXX,0,I!M!MAXX,0 ; DRAW OUT TO RIGHT AND BACK.
6625 065416 113300          .WORD LVEC!VIOLET
6626 065420 020002 000002          CSMF4F: .WORD M!2,2         ; GO TO ABOVE MATCH POINT
6627 065424 040000 001776 040000 .WORD I!0,MAXY,I!0,M!MAXY ; ... AND DRAW UP AND BACK.
6628 065434 113500          .WORD LVEC!EGGBL
6629 065436 020002 020002          CSMF4G: .WORD M!2,M!2
6630 065442 061776 000000 041776 .WORD I!M!MAXX,0,I!MAXX,0 ; DRAW LEFT AND BACK.
6631 065452 113200          .WORD LVEC!GOLD
6632 065454 000002 020002          CSMF4H: .WORD 2,M!2
6633 065460 040000 021776 040000 .WORD I!0,M!MAXY,I!0,MAXY
6634 065470 173000 173000    CSMF4I: .WORD STOPN,STOPN
6635
6636
6637 :SUBROUTINES:
6638 065474 004737 027454    :GET CURSOR COORDINATES, AND READ MASKED DXR, DYR INTO R2, R4 --
6639 065500          GETCURS: JSR PC,WAITF          ; WAIT FOR STOP BIT.
        BISW3 CSCH.8,#WRTJSS!BIT6,@DXR ; RETRIEVE CURSOR FROM SELECTED CHAN.
        MOV #WRTJSS!BIT6,-(SP)
        BIS CSCH.8,(SP)
        MOV (SP)+,@DXR
6640 065514 004737 027454          JSR PC,WAITF          ; WAIT FOR STOP.
6641 065520          RDDXY: BICW3 #^C<BIT15!^03777>,@DXR,R2 ; DXR TO R2, LEAVE FLAG & COORD.
        MOV @DXR,R2
        BIC #^C<BIT15!^03777>,R2
6642 065530          BICW3 #^C<BIT15!^03777>,@DYR,R4 ; DYR TO R4, LEAVE FLAG & COORD.
        MOV @DYR,R4
        BIC #^C<BIT15!^03777>,R4
6643 065540 000207          RTS PC
6644
6645
6646 065542 004737 027454    :GET JOYSTICK ENABLES-
6647 065546          GETJSE: JSR PC,WAITF          ; WAIT FOR STOP.
        BISW3 CSCH.8,#WRTJSS!BIT6,@DXR ; READ CURSOR
        MOV #WRTJSS!BIT6,-(SP)
        BIS CSCH.8,(SP)
        MOV (SP)+,@DXR
6648 065562 004737 027454          JSR PC,WAITF

```

```
6649 065566 RDJSE: BICW3 #^C<BIT15!JSSIES!JSMIES>, @DXR, R2 ;DXR TO R2, LEAVE ENABLES
      065566 017702 115244 MOV @DXR, R2
      065572 042702 063777 BIC #^C<BIT15!JSSIES!JSMIES>, R2
6650 065576 BICW3 #^C<BIT15!CHIE>, @DYR, R4 ;DYR TO R4.
      065576 017704 115236 MOV @DYR, R4
      065602 042704 067777 BIC #^C<BIT15!CHIE>, R4
6651 065606 000207 RTS PC
6652
6653 065610 END.TEST
```

```
:*****
: *
: * END TEST 23
: *
:*****
```


7086	070564	000000			0	
7087	070566	164000	173000		DNOP, STOPN	: BUFFER ADDRESS.
7088	070572	160000	070554		DJMP, BMOP1+2	
7089						
7090	070576	170140			BMOP2: CLRMEM	
7091	070600	114000	000000	000000	APNT, 0, 0	
7092	070606	134000			BM04!0	
7093	070610	000000			0	
7094	070612	164000	173000		DNOP, STOPN	
7095	070616	160000	070576		DJMP, BMOP2	
7096						
7097	070622				END.TEST	

```
*****  
: *  
: * END TEST 27  
: *  
*****
```



```

7259 071622
7260 071624
7261 071626
7262 071630
7263 071632
7264 071634
7265 071636
7266 071640
7267 071642
7268 071644 001774
7269 071646 177777
7270 071650 000000 000000 000000
7271 071664 000000 000000 000000
7272
7273
7274 071704 176074 176474 177074
7275 071714 170140 173000
7276
7277 071720 117774
7278 071722 176074 176474 177074
7279 071732 170100 173000
7280
7281
7282
7283
7284
7285
7286
7287
7288
7289
7290
7291
7292
7293 071736 013700 072552
7294 071742 000300
7295 071744 042737 001400 072566
7296 071752 050037 072566
7297 071756 013737 072510 072512
7298 071764 012777 072556 111040
7299 071772 004737 027454
7300 071776 013702 072606
7301 072002 062702 001000
7302 072006 013703 072576
7303 072012 027702 111014
7304 072016 001003
7305 072020 027703 111010
7306 072024 001410
7307 072026
7308 072026
7309 072046 005004
7310 072050 013705 072574
7311 072054 062705 000002
7312 072060 027704 110752
7313 072064 001003
7314 072066 027705 110746
7315 072072 001410
  
```

```

CLMSHT: CLP.D 2
         CLP.D 1
         CLP.D 14
         CLP.D 11
         CLP.D 12
         CLP.D 3
         CLP.D 5
         CLP.D 6
         CLP.D 7
         1774
         -1
         0,0,0,0,0,0
         0,0,0,0,0,0,0,0

;DISPLAY FILES FOR CLEARING MEMORY:
CLR.Z2: RDWRT!CH0,RDWRT!CH1,RDWRT!CH2,RDWRT!CH3
        CLRMEM,STOPN

CLR.D2: APNT!ALL
        RDWRT!CH0,RDWRT!CH1,RDWRT!CH2,RDWRT!CH3
        SETMEM,STOPN

: SUBROUTINE - READ & VERIFY 1 LINE OF IMAGE MEMORY.
: ENTER WITH:
: IMC.CH = MEMORY CHANNEL TO READ (0, 1, 2, OR 3)
: IMC.Y = Y POSITION OF LINE TO BE READ
: IMC.RA = ADDRESS OF 512.-BYTE BUFFER TO READ DATA INTO
: IMC.TA = ADDRESS OF 512.-BYTE BUFFER TO TEST AGAINST
: (DATA MUST INCLUDE 1'S IN NON-EXISTENT BIT POSITIONS TO
: CATCH BAD DRIVERS & DBUS LINES)

IMCHK:  MOV   IMC.CH,R0           ;GET CHANNEL #
        SWAB  R0                 ;... INTO HIGH BYTE OF R0.
        BIC  #BIT9!BIT8,IMC.ME  ;CLEAR OLD CHANNEL # FROM ENABLING INSTR.
        BIS  R0,IMC.ME          ;SET IN THE NEW CHANNEL

IMCHKR: MOV   #IMC.SR,@DPC       ;START THE READ.
        JSR  PC,WAITF           ;... WAIT FOR IT TO FINISH.
        MOV  IMC.RA,R2          ;R2 GETS FINAL ADDRESS
        ADD  #512.,R2           ;R3 GETS WHAT DSR SHOULD BE.
        MOV  IMC.IR,R3         ;IS DPC CORRECT?
        CMP  @DPC,R2           ;IS DSR CORRECT?
        BNE  1$
        CMP  @DSR,R3
        BEQ  2$

1$:
2$:    HRDERR IMRF,PCSERR        ;PRINT IF NOT.
        CLR  R4                 ;DXR SHOULD BE 0.
        MOV  IMC.Y,R5
        ADD  #2,R5              ;DYR SHOULD BE Y+2
        CMP  @DXR,R4           ;TEST DXR
        BNE  3$
        CMP  @DYR,R5           ;TEST DYR
        BEQ  4$
  
```

```

7316 072074      3$:
7317 072074
7318 072114      012701 000000
7319 072120      013704 072554
7320 072124      013705 072606
7321 072130      022425
7322 072132      001006
7323 072134      062701 000004
7324 072140      020127 002000
7325 072144      002771
7326 072146      000471
7327
7328 072150      005337 072512      10$:
7329 072154      100402
7330 072156      000137 071764
7331 072162      010137 072504
7332 072166      126465 177776 177776
7333 072174      001440
7334 072176      116437 177776 023056
7335 072204      116537 177776 023060
7336 072212      006337 023056
7337 072216      006337 023056
7338 072222      006337 023060
7339 072226      006337 023060
7340 072232      042737 176003 023056
7341 072240      042737 176003 023060
7342 072246
7343 072266      032737 000002 072504
7344 072274      001317
7345 072276      062737 000002 072504
7346 072304      126465 177777 177777
7347 072312      001710
7348 072314      116437 177777 023056
7349 072322      116537 177777 023060
7350 072330      000730
7351
7352 072332      000207      9$:
7353
7354 072334      BGNMSG
7355 072334
7356 072354      013737 072504 072506
7357 072362      006237 072506
7358 072366
7359 072432
7360 072462
7361 072502      ENDMSG
7362
7363 072504      000000      IMCBDA: .WORD 0
7364 072506      000000      IMCRAM: .WORD 0
7365 072510      000000      IMCTRY: .WORD 0
7366 072512      000000      IMCRCT: .WORD 0
7367
7368 072514      040      040      104      IMRF: .ASCIZ / DMA PIXEL READBACK FAILURE/
7369
7370
7371
7372 072552      163000      IMR8=163000
000000      IMC.CH: 0
  
```

```

:R1 = X ADDRESS UNDER TEST.
:R4 POINTS TO EXPECTED DATA
:R5 POINTS TO RECEIVED DATA
:CHECK THE DATA

:BUMP THE X ADDRESS

; EXIT WHEN DONE WITH LINE.

:GET POSITION OF BAD DATA
:LOW BYTE BAD?
:BR IF NOT.
:IF YES, GET THE DATA

:SHIFT IT TO LOOK LIKE DBUS DATA

:DONE HIGH BYTE?
;IF YES, GO BACK TO MAIN LOOP.

:COMPARE HIGH BYTES
  
```

*** TEST 28 IMAGE MEMORY CLEAR-SET

7373	072554	000000			IMC.TA: 0	: ADDRESS OF TEST BUFFER
7374	072556	176000	176400	177000	IMC.SR: PROTEC!CH0,PROTEC!CH1,PROTEC!CH2,PROTEC!CH3	
7375	072566	176074			IMC.ME: RDWRT	: ENABLE SELECTED MEMORY
7376	072570	114000	000000		APNT,0	: START AT X=0
7377	072574	000000			IMC.Y: 0	: Y
7378	072576	163006			IMC.IR: IMR8!6	: READ, 2 8-BIT PIXELS/WORD, ALL BITS
7379	072600	001000	000001		512.,1	: 1 FULL LINE
7380	072604	000400			256.	: WORD COUNT = 256.
7381	072606	160000			IMC.RA: .WORD 160000	: READ BUFFER
7382	072610	173000			STOPN	: (SHOULD STOP BY ITSELF)
7383	072612				END.TEST	

```

:*****
:
:      END TEST 28
:
:*****

```



```

7426 073076             ENDSEG             ; <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<< END SEGMENT.
7427 073100             COMEND 1$
      073100 004737 027646  JSR PC, LOOP           ; REPEAT 'TIL LOOPER EXPIRES.
      073104 103401          BCS 65$
      073106 000402          BR 66$
      073110 000137 072722 65$: JMP 1$
      073114 004737 030072 66$: JSR PC, TSTEND          ; PRINT ERROR SUMMARY, IF REQ'D.
7428 073120             EXIT TST             ; EXIT THIS TEST >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
7429
7430
7431                   ; DISPLAY FILE.
7432
7433 073124 176000 176400 177000 10$: PROTEC!CHO,PROTEC!CH1,PROTEC!CH2,PROTEC!CH3
7434 073134 176034          11$: WRT!CHO
7435 073136 117774 040000 000000  APNT!ALL,I!O,O
7436 073144 173000          STOPN
7437
7438 073146             END.TEST
  
```

```

:*****
: *
: *   END TEST 29
: *
:*****
  
```


:*
:*****


```

7700
7701      ; NOW 1 LARGE AND 2 SMALL OCTAGNS.
7702      ;
7703 074714 012777 074754 106110 VIS2:  MOV #LOCT,@DPC ;1 LARGE ONE...
7704 074722 004737 027454      JSR PC,WAITF
7705 074726 012777 075026 106076      MOV #SOCT1,@DPC ;...AND 2 SMALL ONES.
7706 074734 004737 027454      JSR PC,WAITF
7707 074740 012777 075040 106064      MOV #SOCT2,@DPC
7708 074746 004737 027454      JSR PC,WAITF
7709      ; JSR PC,PAUS.5
7710 074752 000456      BR VIS3 ;...AND CONTINUE.
7711
7712 074754 114000 000500 000002 LOCT: APNT,500,2
7713 074762 113774      LVEC!ALL ; 1 LARGE OCTAGON...
7714 074764      LXI 1,576,0
7715 074770      LXI 1,436,436
7716 074774      LXI 1,0,576
7717 075000      LXI 1,-436,436
7718 075004      LXI 1,-576,0
7719 075010      LXI 1,-436,-436
7720 075014      LXI 1,0,-576
7721 075020      LXI 1,436,-436
7722 075024 173000      STOPN
7723
7724 075026 114000 000220 000620 SOCT1: APNT,220,620 ;...AND 2 SMALL ONES.
7725 075034 160000 075046      DJMP,+.10
7726 075040 114000 001460 000620 SOCT2: APNT,1460,620
7727 075046 107774      SVEC!ALL
7728 075050      SXY 1,76,0
7729 075052      SXY 1,56,56
7730 075054      SXY 1,0,76
7731 075056      SXY 1,-56,56
7732 075060      SXY 1,-76,0
7733 075062      SXY 1,-56,-56
7734 075064      SXY 1,0,-76
7735 075066      SXY 1,56,-56
7736 075070 130000      RPNT ;5 REL POINTS IN CENTER.
7737 075072      SXY U,20,50
7738 075074      SXY 1,20,50 ;CENTER.
7739 075076      SXY 1,20,0
7740 075100      SXY 1,-20,20
7741 075102      SXY 1,-20,-20
7742 075104      SXY 1,20,-20
7743 075106 173000      STOPN
  
```

```

7745
7746      : NEXT, SHOW GRAPH/HISTO Y IN UPPER LEFT.
7747      :
7748 075110 012737 011240 075262 VIS3:  MOV      #1240!GHIINH,GY+2 ;PLOT GRAPH Y STARTING AT 1240!GHIINH.
7749 075116 012777 075236 105706      MOV      #GRY,@DPC      ;START GRAPH Y.
7750 075124 000402
7751 075126 005277 105700      1$:   INC      @DPC
7752 075132 004737 027454      JSR      PC,WAITF
7753 075136 042737 010000 075262      BIC      #GHIINH,GY+2
7754 075144 062737 000004 075262      ADD      #4,GY+2      ;INCR Y AMPLITUDE...
7755 075152 023727 075262 001604      CMP      GY+2,#1604
7756 075160 001362      BNE      1$          ;LOOP UNTIL DONE.
7757
7758 075162 012737 051240 075310      MOV      #HST!1240!GHIINH,HY+2 ;NOW, HISTO Y STARTING AT 1240!GHIINH.
7759 075170 012777 075272 105634      MOV      #HSY,@DPC
7760 075176 000402
7761 075200 005277 105626      2$:   INC      @DPC
7762 075204 004737 027454      JSR      PC,WAITF
7763 075210 042737 010000 075310      BIC      #GHIINH,HY+2
7764 075216 062737 000010 075310      ADD      #10,HY+2     ;INCR Y AMPLITUDE...
7765 075224 023727 075310 041550      CMP      HY+2,#HST!1550
7766 075232 001362      BNE      2$          ;...AND LOOP TIL DONE.
7767
7768      :
7769 075234 000431      JSR      PC,PAUS.5
7770      BR      VIS4          ;...AND CONTINUE.
7771 075236 114000 000100 001240 GRY:  APNT,100,1240
7772 075244 113774      LVEC:ALL      ;DRAW BASE LINE.
7773 075246      LXY      I,340,0
7774 075252      LXY      U,-340,0
7775 075256 174104      GXI!4          ;X INCR = 4 FOR GRAPH.
7776 075260 127774 001240 GY:   GHY!ALL,0!1240
7777 075264 173000 160000 075260      STOPN,DJMP,GY
7778
7779 075272 150000 001240 HSY:  SETHB,1240      ;HISTO BASE LINE.
7780 075276 114000 000140 001240      APNT,140,1240
7781 075304 174110      GXI!10
7782 075306 127774 041240 HY:   GHY!ALL,HST!1240
7783 075312 173000 160000 075306      STOPN,DJMP,HY
  
```

