

VSV11

VSV11 CSS DIAG
CVVSA00

AH-E769A-MC
FICHE 1 OF 2

AUG 1981
COPYRIGHT © 1981
MADE IN USA



VSV11

VSV11 CSS DIAG
CVVSAAO

AH-E769A-MC
FICHE 2 OF 2

AUG 1981
COPYRIGHT © 1981
MADE IN USA



.REM%

IDENTIFICATION

PRODUCT CODE: AC-E768A-MC
 PRODUCT NAME: CVVSAA0 VSV11 CSS DIAG
 MAINTAINER: CSS PRODUCTS
 AUTHORS: DON MACOMBER
 GUS PASQUANTONIO
 BILL WEISKE
 DATE: 23 FEB 81

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

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

COPYRIGHT (C) 1981 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL	PDP	UNIBUS	MASSBUS
DEC	DECUS	DECTAPE	

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

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

106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161

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.

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(CEED)	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/SWI*CH: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
IEK	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)

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.

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.
PROCEED TO RESUME FROM THE POINT OF THE ERROR CALL.
CONTINUE TO RESTART AT THE BEGINING OF THE TEST IN WHICH THE ERROR OCCURED.

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

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

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)

415
416
417
418
419
420
421
422
423
424
425
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

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.

469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523

6.1.6 TEST 6. DPU START-STOP

TEST 1 VERIFIES THAT THE DISPLAY PROCESSOR (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 INSTRUCTION, 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.

525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572

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 INTERRUPT

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.

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 PSUEDO-CHARACTER AND SETS A UNIQUE BASE ADDRESS FOR EACH CHARACTER CODE SUCH THAT: $\text{BASE} + 2(\text{CODE}) = \text{SUBROUTINE ADDRESS}$. 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 $X, Y = \text{MAX}, \text{MAX}$. THE X AND Y POSITION REGISTERS ARE TESTED AS EACH VECTOR IS DRAWN.

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

632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685

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.

687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734

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.

736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784

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

786
787
788
789
790
791
792
793
794
795
796
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

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 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.

833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
889
890

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

%

```

892      .SBTTL  PROGRAM HEADER
893
894      .MCALL  SVC
895 000000 SVC          ; INITIALIZE SUPERVISOR MACROS
896
897      ;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
898      ; IF STRUCTURED MACROS ARE TO BE USED, YOU MUST ADD
899      .REQ    SPMAC.SML (OR THE CURRENT EQUIVALENT)
900      .MCALL  STRUCT
901      STRUCT
902
903      SVCGBL= 1          ; LIST GLOBAL TAGS AT RIGHT MARGIN.
904      SVCTST= 1          ;      DITTO  TEST TAGS.
905      SVCSUB= 1          ;      DITTO  SUBTEST TAGS.
906      SVCTAG= 1          ;      DITTO  ANY OTHER TAGS.
907      SVCINS= 1          ;      DITTO  INSTRUCTIONS AND DATA.
908
909      ; THESE SYMBOLS CONTROL THE LISTING FIELD OF ALL SVC MACRO
910      ; EXPANSIONS.  YOU MAY CHANGE THEM AT ANY TIME OR PLACE.
911
912      ;      1 = RIGHT-JUSTIFY (MAKES IT EASY TO DISTINGUISH
913      ;          SVC'S MACRO CODE FROM YOUR OWN).
914      ;      0 = LEFT-JUSTIFY (ALIGN IN A NORMAL FASHION).
915      ;     -1 = DON'T LIST THE EXPANSIONS AT ALL.
916      ;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
917
918      .ENABL  ABS,AMA
919      .-      2000
920
921      ;++
922      ; THE PROGRAM HEADER IS THE INTERFACE BETWEEN
923      ; THE DIAGNOSTIC PROGRAM AND THE SUPERVISOR.
924      ;--
925
926      PRGSIZ= <L$LAST>/16.      ; PROGRAM SIZE IN 1/8 K UNITS (OCTAL).
927      SVCGBL= 0                  ; ALIGN THE HEADER STUFF.
928      SVCINS= 0
929
930      POINTER BGNSW,BGNSFT,BGNAU,BGNDU,BGNRPT
931      HEADER  CVVSA,A,0,655..0
932
933      L$NAME::                   ;DIAGNOSTIC NAME
934      .ASCII  /C/
935      .ASCII  /V/
936      .ASCII  /V/
937      .ASCII  /S/
938      .ASCII  /A/
939      .BYTE   0
940      .BYTE   0
941      .BYTE   0
942
943      L$REV::                     ;REVISION LEVEL
944      .ASCII  /A/
945
946      L$DEPO::                   ;0
947      .ASCII  /0/
948
949      L$UNIT::                   ;NUMBER OF UNITS
950      .WORD   0
951
952      L$TIML::                   ;LONGEST TEST TIME
953      .WORD   655.
954
955      L$HP(P::                   ;POINTER TO H.W. QUES.

```

002016	002240	L\$SPCP::	.WORD	L\$HARD	
002020					; POINTER TO S.W. QUES.
002020	002522	L\$HPTP::	.WORD	L\$SOFT	
002022					; PTR. TO DEF. H.W. PTABLE
002022	002224	L\$SPTP::	.WORD	L\$HW	
002024					; PTR. TO S.W. PTABLE
002024	002502	L\$ADP::	.WORD	L\$SW	
002026					; DIAG. END ADDRESS
002026	113404	L\$STA::	.WORD	L\$LAST	
002030					; RESERVED FOR APT STATS
002030	000000	L\$CO::	.WORD	0	
002032					
002032	000000	L\$DTP::	.WORD	0	
002034					; DIAGNOSTIC TYPE
002034	000000	L\$APT::	.WORD	0	
002036					; APT EXPANSION
002036	000000	L\$DTP::	.WORD	0	
002040					; PTR. TO DISPATCH TABLE
002040	106760	L\$DISPATCH	.WORD		
002042					; DIAGNOSTIC RUN PRIORITY
002042	000000	L\$ENVI::	.WORD	0	
002044					; FLAGS DESCRIBE HOW IT WAS SETUP
002044	000000	L\$EXP1::	.WORD	0	
002046					; EXPANSION WORD
002046	000000	L\$MREV::	.WORD	0	
002050					; SVC REV AND EDIT #
002050	003		.BYTE	C\$REVISION	
002051	003		.BYTE	C\$EDIT	
002052		L\$EF::			; DIAG. EVENT FLAGS
002052	000000		.WORD	0	
002054	000000		.WORD	0	
002056		L\$SPC::	.WORD	0	
002056	000000				
002060		L\$DEVP::	.WORD	0	; POINTER TO DEVICE TYPE LIST
002060	002162	L\$REPP::	.WORD	L\$DVTYP	
002062					; PTR. TO REPORT CODE
002062	033272	L\$EXP4::	.WORD	L\$RPT	
002064					
002064	000000	L\$EXP5::	.WORD	0	
002066					
002066	000000	L\$AUT::	.WORD	0	; PTR. TO ADD UNIT CODE
002070					
002070	033012	L\$DUT::	.WORD	L\$AU	
002072					; PTR. TO DROP UNIT CODE
002072	033110	L\$LUN::	.WORD	L\$DU	
002074					; LUN FOR EXERCISERS TO FILL
002074	000000	L\$DESP::	.WORD	0	
002076					; POINTER TO DIAG. DESCRIPTION
002076	002122	L\$LOAD::	.WORD	L\$DESC	
002100					; GENERATE SPECIAL AUTOLOAD EMT
002100	104035	L\$ETP::	EMT	E\$LOAD	
002102					; POINTER TO ERRIBL
002102	000000	L\$ICP::	.WORD	0	
002104					; PTR. TO INIT CODE
002104	032014	L\$CCP::	.WORD	L\$INIT	
002106					; PTR. TO CLEAN-UP CODE
002106	033220		.WORD	L\$CLEAN	

002110				L\$ACP::				;PTR. TO AUTO CODE
002110	033216			L\$PRT::	.WORD	L\$AUTO		;PTR. TO PROTECT TABLE
002112				L\$TEST::	.WORD	L\$PROT		;TEST NUMBER
002112	032004							
002114					.WORD	0		
002114	000000			L\$DLY::				;DELAY COUNT
002116					.WORD	0		
002116	000000			L\$HIME::				;PTR. TO HIGH MEM
002120					.WORD	0		
002120	000000							
931 002122				DESCRIPT	<***** VSV11-VS11 DIAGNOSTIC *****>			
002122				L\$DESC::	.ASCIZ	/***** VSV11-VS11 DIAGNOSTIC *****/		
002122	052	052	052		.EVEN			
				DEVTYPE	<VSV11-VS11, AND DISPLAY MONITOR>			
932 002162				L\$DVTYP::	.ASCIZ	/VSV11-VS11, AND DISPLAY MONITOR/		
002162					.EVEN			
002162	126	123	126					
933								
934	000001			SVCGBL=	1	; SHOVE EVERYTHING BACK TO THE RIGHT.		
935	000001			SVCINS=	1			
936	177777			SVCGBL=-	1	; KILL ALL SVC STUFF.		
937	177777			SVCTST=-	1			
938	177777			SVCSUB=-	1			
939	177777			SVCTAG=-	1			
940	177777			SVCINS=-	1			

942
943
944
945
946
947
948
949 002222
950
951 002224 172010
952
953 002226 000320
954
955 002230 000200
956 002232 000000
957 002234 000000
958
959 002236

.SBTTL DEFAULT HARDWARE P-TABLE

;++
: THE DEFAULT HARDWARE P-TABLE CONTAINS DEFAULT VALUES OF
: THE TEST-DEVICE PARAMETERS. THE STRUCTURE OF THIS TABLE
: IS IDENTICAL TO THE STRUCTURE OF THE RUN-TIME P-TABLE.
:--

BGNHW DFPTBL

:DEFAULT HARD-P-TABLE

.WORD 172010

: 1ST (OF 4) REGISTER(S).
: (7 IF LUT INSTALLED).

.WORD 320

: 1ST (OF 4) VECTOR(S).
: (5 IF LUT INSTALLED).

.WORD PRI04

: INTERRUPT PRIORITY.

.WORD 0

: LUT AVAILABLE (IF NONZERO)

.WORD 0

: SYNC FREQUENCY: 0 60HZ

NZ = 50HZ

ENDHW

```

961      .SBTTL  HARDWARE PARAMETER CODING SECTION
962
963      : **
964      : THE HARDWARE PARAMETER CODING SECTION CONTAINS MACROS
965      : THAT ARE USED BY THE SUPERVISOR TO BUILD P-TABLES.  THE
966      : MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE
967      : INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES.  THE
968      : MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS
969      : WITH THE OPERATOR.
970      : --
971      002236      BGNHRD
972
973      002240      GPRMA  HPM1,0,0,160000,177776,YES ;GET 1ST REGISTER ADDRESS.
974      002250      GPRMA  HPM2,2,0,0,776,YES      ;GET 1ST VECTOR ADDRESS.
975      002260      GPRMD  HPM3,4,0,340,0,7,YES     ;GET INTERRUPT PRIORITY.
976      002272      GPRML  HPM4,6,-1,YES           ;ASK IF LUT AVAILABLE
977      002300      GPRML  HPM5,10,-1,YES          ;ASK IF FREQUENCY = 50 HZ.
978      002306      EXIT HRD
979      002310      104    105    126  HPM1:  .ASCIZ  /DEVICE ADDRESS      /
980      002340      061    123    124  HPM2:  .ASCIZ  /1ST INTERRUPT VECTOR /
981      002370      111    116    124  HPM3:  .ASCIZ  /INTERRUPT PRIORITY  /
982      002420      114    125    124  HPM4:  .ASCIZ  /LUT INSTALLED      /
983      002450      106    122    105  HPM5:  .ASCIZ  /FREQUENCY = 50HZ     /
984      002500      .EVEN
985      002500      ENDRD

```

987			.SBTTL SOFTWARE P-TABLE	
988				
989			;++	
990			: THE SOFTWARE P-TABLE CONTAINS THE VALUES OF THE PROGRAM	
991			: PARAMETERS THAT CAN BE CHANGED BY THE OPERATOR.	
992			:--	
993	002500		BGNSW SFPTBL	
994	002502	000000	DPUMOD:: .WORD 0	; DPU/MEM TEST MODE...
995				... 0 = RUN ALL...
996				...NZ = RUN DPU ONLY.
997	002504	000000	IMTMOD:: .WORD 0	; IMAGE MEMORY TEST MODE...
998				... 0 = SHORT TEST...
999				...NZ = LONG TEST.
1000	002506	000000	TIDFLG:: .WORD 0	; TEST ID'S...
1001				... 0 = TYPE TEST ID AT EACH TTET START...
1002				...NZ = DON'T.
1003	002510	000000	NOITS:: .WORD 0	; INHIBIT ITERATION OPTION.
1004				... 0 = ITERATE.
1005				...NZ = INHIBIT ITERATE.
1006	002512	000000	MFGFLG:: .WORD 0	; MANUFACTURING TEST MODE...
1007				... 0 = USER/FIELD SVC...
1008				...NZ = MANUFACTURING.
1009	002514	000031	LERRMAX:: .WORD 25.	; LOCAL (PER TEST) ERROR LIMIT
1010	002516	000310	GERRMAX:: .WORD 200.	; GLOBAL (PER UNIT) ERROR LIMIT
1011	002520		ENDSW	
1012				
1013				

```

1015          .SBTTL  SOFTWARE PARAMETER CODING SECTION
1016
1017          ;++
1018          ; THE SOFTWARE PARAMETER CODING SECTION CONTAINS MACROS
1019          ; THAT ARE USED BY THE SUPERVISOR TO BUILD P-TABLES.  THE
1020          ; MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE
1021          ; INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES.  THE
1022          ; MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS
1023          ; WITH THE OPERATOR.
1024          ;--
1025          BGNSFT
1026          GPRML  SPM1,0,-1,YES          ; GET OPTIONAL DPU ONLY FLAG.
1027          GPRML  SPM2,2,-1,YES          ; GET MEM TEST MODE.
1028          ;GPRML  SPM3,4,-1,YES          ; GET TEST ID CONTROL.
1029          GPRML  SPM4,6,-1,YES          ; GET ITERATION CONTROL.
1030          GPRML  SPM5,10,-1,YES         ; GET MANUFACTURING MODE.
1031          GPRMD  SPM6,12,D,7777,0,7777,YES ; GET LOCAL ERROR LIMIT
1032          GPRMD  SPM7,14,D,7777,0,7777,YES ; GET GLOBAL ERROR LIMIT
1033          EXIT SFT
1034          122    125    116  SPM1:  .ASCIIZ /RUN DPL TESTS ONLY /
1035          114    117    116  SPM2:  .ASCIIZ /LONG MEMORY TESTS /
1036          ;SPM3:  .ASCIIZ /INHIBIT TEST ID'S /
1037          111    116    110  SPM4:  .ASCIIZ /INHIBIT ITERATIONS /
1038          115    101    116  SPM5:  .ASCIIZ /MANUFACTURING MODE /
1039          120    105    122  SPM6:  .ASCIIZ /PER TEST ERROR LIMIT /
1040          120    105    122  SPM7:  .ASCIIZ /PER UNIT ERROR LIMIT /
1041          ENDSFT          ; UNUSED.
  
```

1043
1044
1045
1046
1047
1048
1049
1050 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

1051

1053		.SBTTL MEMORY MANAGEMENT DEFINITIONS
1054		
1055		;*KT11 VECTOR ADDRESS
1056		
1057	000250	MMVEC= 250
1058		
1059		;*KT11 STATUS REGISTER ADDRESSES
1060		
1061	177572	SR0= 177572
1062	177574	SR1= 177574
1063	177576	SR2= 177576
1064	172516	SR3= 172516
1065		
1066		;*KERNEL 'I' PAGE DESCRIPTOR REGISTERS
1067		
1068	172300	KIPDR0= 172300
1069	172302	KIPDR1= 172302
1070	172304	KIPDR2= 172304
1071	172306	KIPDR3= 172306
1072	172310	KIPDR4= 172310
1073	172312	KIPDR5= 172312
1074	172314	KIPDR6= 172314
1075	172316	KIPDR7= 172316
1076		
1077		;*KERNEL 'I' PAGE ADDRESS REGISTERS
1078		
1079	172340	KIPAR0= 172340
1080	172342	KIPAR1= 172342
1081	172344	KIPAR2= 172344
1082	172346	KIPAR3= 172346
1083	172350	KIPAR4= 172350
1084	172352	KIPAR5= 172352
1085	172354	KIPAR6= 172354
1086	172356	KIPAR7= 172356

```

1088      .SBTTL  VSV-11 INSTRUCTION EQUATES.
1089
1090      ;PLOTING INSTRUCTIONS.
1091
1092      100000      CHAR== 100000      ; CHARACTER MODE.
1093      104000      SVEC== 104000      ; SHORT VECTOR MODE.
1094      110000      LVEC== 110000      ; LONG VECTOR MODE.
1095      114000      APNT== 114000      ; ABSOLUTE POINT MODE.
1096      120000      GHX== 120000      ; GRAPH/HISTOGRAM X MODE.
1097      124000      GHY== 124000      ; GRAPH/HISTOGRAM Y MODE.
1098      130000      RPNT== 130000      ; RELATIVE POINT MODE.
1099
1100      ;PIXEL DATA FIELD <10:2> FOR THE ABOVE OPCODES.
1101
1102      002000      L0== BIT10          ; LEVEL 0 (BLACK) IS JUST THE ENABLE.
1103      003774      L255== L0.<255.*BIT2> ; LEVEL 255 IS THE MAX POSSIBLE.
1104      003774      ALL=- L255
1105      003774      ALLB== 3774          ; ALL INTENSITIES (WITH BLINK).
1106      003374      ALLNB== 3374        ; ALL INTENSITIES (WITHOUT BLINK).
1107
1108
1109      ;COLOR DEFINITIONS:
1110      003774      WHITE == ^03774 ; ALL
1111      002400      GRN1 == ^0<400!2000> ; BIT8.L0
1112      003000      GRN2 == ^0<1000!2000> ; BIT9.L0
1113      003400      GRN3 == GRN1!GRN2 ; BIT8!BIT9
1114      002200      REDG == ^0<200!2000> ; BIT7!L0
1115      002100      BLUEG == ^0<100!2000> ; BIT6.L0
1116      003600      YELLOW == GRN3!REDG
1117      003500      EGGBL == GRN3!BLUEG
1118      003300      VIOLET == GRN2!REDG!BLUEG
1119      003200      GOLD == GRN2!REDG
1120      002300      MAGEN == REDG!BLUEG
1121
1122      ; COLOR LOOK-UP-RAM DEFINITIONS.
1123
1124      010000      LREAD== 1*BIT12      ; CSR -- READ DATA FROM ADDR <7:0>.
1125      002000      LDI== 1*BIT10      ; INTERRUPT ON READ/WRITE DONE.
1126
1127      010000      LSET== 1*BIT12      ; DATA REG -- SET ALL ADDRESSES TO...
1128      ;...DATA PATTERN IN <11:0>...
1129      ;...BLU<11:8>, GRN<7:4>, RED<3:0>.
1130
1131      000100      GCOFF== 1*BIT6      ; MAINT REG -- GAMMA CORRECT OFF.
1132      000040      CVC== 1*BIT5      ; COMP VIDEO > MDAC (READ-ONLY).
1133      000020      REDC== 1*BIT4      ; RED DAC > MDAC (READ-ONLY).
1134      000010      GRNC== 1*BIT3      ; GRN DAC > MDAC (READ-ONLY).
1135      000004      BLUC== 1*BIT2      ; BLU DAC > MDAC (READ-ONLY).
1136      000000      MDV0== 0          ; SET MAINT DAC TO 0.0 V.
1137      000001      MDV1== 1          ; 0.5 V.
1138      000002      MDV2== 2          ; 1.0 V.
1139      000003      MDV3== 3          ; 1.35 V.
1140
1141      ; BIT MAP MODE DEFINITIONS.
1142
1143      136000      BM14== 136000      ; MODE 1 WITH 4 BIT PIXELS.
1144      137000      BM18== 137000      ; MODE 1 WITH 8 BIT PIXELS.

```

1145				; PIXEL COUNT IN <8:0>.
1146				
1147	134000	BM04==	134000	; MODE 0 WITH 4 BIT PIXELS.
1148	135000	BM08==	135000	; MODE 0 WITH 8 BIT PIXELS.
1149	000000	M32==	0	; 32 SQUARE PIXEL ARRAY.
1150	000001	M64==	1	; 64 SQUARE.
1151	000002	M128==	2	; 128 SQUARE.
1152	000003	M256==	3	; 256 SQUARE.
1153	000010	EX2==	10	; EXPAND BY 2, NO SMOOTHING.
1154	000020	EX4==	20	; EXPAND BY 4, NO SMOOTHING.
1155	000050	EXSM2==	50	; EXPAND AND SMOOTH BY 2.
1156	000060	EXSM4==	60	; EXPAND AND SMOOTH BY 4.
1157	000100	R256==	100	; 256 X 256 RESOLUTION.
1158	000400	RND8==	400	; ROUND FOR 8-BIT-WIDE IMAGE MEMORY
1159	000200	ODDSKP==	200	; SKIP ODD-NUMBERED LINES ON SCREEN (FOR NON-INTERLACED) ; 0001
1160				
1161	144000	RNLN==	144000	; OP-CODE FOR RUN-LENGTH MODE
1162	000100	DBLPIX==		; DOUBLE THE PIXEL COUNT
1163	000040	YDOWN==	BIT5	; DISPLAY LINES TOP-TO-BOTTOM
1164	000400	RLSKIP==	BIT8	; SKIP LINES, WITHOUT THROWING THE DATA AWAY
1165				; (FOR RUN-LENGTH MODE ONLY)
1166				; SKIPS EVEN LINES IF START ON EVEN, & VICE-VERSA
1167	162000	IMREAD==	162000	

1169		;CONTROL AND STATUS INSTRUCTIONS.	
1170			
1171	146040	CUOFF== 146040	;CURSOR (JOY-STICK) VIDEO OFF.
1172	146060	CUON== 146060	;CURSOR VIDEO ON.
1173	146100	CURD== 146100	;READ JSX,JSY => DXR,DYR.
1174	146016	CUIM== 146016	;INTERRUPT ON MATCH (SWITCH DISABLED).
1175	146013	CUIS== 146013	;INTERRUPT ON SWITCH (MATCH DISABLED).
1176	146012	CUIOFF== 146012	;INTERRUPTS (BOTH) OFF.
1177	175000	CUWT== 175000	; CURSOR WRITE.
1178	175000	LDCP== 175000	; LOAD CURSOR POSITION (-CUWT)
1179	146000	LDJSS== 146000	;LOAD JOYSTICK STATUS DISPLAY INSTRUCTION
1180	175400	LDECC== 175400	;LOAD EXTENDED CURSOR CONTROL
1181	000002	SWCHON== BIT1	; SOFT SWITCH ON.
1182	175401	BLINK== 175401	; BLINK.
1183	175400	NBLINK==175400	; NO BLINK.
1184			
1185	000002	JSWE == BIT1	; JOYSTICK SWITCH-ENABLE WRITE ENABLE
1186	000001	JSWD == BIT0	; JOYSTICK SWITCH-ENABLE WRITE DATA
1187	000010	JMWE == BIT3	; CURSOR MATCH-ENABLE WRITE ENABLE
1188	000004	JMWD == BIT2	; CURSOR MATCH-ENABLE WRITE DATA
1189	000040	JCWE == BIT5	; CROSSHAIR-INTENSITY WRITE-ENABLE
1190	000020	JCWD == BIT4	; CROSSHAIR-INTENSITY WRITE DATA
1191			
1192			
1193	150000	SETHB== 150000	;SET GRAPH/HISTO BASE LINE.
1194	152000	SETCB== 152000	;SET CHARACTER BASE ADDRESS.
1195			
1196	160000	DJMP== 160000	;DISPLAY JUMP.
1197	160001	DJMS== 160001	;DISPLAY JUMP-TO-SUBROUTINE
1198	164000	DNOP== 164000	;DISPLAY NO-OP.
1199	165000	DPOP== 165000	;DPOP = RTS FROM CHARACTER SUB-PIX.
1200	164001	SYNC== DNOP+1	;PROCEED IN SYNC = DNOP + N <8:0>.
1201			;WAITS FOR N OCCURRENCES OF VERTICAL...
1202			;...SYNC START, THEN PROCEEDS...
1203			;...(16.6MS @ 60HZ OR 20MS @ 50HZ).
1204	000001	AUXSEG== BIT0	;SPECIFY AUX. SEGMENT IN DISPATCH ADDRESSES. aaaI
1205			
1206	170200	SWTCH== 170200	;SWITCH READ/WRITE STATUS.
1207	170140	CLRMEM== 170140	;CLEAR IMAGE MEMORY TO 0'S.
1208	170100	SETMEM== 170100	;SET IMAGE MEMORY TO CURRENT PIX DATA.
1209			
1210	172000	STOP== 172000	;DISPLAY STOP.
1211	171000	SIOFF== 171000	;STOP INTERRUPT DISABLE.
1212	171400	SION== 171400	;STOP INTERRUPT ENABLE.
1213	173000	STOPN== STOP!SIOFF	;STOP, DO NOT INTERRUPT.
1214	173400	STOPI== STOP!SION	;STOP AND INTERRUPT.
1215			
1216	174100	GXI== 174100	;SET X INCREMENT <5:0> FOR GRAPH/HST Y.
1217	174100	GYI== GXI	;SET Y INCREMENT <5:0> FOR GRAPH/HST X.
1218			
1219	000001	HCPY== BIT0	;ENABLE HARD-COPY.
1220	000010	SWE== BIT3	;ENABLE SWITCH RD/WRT.
1221	176000	PROTEC== 176000	;PROTECT MEMORY (OFF).
1222	176050	READ== 176040!SWE	;READ MEM (DISPLAY), SWITCH ENABLED.
1223	176034	WRT== 176024!SWE	;WRITE MEM (1'S AND 0'S), SWITCH ENABLED
1224	176036	WRT1== 176026!SWE	;WRITE MEM (1'S ONLY), SWITCH ENABLED.
1225	176074	RDWRT== READ!WRT	;READ, WRITE ALL DURING RETRACE.

1226	176076	RDWRT1== READ.WRT1	;READ, WRITE 1'S DURING RETRACE.
1227	176051	HCOPY== READ.HCPY	;READ MEM => HARD COPY DEVICE.
1228			
1229	000000	CH0== 0	;CHANNEL SELECT BITS...
1230	000400	CH1== BIT8	FOR MEMORY CONTROL...
1231	001000	CH2== BIT9	INSTRUCTIONS (176xxx).
1232	001400	CH3== BIT8.BIT9	
1233			
1234	000000	JS0== CH0	;JOY-STICK UNIT SELECT BITS...
1235	000400	JS1== CH1	FOR CURSOR CONTROL...
1236	001000	JS2== CH2	INSTRUCTIONS (146xxx).
1237	001400	JS3== CH3	

```

1239      ;OTHER USEFUL DEFINITIONS.
1240      ;
1241      040000      I==      BIT14      ;INTENSIFY VECTOR OR POINT.
1242      020000      MSX==     BIT13      ;MINUS SIGNS FOR...
1243      000100      MSY==     BIT6       ;...SHORT FORM VECTOR DATA.
1244      020000      MXY==     BIT13      ;MINUS SIGN FOR LONG FORM DATA.
1245      020000      M==      BIT13      ;ANOTHER MINUS SIGN
1246      040000      HST==     BIT14      ;USE GHX/GHY DATA AS HISTOGRAM.
1247      020000      CGRPH==    BIT13      ;USE GHX/GHY DATA AS CONNECTED GRAPH
1248      060000      FHST==    HST!BIT13  ;USE GHX/GHY DATA AS FILLED HISTOGRAM (BAR-GRAPH)
1249      010000      GHIINH== BIT12      ; INHIBIT ERROR DURING PLOT.
1250
1251      001776      MAXX==    511.*2      ;MAXIMUM X ADDRESS (512 X 512 X ?).
1252      001776      MAXY==    MAXX       ;SAME FOR Y.
1253      000776      HAFX==    <MAXX/2>-1 ;HALF MAX X (CENTER SCREEN).
1254      000776      HAFY==    HAFX       ;SAME FOR Y.
1255
1256      001776      MAXY50== 511.*2      ;MAX VISIBLE Y ON 50HZ SYSTEM.
1257      000776      HAFY50== <MAXY50/2>-1 ;HALF VISIBLE Y.
1258
1259      001676      MAXY60== <511.-32.>*2 ;MAX VISIBLE Y ON 60HZ SYSTEM.
1260      000736      HAFY60== <MAXY60/2>-1 ;HALF VISIBLE Y.
1261
1262      000002      DX==      2           ;DELTA X (1 PIXEL UNIT).
1263      000002      DYI==    DX         ;DELTA Y, INTERLACED MODE.
1264      000004      DYNI==   DX*2       ;DELTA Y, NON-INTERLACED MODE.
1265
1266      ;DSR REGISTER SELECT CODES:
1267      000000      SELDSR== 0           ;SELECT REAL DSR
1268      000001      SELPCS== 1           ;SELECT PCSAVE
1269      000002      SELFLG== 2           ;SELECT FLAGS
1270      000003      SELCSR== 3           ;SELECT CSR
1271      000004      SELMRR== 4           ;SELECT MAIN RELOC
1272      000004      SELMPM== SELMRR      ;SELECT MAIN PROTECT
1273      000006      SELXRR== 6           ;SELECT AUX. RELOC
1274      000006      SELXPM== SELXRR     ;SELECT AUX. PROTECT
1275
1276      000005      SELHBA== 5           ;SELECT H-BASE
1277      000007      SELCBA== 7           ;SELECT C-BASE
1278      000010      WE==      BIT3       ;DSR WRITE-ENABLE
1279      000014      SETMRR== SELMRR.WE   ;SET MAIN-SEG RELOCATION REGISTER TO 0...
1280      ;...OR SETMR!<ADDR/2> TO RELOCATE.
1281      000015      SETMPM== -15         ;SET MAIN-SEG PROTECTION MASK
1282      000016      SETXRR== SELXRR.WE   ;SET AUX.-SEG RELOCATION REGISTER
1283      000017      SETXPM== 17         ;SET AUX.-SEG PROTECTION MASK
1284      000013      WRTCSR== SELCSR.WE   ;WRITE THE CSR REGISTER
1285      000012      CLFLGS== SELFLG.WE   ;CLEAR THE FLAGS REGISTER
1286      000010      SETDSR== SELDSR.WE   ;SET REAL DSR.
1287      000013      SETCSR== WRTCSR      ;SET THE CSR REGISTER.
1288      000012      SETFLG== CLFLGS      ;CLEAR THE FLAGS REGISTER.
1289
1290      ;'WRITE' FUNCTIONS FOR DXR (BITS 15-14):
1291      000000      PIXRBK== 0*BIT14      ;PIXEL READBACK -- <15:14>=00
1292      042000      WRTJSS== 1*BIT14!BIT10 ;WRITE JOYSTICK STATUS REGISTER -- <15:14>=01
1293      100000      RSTPOS== 2*BIT14      ;RESTORE TRUE X-Y POSITION TO DXR,DYR -- <15:14>=10 ;0001
1294      140000      WRTCPX== 3*BIT14      ;WRITE CURSOR POSITION X-COORDINATE -- <15:14>=11
1295      040000      GTJSSW== BIT14       ;GET JOYSTICK SWITCH INTERRUPT COORDINATES

```

```

1296
1297
1298
1299      040000
1300      100000
1301      140000
1302
1303
1304      000040
1305      000100
1306      000200
1307
1308
1309      000001
1310      000002
1311      000004
1312      000010
1313
1314
1315
1316      100000
1317      104003
1318      114003
1319      124003
1320      120003
1321      140003
1322      144003
1323      150003
1324
1325
1326
1327      010000
1328      010000
1329      004000
1330      042000
1331      160000
1332
1333
1334
1335
1336
1337      000403
1338      000402
1339      000401
1340      000400
1341
1342
1343
1344
1345
1346      000004

; 'WRITE' FUNCTIONS FOR DYR (BITS 15-14):
; PIXEL READBACK IS THE SAME - 00
WRTMSR== 1*BIT14      ;WRITE MEMORY STATUS REGISTER BA -- <15:14>=01
SINIT== 2*BIT14      ;PERFORM 'SOFT-INIT' -- <15:14>=10
WRTCPY== 3*BIT14      ;WRITE CURSOR POSITION Y-COORDINATE -- <15:14>=11

; CSR BIT DEFINITIONS:
FORCSI== BIT5      ;FORCE STOP INTERRUPT
ENECHK== BIT6      ;ENABLE ERROR CHECKING
CHPROT== BIT7      ;CHANNEL PROTECT
;0001
; FLAGS REGISTER BIT POSITIONS:
PINTRA== BIT0      ;PENDING INTERRUPT, CHANNEL A
PINTRB== BIT1      ;PENDING INTERRUPT, CHANNEL B
PVEC2== BIT2      ;VECTOR BIT 2 FOR INTERRUPT
JSLCKO== BIT3      ;JOYSTICK INTERRUPT LOCKOUT

; ERROR CODE DEFINITIONS.
ERR== 100000      ; ERROR (COMPOSITE).
NXME== ERR!4000!SELCSR ; NONEXISTANT MEMORY ERROR.
MPE== ERR!14000!SELCSR ; MEMORY PROTECTION ERROR.
RSVDOP== ERR!24000!SELCSR ; RESERVED OPCODE, RESERVED OPERATION.
SEQERR== ERR!20000!SELCSR ; SEQUENCE ERROR.
SYNCTO== ERR!40000!SELCSR ; SYNC TIMEOUT FROM IMAGE MEMORY.
DAVTO== ERR!44000!SELCSR ; DATA AVAILABLE TIMEOUT (PIXEL READBACK).
DRDYTO== ERR!50000!SELCSR ; DATA READY TIMEOUT (JOYSTICK STATUS).

; DXR, DYR BITS
CHIE ==BIT12      ; CROSSHAIR INTENSITY ENABLE STATUS (1=ON)
JSMIES ==BIT12      ; JOYSTICK MATCH INTERRUPT ENABLE STATE
JSSIES ==BIT11      ; JOYSTICK SWITCH INTERRUPT ENABLE STATE
WJSS ==BIT14.BIT10 ; WRITE JOYSTICK STATUS REGISTER
WECC ==160000      ; WRITE EXTENDED CURSOR CONTROL

; SOME HANDY -11 OPDEF'S.
SKP3= BR+3      ;SKIP NEXT 3 WORDS.
SKP2= BR+2      ;SKIP NEXT 2 WORDS.
SKP1= BR+1      ;SKIP NEXT WORD.
SKP0= BR+0      ;SAME AS A NOP.

; SOME GENERAL EQUATES.
ERRVEC==4      ; POINTER TO ERROR VECTOR FOR BUS TIME OUT.

```

1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
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

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

```

: MACROS TO STANDARDIZE THE INITIALIZATION OF EACH TEST.
: PROVIDES AN ASCII TITLE STRING, AND POINTER FOR
: SUBSEQUENT PRINTING (IF REQ'D) VIA 'PRINTF TNAM,TNUM' CALL.
: ALSO INIT'S A SEQUENTIAL ERROR NUMBER SEQUENCE, AND
: SETS AN ITERATION COUNT FOR THE CURRENT TEST.
:

```

```

: .MACRO BEGIN.TEST TEXT,LK,?TAG1,?TAG2
.RADIX 10
TSTITLE \TV+1,<TEXT>,<LK>
.RADIX
.NLIST
.LIST MC
.LIST
BGNTEST
TN=TSTESTNUM ;TEST NUMBER
EN=TN*100. ;INIT ERROR NUMBER SEQUENCE.
.IIF B <LK>, MOV #1,LOOPK ;DEFAULT = 1.
.IIF NB <LK>, MOV #LK,LOOPK
MOV #TN,TNUM ;SET TEST NUMBER...
MOV #TAG1,TNAM ;...AND NAME POINTER.
BR TAG2 ;SKIP OVER THE ASCII.
:TAG1: .ASCII \NZATEST %D2%A: TEXT\
TAG1: .ASCII \ZA 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

```

1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
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
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

```
.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  SFERR ADDR,PNTR,CK      ; SOFT ERROR.
: EN-EN+1
: JSR    PC,INCERK
: ERRSOF EN,ADDR,PNTR
: JSR    PC,CKEMAX
: .IF    NB,CK
: CKLOOP
: .ENDC
: .ENDM  SFERR

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

```

1504 .NLIST
1505 .NLIST ME
1506 .LIST
1507     EN=EN+1
1508     JSR     PC,INCERK
1509     ERRHRD  EN,ADDR,PNTR
1510     JSR     PC,CKEMAX
1511     .IF     NB,CK
1512     CKLOOP
1513     .ENDC
1514     .ENDM   HRDERR
1515
1516
1517 ;
1518 ; MACRO - DO SOME COMMON STUFF AT THE BEGINNING OF A TEST.
1519 ;
1520     .MACRO  COMBEG
1521 .NLIST
1522 .LIST ME
1523 .LIST
1524     JSR     PC,TSTGO      ; TITLE.
1525     JSR     PC,DPRESET    ; DO SOFT INIT.
1526 .NLIST ME
1527     .ENDM   COMBEG
1528
1529 ;
1530 ; MACRO - DO SOME COMMON STUFF AT THE END OF A TEST.
1531 ;
1532     .MACRO  COMEND TAG,?L,?X
1533 .NLIST
1534 .LIST ME
1535 .LIST
1536     JSR     PC,LOOP      ; REPEAT 'TIL LOOPER EXPIRES.
1537     BCS     L
1538     BR      X
1539
1540 L:     JMP     TAG
1541
1542 X:     JSR     PC,TSTEND   ; PRINT ERROR SUMMARY, IF REQ'D.
1543 .NLIST
1544 .NLIST ME
1545 .LIST
1546     .ENDM   COMEND
1547
1548 ;
1549 ; MACRO TO CONTROL ITERATION LOOPS.
1550 ;
1551     .MACRO  LOOPTO TAG
1552 .NLIST
1553 .LIST ME
1554 .LIST
1555     JSR     PC,LOOP
1556     BCS     TAG
1557 .NLIST
1558 .NLIST ME
1559 .LIST
1560     .ENDM   LOOPTO

```

```

1561
1562
1563      ; MACRO - WRITE FROM SPEC'D DATA SOURCE INTO SPEC'D DSR ACCESS REGISTER.
1564      ; ONLY DSR, FLG, CSR, MRR, MPM, XRR, XPM ARE VALID ARGUMENTS FOR 'DSRREG'.
1565      ; RO=WORK.
1566      ;
1567      .MACRO WTDSRA DATSRC,DSRREG
1568      .NLIST
1569      .LIST MEB
1570      .LIST
1571          .NTYPE      .NT,DATSRC
1572          .IF          EQ,.NT&^077-^027
1573          MOV          DATSRC&^C17!SET'DSRREG,@DSR
1574          .IFF
1575          MOV          #SET'DSRREG',RO ; WRITE DATSRC INTO DSRREG .
1576          BIS          DATSRC,RO
1577          MOV          RO,@DSR
1578          .ENDC
1579      .NLIST
1580      .LIST MEB
1581      .LIST
1582          .ENDM      WTDSRA
1583
1584      ;
1585      ; MACRO - READ FROM SPEC'D DSR ACCESS REGISTER INTO SPEC'D DATA DESTINATION.
1586      ; ONLY DSR, PCS, FLG, CSR, MRR, MPM, XRR, XPM, HBA, CBA ARE VALID
1587      ; 'DSRREG' ARGUMENTS.
1588      ; RO=WORK.
1589      ;
1590      .MACRO RDDSRA DSRREG,DATDST
1591      .NLIST
1592      .LIST ME
1593      .LIST
1594          MOV          #SEL'DSRREG',@DSR      ; READ DSRREG INTO DATDST .
1595          MOV          @DSR,DATDST
1596      .NLIST
1597      .LIST ME
1598      .LIST
1599          .ENDM      RDDSRA
1600
1601      ;
1602      ; MACRO - IF R1-R2, BRANCH TO OK, ELSE CALL HARDWARE ERROR.
1603      ;
1604      .MACRO IFERROR ARG1,ARG2,?OK$,CK
1605      .NLIST
1606      .LIST ME
1607      .LIST
1608          CMP          R1,R2                  ; OK?
1609          BEQ          OK$                     ; YES.
1610          HRDERR       ARG1,ARG2,CK           ; NO.
1611      .NLIST
1612      .LIST ME
1613      .LIST
1614      OK$:
1615      .NLIST ME
1616          .ENDM      IFERROR
1617

```

```

1618      ;
1619      ; MACRO - IF R1=R2 AND R3=R4, BRANCH TO OK, ELSE CALL HARDWARE ERROR.
1620      ;
1621      .MACRO IFERRX2 ARG1,ARG2,?OK$,?ERR,CK
1622      .NLIST
1623      .LIST ME
1624      .LIST
1625      CMP      R1,R2      ; 1ST ITEM OK?
1626      BNE      ERR      ; NO.
1627      CMP      R3,R4      ; 2ND ITEM OK?
1628      BEQ      OK$      ; YES.
1629      ERR:      HRDERR ARG1,ARG2,CK ; NO.
1630      .NLIST
1631      .LIST ME
1632      .LIST
1633      OK$:
1634      .NLIST ME
1635      .ENDM IFERRX2
1636
1637      ;
1638      ; MACRO - SOFTWARE INITIALIZE THE DPU. THIS DIFFERS FROM 'DPINIT'
1639      ; IN TWO WAYS. FIRST, IT IS ASSUMED THAT THE DPU IS STOPPED AND
1640      ; NOT HUNG UP. SECOND, NO CHECKING IS DONE BY THIS CODE.
1641      ;
1642      .MACRO INITDP
1643      .NLIST
1644      .LIST ME
1645      .LIST
1646      JSR      PC,DPINIT ; GO DO SOFT INIT, CHECK THE STATE.
1647      .NLIST
1648      .NLIST ME
1649      .LIST
1650      .ENDM INITDP
1651
1652      .MACRO DPSTART ADRS
1653      .NLIST
1654      .LIST ME
1655      .LIST
1656      MOV      ADRS,@DPC ; START THE DPU.
1657      JSR      PC,WAITF ; WAIT FOR DISPLAY STOP.
1658      .NLIST
1659      .NLIST ME
1660      .LIST
1661      .ENDM DPSTART
1662
1663      .MACRO DPCONT
1664      .NLIST
1665      .LIST ME
1666      .LIST
1667      BIS      #BIT0,@DPC ; CONTINUE THE DPU.
1668      JSR      PC,WAITF ; WAIT FOR DISPLAY STOP.
1669      .NLIST
1670      .NLIST ME
1671      .LIST
1672
1673
1674

```

```

1675             .ENDM   DPCONT
1676
1677 ;MACRO TO DO A 3-OPERAND BIT-SET:
1678             .MACRO   BISW3   S1,S2,DST
1679 .NLIST
1680 .LIST MEB
1681 .LIST
1682             .NTYPE   .NT,DST
1683             .IF      EQ,.NT&^070
1684             MOV      S2,DST
1685             BIS      S1,DST
1686             .IFF
1687             MOV      S2,-(SP)
1688             BIS      S1,(SP)
1689             MOV      (SP)+,DST
1690             .ENDC
1691 .NLIST
1692 .NLIST MEB
1693 .LIST
1694             .ENDM   BISW3
1695
1696 ;MACRO TO DO A 3-OPERAND BIT-CLEAR:
1697             .MACRO   BICW3   S1,S2,DST
1698 .NLIST
1699 .LIST MEB
1700 .LIST
1701             .NTYPE   .NT,DST
1702             .IF      EQ,.NT&^070
1703             MOV      S2,DST
1704             BIC      S1,DST
1705             .IFF
1706             MOV      S2,-(SP)
1707             BIC      S1,(SP)
1708             MOV      (SP)+,DST
1709             .ENDC
1710 .NLIST
1711 .NLIST MEB
1712 .LIST
1713             .ENDM   BICW3
1714
1715 ;MACRO TO SET UP THE INTERRUPT MASK AND PRIME THE INTERRUPT FLAG:
1716             .MACRO   INTSET   IEX
1717 .NLIST
1718 .LIST MEB
1719 .NLIST ME
1720 .LIST
1721             .IF      NB,IEX
1722             MOV      #^0177400!IOKCKIN.IOK'IEX,INTMASK      ; PRIME FLAG, EXPECT IEX
1723             .IFF
1724             MOV      #^0177400!IOKCKIN,INTMASK      ; PRIME FLAG, NO INTR. EXPECTED.
1725             .ENDC
1726 .NLIST
1727 .NLIST MEB
1728 .LIST
1729             .ENDM   INTSET
1730
1731

```

1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745

```

: 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
```

1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770

.SBTTL USER PATCHABLE FLAGS

:
: THE FOLLOWING LOCATIONS MAY BE PATCHED BY THE USER
: TO OBTAIN THE RESULTS DESCRIBED FOR EACH.
:

FORCER: 0 : FORCE TYPE ALL HARD ERRORS (THE ONES CALLED -
: - BY THE MACRO 'IFERROR'). AN ERROR NEED NOT -
: - EXIST, JUST ASSUME AND TYPE THE MESSAGE.

MFGMO: 1700 : MANUFACTURING MEMORY 0 MASTER SPECIFICATION.
: IN MANUFACTURING MODE, MEMORY 0 IS COMPARED
: WITH THIS VALUE WHEN SYSTEM CONFIGURATION IS
: EXECUTED. DISCREPANCY IS REPORTED.
: (CURRENTLY SHOWN FOR 4 BIT MEMORY.)

MFGSO: 120000 : MANUFACTURING SYNC CHAN 0 MASTER SPECIFICATION.
: IN MANUFACTURING MODE, SYNC CHAN 0 IS COMPARED
: WITH THIS VALUE WHEN SYSTEM CONFIGURATION IS
: EXECUTED. DISCREPANCY IS REPORTED.
: (CURRENTLY SHOWN FOR INTERLACED SYNC (CHAN.)

```

1772 .SBTTL GLOBAL DATA SECTION
1773
1774 : **
1775 : THE GLOBAL DATA SECTION CONTAINS DATA THAT ARE USED
1776 : IN MORE THAN ONE TEST.
1777 : --
1778
1779 :
1780 : THE FOLLOWING DATA ARE SET FOR EACH UNIT AT INIT TIME.
1781 : SINGLE UNIT DEFAULTS (LISTED) ARE IN THE DEFAULT P-TABLE.
1782
1783 003026 000000 UNITN:: 0 ;UNIT # UNDER TEST.
1784 003030 000000 QVP:: 0 ;QUICK VERIFY FLAG.
1785
1786 :
1787 : DEVICE REGISTER ADDRESSES *** KEEP IN THE ORDER GIVEN ***
1788
1789 003032 172010 DPC:: 172010 ;DISPLAY PC
1790 003034 172012 DRR:: ;
1791 003034 172012 DSR:: 172012 ;DISPLAY STATUS REGISTER (ALSO REL REG).
1792 003036 172014 DXR:: 172014 ;DISPLAY X POSITION REGISTER.
1793 003040 172016 DYR:: 172016 ;DISPLAY Y POSITION REGISTER.
1794
1795 003042 172020 LSR:: 172020 ; LOOK-UP-RAM STATUS.
1796 003044 172022 LDR:: 172022 ; DATA.
1797 003046 172024 LMR:: 172024 ; AND MAINT REGISTERS.
1798
1799 003050 000200 DPRI:: PRI04 ;INTERRUPT PRIORITY.
1800 003052 000320 STPV:: 320 ;STOP VECTOR
1801 003054 000324 JSMV:: 324 ;JOY-STICK MATCH VECTOR.
1802 003056 000330 TOTV:: 330 ;TIME-OUT VECTOR
1803 003060 000334 JSSV:: 334 ;JOY-STICK SWITCH (DEMAND) VECTOR.
1804 003062 000340 LUTV:: 340 ; LUT READ/WRITE DONE VECTOR.
1805
1806 003064 040000 OPTI:: I ;OPTIONAL "INTENSIFY" BIT FOR DPU...
1807 : ...IF DPUMOD = 0, OPTI = 1 (INTENSIFY).
1808 : ...OTHERWISE, OPTI = 0 (UNINTENSIFY).
1809
1810 :
1811 : AND THESE ARE SET BY THE IMAGE MEMORY SIZER.
1812
1813 003066 000004 PDBITS:: 4 ;NUMBER OF PIXEL BITS (2, 4, 6, OR 8).
1814 003070 000100 PDI:: 1*BIT6 ;PIXEL DATA INCREMENT (MIN INTENSITY).
1815 003072 001700 PDF:: 17*BIT6 ;PIXEL DATA FIELD (MAX INTENSITY).
1816 003074 176077 PDM:: ^C<17*BIT6> ;PIXEL DATA FIELD MASK.
1817
1818 :
1819 : STANDARD CONFIGURATIONS
1820 :
1821 : BITS PDI PDF
1822 : 2 0400 1400
1823 : 4 0100 1700

```

```

1825      ;
1826      ; STORAGE FOR DEVICE REGISTERS
1827      ;
1828 003076 000000 100000 000000 DUMMY: 0,100000,0,0 ; DUMMY DEVICE REGISTERS...
1829 003106 000000 000000 000000      0,0,0,0,0,0,0,0 ; ...FOR MULTI-UNIT CHECKOUT.
1830      ;
1831      ;
1832 003126 000000 IDPC:: 0 ; SAVED DPC ON ANY INTERRUPT.
1833 003130 000000 IDSR:: 0 ; DITTO DSR
1834 003132 000000 IDXR:: 0 ; DITTO DXR
1835 003134 000000 IDYR:: 0 ; DITTO DYR
1836 003136 000000 ILSR:: 0 ; DITTO LUT CSR
1837 003140 000000 ILDR:: 0 ; DITTO LUT DATA
1838 003142 000000 ILMR:: 0 ; DITTO LUT MAINT.
1839      ;
1840      ; SAVED "INTERNAL" REGISTERS ON AN INTERRUPT (IN ORDER OF SELECT CODE):
1841 003144 000000 ISDSR:: 0 ; SAVED "SELECTED" DSR
1842 003146 000000 IPCSAV:: 0 ; SAVED PCSAVE
1843 003150 000000 IFLAGS:: 0 ; SAVED FLAGS
1844 003152 000000 ICSR:: 0 ; SAVED CSR
1845 003154 000000 IMAIN:: 0 ; SAVED MAIN SEG. MMGT.
1846 003156 000000 IHBASE:: 0 ; SAVE HISTOGRAM BASE
1847 003160 000000 IAUX:: 0 ; SAVED AUX. SEG MMGT.
1848 003162 000000 ICBASE:: 0 ; SAVED CHARACTER BASE
1849      ;
1850 003164 000000 SDPC:: 0 ; SAVED DPC AT ANY OTHER TIME.
1851 003166 000000 SDSR:: 0 ; DITTO DSR
1852 003170 000000 SDXR:: 0 ; DITTO DXR
1853 003172 000000 SDYR:: 0 ; DITTO DYR
1854      ; SAVED INTERNAL REGISTERS AT OTHER TIMES
1855 003174 000000 SSDSR:: 0 ; SAVED "SELECTED" DSR
1856 003176 000000 SPCSAV:: 0 ; SAVED PCSAVE
1857 003200 000000 SFLAGS:: 0 ; SAVED FLAGS
1858 003202 000000 SCSR:: 0 ; SAVED CSR
1859 003204 000000 SMAIN:: 0 ; SAVED MAIN SEG. MMGT.
1860 003206 000000 SHBASE:: 0 ; SAVE HISTOGRAM BASE
1861 003210 000000 SAUX:: 0 ; SAVED AUX. SEG MMGT.
1862 003212 000000 SCBASE:: 0 ; SAVED CHARACTER BASE
1863      ;
1864      ; AND FINALLY SOME MISCELLANEOUS REGISTERS.
1865      ;
1866 003214 000000 SCFLG:: 0 ; SYS CONFIG FLAG (0=1ST TIME, NZ=SUBSEQUENT).
1867 003216 000000 LUTAV:: 0 ; LUT AVAILABLE FLAG (NZ = USE LUT).
1868 003220 000000 SHADLY:: 0 ; SHADING DELAY COUNTER (OPTIONAL).
1869 003222 110070 HUE:: NTSC8+2 ; CURRENT COLOR POINTER (OPTIONAL).
1870      ;
1871      ; STIK:: .BYTE -1, -1 ; 8 JOY-STICK AVAILABLE FLAGS...
1872      ; .BYTE -1, -1 ; ... -1 - I DUNNO...
1873      ; .BYTE -1, -1 ; ... 0 - NO...
1874      ; .BYTE -1, -1 ; ... +1 - YES.
1875      ;
1876 003224 000000 DUFLG:: 0 ; "DROPPED UNIT" FLAG. INHIBITS DPU...
1877      ; ...CODE IN "CLEAN-UP".
1878 003226 000000 NODEV:: 0 ; FLAG TO SAY NO DEVICE.
1879      ;
1880 003230 000001 INTLAC:: 1 ; INTERLACED MODE
1881 003232 000000 TEMP1:: 0 ; SOME TEMP LOCATIONS.

```

1882	003234	000000	TEMP2:: 0			
1883	003236	000000	XXCOMM::	0		; XXDP+ COMM BLOCK POINTER.
1884	003240	000000	FREE:: 0			; 1ST FREE MEMORY ADDRESS...
1885	003242	000000	FRESIZ::	0		; ...AND SIZE (IN WORDS).
1886	003244	000000	KTFLG:: 0			; KT11, MEM AVAIL FLAG -
1887						; - 0 = <24K OR NO KT -
1888						; - NZ = >24K AND KT.
1889	003246	000074	HZ:: 60.			; LINE FREQUENCY.
1890	003250	000000	YMAX:: 0			; MAX Y ACCORDING TO LINE FREQUENCY.
1891	003252	000000	MEMFLG:: 0			; MEM FLAGS (1'S IN BITS 0-3 REFLECT MEMS AVAIL).
1892	003254	000000	SYCF LG:: 0			; SYNC CHAN FLAGS (1'S IN BITS 0-3 REFLECT SYNC CHANS AVAIL)
1893	003256		CTAB::			; CONFIGURATION TABLES.
1894	003256		MEMTAB::			
1895	003256	000000	CTABM:: 0			; MEMORY CONFIG WORK.
1896	003260	000000		0		
1897	003262	000000		0		
1898	003264	000000		0		
1899	003266	177777		-1		; END OF MEM TABLE.
1900	003270		SYCTAB::			
1901	003270	000000	CTABS:: 0			; SYNC CHAN CONFIG WORK.
1902	003272	000000		0		
1903	003274	000000		0		
1904	003276	000000		0		
1905	003300	177777		-1		; END OF SYNC CHAN TABLE.
1906	003302		CTABE::			; END OF CONFIG WORK AREA.
1907						
1908			; ERROR STATISTICS TABLE (1 WORD PER UNIT), 64 UNITS MAX:			
1909			:			
1910			:	0	=	UNIT NOT TESTED
1911			:	100000	=	UNIT ONLINE, NO ERRORS
1912			:	10XXXX	=	UNIT ONLINE, ENCOUNTERED XXXX ERRORS
1913			:	160000	-	UNIT DROPPED, NON-EXISTENT DEVICE REGISTER
1914			:	160001	-	UNIT DROPPED, NOT IDLE AT START
1915			:	14XXXX		UNIT DROPPED, ENCOUNTERED XXXX EPRORS
1916			:			
1917	003302		ERTABL: .BLKW	64.		
1918	003502	000000	ERTABE: 0			

```

1920
1921
1922
1923
1924
1925
1926
1927
1928 003504      040      040      116 NXR:      .ASCIIZ / NON-EXISTANT DEVICE REGISTER/
1929 003543      045      101      040 NXRX:     .ASCIIZ /%A ADDRESS: %06/
1930 003564      045      101      040 PCSX:     .ASCII  /%A DPC,DSR EXP'D: %06%A,%06%N/
1931 003622      045      101      040      .ASCIIZ /%A DPC,DSR REC'D: %06%A,%06/
1932 003657      045      101      040 XYX:      .ASCII  /%A DXR,DYR EXP'D: %06%A,%06%N/
1933 003715      045      101      040      .ASCIIZ /%A DXR,DYR REC'D: %06%A,%06/
1934 003752      045      116      045 FUSI:     .ASCII  /%N%A/
1935 003756      040      040      125 USI:      .ASCIIZ / UNEXPECTED 'STOP' INTERRUPT/
1936 004014      040      040      042 NSI:      .ASCIIZ / 'STOP' INTERRUPT EXPECTED, NOT RECEIVED/
1937 004066      045      116      045 FUTO:     .ASCII  /%N%A/
1938 004072      040      040      125 UTO:      .ASCIIZ / UNEXPECTED DPU 'ERROR' INTERRUPT/
1939 004135      040      040      104 NTO:      .ASCIIZ / DPU 'ERROR' INTERRUPT EXP'D, NOT REC'D/
1940 004206      045      116      045 FUMI:     .ASCII  /%N%A/
1941 004212      040      040      125 UMI:      .ASCIIZ / UNEXPECTED 'MATCH' INTERRUPT/
1942 004251      045      116      045 FUSWI:    .ASCII  /%N%A/
1943 004255      040      040      125 USWI:    .ASCIIZ / UNEXPECTED 'SWITCH' INTERRUPT/
1944 004315      045      116      045 FNOINTR:  .ASCII  /%N%A/
1945 004321      040      040      116 NOINTR:  .ASCIIZ / NO INTERRUPT WAS GENERATED/
1946 004356      040      040      111 IFAULT:  .ASCIIZ / INTERRUPT FAULT/
1947 004400      045      101      040 INTX:     .ASCIIZ /%A CPU PC: %06%A DPC: %06/
1948 004434      040      040      042 NOINIT:  .ASCIIZ / 'BUS-INIT' DIDN'T INITIALIZE DISPLAY/
1949 004503      040      040      123 SSF:      .ASCIIZ / START-NOP-STOP FAILURE/
1950 004534      040      040      122 RJF:      .ASCIIZ / RESUME-NOP-JUMP FAILURE/
1951 004566      040      040      112 JSF:      .ASCIIZ / JUMP SELF FAILURE/
1952 004612      040      040      105 ESF:      .ASCIIZ / EXTERNAL STOP FAILURE/
1953 004642      040      040      120 PDXI:    .ASCIIZ / PIXEL DATA XFER INCORRECT/
1954 004676      040      040      116 OPCF:    .ASCIIZ / NULL OPCODE FAILURE/
1955 004724      045      116      045 OPCFX:    .ASCIIZ /%N%A OPCODE: %06/
1956 004746      040      040      122 RELF:    .ASCIIZ / RELOCATION FAILURE/
1957 004773      045      116      045 RELFX:    .ASCIIZ /%N%A INITIAL DPC: %06%A, PELOC(DSR): %06%N/
1958 005047      040      040      101 APF:      .ASCIIZ / ABS POINT FAILURE/
1959 005073      040      040      114 LVF:      .ASCIIZ / LONG VECTOR FAILURE/
1960 005121      045      116      045 LONGX:    .ASCIIZ /%N%A ORIGIN: %04%A,%04%A DX,DY: %06%A,%06/
1961 005175      040      040      122 RPF:      .ASCIIZ / REL POINT FAILURE/
1962 005221      040      040      123 SVF:      .ASCIIZ / SHORT VECTOR FAILURE/
1963 005250      045      116      045 SHRTX:    .ASCIIZ /%N%A ORIGIN: %04%A,%04%A DXY: %06/
1964 005314      040      040      107 GHXF:    .ASCIIZ / GRAPH-HISTO X FAILURE/
1965 005344      040      040      107 GHYF:    .ASCIIZ / GRAPH-HISTO Y FAILURE/
1966 005374      045      116      045 GHXYX:    .ASCIIZ /%N%A AMPL: %06%A INCR: %02/
1967 005431      040      040      102 BMF:      .ASCIIZ / BIT MAP FAILURE/
1968 005453      045      116      045 BMX:      .ASCIIZ /%N%A OPCODE: %06/
1969 005475      040      040      103 CHRf:    .ASCIIZ / CHAR MODE FAILURE/
1970 005521      045      116      045 CHRFX:    .ASCIIZ /%N%A BASE: %06%A CHAR: %03/
1971 005556      040      040      124 PRBHNG:  .ASCII  / TEST ABORTED/
1972 005574      040      040      120 PRBH:     .ASCIIZ / PIXEL-READ-BACK FAILS OR HANGS DPU/
1973 005641      040      040      111 IMDI:    .ASCIIZ / IMAGE MEMORY DATA INCORRECT/
1974 005677      045      101      040 IMDIX:    .ASCII  /%A ADDRESS (X,Y): %04%A,%04/
1975 005733      045      116      045      .ASCIIZ /%N%A FIELD: %04%A EXP'D: %04%A REC'D: %04/
1976 006010      045      116      045 IMDIX2:   .ASCIIZ /%N%A SUSPECT RAM GROUP %01%A, IMAGE MEMORY CHANNEL: %01/

```