```

7785
7786      ; NEXT, GRAPH/HISTO X IN LOWER RIGHT.
7787      ;
7788 075320 012737 011340 075472 VIS4:  MOV      #1340!GHIINH,GX+2 ;PLOT GRAPH X STARTING AT 1340!GHIINH.
7789 075326 012777 075446 105476      MOV      #GRX,@DPC      ;START GRAPH X.
7790 075334 000402
7791 075336 005277 105470      1$:   INC      @DPC
7792 075342 004737 027454      JSR      PC,WAITF
7793 075346 042737 010000 075472      BIC      #GHIINH,GX+2
7794 075354 062737 000004 075472      ADD      #4,GX+2      ;INCR X AMPLITUDE...
7795 075362 023727 075472 001704      CMP      GX+2,#1704
7796 075370 001362      BNE      1$          ;LOOP UNTIL DONE.
7797
7798 075372 012737 051340 075520      MOV      #HST!1340!GHIINH,HX+2 ;NOW, HISTO X STARTING AT 1340!GHIINH.
7799 075400 012777 075502 105424      MOV      #HSX,@DPC
7800 075406 000402
7801 075410 005277 105416      2$:   INC      @DPC
7802 075414 004737 027454      JSR      PC,WAITF
7803 075420 042737 010000 075520      BIC      #GHIINH,HX+2
7804 075426 062737 000010 075520      ADD      #10,HX+2     ;INCR X AMPLITUDE...
7805 075434 023727 075520 041650      CMP      HX+2,#HST!1650
7806 075442 001362      BNE      2$          ;...AND LOOP 1.L DONE.
7807
7808      ; JSR      PC,PAUS.5
7809 075444 000431      BR       VIS5        ;...AND CONTINUE.
7810
7811 075446 114000 001340 000100 GRX:   APNT,1340,100
7812 075454 113774      LVEC:ALL      ;DRAW BASE LINE.
7813 075456      LXY      1,0,340
7814 075462      LXY      0,0,-340
7815 075466 174104      GYI!4        ;Y INCR = 4 FOR GRAPH.
7816 075470 123774 001340      GX:   GHX!ALL,0!1340
7817 075474 173000 160000 075470      STOPN,DJMP,GX
7818
7819 075502 150000 001340      HSX:   SETHB,1340      ;HISTO BASE LINE.
7820 075506 114000 001340 000140      APNT,1340,140
7821 075514 174110      GYI!10     ;Y INCR = 10 FOR HISTO.
7822 075516 123774 041340      HX:   GHX!ALL,HST!1340
7823 075522 173000 160000 075516      STOPN,DJMP,HX
  
```

```

7825
7826      : NEXT SHOW BIT MAP MODE 1 AT BOTTOM CENTER,
7827      : AND BIT MAP MODE 0 AT TOP CENTER.
7828
7829 075530 012700 177777      VIS5:      MOV      #-1,R0      ; SET PIXEL DATA PATTERN...
7830 075534 004737 027020      JSR      PC,FILL2      ;...AND FILL BUFFER WITH...
7831                                     ;...ALTERNATE WORDS.
7832
7833 075540 013737 003240 075710      MOV      FREE,B148      ; SET BUFFER ADDRESS.
7834 075546 012737 136340 075706      MOV      #BM14!340,B148-2 ; SET OPCODE = BM14 X 340
7835 075554 012777 075666 105250      MOV      #B14,@DPC      ;XCT BM14, DRAWS A DASHED...
7836 075562 004737 027454      JSR      PC,WAITF      ;...LINE (4 ON, 4 OFF)...
7837                                     ;...FOR 340 PIXELS.
7838 075566 012737 137400 075706      MOV      #BM18!400,B148-2 ;SET OPCODE = BM18 X 400
7839 075574 012777 075700 105230      MOV      #B18,@DPC      ;XCT BM18, DRAWS A DASHED...
7840 075602 004737 027454      JSR      PC,WAITF      ;...LINE (2 ON, 2 OFF)...
7841                                     ;...FOR 400 PIXELS.
7842
7843 075606 012700 000377      MOV      #377,R0
7844 075612 004737 027014      JSR      PC,FILL1
7845
7846 075616 013737 003240 075736      MOV      FREE,B048      ; SET BUFFER ADDRESS.
7847      :      MOV      #BM04!M64,B048-2 ; 4 BIT X 64 SQUARE.
7848 075624 012737 134010 075734      :      MOV      #BM04!M32!EX2,B048-2 ; 4 BIT X 64 SQUARE.
7849 075632 012777 075714 105172      :      MOV      #B04,@DPC      ; DO IT.
7850 075640 004737 027454      JSR      PC,WAITF
7851      :      MOV      #BM08!M64,B048-2 ; 8 BIT X 64 SQUARE.
7852 075644 012737 135010 075734      :      MOV      #BM08!M32!EX2,B048-2 ; 8 BIT X 64 SQUARE.
7853 075652 012777 075726 105152      :      MOV      #B08,@DPC      ; AND DO THAT TOO.
7854 075660 004737 027454      JSR      PC,WAITF
7855
7856 075664      1$:      ;JSR      PC,PAUS.5
7857 075664 000430      BR      VIS5A      ;...AND CONTINUE.
7858
7859 075666 114000 000400 000240 B14:      APNT,400,240      ; 4 ON 4 OFF FOR 340 PIXELS.
7860 075674 160000 075706      DJMP, +10
7861 075700 114000 000300 000340 B18:      APNT,300,340      ; 2 ON 2 OFF FOR 400 PIXELS.
7862 075706 136340      BM14!340
7863 075710 000000      B148:      0
7864 075712 173000      STOPN
7865
7866 075714 114000 000540 001340 B04:      APNT, 540, 1340
7867 075722 160000 075734      DJMP, +10
7868 075726 114000 001040 001340 B08:      APNT, 1040, 1340
7869 075734 134001      BM04!M64
7870 075736 000000      B048:      0
7871 075740 164000 164000      DNOP, DNOP
7872 075744 173000      STOPN
  
```



```

7874
7875
7876
7877
7878
7879
7880 075746 112737 000004 076027
7881 075754 012777 076016 105050
7882 075762 000403
7883 075764 012777 000001 105040
7884 075772 004737 027454
7885 075776 123727 076027 000024
7886 076004 103003
7887 076006 105237 076027
7888 076012 000764
7889 076014
7890 076014 000430
7891
7892
7893 076016 114000 000300 000320
7894 076024 144040
7895 076026 000 004
7896 076030 006020 006040 006060
7897 076036 006100 006120 006140
7898 076046 006200 006220 006240
7899 076056 006300 006320 006340
7900 076066 173000
7901 076070 000000
7902 076072 160000 076026

```

```

: NEXT SHOW RUN-LENGTH MODE DIAGONAL BARS BETWEEN THE BM1 AND BMO LINES.
: USES 16 INTENSITY LEVELS, FROM 0 TO MAX. THE 0 LEVEL RUN-LENGTH IS VARIED
: FROM 4 TO 24(8) FOR 20 LINES OF DATA, WHILE THE OTHER RUN LENGTHS REMAIN
: CONSTANT AT 14. THE LINES ARE DRAWN FROM TOP TO BOTTOM (YDOWN=1).

```

```

VIS5A:      MOV      #4,VRLF2+1      ;SET INITIAL COUNT FOR 0-INTENS AT 4
            MOV      #VRLF,@DPC      ;START THE FILE
            BR       2$              ;SKIP OVER THE RESUME
1$:         MOV      #1,@DPC          ;RESUME THE DISPLAY
2$:         JSR      PC,WAITF         ;WAIT FOR DONE
            CMPB    VRLF2+1,#24      ;DONE 20(8) LINES?
            BHIS    3$              ;FINISH IF YES.
            INCB    VRLF2+1          ;BUMP INITIAL COUNT IF NO.
            BR      1$              ;...THEN XCT AGAIN.
3$:         JSR      PC,PAUS.5        ;PAUSE
            BR      VIS6            ;...THEN CONTINUE.

```

```

:DISPLAY FILE FOR VISUAL RUN-LENGTH MODE
VRLF:      APNT,300,320              ;START BETWEEN BM1 & BMO
VRLF1:     RNLN!YDOWN              ;ENTER RUN-LENGTH MODE, GO DOWN THE SCREEN
VRLF2:     .BYTE 0,4                ;0 INTENSITY, START AT LENGTH=4
            .WORD 6020,6040,6060    ;INCREASING INTENSITY, LENGTH=14
            .WORD 6100,6120,6140,6160 ;....AND MORE
            .WORD 6200,6220,6240,6260 ;....AND MORE
            .WORD 6300,6320,6340,6360 ;....AND MORE
            STOPN                    ;STOP
            .WORD 0                  ;COUNT = 0 FOR NEWLINE
            DJMP,VRLF2              ;THEN JUMP BACK FOR MORE

```

```

7904
7905
7906
7907
7908
7909
7910
7911
7912
7913
7914
7915
7916
7917
7918
7919 076076 012737 112000 076536 VIS6: MOV #LVEC!LO,RBL ;INIT DISPLAY FILE.
7920 076104 012737 040000 076540 MOV #I!O,RBX
7921 076112 012737 000300 076542 MOV #300,RBY
7922 076120 012701 000003 MOV #3,R1 ; SET FOR 3 VECTORS PER LEVEL...
7923 076124 012703 000004 MOV #4,R3 ;...AND INCR LEVEL BY 'L1'.
7924 076130 010102 1$: MOV R1,R2 ; SCRATCH COPY LEVEL COUNTER.
7925 076132 012777 076530 104672 MOV #RBO,@DPC ;START, QUAD 1.
7926 076140 004737 027454 JSR PC,WAITF
7927 076144 023727 076542 000000 2$: CMP RBY,#0 ;DELTA Y MIN ??
7928 076152 001403 BEQ 3$ ;DONE QUAD 1 IF SO.
7929 076154 004737 076366 JSR PC,Q1
7930 076160 000771 BR 2$
7931 076162 052737 020000 076542 3$: BIS #MXY,RBY ;QUAD 2, DX POS, DY NEG.
7932 076170 023727 076540 040000 4$: CMP RBX,#I!O ;DELTA X MIN ??
7933 076176 001403 BEQ 5$ ;DONE QUAD 2 IF SO.
7934 076200 004737 076424 JSR PC,Q2
7935 076204 000771 BR 4$
7936 076206 052737 020000 076540 5$: BIS #MXY,RBX ;QUAD 3, DX NEG, DY NEG.
7937 076214 023727 076542 020000 6$: CMP RBY,#MXY!O ;DELTA Y -MIN ??
7938 076222 001403 BEQ 7$ ;DONE QUAD 3 IF SO.
7939 076224 004737 076366 JSR PC,Q3
7940 076230 000771 BR 6$
7941 076232 042737 020000 076542 7$: BIC #MXY,RBY ;QUAD 4, DX NEG, DY POS.
7942 076240 023727 076540 060000 8$: CMP RBX,#I!MXY!O ;DELTA X -MIN ??
7943 076246 001403 BEQ 9$ ;DONE QUAD 4 IF SO.
7944 076250 004737 076424 JSR PC,Q4
7945 076254 000771 BR 8$
7946
7947 076256 012737 004000 003220 9$: MOV #4000,SHADLY ; FINALLY, SET SHADING DELAY...
7948 076264 012705 110224 MOV #NTSC8+2,R5 ;...AND CYCLE THRU 16 SHADES...
7949 076270 012501 10$: MOV (R5)+,R1 ;...OF EACH BASIC (NTSC) COLOR.
7950 076272 001403 BEQ 11$ ; BR WHEN ALL DONE.
7951 076274 004737 027172 JSR PC,SHAD16
7952 076300 000773 BR 10$
7953
7954 076302 004737 104762 11$: JSR PC,OUTLIN ; OUTLINE THE SCREEN.
7955 076306 004737 027514 JSR PC,PAUSE1
7956 076312 004737 027514 JSR PC,PAUSE1
7957 076316 004737 027514 JSR PC,PAUSE1 ; PAUSE 3 SECONDS...
7958 076322 004737 105330 JSR PC,EBLINK ; ENABLE BLINKING.
7959 076326 004737 027514 JSR PC,PAUSE1
7960 076332 004737 027514 JSR PC,PAUSE1 ; PAUSE 3 SECONS...

```

```

: FINALLY, AT DEAD-CENTER:
: A CIRCULAR SWEEP FROM 12 O'CLOCK THRU 360 DEGREES.
: INCREMENT INTENSITY/COLOR LEVEL SUCH THAT ALL LEVELS
: ARE DISPLAYED IN 360 DEGREES.
: 3 VECTORS PER LEVEL * 256 LEVELS = 768 VECTORS TOTAL.
: (OCTAL 3 * 400 = 1400)
: ACTUAL COLOR/INTENSITY SPECTRUM DEPENDS UPON HOW MANY
: PIXEL BITS ARE AVAILABLE, AND HOW THE LOOK-UP-RAM IS
: PROGRAMMED (GREY-SCALE BY DEFAULT, IF INSTALLED).
: 2 BITS <9:8> = 4 LEVELS.
: 4 BITS <9:6> = 16 LEVELS (4 PER QUADRANT).
: 6 BITS <9:4> = 64 LEVELS (LUT REQ'D).
: 8 BITS <9:2> = 256 LEVELS (LUT REQ'D).

```



```
7968 076366 013705 076540      Q1:Q3:  MOV    RBX,R5      ;QUAD 1 AND 3, INCR DX 'TIL...
7969 076372 042705 176000      BIC    #^C1777,R5 ;...MAX, THEN DECR DY.
7970 076376 020527 000300      CMP    R5,#300    ;DELTA X MAX ??
7971 076402 001404                BEQ    1$          ;
7972 076404 062737 000002 076540  ADD    #2,RBX      ;NO, INCR DELTA X.
7973 076412 000403                SKP3
7974 076414 162737 000002 076542 1$:    SUB    #2,RBY      ;YES, DECR DELTA Y.
7975 076422 000416                BR     QXIT        ;CHECK FOR LEVEL CHANGE AND EXIT
7976
7977 076424 013705 076542      Q2:Q4:  MOV    RBY,R5      ;QUAD 2 AND 4, INCR DY 'TIL...
7978 076430 042705 176000      BIC    #^C1777,R5 ;...MAX, THEN DECR DX.
7979 076434 020527 000300      CMP    R5,#300    ;DELTA Y MAX ??
7980 076440 001404                BEQ    1$          ;
7981 076442 062737 000002 076542  ADD    #2,RBY      ;NO, INCR DELTA Y.
7982 076450 000403                SKP3
7983 076452 162737 000002 076540 1$:    SUB    #2,RBX      ;YES, DECR DELTA X.
7984
7985 076460 012700 000600      QXIT:   MOV    #600,R0 ; DELAY APPROX 3 MSEC...
7986 076464 005300      1$:    DEC    R0          ;...BETWEEN VECTORS.
7987 076466 001376                BNE    1$          ;
7988 076470 005302                DEC    R2          ; LEVEL CHANGE REQ'D ??
7989 076472 001011                BNE    2$          ;NOT YET.
7990 076474 010102                MOV    R1,R2       ;YES, RESET COUNTER...
7991 076476 060337 076536      ADD    R3,RBL      ;...INCREMENT...
7992 076502 042737 176000 076536  BIC    #^C1777,RBL ;...AND STRIP LEVEL...
7993 076510 052737 112000 076536  BIS    #LVEC!LO,RBL ;...REPLACE OPCODE.
7994 076516 005277 104310      2$:    INC    @DPC       ;RESUME...
7995 076522 004737 027454      JSR    PC,WAITF    ;...WAIT...
7996 076526 000207                RTS    PC          ;...AND EXIT.
7997
7998      ; DISPLAY FILE FOR CIRCULAR RAINBOW.
7999
8000 076530 114000 000776 000736  RBO:   APNT,HAFX,HAFY60 ;SET ORIGIN AT CENTER.
8001 076536 112000                RBL:   LVEC!LO      ;LEVEL WILL INCREMENT.
8002 076540 040002                RBX:   I!2         ;DITTO X...
8003 076542 000300                RBY:   300         ;...AND Y.
8004 076544 164000 173000 164000  DNOP,STOPN,DNOP
8005 076552 160000 076530      DJMP,RBO
8006
8007 076556                END.TEST
```

```
*****
;*
;*      END TEST 32
;*
*****
```



```

8058 077146 000403          SKP3
8059 077150 164000          1$: DNOP ;TRY TO READ CURSOR POSITION.
8060 077152 146100 173000 10$: CURD,STOPN
8061 077156 004737 027454 JSR PC,WAITF
8062 077162 103416 BCS 3$ ;BR IF DPU RESPONDS TO STICK.
8063 :*****
8064 : CLRB STIK(R1) ;OTHERWISE, SET FLAG = NO STICK.
8065 : 240, 240
8066 :*****
8067 077164 004737 026756 JSR PC,RELEAS ;RELEASE 'HUNG' DPU...
8068 077170 006200 2$: ASR R0
8069 077172 PRINTF #NOSTIK,R0 ;...TELL THE MAN...
8070 077214 000137 076744 JMP JSVA ; KEEP TRYING (FOREVER) !!
8071
8072 077220 3$: ; MOVB #1,STIK(R1) ;SET FLAG = STICK AVAILABLE...
8073 077220 000431 BR JSTST ;...AND START 'EM UP.
8074
8075 077222 045 101 040 NOSTIK: .ASCIZ /%A -- NON-EXISTENT SYNC CHANNEL : %01/
8076 .EVEN
8077
8078 ;
8079 ; CHANNEL SELECTION QUESTION AND ANSWER.
8080 ;
8081 077270 103 110 101 MCHAN: .ASCIZ /CHANNEL /
8082 .EVEN
8083 077302 000000 SELCHA: 0 ; USER SELECTED CHANNEL (DEFAULT=0).
  
```