```

1977 006101 045 116 045 IMDIXC: .ASCII /%NZA ADDRESS (X,Y): %04XA,%04/
1978 006137 045 116 045 .ASCII /%NZA FIELD: %04XA EXP'D: %04XA REC'D: %04/
1979 006214 045 116 045 .ASCII /%NZA SUSPECT RAM GROUP %01XA, IMAGE MEMORY CHANNEL: %01/
1980 006306 040 040 114 NLRI: .ASCII / LUT READ INTERRUPT EXPECTED, NOT RECEIVED/
1981 006362 040 040 114 NLWI: .ASCII / LUT WRITE INTERRUPT EXPECTED, NOT RECEIVED/
1982 006437 040 040 125 ULRI: .ASCII / UNEXPECTED LUT READ INTERRUPT/
1983 006477 040 040 125 ULWI: .ASCII / UNEXPECTED LUT WRITE INTERRUPT/
1984 006540 045 101 040 LINTX: .ASCII /%A CSR: %06/
1985 006555 040 040 114 LAIER: .ASCII / LUT AUTO-INCR INCORRECT ON READ/
1986 006617 040 040 114 LAIEW: .ASCII / LUT AUTO-INCR INCORRECT ON WRITE/
1987 006662 040 040 114 LDINC: .ASCII / LUT DATA INCORRECT/
1988 006707 045 101 040 LRWX: .ASCII /%A CSR: %06XA EXP'D: %06XA REC'D: %06/
1989 006760 040 040 042 NSINIT: .ASCII / 'SOFT-INIT' DIDN'T INITIALIZE THE DPU/
1990 007030 040 040 042 BRINIT: .ASCII / 'BUS-RESET' DIDN'T INITIALIZE THE DPU/
1991 007100 040 040 104 IDPCE: .ASCII # DPC READ/WRITE ERROR#
1992 007127 040 040 104 IDSRE: .ASCII # DSR READ/WRITE ERROR#
1993 007156 040 040 103 ICSRE: .ASCII # CSR READ/WRITE ERROR#
1994 007205 040 040 115 IMRRE: .ASCII # MRR READ/WRITE ERROR#
1995 007234 040 040 115 IMPWPE: .ASCII # MPM WRITE/WWP (HBASE) READ ERROR#
1996 007277 040 040 115 IMPME: .ASCII # MPM READ/WRITE ERROR#
1997 007326 040 040 130 IXRRE: .ASCII # XRR READ/WRITE ERROR#
1998 007355 040 040 130 IXPME: .ASCII # XPM READ/WRITE ERROR#
1999 007404 040 040 130 IXPWPE: .ASCII # XPM WRITE/XWP (HBASE) READ ERROR#
2000 007447 040 040 110 IHBAE: .ASCII # HBA READ/WRITE ERROR#
2001 007476 040 040 103 ICBAE: .ASCII # CBA READ/WRITE ERROR#
2002 007525 040 040 104 JPDPCSE: .ASCII # DJMS/DPOP FAILURE (DPC)#
2003 007557 040 040 104 JPPCSE: .ASCII # DJMS/DPOP FAILURE (PCS)#
2004 007611 040 040 104 DPDSE: .ASCII / DPC(W),DSR(R) DUAL ADRS FAULT/
2005 007651 040 040 104 DSDPE: .ASCII / DSR(W),DPC(R) DUAL ADRS FAULT/
2006 007711 040 040 104 DSDXE: .ASCII / DSR(W),DXR(R) DUAL ADRS FAULT/
2007 007751 040 040 104 DSDYE: .ASCII / DSR(W),DYR(R) DUAL ADRS FAULT/
2008 010011 040 040 104 DSPCE: .ASCII / DSR(W),PCS(R) DUAL ADRS FAULT/
2009 010051 040 040 104 DSFLE: .ASCII / DSR(W),FLG(P) DUAL ADRS FAULT/
2010 010111 040 040 104 DSCSE: .ASCII / DSR(W),CSR(W) DUAL ADRS FAULT/
2011 010151 040 040 104 DSMRE: .ASCII / DSR(W),MRR(R) DUAL ADRS FAULT/
2012 010211 040 040 104 DSMPE: .ASCII / DSR(W),MPM(R) DUAL ADRS FAULT/
2013 010251 040 040 104 DSXRE: .ASCII / DSR(W),XRR(R) DUAL ADRS FAULT/
2014 010311 040 040 104 DSXPE: .ASCII / DSR(W),XPM(R) DUAL ADRS FAULT/
2015 010351 040 040 103 CSDSE: .ASCII / CSR(W),DSR(R) DUAL ADRS FAULT/
2016 010411 040 040 103 CSPCE: .ASCII / CSR(W),PCS(R) DUAL ADRS FAULT/
2017 010451 040 040 103 CSFLE: .ASCII / CSR(W),FLG(R) DUAL ADRS FAULT/
2018 010511 040 040 103 CSMRE: .ASCII / CSR(W),MRR(R) DUAL ADRS FAULT/
2019 010551 040 040 103 CSMPE: .ASCII / CSR(W),MPM(R) DUAL ADRS FAULT/
2020 010611 040 040 103 CSXRE: .ASCII / CSR(W),XRR(R) DUAL ADRS FAULT/
2021 010651 040 040 103 CSXPE: .ASCII / CSR(W),XPM(R) DUAL ADRS FAULT/
2022 010711 040 040 115 MRDSE: .ASCII / MRR(W),DSR(R) DUAL ADRS FAULT/
2023 010751 040 040 115 MRPCE: .ASCII / MRR(W),PCS(R) DUAL ADRS FAULT/
2024 011011 040 040 115 MRFLE: .ASCII / MRR(W),FLG(R) DUAL ADRS FAULT/
2025 011051 040 040 115 MRCSE: .ASCII / MRR(W),CSR(R) DUAL ADRS FAULT/
2026 011111 040 040 115 MRMPPE: .ASCII / MRR(W),MPM(R) DUAL ADRS FAULT/
2027 011151 040 040 115 MRXRE: .ASCII / MRR(W),XRR(R) DUAL ADRS FAULT/
2028 011211 040 040 115 MRXPE: .ASCII / MRR(W),XPM(R) DUAL ADRS FAULT/
2029 011251 040 040 115 MPDSE: .ASCII / MPM(W),DSR(R) DUAL ADRS FAULT/
2030 011311 040 040 115 MPPCE: .ASCII / MPM(W),PCS(R) DUAL ADRS FAULT/
2031 011351 040 040 115 MPFLE: .ASCII / MPM(W),FLG(R) DUAL ADRS FAULT/
2032 011411 040 040 115 MPCSE: .ASCII / MPM(W),CSR(R) DUAL ADRS FAULT/
2033 011451 040 040 115 MPMRE: .ASCII / MPM(W),MRR(R) DUAL ADRS FAULT/

```

2034	011511	040	040	115	MPXRE: .ASCIIZ /	MPM(W),XRR(R) DUAL ADRS FAULT/
2035	011551	040	040	115	MPXPE: .ASCIIZ /	MPM(W),XPM(R) DUAL ADRS FAULT/
2036	011611	040	040	130	XRDSE: .ASCIIZ /	XRR(W),DSR(R) DUAL ADRS FAULT/
2037	011651	040	040	130	XRPCE: .ASCIIZ /	XRR(W),PCS(R) DUAL ADRS FAULT/
2038	011711	040	040	130	XRFLE: .ASCIIZ /	XRR(W),FLG(R) DUAL ADRS FAULT/
2039	011751	040	040	130	XRCSE: .ASCIIZ /	XRR(W),CSR(R) DUAL ADRS FAULT/
2040	012011	040	040	130	XRMRE: .ASCIIZ /	XRR(W),MRR(R) DUAL ADRS FAULT/
2041	012051	040	040	130	XRMPE: .ASCIIZ /	XRR(W),MPM(R) DUAL ADRS FAULT/
2042	012111	040	040	130	XRXPE: .ASCIIZ /	XRR(W),XPM(R) DUAL ADRS FAULT/
2043	012151	040	040	130	XPDSE: .ASCIIZ /	XPM(W),DSR(R) DUAL ADRS FAULT/
2044	012211	040	040	130	XPPCE: .ASCIIZ /	XPM(W),PCS(R) DUAL ADRS FAULT/
2045	012251	040	040	130	XPFLC: .ASCIIZ /	XPM(W),FLG(R) DUAL ADRS FAULT/
2046	012311	040	040	130	XPCSE: .ASCIIZ /	XPM(W),CSR(R) DUAL ADRS FAULT/
2047	012351	040	040	130	XPMRE: .ASCIIZ /	XPM(W),MRR(R) DUAL ADRS FAULT/
2048	012411	040	040	130	XPMPE: .ASCIIZ /	XPM(W),MPM(R) DUAL ADRS FAULT/
2049	012451	040	040	130	XPXRE: .ASCIIZ /	XPM(W),XRR(R) DUAL ADRS FAULT/
2050	012511	040	040	115	MMACE: .ASCIIZ /	MAIN MEM MGT ACCESS FAULT/
2051	012545	040	040	101	XMMACE: .ASCIIZ /	AUX MEM MGT ACCESS FAULT/
2052	012600	040	040	115	MMXRE: .ASCIIZ /	MAIN MEM MGT NON-EXIST MEM FAULT/
2053	012643	045	101	040	DPCERA: .ASCIIZ /	%A (DPC) EXP'D: %06% REC'D: %06/
2054	012706	045	101	040	DSRERA: .ASCIIZ /	%A (DSR) EXP'D: %06% REC'D: %06/
2055	012751	045	101	040	DXRERA: .ASCIIZ /	%A (DXR) EXP'D: %06% REC'D: %06/
2056	013014	045	101	040	DYRERA: .ASCIIZ /	%A (DYR) EXP'D: %06% REC'D: %06/
2057	013057	045	101	040	PCSERA: .ASCIIZ /	%A (PCS) EXP'D: %06% REC'D: %06/
2058	013122	045	101	040	FLGERA: .ASCIIZ /	%A (FLG) EXP'D: %06% REC'D: %06/
2059	013165	045	101	040	CSRERA: .ASCIIZ /	%A (CSR) EXP'D: %06% REC'D: %06/
2060	013230	045	101	040	MRRERA: .ASCIIZ /	%A (MRR) EXP'D: %06% REC'D: %06/
2061	013273	045	101	040	MPMERA: .ASCIIZ /	%A (MPM) EXP'D: %06% REC'D: %06/
2062	013336	045	101	040	XRRERA: .ASCIIZ /	%A (XRR) EXP'D: %06% REC'D: %06/
2063	013401	045	101	040	XPMERA: .ASCIIZ /	%A (XPM) EXP'D: %06% REC'D: %06/
2064	013444	045	101	040	HBAERA: .ASCIIZ /	%A (HBA) EXP'D: %06% REC'D: %06/
2065	013507	045	101	040	CBAERA: .ASCIIZ /	%A (CBA) EXP'D: %06% REC'D: %06/
2066	013552	040	040	104	ECE: .ASCIIZ /	DIDN'T GET EXPECTED ERROR CODE/
2067						;MORE ERROR MESSAGES --
2068						
2069	013613	040	040	103	CRWRZ: .ASCIIZ	.. CURSOR REGISTER WRITE/READ ZEROS ERROR..
2070	013664	040	040	103	CRWRO: .ASCIIZ	.. CURSOR REGISTER WRITE/READ ONES ERROR..
2071	013734	040	040	103	CRDAX: .ASCIIZ	.. CURSOR REGISTER DUAL ADDRESS - WRITE Y CHANGED X..
2072	014017	040	040	103	CRDAY: .ASCIIZ	.. CURSOR REGISTER DUAL ADDRESS - WRITE X CHANGED Y..
2073	014102	040	040	103	CRWRE: .ASCIIZ	.. CURSOR REGISTER WRITE/READ ERROR..
2074	014145	040	040	112	JSEWRZ: .ASCIIZ	.. JOYSTICK ENABLES WRT/RD 0'S ERROR..
2075	014211	040	040	112	JSEWRO: .ASCIIZ	.. JOYSTICK ENABLES WRT/RD 1'S ERROR..
2076	014255	040	040	112	CEZWE: .ASCIIZ	.. JOYSTICK ENABLES ZERO-THEN-WRITE ERROR..
2077	014326	040	040	112	CESWE: .ASCIIZ	.. JOYSTICK ENABLES SET-THEN-WRITE ERROR..
2078	014376	040	040	112	JENOCLE: .ASCIIZ	.. J.S. CHANNEL DECODE ERROR - ENABLES DIDN'T CLEAR..
2079	014461	040	040	122	CSRGSE: .ASCIIZ	.. REGISTER ERROR AFTER J.S. SWITCH SET, NO ENABLE..
2080	014543	040	040	122	CSRGSE: .ASCIIZ	.. REGISTER ERROR AFTER J.S. SW ENABLE SET, NO SWITCH..
2081	014630	040	040	111	CSIFC: .ASCIIZ	.. INTERNAL REG. ERROR AFTER CLEARING J.S. SW INTR..
2082	014712	040	040	105	NOSWPI: .ASCIIZ	.. ERROR IN FLAGS REG. -- SHOULD SHOW PENDING SW INTR..
2083	014777	040	040	106	NOSWI: .ASCIIZ	.. FAULTY J.S. SWITCH INTERRUPT..
2084	015036	040	040	111	SWIBV: .ASCIIZ	.. INCORRECT VECTOR ON J.S. SWITCH INTERRUPT..
2085	015112	040	040	123	SWNOHLT: .ASCIIZ	.. SWITCH INTR REQ DIDN'T HALT DISPLAY PROCESSING - BAD DPC..
2086	015205	040	040	106	SWNOCU: .ASCIIZ	.. FAULTY CURSOR RETRIEVAL ON J.S. SW INTR REQ..
2087						
2088	015263	040	040	115	NOCMI: .ASCIIZ	.. MATCH INTERRUPT FAULTY OR NOT RECEIVED..
2089	015334	040	040	116	NOMST: .ASCIIZ	.. NO STOP ON MATCH INTERRUPT..
2090	015371	040	040	115	MMNOMI: .ASCIIZ	.. MATCH BUT NO MATCH INTERRUPT..

```

2091 015430      040      040      111 CMIVB: .ASCIZ  '' INCORRECT VECTOR ON CURSOR MATCH INTERRUPT''
2092 015505      040      040      127 MATPCS: .ASCIZ  '' WRONG DPC OR DSR FOR MATCH STOP''
2093 015547      040      040      127 MATXYE: .ASCIZ  '' WRONG DXR OR DYR FOR MATCH STOP''
2094 015611      040      040      127 MATPOS: .ASCIZ  '' WRONG DXR OR DYR AFTER POSITION-RESTORE''
2095 015663      040      040      111 MATIRE: .ASCIZ  '' INTERNAL REGISTER INCORRECT FOR MATCH''
2096 015733      040      040      111 MEREG: .ASCIZ  '' INTERNAL REGISTER INCORRECT AFTER SETUP FOR MATCH''
2097 016017      040      040      106 MATFLE: .ASCIZ  '' FLAGS REGISTER INCORRECT FOR MATCH''
2098 016064      040      040      125 MATUI: .ASCIZ  '' UNEXPECTED INTERRUPT ON RESUME AFTER MATCH''
2099 016141      040      040      104 MENDPCS: .ASCIZ  '' DPC OR DSR WRONG ON AFTER RESUME FOLLOWING MATCH''
2100 016224      040      040      103 CMPE: .ASCIZ  '' CURSOR MATCHED AT WRONG POSITION''
2101 016267      040      040      116 NOMAT: .ASCIZ  '' NO CURSOR MATCH DETECTED''
2102 016322      045      116      045 EXPGT2: .ASCIZ  /%N% EXP'D: %06%A, %06%N% REC'D: %06%A, %06/
2103 016377      045      116      045 FCOORD: .ASCII  /%N% CURSOR (X,Y) - %06%A, %06/
2104 016437      045      116      045 .ASCIZ  /%N% REC'D DXR, DYR = %06%A, %06/
2105 016500      045      116      045 FSYCHAN: .ASCIZ  ''%N% SYNC GEN/JOYSTICK CHANNEL = %01''
2106
2107
2108 016546      000
2109 016547      045      116      000 NUL: .ASCIZ  //
2110 016552      040      040      122 NULCR: .ASCIZ  /%N/
2111 016613      040      040      122 RLXIE: .ASCIZ  / RUN-LENGTH X-INCREMENT FAILURE/
2112 016663      040      040      122 RLX2IE: .ASCIZ  / RUN-LENGTH X-DOUBLE INCREMENT FAILURE/
2113 016715      040      040      122 RLYUE: .ASCIZ  / RUN-LENGTH Y-UP FAILURE/
2114 016751      040      040      122 RLYDE: .ASCIZ  / RUN-LENGTH Y-DOWN FAILURE/
2115 017010      040      040      122 RLY2UE: .ASCIZ  / RUN-LENGTH Y-SKIP UP FAILURE/
2116 017051      045      101      040 RLY2DE: .ASCIZ  / RUN-LENGTH Y-SKIP DOWN FAILURE/
2117 017105      045      101      040 EXPGT: .ASCIZ  /%A EXP'D: %06%A, REC'D: %06/
2118 017207      040      040      115 DUAD12: .ASCIZ  /%A REG(W) WRITTEN TO: %06%A REG(R) READ; EXP'D: %06%A, REC'D: %06/
2119 017235      045      101      040 INLE: .ASCIZ  / MEM INTERLACE ERROR/
2120 017326      040      040      120 INLEXF: .ASCIZ  /%A PIXEL READBACK EXP'D: %06%A, REC'D: %06%A, CHAN: %01/
2121 017355      040      040      123 SCPRE: .ASCIZ  / PIXEL READBACK ERROR/
2122 017416      040      040      103 SCIDE: .ASCIZ  / SYNC CHAN INTERLACE DIFFERENCE/
2123 017461      045      101      040 SCME: .ASCIZ  / CONFIG DOESN'T MATCH MFG. MASTER/
2124 017534      045      116      045 SCMFG: .ASCIZ  /%A MFG MEM 0: %06%A, MFG SYNC CHAN 0: %06/
2125 017552      045      101      040 SCM: .ASCII  /%N% MEMORYS:/
2126 017565      045      101      054 .ASCII  /%A 0 = %06/
2127 017601      045      101      054 .ASCII  /%A, 1 = %06/
2128 017615      045      101      054 .ASCII  /%A, 2 = %06/
2129 017632      045      116      045 .ASCII  /%A, 3 = %06/
2130 017653      045      101      040 SCS: .ASCII  /%N% SYNC CHANS:/
2131 017666      045      101      054 .ASCII  /%A 0 = %06/
2132 017702      045      101      054 .ASCII  /%A, 1 = %06/
2133 017716      045      101      054 .ASCII  /%A, 2 = %06/
2134 017735      045      116      045 .ASCII  /%A, 3 = %06%N/
2135 .EVEN  /%N%**** MANUAL INTERVENTION NOT ALLOWED -- ABORTING ****%N/

```

```

2137      .SBTTL  GLOBAL ERROR REPORT SECTION
2138
2139      :++
2140      : THE GLOBAL ERROR REPORT SECTION CONTAINS THE PRINTB AND PRINTX
2141      : CALLS THAT ARE USED IN MORE THAN ONE TEST.
2142      : ASCII TEXT STRINGS ARE FOUND IN THE GLOBAL TEXT SECTION.
2143      :--
2144
2145 020032      BGNMSG  NXRERR      ;NON-EXISTANT DEVICE REGISTER.
2146 020032      PRINTX  #NXRX,NODEV ;NODEV = NEXM ADDRESS.
2147 020056 004737 022276      JSR    PC,EXTEND ; PRINT EXTENSION IF REQUIRED.
2148 020062      ENDMSG
2149
2150 020064      BGNMSG  PCSERR      ;BAD DPC AND/OR DSR SIGNATURE.
2151 020064      PRINTX  #PCSX,R2,R3,@DPC,@DSR ;R2,R3 = EXP PC,SR
2152 020120 004737 022276      JSR    PC,EXTEND ; PRINT EXTENSION IF REQUIRED.
2153 020124      ENDMSG
2154
2155 020126      BGNMSG  XYERR      ;BAD DXR AND/OR DYR SIGNATURE.
2156 020126      PRINTX  #XYX,R4,R5,@DXR,@DYR ;R4,R5 = EXP X,Y
2157 020162 004737 022276      JSR    PC,EXTEND ; PRINT EXTENSION IF REQUIRED.
2158 020166      ENDMSG
2159
2160 020170      BGNMSG  RLXYE      ;BAD DXR AND/OR DYR SIGNATURE.
2161 020170      PRINTX  #XYX,R1,R3,R2,R4 ; R1,R3 = EXP, R2,R4 = REC.
2162 020220 004737 022276      JSR    PC,EXTEND ; PRINT EXTENSION IF REQUIRED.
2163 020224      ENDMSG
2164
2165 020226      BGNMSG  INTERR      ;DISPLAY INTERRUPT ERROR SIGNATURE.
2166 020226      PRINTX  #INTX,R1,@DPC ;R1 = CPU PC
2167 020254      PRINTX  #NULCR      ; PRINT A NEWLINE
2168 020274 122737 000377 026107      CMPB  #^0377,INTFLAG ;WAS ANY INTERRUPT GENERATED?
2169 020302 001011      BNE  1$      ;BR IF YES
2170 020304      PRINTX  #FNOINTR      ; NO -- REPORT THE FACT
2171 020324 000465      BR  20$
2172 020326 132737 000001 026107 1$:  BITB  #I0KSTP,INTFLAG ; WAS IT AN UNEXPECTED 'STOP' INTERRUPT?
2173 020334 001411      BEQ  2$      ; BR IF NO.
2174 020336      PRINTX  #FUSI      ; YES -- REPORT IT
2175 020356 000450      BR  20$
2176 020360 132737 000002 026107 2$:  BITB  #I0KJSM,INTFLAG ; WAS IT AN UNEXPECTED 'MATCH' INTERRUPT?
2177 020366 001411      BEQ  3$      ; BR IF NO.
2178 020370      PRINTX  #FUMI      ; REPORT IF YES.
2179 020410 000433      BR  20$
2180 020412 132737 000010 026107 3$:  BITB  #I0KJSS,INTFLAG ; WAS IT AN UNEXPECTED 'SWITCH' INTERRUPT?
2181 020420 001411      BEQ  4$      ; BR IF NO
2182 020422      PRINTX  #FUSWI      ; REPORT IF YES
2183 020442 000416      BR  20$
2184 020444 132737 000004 026107 4$:  BITB  #I0KERR,INTFLAG ; WAS IT AN UNEXPECTED 'ERROR' INTERRUPT?
2185 020452 001412      BEQ  20$
2186 020454      PRINTX  #FUTO      ; REPORT IT.
2187 020474 004737 023516      JSR    PC,ERICHK ;CHECK THE ERROR CODE
2188 020500 004737 022276      JSR    PC,EXTEND ; PRINT EXTENSION IF REQUIRED.
2189 020504      ENDMSG
2190
2191 020506      BGNMSG  IMDERR      ;IMAGE MEM DATA ERROR.
2192 020506 010137 072350      MOV  R1,IMCRAM ;TRANSLATE X ADDR...
2193 020512 006237 072350      ASR  IMCRAM ;...TO RAM NUMBER.

```

2194	020516		PRINTX	#IMDIX,R1,R2,PDF,R3,R4	
2195	020552		PRINTX	#IMDIX2,IMCRAM,IMC.CH	
2196	020602	004737 022276	JSR	PC,EXTEND	; PRINT EXTENSION IF REQUIRED.
2197	020606		ENDMSG		
2198					
2199	020610		BGNMSG	LINTER	; LUT INTERRUPT ERROR.
2200	020610		PRINTX	#LINTX,@LSR	; SHOW CSR.
2201	020634	004737 022276	JSR	PC,EXTEND	; PRINT EXTENSION IF REQUIRED.
2202	020640		ENDMSG		
2203					
2204	020642		BGNMSG	LRWERR	; LUT READ/WRITE ERROR.
2205	020642		PRINTX	#LRWX,@LSR,R1,R2	; CSR, EXP'D, REC'D.
2206	020672	004737 022276	JSR	PC,EXTEND	; PRINT EXTENSION IF REQUIRED.
2207	020676		ENDMSG		
2208					
2209	020700		BGNMSG	EXPREC	
2210	020700		PRINTX	#EXPGOT,R1,R2	; R1=EXP'D, R2=REC'D.
2211	020724	004737 022276	JSR	PC,EXTEND	; PRINT EXTENSION IF REQUIRED.
2212	020730		ENDMSG		
2213					
2214	020732		BGNMSG	DPCER	
2215	020732		PRINTX	#DPCERA,R1,R2	; R1=EXPECTED, R2=RECEIVED.
2216	020756	004737 022276	JSR	PC,EXTEND	; PRINT EXTENSION IF REQ'D.
2217	020762		ENDMSG		
2218					
2219	020764		BGNMSG	DSRER	
2220	020764		PRINTX	#DSRERA,R1,R2	; R1=EXPECTED, R2=RECEIVED.
2221	021010	004737 022276	JSR	PC,EXTEND	; PRINT EXTENSION IF REQ'D.
2222	021014		ENDMSG		
2223					
2224	021016		BGNMSG	DXRER	
2225	021016		PRINTX	#DXRERA,R1,R2	; R1=EXPECTED, R2=RECEIVED.
2226	021042	004737 022276	JSR	PC,EXTEND	; PRINT EXTENSION IF REQ'D.
2227	021046		ENDMSG		
2228					
2229	021050		BGNMSG	DYRER	
2230	021050		PRINTX	#DYRERA,R1,R2	; R1=EXPECTED, R2=RECEIVED.
2231	021074	004737 022276	JSR	PC,EXTEND	; PRINT EXTENSION IF REQ'D.
2232	021100		ENDMSG		
2233					
2234	021102		BGNMSG	PCSER	
2235	021102		PRINTX	#PCSERA,R1,R2	; R1=EXPECTED, R2=RECEIVED.
2236	021126	004737 022276	JSR	PC,EXTEND	; PRINT EXTENSION IF REQ'D.
2237	021132		ENDMSG		
2238					
2239	021134		BGNMSG	FLGER	
2240	021134		PRINTX	#FLGERA,R1,R2	; R1=EXPECTED, R2=RECEIVED.
2241	021160	004737 022276	JSR	PC,EXTEND	; PRINT EXTENSION IF REQ'D.
2242	021164		ENDMSG		
2243					
2244	021166		BGNMSG	CSRER	
2245	021166		PRINTX	#CSRERA,R1,R2	; R1=EXPECTED, R2=RECEIVED.
2246	021212	004737 022276	JSR	PC,EXTEND	; PRINT EXTENSION IF REQ'D.
2247	021216		ENDMSG		
2248					
2249	021220		BGNMSG	MRRER	
2250	021220		PRINTX	#MRRERA,R1,R2	; R1=EXPECTED, R2=RECEIVED.

```

2251 021244 004737 022276      JSR      PC,EXTEND      ; PRINT EXTENSION IF REQ'D.
2252 021250      ENDMSG
2253
2254 021252      BGNMSG      MPMER
2255 021252      PRINTX      #MPMERA,R1,R2      ; R1=EXPECTED, R2=RECEIVED.
2256 021276 004737 022276      JSR      PC,EXTEND      ; PRINT EXTENSION IF REQ'D.
2257 021302      ENDMSG
2258
2259 021304      BGNMSG      XRRER
2260 021304      PRINTX      #XRRERA,R1,R2      ; R1=EXPECTED, R2=RECEIVED.
2261 021330 004737 022276      JSR      PC,EXTEND      ; PRINT EXTENSION IF REQ'D.
2262 021334      ENDMSG
2263
2264 021336      BGNMSG      XPMER
2265 021336      PRINTX      #XPMERA,R1,R2      ; R1=EXPECTED, R2=RECEIVED.
2266 021362 004737 022276      JSR      PC,EXTEND      ; PRINT EXTENSION IF REQ'D.
2267 021366      ENDMSG
2268
2269 021370      BGNMSG      HBAER
2270 021370      PRINTX      #HBAERA,R1,R2      ; R1=EXPECTED, R2=RECEIVED.
2271 021414 004737 022276      JSR      PC,EXTEND      ; PRINT EXTENSION IF REQ'D.
2272 021420      ENDMSG
2273
2274 021422      BGNMSG      CBAER
2275 021422      PRINTX      #CBAERA,R1,R2      ; R1=EXPECTED, R2=RECEIVED.
2276 021446 004737 022276      JSR      PC,EXTEND      ; PRINT EXTENSION IF REQ'D.
2277 021452      ENDMSG
2278
2279 021454      BGNMSG      DUADRE
2280 021454      PRINTX      #DUAD12,R3,R1,R2      ; R3=WRITTEN, R1=EXP'D, R2=REC'D.
2281 021502 004737 022276      JSR      PC,EXTEND      ; PRINT EXTENSION IF REQ'D.
2282 021506      ENDMSG
2283
2284 021510      BGNMSG      SYSCON
2285 021510 005737 002512      TST      MFGFLG      ; MFG MODE?
2286 021514 001414      BEQ      1$      ; NO.
2287 021516      PRINTX      #SCMFG,MFGMO,MFGSO
2288 021546      1$:      PRINTX      #SCM,MEMTAB,MEMTAB+2,MEMTAB+4,MEMTAB+6
2289 021606      PRINTX      #SCS,SYCTAB,SYCTAB+2,SYCTAB+4,SYCTAB+6
2290 021646      ENDMSG
2291
2292 021650      BGNMSG      MPMER
2293 021650      PRINTX      #MPMERA,R1,R2
2294 021674      PRINTX      #NULCR
2295 021714      PRINTX      #DPCERA,R1,R2
2296 021740      PRINTX      #NULCR      ;PRINT BLANK LINE
2297 021760      ENDMSG
2298
2299
2300 021762      BGNMSG      XPMER
2301 021762      PRINTX      #XPMERA,R1,R2
2302 022006      PRINTX      #NULCR
2303 022026      PRINTX      #DPCERA,R1,R2
2304 022052      PRINTX      #NULCR      ;PRINT BLANK LINE
2305 022072      ENDMSG
2306
2307

```

```

2308 022074          BGNMSG  EXPRC2
2309 022074          PRINTX  #EXPGT2,R1,R3,R2,R4      ;R1,R3 = EXP'D, R2,R4 = REC'D
2310 022124 004737 022276      JSR    PC,EXTEND      ; PRINT EXTENSION, IF REQ'D.
2311 022130          ENDMSG
2312
2313 022132          BGNMSG  PNTCOOR
2314 022132          PRINTX  #FCOORD,CSMF4X,CSMF4Y,SDXR,SDYR
2315 022172 004737 022276      JSR    PC,EXTEND
2316 022176          ENDMSG
2317
2318 022200          BGNMSG  PNTSYCH
2319 022200          PRINTX  #FSYCHAN,CSCHAN
2320 022224          ENDMSG
2321
2322 022226          BGNMSG  INLEX
2323 022226          PRINTX  #INLEXF,R1,R2,R5
2324 022254          PRINTX  #NULCR                    ;PRINT BLANK LINE
2325 022274          ENDMSG
2326
2327          : THIS ROUTINE APPENDS A UNIQUE EXTENSION (IF REQUIRED)
2328          : TO ANY OF THE ABOVE ERROR SIGNATURES.
2329          :
2330 022276 005727      EXTEND: TST      (PC)+
2331 022300 000000      EXTA:  0              ; 0 = NU EXTENSION.
2332 022302 001402      BEQ      1$
2333 022304 004777 177770      JSR    PC,@EXTA      ; APPEND EXTENSION TEXT.
2334 022310          1$:  PRINTX  #NULCR      ; PRINT A BLANK LINE
2335 022330 000207      RTS    PC

```

```

2337          .SBTTL  INTERNAL REGISTER CHECKER, ERROR CODE HANDLER, DPINIT
2338
2339          ;ROUTINES TO CHECK CONTENTS OF INTERNAL REGISTERS.
2340          ; UPON ENTRY, THE EXPECTED DATA SHOULD BE IN LOCATION 'GDDAT'
2341
2342          ;MACRO TO ASSEMBLE REGISTER CHECK FRONT-END:
2343          .MACRO  CHECK  RGS,RNAM
2344          MOV     #SEL'RGS,-(SP)  ;GET REGISTER-SELECT CODE ONTO THE STACK.
2345          MOV     #99$,-(SP)      ;AND THE ADDRESS OF THE REGISTER-NAME MESSAGE.
2346          JMP     IREGCK          ;THEN GO TO COMMON HANDLER.
2347          99$:  .ASCIZ  /%A 'RNAM'/'
2348          .EVEN
2349          .ENDM  CHECK
2350
2351          DSRCHK: CHECK  DSR,<DSR>          ;CHECK THE DSR
2352          PCSCHK: CHECK  PCS,<PCSAVE>        ;...PCSAVE
2353          FLGCHK: CHECK  FLG,<FLAGS>          ;...FLAGS
2354          CSRCHK: CHECK  CSR,<CSR>            ;...CSR
2355          MMCHK:  CHECK  MRR,<MAIN MEM MGMT>  ;...MAIN-SEG MEM. MGMT.
2356          HBCHK:  CHECK  HBA,<HISTOGRAM BASE> ;...HISTOGRAM BASE
2357          XMCHK:  CHECK  XRR,<AUX MEM MGMT>   ;...AUX-SEG MEM. MGMT.
2358          CBCHK:  CHECK  CBA,<CHARACTER BASE> ;...CHARACTER BASE
2359
2360
2361          022332      040      040      111  RGE:  .ASCIZ  / INCORRECT DATA IN INTERNAL REGISTER .../
2362          022360      045      116      045  GDBAD: .ASCIZ /%N%A EXP'D: %06%A REC'D: %06/
2363          022410      045      116      045  TINERR: .ASCIZ /%N%A TEST: /
2364          022440      045      116      045  SIMSG:  .ASCIZ /%N%A ... AFTER DOING SOFT INIT./
2365          .EVEN
2366
2367          023054      000000
2368          023056      000000
2369          023060      000000
2370          023062      000000
2371          023064      000000
2372
2373          ;COME HERE WITH (SP)=ADDRESS OF NAME STRING, 2(SP)=SELECT CODE.
2374          023066      012657      023062  IREGCK: MOV     (SP)+,REGNAM      ;GET STRING ADDRESS.
2375          023072      012677      157736      MOV     (SP)+,@DSR        ;SELECT THE REGISTER.
2376          023076      017737      157732      023060      MOV     @DSR,BADDAT      ;GET THE DATA.
2377          023104      012777      000000      157722      MOV     #SELDJR,@DSR      ;SELECT THE REAL DSR
2378          023112      023737      023056      023060      CMP     GDDAT,BADDAT      ;COMPARE EXPECTED W/ ACTUAL.
2379          023120      001410
2380          023122
2381          023142      000207      1$:  BEQ     1$          ;OK IF EQUAL; JUST EXIT.
2382          HRDERR      RGE,REGERR          ;IF NOT, SAY SO & PRINT DATA.
2383          RTS     PC
2384
2385          BGNMSG      REGERR
2386          PRINTX      REGNAM              ;PRINT THE REGISTER NAME.
2387          PRINTX      #GDBAD,GDDAT,BADDAT ;AND THE DATA.
2388          PRINTX      #TINERR
2389          PRINTX      TNAM                ;PRINT TEST NAME
2390          TST         SIFLA              ;SOFT INIT?
2391          BEQ         2$                ;BR IF NO
2392          PRINTX      #SIMSG              ;PRINT IF YES
2393          PRINTX      #NULCR              ;PRINT BLANK LINE
2394          ENDMSG

```

```

2394 023324
2395 023324
2396 023344
2397 023364
2398 023404 000207
2399
2400
2401
2402
2403
2404
2405
2406 023406 012737 100000 023056 IIRCNC: MOV #CHAR,GDDAT ;SEE IF FLAGS HAS 'CHAR' MODE ONLY.
2407 023414 004737 022410 JSR PC,FLGCHK
2408 023420 000414 BR IIRNFC ;BYPASS CSR CHECK.
2409 023422 012737 100000 023056 IIRCHK: MOV #CHAR,GDDAT ;SEE IF FLAGS HAS 'CHAR' MODE ONLY.
2410 023430 004737 022410 JSR PC,FLGCHK
2411 023434 004737 023524 IIRCNC: JSR PC,ERRCHK ;SPECIFICALLY CHECK FOR AN ERROR CODE.
2412 023440 012737 000003 023056 MOV #SELCR,GDDAT ;WE SHOULD ONLY SEE SELECT CODE IN CSR.
2413 023446 004737 022440 JSR PC,CSRCHK
2414 023452 012737 000001 023056 IIRNFC: MOV #1,GDDAT ;EXPECT TO SEE 'EMPTY' STATUS IN PCSAVE.
2415 023460 004737 022360 JSR PC,PCSRCHK ;...GO CHECK IT.
2416 023464 005037 023056 CLR GDDAT ;REMAINING REGISTERS S/B CLEAR.
2417 023470 004737 022466 JSR PC,MMCHK ;...MAIN MEM MGMT
2418 023474 004737 022526 JSR PC,HCHK ;...HISTOGRAM BASE
2419 023500 004737 022566 JSR PC,XMCHK ;...AUX MEM MGMT
2420 023504 004737 022624 JSR PC,CBCHK ;...CHARACTER BASE.
2421 023510 005037 023064 CLR SIFLAG ;CLEAR SOFT-INIT FLAG.
2422 023514 000207 RTS PC
2423
2424
2425 023516 012746 000001
2426 023522 000401
2427 023524 005046
2428 023526 012777 000003 157300
2429 023534 017746 157274
2430 023540 012777 000000 157266
2431 023546 011637 023060
2432 023552 042716 177760
2433 023556 022627 000003
2434 023562 001424
2435 023564 005716
2436 023566 001011
2437 023570
2438 023610 000466
2439 023612
2440 023632 000455
2441
2442 023634 005737 023060
2443 023640 100052
2444 023642 000337 023060
2445 023646 006237 023060
2446 023652 006237 023060
2447 023656 042737 177741 023060
2448 023664 013746 023060
2449 023670 062716 024266
2450 023674 013637 024266

PTINER: PRINTX #TINERR ;PRINT 'TEST:'
PRINTX TNAM ;PRINT NAME OF TEST
PRINTX #NULCR ;PRINT BLANK LINE
RTS PC

;ROUTINES TO CHECK FOR THE INITIALIZED STATE OF INTERNAL REGISTERS:
; IIRCHK CHECKS THEM ALL.
; IIRCNC CHECKS ALL BUT THE FLAGS REGISTER.
; IIRCNC CHECKS ALL BUT THE CSR
; IIRNFC CHECKS ALL BUT FLAGS AND CSR.