```

8085
8086
8087
8088
8089
8090
8091 077304 004737 027054
8092 077310 004737 026650
8093 077314 012700 177777
8094 077320 004737 027014
8095 077324 013737 003240 101062
8096 077332 012777 077406 103512
8097 077340 012777 077442 103506
8098 077346 012777 077512 103504
8099 077354 005037 003170
8100 077360 005037 003172
8101 077364
      077364 012777 100306 103440
      077372 004737 027454
8102 077376 012777 100316 103426 1$:
8103 077404 000777
8104
8105 077406
      2$:
8106 077410 004737 077752
8107 077414 023737 100004 003170
8108 077422 001103
8109 077424 023737 100006 003172
8110 077432 001077
8111 077434 005277 103372
8112 077440 000002
8113
8114 077442 052737 001777 100470 4$:
8115 077450 017737 103362 003170
8116 077456 100404
8117 077460 017737 103354 003172
8118 077466 100010
8119 077470
      41$:
8120 077510 000433
      42$:
8121 077512 042737 001777 100470 5$:
8122 077520 004737 023524 51$:
8123 077524 017737 103306 003170
8124 077532 100004
8125 077534 017737 103300 003172
8126 077542 100410
8127 077544
      52$:
8128 077564 042737 174000 003170 53$:
8129 077572 042737 174000 003172
8130 077600
      54$:
8131 077600 013737 100442 100360
8132 077606 013737 100444 100362
8133 077614 013737 003170 100442
8134 077622 013737 003172 100444
8135 077630 000411
8136
8137 077632 042737 001777 100470 6$:
8138 077640 013737 100004 003170
8139 077646 013737 100006 003172
  
```

```

:
: TURN ON THE CURSOR AND DISPLAY IT'S CO-ORDINATES ON SCREEN.
: ON 'STOP' INTERRUPT, UPDATE READ-OUT IF STICK IS MOVING.
: ON 'SWITCH' INTERRUPT, MARK THE CO-ORD WITH A LITTLE 'X'.
: ON 'MATCH' INTERRUPT, MARK THE SPOT AS BEFORE, AND SAY 'MATCH'
:
JSTST: JSR      PC,LUMIN      ; BLAST THE LUT.
        JSR      PC,CLRRW    ; INIT IN READ/WRITE MODE.
        MOV      #-1,R0
        JSR      PC,FILL1    ; FILL PIXEL BUFFER.
        MOV      FREE,BMXX   ; SET BUFFER POINTER.
        MOV      #2$,@STPV   ; SET STOP...
        MOV      #4$,@JSMV   ; ...MATCH...
        MOV      #5$,@JSSV   ; ...AND SWITCH VECTORS.
        CLR      SDXR        ; CLEAR OLD JSX,JSY CO-ORDS.
        CLR      SDYR
        DPSTART #JSPIXA
        MOV      #JSPIXA,@DPC ; START THE DPU.
        JSR      PC,WAITF    ; WAIT FOR DISPLAY STOP.
        MOV      #JSPIX,@DPC ; START THE DISPLAY...
        BR      .           ; ...AND WAIT FOR INTERRUPT.

2$:    BREAK
        JSR      PC,GTCURS   ; ON 'STOP', BREAK...
        CMP      MKDXR,SDXR  ; GET & MASK THE CURSOR DATA
        BNE      6$         ; ...AND CHECK STICK MOTION.
        CMP      MKDYR,SDYR  ; BR IF STICK IS MOVING.
        BNE      6$
        INC      @DPC
        RTI
        ; DITTO
        ; OTHERWISE, JUST RESUME...
        ; ...AND DISMISS.

4$:    BIS      #1777,MATCH  ; 'MATCH' -- ENABLE TEXT.
        MOV      @DXR,SDXR   ; SEE THAT NO SIGN BIT SET ON MATCH
        BMI      41$        ; ...IF SET, REPORT ERROR
        MOV      @DYR,SDYR   ; AND ALSO FOR DYR
        BPL      42$        ; SKIP IF OK
        HRDERR  DXYPE,JXYPE  ; REPORT THE ERROR
        BR      54$        ; GC PROCESS THE INTERRUPT
        BIC      #1777,MATCH ; 'SWITCH' -- DISABLE TEXT.
        JSR      PC,ERRCHK   ; CHECK FOR ERROR CODE IN CSR
        MOV      @DXR,SDXR   ; GET X-COORDINATE & TEST SIGN
        BPL      52$        ; ...ERROR IF NOT SET.
        MOV      @DYR,SDYR   ; GET Y-COORDINATE & TEST SIGN
        BMI      53$        ; ...OK IF SET.
        HRDERR  JSDXE,JXE    ; REPORT THE ERROR.
        BIC      #174000,SDXR ; CLEAR SIGN & STATUS ON SAVED X
        BIC      #174000,SDYR ; CLEAR SIGN & STATUS ON SAVED Y

54$:   MOV      MARK,EMK     ; ON EITHER...
        MOV      MARK+2,EMK+2 ; ...ERASE OLD MARKER...
        MOV      SDXR,MARK
        MOV      SDYR,MARK+2 ; ...AND SET A NEW ONE.
        BR      62$        ; GOTO COMMON HANDLER.

6$:    BIC      #1777,MATCH  ; STICK MOVING, DISABLE TEXT.
        MOV      MKDXR,SDXR
        MOV      MKDYR,SDYR ; SAVE OBSERVED CO-ORDS.
  
```

```

8140 077654
8141 077654 013737 100414 100336
8142 077662 013737 100416 100340
8143 077670 013737 100434 100352
8144 077676 013737 100436 100354
8145 077704 013701 003170
8146 077710 004737 027324
8147 077714 010137 100414
8148 077720 010237 100416
8149 077724 013701 003172
8150 077730 004737 027324
8151 077734 010137 100434
8152 077740 010237 100436
8153 077744 012716 077376
8154 077750 000002
8155 077752 017737 103060 100004
8156 077760 017737 103054 100006
8157 077766 042737 174000 100004
8158 077774 042737 174000 100006
8159 100002 000207
8160 100004 000000
8161 100006 000000
8162
8163
8164 100010 040 040 127
8165 100064 040 040 104
8166 100140 040 040 104
8167 100216 045 101 040
8168
8169 100254
8170 100254
8171 100304
8172
  
```

```

62$:
MOV JSX,EX
MOV JSX+2,EX+2
MOV JSY,EY
MOV JSY+2,EY+2 ;ERASE OLD X/Y.
MOV SDXR,R1
MOV PC,CNVRT ;CONVERT X TO ASCII...
JSR R1,JSX ;...UPDATE READ-OUT.
MOV R2,JSX+2
MOV SDYR,R1 ;CONVERT Y TO ASCII...
JSR PC,CNVRT ;...UPDATE READ-OUT.
MOV R1,JSY ; SET PC TO RESTART...
MOV R2,JSY+2 ;...AND DISMISS.
MOV #1$, (SP)
RTI
GTCURS: MOV @DXR,MKDXR
MOV @DYR,MKDYR
BIC #174000,MKDXR
BIC #174000,MKDYR
RTS PC
MKDXR: 0
MKDYR: 0

;ERROR MESSAGES FOR JOYSTICK OPERATIONS:
JSWTEX: .ASCIZ / WAITED TOO LONG FOR J.S. STATUS RETRIEVAL/
JSDXYE: .ASCIZ / DXR OR DYR SIGN NOT SET ON J.S. RETRIEVAL/
DXYME: .ASCIZ / DXR OR DYR SIGN SET ON J.S. MATCH INTERRUPT/
JXYPX: .ASCIZ /%A DXR: %06%A DYR: %06%A%N/
.EVEN
BGNMSG JXYPE
PRINTX #JXYPX,@DXR,@DYR
ENDMSG
  
```


8174					:					
8175					:	DISPLAY CODE.				
8176					:					
8177	100306	175000	000100	000100	JSPIXA:	CUWT,100,100				
8178	100314	173000				STOPN				
8179										
8180	100316	164000			JSPIX:	DNOP				:CURSOR ON, INT'S OFF.
8181	100320	146072			JCXA:	CUON!CUIOFF				
8182	100322	152000	110302			SETCB,ACAT				:ASCII BASE ADDRESS.
8183	100326	114000	001620	001600		APNT,1620,1600				
8184	100334	102000				CHAR!LO				:ERASE OLD X ...
8185	100336	060	060	060	EX:	.ASCII /0000/				
8186	100342	114000	001620	001540		APNT,1620,1540				
8187	100350	100000				CHAR				:...AND Y NUMERICS...
8188	100352	060	060	060	EY:	.ASCII /0000/				
8189	100356	114000				APNT				
8190	100360	000000	000000		EMK:	0,0				
8191	100364	104000				SVEC				:...AND OLD MARKER.
8192	100366					SXY I,-10,-10				
8193	100370					SXY U,20,0				
8194	100372					SXY I,-20,20				
8195	100374					SXY U,20,0				
8196	100376					SXY I,-10,-10				
8197										
8198	100400	114000	001460	001600		APNT,1460,1600				
8199	100406	103774				CHAR!ALL				:DISPLAY NEW X...
8200	100410	112	123	130		.ASCII /JSX /				
8201	100414	060	060	060	JSX:	.ASCII /0000/				
8202	100420	114000	001460	001540		APNT,1460,1540				
8203	100426	100000				CHAR				:...AND Y...
8204	100430	112	123	131		.ASCII /JSY /				
8205	100434	060	060	060	JSY:	.ASCII /0000/				
8206	100440	114000				APNT				
8207	100442	000000	000000		MARK:	0,0				:...AND NEW MARKER.
8208	100446	104000				SVEC				
8209	100450					SXY I,-10,-10				
8210	100452					SXY U,20,0				
8211	100454					SXY I,-20,20				
8212	100456					SXY U,20,0				
8213	100460					SXY I,-10,-10				
8214										
8215	100462	114000	001540	001500	MATCH:	APNT,1540,1500				: 'CHAR!ALL' IF MATCH INTERRUPT
8216	100470	102000				CHAR!LO				
8217	100472	040	115	101		.ASCII / MATCH/				
8218										
8219	100500	114000	000010	000716		APNT,10,HAFY60-20				:NOW LABEL THE BOX VECTORS.
8220	100506	103774				CHAR!ALL				
8221	100510	060	060	060		.ASCII /0000/				
8222	100514	114000	000260	000716		APNT,260,HAFY60-20				
8223	100522	103774				CHAR!ALL				
8224	100524	060	062	065		.ASCII /0250/				
8225	100530	114000	000530	000716		APNT,530,HAFY60-20				
8226	100536	103774				CHAR!ALL				
8227	100540	060	065	062		.ASCII /0520/				
8228	100544	114000	001116	000716		APNT,1256-140,HAFY60-20				
8229	100552	103774				CHAR!ALL				
8230	100554	061	062	065		.ASCII /1256/				

8231	100560	114000	001366	000716		APNT,1526-140,HAFY60-20	
8232	100566	103774				CHAR!ALL	
8233	100570	061	065	062		.ASCII /1526/	
8234	100574	114000	001636	000716		APNT,1776-140,HAFY60-20	
8235	100602	103774				CHAR!ALL	
8236	100604	061	067	067		.ASCII /1776/	
8237	100610	114000	000716	000010		APNT,HAFX-60,10	
8238	100616	103774				CHAR!ALL	
8239	100620	060	060	060		.ASCII /0000/	
8240	100624	114000	000716	000250		APNT,HAFX-60,250	
8241	100632	103774				CHAR!ALL	
8242	100634	060	062	064		.ASCII /0240/	
8243	100640	114000	000716	000510		APNT,HAFX-60,510	
8244	100646	103774				CHAR!ALL	
8245	100650	060	065	060		.ASCII /0500/	
8246	100654	114000	000716	001136		APNT,HAFX-60,1176-40	
8247	100662	103774				CHAR!ALL	
8248	100664	061	061	067		.ASCII /1176/	
8249	100670	114000	000716	001376		APNT,HAFX-60,1436-40	
8250	100676	103774				CHAR!ALL	
8251	100700	061	064	063		.ASCII /1436/	
8252	100704	114000	000716	001636		APNT,HAFX-60,1676-40	
8253	100712	103774				CHAR!ALL	
8254	100714	061	066	067		.ASCII /1676/	
8255							
8256	100720	114000	001706	000010		APNT,1776-70,10	:FINALLY...
8257	100726	103774				CHAR!ALL	
8258	100730	107	120			.ASCII /GP/	:...MY INITIALS.
8259							
8260	100732	164000	164000	164000	BOXES:	DNOP, DNOP, DNOP	:*** FOR DEBUG ***
8261	100740	146016			JCXB:	CUIM	: MATCH ON, SWITCH OFF.
8262	100742	114000	000000	000000		APNT,0,0	
8263	100750	113774				LVEC!ALL	:AND DRAW NESTED BOXES.
8264	100752					LXY 1,1776,0	
8265	100756					LXY 1,0,1676	
8266	100762					LXY 1,-1776,0	
8267	100766					LXY 1,0,-1676	
8268	100772	114000	000250	000240		APNT,250,240	
8269	101000	110000				LVEC	
8270	101002					LXY 1,1256,0	
8271	101006					LXY 1,0,1176	
8272	101012					LXY 1,-1256,0	
8273	101016					LXY 1,0,-1176	
8274	101022	114000	000520	000500		APNT,520,500	
8275	101030	110000				LVEC	
8276	101032					LXY 1,536,0	
8277	101036					LXY 1,0,476	
8278	101042					LXY 1,-536,0	
8279	101046					LXY 1,0,-476	
8280	101052	114000	000676	000636		APNT,HAFX-64.,HAFY60-64.	
8281	101060	134001				BM04!M64	: A 64 SQUARE BIT MAP AT CENTER.
8282	101062	000000			BMXX.	0	: POINTS TO FREE CORE BUFFER.
8283	101064	114000	000756	000736		APNT,HAFX-20,HAFY60	
8284	101072	106000				SVEC!LO	: CROSS AT CENTER (BLK ON WHITE)
8285	101074					SXY 1,40,0	
8286	101076					SXY 0,-20,20	
8287	101100					SXY 1,0,-40	

8288
8289 101102 164000 164000 164000
8290 101110 146013 164003
8291 101114 146100 173400
8292 101120 164000 164000 164000
8293 101126 160000 100732
8294
8295 101132

JCXC: DNOP,DNOP,DNOP :*** FOR DEBUG ***
 CUI,S,SYNC+2 :SWITCH ON, MATCH OFF.
JCXD: CURD,STOPI :READ JSX/JSY, STOP, INTERRUPT.
 DNOP,DNOP,DNOP :*** FOR DEBUG ***
 DJMP,BOXES :LOOP

 END.TEST

:*****
: *
: * END TEST 33
: *
:*****


```

8347
8348
8349      ; IDLE LOOP. WAIT FOR KBD ENTRY TO CHANGE MODES.
8350
8351 101342 IDLEX: PRINTF #SCOUT
8352 101362 IDLE:  GMANID SDLIST,DISSEL,D,-1,0,DTHIL,YES
8353 101402 000410 BR IDLEB
8354 101404 IDLEA: GMANID SDASK,DISSEL,D,-1,0,DTHIL,YES
8358 101424 005077 101404 IDLEB: CLR @DSR ;STOP THE DPU (IE., FOR BLOOM)
8359      ;***B JSR PC,WAITF
8360      ;***B JSR PC,WAITF
8361      ;***B JSR PC,WAITF
8362      ;***B JSR PC,WAITF ;WAIT FOR STOP
8363 101430 012701 000050 MOV #40.,R1 ;SET UP A TIMEOUT COUNTER. ;***B
8364 101434 004737 027454 1$: JSR PC,WAITF ;WAIT FOR THE STOP BIT. ;***B
8365 101440 103402 BCS 2$ ;BR IF STOPPED. ;***B
8366 101442 005301 DEC R1 ;NOT STOPPED. BUMP COUNTER. ;***B
8367 101444 001373 BNE 1$ ;LOOP IF NOT STOPPED. ;***B
8368 101446
8369 101446 013700 101470 2$: MOV DISSEL,R0
8370 101452 001743 BEQ IDLE
8371 101454 005037 101470 CLR DISSEL
8372 101460 006300 ASL R0
8373 101462 004770 101236 JSR PC,@DTAB(R0)
8374 101466 000746 BR IDLEA
8375
8376 101470 000000 DISSEL: 0 ; DISPLAY CURRENTLY SELECTED.
8377
8378 101472 040 040 060 SDLIST: .ASCII / 0 = 'TYPE THIS'/
8379 101513 015 012 .BYTE 15,12
8380 101515 040 040 061 .ASCII / 1 = TURN BLINKING ON/
8381 101543 015 012 .BYTE 15,12
8382 101545 040 040 062 .ASCII / 2 = TURN BLINKING OFF/
8383 101574 015 012 .BYTE 15,12
8384 101576 040 040 063 .ASCII / 3 = COLOR BARS/
8385 101616 015 012 .BYTE 15,12
8386 101620 040 040 064 .ASCII / 4 = 7 X 7 DOTS/
8387 101640 015 012 .BYTE 15,12
8388 101642 040 040 065 .ASCII / 5 = 7 X 7 CROSS-HATCH/
8389 101671 015 012 .BYTE 15,12
8390 101673 040 040 066 .ASCII / 6 = PERIMETER OUTLINE/
8391 101722 015 012 .BYTE 15,12
8392 101724 040 040 067 .ASCII / 7 = BASIC COLOR ID/
8393 101750 015 012 .BYTE 15,12
8394 101752 040 040 070 .ASCII / 8 = GUNS ID/
8395 101767 015 012 .BYTE 15,12
8396 101771 040 040 071 .ASCII / 9 = ALL WHITE SCREEN/
8397 102017 015 012 .BYTE 15,12
8398 102021 040 061 060 .ASCII / 10 = ALTERNATING WHITE SCREEN & PERIMETER OUTLINE/
8399 102103 015 012 .BYTE 15,12
8400 102105 040 061 061 .ASCII / 11 = ALL RED SCREEN/
8401 102131 015 012 .BYTE 15,12
8402 102133 040 061 062 .ASCII / 12 = ALL BLUE SCREEN/
8403 102160 015 012 .BYTE 15,12
8404 102162 040 061 063 .ASCII / 13 = ALL GREEN SCREEN/
8405 102210 015 012 .BYTE 15,12
8406 102212 015 012 SDASK: .BYTE 15,12
  
```

8407	102214	050	102	114	.ASCII	/(BLINKING IS /
8408	102231	117	106	106	.ASCII	/OFF), /
8409	102241	123	105	114	.ASCII	/SELECT DISPLAY /
8410					.EVEN	
8411						

```
8413
8414
8415
8416 102262 004737 105330
8417 102266 112737 000116 102232
8418 102274 112737 000051 102233
8419 102302 112737 000054 102234
8420 102310 112737 000040 102235
8421 102316 000207
8422
8423
8424
8425
8426
8427
8428
8429 102320 004737 105356
8430 102324 112737 000106 102232
8431 102332 112737 000106 102233
8432 102340 112737 000051 102234
8433 102346 112737 000054 102235
8434 102354 000207

:
: TURN BLINKING ON.
:
: TBON: JSR PC,EBLINK ; ENABLE BLINKING.
: MOVB #'N,SDO+1 ; ADJUST PROMPT MSG.
: MOVB #' ),SDO+2
: MOVB #' ,SDO+3
: MOVB #' ,SDO+4
: RTS PC

:
: TURN BLINKING OFF.
:
: TBOFF: JSR PC,DBLINK ; DISABLE BLINKING.
: MOVB #'F,SDO+1 ; ADJUST PROMPT MSG.
: MOVB #'F,SDO+2
: MOVB #' ),SDO+3
: MOVB #' ,SDO+4
: RTS PC
```

```

8436
8437
8438
8439
8440
8441
8442
8443
8444
8445
8446 102356 004737 026650
8447 102362 004737 027066
8448 102366 012737 112000 102466
8449 102374 012701 000004
8450 102400 012702 000400
8451 102404 012777 102460 100420
8452 102412 000402
8453 102414 005277 100412
8454 102420 004737 027454
8455 102424 005302
8456 102426 001411
8457 102430 060137 102466
8458 102434 042737 176000 102466
8459 102442 052737 112000 102466
8460 102450 000761
8461
8462 102452 004737 104762
8463 102456 000207
8464
8465 102460 114000 000000 000000
8466 102466 112000
8467 102470 040000 001776
8468 102474 000002 000000
8469 102500 040000 021776
8470 102504 000002 000000
8471 102510 173000
8472 102512 160000 102466
  
```

```

: COLOR BAR GENERATOR.
: NUMBER AND COLOR OF BARS DEPENDANT ON PIXEL SIZE.
: ROUTINE DEFAULTS TO 8 BAR NTSC RAINBOW (IF LUT INSTALLED),
: OR GRAY-SCALE SHADES DEPENDANT ON PIXEL SIZE AS FOLLOWS:
:   2 BITS <9:8> = 4 COLORS/SHADES.
:   4 BITS <9:6> = 16 COLORS/SHADES.
:   6 BITS <9:4> = 64 COLORS/SHADES.
:   8 BITS <9:2> = 256 COLORS/SHADES.
  
```

```

BARS: JSR PC,CLRRW ;INIT DISPLAY.
      JSR PC,NTSC ; NTSC COLORS (IF LUT AVAILABLE)
      MOV #LVEC!LO,BV ;INIT DISPLAY CODE AT LVL 0...
      MOV #4,R1 ;...AND INCREMENT BY LVL 1.
1$: MOV #256.,R2 ; 256 PAIRS = 512 TOTAL.
     MOV #BGEN,@DPC
     SKP2
2$: INC @DPC
     JSR PC,WAITF
     DEC R2
     BEQ 3$ ;BR WHEN FINISHED.
     ADD R1,BV ; OTHERWISE, INCREMENT...
     BIC #^C1777,BV ;...AND STRIP THE LEVEL...
     BIS #LVEC!LO,BV ;...AND REPLACE OPCODE.
     BR 2$ ;...AND LOOP.
3$: JSR PC,OUTLIN ; OUTLINE THE SCREEN.
    RTS PC ; RETURN TO IDLE LOOP.
  
```

```

BGEN: APNT, 0, 0
BV: LVEC!LO
    I!0, MAXY ; UP...
    2, 0 ;...AND OVER...
    I!0, MXY!MAXY ;...DOWN...
    2, 0 ;...AND OVER AGAIN.
    STOPN
    DJMP, BV
  
```


8474
8475
8476
8477 102516
8478 102516 000104
8479 102520 000330
8480 102522 000554
8481 102524 001000
8482 102526 001224
8483 102530 001450
8484 102532 001674
8485 102534 177777
8486 102536 000110
8487 102540 000320
8488 102542 000530
8489 102544 000740
8490 102546 001150
8491 102550 001360
8492 102552 001570
8493 102554 177777

∴ 7 X 7 POINT TABLE FOR DOTS AND CROSS-HATCH.
∴
XTBL: ;X POINTS...
Y50TBL: 34.*2 ;...AND 50 HZ Y POINTS.
108.*2 ;DELTA = 74.
182.*2
256.*2
330.*2
404.*2
478.*2
-1
Y60TBL: 36.*2 ; 60 HZ Y POINTS.
104.*2 ;DELTA = 68.
172.*2
240.*2
308.*2
376.*2
444.*2
-1

```

8495
8496          : DOT GENERATOR.
8497          :
8498 102556 004737 026650          : DOTS: JSR PC,CLRRW      ;INIT DISPLAY.
8499 102562 012702 102536          :      MOV #Y60TBL,R2    ;ASSUME 60 HZ Y'S.
8500 102566 023727 003246 000074 :      CMP HZ,#60.
8501 102574 001402          :      BEQ 1$
8502 102576 012702 102516          :      MOV #Y50TBL,R2    ;WRONG, USE 50 HZ Y'S.
8503
8504 102602 012237 102664          : 1$:  MOV (R2)+,DGY      ;SET Y POINT.
8505 102606 100416          :      BMI 3$           ;BR WHEN ALL DOTS DONE.
8506 102610 012701 102516          :      MOV #XTBL,R1
8507 102614 012137 102662          : 2$:  MOV (R1)+,DGX      ;SET X POINT.
8508 102620 100770          :      BMI 1$           ;X DONE, GET NEXT Y.
8509 102622 052737 040000 102662 :      BIS #I,DGX        ;SET I BIT.
8510 102630 012777 102660 100174 :      MOV #DGEN,@DPC
8511 102636 004737 027454          :      JSR PC,WAITF
8512 102642 000764          :      BR 2$           ;LOOP
8513
8514 102644 012737 000001 105070 : 3$:  MOV #1,OUTFLG      ; SAY WE WANT DOTS ON THE OUTLINE
8515 102652 004737 104762          :      JSR PC,OUTLIN    ; OUTLINE THE SCREEN.
8516 102656 000207          :      RTS PC           ; RETURN TO IDLE LOOP.
8517
8518 102660 117774          : DGEN: APNT!ALL
8519 102662 040000          : DGX:  I!0
8520 102664 000000          : DGY:  0
8521 102666 173000          :      STOPN           ; *** NOP TO ENLARGE THE DOTS ***
8522 102670 130000          :      RPNT
8526 102700 173000          :      STOPN
  
```

```

8528
8529      : CROSS-HATCH GENERATOR.
8530      :
8531 102702 004737 026650      HATCH: JSR    PC,CLRRW      ;INIT DISPLAY.
8532 102706 005037 103052      CLR    CHX
8533 102712 012737 041776 103060  MOV    #I!MAXX,CHV      ;SET FOR HORIZ VECTORS.
8534 102720 005037 103062      CLR    CHV+2
8535 102724 012701 102536      MOV    #Y60TBL,R1      ;ASSUME 60 HZ Y'S.
8536 102730 023727 003246 000074  CMP    HZ,#60.
8537 102736 001402      BEQ    1$
8538 102740 012701 102516      MOV    #Y50TBL,R1      ;WRONG AGAIN.
8539
8540 102744 012137 103054      1$:  MOV    (R1)+,CHY      ;SET NEXT Y.
8541 102750 100406      BMI    2$              ;BR AT END OF Y'S.
8542 102752 012777 103050 100052  MOV    #HGEN,@DPC
8543 102760 004737 027454      JSR    PC,WAITF
8544 102764 000767      BR     1$              ;LOOP FOR 7 HORIZ VECTORS.
8545
8546 102766 005037 103054      2$:  CLR    CHY
8547 102772 012737 040000 103060  MOV    #I!0,CHV        ;SET FOR VERT VECTORS.
8548 103000 012737 001776 103062  MOV    #MAXY,CHV+2
8549 103006 012701 102516      MOV    #XTBL,R1
8550 103012 012137 103052      3$:  MOV    (R1)+,CHX      ;SET NEXT X.
8551 103015 100406      BMI    4$              ;BR AT END OF X'S.
8552 103020 012777 103050 100004  MOV    #HGEN,@DPC
8553 103026 004737 027454      JSR    PC,WAITF
8554 103032 000767      BR     3$
8555 103034 012737 000001 105070  4$:  MOV    #1,OUTFLG      ;LOOP FOR 7 VERT VECTORS.
8556 103042 004737 104762      JSR    PC,OUTLIN      ; SAY WE WANT DOTS ON THE OUTLINE.
8557 103046 000207      RTS    PC              ; OUTLINE THE SCREEN.
8558                                     ; RETURN TO IDLE LOOP.
8559 103050 114000      HGEN:  APNT
8560 103052 000000      CHX:   0
8561 103054 000000      CHY:   0
8562 103056 113774      LVEC!ALL
8563 103060 041776 000000      CHV:   I!MAXX,0
8564 103064 173000      STOPN
  
```

```
8566  
8567      ; DISPLAY A SQUARE, OUTLINING THE PERIMETER OF THE SCREEN.  
8568      ;  
8569 103066 004737 026650 PERIM: JSR PC,CLRRW      ; CLEAR FOR READ/WRITE.  
8570 103072 012737 000001 105070 MOV #1,OUTFLG    ; SAY WE WANT DOTS ON THE OUTLINE  
8571 103100 004737 104762 JSR PC,OUTLIN    ; OUTLINE THE SCREEN.  
8572 103104 000207 RTS PC      ; RETURN TO IDLE LOOP.
```

```

8574
8575      : DISPLAY THE FOUR BASIC COLORS (BLUE, RED, LT GREEN, DK GREEN)
8576      : AND DISPLAY THE NAME OF EACH UNDERNEATH.
8577      : EACH COLOR BAND WILL BE 1/4 SCREEN WIDTH.
8578
8579 103106 004737 026650 COLID: JSR PC,CLRRW ; CLEAR FOR READ/WRITE.
8580 103112 012737 112000 103240 MOV #LVEC!LO,11$
8581 103120 012701 000100 MOV #100,R1 ; START WITH BLUE.
8582 103124 050137 103240 BIS R1,11$
8583 103130 012702 000400 MOV #400,R2 ; (R2 DIVIDES SCREEN BY 4 & CHANGES COLORS).
8584 103134 012777 103232 077670 MOV #10$,@DPC ; START THE DPU.
      103142 004737 027454 JSR PC,WAITF ; WAIT FOR DISPLAY STOP.
8585 103146 000405 BR 3$
8586
8587 103150 2$:
      103150 052777 000001 077654 BIS #BIT0,@DPC ; CONTINUE THE DPU.
      103156 004737 027454 JSR PC,WAITF ; WAIT FOR DISPLAY STOP.
8588 103162 005302 3$: DEC R2 ; ALL DONE?
8589 103164 001412 BEQ 4$ ; YES.
8590 103166 032702 000077 BIT #77,R2 ; NO. TIME TO CHANGE COLOR?
8591 103172 001366 BNE 2$ ; NO.
8592 103174 006301 ASL R1 ; YES.
8593 103176 042737 001700 103240 BIC #1700,11$
8594 103204 050137 103240 BIS R1,11$
8595 103210 000757 BR 2$
8596
8597 103212 4$:
      103212 012777 103270 077612 MOV #12$,@DPC ; START THE DPU.
      103220 004737 027454 JSR PC,WAITF ; WAIT FOR DISPLAY STOP.
8598 103224 004737 104762 JSR PC,OUTLIN ; OUTLINE THE SCREEN.
8599 103230 000207 RTS PC ; RETURN TO IDLE LOOP.
8600
8601      :
8602      : DISPLAY CODE.
8603
8604 103232 114000 000000 000000 10$: APNT,0,0
8605 103240 112000 11$: LVEC!LO
8606 103242 040000 001776 I!0,MAXY
8607 103246 000002 000000 2,0
8608 103252 040000 021776 I!0,MXY!MAXY
8609 103256 000002 000000 2,0
8610 103262 173000 STOPN
8611 103264 160000 103240 DJMP,11$
8612 103270 114000 000004 000004 12$: APNT,4,4
8613 103276 152000 110302 SETCB,ACAT
8614 103302 103774 CHAR!ALL
8615 103304 102 114 125 .ASCIZ /BLUE RED LSB-GRN MSB-GRN /
8616 .EVEN
8617 103356 173000 STOPN
  
```

```

8619
8620 : GUNS ID DISPLAY
8621 : THIS DISPLAY IS SIMILAR TO THE COLOR BARS DISPLAY, EXCEPT:
8622 : 1. THE BARS ARE HORIZONTAL.
8623 : 2. THE GUNS ACTIVATED FOR EACH BAR ARE PRINTED INSIDE EACH BAR.
8624
8625 103360 004737 026650 GUNID: JSR PC,CLRRW ; CLEAR FOR READ/WRITE.
8626 103364 012777 103636 077440 MOV #10$,@DPC ; START THE DPU.
      103372 004737 027454 JSR PC,WAITF ; WAIT FOR DISPLAY STOP.
8627 103376 012704 112000 MOV #LVEC!LO,R4 ; INIT VECTOR CONTROL.
8628 103402 012702 000020 MOV #16.,R2 ; INIT BAR COUNT.
8629 103406 012703 000017 1$: MOV #15.,R3 ; INIT VECTORS PER BAR -
8630 103412 023727 003246 000074 CMP HZ,#60. ; - 15 FOR 60HZ, 16 FOR 50HZ.
8631 103420 001402 BEQ 2$
8632 103422 012703 000020 MOV #16.,R3
8633 103426 010437 103646 2$: MOV R4,11$ ; MODIFY DISPLAY FILE VECTOR CONTROL.
8634 103432 3$:
      103432 052777 000001 077372 BIS #BIT0,@DPC ; CONTINUE THE DPU.
      103440 004737 027454 JSR PC,WAITF ; WAIT FOR DISPLAY STOP.
8635 103444 005303 DEC R3 ; THIS BAR COMPLETED?
8636 103446 001371 BNE 3$ ; NO.
8637 103450 062704 000100 ADD #100,R4 ; YES. UPDATE VECTOR CONTROL.
8638 103454 005302 DEC R2 ; ALL BARS COMPLETED?
8639 103456 001353 BNE 1$ ; NO.
8640 103460 012704 000004 MOV #4,R4 ; YES. INIT MSG POSITION CONTROL.
8641 103464 012701 103776 MOV #30$,R1 ; INIT MESSAGE POINTER.
8642 103470 012702 000020 MOV #16.,R2 ; INIT MESSAGE COUNTER.
8643 103474 012737 103774 103720 MOV #CHAR!ALL,23$ ; INIT DISPLAY FILE INTENSITY CONTROL.
8644 103502 012777 103676 077322 MOV #20$,@DPC ; START THE DPU.
      103510 004737 027454 JSR PC,WAITF ; WAIT FOR DISPLAY STOP.
8645 103514 010437 103712 4$: MOV R4,22$ ; MODIFY DISPLAY FILE MSG POSITION CONTROL.
8646 103520 012700 103722 MOV #24$,R0 ; MODIFY DISPLAY FILE MESSAGE.
8647 103524 112120 5$: MOVB (R1)+,(R0)+
8648 103526 001376 BNE 5$
8649 103530 112760 000040 177777 MOVB #' ,-1(R0)
8650 103536 112720 000040 6$: MOVB #' ,(R0)+
8651 103542 020027 103770 CMP R0,#25$
8652 103546 001373 BNE 6$
8653 103550 105060 177777 CLRB -1(R0)
8654 103554 052777 000001 077250 BIS #BIT0,@DPC ; CONTINUE THE DPU.
      103562 004737 027454 JSR PC,WAITF ; WAIT FOR DISPLAY STOP.
8655 103566 062704 000074 ADD #74,R4 ; UPDATE MSG POSITION CONTROL.
8656 103572 023727 003250 001776 CMP YMAX,#MAXY50 ; ARE WE RUNNING 50HZ?
8657 103600 001002 BNE 60$ ; BR IF NO
8658 103602 062704 000004 ADD #4,R4 ; YES -- NEED TO BUMP UP 2 MORE PIXELS.
8659 103606 005302 60$: DEC R2 ; ALL MSGS DONE?
8660 103610 001407 BEQ 7$ ; YES.
8661 103612 020227 000010 CMP R2,#10 ; TIME TO FLIP INTENSITY?
8662 103616 001336 BNE 4$ ; NOT YET.
8663 103620 012737 102000 103720 MOV #CHAR!LO,23$ ; YES. CHANGE INTENSITY FROM WHITE TO BLACK.
8664 103626 000732 BR 4$
8665
8666 103630 004737 104762 7$: JSR PC,OUTLIN ; OUTLINE THE SCREEN.
8667 103634 000207 RTS PC ; RETURN TO IDLE LOOP.
8668
8669
8670 : DISPLAY FILES.

```