ERRCHK: CHECKS THE CSR FOR AN ERROR CODE & REPORTS IT.
ERRCHK: MOV #1,-(SP) ; SET FLAG TO INHIBIT USE OF 'DFERR'
BR ERRCK1
ERRCHK: CLR -(SP) ; CLEAR FLAG
ERRCK1: MOV #SELCR,@DSR ;SELECT THE CSR.
MOV @DSR,-(SP) ;READ IT.
MOV #SELDNR,@DSR ;SELECT REAL DSR SO PEOPLE WON'T GET MIXED UP..
MOV (SP),BADDAT ;STORE CSR CONTENTS.
BIC #C17,(SP) ;EXTRACT THE SELECT.WE FIELD.
CMP (SP)+,#SELCR ;WE SHOULD SEE THE CSR REG SELECT CODE.
BEQ 1$
TST (SP) ; SHOULD WE DO DFERR?
BNE 20$ ; BR IF NO
DFERR NOCSR ;IF NOT, SAY WE CAN'T READ IT.
BR 2$ ; GO TO EXIT
20$: PRINTX #FNOCNR ; PRINT MESSAGE 'CAN'T READ CSR'
BR 2$ ; GO TO EXIT

1$: TST BADDAT ;TEST THE ERROR BIT.
BPL 2$ ;OK IF CLEAR; JUST EXIT.
SWAB BADDAT ;IF SET, GET ERROR CODE
ASR BADDAT
ASR BADDAT ;...INTO BITS<4:1>.
BIC #C36,BADDAT ;ZAP OUT JUNK.
MOV BADDAT,-(SP) ;SAVE ERROR CODE * 2 ON STACK
ADD #ERTBL,(SP) ;GET TO THE MESSAGE ADDRESS TABLE.
MOV @((SP)+,ERCNAM ;GET THE ADDRESS OF THE REG NAME MESSAGE.

```

```

2451 023700 005716      TST      (SP)      ; USE 'DFERR'?
2452 023702 001011      BNE      11$      ; BR IF NO
2453 023704      HRDERR  ERCM,ERCOD      ;REPORT THE BAD NEWS.
2454 023724 000420      BR      2$
2455 023726      11$:    PRINTX  #FERCM      ; PRINT 'CSR ERROR CODE DETECTED ='
2456 023746      PRINTX  ERCNAM      ; ... AND THE ERROR NAME.
2457 023766 005726      2$:    TST      (SP)+
2458 023770 000207      RTS      PC

2459
2460 023772      045      116      045  FNOC SR: .ASCII  /%N%A  ***/
2461 024003      040      040      103  NOCSR: .ASCIIZ / CAN'T READ THE CSR/
2462 024030      045      116      045  FERCM: .ASCII  /%N%A  ***/
2463 024041      040      040      103  ERCM:  .ASCIIZ / CSR ERROR CODE REC'D - /
2464
2465 024074      BGNMSG  ERCD
2466 024074      PRINTX  ERCNAM
2467 024114 023727 023060 000026      CMP      BADDAT,#13*2      ; IS IT 'DBUS DATA R/W ERROR'?
2468 024122 001014      BNE      1$      ;BR IF NO
2469 024124      PRINTX  #GDBAD,@DXR,@DYR      ; YES -- DXR - EXP'D, DYR - REC'D
2470 024154      1$:    PRINTX  #TINERR
2471 024174      PRINTX  TNAM
2472 024214 005737 023064      TST      SIFLAG      ;SOFT INIT?
2473 024220 001410      BEQ      2$      ;BR IF NO
2474 024222      PRINTX  #SIMSG      ;PRINT IF YES
2475 024242      2$:    PRINTX  #NULCR      ;PRINT BLANK LINE
2476 024262      ENDMMSG
2477 024264 000000      ERCNAM: 0
2478 024266 024326 024365 024430      ERCTBL: 77$,1$,2$,3$,4$,5$,6$,7$
2479 024306 024735 025004 025067      10$,11$,12$,13$,14$,15$,16$,17$

2480
2481      ;ERROR CODE EXPLANATIONS:
2482 024326      045      101      056  77$: .ASCIIZ /%A... 0: (ILLEGAL ERROR CODE)/
2483 024365      045      101      056  1$: .ASCIIZ /%A... 1: NXM TIMEOUT ON DMA X'FER/
2484 024430      045      101      056  2$: .ASCIIZ /%A... 2: (ILLEGAL ERROR CODE)/
2485 024467      045      101      056  3$: .ASCIIZ /%A... 3: MEMORY PROTECTION ERROR/
2486 024531      045      101      056  4$: .ASCIIZ /%A... 4: SEQUENCE ERROR/
2487 024562      045      101      056  5$: .ASCIIZ /%A... 5: RESERVED OPERATION/
2488 024617      045      101      056  6$: .ASCIIZ /%A... 6: VECTOR ENDPOINT MISMATCH/
2489 024662      045      101      056  7$: .ASCIIZ /%A... 7: U-PROC EXECUTED UNUSED MICROWORD/
2490 024735      045      101      056 10$: .ASCIIZ /%A... 10: IMAGE MEMORY 'SYNC' TIMEOUT/
2491 025004      045      101      056 11$: .ASCIIZ /%A... 11: PIXEL READBACK 'DATA AVAILABLE' TIMEOUT/
2492 025067      045      101      056 12$: .ASCIIZ /%A... 12: J.S. STATUS 'DATA READY' TIMEOUT/
2493 025143      045      101      056 13$: .ASCIIZ /%A... 13: DBUS DATA READ-WRITE ERROR/
2494 025211      045      101      056 14$: .ASCIIZ /%A... 14: DBUS SIGNAL HUNG/
2495 025245      045      101      056 15$: .ASCIIZ /%A... 15: VBUS SIGNAL HUNG/
2496 025301      045      101      056 16$: .ASCIIZ /%A... 16: (ILLEGAL ERROR CODE)/
2497 025341      045      101      056 17$: .ASCIIZ /%A... 17: (ILLEGAL ERROR CODE)/
2498      .EVEN
2499
2500      ;ROUTINE TO DO A 'SOFT' INIT ON THE DPU:
2501      .ENABL  LSB
2502 025402 004737 031412      DPINIT: JSR      PC,SAVERS      ; SAVE R0-R5.
2503 025406 012737 000001 023064      MOV      #1,SIFLAG      ; SAY DOING SOFT INIT
2504 025414 012777 000000 155412      MOV      #0,@DSR      ;SELECT THE REAL DSR
2505 025422 005777 155406      TST      @DSR      ;ARE WE STOPPED?
2506 025426 100423      BMI      11$      ;BR IF YES
2507 025430 004737 026756      JSR      PC,RELEAS      ;SEE IF WE'RE HUNG.

```



```

2566 .SBTTL GLOBAL SUBROUTINES SECTION
2567
2568 :++
2569 : THE GLOBAL SUBROUTINES SECTION CONTAINS THE SUBROUTINES
2570 : THAT ARE USED IN MORE THAN ONE TEST.
2571 :--
2572
2573 :
2574 : DEFAULT DISPLAY INTERRUPT HANDLERS.
2575 : IF DISPLAY TIME-OUT, REPORT DEV FATAL, AND ABORT PASS.
2576 : OTHERWISE, SAVE DPU REGISTERS AND DISMISS.
2577 :
2578
2579 : BIT DEFINITIONS FOR 'INTMASK' AND 'INTFLAG' BYTES:
2580 :
2581 : IOKCKIN=BIT7 : DON'T CHECK FOR BAD INTERRUPTS -- TEST WILL.
2582 : IOKSTP=BIT0 : EXPECT 'STOP' INTERRUPT.
2583 : IOKJSM=BIT1 : EXPECT JOYSTICK 'MATCH' INTERRUPT.
2584 : IOKERR=BIT2 : EXPECT 'ERROR' INTERRUPT.
2585 : IOKJSS=BIT3 : EXPECT 'JOYSTICK SWITCH' INTERRUPT.
2586 :
2587 : INTERRUPT MASK -- SAYS EXPECTING INTERRUPTS
2588 026106 000 INTMASK: .BYTE 0
2589 : INTERRUPT FLAG -- SAYS WE GOT ONE (IF POSITIVE)
2590 026107 000 INTFLAG: .BYTE 0
2591
2592 : SAVED INTERRUPT VECTOR:
2593 026110 000000 INTVEC: .WORD 0
2594 : SAVE CPU PC
2595 026112 000000 INTCPC: .WORD 0
2596
2597 : SUBROUTINE TO ENABLE INTERRUPTS:
2598 026114 010046 ENAINT: MOV R0, -(SP) ; SAVE R0
2599 026116 013700 003052 MOV STPV, R0 ; GET POINTER TO VECTORS
2600 026122 012720 026242 MOV #DSTP, (R0)+ ; SET UP STOP VECTOR
2601 026126 012720 000340 MOV #PRI07, (R0)+ ;
2602 026132 012720 026274 MOV #DJM, (R0)+ ; SET UP MATCH VECTOR
2603 026136 012720 000340 MOV #PRI07, (R0)+ ;
2604 026142 012720 026210 MOV #DIO, (R0)+ ; SET UP ERROR VECTOR
2605 026146 012720 000340 MOV #PRI07, (R0)+ ;
2606 026152 012720 026326 MCV #DJS, (R0)+ ; SET UP JOYSTICK SWITCH VECTOR
2607 026156 012710 000340 MOV #PRI07, (R0)
2608 026162 012600 MOV (SP)+, R0 ; RESTORE R0
2609 026164 011646 MOV (SP), -(SP)
2610 026166 012766 000000 000002 MOV #0, 2(SP) ; SET CPU TO LEVEL 0
2611 026174 000002 RTI
2612
2613 : SUBROUTINE TO DISABLE INTERRUPTS (RAISE PRIORITY TO LEVEL 7)
2614 026176 011646 DSBINT: MOV (SP), -(SP)
2615 026200 012766 000340 000002 MOV #PRI07, 2(SP)
2616 026206 000002 RTI
2617
2618 : ERROR (TIME-OUT) INTERRUPT SERVICE:
2619 026210 013737 003056 026110 DTO: MOV TOTV, INTVEC ; SAVE THE VECTOR
2620 026216 105037 026107 CLRB INTFLAG ; CLEAR FLAG TO SAY WE GOT INTERRUPT
2621 026222 132737 000004 026106 BITB #IOKERR, INTMASK ; EXPECTING ERROR INTERRUPT?
2622 026230 001053 BNE DPUSAV ; BR IF YES; NO ERROR.

```

```

2623 026232 152737 000004 026107      BISB  #IOKERR,INTFLAG ;NO. SET THE ERROR FLAG.
2624 026240 000447      BR      DPUSAV      ;GO FINISH UP.
2625
2626 026242 013737 003052 026110 DSTP: MOV    STPV,INTVEC  ;SAVE THE VECTOR
2627 026250 105037 026107      CLRB  INTFLAG    ;CLEAR FLAG TO SAY WE GOT INTERRUPT
2628 026254 132737 000001 026106      BITB  #IOKSTP,INTMASK ;EXPECTING STOP INTERRUPT?
2629 026262 001036      BNE     DPUSAV      ;BR IF YES
2630 026264 152737 000001 026107      BISB  #IOKSTP,INTFLAG ;NO. SET THE ERROR FLAG.
2631 026272 000432      BR      DPUSAV      ;GO FINISH UP.
2632
2633 026274 013737 003054 026110 DJM:  MOV    JSMV,INTVEC  ;SAVE THE VECTOR
2634 026302 105037 026107      CLRB  INTFLAG    ;CLEAR FLAG TO SAY WE GOT INTERRUPT
2635 026306 132737 000002 026106      BITB  #IOKJSM,INTMASK ;EXPECTING MATCH INTERRUPT?
2636 026314 001021      BNE     DPUSAV      ;BR IF YES
2637 026316 152737 000002 026107      BISB  #IOKJSM,INTFLAG ;NO. SET THE ERROR FLAG.
2638 026324 000415      BR      DPUSAV      ;GO FINISH UP.
2639
2640 026326 013737 003060 026110 DJS:  MOV    JSSV,INTVEC  ;SAVE THE VECTOR
2641 026334 105037 026107      CLRB  INTFLAG    ;CLEAR FLAG TO SAY WE GOT INTERRUPT
2642 026340 132737 000010 026106      BITB  #IOKJSS,INTMASK ;EXPECTING SWITCH INTERRUPT?
2643 026346 001004      BNE     DPUSAV      ;BR IF YES
2644 026350 152737 000010 026107      BISB  #IOKJSS,INTFLAG ;NO. SET THE ERROR FLAG.
2645 026356 000400      BR      DPUSAV      ;GO FINISH UP.
2646
2647
2648 026360 011637 026112      DPUSAV: MOV    (SP),INTPC   ;SAVE CPU PC.
2649 026364 010146      MOV    R1,-(SP)    ;SAVE R1
2650 026366 013701 003034      MOV    DSR,R1      ;GET DSR ADDRESS INTO R1
2651 026372 017737 154434 003126      MOV    @DPC,IDPC    ;SAVE DISPLAY PC...
2652 026400 011137 003130      MOV    (R1),IDSR    ;...STATUS...
2653 026404 017737 154426 003132      MOV    @DXR,IDX    ;...X POSITION...
2654 026412 017737 154422 003134      MOV    @DYR,IDY    ;...Y POSITION...
2655 026420 012711 000001      MOV    #SELP,CS,(R1) ;SAVE PCSAVE...
2656 026424 011137 003146      MOV    (R1),IPCSAV    ;
2657 026430 012711 000002      MOV    #SELF,FLG,(R1) ;SAVE FLAGS...
2658 026434 011137 003150      MOV    (R1),IFLAGS    ;
2659 026440 012711 000003      MOV    #SEL,CSR,(R1) ;SAVE CSR...
2660 026444 011137 003152      MOV    (R1),ICSR     ;
2661 026450 012711 000004      MOV    #SELM,RR,(R1) ;SAVE MAIN MEM-MGMT..
2662 026454 011137 003154      MOV    (R1),IMAIN     ;
2663 026460 012711 000005      MOV    #SEL,HBA,(R1) ;SAVE HISTO. BASE...
2664 026464 011137 003156      MOV    (R1),IHBASE    ;
2665 026470 012711 000006      MOV    #SEL,XRR,(R1) ;SAVE AUX. MEM-MGMT..
2666 026474 011137 003160      MOV    (R1),IAUX      ;
2667 026500 012711 000007      MOV    #SEL,CBA,(R1) ;SAVE CHAR. BASE...
2668 026504 011137 003162      MOV    (R1),ICBASE    ;
2669 026510 012711 000000      MOV    #SEL,DSR,(R1) ;END UP AT DSR...
2670 026514 011137 003144      MOV    (R1),ISDSR    ;...AND SAVE IT.
2671      BITB  #IOKERR,INTFLAG ;DID WE GET AN ERROR INTERRUPT?
2672 026520 105737 026107      TSTB  INTFLAG    ;ANY ERROR FLAGS SET?
2673 026524 001424      BEQ     1$        ;BR IF NO
2674 026526 105737 026106      TSTB  INTMASK    ;YES. IS CHECKING TURNED OFF?
2675 026532 100421      BMI     1$        ;BR IF YES
2676 026534 013746 022300      MOV    EXTA,-(SP)    ;SAVE ERROR EXTENSION ADDRESS.
2677 026540 012737 023324 022300      MOV    #PTINER,EXTA  ;SET EXTENSION TO PRINT TEST IN ERROR.
2678 026546 013701 026112      MOV    INTCP,R1    ;GET CPU PC.
2679 026552 004737 030216      JSR    PC,INCERK    ;INCREMENT THE ERROR COUNTS

```

```

2680 026556      ERRDF 0,IFAU,INTERR ;REPORT ERROR...
2681 026566 012637 022300      MOV (SP)+,EXTA ;RESTORE EXTA
2682 026572 004737 030254      JSR PC,CKEMAX
2683
2684 026576 012601      1$: MOV (SP)+,R1 ;RESTORE R1
2685 026600 000002      RTI ;...AND DISMISS.
2686
2687      ; AND A SIMILAR ONE FOR THE LUT INTERRUPTS.
2688
2689 026602 000137      LDUN: JMP @('C)+
2690 026604 026606      LUTSAV
2691 026606 017737 154230 003136 LUTSAV: MOV @LSR,ILSR ; SAVE LUT REGISTERS.
2692 026614 017737 154224 003140      MOV @LDR,ILDR
2693 026622 017737 154220 003142      MOV @LMR,ILMR
2694 026630 000002      RTI ; AND DISMISS.

```

```

2696      : SUBROUTINE TO INITIALIZE THE CURRENT UNIT DISPLAY.
2697      : CALL 'CLR' TO INIT WITH IM'S IN WRITE MODE.
2698      : CALL 'CLRW' TO INIT WITH IM'S IN WRITE-RETRACE MODE.
2699      : CALL 'CLRRW' TO INIT WITH IM'S IN WRITE-RETRACE, 1'S-ONLY MODE.
2700      :
2701      :
2702      :
2703      :
2704      :
2705      :
2706      :
2707      :
2708      :
2709      :
2710      :
2711      :
2712      :
2713      :
2714      :
2715      :
2716      :
2717      :
2718      :
2719      :
2720      :
2721      :
2722      :
2723      :
2724      :
2725      :
2726      :
2727      :
2728      :
2729      :
2730      :
2731      :
2732      :
2733      :
2734      :
2735      :
2736      :
2737      :
2738      :
2739      :
2740      :
2741      :
2742      :
2743      :
2744      :
2745      :
2746      :
2747      :
2748      :
2749      :
2750      :
2751      :
2752      :
2753      :
2754      :
2755      :
2756      :
2757      :
2758      :
2759      :
2760      :
2761      :
2762      :
2763      :
2764      :
2765      :
2766      :
2767      :
2768      :
2769      :
2770      :
2771      :
2772      :
2773      :
2774      :
2775      :
2776      :
2777      :
2778      :
2779      :
2780      :
2781      :
2782      :
2783      :
2784      :
2785      :
2786      :
2787      :
2788      :
2789      :
2790      :
2791      :
2792      :
2793      :
2794      :
2795      :
2796      :
2797      :
2798      :
2799      :
2800      :
2801      :
2802      :
2803      :
2804      :
2805      :
2806      :
2807      :
2808      :
2809      :
2810      :
2811      :
2812      :
2813      :
2814      :
2815      :
2816      :
2817      :
2818      :
2819      :
2820      :
2821      :
2822      :
2823      :
2824      :
2825      :
2826      :
2827      :
2828      :
2829      :
2830      :
2831      :
2832      :
2833      :
2834      :
2835      :
2836      :
2837      :
2838      :
2839      :
2840      :
2841      :
2842      :
2843      :
2844      :
2845      :
2846      :
2847      :
2848      :
2849      :
2850      :
2851      :
2852      :
2853      :
2854      :
2855      :
2856      :
2857      :
2858      :
2859      :
2860      :
2861      :
2862      :
2863      :
2864      :
2865      :
2866      :
2867      :
2868      :
2869      :
2870      :
2871      :
2872      :
2873      :
2874      :
2875      :
2876      :
2877      :
2878      :
2879      :
2880      :
2881      :
2882      :
2883      :
2884      :
2885      :
2886      :
2887      :
2888      :
2889      :
2890      :
2891      :
2892      :
2893      :
2894      :
2895      :
2896      :
2897      :
2898      :
2899      :
2900      :
2901      :
2902      :
2903      :
2904      :
2905      :
2906      :
2907      :
2908      :
2909      :
2910      :
2911      :
2912      :
2913      :
2914      :
2915      :
2916      :
2917      :
2918      :
2919      :
2920      :
2921      :
2922      :
2923      :
2924      :
2925      :
2926      :
2927      :
2928      :
2929      :
2930      :
2931      :
2932      :
2933      :
2934      :
2935      :
2936      :
2937      :
2938      :
2939      :
2940      :
2941      :
2942      :
2943      :
2944      :
2945      :
2946      :
2947      :
2948      :
2949      :
2950      :
2951      :
2952      :
2953      :
2954      :
2955      :
2956      :
2957      :
2958      :
2959      :
2960      :
2961      :
2962      :
2963      :
2964      :
2965      :
2966      :
2967      :
2968      :
2969      :
2970      :
2971      :
2972      :
2973      :
2974      :
2975      :
2976      :
2977      :
2978      :
2979      :
2980      :
2981      :
2982      :
2983      :
2984      :
2985      :
2986      :
2987      :
2988      :
2989      :
2990      :
2991      :
2992      :
2993      :
2994      :
2995      :
2996      :
2997      :
2998      :
2999      :
3000      :
3001      :
3002      :
3003      :
3004      :
3005      :
3006      :
3007      :
3008      :
3009      :
3010      :
3011      :
3012      :
3013      :
3014      :
3015      :
3016      :
3017      :
3018      :
3019      :
3020      :
3021      :
3022      :
3023      :
3024      :
3025      :
3026      :
3027      :
3028      :
3029      :
3030      :
3031      :
3032      :
3033      :
3034      :
3035      :
3036      :
3037      :
3038      :
3039      :
3040      :
3041      :
3042      :
3043      :
3044      :
3045      :
3046      :
3047      :
3048      :
3049      :
3050      :
3051      :
3052      :
3053      :
3054      :
3055      :
3056      :
3057      :
3058      :
3059      :
3060      :
3061      :
3062      :
3063      :
3064      :
3065      :
3066      :
3067      :
3068      :
3069      :
3070      :
3071      :
3072      :
3073      :
3074      :
3075      :
3076      :
3077      :
3078      :
3079      :
3080      :
3081      :
3082      :
3083      :
3084      :
3085      :
3086      :
3087      :
3088      :
3089      :
3090      :
3091      :
3092      :
3093      :
3094      :
3095      :
3096      :
3097      :
3098      :
3099      :
3100      :
3101      :
3102      :
3103      :
3104      :
3105      :
3106      :
3107      :
3108      :
3109      :
3110      :
3111      :
3112      :
3113      :
3114      :
3115      :
3116      :
3117      :
3118      :
3119      :
3120      :
3121      :
3122      :
3123      :
3124      :
3125      :
3126      :
3127      :
3128      :
3129      :
3130      :
3131      :
3132      :
3133      :
3134      :
3135      :
3136      :
3137      :
3138      :
3139      :
3140      :
3141      :
3142      :
3143      :
3144      :
3145      :
3146      :
3147      :
3148      :
3149      :
3150      :
3151      :
3152      :
3153      :
3154      :
3155      :
3156      :
3157      :
3158      :
3159      :
3160      :
3161      :
3162      :
3163      :
3164      :
3165      :
3166      :
3167      :
3168      :
3169      :
3170      :
3171      :
3172      :
3173      :
3174      :
3175      :
3176      :
3177      :
3178      :
3179      :
3180      :
3181      :
3182      :
3183      :
3184      :
3185      :
3186      :
3187      :
3188      :
3189      :
3190      :
3191      :
3192      :
3193      :
3194      :
3195      :
3196      :
3197      :
3198      :
3199      :
3200      :
3201      :
3202      :
3203      :
3204      :
3205      :
3206      :
3207      :
3208      :
3209      :
3210      :
3211      :
3212      :
3213      :
3214      :
3215      :
3216      :
3217      :
3218      :
3219      :
3220      :
3221      :
3222      :
3223      :
3224      :
3225      :
3226      :
3227      :
3228      :
3229      :
3230      :
3231      :
3232      :
3233      :
3234      :
3235      :
3236      :
3237      :
3238      :
3239      :
3240      :
3241      :
3242      :
3243      :
3244      :
3245      :
3246      :
3247      :
3248      :
3249      :
3250      :
3251      :
3252      :
3253      :
3254      :
3255      :
3256      :
3257      :
3258      :
3259      :
3260      :
3261      :
3262      :
3263      :
3264      :
3265      :
3266      :
3267      :
3268      :
3269      :
3270      :
327
```

2753	027014	010001	FILL1:	MOV	R0,R1	; FILL WITH (R0).
2754	027016	000402		SKP2		
2755	027020	010001	FILL2:	MOV	R0,R1	
2756	027022	005101		COM	R1	; FILL WITH ALTERNATE WORDS.
2757	027024	013702		MOV	FREE,R2	; POINTER.
2758	027030	013703		MOV	FRESIZ,R3	; WORD COUNT.
2759	027034	042703		BIC	#100003,R3	; INSURE WC IS EVEN.
2760	027040	010022	1\$:	MOV	R0,(R2)+	; 1ST WORD...
2761	027042	010122		MOV	R1,(R2)+	; ...NEXT WORD.
2762	027044	162703		SUB	#2,R3	
2763	027050	003373		BGT	1\$	
2764	027052	000207		RTS	PC	

```

2821
2822
2823
2824 027054 012701 107066
2825 027060 012746 000001
2826 027064 000404
2827 027066 012701 110066
2828 027072 012746 000040
2829 027076 000404
2830 027100 012701 110106
2831 027104 012746 000020
2832
2833 027110 005737 003216
2834 027114 001424
2835 027116 005077 153720
2836 027122 052777 000100 153716
2837 027130 011600
2838 027132 012177 153706
2839 027136 005777 153700
2840 027142 100375
2841 027144 105777 153672
2842 027150 001406
2843 027152 005300
2844 027154 001765
2845 027156 016177 177776 153660
2846 027164 000764
2847 027166 005726
2848 027170 000207
2849
2850
2851
2852
2853
2854 027172 005737 003216
2855 027176 001433
2856 027200
2857 027202 005077 153634
2858 027206 052777 000100 153632
2859 027214 042701 177356
2860 027220 005002
2861 027222 012703 000020
2862 027226 010277 153612
2863 027232 005777 153604
2864 027236 100375
2865 027240 013700 003220
2866 027244 005300
2867 027246 001376
2868 027250 105777 153566
2869 027254 001404
2870 027256 005303
2871 027260 001362
2872 027262 060102
2873 027264 000756
2874 027266 000207

; SUBROUTINE TO BLAST THE LUT RAM WITH COLOR OR GREY-SCALE DATA.
LUMIN: MOV #LUMTBL,R1 ; GREY-SCALE TABLE ADDRESS.
      MOV #1,-(SP) ; LUT WORDS / SHADE.
      SKP2+2
NTSC: MOV #NTSC8,R1 ; BASIC 8 COLOR TABLE ADDRESS.
      MOV #32,-(SP) ; LUT WORDS / COLOR.
      SKP2+2
EXPR: MOV #EXPRM,R1 ; EXPERIMENTAL 16 COLOR TABLE.
      MOV #16,-(SP) ; LUT WORDS / COLOR.

      TST LUTAV
      BEQ 3$ ; RETURN IF NO LUT INSTALLED.
      CLR @LSR ; ADDR 0, NO INTERRUPTS.
      BIS #GCOFF,@LMR ; GAMMA CORRECTION OFF.
1$: MOV (SP),R0 ; SET ITERATION K.
      MOV (R1)+,@LDR ; BLAST THE RAM.
2$: ST @LSR ; WAIT FOR READY.
      BPL -4
      TSTB @LSR ; LAST ADDRESS DONE ??
      BEQ 3$ ; EXIT IF SO.
      DEC R0
      BEQ 1$ ; GET NEXT COLOR.
      MOV -2(R1),@LDR ; REPEAT THIS COLOR.
      BR 2$
3$: TST (SP)+ ; FIX THE STACK...
      RTS PC ; ...AND RETURN.