```

8671
8672 103636 114000 000000 000000 10$: APNT,0,0
8673 103644 173000 STOPN
8674 103646 112000 11$: LVEC!LO ; THE COLOR BARS...
8675 103650 041776 000000 I!MAXX,0
8676 103654 000000 000002 0,2
8677 103660 061776 000000 I!MXY!MAXX,0
8678 103664 000000 000002 0,2
8679 103670 173000 STOPN
8680 103672 160000 103646 DJMP,11$
8681
8682 103676 114000 000000 000000 20$: APNT,0,0
8683 103704 173000 STOPN
8684 103706 114000 21$: APNT ; THE IDENTIFICATION MESSAGES.
8685 103710 000004 4
8686 103712 000004 22$: 4
8687 103714 152000 110302 SETCB,ACAT
8688 103720 103774 23$: CHAR!ALL
8689 103722 130 130 24$: .ASCIZ /XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX/
8690 103770 173000 25$: STOPN
8691 103772 160000 103706 DJMP,21$
8692
8693
8694
8695 ; MESSAGES FOR DISPLAY.
8696
8697 103776 050 060 060 30$: .ASCIZ /(0000)/
8698 104005 050 060 060 .ASCIZ /(0001) BLUE/
8699 104022 050 060 060 .ASCIZ /(0010) RED/
8700 104036 050 060 060 .ASCIZ /(0011) BLUE, RED/
8701 104060 050 060 061 .ASCIZ /(0100) LSB-GRN/
8702 104100 050 060 061 .ASCIZ /(0101) BLUE, LSB-GRN/
8703 104126 050 060 061 .ASCIZ /(0110) RED, LSB-GRN/
8704 104153 050 060 061 .ASCIZ /(0111) BLUE, RED, LSB-GRN/
8705 104206 050 061 060 .ASCIZ /(1000) MSB-GRN/
8706 104226 050 061 060 .ASCIZ /(1001) BLUE, MSB-GRN/
8707 104254 050 061 060 .ASCIZ /(1010) RED, MSB-GRN/
8708 104301 050 061 060 .ASCIZ /(1011) BLUE, RED, MSB-GRN/
8709 104334 050 061 061 .ASCIZ /(1100) LSB-GRN, MSB-GRN/
8710 104365 050 061 061 .ASCIZ /(1101) BLUE, LSB-GRN, MSB-GRN/
8711 104424 050 061 061 .ASCIZ /(1110) RED, LSB-GRN, MSB-GRN/
8712 104462 050 061 061 .ASCIZ /(1111) BLUE, RED, LSB-GRN, MSB-GRN/
8713 .EVEN
  
```

```

8715
8716      ; DISPLAYS FOR FILLING SCREEN WITH SOLID COLOR
8717      ;
8718 104526 012746 003774      SCRWHT: MOV #ALL,-(SP)
8719 104532 000410              BR      SCRCOM      ; GO TO COMMON SECTION
8720
8721 104534 012746 002200      SCRRED: MOV #LO!200,-(SP) ; GET RED ONTO STACK,
8722 104540 000405              BR      SCRCOM      ; GO TO COMMON
8723
8724 104542 012746 002100      SCRBLU: MOV #LO!100,-(SP) ; GET BLUE ONTO STACK,
8725 104546 000402              BR      SCRCOM      ; GO TO COMMON
8726
8727 104550 012746 003400      SCRGRN: MOV #LO!1400,-(SP) ; GET FULL GREEN ONTO STACK,
8728
8729 104554 004737 026650      SCRCOM: JSR PC,CLRRW      ; CLEAR FOR READ/WRITE
8730 104560 004737 027066              JSR PC,NTSC      ; SETUP COLOR TABLE (IF AVAILABLE)
8731 104564 012737 114000 104612      MOV #APNT,10$      ; INIT PIXEL DATA SELECTION
8732 104572 052637 104612              BIS (SP)+,10$      ; INSERT THE COLOR
8733 104576 012777 104612 076226      MOV #10$,@DPC      ; START THE DPU
8734 104604 004737 027454              JSR PC,WAITF      ; WAIT FOR DONE
8735 104610 000207              RTS PC      ; RETURN TO IDLE
8736
8737 104612 117774      10$: APNT!ALL
8738 104614 170100              SETMEM      ; SET THE MEMORY
8739 104616 173000              STOPN
  
```



```

8741
8742
8743
8744
8745
8746 104620 004737 025402
8747 104624 004737 102320
8748 104630 004737 027066
8749 104634 013737 003250 104730
8750 104642 013737 003250 104740
8751 104650 052737 020000 104740
8752 104656 012777 104666 076146
8753 104664 000207
8754
8755
8756
8757
8758 104666 176034
8759 104670 176434
8760 104672 177034
8761 104674 177434
8762
8763 104676 117774 000000 000000
8764 104704 170100
8765 104706 170200
8766 104710 164170
8767 104712 170200
8768 104714 170140
8769 104716 117774 000000 000000
8770 104724 110000
8771 104726 040000
8772 104730 001776
8773 104732 041776
8774 104734 000000
8775 104736 040000
8776 104740 021776
8777 104742 061776
8778 104744 000000
8779 104746 170200
8780 104750 164170
8781 104752 170200
8782 104754 160000 104676
8783
8784
  
```

```

:
: 'BLOOM' TEST DISPLAY
: THIS DISPLAY ALTERNATELY DISPLAYS A FULL WHITE SCREEN AND A PERIMETER OUTLINE.
: THE DISPLAY CHANGES ONCE EVERY 2 SECONDS.
:
  
```

```

BLOOM: JSR PC,DPINIT ;INIT THE DPU
        JSR PC,TBOFF ;TURN BLINKING OFF
        JSR PC,NTSC ;SET UP THE COLOR TABLE (IF AVAILABLE)
        MOV YMAX,12$ ;SET UP THE MAX Y VALUES, BASED ON FREQUENCY.
        MOV YMAX,13$
        BIS #MXY,13$
        MOV #10$,@DPC ;START THE DISPLAY & LET IT RUN.
        RTS PC ;RETURN TO THE IDLE LOOP
  
```

```

: DISPLAY FILE:
:
  
```

```

10$: WRT!CHO ;SET ALL CHANNELS TO WRITE-ONLY, SWITCH-ENABLED.
     WRT!CH1
     WRT!CH2
     WRT!CH3
: LOOP TO HERE:
11$: APNT!ALL,0,0 ;SET WHITE PIXEL DATA
     SETMEM ;SET MEMORIES TO ALL WHITE
     SWTCH ;SWITCH TO READ-ONLY TO DISPLAY WHITE SCREEN
     DNOP+120. ;SHOW IT FOR 2 SECONDS.
     SWTCH ;SWITCH TO WRITE-ONLY MODE.
     CLRMEM ;CLEAR MEMORIES.
     APNT!ALL,0,0 ;START PERIMETER OUTLINE AT <0,0>
     LVEC ;ENTER LONG-VECTOR MODE.
12$: I!0 ;DRAW LEFT EDGE
     MAXY ;... TO TOP OF VISIBLE AREA.
     I!MAXX ;DRAW TOP EDGE
     0 ;... TO RIGHT SIDE.
     I!0 ;DRAW RIGHT EDGE ...
     ;... TO BOTTOM.
13$: MXY!MAXY ;DRAW BOTTOM EDGE ...
     I!MXY!MAXX ;... BACK TO <0,0>
     0 ;SWITCH TO READ-ONLY TO DISPLAY PERIMETER
     SWTCH ;SHOW IT FOR 2 SECONDS.
     DNOP+120. ;SWITCH BACK TO WRITE-ONLY.
     SWTCH ;LOOP BACK TO DO IT ALL OVER AGAIN.
     DJMP,11$
  
```

```

:*****
:
: END TEST 34
:
:*****
  
```

```

8786
8787
8788
8789
8790
8791
8792
8793 104762 013737 003250 105050
8794 104770 013737 003250 105060
8795 104776 052737 020000 105060
8796 105004 012777 105036 076020
8797 105012 004737 027454
8798 105016 005737 105070
8799 105022 001404
8799 105024 004737 105072
8800 105030 005037 105070
8801 105034
8802 105034 000207
8803
8804
8805
8806
8807 105036 114000 000000 000000
8808 105044 113774
8809 105046 040000
8810 105050 001776
8811 105052 041776
8812 105054 000000
8813 105056 040000
8814 105060 021776
8815 105062 061776
8816 105064 000000
8817 105066 173000
8818
8819
8820 105070 000000
8821 105072 013737 003250 105162
8822 105100 013746 003250
8823 105104 062716 000002
8824 105110 006216
8825 105112 006216
8826 105114 012637 105302
8827 105120 012777 105134 075704
8828 105126 004737 027454
8829 105132 000207
8830
8831 105134 116000 000000 000000
8832 105142 160001 105246
8833 105146 160001 105246
8834 105152 160001 105246
8835 105156 116000
8836 105160 000000
8837 105162 001676
8838 105164 160001 105246
8839 105170 160001 105246
8840 105174 160001 105246

```

: SOME MISCELLANEOUS DISPLAY SUBROUTINES.

```

: SUBROUTINE TO OUTLINE THE PERIMETER OF THE SCREEN.
OUTLIN: MOV     YMAX,11$      ; SET Y COORDINATES AS PER HZ.
        MOV     YMAX,12$
        BIS     #MXY,12$
        MOV     #10$,@DPC    ; START THE DPU.
        JSR     PC,WAITF     ; WAIT FOR DISPLAY STOP.
        TST     OUTFLG      ; MAKE IT DOTTED (3 DOTS PER SIDE)?
        BEQ     1$          ; BR IF NO
        JSR     PC,OUTDOT    ; YES -- GO DO IT
        CLR     OUTFLG      ; THEN CLEAR THE FLAG
1$:     RTS     PC

: DISPLAY CODE.
10$:    APNT,0,0
        LVEC!ALL
        I!0
11$:    MAXY
        I!MAXX
        0
        I!0
12$:    MXY!MAXY
        I!MXY!MAXX
        0
        STOPN

:ROUTINE TO MAKE 3 DOTS IN EACH LINE OF THE PERIMETER:
OUTFLG: 0 ;FLAG FOR ENABLING
OUTDOT: MOV     YMAX,OUTDT1  ; SET UP THE DISPLAY FILE
        MOV     YMAX,-(SP)   ; ... WITH FREQ.- DEPENDENT Y DATA.
        ADD     #2,(SP)
        ASR     (SP)
        ASR     (SP)
        MOV     (SP)+,OUTDY1
        MOV     #OUTDTF,@DPC ; START THE DPU.
        JSR     PC,WAITF     ; WAIT FOR DISPLAY STOP.
        RTS     PC

: DISPLAY FILE FOR OUTLINE DOTS:
OUTDTF: APNT!LO,0,0 ; BEGIN AT 0,0 WITH BLACK
        DJMS,OUTDFX ; DO DOT AT 400,0
        ; ...AND AT 1000,0
        DJMS,OUTDFX ; ... AND AT 1400,0.
        APNT!LO
        0
OUTDT1: MAXY60 ; GETS YMAX
        DJMS, OUTDFX ; DO DOT AT 400,YMAX
        ; ... AND AT 1000, YMAX
        DJMS, OUTDFX ; ... AND AT 1400,YMAX

```

```

8841 105200 116000 000000 000000      APNT!LO,0,0      : GO AGAIN TO 0,0
8842 105206 160001 105276      DJMS, OUTDFY    : DO A DOT AT 0,YMAX+2/4
8843 105212 160001 105276      DJMS, OUTDFY    : ... AND AT 0, 2*<YMAX+2/4>
8844 105216 160001 105276      DJMS, OUTDFY    : ... AND AT 0, 3*<YMAX+2/4>
8845 105222 116000 001776 000000      APNT!LO,MAXX,0
8846 105230 160001 105276      DJMS, OUTDFY    : DO A DOT AT 0,YMAX+2/4
8847 105234 160001 105276      DJMS, OUTDFY    : ... AND AT 0, 2*<YMAX+2/4>
8848 105240 160001 105276      DJMS, OUTDFY    : ... AND AT 0, 3*<YMAX+2/4>
8849 105244 173000
8850
8851 105246 110000      OUTDFX: LVEC
8852 105250 000376 000000      MAXX+2/4-2,0    : SPACE OVER +400-2
8853 105254 060016 000000      I!M!14.,0       : MAKE 8 BLANK PIXELS
8854 105260 000022 000000      18.,0           : GO TO BEYOND THE TARGET POINT
8855 105264 040016 000000      I!14.,0         : MAKE 8 BLANK PIXELS
8856 105270 020020 000000      M!16.,0         : GO BACK TO TARGET
8857 105274 165000      DPOP            : RETURN
8858
8859 105276 112000      OUTDFY: LVEC!LO
8860 105300 000000      0
8861 105302 000360      OUTDY1: MAXY60+2/4 : GETS YMAX+2/4
8862 105304 000000 020020      0,M!16.         : BACK UP
8863 105310 040000 000040      I!0,32.         : ZAP 17 PIXELS
8864 105314 000000 020020      0,M!16.         : BACK UP
8865 105320 113774      LVEC!WHITE
8866 105322 040000 000000      I.0,0           : WRITE A DOT
8867 105326 165000      DPOP
8868
8869
8870
8871
8872      : SUBROUTINE TO ENABLE BLINKING (ALL CHANS).
8873
8874 105330      EBLINK:
8875 105330 012777 105344 075474      MOV #10$,@DPC    : START THE DPU.
8876 105336 004737 027454      JSR PC,WAITF    : WAIT FOR DISPLAY STOP.
8877 105342 000207      RTS PC
8878
8879      : DISPLAY CODE.
8880 105344 175401      10$: BLINK
8881 105346 175501      BLINK+100
8882 105350 175601      BLINK+200
8883 105352 175701      BLINK+300
8884 105354 173000      STOPN
8885
8886
8887
8888
8889      : SUBROUTINE TO DISABLE BLINKING (ALL CHANS).
8890
8891 105356      DBLINK:
8892 105356 012777 105372 075446      MOV #10$,@DPC    : START THE DPU.
8893 105364 004737 027454      JSR PC,WAITF    : WAIT FOR DISPLAY STOP.
8894 105370 000207      RTS PC
  
```

8894
8895
8896
8897 105372 175400
8898 105374 175500
8899 105376 175600
8900 105400 175700
8901 105402 173000

⋮ DISPLAY CODE.
108: NBLINK
NBLINK+100
NBLINK+200
NBLINK+300
STOPN

8952	105716	006302		ASL	R2	:	
8953	105720	006302		ASL	R2	:	
8954	105722	006302		ASL	R2	:	
8955	105724	006302	4\$:	ASL	R2	:	
8956	105726	100002		BPL	5\$:	
8957	105730	005305		DEC	R5	:	
8958	105732	001374		BNE	4\$:	
8959	105734	010102	5\$:	MOV	R1,R2	:	GET NUMBER OF THIS MEM IN R2.
8960	105736	162702	003260	SUB	#MEMTAB+2,R2	:	
8961	105742	006202		ASR	R2	:	
8962	105744	016104	177776	MOV	-2(R1),R4	:	GET TABLE ENTRY FOR THIS MEM IN R4.
8963	105750			PRINTF	#SCMEM,R2,R4,R3	:	PRINT DATA FOR THIS MEM.
8964	105776	005705		TST	R5	:	NON-STANDARD?
8965	106000	001720		BEQ	1\$:	NO. NEXT MEM.
8966	106002			PRINTF	#SCNSTD	:	YES. TELL THE WORLD!.
8967	106022	000707		BR	1\$:	NEXT MEM.
8968						:	
8969	106024	012102	6\$:	MOV	(R1)+,R2	:	GET SYNC CHAN TABLE ENTRY. END OF TABLE?
8970	106026	100376		BPL	6\$:	NO. (BUT NO SYNC CCHAN THERE).
8971	106030	020227	177777	CMP	R2,#-1	:	MAYBE. IS IT EOT OR A SYNC CHAN?
8972	106034	001443		BEQ	8\$:	EOT.
8973	106036	010103		MOV	R1,R3	:	SYNC CHAN. GET ITS NUMBER IN R3.
8974	106040	162703	003272	SUB	#SYCTAB+2,R3	:	
8975	106044	006203		ASR	R3	:	
8976	106046			PRINTF	#SCSYC,R3,R2	:	PRINT DATA FOR THIS SYNC CHAN.
8977	106072	032702	020000	BIT	#BIT13,R2	:	INTERLACED?
8978	106076	001011		BNE	7\$:	YES.
8979	106100			PRINTF	#SCNINT	:	NO.
8980	106120	000741		BR	6\$:	NEXT SYNC CHAN.
8981						:	
8982	106122		7\$:	PRINTF	#SCINT	:	
8983	106142	000730		BR	6\$:	NEXT SYNC CHAN.
8984						:	
8985	106144	005737	002512	8\$:	TST	MFGFLG	MFG MODE?
8986	106150	001521		BEQ	15\$:	NO.
8987	106152	013702	003022	MOV	MFGMO,R2	:	GET MFG MASTER FOR MEM 0.
8988	106156	005003		CLR	R3	:	INIT BIT COUNTER.
8989	106160	012704	000010	MOV	#8.,R4	:	INIT SHIFT COUNTER.
8990	106164	005005		CLR	R5	:	INIT NON-STDD FLAG.
8991	106166	006302		ASL	R2	:	POSITION THE MEMORY BITS.
8992	106170	006302		ASL	R2	:	
8993	106172	006302		ASL	R2	:	
8994	106174	006302		ASL	R2	:	
8995	106176	006302		ASL	R2	:	
8996	106200	006302	9\$:	ASL	R2	:	SHIFT. GOT A ONE?
8997	106202	100001		BPL	10\$:	NO.
8998	106204	005203		INC	R3	:	YES. COUNT IT.
8999	106206	005304	10\$:	DEC	R4	:	DONE ENOUGH SHIFTING?
9000	106210	001373		BNE	9\$:	NO.
9001	106212	016102	177776	MOV	-2(R1),R2	:	YES. NOW CHECK IF ALL ONE BITS ARE -
9002	106216	010305		MOV	R3,R5	:	- CONSECUTIVE, STARTING AT BIT 9 -
9003	106220	006302		ASL	R2	:	- (IF NOT, WE HAVE A NON-STANDARD -
9004	106222	006302		ASL	R2	:	- MEMORY.)
9005	106224	006302		ASL	R2	:	
9006	106226	006302		ASL	R2	:	
9007	106230	006302		ASL	R2	:	
9008	106232	006302	11\$:	ASL	R2	:	