; BLAST 16 SHADES OF THE BASIC COLOR IN (R1).
; LOCATION 'SHADLY' SET BY CALLER TO PRODUCE A PLEASING
; VISUAL EFFECT.
SHAD16: TST LUTAV
      BEQ 3$ ; RETURN IF NO LUT INSTALLED.
      BREAK
      CLR @LSR ; ADDR 0, NO INTERRUPTS.
      BIS #GCOFF,@LMR ; GAMMA CORRECT OFF.
      BIC #^C421,R1 ; SET DIMMEST SHADE THIS COLOR.
      CLR R2 ; 1ST SHADE IS ALWAYS BLACK.
1$: MOV #16,R3 ; 16 LUT WORDS / SHADE.
2$: MOV R2,@LDR ; WRITE DATA.
      TST @LSR
      BPL -4
      MOV SHADLY,R0 ; DELAY.
      DEC R0
      BNE -2
      TSTB @LSR ; LUT RAM FULL ??
      BEQ 3$ ; EXIT IF SO.
      DEC R3 ; THIS SHADE DONE ??
      BNE 2$ ; NOT YET, LOOP.
      ADD R1,R2 ; YES, INCREMENT THE SHADE...
      BR 1$ ; ...AND CONTINUE.
3$: RTS PC

```

```

2876
2877
2878
2879
2880
2881
2882 027270 020027 001000
2883 027274 103003
2884 027276 005400
2885 027300 062700 000100
2886 027304 042700 177600
2887 027310 000207
2888
2889 027312 004737 027270
2890 027316 000300
2891 027320 006200
2892 027322 000207
2893
2894
2895
2896
2897
2898
2899 027324 012737 030060 003232
2900 027332 013737 003232 003234
2901 027340 042701 170000
2902 027344 010102
2903 027346 000302
2904 027350 006202
2905 027352 042702 177770
2906 027356 150237 003232
2907 027362 010102
2908 027364 012700 000006
2909 027370 006202
2910 027372 005300
2911 027374 001375
2912 027376 042702 177770
2913 027402 150237 003233
2914 027406 010102
2915 027410 012700 000003
2916 027414 006202
2917 027416 005300
2918 027420 001375
2919 027422 042702 177770
2920 027426 150237 003234
2921 027432 042701 177770
2922 027436 150137 003235
2923 027442 013701 003232
2924 027446 013702 003234
2925 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.

```

```

2927
2928
2929
2930
2931
2932 027454 012727 005670
2933 027456 005670
2934 027462 005777 153344
2935 027464 100407
2936 027470 004737 027530
2937 027472 005337 027462
2938 027476 001370
2939 027502 000241
2940 027504 000401
2941 027506 000261
2942 027510 000207
2943 027512
2944
2945
2946
2947
2948
2949
2950
2951
2952
2953 027514 012746 023420
2954 027520 000402
2955 027522 012746 011610
2956 027526 000402
2957 027530 012746 000001
2958 027534 011646
2959 027536 010066 000002
2960 027542 013700 027564
2961 027546 005300
2962 027550 001376
2963 027552 005316
2964 027554 001372
2965 027556 005726
2966 027560 012600
2967 027562 000207
2968 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
MOV #3000.,(PC)+ ; DO A SUPVSR BREAK FIRST.
; 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 R0
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 R0, 2(SP) ; SAVE R0
1$: MOV K100US, R0
DEC R0
BNE .-2 ; 100 USEC (MORE OR LESS) LOOP.
DEC (SP)
BNE 1$ ; VARIABLE LOOP.
TST (SP)+ ; FIX STACK...
MOV (SP)+, R0 ; RESTORE R0
RTS PC ;...AND RETURN.
K100US: 8. ; 100 USEC TIMER.

```

```

2970
2971
2972
2973
2974
2975
2976
2977
2978
2979
2980 027566 012737 027620 000004
2981 027574 012737 000200 000006
2982 027602 005003
2983 027604 005711
2984
2985 027606 020102
2986 027610 001407
2987 027612 062701 000002
2988 027616 000772
2989
2990 027620 005103
2991 027622 012716 027630
2992 027626 000002
2993 027630
2994 027636 005703
2995 027640 001401
2996 027642 000261
2997 027644 000207
2998
2999
3000
3001
3002
3003
3004
3005 027646
3006 027646 005737 002510
3007 027652 001006
3008 027654 005737 003030
3009 027660 100403
3010 027662 005337 027700
3011 027666 001002
3012 027670 000241
3013 027672 000401
3014 027674 000261
3015 027676 000207
3016 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$,a#4 ; SET BUSERR VECTOR.
     MOV #PRI04,a#6
     CLR R3 ; FLAG.
1$: 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...
    ; ...AND GIVE BACK THE VECTOR.
3$: CLRVEC #4 ; DID WE CATCH ONE ??
    TST R3 ; NO, 'C' = 0, SKIP NEXT.
    BEQ .+4 ; YES, 'C' = 1, (R1) = NEXM ADDR.
    SEC
    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.

```

```

3018
3019
3020
3021
3022
3023
3024
3025
3026
3027 027702 005037 023064
3028 027706 005037 030130
3029 027712 005037 022300
3030 027716 105037 026106
3031 027722 013700 003026
3032 027726 006300
3033 027730 005737 003226
3034 027734 001430
3035 027736 100010
3036 027740 052760 160000 003302
3037 027746
3038 027756 000407
3039 027760 052760 160001 003302 3$:
3040 027766
3041 027776 012737 177777 003224 2$:
3042 030004
3043 030012
3044 030014 000422
3045
3046 030016
3047 030020 032700 001000
3048
3049
3050 030024 001412
3051 030026
3052 030052 005237 030070
3053 030056 004737 026114
3054 030062 000207
3055 030064 000000
3056 030066 000000
3057 030070 000000
3058
3059
3060
3061
3062 030072
3063 030074 030027 020000
3064 030100 001412
3065 030102
3066 030126 000207
3067
3068 030130 000000
3069 030132 045 101 040
3070 030151 105 122 122
3071
3072
3073
3074

; 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 SI:LAG ; 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$
        BIS #160001,ERTABL(R0) ; FLAG ERROR IN THE ERROR TABLE
        ERRDF 2,NOINIT ; DEVICE NOT IDLE
        MOV #-1,DUFLG ; DROP THE UNIT
        DODU UNITN ; ABORT THE PASS
        DOCLN 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.
    INC TESTK ; BUMP TEST COUNTER.
    JSR PC,ENAINIT ; ENABLE INTERRUPTS
    RTS PC

5$:
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
        BEQ 1$ ; BR IF 'IER' NOT SET.
        PRINTF #ESUM,ERRK ; PRINT ERROR COUNT.
        RTS PC

1$: ERRK: 0 ; LOCAL ERROR COUNT.
    ESUM: .ASCIZ /%A %D%A ERRORS/
    EMAXDO: .ASCIZ /ERROR LIMIT REACHED -- DROPPING UNIT/
           .EVEN

; ROUTINES TO INCREMENT LOCAL ERROR COUNT AND CHECK FOR LIMIT:

```

```

3075 030216 005237 030130      INCERK: INC      ERRK      ; INCREMENT LOCAL ERROR COUNT
3076 030222 010046              MOV      R0,-(SP)    ; SAVE R0
3077 030224 013700 003026      MOV      UNITN,R0    ; GET UNIT NUMBER,
3078 030230 006300              ASL      R0          ; ... AND MAKE IT A WORD OFFSET.
3079 030232 062700 003302      ADD      #ERTABL,R0   ; R0 GETS ADDRESS OF ERROR TABLE ENTRY.
3080 030236 005210              INC      (R0)        ; INCREMENT THE DEVICE ERROR COUNT
3081 030240 032710 007777      BIT      #7777,(R0)   ; DID WE OVERFLOW THE FIELD?
3082 030244 001001              BNE      1$          ; BR IF NO.
3083 030246 005310              DEC      (R0)        ; YES -- BACK IT UP TO 7777.
3084 030250 012600 1$:        MOV      (S^)+,R0      ; RESTORE R0
3085 030252 000207              RTS      PC          ; RETURN TO CALLER.
3086
3087 030254 010046              CKEMAX: MOV      R0,-(SP)    ; SAVE R0
3088 030256 013700 003026      MOV      UNITN,R0    ; GET UNIT NUMBER
3089 030262 006300              ASL      R0          ; ... AND MAKE IT A WORD OFFSET
3090 030264 016000 003302      MOV      ERTABL(R0),R0 ; GET ERROR TABLE ENTRY
3091 030270 042700 170000      BIC      #170000,R0   ; EXTRACT ERROR COUNT FIELD
3092 030274 020037 002516      CMP      R0,GERRMAX   ; IS GLOBAL LIMIT EXCEEDED FOR THIS UNIT?
3093 030300 103004              BHIS     1$          ; BR IF YES
3094 030302 023737 030130 002514  CMP      ERRK,LERRMAX ; IS LOCAL LIMIT EXCEEDED FOR THIS TEST?
3095 030310 103417              BLO      2$          ; BR IF NO
3096 030312              1$:      RFLAGS   R0          ; GET OPERATOR FLAGS
3097 030314 032700 000040      BIT      #IDU,R0      ; IS DROPPING INHIBITED?
3098 030320 001013              BNE      2$          ; BR IF YES.
3099 030322 012737 177777 003224  MOV      #-1,DUFLG   ; NO -- DROP THE UNIT
3100 030330              ERRDF      4,EMAXDU
3101 030340              DODU      UNITN
3102 030346              DOCLN
3103 030350 012600 2$:      MOV      (SP)+,R0      ; RESTORE R0
3104 030352 000207              RTS      PC          ; RETURN TO CALLER
3105
3106 030354 010046              CKDROP: MOV      R0,-(SP)    ;
3107 030356              RFLAGS   R0
3108 030360 032700 000040      BIT      #IDU,R0
3109 030364 001011              BNE      1$
3110 030366 012600              MOV      (SP)+,R0
3111 030370 012737 177777 003224  MOV      #-1,DUFLG
3112 030376              DODU      UNITN
3113 030404              DOCLN
3114 030406 000401              BR      2$
3115 030410 012600 1$:      MOV      (SP)+,R0
3116 030412 000207 2$:      RTS      PC
;ABORT THE PASS

```

```

3118
3119      SUBROUTINE - CONFIGURE VSV11 SYSTEM.
3120
3121      CONFIG:
3122      CONMEM: JSR      PC,DPINIT
3123               MOV      #40000,R0      ; PROTECT ALL MEMORIES
3124      10$:    MOV      R0,@DYR
3125               JSR      PC,WAITF
3126               ADD      #CH1,R0
3127               BIT      #C13,R0
3128               BNE      10$
3129
3130      030446 012777 000000 152362      MOV      #0,@DXR      ; TRY A PIXEL READ.
3131      030454 004737 027454              JSR      PC,WAITF
3132      030460 103410              BCS      1$      ; BR IF UNSUCCESSFUL.
3133      030462              DFERR      PRBH      ; PIXEL READBACK HANGS UP.
3134      030502 012737 144003 023056 1$:  MOV      #144003,GDDAT
3135      030510 004737 022440              JSR      PC,CSRCHK
3136      030514 005005              CLR      R5      ; INIT MEM TABLE PTR.
3137      030516 012777 000013 152310 2$:  MOV      #13,@DSR      ; CLEAR CSR.
3138      030524 005065 003256              CLR      CTABM(R5)      ; CLEAR TABLE ENTRY.
3139      030530 010501              MOV      R5,R1      ; ENABLE MEM R/W.
3140      030532 006201              ASR      R1
3141      030534 000301              SWAB      R1
3142      030536 052701 040060              BIS      #40060,R1
3143      030542 010177 152272              MOV      R1,@DYR
3144      030546 004737 027454              JSR      PC,WAITF
3145      030552 012777 000000 152256      MOV      #0,@DXR      ; PIXEL READ.
3146      030560 004737 027454              JSR      PC,WAITF
3147      030564 012777 000003 152242      MOV      #3,@DSR
3148      030572 017702 152236              MOV      @DSR,R2      ; IS MEM THERE?
3149      030576 100440              BMI      5$      ; NO.
3150      030600 012777 000000 152226      MOV      #0,@DSR      ; YES. GET PIXEL DATA.
3151      030606 017702 152222              MOV      @DSR,R2
3152      030612 052702 176003              BIS      #176003,R2      ; INVERT PIXEL DATA.
3153      030616 005102              COM      R2
3154      030620 010265 003256              MOV      R2,CTABM(R5)      ; PUT PIXEL DATA IN MEM TABLE.
3155      030624 012703 000010              MOV      #8,R3      ; GOT AN EVEN NUMBER OF PIXEL BITS?
3156      030630 005004              CLR      R4
3157      030632 006202              ASR      R2
3158      030634 006202      3$:    ASR      R2
3159      030636 032702 000001              BIT      #1,R2
3160      030642 001401              BEQ      4$
3161      030644 005204              INC      R4
3162      030646 005303      4$:    DEC      R3
3163      030650 001371              BNE      3$
3164      030652 032704 000001              BIT      #1,R4
3165      030656 001410              BEQ      5$
3166      030660              HRDERR      SCPRE,SYSCON      ; NO.
3167      030700 042701 000060      5$:    BIC      #60,R1      ; YES. RE-PROTECT THE MEM.
3168      030704 010177 152130              MOV      R1,@DYR
3169      030710 004737 027454              JSR      PC,WAITF
3170      030714 062705 000002              ADD      #2,R5      ; NEXT MEM.
3171      030720 020527 000010              CMP      R5,#10      ; ALL MEMS CHECKED?
3172      030724 001274              BNE      2$      ; NO.
3173      030726 012737 000010 003252 6$:  MOV      #BIT3,MEMFLG      ; YES. SET ONE FLAG TO INDICATE MEMS AVAIL.
3174      030734 012701 003256              MOV      #MEMTAB,R1

```

```

3175 030740 005721      7$:   TST      (R1)+      :
3176 030742 001403      BEQ      8$          :
3177 030744 052737 000020 003252  8$:   BIS      #BIT4, MEMFLG  :
3178 030752 006237 003252      ASR      MEMFLG    :
3179 030756 103370      BCC      7$          :
3180 030760 004737 025402  CONSY: JSR      PC, DPINIT  :
3181 030764 005003      CLR      R3          :
3182 030766 005005      CLR      R5          : INIT SYNC CHAN TABLE PTR.
3183 030770 012777 000013 152036 1$:   MOV      #13, @DSR    : CLEAR CSR.
3184 030776 005065 003270      CLR      CTABS(R5) : CLEAR TABLE ENTRY.
3185 031002 010501      MOV      R5, R1        : READ CURSOR STATUS.
3186 031004 006201      ASR      R1
3187 031006 000301      SWAB     R1
3188 031010 052701 042100      BIS      #42100, R1
3189 031014 010177 152016      MOV      R1, @DXR
3190 031020 004737 027454      JSR      PC, WAITF
3191 031024 012777 000003 152002  MOV      #3, @DSR    : GOT A DATA READY ERROR?
3192 031032 017702 151776      MOV      @DSR, R2
3193 031036 042702 003777      BIC      #3777, R2
3194 031042 020227 150000      CMP      R2, #150000
3195 031046 001412      BEQ      2$          : YES. (NO CHAN HERE).
3196 031050 052765 100000 003270  BIS      #BIT15, CTABS(R5) : NO. SET CHAN EXISTS IN TABLE.
3197 031056 017702 151756      MOV      @DYR, R2    : GET INTERLACE BIT.
3198 031062 042702 157777      BIC      #157777, R2
3199 031066 050265 003270      BIS      R2, CTABS(R5)
3200 031072 050203      BIS      R2, R3
3201 031074 062705 000002 2$:   ADD      #2, R5
3202 031100 020527 000010      CMP      R5, #10
3203 031104 001331      BNE      1$          : NO.
3204 031106 012705 003270      MOV      #CTABS, R5
3205 031112 010337 003230      MOV      R3, INTLAC
3206 031116 052703 100000      BIS      #BIT15, R3
3207 031122 005715      3$:   TST      (R5)
3208 031124 001402      BEQ      4$          : NO.
3209 031126 021503      CMP      (R5), R3
3210 031130 001006      BNE      5$          : YES. INTERLACE BIT SAME?
3211 031132 062705 000002 4$:   ADD      #2, R5
3212 031136 020527 003300      CMP      R5, #CTABS+10
3213 031142 001367      BNE      3$          : NO.
3214 031144 000410      BR       6$          : YES.
3215
3216 031146      5$:   HRDERR  SCIDE, SYSCON
3217
3218 031166 005737 002512 6$:   TST      MFGFLG
3219 031172 001420      BEQ      9$          : MANUFACTURING FLAG SET?
3220 031174 023737 003256 003022  CMP      MEMTAB, MFGMO
3221 031202 001004      BNE      7$          : NO.
3222 031204 023737 003270 003024  CMP      SYCTAB, MFGSO
3223 031212 001410      BEQ      9$          : YES. NEW MEM 0 = MFG MEM 0 MASTER?
3224 031214      7$:   HRDERR  SCME, SYSCON
3225
3226 031234 012737 000010 003254 9$:   MOV      #BIT3, SYCFLG
3227 031242 012701 003270      MOV      #SYCTAB, R1
3228 031246 005721      10$:  TST      (R1)+
3229 031250 001403      BEQ      11$
3230 031252 052737 000020 003254  BIS      #BIT4, SYCFLG
3231 031260 006237 003254      11$:  ASR      SYCFLG

```

3232 031264 103370
3233 031266 000207
3234
3235
3236

BCC 108 ;
RTS PC

```
3238
3239      :: SUBROUTINE - ENABLE MEM MGT.
3240
3241 031270 005737 003244      KTON:  TST      KTFLG      ; GOT KT?
3242 031274 001403           :      BEQ      1$      ; NO.
3243 031276 012737 000001 177572 :      MOV     #1,SRO    ; YES. ENABLE KT11.
3244 031304 000207      1$:      RTS      PC
3245
3246
3247
3248      :: SUBROUTINE - DISABLE MEM MGT.
3249
3250
3251 031306      KTOFF:  ;TST      KTFLG      ; GOT KT11?
3252           :BEQ      1$      ; NO.
3253 031306 000240      NOP
3254 031310 000240      NOP
3255 031312 012737 000000 177572 :      MOV     #0,SRO    ; DISABLE KT.
3256 031320 000207      1$:      RTS      PC
```

```

3258
3259
3260
3261
3262 031322 005737 003244
3263 031326 001430
3264 031330 004737 031270
3265 031334 012700 173000
3266 031340 012737 001600 172354
3267 031346 012701 140000
3268 031352 012702 150000
3269 031356 010021
3270 031360 020102
3271 031362 001375
3272 031364 023737 172354 003244
3273 031372 001404
3274 031374 062737 000200 172354
3275 031402 000761
3276
3277 031404 004737 031306
3278 031410 000207

: SUBROUTINE - FILL ALL MEMORY OVER 28K WITH 'STOPN'.
: (NOTE - ENABLE KT BEFORE CALLING THIS).
KTBKGD: TST KTFLG : GOT KT?
        BEQ 9$ : NO. GET OUT.
        JSR PC,KTON : YES. ENABLE KT.
        MOV #STOPN,R0 : FILL ALL MEM (>28K) WITH BACKGROUND.
        MOV #1600,@KIPAR6
        1$: MOV #140000,R1
        MOV #150000,R2
        2$: MOV R0,(R1)+
        CMP R1,R2
        BNE 2$
        CMP @KIPAR6,KTFLG
        BEQ 3$
        ADD #200,@KIPAR6
        BR 1$
        3$: JSR PC,KTOFF : DISABLE KT.
        9$: RTS PC

```

```

3280
3281      : SUBROUTINE TO SAVE R0-R5.
3282
3283 031412 010037 031462  SAVERS: MOV    R0,SAVED
3284 031416 012700 031464      MOV    #SAVED+2,R0
3285 031422 010120      MOV    R1,(R0)+
3286 031424 010220      MOV    R2,(R0)+
3287 031426 010320      MOV    R3,(R0)+
3288 031430 010420      MOV    R4,(R0)+
3289 031432 010520      MOV    R5,(R0)+
3290 031434 000207      RTS     PC
3291
3292      : SUBROUTINE TO RESTORE R0-R5.
3293
3294
3295 031436 012700 031464  RESTRS: MOV    #SAVED+2,R0
3296 031442 012001      MOV    (R0)+,R1
3297 031444 012002      MOV    (R0)+,R2
3298 031446 012003      MOV    (R0)+,R3
3299 031450 012004      MOV    (R0)+,R4
3300 031452 012005      MOV    (R0)+,R5
3301 031454 013700 031462  MOV    SAVED,R0
3302 031460 000207      RTS     PC
3303
3304
3305
3306 031462 000000  SAVER: 0
3307 031464 000000      0
3308 031466 000000      0
3309 031470 000000      0
3310 031472 000000      0
3311 031474 000000      0

```

```

3313      ;
3314      ; SUBROUTINE TO SET-UP VARIOUS ENVIRONMENTAL PARAMETERS.
3315      ;
3316      ENVIRN: MEMORY R0
3317      031476      010037 003240      MOV R0,FREE      ; GET 1ST FREE ADDRESS...
3318      031500      062737 000002 003240      ADD #2,FREE
3319      031512      011037 003242      MOV (R0),FRESIZ      ;...AND WORD COUNT.
3320      031516      162737 000004 003242      SUB #4,FRESIZ
3321      031524      013702 002012      MOV L$UNIT,R2      ; GET NUMBER OF UNITS
3322      031530      162737 000007 003242 10$: SUB #7,FRESIZ      ; TAKE AWAY 7 WORDS PER UNIT
3323      031536      005302      DEC R2
3324      031540      001373      BNE 10$
3325
3326      031542      012737 000010 027564      MOV #8.,K100US      ; ASSUME LSI FOR 100US TIMER.
3327      031550      READBUS
3328      031552      BCOMPLETE 1$
3329      031554      012737 000040 027564      MOV #32.,K100US      ; NOT LSI -- CHANGE TIMER.
3330      031562      1$:      ;MOV #60.,HZ      ; ASSUME 60 HZ SYSTEM.
3331      ;
3332      ;
3333      ;
3334      ;
3335      ;
3336      ;
3337      ;
3338      ;
3339      031562      004737 031656      2$:      ;MOV #MAXY60,YMAX
3340      031566      000240 000240      ;CLOCK L,R0
3341      ;
3342      ;
3343      ;
3344      ;
3345      ;
3346      ;
3347      ;
3348      ;
3349      ;
3350      ;
3351      ;
3352      ;
3353      ;
3354      ;
3355      ;
3356      ;
3357      ;
3358      ;
3359      ;
3360      ;
3361      ;
3362      ;
3363      ;
3364      ;
3365      ;
3366      ;
3367      ;

```

NOW THE FOLLOWING CODE PROVIDES A KEY RESTART FOR
THE DIAGNOSTIC SUPERVISOR (DRS), AND FOR XXDP+ MONITOR.
IF EITHER ONE CHANGES, THIS WILL PROBABLY TURN TO SHIT ..
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

```

```

3369
3370      ; SUBROUTINE - IF OVER 28K & KT11 AVAIL, INIT KT11.
3371      ;
3372      KTINIT: CLR      KTFLG
3373      CMP      L$HIME,#1577      ; GOT ENOUGH MEMORY (>28K)?
3374      BLOS     9$                ; NO.
3375      MOV      @#ERRVEC,R0      ; SAVE OLD ERR VEC PTR.
3376      MOV      #2$,@#ERRVEC    ; SET ERR VEC PTR.
3377      TST      @#SRO           ; GOT KT11?
3378      NOP                     ; (TRAP IF NO).
3379      MOV      L$HIME,KTFLG     ; YES. SET KT FLAG.
3380      BIC      #177,KTFLG
3381      MOV      R0,@#ERRVEC      ; RESTORE OLD ERR VEC PTR.
3382      CLR      R0              ; R0 = AR DATA.
3383      MOV      #KIPAR0,R1      ; R1 = KI REGS PTR.
3384      MOV      #77406,-40(R1)  ; SET DESCRIPTOR REG.
3385      MOV      R0,(R1)+        ; SET KIPAR REG.
3386      ADD      #200,R0          ; BUMP AR DATA BY '4K'.
3387      CMP      R0,#2000        ; AT 'I/O'?
3388      BNE      1$              ; NO.
3389      MOV      #177600,-(R1)   ; YES. SET KIPAR7 FOR I/O.
3390      BR       9$
3391
3392      2$:  FORRTI
3393      MOV      #.+6,(SP)      ; FORCE RTI RETURN TO -
3394      RTI                          ; - NEXT LOC.
3395      MOV      R0,@#ERRVEC    ; RESTORE OLD ERR VEC PTR.
3396
3397      9$:  RTS      PC

```

```

3397      .SBTTL  INITIALIZE SECTION
3398
3399      :++
3400      : THE INITIALIZE SECTION CONTAINS THE CODING THAT IS PERFORMED
3401      : AT THE BEGINNING OF EACH PASS.
3402      :--
3403 032004      BGNPROT
3404 032004 177777 177777 177777      -1, -1, -1, -1 ; NO DEVICE PROTECTION REQUIRED.
3405 032014      ENDPROT
3406
3407 032014      BGNINIT
3408
3409      : IF 'START' OR 'RESTART', SET QUICK-PASS FLAG AND BUS-INIT.
3410      : IF 'CONTINUE', NOTHING IS REQUIRED.
3411      :
3412 032014 005037 023064      CLR      SIFLAG      ;CLEAR 'SOFT INIT' FLAG
3413 032020      READEF #EF.CONTINUE
3414 032026      BNCOMPLETE 1$
3415 032030 023737 003026 002012      CMP      UNITN,L$UNIT      ;UNIT IN RANGE?
3416 032036 103056      BHIS      4$      ;BR IF NO.
3417 032040 005737 003224      TST      DUFLG      ;DROPPED UNIT?
3418 032044 100460      BMI      NXTU      ;BR IF YES
3419 032046 013701 003026      MOV      UNITN,R1
3420 032052 006301      ASL      R1
3421 032054 005761 003302      TST      ERTABL(R1)
3422 032060 001504      BEQ      SETU
3423 032062 032761 040000 003302      BIT      #BIT14,ERTABL(R1)      ; DROPPED?
3424 032070 001046      BNE      NXTU
3425 032072      EXIT      INIT      ;DO NOTHING IF 'CONTINUE'.
3426 032076      1$: READEF #EF.NEW
3427 032104      BNCOMPLETE NXTU      ;TAKE NEXT UNIT IF NOT NEW PASS.
3428 032106      READEF #EF.START
3429 032114      BCOMPLETE 2$
3430 032116      READEF #EF.RESTART
3431 032124      BNCOMPLETE 31$
3432 032126      2$:      ; 1ST PASS, BUS-INIT...
3433 032126      BRESET      ; BUS RESET.
3434 032130 012737 177777 003030      MOV      #-1,QVP      ;...QUICK VERIFY...
3435 032136 005037 030070      CLR      TESTK      ;...CLEAR TOTAL TEST COUNT.
3436 032142 004737 031476      JSR      PC,ENVIRN      ; SET ENVIRONMENT.
3437 032146 012700 003302      MOV      #ERTABL,R0
3438 032152 005020      30$: CLR      (R0)+      ;CLEAR THE ERROR TABLE
3439 032154 020027 003502      CMP      R0,#ERTABE
3440 032160 103774      BLO      30$
3441 032162 000404      BR      4$
3442 032164 005037 003030      31$: CLR      QVP
3443 032170 000137 032240      JMP      PASRPT      ;GO REPORT THE STATUS
3444
3445 032174      4$:
3446 032174 012737 177777 003026      NEWPAS: MOV      #-1,UNITN      ;INIT UNIT NUMBER...
3447 032202 005037 032726      CLR      DEVCNT      ;CLEAR COUNT OF DEVICES RUNNING
3448 032206      NXTU: BREAK
3449 032210 005237 003026      INC      UNITN      ;...AND SET NEXT UNIT NUMBER.
3450 032214 023737 003026 002012      CMP      UNITN,L$UNIT
3451 032222 103423      BLO      SETU
3452 032224 012737 177777 003224      MOV      #-1,DUFLG
3453 032232 000401      BR      11$

```

3454	032234				DOCLN		:ABORT, NO MORE UNITS.
3455	032236	000240			NOP		
3456	032240						
3457	032240	023727	002012	000001	PASRPT:	CMP L\$UNIT,#1	:HOW MANY UNITS SELECTED?
3458	032246	101752			BLOS NEWPAS		:BR IF ONLY 1
3459	032250	005737	032726		TST DEVCNT		:ARE ANY STILL RUNNING?
3460	032254	001747			BEQ NEWPAS		:BR IF NO
3461	032256				RFLAGS R0		
3462	032260	032700	000100		BIT #ISR,R0		:SHOULD WE PRINT STATISTICS
3463	032264	001343			BNE NEWPAS		:BR IF NO
3464							
3465	032266				DORPT		
3466	032270	000741			BR NEWPAS		
3467	032272						
3468					10\$:		
3469	032272				SETU:	GPHARD UNITN,R0	:GET UNIT N P-TABLE POINTER.
3470	032300				BNCOMplete NXTU		:BR IF UNIT NOT AVAILABLE.
3471	032302	005037	003224		CLR DUFLG		:CLEAR 'DROPPED' FLAG.
3472	032306	005237	032726		INC DEVCNT		
3473	032312	012001			MOV (R0)+,R1		: GET 1ST REGISTER ADDRESS.
3474	032314	012702	003032		MOV #DPC,R2		
3475	032320	010122			20\$: MOV R1,(R2)+		
3476	032322	062701	000002		ADD #2,R1		
3477	032326	020227	003046		CMP R2,#LMR		
3478	032332	101772			BLOS 20\$		
3479							
3480							
3481	032334	012001			MOV (R0)+,R1		: GET 1ST VECTOR ADDRESS.
3482	032336	011002			MOV (R0),R2		: GET INTERRUPT PRIORITY
3483	032340	012037	003050		MOV (R0)+,DPRI		: SET INTERRUPT PRIORITY.
3484	032344	010137	003052		MOV R1,STPV		: SET STOP VECTOR POINTER...
3485	032350	012721	026242		MOV #DSTP,(R1)+		:...VECTOR...
3486	032354	010221			MOV R2,(R1)+		:...AND PRIORITY.
3487	032356	010137	003054		MOV R1,JSMV		: DITTO JOY-STICK MATCH.
3488	032362	012721	026274		MOV #DJM,(R1)+		
3489	032366	010221			MOV R2,(R1)+		
3490	032370	010137	003056		MOV R1,TOTV		: DITTO TIME-OUT.
3491	032374	012721	026210		MOV #DTO,(R1)+		
3492	032400	010221			MOV R2,(R1)+		
3493	032402	010137	003060		MOV R1,JSSV		: DITTO JOY-STICK SWITCH.
3494	032406	012721	026326		MOV #DJS,(R1)+		
3495	032412	010221			MOV R2,(R1)+		
3496	032414	010137	003062		MOV R1,LUTV		: DITTO LUT DONE.
3497	032420	012721	026602		MOV #LDUN,(R1)+		
3498	032424	010221			MOV R2,(R1)+		
3499	032426	012037	003216		MOV (R0)+,LUTAV		: GET 'LUT AVAILABLE' ENTRY
3500	032432	012737	001676	003250	MOV #MAXY60,YMAX		: ASSUME 60HZ FREQUENCY.
3501	032440	012737	000074	003246	MOV #60.,HZ		:
3502	032446	005720			TST (R0)+		: WHAT IS SELECTED FREQUENCY?
3503	032450	001406			21\$: BEQ 21\$		: BR IF 60HZ
3504	032452	012737	000062	003246	MOV #50.,HZ		: SAY 50 HZ
3505	032460	012737	001776	003250	MOV #MAXY50,YMAX		: AND SET UP FOR 512 VISIBLE LINES
3506	032466				21\$:		
3507	032466	005037	003064		CLR OPTI		
3508	032472	005737	002502		TST DPUMOD		: DPU ONLY?
3509	032476	001003			BNE 1\$		: SKIP NEXT IF DPU-ONLY.
3510	032500	012737	040000	003064	MOV #I,OPTI		:SET OPTIONAL I(NTENSIFY) BIT.

3511 032506
3512
3513

1\$:
:
:

TST
BEQ
5\$

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

```

3515
3516
3517
3518
3519 032506 013701 003026
3520 032512 006301
3521 032514 052761 100000 003302
3522 032522 005037 022300
3523 032526 023727 002012 000001
3524 032534 101416
3525 032536
3526 032540 032700 001000
3527 032544 001412
3528 032546
3529 032572
3530 032572 005037 003226
3531 032576 013701 003032
3532 032602 013702 003040
3533 032606 004737 027566
3534 032612 103003
3535 032614 010137 003226
3536 032620 000416
3537 032622 012777 000000 150204 2$:
3538 032630 005777 150200
3539 032634 100413
3540 032636 013701 003026
3541 032642 006301
3542 032644 005261 003302
3543 032650 012737 000001 003226
3544 032656 012737 177777 003224 3$:
3545
3546 032664 005737 003216 4$:
3547 032670 001412
3548 032672 013701 003042
3549 032676 013702 003046
3550 032702 004737 027566
3551 032706 103003
3552 032710 012737 000000 003216
3553
3554
3555
3556 032716 5$:
3557 032724
3558
3559 032726 000000
3560 032730 045 116 045
3561
3562
3563
3564
3565
3566
3567
3568
3569 032776 005737 003020
3570 033002 001402
3571 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$:
CLR NODEV
MOV DPC,R1
MOV DYP,R2
JSR PC,NXM ; TEST ALL 4 DPU REGISTERS...
BCC 2$ ; ...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
MOV #-1,DUFLG
3$:
TST LUTAV ; NOW, SEE IF LUT INSTALLED.
BEQ 5$ ; 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.
4$:
5$:
; FINALLY, SET CPU PRIORITY AND WE'RE DONE.
5$: SETPRI #PRI00 ; ENABLE INTERRUPTS.
ENDINIT
DEVcnt: 0 ; COUNT OF RUNNING DEVICES
PUNIT: .ASCIZ /%N%N%A***** 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.

```

VSV11/VS11 -- GRAPHIC SYS. DIAG MACRO M1113 29-JAN-81 17:22 PAGE ^{D 7}50-1
INITIALIZE SECTION

SEQ 0081

3572 033010 000207

1\$: RTS PC ;

```

3574
3575
3576
3577
3578
3579
3580
3581 033012
3582 033012 010001
3583 033014 006301
3584 033016 052761 000000 003302
3585 033024 042761 040000 003302
3586 033032
3587 033054
3588 033060 045 116 045 1$:
3589
3590
3591 033106
3592
3593
3594
3595
3596
3597
3598
3599
3600
3601
3602
3603 033110
3604 033110 012737 177777 003224
3605 033116 010001
3606
3607 033120 006301
3608 033122 052761 140000 003302
3609 033130 000240 000240 000240
3610 033136
3611 033160
3612 033164 045 116 045 1$:
3613
3614 033214
3615
3616
3617
3618 033216
3619 033216

.SBTTL ADD AND DROP UNITS SECTIONS

;+
; THE ADD-UNIT SECTION CONTAINS THE CODING THAT CAUSES A DEVICE
; TO BE (A) ADDED TO THE TEST LIST FOR THE FIRST TIME,
; OR (B) RE-INSERTED IF IT HAD BEEN PREVIOUSLY DROPPED.
;--

BGNAU
MOV R0,R1 ; GET UNIT TO BE ADDED (R0)
ASL R1 ; MAKE IT A WORD INDEX
BIS #100000,ERTABL(R1) ; SET THE 'ACTIVE' BIT
BIC #40000,ERTABL(R1) ; CLEAR THE 'DROPPED' BIT
PRINTF #1$,R0
EXIT AU
.ASCIIZ /%N% UNIT %D% ADDED/
.EVEN

ENDAU ; UNUSED.

;+
; THE DROP-UNIT SECTION CONTAINS THE CODING THAT CAUSES A DEVICE
; TO BE REMOVED FROM THE TEST LIST.
;--

SUPVSR DOES THE 'DROPPING'. THIS IS JUST TO TELL THE MAN.
'DROPPED' UNITS ARE RE-SELECTED ON OPERATOR 'STA' OR 'ADD'
COMMAND, OTHERWISE REMAIN INACTIVE. THE 'DISPLAY' COMMAND
WILL PRINT ALL DROPPED UNITS, AND THE P-TABLES OF THOSE
WHICH ARE STILL ACTIVE.
UPON ENTRY, R0 CONTAINS THE UNIT TO BE DROPPED.

BGNDU
MOV #-1,DUFLG
MOV R0,R1
MOVB #-1,STIK(R1) ; REMOVE 'JOY-STICK' FLAG.
ASL R1
BIS #140000,ERTABL(R1) ; SAY DROPPED
240,240,240 ; ?????????
PRINTF #1$,R0
EXIT DU
.ASCIIZ /%N% UNIT %D% DROPPED/
.EVEN
ENDDU

;+
; AUTO-DROP CODE SECTION.
;--

BGNAUTO
ENDAUTO ; UNUSED.

```

3621
3622
3623
3624
3625
3626
3627
3628 033220
3629 033220 005737 003224
3630 033224 100414
3631
3632
3633 033226 004737 026756
3634 033232 012777 040000 147576
3635 033240 004737 027454
3636 033244 012777 100000 147506
3637 033252 004737 027454
3638 033256 005737 003216
3639 033262 001402
3640 033264 005077 147552
3641 033270
3642
3643
3644
3645
3646 033272
3647 033272
3648 033312 010246
3649 033314 010346
3650 033316 010446
3651 033320 012704 003302
3652 033324 005003
3653 033326 011402
3654 033330 001467
3655 033332 100066
3656 033334 032702 040000
3657 033340 001015
3658 033342 042702 170000
3659 033346
3660 033372 000446
3661 033374 020227 160000
3662 033400 001012
3663 033402
3664 033424 000431
3665 033426 020227 160001
3666 033432 001012
3667 033434
3668 033456 000414
3669 033460 042702 170000
3670 033464
3671 033510 062704 000002
3672 033514 005203
3673 033516 020427 003502
3674 033522 103701
3675 033524 012604
3676 033526 012603
3677 033530 012602

.SBTTL CLEAN-UP AND REPORT CODING SECTIONS

;;
; THE CLEANUP CODING SECTION CONTAINS THE CODING THAT IS
; EXECUTED AT THE END OF EACH PASS (OR SUB-PASS).
; USE TO RETURN DEVICE UNDER TEST TO A NEUTRAL STATE.
;--

```

      BGNCLN
      TST     DUFLG      ;'DROPPED' FLAG IS 'SET ON...
      BMI     1$         ;...AND GROSS DPU FAULT...
                          ;...DON'T TRY TO XCT DPU CODE.

      JSR     PC,RELEAS  ;CHECK FOR 'HUNG' DPU.
      MOV     #GTJSSW,@DXR ;CLEAR PENDING SWITCH INTERRUPT
      JSR     PC,WAITF
      MOV     #100000,@DYR ;DO SOFT INIT.
      JSR     PC,WAITF
1$:   TST     LUTAV
      BEQ     2$
      CLR     @LSR       ; IF LUT, CLEAR STATUS.
2$:   ENDCLN

```

;;
; THE REPORT CODING SECTION CONTAINS THE
; 'PRINTS' CALLS THAT GENERATE STATISTICAL REPORTS.
;--

```

      BGNRPT
      PRINTS #DEVSUM
      MOV     R2,-(SP)
      MOV     R3,-(SP)
      MOV     R4,-(SP)
      MOV     #ERTABL,R4 ; GET START OF ERROR TABLE.
      CLR     R3          ; CLEAR UNIT NUMBER
1$:   MOV     (R4),R2     ; GET ERROR TABLE ENTRY & TEST IT.
      BEQ     4$          ; ZERO IF UNIT NOT RUN
      BPL     4$
      BIT     #BIT14,R2   ; WAS UNIT DROPPED?
      BNE     2$          ; BR IF YES
      BIC     #^C7777,R2 ; GET ERROR COUNT FIELD
      PRINTS #DEVONL,R3,R2 ; PRINT
      BR      4$
2$:   CMP     R2,#160000  ; WAS UNIT NON-EXISTENT?
      BNE     3$          ; BR IF NO
      PRINTS #DEVNXR,R3
      BR      4$
3$:   CMP     R2,#160001  ; WAS UNIT NOT READY AT STARTUP?
      BNE     30$         ; BR IF NO.
      PRINTS #DEVNRD,R3
      BR      4$
30$:  BIC     #^C7777,R2
      PRINTS #DEVDRD,R3,R2
4$:   ADD     #2,R4
      INC     R3
      CMP     R4,#ERTABE
      BLO     1$
      MOV     (SP)+,R4
      MOV     (SP)+,R3
      MOV     (SP)+,R2

```

3678 033532

ENDRPT ; UNUSED.

3679

3680

3681 033534 045 116 045 DEVSUM: .ASCIZ /%N%ADEVICE STATUS SUMMARY:%N/
3682 033571 045 101 040 DEVONL: .ASCIZ /%A UNIT %D3%A ONLINE, ERRORS - %D%N/
3683 033641 045 101 040 DEVNXR: .ASCIZ /%A UNIT %D3%A DROPPED, NON-EXISTENT REGISTER%N/
3684 033723 045 101 040 DEVNRD: .ASCIZ /%A UNIT %D3%A DROPPED, NOT READY AT STARTUP%N/
3685 034004 045 101 040 DEVDRD: .ASCIZ /%A UNIT %D3%A DROPPED, ERRORS - %D%N/
3686 .EVEN

3688		.SBTTL	
3689		.SBTTL	HARDWARE TEST SECTION.
3690		.SBTTL	
3691		.SBTTL	TEST # DESCRIPTION
3692		.SBTTL	-----
3693		.SBTTL	
3694	000000	IN=0	
3695		.NLIST	MC,ME

3697

```
*****
*
* BEGIN TEST 1 - STATIC RESET
*
*****
```

```

; (ITERATION COUNT = 1.)

```

```

: EXECUTE A BUS RESET INSTRUCTION AND CHECK THE STATE OF THE DPU.
: THIS TEST IS ALMOST IDENTICAL TO THE 'SOFT-INIT' ROUTINE (DPINIT:)
: USED TO INIT THE DPU AT THE START OF MANY OF THE TESTS IN THIS
: PROGRAM. THE DIFFERENCES ARE THAT INSTEAD OF A SOFTWARE DPU INIT,
: A BUS RESET IS USED, AND THAT THE ERROR MESSAGE READS 'BUS RESET',
: INSTEAD OF 'SOFT-INIT'.

```

```
STRSET: JSR      PC,TSTGO      ; TITLE.
```

```

18:      BGNSEG          ; <<<<<<<<<<<<<<<<<<<<<< BEGIN SFMENT.
       BRSEF1         : BUS RESE1.

```

```

JSR      PC, WAITF      ;WAIT FOR A COMPLETE
CLR      R2              ;EXPECT DPC TO BE 0.
MOV      #BIT15, R3      ;EXPECT DSR TO BE 100000.
TST      @DPC             ;TEST THE DPC, S/B -0.
BNE      Z$              ;ERROR IF NOT 0.
CMP      @DSR, R3         ;CHECK DSR.

```

```

2$: DFERR BRINIT,PCSERR ;REPORT ERROR IN DPC OR DSR.
3$: JSR PC,IIRCHK ;CHECK THE INTERNAL REGISTERS

```

```

501      CLR      R4
502      CLR      R5
503      ;DXR AND DYR SHOULD BE 0.

```

CEA	AD
TST	ADXR
BNE	48
TST	ADYR
BEQ	58

48: DEFERR BRINIT,XVERR

[illegible]

```
COMEND 1$  
JSR PC,LOOP : REPEAT 'TIL LOOPER EXPIRES.  
BCS 64$  
BR 65$
```

648: JMP 18

[illegible]

END. TEST

END TEST 1

VS11/VS11 -- GRAPHIC SVS. DIAG MACRO M1113 29-JAN-81 17:22 PAGE 54-1
... TEST 1 STATIC RESET

SEQ 0087

VS
..

```
*****
*
* BEGIN TEST 2      -   RESET ONES
*
*****
```

3735				
3736				
3737				
3738				
3739				
3740				
3741				
3742	034340	004737	027702	
3743	034344			
3744				
3745	034346	012701	177763	
3746	034352			
	034352	012700	000013	
	034356	050100		
	034360	010077	146450	
3747	034364			
	034364	012777	000003	146442
	034372	017702	146436	
3748	034376			
	034376	020102		
	034400	001411		
	034402			
	034424			
3749				
3750	034424			
	034424	012777	177774	146402
3751	034432			
	034432	012777	170015	146374
3752	034440			
	034440	012777	000004	146366
	034446	017702	146362	
3753	034452	010203		
3754	034454	042702	170000	
3755	034460	012701	007777	
3756	034464			
	034464	020102		
	034466	001411		
	034470			
	034512			
3757	034512	010302		
3758	034514	042702	007777	
37	034520	012701	170000	
.	524			
	524	020102		
	526	001411		
	034530			
	034552			
3761				

[illegible]

```

MOV      #177763,R1
WTDSPA  R1,CSR
MOV      #SETCSR,R0      ; WRITE R1 INTO CSR .
BIS      R1,R0
MOV      R0,@DSR
RDDSRA  CSR,R2
MOV      #SELCSR,@DSR    ; READ CSR INTO R2 .
MOV      @DSR,R2
IFERROR  ICSRE,EXPREC,,CKLOOP
CMP      R1,R2            ; OK?
BEQ      64$              ; YES.
HRDERR   ICSRE,EXPREC,CKLOOP ; NO.

```

648:

```

WTD SRA      #177760,MRR
MOV          #177760^C17!SETMRR,@DSR
WTD SRA      #170000,MPM
MOV          #170000^C17!SETMPM,@DSR
RDD SRA      MRR,R2
MOV          #SELMRR,@DSR      ; READ MRR INTO R2
MOV          @DSR,R2
MOV          R2,R3
BIC          #170000,R2      ; MRR OK?
MOV          #7777,R1
IF ERROR     IMRRE,EXPREC,,CKLOOP
CMP          R1,R2      ; OK?
BEQ          65$      ; YES.
HRD ERR      IMRRE,EXPREC,CKLOOP      ; NO.

```

658:

```
MOV      R3,R2                ; MPM OK?
BIC      #7777,R2
MOV      #170000,R1
IFERROR  IMPME,EXPREC,,CKLOOP
CMP      R1,R2                ; OK?
BEQ      66$                  ; YES.
HRDERR   IMPME,EXPREC,CKLOOP  ; NO.
```

668:

[illegible][illegible]

```

: *
: *      END TEST 2
: *
: *****

```

```

*****
*
* BEGIN TEST 3      -   SOFT INIT ONES
*
*****

```

3800				
3801				
3802				
3803				
3804				
3805				
3806				
3807	035114	004737	027702	
3808	035120			
3809				
3810	035122	012701	177763	
3811	035126			
	035126	012700	000013	
	035132	050100		
	035134	010077	145674	
3812	035140			
	035140	012777	000003	145666
	035146	017702	145662	
3813	035152			
	035152	020102		
	035154	001411		
	035156			
	035200			
3814				
3815	035200			
	035200	012777	177774	145626
3816	035206			
	035206	012777	170015	145620
3817	035214			
	035214	012777	000004	145612
	035222	017702	145606	
3818	035226	010203		
3819	035230	042702	170000	
3820	035234	012701	007777	
3821	035240			
	035240	020102		
	035242	001411		
	035244			
	035266			
3822	035266	010302		
3823	035270	042702	007777	
3824	035274	012701	170000	
3825	035300			
	035300	020102		
	035302	001411		
	035304			
	035326			
3826				

[illegible]

```

MOV      #177763,R1
WTDSPRA R1,CSR
MOV      #SETCSR,R0      ; WRITE R1 INTO CSR.
BIS      R1,R0
MOV      R0,@DSR
RDDSPRA CSR,R2
MOV      #SELCSR,@DSR    ; READ CSR INTO R2.
MOV      @DSR,R2
IFERROR  ICSRE,EXPREC,,CKLOOP
CMP      R1,R2           ; OK?
BEQ      64$             ; YES.
HRDERR   ICSRE,EXPREC,CKLOOP ; NO.

```

648:

```

WTD SRA    #177760,MRR
MOV        #177760^C17!SETMRR,@DSR
WTD SRA    #170000,MPM
MOV        #170000^C17!SETMPM,@DSR
RDD SRA    MRR,R2
MOV        #SELMRR,@DSR      ; READ MRR INTO R2.
MOV        @DSR,R2
MOV        R2,R3
BIC        #170000,R2        ; MRR OK?
MOV        #7777,R1
IFERROR    IMRRE,EXPREC,,CKLOOP
CMP        R1,R2              ; OK?
BEQ        65$                ; YES.
HRDERR     IMRRE,EXPREC,CKLOOP ; NO.

```

658:

```

055:      MOV      R3,R2                ; MPM OK?
      BIC      #7777,R2
      MOV      #170000,R1
      IFERROR  IMPME,EXPREC,,CKLOOP
      CMP      R1,R2                ; OK?
      BEQ      66$                  ; YES.
      HRDERR   IMPME,EXPREC,CKLOOP ; NO.

```

66\$:

3827	035326				WTDSRA	#177760,XRR	
	035326	012777	177776	145500	MOV	#177760,*C17!SETXRR,@DSR	
3828	035334				WTDSRA	#170000,XPM	
	035334	012777	170017	145472	MOV	#170000,*C17!SETXPM,@DSR	
3829	035342				RDDSRA	XRR,R2	
	035342	012777	000006	145464	MOV	*SELXRR,@DSR ; READ XRR INTO R2 .	
	035350	017702	145460		MOV	@DSR,R2	
3830	035354	010203			MOV	R2,R3	
3831	035356	042702	170000		BIC	#170000,R2 ; XRR OK?	
3832	035362	012701	007777		MOV	#7777,R1	
3833	035366				IFERROR	IXRRE,EXPREC,,CKLOOP	
	035366	020102			CMP	R1,R2 ; OK?	
	035370	001411			BEQ	67\$; YES.	
	035372				HRDERR	IXRRE,EXPREC,CKLOOP ; NO.	
	035414						
3834	035414	010302			MOV	R3,R2 ; MPM OK?	
3835	035416	042702	007777		BIC	#7777,R2	
3836	035422	012701	170000		MOV	#170000,R1	
3837	035426				IFERROR	IXPME,EXPREC	
	035426	020102			CMP	R1,R2 ; OK?	
	035430	001410			BEQ	68\$; YES.	
	035432				HRDERR	IXPME,EXPREC, ; NO.	
	035452						
3838							
3839							
3840	035452	012777	100000	145360	MOV	#100000,@DYR ;DO SOFT INIT.	
3841	035460	004737	027454		JSR	PC,WAITF ;WAIT FOR STOP	
3842	035464	005002			CLR	R2 ;EXPECT DPC TO BE 0.	
3843	035466	012703	100000		MOV	#BIT15,R3 ;EXPECT DSR TO BE 100000.	
3844	035472	005777	145334		TST	@DPC ;TEST THE DPC, S/B =0.	
3845	035476	001003			BNE	11\$;ERROR IF NOT 0.	
3846	035500	027703	145330		CMP	@DSR,R3 ;CHECK DSR.	
3847	035504	001410			BEQ	12\$	
3848	035506				DFERR	NSINIT,PCSERR ;REPORT ERROR IN DPC OR DSR.	
3849	035526	004737	023422		JSR	PC,IIRCHK ;CHECK THE INTERNAL REGISTERS	
3850	035532	005004			CLR	R4	
3851	035534	005005			CLR	R5 ;DXR AND DYR SHOULD BE 0.	
3852	035536	005777	145274		TST	@DXR	
3853	035542	001003			BNE	13\$	
3854	035544	005777	145270		TST	@DYR	
3855	035550	001410			BEQ	14\$	
3856	035552				DFERR	NSINIT,XYERR	
3857							
3858	035572				ENDSEG	; <<<<<<<<<<<<<<<<<<<<<<<<<<<< END SEGMENT.	
3859							
3860	035574				COMEND	1\$	
	035574	004737	027646		JSR	PC,LOOP ; REPEAT 'TIL LOOPER EXPIRES.	
	035600	103401			BCS	69\$	
	035602	000402			BR	70\$	
	035604	000137	035120		JMP	1\$	
	035610	004737	030072		JSR	PC,TSTEND ; PRINT ERROR SUMMARY, IF REQ'D.	
3861	035614				EXIT	TST ; EXIT THIS TEST >>>>>>>>>>>>>>>>>>>>>>>>>>>>	
3862							
3863	035620				END.TEST		

*
* END TEST 3
*

3865

3866			
3867			
3868			
3869			
3870			
3871			
3872	035704	004737	027702
3873	035710		
3874	035712		
	035712	004737	025402
3875	035716	012703	177760
3876	035722	010377	145106
3877	035726	012701	000000
3878	035732	017702	145074
3879	035736		
	035736	020102	
	035740	001410	
	035742		
	035762		
3880	035762	017702	145050
3881	035766		
	035766	020102	
	035770	001410	
	035772		
	036012		
3882	036012	017702	145022
3883	036016		
	036016	020102	
	036020	001410	
	036022		
	036042		
3884	036042		
3885	036044		
3886	036046		
	036046	004737	025402
3887	036052	012703	177760
3888	036056		
	036056	012700	000010
	036062	050300	
	036064	010077	144744
3889	036070	012701	000001
3890	036074		
	036074	012777	000001
	036102	017702	144726
3891	036106		
	036106	020102	
	036110	001410	
	036112		

[illegible]

[illegible]

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

3933	036770	042702	170000		BIC	#170000,R2	; MASK OUT XPM.
3934	036774				IFERROR	CSXRE,DUADRE	
	036774	020102			CMP	R1,R2	; OK?
	036776	001410			BEQ	79\$	; YES.
	037000				HRDERR	CSXRE,DUADRE,	; NO.
	037020			79\$:			
3935	037020				RDDSR	XPM,R2	
	037020	012777	000006	144006	MOV	#SELXPM,@DSR	; READ XPM INTO R2 .
	037026	017702	144002		MOV	@DSR,R2	
3936	037032	042702	007777		BIC	#7777,R2	; MASK OUT XRR.
3937	037036				IFERROR	CSXPE,DUADRE	
	037036	020102			CMP	R1,R2	; OK?
	037040	001410			BEQ	80\$	; YES.
	037042				HRDERR	CSXPE,DUADRE,	; NO.
	037062			80\$:			
3938	037062				ENDSEG		; <<<<<<<<<<<<<<<<<<<<<< END SEGMENT.
3939	037064				BGNSEG		; <<<<<<<<<<<<<<<<<<<<<< BEGIN SEGMENT.
3940	037066				INITDP		
	037066	004737	025402		JSR	PC,DPINIT	; GO DO SOFT INIT, CHECK THE STATE.
3941	037072	012703	007777		MOV	#7777,R3	
3942	037076				WTDSP	R3,MRR	
	037076	012700	000014		MOV	#SETMRR,R0	; WRITE R3 INTO MRR .
	037102	050300			BIS	R3,R0	
	037104	010077	143724		MOV	R0,@DSR	
3943	037110	012701	100000		MOV	#100000,R1	
3944	037114				RDDSR	DSR,R2	
	037114	012777	000000	143712	MOV	#SELDSP,@DSR	; READ DSR INTO R2 .
	037122	017702	143706		MOV	@DSR,R2	
3945	037126				IFERROR	MWDSE,DUADRE	
	037126	020102			CMP	R1,R2	; OK?
	037130	001410			BEQ	81\$	; YES.
	037132				HRDERR	MWDSE,DUADRE,	; NO.
	037152			81\$:			
3946	037152	012701	000001		MOV	#1,R1	
3947	037156				RDDSR	PCS,R2	
	037156	012777	000001	143650	MOV	#SELPSC,@DSR	; READ PCS INTO R2 .
	037164	017702	143644		MOV	@DSR,R2	
3948	037170				IFERROR	MWPCE,DUADRE	
	037170	020102			CMP	R1,R2	; OK?
	037172	001410			BEQ	82\$	; YES.
	037174				HRDERR	MWPCE,DUADRE,	; NO.
	037214			82\$:			
3949	037214	012701	100000		MOV	#100000,R1	
3950	037220				RDDSR	FLG,R2	
	037220	012777	000002	143606	MOV	#SELFGL,@DSR	; READ FLG INTO R2 .
	037226	017702	143602		MOV	@DSR,R2	
3951	037232				IFERROR	MRFLE,DUADRE	
	037232	020102			CMP	R1,R2	; OK?
	037234	001410			BEQ	83\$	; YES.
	037236				HRDERR	MRFLE,DUADRE,	; NO.
	037256			83\$:			
3952	037256	012701	000003		MOV	#3,R1	
3953	037262				RDDSR	CSR,R2	
	037262	012777	000003	143544	MOV	#SELCSR,@DSR	; READ CSR INTO R2 .
	037270	017702	143540		MOV	@DSR,R2	
3954	037274				IFERROR	MRCSE,DUADRE	
	037274	020102			CMP	R1,R2	; OK?

	037276	0014'0			BEQ	84\$	:	YES.
	037300				HRDERR	MRCSE ,DUADRE ,	:	NO.
	037320							
	037320	012701	000000		MOV	#0,R1		
3955	037320				RDDSRA	MPM,R2		
3956	037324				MOV	#SELMPM,@DSR	:	READ MPM INTO R2 .
	037324	012777	000004	143502	MOV	@DSR,R2		
	037332	017702	143476		BIC	#7777,R2	:	MASK OUT MRR.
3957	037336	042702	007777		IFERROR	MRMPE,DUADRE		
3958	037342				CMP	R1,R2	:	OK?
	037342	020102			BEQ	85\$	:	YES.
	037344	001410			HRDERR	MRMPE,DUADRE,	:	NO.
	037346							
	037366							
	037366							
3959	037366				RDDSRA	XRP,R2		
	037366	012777	000006	143440	MOV	#SELXRR,@DSR	:	READ XRR INTO R2 .
	037374	017702	143434		MOV	@DSR,R2		
3960	037400	042702	170000		BIC	#170000,R2	:	MASK OUT XPM.
3961	037404				IFERROR	MRXRE,DUADRE		
	037404	020102			CMP	R1,R2	:	OK?
	037406	001410			BEQ	86\$	:	YES.
	037410				HRDERR	MRXRE,DUADRE,	:	NO.
	037430							
	037430							
3962	037430				RDDSRA	XPM,R2		
	037430	012777	000006	143376	MOV	#SELXPM,@DSR	:	READ XPM INTO R2 .
	037436	017702	143372		MOV	@DSR,R2		
3963	037442	042702	007777		BIC	#7777,R2	:	MASK OUT XRR.
3964	037446				IFERROR	MRXPE,DUADRE		
	037446	020102			CMP	R1,R2	:	OK?
	037450	001410			BEQ	87\$	:	YES.
	037452				HRDERR	MRXPE,DUADRE,	:	NO.
	037472							
	037472							
3965	037472				ENDSEG		:	<<<<<<<<<<<<<<<<<< END SEGMENT.
3966	037474				BGNSEG		:	<<<<<<<<<<<<<<<<<< BEGIN SEGMENT.
3967	037476				INITDP			
	037476	004737	025402		JSR	PC,DPINIT	:	GO DO SOFT INIT, CHECK THE STATE.
3968	037502	012703	170000		MOV	#170000,R3		
3969	037506				WTDSRA	R3,MPM		
	037506	012700	000015		MOV	#SETMPM,R0	:	WRITE R3 INTO MPM .
	037512	050300			BIS	R3,R0		
	037514	010077	143314		MOV	R0,@DSR		
3970	037520	012701	100000		MOV	#100000,R1		
3971	037524				RDDSRA	DSR,R2		
	037524	012777	000000	143302	MOV	#SELDJR,@DSR	:	READ DSR INTO R2 .
	037532	017702	143276		MOV	@DSR,R2		
3972	037536				IFERROR	MPDSE,DUADRE		
	037536	020102			CMP	R1,R2	:	OK?
	037540	001410			BEQ	88\$	:	YES.
	037542				HRDERR	MPDSE,DUADRE,	:	NO.
	037562							
	037562							
3973	037562	012701	000001		MOV	#1,R1		
3974	037566				RDDSRA	PCS,R2		
	037566	012777	000001	143240	MOV	#SELPCS,@DSR	:	READ PCS INTO R2 .
	037574	017702	143234		MOV	@DSR,R2		
3975	037600				IFERROR	MPPCE,DUADRE		
	037600	020102			CMP	R1,R2	:	OK?
	037602	001410			BEQ	89\$	:	YES.
	037604				HRDERR	MPPCE,DUADRE,	:	NO.

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

3998	040134				RDDSR	DSR,R2		
	040134	012777	000000	142672	MOV	#SELDSE,@DSR	:	READ DSR INTO R2 .
	040142	017702	142666		MOV	@DSR,R2		
3999	040146				IFERROR	XRDSE,DUADRE		
	040146	020102			CMP	R1,R2	:	OK?
	040150	001410			BEQ	95\$	:	YES.
	040152				HRDERR	XRDSE,DUADRE.	:	NO.
	040172							95\$:
4000	040172	012701	000001		MOV	#1,R1		
4001	040176				RDDSR	PCS,R2		
	040176	012777	000001	142630	MOV	#SELPCS,@DSR	:	READ PCS INTO R2 .
	040204	017702	142624		MOV	@DSR,R2		
4002	040210				IFERROR	XRPC,DUADRE		
	040210	020102			CMP	R1,R2	:	OK?
	040212	001410			BEQ	96\$	:	YES.
	040214				HRDERR	XRPC,DUADRE.	:	NO.
	040234							96\$:
4003	040234	012701	100000		MOV	#100000,R1		
4004	040240				RDDSR	FLG,R2		
	040240	012777	000002	142566	MOV	#SELFLG,@DSR	:	READ FLG INTO R2 .
	040246	017702	142562		MOV	@DSR,R2		
4005	040252				IFERROR	XRFLE,DUADRE		
	040252	020102			CMP	R1,R2	:	OK?
	040254	001410			BEQ	97\$	:	YES.
	040256				HRDERR	XRFLE,DUADRE.	:	NO.
	040276							97\$:
4006	040276	012701	000003		MOV	#3,R1		
4007	040302				RDDSR	CSR,R2		
	040302	012777	000003	142524	MOV	#SELCSR,@DSR	:	READ CSR INTO R2 .
	040310	017702	142520		MOV	@DSR,R2		
4008	040314				IFERROR	XRCSE,DUADRE		
	040314	020102			CMP	R1,R2	:	OK?
	040316	001410			BEQ	98\$	:	YES.
	040320				HRDERR	XRCSE,DUADRE.	:	NO.
	040340							98\$:
4009	040340	012701	000000		MOV	#0,R1		
4010	040344				RDDSR	MRR,R2		
	040344	012777	000004	142462	MOV	#SELMRR,@DSR	:	READ MRR INTO R2 .
	040352	017702	142456		MOV	@DSR,R2		
4011	040356	042702	170000		BIC	#170000,R2	:	MASK OUT MPM.
4012	040362				IFERROR	XRMRE,DUADRE		
	040362	020102			CMP	R1,R2	:	OK?
	040364	001410			BEQ	99\$	:	YES.
	040366				HRDERR	XRMRE,DUADRE.	:	NO.
	040406							99\$:
4013	040406				RDDSR	MPM,R2		
	040406	012777	000004	142420	MOV	#SELMPM,@DSR	:	READ MPM INTO R2 .
	040414	017702	142414		MOV	@DSR,R2		
4014	040420	042702	007777		BIC	#7777,R2	:	MASK OUT MPM.
4015	040424				IFERROR	XRMPE,DUADRE		
	040424	020102			CMP	R1,R2	:	OK?
	040426	001410			BEQ	100\$	:	YES.
	040430				HRDERR	XRMPE,DUADRE.	:	NO.
	040450							100\$:
4016	040450				RDDSR	XPM,R2		
	040450	012777	000006	142356	MOV	#SELXPM,@DSR	:	READ XPM INTO R2 .
	040456	017702	142352		MOV	@DSR,R2		

4017	040462	042702	007777		BIC	#7777,R2	; MASK OUT XRR.
4018	040466				IFERROR	XRXPE,DUADRE	
	040466	020102			CMP	R1,R2	; CK?
	040470	001410			BEQ	101\$	; YES.
	040472				HRDERR	XRXPE,DUADRE,	; NO.
	040512			101\$:			
4019	040512				ENDSEG		; <<<<<<<<<<<<<<<<<<<<<< END SEGMENT.
4020	040514				BGNSEG		; <<<<<<<<<<<<<<<<<<<<<< BEGIN SEGMENT.
4021	040516				INITDP		
	040516	004737	025402		JSR	PC,DPINIT	; GO DO SOFT I.P.I.T., CHECK THE STATE
4022	040522	012703	170000		MOV	#170000,R3	
4023	040526				WTDSPA	R3,XPM	
	040526	012700	000017		MOV	#SETXPM,R0	; WRITE R3 INTO XPM .
	040532	050300			BIS	R3,R0	
	040534	010077	142274		MOV	R0,@DSR	
4024	040540	012701	100000		MOV	#100000,R1	
4025	040544				RDDSPA	DSR,R2	
	040544	012777	000000	142262	MOV	#SELDSP,@DSR	; READ DSR INTO R2 .
	040552	017702	142256		MOV	@DSR,R2	
4026	040556				IFERROR	XPDSE,DUADRE	
	040556	020102			CMP	R1,R2	; OK?
	040560	001410			BEQ	102\$	; YES.
	040562				HRDERR	XPDSE,DUADRE,	; NO.
	040602			102\$:			
4027	040602	012701	000001		MOV	#1,R1	
4028	040606				RDDSPA	PCS,R2	
	040606	012777	000001	142220	MOV	#SELPCS,@DSR	; READ PCS INTO R2 .
	040614	017702	142214		MOV	@DSR,R2	
4029	040620				IFERROR	XPPCE,DUADRE	
	040620	020102			CMP	R1,R2	; OK?
	040622	001410			BEQ	103\$	; YES.
	040624				HRDERR	XPPCE,DUADRE,	; NO.
	040644			103\$:			
4030	040644	012701	100000		MOV	#100000,R1	
4031	040650				RDDSPA	FLG,R2	
	040650	012777	000002	142156	MOV	#SELFLG,@DSR	; READ FLG INTO R2 .
	040656	017702	142152		MOV	@DSR,R2	
4032	040662				IFERROR	XPFLC,DUADRE	
	040662	020102			CMP	R1,R2	; OK?
	040664	001410			BEQ	104\$	; YES.
	040666				HRDERR	XPFLC,DUADRE,	; NO.
	040706			104\$:			
4033	040706	012701	000003		MOV	#3,R1	
4034	040712				RDDSPA	CSR,R2	
	040712	012777	000003	142114	MOV	#SELCSR,@DSR	; READ CSR INTO R2 .
	040720	017702	142110		MOV	@DSR,R2	
4035	040724				IFERROR	XPCSE,DUADRE	
	040724	020102			CMP	R1,R2	; OK?
	040726	001410			BEQ	105\$	; YES.
	040730				HRDERR	XPCSE,DUADRE,	; NO.
	040750			105\$:			
4036	040750	012701	000000		MOV	#0,R1	
4037	040754				RDDSPA	MRR,R2	
	040754	012777	000004	142052	MOV	#SELMRR,@DSR	; READ MRR INTO R2 .
	040762	017702	142046		MOV	@DSR,R2	
4038	040766	042702	170000		BIC	#170000,R2	; MASK OUT MPM.
4039	040772				IFERROR	XPMRE,DUADRE	

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

```

:*****
:
:
:
:
:
:*****

```

```

*****
*
*   BEGIN TEST 5   -   INCREMENTING REGISTERS
*
*****

```

```

: THE READ/WRITE CAPABILITIES OF THE REGISTERS WILL BE CHECKED
: BY "INCREMENTING" EACH REGISTER FROM ITS LOWEST POSSIBLE
: VALUE, MINUS ONE, TO ITS HIGHEST POSSIBLE VALUE, PLUS ONE,
: IE. 1 TO 177777 TO 0.
: R1=EXPECTED, R2=RECEIVED, R0,R3=WORK REGS.