```

9009 106234 100002          BPL      12$
9010 106236 005305          DEC      R5
9011 106240 001374          BNE
9012 106242 013704 003022 12$:  MOV      MFGM0,R4
9013 106246          PRINTF  #SCMM0,R4,R3
9014 106272 005705          TST      R5
9015 106274 001410          BEQ      13$
9016 106276          PRINTF  #SCNSTD
9017 106316 013702 003024 13$:  MOV      MFGS0,R2
9018 106322          PRINTF  #SCMMS0,R2
9019 106344 032702 020000  BIT      #BIT13,R2
9020 106350 001011          BNE      14$
9021 106352          PRINTF  #SCNINT
9022 106372 000410          BR       15$
9023
9024 106374          PRINTF  #SCINT
9025 106414 013701 002012 14$:  MOV      L$UNIT,R1
9026 106420 162701 000001 15$:  SUB      #1,R1
9027 106424 006301          ASL      R1
9028 106426 005761 003302  TST      ERTABL(R1)
9029 106432 001002          BNE      16$
9030 106434          EXIT
9031 106440          PRINTF  #SCOUT
9032 106460          PRINTF  #SCOUT
9033 106462 000776          BREAK
9034          BR       17$
9035 106464          045      116      045  SCMEM:  .ASCIZ  /%N%AMEMORY:  %D1%A,  MEMTAB VAL:  %06%A,  %D1%A BITS/
9036 106552          045      101      054  SCNSTD: .ASCIZ  /%A,  (NON-STD)/
9037 106571          045      116      045  SCSYC:  .ASCIZ  /%N%ASYNC CHAN:  %01%A,  SYCTAB VAL:  %06/
9038 106640          045      101      054  SCINT:  .ASCIZ  /%A,  INTERLACED/
9039 106660          045      101      054  SCNINT: .ASCIZ  /%A,  NON-INTERLACED/
9040 106704          045      116      045  SCMM0:  .ASCIZ  /%N%AMFG MASTER MEM 0,  MEMTAB VAL:  %06%A,  %D1%A BITS/
9041 106772          045      116      045  SCMMS0: .ASCIZ  /%N%AMFG SYNC CHAN 0,  SYCTAB VAL:  %06/
9042 107040          045      116      045  SCOUT:  .ASCIZ  /%N%N%N% ( <^C> TO GET BACK TO 'DR>' )%N/
9043          .EVEN
9044
9045 107110          END.TEST
  
```

```

*****
*
*      END TEST 35
*
*****
  
```

9047
9048
9049
9050
9051
9052
9053
9054

.SBTTL
.SBTTL SUPERVISORS DISPATCH TABLE

:++
: THIS TABLE HOLDS THE STARTING ADDRESS OF EACH TEST
: FOR THE SUPERVISOR'S DISPATCHER.
:--


```

9057 .SBTTL *** LUMINANCE (GREY-SCALE) AND COLOR DATA TABLES
9058 :
9059 : TABLE OF 256 (8 BITS) LUMINANCE VALUES TO PRODUCE
9060 : A LINEAR 12 BIT GREY-SCALE.
9061 :
9062 LUMTBL:
9063 0, 400, 1000, 1
9064 1400, 401, 2000, 1001
9065 2400, 2, 3000, 402
9066 3400, 2401, 4000, 3
9067 3001, 2020, 5000, 2420
9068 4, 5400, 3002, 6000
9069 1040, 6400, 2403, 5
9070 3003, 2040, 7400, 5020 ; LEVEL 37
9071 2404, 60, 6, 1441
9072 3004, 2023, 1042, 2405
9073 7, 4440, 2024, 1061
9074 100, 10, 462, 2025
9075 1044, 2407, 11, 501
9076 7440, 1063, 120, 12
9077 502, 3500, 1064, 2411
9078 5407, 503, 2030, 1065 ; LEVEL 77
9079 140, 14, 504, 2031
9080 1066, 2413, 15, 505
9081 7444, 2540, 160, 16
9082 3014, 3504, 2541, 125
9083 17, 2160, 2034, 200
9084 126, 3142, 2161, 3506
9085 1072, 1600, 7066, 2162
9086 2036, 220, 1601, 3126 ; LEVEL 137
9087 2163, 5161, 2563, 131
9088 4600, 513, 5162, 240
9089 132, 3146, 2165, 1222
9090 4220, 4074, 4602, 2166
9091 3513, 260, 4075, 3132
9092 3640, 1152, 2477, 3223
9093 607, 3567, 300, 210
9094 5476, 2243, 5241, 1154 ; LEVEL 177
9095 4152, 4660, 3153, 2172
9096 3517, 320, 4117, 646
9097 2173, 1230, 2555, 157
9098 647, 7624, 340, 2574
9099 5554, 2321, 2213, 2703
9100 267, 3265, 4612, 3647
9101 360, 1741, 3266, 4631
9102 2177, 5213, 5703, 1634 ; LEVEL 237
9103 671, 3651, 5176, 272
9104 1617, 6266, 1362, 4360
9105 273, 7232, 655, 3653
9106 1237, 4235, 4743, 1256
9107 7615, 1312, 5237, 3345
9108 731, 7616, 2712, 1747
9109 6235, 2331, 3656, 4364
9110 277, 3275, 2332, 5330 ; LEVEL 277
9111 2714, 5712, 3276, 7745
9112 1370, 4366, 7364, 3277
9113 2334, 2371, 4367, 5714
  
```

9114	110052	004751	006276	001372	4751.	6276.	1372.	2717	
9115	110062	005715	004752	006277	5715.	4752.	6277.	1373	
9116	110072	004371	007367	004753	4371.	7367.	4753.	2337	
9117	110102	001374	004372	005717	1374.	4372.	5717.	4754	
9118	110112	007752	001375	004373	7752.	1375.	4373.	1757	: LEVEL 337
9119	110122	004755	007753	005337	4755.	7753.	5337.	4374	
9120	110132	007372	003357	006355	7372.	3357.	6355.	7754	
9121	110142	005374	002776	005774	5374.	2776.	5774.	7337	
9122	110152	006356	005357	004376	6356.	5357.	4376.	7374	
9123	110162	004776	007756	006757	4776.	7756.	6757.	4377	
9124	110172	007375	006376	007757	7375.	6376.	7757.	2777	
9125	110202	006776	005777	006376	6776.	5777.	6376.	6377	
9126	110212	007776	006777	007377	7776.	6777.	7377.	7777	: LEVEL 377

9127
 9128 ; THE BASIC (NTSC) COLORS, PLUS BLACK AND WHITE.
 9129 ;

9130		007400			BLU=	17*BIT8			
9131		000017			RED=	17*BIT0			
9132		000360			GRN=	17*BIT4			

9133
 9134 NTSC8: 0 ; BLACK (NO COLOR AT ALL).
 9135 BLU

9136 RED
 9137 RED+BLU ; MAGENTA
 9138 GRN

9139 GRN+BLU ; CYAN
 9140 GRN+RED ; YELLOW
 9141 GRN+RED+BLU ; WHITE
 9142 ;

9143 ; A 16 LEVEL VARIATION OF THE ABOVE TO EXPERIMENT WITH.
 9144 ;

9145	110242	000000			EXPRM:	0			
9146	110244	000043				43			: BROWN (3RD LEVEL YELLOW).
9147	110246	000000				0			
9148	110250	000017				RED			: RED
9149	110252	000000				0			
9150	110254	000117				117			: ORANGE (RED + HI GREEN)
9151	110256	000000				0			
9152	110260	000377				RED+GRN			: YELLOW
9153	110262	000000				0			
9154	110264	000360				GRN			: GREEN
9155	110266	000000				0			
9156	110270	007400				BLU			: BLUE
9157	110272	000000				0			
9158	110274	007417				RED+BLU			: VIOLET (MAGENTA)
9159	110276	000000				0			
9160	110300	007777				BLU+RED+GRN			: WHITE.

```

9162          .SBTTL *** ASCII CHARACTER ADDRESS TABLE AND SUBPIX
9163          :
9164          :THE FOLLOWING 128. WORDS CONTAIN THE CHARACTER SUBPIX
9165          :STARTING ADDRESSES. 0-37 AND 140-177 ARE NULL FOR NOW.
9166          :          (EXCEPT FOR 12 AND 15)
9167          :
9168 110302 110702 ACAT:  ..NULL          ;0-37 ARE NULL (NLISTED).
9174          :
9175          110402      $SVPC=      ;SAVE PC, BACK UP AND...
9176          110326      .=ACAT+<12*2>
9177 110326 110704      ..LF          ;...INSERT LINE FEED...
9178          110334      .=ACAT+<15*2>
9179 110334 110714      ..CR          ;...AND CARRIAGE RETURN ADDRESSES.
9180          110402      .=SVPC      ;RESTORE PC
9181          :
9182 110402 110730      ..SPC          ;SPACE
9183 110404 110736      ..EXC          ;EXCLAM
9184 110406 110754      ..QUO          ;DOUBLE QUOTE.
9185 110410 110772      ..NUM          ;NUMBER SIGN
9186 110412 111020      ..DOL          ;DOLLAR
9187 110414 111052      ..PCT          ;PER CENT
9188 110416 111112      ..AND          ;AMPERSAND
9189 110420 111152      ..QUO1         ;SINGLE QUOTE
9190 110422 111164      ..LPAR          ;LFT PAREN
9191 110424 111202      ..RPAR          ;RT PAREN
9192 110426 111220      ..AST          ;ASTERISK
9193 110430 111242      ..PLS          ;PLUS
9194 110432 111260      ..CMA          ;COMMA
9195 110434 111272      ..MNS          ;MINUS
9196 110436 111304      ..DOT          ;PERIOD
9197 110440 111314      ..SLSH         ;SLASH
9198 110442 111330      ..00
9199 110444 111362      ..11
9200 110446 111402      ..22
9201 110450 111434      ..33
9202 110452 111474      ..44
9203 110454 111514      ..55
9204 110456 111544      ..66
9205 110460 111576      ..77
9206 110462 111616      ..88
9207 110464 111666      ..99
9208 110466 111716      ..COLN         ;COLON
9209 110470 111734      ..SEMI         ;SEMI COLON
9210 110472 111752      ..LANG         ;LFT ANGLE
9211 110474 111766      ..EQ          ;EQUALS
9212 110476 112004      ..RANG         ;RT ANGLE
9213 110500 112016      ..QUES         ;QUESTION
9214 110502 112044      ..ATS         ;AT SIGN
9215 110504 112100      ..AA
9216 110506 112124      ..BB
9217 110510 112160      ..CC
9218 110512 112206      ..DD
9219 110514 112230      ..EE
9220 110516 112236      ..FF
9221 110520 112254      ..GG
9222 110522 112272      ..HH
9223 110524 112312      ..II
  
```

9224	110526	112334	..JJ	
9225	110530	112354	..KK	
9226	110532	112372	..LL	
9227	110534	112406	..MM	
9228	110536	112424	..NN	
9229	110540	112444	..OO	
9230	110542	112474	..PP	
9231	110544	112516	..QQ	
9232	110546	112532	..RR	
9233	110550	112546	..SS	
9234	110552	112604	..TT	
9235	110554	112622	..UU	
9236	110556	112644	..VV	
9237	110560	112664	..WW	
9238	110562	112704	..XX	
9239	110564	112730	..YY	
9240	110566	112754	..ZZ	
9241	110570	112776	..LBRK	:LFT BRACKET
9242	110572	113014	..BSLH	:BACK SLASH
9243	110574	113032	..RBRK	:RT BRACKET
9244	110576	113050	..CRT	:CARROT
9245	110600	113064	..USC	:UNDERSCORE
9246	110602	110702	..NULL	:140-177 ARE NULL (NLISTED).