```

```
MOV    #1,R1          ; INCREMENT MRR.
MOV    #20,R3
```

```

3$:   BGNSEG          ; <<<<<<<<<<<<<<<<<<<<<<<<<<<< BEGIN SEGMENT.
      WTDSRA        R3,MRR
      MOV           #SETMRR,R0          ; WRITE R3 INTO MRR .
      BIS           R3,R0
      MOV           R0,@DSR
      RDDSRA        MRR,R2
      MOV           #SELMRR,@DSR       ; READ MRR INTO R2 .
      MOV           @DSR,R2
      IFERROR       IMRRE,EXPREC
      CMP           R1,R2               ; OK?
      BEQ           65$                 ; YES.
      HRDFERR       IMRRE,EXPREC.      ; NO.
```

```

65$:      CKLOOP      : LOOP IF ERROR.

```

4052				
4053				
4054				
4055				
4056				
4057				
4058				
4059	041232			
	041232	004737	027702	
	041236	004737	026074	
4060	041242	012701	000023	
4061				
4062	041246			
4063	041250			
	041250	012700	000013	
	041254	050100		
	041256	010077	141552	
4064	041262			
	041262	012777	000003	141544
	041270	017702	141540	
4065	041274			
	041274	020102		
	041276	001410		
	041300			
	041320			
4066	041320			
4067	041322	062701	000020	
4068	041326	020127	000003	
4069	041332	001346		
4070	041334			
4071				
4072	041336	012701	000001	
4073	041342	012703	000020	
4074				
4075	041346			
4076	041350			
	041350	012700	000014	
	041354	050300		
	041356	010077	141452	
4077	041362			
	041362	012777	000004	141444
	041370	017702	141440	
4078	041374			
	041374	020102		
	041376	001410		
	041400			
	041420			
4079	041420			

4080	041422	062703	000020		ADD	#20,R3	; INCREMENT.
4081	041426	062701	000001		ADD	#1,R1	
4082	041432	020127	000001		CMP	R1,#1	; DONE?
4083	041436	001403			BEQ	4\$	; YES.
4084	041440	042701	170000		BIC	#170000,R1	; NO.
4085	041444	000741			BR	3\$	
4086							
4087	041446			4\$:	ENDSEG		; <<<<<<<<<<<<<<<<<<<<<< END SEGMENT.
4088							
4089	041450	012703	000020		MOV	#20,R3	; INCREMENT MPM.
4090							
4091	041454				BGNSEG		; <<<<<<<<<<<<<<<<<<<<<< BEGIN SEGMENT.
4092	041456			5\$:	WTDSRA	R3,MPM	
	041456	012700	000015		MOV	#SETMPM,R0	; WRITE R3 INTO MPM .
	041462	050300			BIS	R3,R0	
	041464	010077	141344		MOV	R0,@DSR	
4093	041470	010301			MOV	R3,R1	; GET WRITTEN DATA
4094	041472	042701	007777		BIC	#7777,R1	; LEAVE MASK BITS IN R1
4095	041476				RDDSDRA	MPM,R2	
	041476	012777	000004	141330	MOV	#SELMPM,@DSR	; READ MPM INTO R2 .
	041504	017702	141324		MOV	@DSR,R2	
4096	041510				IFERROR	IMPME,EXPREC	
	041510	020102			CMP	R1,R2	; OK?
	041512	001410			BEQ	66\$	; YES.
	041514				HRDERR	IMPME,EXPREC,	; NO.
	041534			66\$:			
4097	041534	005001			CLR	R1	; ASSUME WRITE-PROTECT BIT IS 0
4098	041536	032703	001000		BIT	#BIT9,R3	; DID WE WRITE IT TO 1?
4099	041542	001402			BEQ	50\$	; BR IF NOT.
4100	041544	052701	100000		BIS	#BIT15,R1	; SET BIT 15 FOR HBASE COMPARE IF YES.
4101	041550			50\$:	RDDSDRA	HBA,R2	
	041550	012777	000005	141256	MOV	#SEHBA,@DSR	; READ HBA INTO R2 .
	041556	017702	141252		MOV	@DSR,R2	
4102	041562				IFERROR	IMPWPE,EXPREC	
	041562	020102			CMP	R1,R2	; OK?
	041564	001410			BEQ	67\$	; YES.
	041566				HRDERR	IMPWPE,EXPREC,	; NO.
	041606			67\$:			
4103	041606				CKLOOP		; LOOP IF ERROR.
4104	041610	062703	000020		ADD	#20,R3	; INCREMENT.
4105	041614	020327	000020		CMP	R3,#20	; DONE?
4106	041620	001316			BNE	5\$	; NO.
4107	041622				ENDSEG		; <<<<<<<<<<<<<<<<<<<<<< END SEGMENT.
4108							
4109	041624	012701	000001		MOV	#1,R1	; INCREMENT XRR.
4110	041630	012703	000020		MOV	#20,R3	
4111							
4112	041634				BGNSEG		; <<<<<<<<<<<<<<<<<<<<<< BEGIN SEGMENT.
4113	041636			6\$:	WTDSRA	R3,XRR	
	041636	012700	000016		MOV	#SETXRR,R0	; WRITE R3 INTO XRR .
	041642	050300			BIS	R3,R0	
	041644	010077	141164		MOV	R0,@DSR	
4114	041650				RDDSDRA	XRR,R2	
	041650	012777	000006	141156	MOV	#SELXRR,@DSR	; READ XRR INTO R2 .
	041656	017702	141152		MOV	@DSR,R2	
4115	041662				IFERROR	IXRR,E,EXPREC	
	041662	020102			CMP	R1,R2	; OK?

[illegible]

4148
4149 042136

END.TEST

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


4266

```

4$:      CMP      OPC1,#RPNT!3777 ;DONE THRU RPNT ???
        BEQ      5$                ;BR IF SO.
        ADD      #BIT11,OPC1       ;NO, SET NEXT OPCODE...
        BIC      #1777,OPC1        ;...LEVEL 0...
        BR       1$                ;...AND CONTINUE.

```

[illegible]

```
4322 043406 000137 042654      13$:  JMP      14$           ;USE JMP TO GET BACK FOR LOOP.
4323                          :
4324                          : ON ERROR, SHOW THE FAILING DPU OPCODE.
4325                          :
4326 043412                      20$:  PRINTX #OPCFX,OPC1      ; EXTENSION SHOWS BAD OPCODE.
4327 043436 000207              RTS      PC
4328                          :
4329                          : DISPLAY CODE.
4330                          :
4331 043440 102000 164000 173000 OPC1:  CHAR!LO, D'OP, STOPN    ; DISPLAY CODE.
4332                          :
4333 043446                      END.TEST
```

```
.....
:
: END TEST 7
:
:.....
```

BEGIN TEST 8 - INCREMENTING HISTOGRAM BASE ADDRESS

```

: THE READ/WRITE CAPABILITY OF THE BASE HISTOGRAM COORDINATE WILL BE
: CHECKED BY "INCREMENTING" IT FROM 2 TO 1776 TO 0.
: (NOTE THAT THE WRITTEN DATA WILL APPEAR SHIFTED 1 PLACE TO THE RIGHT.)
: R1- EXPECTED, R2=RECEIVED, R0,R3=WORK REGS.

```

[illegible]

4369
4370 043706

END TEST

.....
END TEST 8
.....

```
*****
*
*   BEGIN TEST 9   -   INCREMENTING CHARACTER BASE ADDRESS
*
*****
```

```

: THE READ/WRITE CAPABILITY OF THE BASE CHARACTER ADDRESS WILL BE
: CHECKED BY 'INCREMENTING' IT FROM 1 TO 177777 TO 0.
: R1= EXPECTED, R2=RECEIVED, R0,R3=WORK REGS.

```

```
INCCBA: COMBEG      PC,TSTGO          ; TITLE.  
JSR                PC,DPRESET        ; DO SOFT INIT.  
1$: MOV            #1,R1  
BGNSEG             ; <<<<<<<<<<<<<<<<<<<<<< BEGIN SEGMENT.  
2$: MOV            R1,11$           ; SET VALUE FOR CBA CODE.  
MOV                #10$,@DPC         ; WRITE CBA.  
JSR                PC,WAITF  
RDDSR              CBA,R2  
MOV                #SELCBA,@DSR       ; READ CBA INTO R2 .  
MOV                @DSR,R2  
IFERROR            ICBAE,EXPREC  
CMP                R1,R2             ; OK?  
BEQ                64$               ; YES.  
HRDERR             ICBAE,EXPREC,     ; NO.  
  
64$: CKLOOP        ; LOOP ON ERROR.  
ADD                #1,R1             ; INCREMENT.  
CMP                R1,#0             ; DONE?  
BNE                2$               ; NO.  
30$: ENDSEG        ; <<<<<<<<<<<<<<<<<<<<<< END SEGMENT.  
COMEND            1$  
JSR                PC,LOOP           ; REPEAT 'TIL LOOPER EXPIRES.  
BCS                65$  
BR                 66$  
  
65$: JMP            1$  
66$: JSR            PC,TSTEND        ; PRINT ERROR SUMMARY, IF REQ'D.  
EXIT              TST                ; EXIT THIS TEST >>>>>>>>>>>>>>>>>>>>>>>  
  
.; DISPLAY CODE.  
10$: SETCB  
11$: 0  
STOPN  
END.TEST
```

[illegible]

VSV11/VS11 -- GRAPHIC SYS. DIAG MACRO M1113 29-JAN-81 17:22 PAGE L⁹ 62-1
*** TEST 9 INCREMENTING CHARACTER BASE ADDRESS

SEQ 0115

END TEST 9

```

:*****
:
:   BEGIN TEST 10   -   DJMS/DPOP
:
:*****
:
: (ITERATION COUNT = 10.)

```

3. CHECK THE OPERATION OF THE DJMS AND DPOP OP CODES.

```

JMSPOP: COMBEG
        JSR     PC,TSTGO          ; TITLE.
        JSR     PC,DPRESET        ; DO SOFT INIT.

```

[illegible]

```

DPSTART #10$ ; DO A DJMS.
MOV #10$,@DPC ; START THE DPU.
JSR PC,WAITF ; WAIT FOR DISPLAY STOP.
MOV #21$,R1
MOV @DPC,R2 ; READ DPC INTO R2.
IFERROR JPDPC,EXPREC,,CKLOOP
CMP R1,R2 ; OK?
BEQ 64$ ; YES.
HRDERR JPDPC,EXPREC,CKLOOP ; NO.

```

```

64$:      MOV      #11$,R1
          RDDSRA   PCS,R2
          MOV      #SELP, @DSR      ; READ PCS INTO R2 .
          MOV      @DSR,R2
          IFERROR  JPPCSE,EXPRES,,CKLOOP
          CMP      R1,R2             ; OK?
          BEQ      65$              ; YES.
          HRDERR   JPPCSE,EXPRES,CKLOOP ; NO.

```

```

65$:      DPCONT                ; NOW DO A DPOP.
        BIS      #BIT0,@DPC    ; CONTINUE THE DPU.
        JSR      PC,WAITF      ; WAIT FOR DISPLAY STOP.
        MOV      #12$,R1
        MOV      @DPC,R2       ; READ DPC.
        IFERROR   JPDPCF,EXPREF,,CKLOOP
        CMP      R1,R2         ; OK?
        BEQ      66$           ; YES.
        HRDERR   JPDPCF,EXPREF,CKLOOP ; NO.

```

```

66$:      DPCONT                ; DO A DJMS (WITH AUX SEG BIT SET).
        BIS      #BIT0,@DPC    ; CONTINUE THE DPU.
        JSR      PC,WAITF      ; WAIT FOR DISPLAY STOP.
        MOV      #21$!AUXSEG,R1
        MOV      @DPC,R2       ; READ DPC INTO R2.
        IFERROR  JDPDCE,EXPREC,,CKLOOP
        CMP      R1,R2         ; OK?
        BEQ      67$           ; YES.
        HRDERR  JDPDCE,EXPREC,CKLOOP ; NO.

```

```
67$:      MOV        #13$,R1  
          RDDSRA    PCS,R2  
          MOV       #SELPDS,@DSR      ; READ PCS INTO R2 .  
          MOV       @DSR,R2  
          IFERROR   JPPCSE,EXPREC,,CKLOOP  
          CMP       R1,R2              ; OK?  
          BEQ       68$                ; YES.  
          HRDERR    JPPCSE,EXPREC,CKLOOP ; NO.  
  
68$:      DPCONT     ; NOW DPOP AGAIN.  
          BIS       #BIT0,@DPC         ; CONTINUE THE DPU.  
          JSR       PC,WAITF           ; WAIT FOR DISPLAY STOP.  
          MOV       #20$,R1  
          MOV       @DPC,R2            ; READ DPC.  
          IFERROR   JPDPCE,EXPREC  
          CMP       R1,R2              ; OK?  
          BEQ       69$                ; YES.  
          HRDERR    JPDPCE,EXPREC,     ; NO.  
  
69$:      ENDSEG      ; <<<<<<<<<<<<<<<<<<<<<< END SEGMENT.  
  
          COMEND    1$  
          JSR       PC,LOOP             ; REPEAT 'TIL LOOPER EXPIRES.  
          BCS       70$  
          BR        71$  
  
70$:      JMP        1$  
  
71$:      JSR       PC,TSTEND           ; PRINT ERROR SUMMARY, IF REQ'D.  
          EXIT      TST                 ; EXIT THIS TEST >>>>>>>>>>>>>>>>>>>>>  
:  
: DISPLAY CODE  
:  
10$:      DJMS,20$  
11$:      STOPN  
12$:      DJMS,20$!AUXSEG  
13$:      STOPN  
  
20$:      STOPN  
21$:      DPOP  
22$:      STOFN  
  
          END.TEST
```

```

:*****
:
:
:      END TEST 10
:
:*****

```

```
*****
: BEGIN TEST 11 - MAIN MEM MGT ACCESS (OVER 28K)
: *****
: (ITERATION COUNT - 1.)
```

```
: CHECK THAT A DISPLAY FILE CAN BE ACCESSED VIA MAIN MEM MGT.  
: IN MEMORY OVER 28K.
```

```
MMMAC: COMSEG  
      JSR    PC,TSTGO           ; TITLE.  
      JSR    PC,DPRESET        ; DO SOFT INIT.  
      TST    KTFLG             ; GOT KT & >28k?  
      BNE    MMMACA            ; YES.  
      RFLAGS RO                ; ARE WE PRINTING TEST NOS.?  
      BIT    #PNT,R0           ; ?  
      BEQ    1$               ; BR IF NO -- DON'T PRINT  
      PRINTF #MNOKT           ; NO.  
1$:   EXIT    TST              ; EXIT THIS TEST >>>>>>>>>>>>>>>>>>>>>>>>>>>>
```

```
MMMACA: JSR    PC,KTBKGD         ; SET UP BACKGROUND.  
1$:     MOV    #1600,R3          ; INIT BANK PTR. (TO START OF 28K PAGE)
```

```
2$:     BGNSEG                  ; <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<  
  
      MOV    R3,#KIPAR6        ; SET KT TO TEST PAGE.  
      MOV    R3,R1              ; SET UP MRR.  
      ASL    R1  
      ASL    R1  
      ASL    R1  
      ASL    R1  
      WTDSPA R1,MRR  
      MOV    #SETMRR,R0         ; WRITE R1 INTO MRK .  
      BIS    R1,R0  
      MOV    R0,#DSR  
      MOV    #140000,R5         ; INIT LOC PTR.  
      JSR    PC,KTON            ; ENABLE KT.  
      MOV    #DNOP,(R5)         ; REPLACE "STOPN" WITH "DNOP" IN TEST LOC.  
      JSR    PC,KTOFF           ; DISABLE KT.  
      DPSTART #0  
      MOV    #0,#DPC : START THE DPU.  
      JSR    PC,WAITF           ; WAIT FOR DISPLAY STOP.  
      BR     4$
```

```
3$:     JSR    PC,KTON            ; ENABLE KT.  
      MOV    #DNOP,(R5)         ; REPLACE "STOPN" WITH "DNOP" IN TEST LOC.  
      JSR    PC,KTOFF           ; DISABLE KT.  
      DPCONT  
      BIS    #BIT0,#DPC         ; CONTINUE THE DPU.  
      JSR    PC,WAITF           ; WAIT FOR DISPLAY STOP.
```

```
4$:     MOV    R5,R1              ; R1 = EXPECTED DPC, AND -
```


AUX MEM MGT ACCESS (OVER 28K)

4521

[illegible]


```
*****
*
* BEGIN TEST 13 - STOP INTERRUPT
*
*****
```

```

; CHECK THE ACTION OF THE STOP INTERRUPT.
; ALSO CHECKED IS THE ACTION OF THE CSR 'FORCE STOP INTR' BIT.
; REPEAT 40 TIMES AFTER QUICK PASS.

```

4595					: CHECK THE ACTION OF THE STOP INTERRUPT.
4596					: ALSO CHECKED IS THE ACTION OF THE CSR 'FORCE STOP INTR' BIT.
4597					: REPEAT 40 TIMES AFTER QUICK PASS.
4598					:
4599					: STPTST: JSR PC,TSTGO ;TITLE
4600	046110	004737	027702		
4601					
4602	046114				19\$: BGNSEG ; <<<<<<<<<<<<<<<<<<<<<< BEGIN SEGMENT.
4603					
4604	046116	004737	025402		JSR PC,DPINIT ;INIT THE DPU SO WE CAN CHECK INTERNAL REGS. ;@@@I
4605	046122				INTSET STP
	046122	012737	177601	026106	MOV #^0177400!IOKCKIN,INTMASK ; PRIME FLAG, EXPECT STP
4606	046130	012777	046742	134674	MOV #STP1,@DPC ;XCT 'STOP AND INT'.
4607	046136	004737	027454		1\$: JSR PC,WAITF
4608	046142	105737	026107		TSTB INTFLAG ; DID WE GET AN INTERRUPT?
4609	046146	001412			BEQ 2\$;BR IF INTERRUPT REC'D.
4610	046150	012701	046136		MOV #1\$,R1 ;PC AT WAIT TIME.
4611	046154				HRDERR NSI,INTERR ;INT EXP'D, NOT REC'D.
4612	046174	004737	023422		2\$: JSR PC,IIRCHK ;CHECK INTERNALS FOR INIT'ED STATE.
4613	046200				INTSET STP
	046200	012737	177601	026106	MOV #^0177400!IOKCKIN,INTMASK ; PRIME FLAG, EXPECT STP
4614	046206	012777	046746	13461'	MOV #STP2,@DPC ;XCT 'STOP', ENABLE STILL SET.
4615	046214	004737	027454		3\$: JSR PC,WAITF
4616	046220	105737	026107		TSTB INTFLAG ;DID WE GET AN INTERRUPT?
4617	046224	001412			BEQ 4\$;BR IF INTERRUPT REC'D.
4618	046226	012701	046214		MOV #3\$,R1 ;SET PC AT WAIT TIME.
4619	046232				HRDERR NSI,INTERR ;INT EXP'D, NOT REC'D.
4620	046252	004737	023422		4\$: JSR PC,IIRCHK ;CHECK INTERNALS FOR INIT'ED STATE
4621	046256				INTSET
	046256	012737	177600	026106	MOV #^0177400!IOKCKIN,INTMASK ; PRIME FLAG, NO INTR. EXPECTED.
4622	046264	012777	046752	134540	MOV #STP3,@DPC ;XCT 'I OFF' AND 'STOP'.
4623	046272	004737	027454		5\$: JSR PC,WAITF
4624	046276	123727	026107	000377	CMPB INTFLAG,#^0377 ;DID WE GET AN INTERRUPT?
4625	046304	001412			BEQ 6\$;BR IF NO INTERRUPT.
4626	046306	012701	046272		MOV #5\$,R1
4627	046312				HRDERR USI,INTERR ;INT REC'D, NOT EXP'D.
4628	046332	004737	023422		6\$: JSR PC,IIRCHK ;CHECK INTERNALS FOR INIT'ED STATE
4629	046336				INTSET
	046336	012737	177600	026106	MOV #^0177400!IOKCKIN,INTMASK ; PRIME FLAG, NO INTR. EXPECTED.
4630	046344	012777	046760	134460	MOV #STP4,@DPC ;XCT 'I ON' AND 'STOP N'.
4631	046352	004737	027454		7\$: JSR PC,WAITF
4632	046356	123727	026107	000377	CMPB INTFLAG,#^0377 ;DID WE GET AN INTERRUPT?
4633	046364	001412			BEQ 8\$;BR IF NO INTERRUPT.
4634	046366	012701	046352		MOV #7\$,R1
4635	046372				HRDERR USI,INTERR ;INT REC'D, NOT EXP'D.
4636	046412	012777	000053	134414	8\$: MOV #FORCSI!WRTCSR,@DSR ;SET BIT TO FORCE STOP INTR.
4637	046420				INTSET STP
	046420	012737	177601	026106	MOV #^0177400!IOKCKIN,INTMASK ; PRIME FLAG, EXPECT STP

11/10/77 -- GRAPHIC SYS. DIAG MACRO M1113 29-AN-81 17:26 PAGE 69-1
... TEST 12 STOP INTERRUPT

SEQ 0124

END TEST 12

REASON FOR - DPL 'TIME-OUT' INTERRUPT

DEFINITION COUNT = 100.)

SEND THE DISPLAY TO A NON-EXISTENT MEMORY LOCATION
AND VERIFY THAT HE RESPONDS WITH A TIME-OUT INTERRUPT.
REPEAT 100 TIMES AFTER QUICK PASS.

FOI/ST: COMDEG

```

JSA      PC,ISTGO      : TITLE.
JSA      PC,DPRESET    : DO SOFT INIT.

```

[illegible]

```

MOV    @CLFLGS,2DSR    ;CLEAR THE FLAGS & PENDING INTERRUPTS.
MOV    @0.WATCSR,2DSR  ;CLEAR THE ERROR CODE.
INTSET
ERR
MOV    #0177400.10RCKIN:10KERR,INTRMASK    ;PRIME FLAG, EXPECT ERR
MOV    #160000,2DPC    ;NOT DISPLAY START...
CLR    R0               ;... SHOULD GET TIME-OUT.

```

```

108:  MOV     $100000, %CR0
118:  CLR     %R0
      DEC     %R0
      BNE     118
      STB     INTFLAG
      BEQ     28
      MOV     $108, %R1
      MERR    NTO, INTERR
      : ... SHOULD GET TIME-OUT.
      : DID WE GET PROPER INTERRUPT?
      : BR IF YES
      : IF DIDN'T, GET ERROR PC...
      : NO TIME-OUT INTERRUPT.

```

[illegible]

```

LOOP TO 18          : REPEAT...
  SA PC.LOOP
  BLS 18
  BR 58              :...TIL LOOPER EXPIRES.

```

```

58:      MOV     #SETPRR:0,DSR          ;INSURE MAIN-SEG RELOC IS 0...
      MOV     #SELDSR,DSR             ;SELECT REAL DSR
      MOV     #DTC,DTCTV              ;...AND RESTORE VECTOR.
      JSR     PC,ISTEND

```

... 47-04

ENC. 155

ENC. 155 16


```

047516 012777 052270 133306      MOV    #ECMPE2,@DPC      ; START THE DPU.
047524 004737 027454          JSR    PC,WAITF        ; WAIT FOR DISPLAY STOP.
4769 047530 105737 026107      TSTB   INTFLAG         ; DID WE GET THE EXPECTED ERROR INTERRUPT?
4770 047534 001412           BEQ     11$              ; BR IF YES.
4771 047536 012701 047524      MOV     #-12,R1       ; NO. SO GET PC FOR ERROR REPORT,
4772 047542          HRDERR NTO,INTERR ; ... AND REPORT THE PROBLEM.
4773 047562          11$:
4774 047562 012701 114003      MOV     #MPE,R1
4775 047566          RDDSRA CSR,R2
047566 012777 000003 133240      MOV     #SELCSR,@DSR ; READ CSR INTO R2 .
047574 017702 133234          MOV     @DSR,R2
4776 047600          IFERROR ECE,CSRR
047600 020102          CMP     R1,R2             ; OK?
047602 001410          BEQ     65$              ; YES.
047604          HRDERR ECE,CSRR,               ; NO.
047624          65$:
4777 047624          ENDSEG                      ; <<<<<<<<<<<<<<<<<<<<<< END SEGMENT.
4778 047626 005737 002502      TST     DPUMOD            ; ARE WE IN DPU-MODE ONLY?
4779 047632 001132          BNE     3$              ; BR IF YES -- CAN'T DO READBCK
4780          ; TO CHECK FOR WRITE-PROTECT VIOLATIONS.
4781
4782          ; TEST MPE - MEMCRY WRITE PROTECTION (MAIN SEGMENT).
4783
4784 047634          BGNSEG
4785 047636          INITDP
047636 004737 025402          JSR     PC,DPINIT        ; GO DO SOFT INIT, CHECK THE STATE.
4786 047642          WTDSRA #BIT9,MPM             ; SET MAIN-SEG WRITE PROTECT
047642 012777 001015 133164      MOV     #BIT9&'C17!SETMPM,@DSR
4787 047650          INTSET ERR
047650 012737 177604 026106      MOV     #'0177400!IOKCKIN!IOKERR,INTMASK ; PRIME FLAG, EXPECT ERR
4788 047656          DPSTART #ECMPE3              ; FORCE MPE BY TRYING DMA READBCK
047656 012777 052302 133146      MOV     #ECMPE3,@DPC      ; START THE DPU.
047664 004737 027454          JSR     PC,WAITF        ; WAIT FOR DISPLAY STOP.
4789 047670 105737 026107      TSTB   INTFLAG         ; DID WE GET THE EXPECTED ERROR INTERRUPT?
4790 047674 001412          BEQ     12$              ; BR IF YES.
4791 047676 012701 047664      MOV     #-12,R1       ; NO. SO GET PC FOR ERROR REPORT,
4792 047702          HRDERR NTO,INTERR ; ... AND REPORT THE PROBLEM.
4793 047722          12$:
4794 047722 012701 114003      MOV     #MPE,R1                ; SET EXPECTED ERROR CODE 'MPE'
4795 047726          RDDSRA CSR,R2
047726 012777 000003 133100      MOV     #SELCSR,@DSR ; READ CSR INTO R2 .
047734 017702 133074          MOV     @DSR,R2
4796 047740          IFERROR ECE,CSRR
047740 020102          CMP     R1,R2             ; OK?
047742 001410          BEQ     66$              ; YES.
047744          HRDERR ECE,CSRR,               ; NO.
047764          66$:
4797 047764          ENDSEG
4798
4799          ; TEST MPE - MEMORY WRITE PROTECTION (AUXILIARY SEGMENT).
4800
4801          BGNSEG
4802 047766          INITDP
4803 047770          JSR     PC,DPINIT        ; GO DO SOFT INIT, CHECK THE STATE.
047770 004737 025402          WTDSRA #BIT9,XPM             ; SET AUX-SEG WRITE PROTECT
4804 047774          MOV     #BIT9&'C17!SETXPM,@DSR
047774 012777 001017 133032

```

[illegible]

4841	050266				DPSTART	#ECRSV2	; START DPU ON ODD Y VALUE.
	050266	012777	052336	132536	MOV	#ECSRVS2,@DPC	; START THE DPU.
	050274	004737	027454		JSR	PC,WAITF	; WAIT FOR DISPLAY STOP.
4842	050300	105737	026107		TSTB	INTFLAG	; DID WE GET THE EXPECTED ERROR INTERRUPT?
4843	050304	001412			BEQ	15\$	; BR IF YES.
4844	050306	012701	050274		MOV	#.-12,R1	; NO. SO GET PC FOR ERROR REPORT,
4845	050312				HRDERR	NTO,INTERR	; ... AND REPORT THE PROBLEM.
4846	050332						
4847	050332	012701	124103		MOV	#RSVDOP!ENECHK,R1	
4848	050336				RDDSRA	CSR,R2	
	050336	012777	000003	132470	MOV	#SELCSR,@DSR	; READ CSR INTO R2 .
	050344	017702	132464		MOV	@DSR,R2	
4849	050350				IFERROR	ECE,CSRRER	
	050350	020102			CMP	R1,R2	; OK?
	050352	001410			BEQ	69\$	; YES.
	050354				HRDERR	ECE,CSRRER,	; NO.
	050374						
4850	050374				ENDSEG		; <<<<<<<<<<<<<<<<<<<<<< END SEGMENT.
4851							
4852							
4853							
4854	050376				BGNSEG		; <<<<<<<<<<<<<<<<<<<<<< BEGIN SEGMENT.
4855	050400				INITDP		
	050400	004737	025402		JSR	PC,DPINIT	; GO DO SOFT INIT, CHECK THE STATE.
4856	050404	012777	000113	132422	MOV	#ENECHK!WRTCSR,@DSR	; ENABLE RSVD-OP ERROR CHECKING.
4857	050412				INTSET	ERR	
	050412	012737	177604	026106	MOV	#^0177400!IOKCKIN!IOKERR,INTMASK	; PRIME FLAG, EXPECT ERR
4858	050420				DPSTART	#ECSRVS3	; START DPU ON ODD X VALUE.
	050420	012777	052346	132404	MOV	#ECSRVS3,@DPC	; START THE DPU.
	050426	004737	027454		JSR	PC,WAITF	; WAIT FOR DISPLAY STOP.
4859	050432	105737	026107		TSTB	INTFLAG	; DID WE GET THE EXPECTED ERROR INTERRUPT?
4860	050436	001412			BEQ	16\$	; BR IF YES.
4861	050440	012701	050426		MOV	#.-12,R1	; NO. SO GET PC FOR ERROR REPORT,
4862	050444				HRDERR	NTO,INTERR	; ... AND REPORT THE PROBLEM.
4863	050464						
4864	050464	012701	124103		MOV	#RSVDOP!ENECHK,R1	
4865	050470				RDDSRA	CSR,R2	
	050470	012777	000003	132336	MOV	#SELCSR,@DSR	; READ CSR INTO R2 .
	050476	017702	132332		MOV	@DSR,R2	
4866	050502				IFERROR	ECE,CSRRER	
	050502	020102			CMP	R1,R2	; OK?
	050504	001410			BEQ	70\$	; YES.
	050506				HRDERR	ECE,CSRRER,	; NO.
	050526						
4867	050526				ENDSEG		; <<<<<<<<<<<<<<<<<<<<<< END SEGMENT.
4868							
4869							
4870							
4871	050530				BGNSEG		; <<<<<<<<<<<<<<<<<<<<<< BEGIN SEGMENT.
4872	050532				INITDP		
	050532	004737	025402		JSR	PC,DPINIT	; GO DO SOFT INIT, CHECK THE STATE.
4873	050536				INTSET	ERR	
	050536	012737	177604	026106	MOV	#^0177400!IOKCKIN!IOKERR,INTMASK	; PRIME FLAG, EXPECT ERR
4874	050544				DPSTART	#ECSEQ1	
	050544	012777	052356	132260	MOV	#ECSEQ1,@DPC	; START THE DPU.
	050552	004737	027454		JSR	PC,WAITF	; WAIT FOR DISPLAY STOP.
4875	050556	105737	026107		TSTB	INTFLAG	; DID WE GET THE EXPECTED ERROR INTERRUPT?