```

9253      ; THESE ARE THE CHAR SUBPIX ROUTINES.
9254      ; EACH IS BUILT ON A 5 X 7 DOT MATRIX (ALA VT05).
9255      ; <CR> WILL INVOKE A "STOP". CPU MUST CALCULATE THE START
9256      ; CO-ORDS FOR NEXT LINE, STUFF IN '..CRX AND ..CRX+2' AND
9257      ; RESUME, TO XCT PSUEDO-CRLF.
9258
9259      ; THE FOLLOWING MACRO (SXY4) APPLIES A 4X SCALE FACTOR AT
9260      ; ASSEMBLY TIME TO YIELD A CHAR SIZE OF 30 X 40 (OCTAL)
9261      ; WHICH PRODUCES A MEDIUM SIZE CHAR FONT.
9262
9263      .MACRO SXY4 A,B,C
9264      SXY  A,B*4,C*4
9265      .ENDM SXY4
9266
9267 110702 165000      ..NULL: DPOP      ;ALL UNDEFINED CHARS EXIT HERE.
9268
9269 110704 164000      ..LF:  DNOP      ;DPOP TO NULL THE <LF>.
9270 110706 104000      SVEC
9272 110712 165000      DPOP
9273
9274 110714 164000      ..CR:  DNOP      ;DPOP TO NULL THE <CRLF>.
9275 110716 173000      STOPN      ;WAIT FOR NEW X AND Y...
9276 110720 114000      APNT
9277 110722 000000 000000 ..CRX: .WORD 0,0      ;...OK, EXECUTE THE <CRLF>
9278 110726 165000      DPOP
9279
9280 110730 104000      ..SPC: SVEC
9282 110734 165000      DPOP
9283 110736 104000      ..EXC: SVEC
9289 110752 165000      DPOP
9290 110754 104000      ..QUO: SVEC
9296 110770 165000      DPOP
9297 110772 104000      ..NUM: SVEC
9307 111016 165000      DPOP
9308 111020 104000      ..DOL: SVEC
9320 111050 165000      DPOP
9321 111052 104000      ..PCT: SVEC
9336 111110 165000      DPOP
9337 111112 104000      ..AND: SVEC
9352 111150 165000      DPOP
9353 111152 104000      ..QUO1: SVEC
9357 111162 165000      DPOP
9358 111164 104000      ..LPAR: SVEC
9364 111200 165000      DPOP
9365 111202 104000      ..RPAR: SVEC
9371 111216 165000      DPOP
9372 111220 104000      ..AST: SVEC
9380 111240 165000      DPOP
9381 111242 104000      ..PLS: SVEC
9387 111256 165000      DPOP
9388 111260 104000      ..CMA: SVEC
9392 111270 165000      DPOP
9393 111272 104000      ..MNS: SVEC
9397 111302 165000      DPOP
9398 111304 104000      ..DOT: SVEC
9401 111312 165000      DPOP
9402 111314 104000      ..SLSH: SVEC
  
```

9407	111326	165000				DPOP
9408	111330	104000		..00:		SVEC
9420	111360	165000				DPOP
9421	111362	104000		..11:		SVEC
9428	111400	165000				DPOP
9429	111402	104000		..22:		SVEC
9441	111432	165000				DPOP
9442	111434	104000		..33:		SVEC
9457	111472	165000				DPOP
9458	111474	104000		..44:		SVEC
9465	111512	165000				DPOP
9466	111514	104000		..55:		SVEC
9477	111542	165000				DPOP
9478	111544	104000		..66:		SVEC
9490	111574	165000				DPOP
9491	111576	104000		..77:		SVEC
9498	111614	165000				DPOP
9499	111616	104000		..88:		SVEC
9518	111664	165000				DPOP
9519	111666	104000		..99:		SVEC
9530	111714	165000				DPOP
9531	111716	104000		..COLN:		SVEC
9537	111732	165000				DPOP
9538	111734	104000		..SEMI:		SVEC
9544	111750	165000				DPOP
9545	111752	104000		..LANG:		SVEC
9550	111764	165000				DPOP
9551	111766	104000		..EQ:		SVEC
9557	112002	165000				DPOP
9558	112004	104000		..RANG:		SVEC
9562	112014	165000				DPOP
9563	112016	104000		..QUES:		SVEC
9573	112042	165000				DPOP
9574	112044	104000		..ATS:		SVEC
9587	112076	165000				DPOP
9588	112100	104000		..AA:		SVEC
9597	112122	165000				DPOP
9598	112124	104000		..BB:		SVEC
9611	112156	165000				DPOP
9612	112160	104000		..CC:		SVEC
9622	112204	165000				DPOP
9623	112206	104000		..DD:		SVEC
9631	112226	165000				DPOP
9632	112230	104000		..EE:		SVEC
9635	112236	104000		..FF:		SVEC
9641	112252	165000				DPOP
9642	112254	104000		..GG:		SVEC
9647	112266	160000	112160			.WORD
9648	112272	104000		..HH:	DJMP,..CC	:FINISH USING 'C'.
9655	112310	165000				DPOP
9656	112312	104000		..II:		SVEC
9664	112332	165000				DPOP
9665	112334	104000		..JJ:		SVEC
9672	112352	165000				DPOP
9673	112354	104000		..KK:		SVEC
9679	112370	165000				DPOP
9680	112372	104000		..LL:		SVEC

9685	112404	165000				DPOP
9686	112406	104000		..MM:		SVEC
9692	112422	165000				DPOP
9693	112424	104000		..NN:		SVEC
9700	112442	165000				DPOP
9701	112444	104000		..OO:		SVEC
9712	112472	165000				DPOP
9713	112474	104000		..PP:		SVEC
9721	112514	165000				DPOP
9722	112516	104000		..QQ:		SVEC
9726	112526	160000	112444		.WORD	DJMP,..OO ;FINISH USING 'O'.
9727	112532	104000		..RR:		SVEC
9731	112542	160000	112474		.WORD	DJMP,..PP ;FINISH USING 'P'.
9732	112546	104000		..SS:		SVEC
9746	112602	165000				DPOP
9747	112604	104000		..TT:		SVEC
9753	112620	165000				DPOP
9754	112622	104000		..UU:		SVEC
9762	112642	165000				DPOP
9763	112644	104000		..VV:		SVEC
9770	112662	165000				DPOP
9771	112664	104000		..WW:		SVEC
9778	112702	165000				DPOP
9779	112704	104000		..XX:		SVEC
9788	112726	165000				DPOP
9789	112730	104000		..YY:		SVEC
9798	112752	165000				DPOP
9799	112754	104000		..ZZ:		SVEC
9807	112774	165000				DPOP
9808	112776	104000		..LBRK:		SVEC
9814	113012	165000				DPOP
9815	113014	104000		..BSLH:		SVEC

9822	113030	165000			DPOP
9823	113032	104000	..RBRK:		SVEC
9829	113046	165000			DPOP
9830	113050	104000	..CRT:		SVEC
9835	113062	165000			DPOP
9836	113064	104000	..USC:		SVEC
9839	113072	165000			DPOP

9841
9842
9843
9844
9845
9846
9847
9848 113074 021557
9849 113076 002165
9850
9851
9852
9853
9854
9855
9856
9857
9858
9859
9860
9861
9862
9863 113100
9864
9865
9866
9868 113400
9870 113404
9871

.SBTTL *** DEVELOPMENT VERSIONS CONTROL

: THE FOLLOWING NUMBERS GET CHANGED AT EACH NEW ASSEMBLY, THEREFORE:
: IF THESE NUMBERS DON'T MATCH WHAT'S IN CORE (OCTALLY) THEN THIS
: LISTING DOESN'T MATCH THE LOADED PROGRAM!

DDMMY: 09071.
HMM: 1141.

.SBTTL *** PATCH AREA
.SBTTL

: FINALLY A GENEROUS PATCH AREA.
: AND AN ADJUSTMENT TO ACCOUNT FOR THE 'LASIAD BIT7' HACK
: DESCRIBED IN 'SUPPRG.MEM' (FOR REV C).
:

PATCH: .BLKW 16.
: .BLKB <<. +400>B^C377>-.
: =
: =. !377+1
L\$LAST::

```

9873          .SBTTL  DEFAULT HARDWARE P-TABLE
9874
9875          :++
9876          : THE DEFAULT HARDWARE P-TABLE CONTAINS DEFAULT VALUES OF
9877          : THE TEST-DEVICE PARAMETERS.  THE STRUCTURE OF THIS TABLE
9878          : IS IDENTICAL TO THE STRUCTURE OF THE RUN-TIME P-TABLE.
9879          :--
9882 113410 172000      .WORD 172000      : 1ST (OF 4) REGISTER(S).
9883                                     : ( 7 IF LUT INSTALLED ).
9884 113412 000320      .WORD 320         : 1ST (OF 4) VECTOR(S).
9885                                     : ( 5 IF LUT INSTALLED ).
9886 113414 000200      .WORD PRI04       : INTERRUPT PRIORITY.
9887 113416 000000      .WORD 0           : LUT AVAILABLE (IF NONZERO)
9888 113420 000000      .WORD 0           : SYNC FREQUENCY: 0 = 60HZ
9889                                     : NZ = 50HZ
9892 113426 172010      .WORD 172010     : 1ST (OF 4) REGISTER(S).
9893                                     : ( 7 IF LUT INSTALLED ).
9894 113430 000340      .WORD 340         : 1ST (OF 4) VECTOR(S).
9895                                     : ( 5 IF LUT INSTALLED ).
9896 113432 000200      .WORD PRI04       : INTERRUPT PRIORITY.
9897 113434 000000      .WORD 0           : LUT AVAILABLE (IF NONZERO)
9898 113436 000000      .WORD 0           : SYNC FREQUENCY: 0 = 60HZ
9899                                     : NZ = 50HZ
9902          .SBTTL  ***** TH - TH - TH - TH - THAT'S ALL FOLKS ! *****
9903 113440 000001      PEND: .END
    
```


SYMBOL TABLE	
DJS	026326
DNOP =	164000 G
DOTS	102556
DPC	003032 G
DPCER	020732 G
DPCERA	012643
DPDSE	007611
DPINIT	025402
DPOP =	165000 G
DPRESE	026074
DPRI	003050 G
DPSINI	025540
DPUMOD	002502 G
DPUSAV	026360
DRDYTO=	150003 G
DRR	003034 G
DSBINT	026176
DSCSE	010111
DSDPE	007651
DSDXE	007711
DSDYE	007751
DSFLE	010051
DSMPE	010211
DSMRE	010151
DSPCE	010011
DSR	003034 G
DSRCHK	022332
DSRER	020764 G
DSRERA	012706
DSTP	026242
DSXPE	010311
DSXRE	010251
DTAB	101236
DTABE	101272
DTHIL =	000015
DTO	026210
DUADR	035704
DUADRE	021454 G
DUAD12	017105
DUFLG	003224 G
DUMMY	003076
DX =	000002 G
DXR	003036 G
DXRER	021016 G
DXRERA	012751
DXYME	100140
DYI =	000002 G
DYNI =	000004 G
DYR	003040 G
DYRER	021050 G
DYRERA	013014
EBLINK	105330
ECE	013552
ECISVC	052416
ECMPE1	052420
ECMPE2	052424
ECMPE3	052436
ECMPE4	052452
ECRSV1	052466
ECRSV2	052472
ECRSV3	052502
ECSEQ1	052512
ECSEQ2	052530
ECSEQ3	052576
ECSEQ4	052630
ECSEQ5	052654
ECSEQ6	052676
ECSEQ7	052726
ECSQ7B	052750
ECSTO	052754
EF.CON=	000036 G
EF.NEW=	000035 G
EF.PWR=	000034 G
EF.RES=	000037 G
EF.STA=	000040 G
EGGBL =	003500 G
EMAXDU	030151
EMK	100360
EN =	006654
ENAINT	026114
ENECHK=	000100 G
ENVIRN	031476
ERCM	024041
ERCNAM	024264
ERCOD	024074 G
ERCODS	047466
ERCTBL	024266
ERICHK	023516
ERR =	100000 G
ERRCHK	023524
ERRCK1	023526
ERRK	030130
ERRVEC=	000004 G
ERTABE	003502
ERTABL	003302
ESF	004612
ESUM	030132
EVL =	000004 G
EX	100336
EXPGOT	017051
EXPGT2	016322
EXPR	027100
EXPRC2	022074 G
EXPREC	020700 G
EXPRM	110242
EXSM2 =	000050 G
EXSM4 =	000060 G
EXTA	022300
EXTEND	022276
EX2 =	000010 G
EX4 =	000020 G
EY	100352
F\$END =	002100
F\$LOAD=	000035
FCOORD	016377
FERCM	024030
FHST =	060000 G
FILL1	027014
FILL2	027020
FLGCHK	022410
FLGER	021134 G
FLGERA	013122
FNOCSR	023772
FNOINT	004315
FORCER	003020
FORCET	032776
FORCSI=	000040 G
FREE	003240 G
FRESIZ	003242 G
FSYCHA	016500
FUMI	004206
FUSI	003752
FUSWI	004251
FUTO	004066
F\$AU =	000015
F\$AUTO=	000020
F\$BGN =	000040
F\$CLEA=	000007
F\$DU =	000016
F\$END =	000041
F\$HARD=	000004
F\$HW =	000013
F\$INIT=	000006
F\$JMP =	000050
F\$MOD =	000000
F\$MSG =	000011
F\$PROT=	000021
F\$PWR =	000017
F\$RPT =	000012
F\$SEG =	000003
F\$SOFT=	000005
F\$SRV =	000010
F\$SUB =	000002
F\$SW =	000014
F\$TEST=	000001
GCOFF =	000100 G
GDBAD	022736
GDDAT	023056 G
GERRMA	002516 G
GETCUR	065474
GETJSE	065542
GHIINH=	010000 G
GHX =	120000 G
GHXF	005314
GHXT	066334
GHXTST	065676
GHXTYP	066332
GHXYX	005374
GHY =	124000 G
GHYF	005344
GHYT	067106
GHYTST	066450
GHYTYP	067104
GOLD =	003200 G
GRN =	000360
GRNC =	000010 G
GRN1 =	002400 G
GRN2 =	003000 G
GRN3 =	003400 G
GRX	075446
GRY	075236
GTCURS	077752
GTJSSW=	040000 G
GUNID	103360
GX	075470
GXI =	174100 G
GY	075260
GYI =	174100 G
G\$CNTO=	000200
G\$DELM=	000372
G\$DISP=	000003
G\$EXCP=	000400
G\$HILI=	000002
G\$LOLI=	000001
G\$NO =	000000
G\$OFFS=	000400
G\$OFFSI=	000376
G\$PRMA=	000001
G\$PRMD=	000002
G\$PRML=	000000
G\$RADA=	000140
G\$RADB=	000000
G\$RADD=	000040
G\$RADL=	000120
G\$RADO=	000020
G\$XFER=	000004
G\$YES =	000010
HAFX =	000776 G
HAFY =	000776 G
HAFY50=	000776 G
HAFY60=	000736 G
HATCH	102702
HBAER	021370 G
HBAERA	013444
HBCHK	022526
HCOPI =	176051 G
HCPY =	000001 G
HGEN	103050
HHMM	113076
HOE =	100000 G
HPM1	002310
HPM2	002340
HPM3	002370
HPM4	002420
HPM5	002450
HST =	040000 G
HSX	075502
HSY	075272
HUE	003222 G
HX	075516
HY	075306
HZ	003246 G
I =	040000 G
IAUX	003160 G
IBE =	010000 G
ICBAE	007476
ICBASE	003162 G
ICSR	003152 G
ICSRE	007156
IDLE	101362
IDLEA	101404
IDLEB	101424
IDLEX	101342
IDPC	003126 G
IDPCE	007100
IDSR	003130 G
IDSRE	007127
IDU =	000040 G
IDXR	003132 G
IDYR	003134 G
IER =	020000 G
IFault	004356
IFLAGS	003150 G
IHBAE	007447
IHBASE	003156 G
IIRCHK	023422
IIRCNC	023406
IIRCNF	023434
IIRNFC	023452
ILDR	003140 G
ILMR	003142 G
ILSR	003136 G
IMAIN	003154 G
IMCBDA	072504
IMCERR	072334 G
IMCHK	071736
IMCHKR	071764
IMCRAM	072506
IMCRCT	072512
IMCTRY	072510
IMC.CH	072552
IMC.IR	072576
IMC.ME	072566
IMC.RA	072606
IMC.SR	072556
IMC.TA	072554
IMC.Y	072574
IMDERR	020506 G
IMDI	005641
IMDIX	005677
IMDIXC	006101
IMDIX2	006010
IMPME	007277
IMPWPE	007234
IMREAD=	162000 G

SYMBOL TABLE	
IMRF	072514
IMRE	007205
IMR8	= 163000
IMTMO	002504 G
INCCBA	044014
INCERK	030216
INCHBA	043544
INCREG	041232
INLAC	072706
INLE	017207
INLEX	022226 G
INLEXF	017235
INTCPC	026112
INTERR	020226 G
INTFLA	026107
INTLAC	003230 G
INTMAS	026106
INTVEC	026110
INTX	004400
IOKCKI	= 000200
IOKERR	= 000004
IOKJSM	= 000002
IOKJSS	= 000010
IOKSTP	= 000001
IPCSAV	003146 G
IREGCK	023066
ISDR	003144 G
ISR	= 000100 G
IXE	= 004000 G
IXPME	007355
IXPWE	007404
IXRRE	007326
ISAU	= 000041
ISAUTO	= 000041
ISCLN	= 000041
ISDU	= 000041
ISHRD	= 000041
ISINIT	= 000041
ISMOD	= 000041
ISMSG	= 000041
ISPROT	= 000040
ISPTAB	= 000041
ISPR	= 000041
ISRPT	= 000041
ISSEG	= 000041
ISSETU	= 000041
ISSFT	= 000041
ISSRV	= 000041
ISSUB	= 000041
ISTST	= 000041
JCWD	= 000020 G
JCWE	= 000040 G
JCXA	100320
JCXB	100740
JCXC	101110
JCXD	101114
JENOCL	014376
JMSPOP	044222
JMWD	= 000004 G
JMWE	= 000010 G
JPDPC	007525
JPPCSE	007557
JSDXE	100064
JSEWRO	014211
JSEWRZ	014145
JSF	004566
JSLCKO	= 000010 G
JSMIES	= 010000 G
JSMV	003054 G
JSPIX	100316
JSPIXA	100306
JSSIES	= 004000 G
JSSV	003060 G
JSTST	077304
JSV	076670
JVA	076744
JVA1	076724
JSWD	= 000001 G
JSWE	= 000002 G
JSWTEX	100010
JSX	100414
JSY	100434
JSO	= 000000 G
JS1	= 000400 G
JS2	= 001000 G
JS3	= 001400 G
JXYPE	100254 G
JXYPX	100216
JSJMP	= 000167
KIPAR0	= 172340
KIPAR1	= 172342
KIPAR2	= 172344
KIPAR3	= 172346
KIPAR4	= 172350
KIPAR5	= 172352
KIPAR6	= 172354
KIPAR7	= 172356
KIPDR0	= 172300
KIPDR1	= 172302
KIPDR2	= 172304
KIPDR3	= 172306
KIPDR4	= 172310
KIPDR5	= 172312
KIPDR6	= 172314
KIPDR7	= 172316
KLUDGE	031572
KTBKGD	031322
KTFLG	003244 G
KTINIT	031656
KTOFF	031306
KTON	031270
K100US	027564
LAIER	006555
LAIEW	006617
LDCP	= 175000 G
LDECC	= 175400 G
LDI	= 002000 G
LDINC	006662
LDJSS	= 146000 G
LDR	003044 G
LDUN	026602
LERRMA	002514 G
LINTER	020610 G
LINTX	006540
LMR	003046 G
LOCT	074754
LOE	= 040000 G
LOGO	074600
LONGX	005121
LOOP	027646
LOOPFL	023054 G
LOOPK	027700
LOT	= 000010 G
LREAD	= 010000 G
LRWERR	020642 G
LRWX	006707
LSET	= 010000 G
LSR	003042 G
LUMIN	027054
LUMTBL	107222
LUTAV	003216 G
LUTSAV	026606
LUTV	003062 G
LV	055606
LVEC	= 110000 G
LVF	005073
LVTST	055224
LVX	055616
LVY	055620
LSACP	002110 G
LSAPT	002036 G
LSAU	033012 G
LSAUT	002070 G
LSAUTO	033216 G
LSCCP	002106 G
LSCLEA	033220 G
LSCO	002032 G
LSDEPO	002011 G
LSDESC	002122 G
LSDESP	002076 G
LSDEVP	002060 G
LSDISP	107114 G
LSDLY	002116 G
LSDTP	002040 G
LSDTYP	002034 G
LSDU	033110 G
LSDUT	002072 G
LSDVTY	002162 G
LSEF	002052 G
LSENV1	002044 G
LSETP	002102 G
LSEXP1	002046 G
LSEXP4	002064 G
LSEXP5	002066 G
LSHARD	002240 G
LSHIME	002120 G
LSHPCP	002016 G
LSHPTP	002022 G
LSHW	002224 G
LSICP	002104 G
LSINIT	032014 G
LSLADP	002026 G
LSLAST	113404 G
LSLOAD	002100 G
LSLUN	002074 G
LSMREV	002050 G
LSNAME	002000 G
LSPRIO	002042 G
LSPROT	032004 G
LSPT	002112 G
LSREPP	002062 G
LSREV	002010 G
LSRPT	033272 G
LSOFT	002522 G
LSPC	002056 G
LSPCP	002020 G
LSPTP	002024 G
LSSTA	002030 G
LSW	002502 G
LSWTEST	002114 G
LSSTIML	002014 G
LSUNIT	002012 G
LO	= 002000 G
L10000	002236
L10001	002500
L10002	002520
L10003	003020
L10004	020062
L10005	020124
L10006	020166
L10007	020224
L10010	020504
L10011	020606
L10012	020640
L10013	020676
L10014	020730
L10015	020762
L10016	021014
L10017	021046
L10020	021100
L10021	021132
L10022	021164
L10023	021216
L10024	021250
L10025	021302
L10026	021334
L10027	021366
L10030	021420
L10031	021452
L10032	021506
L10033	021646
L10034	021760
L10035	022072
L10036	022130
L10037	022176
L10040	022224
L10041	022274
L10042	023322
L10043	024262
L10044	025740
L10046	032724
L10047	033106
L10050	033214
L10051	033216
L10052	033270
L10053	033532
L10054	034272
L10055	035042
L10056	035620
L10057	041150
L10060	042136
L10061	042564
L10062	043446
L10063	043716
L10064	044156
L10065	044652
L10066	045366
L10067	046036
L10070	046766
L10071	047334
L10072	052774
L10073	053326
L10074	054514
L10075	055154
L10076	055624
L10077	056314
L10100	057014
L10101	060062
L10102	065610
L10103	066362
L10104	067134
L10105	067562
L10106	070622
L10107	072612
L10110	071270
L10111	071550
L10112	072502
L10113	073146
L10114	074362
L10115	073660
L10116	074236
L10117	074444
L10120	076556
L10121	101132
L10122	100304

SYMBOL TABLE

L10123	104760	MPT.ST	074334	OPTI	003064	G	PTINER	023324	SCBASE	003212	G
L10124	107110	MPT.WB	074356	OUTDFX	105246		PUNIT	032730	SCFLG	003214	G
L10125	113410	MPT.WE	074344	OUTDFY	105276		PVEC2 =	000004	SCIDE	017355	
L10126	113426	MPT.WY	074352	OUTDOT	105072		QVP	003030	SCINT	106640	
L10127	113422	MPXPE	011551	OUTDTF	105134		QXIT	076460	SCM	017534	
L10131	113440	MPXRE	011511	OUTDT1	105162		Q1	076366	SCME	017416	
L255 =	003774	MRCSE	011051	OUTDY1	105302		Q2	076424	SCMEM	106464	
M =	020000	MRDSE	010711	OUTFLG	105070		Q3	076366	SCMFG	017461	
MAGEN =	002300	MRFLE	011011	OUTLIN	104762		Q4	076424	SCMMMO	106704	
MAPMEM	070326	MRMPE	011111	OSAPTS=	000000		RBL	076536	SCMMSO	106772	
MARK	100442	MRPCE	010751	OSAU =	000001		RBO	076530	SCNINT	106660	
MATCH	100470	MRRER	021220	OSBGNR=	000001		RBX	076540	SCNSTD	106552	
MATFLE	016017	MRRERA	013230	OSBGNS=	000001		RBY	076542	SCONTY	105522	
MATIRE	015663	MRXPE	011211	OSDU =	000001		RDDXY	065520	SCOUT	107040	
MATPCS	015505	MRXRE	011151	OSERRT=	000000		RDJSE	065566	SCPRE	017326	
MATPOS	015611	MSX =	020000	OSGNSW=	000001		RDWRT =	176074	SCRBLU	104542	
MATUI	016064	MSY =	000100	OSPOIN=	000001		RDWRT1=	176076	SCRCOM	104554	
MATXYE	015547	MXY =	020000	OSSETU=	000000		READ =	176050	SCRGRN	104550	
MAXX =	001776	M128 =	000002	PASRPT	032240		RED =	000017	SCRRED	104534	
MAXY =	001776	M256 =	000003	PATCH	113100		REDC =	000020	SCRWHT	104526	
MAXY50=	001776	M32 =	000000	PATS	074250		REDG =	002200	SCS	017632	
MAXY60=	001676	M64 =	000001	PATTST	073240		REGERR	023144	SCSR	003202	G
MCHAN	077270	NBLINK=	175400	PATT1	073262		REGNAM	023062	SCSYC	106571	
MDVO =	000000	NEWPAS	032174	PAUSE1	027514		RELEAS	026756	SDASK	102212	
MDV1 =	000001	NLRI	006306	PAUS.5	027522		RELF	004746	SDLIST	101472	
MDV2 =	000002	NLWI	006362	PCSCHK	022360		RELFX	004773	SDO	102231	
MDV3 =	000003	NOCMI	015263	PCSER	021102	G	RELTST	053050	SDPC	003164	G
MEMFLG	003252	NOCSR	024003	PCSERA	013057		RELT1	053322	SDSR	003166	G
MEMTAB	003256	NODEV	003226	PCSERR	020064	G	RESET1	034340	SDX	027312	
MENDPC	016141	NOINIT	004434	PCSX	003564		RESTRS	031436	SDXR	003170	G
MEREG	015733	NOINTR	004321	PDBITS	003066	G	RGE	022664	SDY	027270	
MFGFLG	002512	NOITS	002510	PDF	003072	G	RJF	004534	SDYR	003172	G
MFGMO	003022	NOMAN	017735	PDI	003070	G	RLMOD	057062	SELCHA=	000007	G
MFGSO	003024	NOMAT	016267	PDM	003074	G	RLMOD1	057076	SELCHA	077302	
MKDXR	100004	NOMST	015334	PDXI	004642		RLMXIT	057676	SELCSR=	000003	G
MKDYR	100006	NOSTIK	077222	PEND	113440		RLNEWL	057346	SELDIS	101272	
MMCHK	022466	NOSWI	014777	PERIM	103066		RLSKIP=	000400	SELSR=	000000	G
MMMAC	044744	NOSWPI	014712	PINTRA=	000001	G	RLXIE	016552	SELFLG=	000002	G
MMACA	045016	NOTOT	047336	PINTRB=	000002	G	RLXYE	020170	SELHBA=	000005	G
MMACE	012511	NSI	004014	PIXRBK=	000000	G	RLX2IE	016613	SELMPP=	000004	G
MMAE	021650	NSINIT	006760	PNT =	001000	G	RLYDE	016715	SELMRR=	000004	G
MMINXE	012600	NTO	004135	PNTCOO	022132	G	RLYUE	016663	SELPCS=	000001	G
MMVEC =	000250	NTSC	027066	PNTSYC	022200	G	RLY2DE	017010	SELXPM=	000006	G
MMOKT	045316	NTSCB	110222	PRBH	005574		RLY2UE	016751	SELXRR=	000006	G
MPAT2	074306	NUL	016546	PRBHNG	005556		RND8 =	000400	SEQERR=	120003	G
MPCBAS	074332	NULCR	016547	PRGSIZ=	174561		RNLN =	144000	SETCB =	152000	G
MPCSE	011411	NXM	027566	PRI =	002000	G	RPD	056310	SETCSR=	000013	G
MPDSE	011251	NXME =	104003	PRI00 =	000000	G	RPF	005175	SETDSR=	000010	G
MPE =	114003	NXR	003504	PRI01 =	000040	G	RPNT =	130000	SETFLG=	000012	G
MPFLE	011351	NXRERR	020032	PRI02 =	000100	G	RPT	056300	SETHB =	150000	G
MPLN =	000011	NXRX	003543	PRI03 =	000140	G	RPTST	055676	SETMEM=	170100	G
MPMER	021252	NXTU	032206	PRI04 =	000200	G	RSTPOS=	100000	SETMPM=	000015	G
MPMERA	013273	ODDSKP=	000200	PRI05 =	000240	G	RSVDOP=	124003	SETMRR=	000014	G
MPMRE	011451	OPCF	004676	PRI06 =	000300	G	R256 =	000100	SETU	032272	
MPOFF	074330	OPCFX	004724	PRI07 =	000340	G	SAUX	003210	SETXPM=	000017	G
MPPCE	011311	OPCOD	042632	PROTEC=	176000	G	SAVED	031462	SETXRR=	000016	G
MPTNEW	073662	OPC1	043440	PRSYCH	065152		SAVERS	031412	SFLAGS	003200	G

SYMBOL TABLE

SFPTBL	002502	G	SWCHON=	000002	G	TSSAU =	010047	UAM	=	000200	G	XPXRE	012451	
SHADLY	003220	G	SWE =	000010	G	TSSAUT=	010051	ULRI	006437			XRCSE	011751	
SHAD16	027172		SWIBV	015036		TSSCLE=	010052	ULWI	006477			XRDSE	011611	
SHBASE	003206	G	SWNOCU	015205		TSSDAT=	010131	UMI	004212			XRFLE	011711	
SHRTX	005250		SWNOHL	015112		TSSDU =	010050	UNITN	003026	G		XRMPE	012051	
SIFLAG	023064		SWTCH =	170200	G	TSSHAR=	010001	USI	003756			XRMRE	012011	
SIMSG	023012		SYCFLG	003254	G	TSSHW =	010000	USWI	004255			XRPCSE	011651	
SINIT =	100000	G	SYCTAB	003270	G	TSSINI=	010046	US100	027530			XRRER	021304	G
SINIT1	035114		SYNC =	164001	G	TSSMSG=	010122	UTO	004072			XRRERA	013336	G
SIOFF =	171000	G	SYCTO=	140003	G	TSSPC =	000002	VIOLET=	003300	G		XXRPE	012111	
SION =	171400	G	SYSCON	021510	G	TSSPRO=	010045	VISXIT	076356			XTBL	102516	
SKPO =	000400		SLSYM=	010000		TSSPTA=	010130	VIS1	074544			XXCOMM	003236	G
SKP1 =	000401		TBOFF	102320		TSSRPT=	010053	VIS2	074714			XXDP =	003726	
SKP2 =	000402		TBON	102262		TSSSEG=	010000	VIS3	075110			XX.ENT=	025570	
SKP3 =	000403		TEMP1	003232	G	TSSSOF=	010003	VIS4	075320			XX.EXI=	026072	
SMAIN	003204	G	TEMP2	003234	G	TSSSUB=	010116	VIS5	075530			XYERR	020126	G
SOCT1	075026		TESTK	030070		TSSSW =	010002	VISSA	075746			XYX	003657	
SOCT2	075040		TIDFLG	002506	G	TSTES=	010124	VIS6	076076			XSALWA=	000000	
SPARTS	074434		TINERR	022775		T1	034054	VRLF	076016			XSALS=	000040	
SPCSAV	003176	G	TN =	000043		T10	044160	VRLF1	076024			X\$OFFS=	000400	
SPM1	002600		TNAM	030066		T11	044654	VRLF2	076026			X\$TRUE=	000020	
SPM2	002630		TNUM	030064		T12	045370	VSHGE	026010			X2INC	057752	
SPM4	002660		TOTV	003056	G	T13	046040	VSHNG	025720	G		X2INCA	057762	
SPM5	002710		TOTVST	047050		T14	046770	VSHUNG	025742			X2INCB	057772	
SPM6	002740		TSTEND	030072		T15	047422	WAITF	027454			YA	067124	
SPM7	002770		TSTGO	027702		T16	052776	WE =	000010	G		YDN	060016	
SRO =	177572		T\$ARGC=	000001		T17	053330	WECC =	160000	G		YDOWN =	000040	G
SR1 =	177574		T\$CODE=	000052		T18	054516	WHITE =	003774	G		YELLOW=	003600	G
SR2 =	177576		T\$ERRN=	006346		T19	055156	WJSS =	042000	G		YI	066340	
SR3 =	172516		T\$EXCP=	000000		T2	034274	WRT =	176034	G		YMAX	003250	G
SSDSR	003174	G	T\$FLAG=	000040		T20	055626	WRTCPX=	140000	G		YUP	060002	
SSF	004503		T\$FREE=	113440		T21	056316	WRTCPY=	140000	G		Y2DN	060046	
SSTST	042210		T\$GMAN=	000000		T22	057016	WRTCSR=	000013	G		Y2UP	060032	
SST1	042516		T\$HILI=	000015		T23	060064	WRTJSS=	042000	G		Y50TBL	102516	
SST2	042526		T\$LAST=	000001		T24	065612	WRTMSR=	040000	G		Y60TBL	102536	
SST3	042550		T\$LOLI=	000000		T25	066364	WRT.D	074262			\$SVPC =	110402	
STOP =	172000	G	T\$LSYM=	010000		T26	067136	WRT.X	074264			.DX =	002000	
STOP1 =	173400	G	T\$LTNO=	000043		T27	067564	WRT.Y	074266			.DY =	000000	
STOPN =	173000	G	T\$NEST=	177777		T28	070624	WRT1 =	176036	G		.I =	000000	
STPTST	046110		T\$NSO =	000001		T28.1	070732	XA	066352			.NT =	000037	
STPV	003052	G	T\$NS1 =	000011		T28.2	071272	XI	067112			..AA	112100	
STP1	046742		T\$NS2 =	000003		T29	072614	XINC	057722			..AND	111112	
STP2	046746		T\$PCNT=	000000		T3	035044	XINCA	057732			..AST	111220	
STP3	046752		T\$PTAB=	010130		T30	073150	XINCB	057742			..ATS	112044	
STP4	046760		T\$PTHV=	000002		T30.1	073254	XMCHK	022566			..BB	112124	
STRSET	034122		T\$PTNU=	000002		T30.2	073662	XMMAC	045456			..BSLH	113014	
SUPVSR=	026050		T\$SAVL=	177777		T31	074364	XMMACA	045530			..CC	112160	
SVCGBL=	000000		T\$SEGL=	177777		T32	074446	XMMACE	012545			..CMA	111260	
SVCINS=	177777		T\$SEKO=	010000		T33	076560	XMMAE	021762	G		..COLN	111716	
SVCSUB=	177777		T\$SIZE=	000016		T34	101134	XPCSE	012311			..CR	110714	
SVCTAG=	177777		T\$SUBN=	000000		T35	105404	XPDSSE	012151			..CRT	113050	
SVCTST=	177777		T\$TAGL=	177777		T4	035622	XPFLSE	012251			..CRX	110722	
SVD	057010		T\$TAGN=	010132		T5	041152	XPMER	021336	G		..DD	112206	
SVEC =	104000	G	T\$TEMP=	000044		T6	042140	XPMERA	013401			..DOL	111020	
SVF	005221		T\$TEST=	000043		T7	042566	XPMPE	012411			..DOT	111304	
SVT	057000		T\$TSTM=	177777		T8	043450	XPMRE	012351			..EE	112230	
SVTST	056364		T\$TSTS=	000001		T9	043720	XPPCE	012211			..EQ	111766	

SYMBOL TABLE

..EXC	110736	..LPAR	111164	..QQ	112516	..SPC	110730	..00	111330
..FF	112236	..MM	112406	..QUES	112016	..SS	112546	..11	111362
..GG	112254	..MNS	111272	..QUO	110754	..TT	112604	..22	111402
..HH	112272	..NN	112424	..QUO1	111152	..USC	113064	..33	111434
..II	112312	..NULL	110702	..RANG	112004	..UU	112622	..44	111474
..JJ	112334	..NUM	110772	..RBRK	113032	..VV	112644	..55	111514
..KK	112354	..OO	112444	..RPAR	111202	..WW	112664	..66	111544
..LANG	111752	..PCT	111052	..RR	112532	..XX	112704	..77	111576
..LBRK	112776	..PLS	111242	..SEMI	111734	..YY	112730	..88	111616
..LF	110704	..PP	112474	..SLSH	111314	..ZZ	112754	..99	111666
..LL	112372								

. ABS. 113440 000
000000 001

ERRORS DETECTED: 0

VIRTUAL MEMORY USED: 34896 WORDS (137 PAGES)

DYNAMIC MEMORY: 20346 WORDS (78 PAGES)

ELAPSED TIME: 00:36:46

CVVSA.BIN,CVVSA.LST/-SP=SVC33.MLB/ML,CVVSAB0.MAC/EN:AMA:ABS/DS:GBL