4876	050562	001412		
4877	050564	012701	050552	
4878	050570			
4879	050610			
4880	050610	012701	120003	
4881	050614			
	050614	012777	000003	132212
	050622	017702	132206	
4882	050626			
	050626	020102		
	050630	001410		
	050632			
	050652			
4883	050652			
4884				
4885				
4886				
4887	050654			
4888	050656			
	050656	004737	025402	
4889	050662			
	050662	012737	177604	026106
4890	050670			
	050670	012777	052374	132134
	050676	004737	027454	
4891	050702	105737	026107	
4892	050706	001412		
4893	050710	012701	050676	
4894	050714			
4895	050734			
4896	050734	012701	120003	
4897	050740			
	050740	012777	000003	132066
	050746	017702	132062	
4898	050752			
	050752	020102		
	050754	001410		
	050756			
	050776			
4899	050776			
4900				
4901				
4902				
4903				
4904	051000			
4905	051002			
	051002	004737	025402	
4906	051006			
	051006	012737	177604	026106
4907	051014			
	051014	012777	052442	132010
	051022	004737	027454	
4908	051026	105737	026107	
4909	051032	001412		
4910	051034	012701	051022	
4911	051040			
4912	051060			

```

BEQ      17$          ; BR IF YES.
MOV      #-12,R1     ; NO. SO GET PC FOR ERROR REPORT,
HRDERR   NTO,INTERR  ; ... AND REPORT THE PROBLEM.

17$:
MOV      #SEQERR,R1
RDDSRA   CSR,R2
MOV      #SELCSR,@DSR ; READ CSR INTO R2 .
MOV      @DSR,R2
IFERROR  ECE,CSRRER
CMP      R1,R2        ; OK?
BEQ      71$          ; YES.
HRDERR   ECE,CSRRER,  ; NO.

71$:
ENDSEG    ; <<<<<<<<<<<<<<<<<<<<<<<<<<<< END SEGMENT.

; TEST SEQERR - SEQUENCE ERROR (CHARACTER DATA IN CHAR. SUBROUTINE)

BGNSEG    ; <<<<<<<<<<<<<<<<<<<<<<<<<<<< BEGIN SEGMENT.
INITDP
JSR      PC,DPINIT    ; GO DO SOFT INIT, CHECK THE STATE.
INTSET   ERR
MOV      #^0177400!IOKCKIN.IOKERR,INTMASK ; PRIME FLAG, EXPECT ERR
DPSTART  #ECSEQ2
MOV      #ECSEQ2,@DPC ; START THE DPU.
JSR      PC,WAITF     ; WAIT FOR DISPLAY STOP.
TSTB     INTFLAG      ; DID WE GET THE EXPECTED ERROR INTERRUPT?
BEQ      18$          ; BR IF YES.
MOV      #-12,R1     ; NO. SO GET PC FOR ERROR REPORT,
HRDERR   NTO,INTERR  ; ... AND REPORT THE PROBLEM.

18$:
MOV      #SEQERR,R1
RDDSRA   CSR,R2
MOV      #SELCSR,@DSR ; READ CSR INTO R2 .
MOV      @DSR,R2
IFERROR  ECE,CSRRER
CMP      R1,R2        ; OK?
BEQ      72$          ; YES.
HRDERR   ECE,CSRRER,  ; NO.

72$:
ENDSEG    ; <<<<<<<<<<<<<<<<<<<<<<<<<<<< END SEGMENT.

; TEST SEQERR - SEQUENCE ERROR (DJMS IN CHARACTER SUBROUTINE)

BGNSEG    ; <<<<<<<<<<<<<<<<<<<<<<<<<<<< BEGIN SEGMENT.
INITDP
JSR      PC,DPINIT    ; GO DO SOFT INIT, CHECK THE STATE.
INTSET   ERR
MOV      #^0177400!IOKCKIN!IOKERR,INTMASK ; PRIME FLAG, EXPECT ERR
DPSTART  #ECSEQ3
MOV      #ECSEQ3,@DPC ; START THE DPU.
JSR      PC,WAITF     ; WAIT FOR DISPLAY STOP.
TSTB     INTFLAG      ; DID WE GET THE EXPECTED ERROR INTERRUPT?
BEQ      19$          ; BR IF YES.
MOV      #-12,R1     ; NO. SO GET PC FOR ERROR REPORT,
HRDERR   NTO,INTERR  ; ... AND REPORT THE PROBLEM.

19$:

```

```

MOV      #SEQERR,R1
RDDSRA   CSR,R2
MOV      #SELCSR,@DSR      ; READ CSR INTO R2 .
MOV      @DSR,R2
IFERROR  ECE,CSRRER
CMP      R1,R2              ; OK?
BEQ      73$                ; YES.
HRDERR   ECE,CSRRER,        ; NO.
73$:
ENDSEG    ; <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<< END SEGMENT.

; TEST SEQERR - SEQUENCE ERROR (BIT-MAP-0 IN SUBROUTINE)

BGNSEG    ; <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<< BEGIN SEGMENT.
INITDP
JSR      PC,DPINIT          ; GO DO SOFT INIT, CHECK THE STATE.
INTSET   ERR
MOV      ^^0177400!IOCKIN!IOKERR,INTMASK      ; PRIME FLAG, EXPECT ERR
DPSTART  #ECSEQ4
MOV      #ECSEQ4,@DPC       ; START THE DPU.
JSR      PC,WAITF           ; WAIT FOR DISPLAY STOP.
TSTB     INTFLAG            ; DID WE GET THE EXPECTED ERROR INTERRUPT?
BEQ      20$                ; BR IF YES.
MOV      #-12,R1             ; NO. SO GET PC FOR ERROR REPORT,
HRDERR   NTO,INTERR         ; ... AND REPORT THE PROBLEM.
20$:
MOV      #SEQERR,R1
RDDSRA   CSR,R2
MOV      #SELCSR,@DSR      ; READ CSR INTO R2 .
MOV      @DSR,R2
IFERROR  ECE,CSRRER
CMP      R1,R2              ; OK?
BEQ      74$                ; YES.
HRDERR   ECE,CSRRER,        ; NO.
74$:
ENDSEG    ; <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<< END SEGMENT.

; TEST SEQERR - SEQUENCE ERROR (BIT-MAP-1 IN SUBROUTINE)

BGNSEG    ; <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<< BEGIN SEGMENT.
INITDP
JSR      PC,DPINIT          ; GO DO SOFT INIT, CHECK THE STATE.
INTSET   ERR
MOV      ^^0177400.IOCKIN!IOKERR,INTMASK      ; PRIME FLAG, EXPECT ERR
DPSTART  #ECSEQ5
MOV      #ECSEQ5,@DPC       ; START THE DPU.
JSR      PC,WAITF           ; WAIT FOR DISPLAY STOP.
TSTB     INTFLAG            ; DID WE GET THE EXPECTED ERROR INTERRUPT?
BEQ      21$                ; BR IF YES.
MOV      #-12,R1             ; NO. SO GET PC FOR ERROR REPORT,
HRDERR   NTO,INTERR         ; ... AND REPORT THE PROBLEM.
21$:
MOV      #SEQERR,R1
RDDSRA   CSR,R2
MOV      #SELCSR,@DSR      ; READ CSR INTO R2 .

```

```

051342 017702 131466      MOV @DSR,R2
4949 051346      IFERROR ECE,CSRER
      051346      CMP R1,R2          ; OK?
      051350      BEQ 75$           ; YES.
      051352      HRDERR ECE,CSRER,    ; NO.
      051372      75$:
      051372      ENDSEG              ; <<<<<<<<<<<<<<<<<<<<<< END SEGMENT.

; TEST SEQERR - SEQUENCE ERROR (DMA READBACK FROM SUBROUTINE)

BGNSEG              ; <<<<<<<<<<<<<<<<<<<<<< BEGIN SEGMENT.
INITDP
JSR PC,DPINIT       ; GO DO SOFT INIT, CHECK THE STATE.
INTSET ERR
MOV #^0177400!IOKCKIN!IOKERR,INTMASK ; PRIME FLAG, EXPECT ERR
DPSTART #ECSEQ6
MOV #ECSEQ6,@DPC    ; START THE DPU.
JSR PC,WAITF        ; WAIT FOR DISPLAY STOP.
TSTB INTFLAG        ; DID WE GET THE EXPECTED ERROR INTERRUPT?
BEQ 22$             ; BR IF YES.
MOV #-12,R1         ; NO. SO GET PC FOR ERROR REPORT,
HRDERR NTO,INTERR   ; ... AND REPORT THE PROBLEM.

22$:
MOV #SEQERR,R1
RDDSRA CSR,R2
MOV #SELCSR,@DSR    ; READ CSR INTO R2 .
MOV @DSR,R2
IFERROR ECE,CSRER
CMP R1,R2           ; OK?
BEQ 76$             ; YES.
HRDERR ECE,CSRER,    ; NO.

76$:
ENDSEG              ; <<<<<<<<<<<<<<<<<<<<<< END SEGMENT.

; TEST SEQERR - SEQUENCE ERROR (ILLEGAL DMA READBACK RESTART)

BGNSEG              ; <<<<<<<<<<<<<<<<<<<<<< BEGIN SEGMENT.
INITDP
JSR PC,DPINIT       ; GO DO SOFT INIT, CHECK THE STATE.
INTSET ERR
MOV #^0177400!IOKCKIN!IOKERR,INTMASK ; PRIME FLAG, EXPECT ERR
DPSTART #ECSEQ7
MOV #ECSEQ7,@DPC    ; START THE DPU.
JSR PC,WAITF        ; WAIT FOR DISPLAY STOP.
TSTB INTFLAG        ; DID WE GET THE EXPECTED ERROR INTERRUPT?
BEQ 23$             ; BR IF YES.
MOV #-12,R1         ; NO. SO GET PC FOR ERROR REPORT,
HRDERR NTO,INTERR   ; ... AND REPORT THE PROBLEM.

23$:
MOV #SEQERR,R1
RDDSRA CSR,R2
MOV #SELCSR,@DSR    ; READ CSR INTO R2 .
MOV @DSR,R2
IFERROR ECE,CSRER
CMP R1,R2           ; OK?

```

```

051620 0014'0      BEQ     77$           ; YES.
051622          HRDERR ECE,CSRER,       ; NO.
051642          77$:      ENDSEG         ; <<<<<<<<<<<<<<<<<<<<<<<<<<<< END SEGMENT.
4984 051642          ; TEST SYNCTO - SYNC TIMEOUT FROM IMAGE MEMORY.
4985
4986
4987
4988 051644          BGNSEG         ; <<<<<<<<<<<<<<<<<<<<<<<<<<<< BEGIN SEGMENT.
4989 051646          INITDP
051646 004737 025402 JSR     PC,DPINIT        ; GO DO SOFT INIT, CHECK THE STATE.
4990 051652          INTSET ERR
051652 012737 177604 026106 MOV    #^0177400!IOKCKIN!IOKERR,INTMASK ; PRIME FLAG, EXPECT ERR
4991 051660          DPSTART #ECSTO
051660 012777 052620 131144 MOV    #ECSTO,@DPC        ; START THE DPU.
051666 004737 027454 JSR     PC,WAITF         ; WAIT FOR DISPLAY STOP.
4992 051672 105737 026107 TSTB   INTFLAG         ; DID WE GET THE EXPECTED ERROR INTERRUPT?
4993 051676 001412 BEQ     24$           ; BR IF YES.
4994 051700 012701 051666 MOV    #-12,R1         ; NO. SO GET PC FOR ERROR REPORT,
4995 051704          HRDERR NTO,INTERR ; ... AND REPORT THE PROBLEM.
4996 051724          24$:      MOV     #SYNCTO,R1
4997 051724 012701 140003 RDDSRA  CSR,R2
051730 012777 000003 131076 MOV    #SELCsr,@DSR      ; READ CSR INTO R2 .
051736 017702 131072 MOV    @DSR,R2
4999 051742          IFERROR ECE,CSRER
051742 020102 CMP     R1,R2          ; OK?
051744 001410 BEQ     78$           ; YES.
051746          HRDERR ECE,CSRER,       ; NO.
051766          78$:      ENDSEG         ; <<<<<<<<<<<<<<<<<<<<<<<<<<<< END SEGMENT.
5000 051766
5001
5002          ; TEST DRDYTO - DATA READY TIMEOUT ON CURSOR READBACK
5003
5004 051770          BGNSEG
5005 051772          INITDP
051772 004737 025402 JSR     PC,DPINIT        ; GO DO SOFT INIT, CHECK THE STATE.
5006 051776 105037 026106 CLRB   INTMASK
5007 052002 012777 040000 131026 MOV    #GTJSSW,@DXR      ; SAY READ PENDING COORDS (BUT THERE ARE NONE)
5008 052010 004737 027454 JSR     PC,WAITF         ; WAIT FOR READY
5009 052014 012701 150003 MOV    #DRDYTO,R1        ; SET EXPECTED ERROR CODE 'DRDYTO' INTO R1
5010 052020          RDDSRA  CSR,R2        ; GET THE CSR
052020 012777 000003 131006 MOV    #SELCsr,@DSR      ; READ CSR INTO R2 .
052026 017702 131002 MOV    @DSR,R2
5011 052032          IFERROR ECE,CSRER
052032 020102 CMP    R1,R2          ; OK?
052034 001410 BEQ     79$           ; YES.
052036          HRDERR ECE,CSRER,       ; NO.
052056          79$:      ENDSEG
5012 052056
5013
5014          ; TEST DAVTO - DATA AVAILABLE TIMEOUT PIXEL READBACK.
5015
5016 052060          BGNSEG
5017 052062          INITDP
052062 004737 025402 JSR     PC,DPINIT        ; GO DO SOFT INIT, CHECK THE STATE.
5018 052066 012777 040000 130744 MOV    #40000,@DYR       ; PROTECT ALL MEMORYS.
5019 052074 004737 027454 JSR     PC,WAITF

```

5020	052100	012777	040400	130732		MOV	#40400,@DVR	:
5021	052106	004737	027454			JSR	PC,WAITF	:
5022	052112	012777	041000	130720		MOV	#41000,@DVR	:
5023	052120	004737	027454			JSR	PC,WAITF	:
5024	052124	012777	041400	130706		MOV	#41400,@DVR	:
5025	052132	004737	027454			JSR	PC,WAITF	:
5026	052136	012777	000000	130672		MOV	#0,@DXR	: TRY PIXEL READ.
5027	052144	004737	027454			JSR	PC,WAITF	:
5028	052150	103410				BCS	2\$	: BRANCH IF UNSUCCESSFUL.
5029	052152					DFERR	PR3H	: PIXEL READBACK HUNG UP.
5030	052172	012701	144003		2\$:	MOV	#DAVTO,R1	
5031	052176					RDDSR	CSR,R2	
	052176	012777	000003	130630		MOV	#SELCSR,@DSR	: READ CSR INTO R2 .
	052204	017702	130624			MOV	@DSR,R2	
5032	052210					IFERROR	ECE,CSRR	
	052210	020102				CMP	R1,R2	: OK?
	052212	001410				BEQ	80\$	: YES.
	052214					HRDERR	ECE,CSRR,	: NO.
	052234				80\$:			
5033	052234					ENDSEG		; <<<<<<<<<<<<<<<<<<<<<< END SEGMENT.
5034						COMEND	1\$	
5035	052236					JSR	PC,LOOP	: REPEAT 'TIL LOOPER EXPIRES.
	052236	004737	027646			BCS	81\$	
	052242	103401				BR	82\$	
	052244	000402						
	052246	000137	047342		81\$:	JMP	1\$	
	052252	004737	030072		82\$:	JSR	PC,TSTEND	: PRINT ERROR SUMMARY, IF REQ'D.
5036	052256					EXIT	TST	: EXIT THIS TEST >>>>>>>>>>>>>>>>>>>>>>>>>
5037								
5038								
5039								
5040								
5041	052262	000002				ECISVC:	RTI	: RETURN FROM INTERUPT.
5042								
5043								
5044								
5045								
5046	052264	164000				ECMPE1:	DNOP	: MPE - MEMORY PROTECTION ERROR.
5047	052266	173000					STOPN	: ... ON MAIN SEGMENT
5048								
5049	052270	160001	052277			ECMPE2:	DJMS,1\$,AUXSEG	: CALL AUXILIARY SEGMENT
5050	052274	173000					STOPN	
5051	052276	164000			1\$:		DNOP	: SHOULD GET MPE ERROR
5052	052300	173000					STOPN	
5053								
5054	052302	162000				ECMPE3:	IMREAD	: TRY DMA READBACK
5055	052304	000001	000001	000001			1,1,1,160000	: ... TO WRITE PROTECTED MAIN SEG.
5056	052314	173000					STOPN	
5057								
5058	052316	162000				ECMPE4:	IMREAD	: TRY DMA READBACK
5059	052320	000001	000001	000001			1,1,1,160000.AUXSEG	: ... TO WRITE-PROTECTED AUX SEG.
5060	052330	173000					STOPN	
5061								
5062	052332	161777				ECRSV1:	161777	: RSVDOP - RESERVED OPCODE, OPERATION.
5063	052334	173000					STOPN	

5064									
5065	052336	114000	000000	000001	ECRSV2:	APNT,0,1			; RSVDOP - ODD Y VALUE.
5066	052344	173000				STOPN			
5067									
5068	052346	114000	000001	000000	ECRSV3:	APNT,1,0			; RSVDOP - ODD X VALUE.
5069	052354	173000				STOPN			
5070									
5071	052356	160001	052364		ECSEQ1:	DJMS,1\$			; SEQERR - SEQUENCE ERROR.
5072	052362	173000				STOPN			
5073	052364	160001	052370		1\$:	DJMS,2\$			;DJMS FROM SUBROUTINE.
5074	052370	173000			2\$:	STOPN			
5075	052372	173000				STOPN			
5076									
5077	052374	114000	000000	000000	ECSEQ2:	APNT,0,0			; SEQERR.
5078	052402	152000	052414			SETCB,2\$			
5079	052406	103774				CHAR!ALL			
5080	052410	000000			1\$:	0			
5081	052412	173000				STOPN			
5082	052414	000000			2\$:	0			;DATA AS 1ST WORD -- STILL IN CHAR MODE.
5083	052416	000000	000000	000000		0,0,0,0,0,0,0,0			
5084	052436	165000				DPOP			
5085	052440	173000				STOPN			
5086									
5087	052442	114000	000000	000000	ECSEQ3:	APNT,0,0			; SEQERR.
5088	052450	152000	052462			SETCB,2\$			
5089	052454	103774				CHAR!ALL			
5090	052456	000000			1\$:	0			
5091	052460	173000				STOPN			
5092	052462	160001	052470		2\$:	DJMS,3\$			;DJMS IN CHAR SUBROUTINE.
5093	052466	173000				STOPN			
5094	052470	173000			3\$:	STOPN			
5095	052472	173000				STOPN			
5096									
5097	052474	170140			ECSEQ4:	CLRMEM			; SEQERR.
5098	052476	114000	000000	000000		APNT,0,0			
5099	052504	160001	052512			DJMS,1\$			
5100	052510	173000				STOPN			
5101	052512	134000			1\$:	BM04			
5102	052514	052514			2\$:	2\$			
5103	052516	173000				STOPN			
5104									
5105	052520	114000	000000	000000	ECSEQ5:	APNT,0,0			; SEQERR.
5106	052526	160001	052534			DJMS,1\$			
5107	052532	173000				STOPN			
5108	052534	136000			1\$:	BM14			
5109	052536	052536			2\$:	2\$			
5110	052540	173000				STOPN			
5111									
5112	052542	114000	000000	000000	ECSEQ6:	APNT,0,0			; SEQERR.
5113	052550	160001	052556			DJMS,1\$			
5114	052554	173000				STOPN			
5115	052556	162000			1\$:	IMREAD			;DMA READBACK FROM SUBROUTINE.
5116	052560	000001				1			
5117	052562	000001				1			
5118	052564	000001				1			
5119	052566	052614				ECSEQ7B			
5120	052570	173000				STOPN			

```
5121
5122 052572 114000 000000 000000 ECSEQ7: APNT,0,0
5123 052600 162001 IMREAD!1 ;ILLEGAL DMA READBACK RESTART.
5124 052602 000001 1
5125 052604 052614 ECSQ7B
5126 052606 173000 STOPN
5127 052610 162101 IMREAD!101
5128 052612 173000 STOPN
5129 052614 ECSQ7B: .BLKW 2
5130
5131 052620 176000 ECSTO: PROTEC!CH0
5132 052622 176400 PROTEC!CH1
5133 052624 177000 PROTEC!CH2
5134 052626 177400 PROTEC!CH3
5135 052630 114000 040000 000000 APNT,I.0,0
5136 052636 173000 STOPN
5137
5138 052640 END.TEST
```

```
*****
*
* END TEST 15
*
*****
```

BEGIN TEST 16 - ADDRESS RELOCATE

: (ITERATION COUNT = 20.)

```

: TEST ALL COMBINATIONS OF DPC AND RELOCATE FACTOR
: WHERE  $DPC + REL = RELT1$ , A LOCATION WITHIN THE PROGRAM
: TEST RELIES OF THE FACT THAT  $(DPC + REL) > 18$  BITS WILL
: WRAP AROUND IN THE FINAL DPC.
: REPEAT 20 TIMES AFTER QUICK PASS.

```

RELIST: COMBEG

```

CSR      PC,TSTGO      ; TITLE.
CSR      PC,DPRESET    ; DO SOFT INIT.
MOV      #20$,EXTA     ; SET ERROR EXTENSION ADDRESS.
MOV      #SELSR,@DSR
MOV      @DSR,R3
BIC      #^C1777,R3
BIS      #STOP,R3      ;...AND STATUS.

```

```
58:  MOV    #RELT1,R4      :SET INITIAL DPC
      MOV    #RELT1+2,R2  :SET INITIAL EXPECTED DPC
      MOV    #SETMRR.0,R5 :SET INITIAL RELOCATION (0)
```

[illegible]

```

MOV      R5,@DRR      ;SET 1ST (NEXT) REL VALUE...
MOV      R4,@DPC      ;...XCT -- RELOCATE TO 'REL1'
JSR      PC,WAITF
CMP      @DPC,R2       ;CHECK DPC = R4 (INITIAL) + 2
BNE      2$
CMP      @DSR,R3       ;...AND DSR = STOP+PIXEL DATA.

```

[illegible]

```

CMP      R5,#SETMRR.177760      ;MAX RELOCATION DONE?
BEQ      48                      ;ALL DONE IF SO.

```

```

ADD      #BIT4,R5      ;BUMP RELOCATION BY 1 (32 WORDS)
SUB      #100,R2        ;REDUCE EXPECTED...
SUB      #100,R4        ;... AND INITIAL DPC BY 32 WORDS)
BPL      1$             ;CONTINUE 'TIL WE UNDERFLOW
TST      R5              ;SEE IF ALREADY FORCED HIGH BITS
BMI      1$             ;JUST CONTINUE IF YES
BIS      #140000,R5      ;FORCE HIGH RELOCATION BITS
BR       1$             ;... THEN CONTINUE

```

```
48:      LOOPTO 5$          ; REPEAT 'TIL LOOPER EXPIRES.
      JSR     PC,LOOP
      BCS     5$
```


5201
5202
5203
5204
5205
5206
5207
5208
5209
5210
5211
5212
5213
5214
5215
5216
5217
5218
5219
5220

053250	012737	053754	053732
053256	012737	053702	022500
053264	013701	003240	
053270	010137	053750	
053274	005002		
053276	012721	100000	
053302	110221		
053304	005202		
053306	020227	000200	
053312	001373		
053314	012721	160000	
053320	012721	053752	
053324	017703	127504	
053330	042703	176000	
053334	052703	172000	
053340			
053342	013737	053732	053744
053350	013737	053732	053762
053356	005001		
053360	013702	053732	
053364	042702	177776	
053370	052702	053760	

[illegible]

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

5300					20\$: PRINTX #CHRF,CT.BAS,R1 ; ERROR EXTENSION.
5301	053702				RTS PC
5302	053730	000207			
5303					CHSEG: CT.SUB ; CHARACTER BASE CONTROL, WITH SEGMENT SELECT.
5304	053732	053754			
5305					DISPLAY CODE FOR CHAR MODE TEST.
5306					
5307					
5308	053734	164000	164000	164000	CT: DNOP,DNOP,DNOP
5309	053742	152000	053754		SETCB,CT.SUB ; INITIAL BASE ADDR = CT.SUB.
5310	053746	160000			DJMP ; XFER PC TO THE CHAR...
5311	053750	000000			CT.ASC: 0 ; ...DATA ARRAY.
5312	053752	173000			CT.RET: STOPN ; FINAL 'DPOP' SHOULD RETURN HERE
5313					
5314	053754	053756			CT.SUB: .+2 ; SUBROUTINE ENTRY PONTNER.
5315	053756	173000			STOPN ; STOP HERE IF CHAR ADDRESSES...
5316					
5317	053760	152000			SETCB ; ...COME OUT RIGHT.
5318	053762	053754			CT.BAS: CT.SUB ; ON RESUME, SET NEW BASE ADDR...
5319	053764	165000			DPOP ; ...AND DPOP TO NEXT CHAR.
5320					
5321					; DISPLAY FILE FOR CHARACTER DISPLAY:
5322					
5323	053766	176074	176474	177074	CHDISP: RDWRT!CH0,RDWRT.CH1,RDWRT!CH2,RDWRT!CH3
5324	053776	170140			CLRMEM
5325	054000	152000	110146		SETCB, ACAT
5326	054004	117774	001000	001500	APNT!ALL,1000,1500
5327	054012	100000			CHAR
5328	054014	040	041	042	.ASCII ? !'#\$%&'()*+,-./?
5329					.EVEN
5330	054034	114000	001000	001400	APNT,1000,1400
5331	054042	100000			CHAR
5332	054044	060	061	062	.ASCII /0123456789/
5333	054056	114000	001000	001300	APNT,1000,1300
5334	054064	100000			CHAR
5335	054066	072	073	074	.ASCII /:;<=>?@ /
5336					.EVEN
5337	054076	114000	001000	001200	APNT,1000,1200
5338	054104	100000			CHAR
5339	054106	101	102	103	.ASCII /ABCDEFGHIJKLM /
5340					.EVEN
5341	054124	114000	001000	001100	APNT,1000,1100
5342	054132	100000			CHAR
5343	054134	116	117	120	.ASCII /NOPQRSTUVWXYZ /
5344					.EVEN
5345	054152	114000	001000	001000	APNT,1000,1000
5346	054160	100000			CHAR
5347	054162	133	134	135	.ASCII /[\\]^_ /
5348					.EVEN
5349					
5350	054170	117774	001000	000602	APNT!ALL,1000,602
5351	054176	100000			CHAR
5352	054200	040	041	042	.ASCII ? !'#\$%&'()*+,-./?
5353					.EVEN
5354	054220	114000	001000	000502	APNT,1000,502
5355	054226	100000			CHAR
5356	054230	060	061	062	.ASCII /0123456789/

5378				
5379				
5380				
5381				
5382				
5383				
5384				
5385				
5386				
5387				
5388	054430	004737	027702	
5389	054434	012737	054740	022300
5390	054442	004737	027054	
5391				
5392	054446	005737	002502	
5393	054452	001002		
5394	054454	004737	026650	
5395				
5396	054460	012737	177777	003232
5397	054466	013737	027700	003234
5398	054474	005004		
5399	054476	005005		
5400	054500	005037	055010	
5401	054504	053737	003064	055010
5402	054512	005037	055012	
5403				
5404	054516			
5405				
5406	054520	012777	055002	126304
5407	054526	004737	027454	
5408	054532	027704	126300	
5409	054536	001003		
5410	054540	027705	126274	
5411	054544	001410		
5412	054546			
5413	054566			
	054566	004737	027646	
	054572	103752		
5414	054574			
5415				
5416	054576	013737	003234	027700
5417	054604	005737	003232	
5418	054610	003027		
5419	054612	001413		
5420				
5421	054614	062704	000002	
5422	054620	062737	000002	055010
5423	054626	020427	001776	

[illegible]


```
*****  
: *  
: * BEGIN TEST 19 - LONG VECTORS  
: *  
: *****
```

```

: GENERATE LONG VECTORS FROM 0,0 VARYING THE LENGTH FROM
: MAX(X) TO HALF(XY) TO MAX(Y).
: THEN REPEAT FROM MAX,MAX USING NEGATIVE DELTAS.

```

[THIS MAY BE WORST CASE !!!]

REPEAT EACH VECTOR 10 TIMES AFTER QUICK PASS.

```

LVTST:  JSR      PC,TSTGO          ;TITLE
          MOV     #20$,EXTA        ; SET ERROR EXTENSION ADDRESS.
          JSR     PC,LUMIN         ; BLAST THE LUT.

```

TST DPUMOD

BNE .+6

JSR PC,CLRRW

CLR LV+2

CLR
CLR LV+4

MOV #MAXX,R4

CLR R5

MOV R4,LVX

BIS OPTI, LVX

MOV R5, L'VY

MOV #2,R2
MOV #3,R3

```
MOV     #2,R3
MOV     LOOPK,TEMP2
```

[illegible]

```
9$:      MOV      #LV,@DPC      ;START 'EM UP.
```

JSR PC, WAITF

```
CMP    @DXR,R4          ;CHECK DXR = R4...
```

BNE 2\$
CND 20X2 25 AND DXP = 25

```

CMP      @DYP,R5          ;...AND DYP = R5.
BEQ      3$

```

28. BEU 35
HDDERR LIVE XYERR : LONG VECTOR FAILS

```
2$: HARDEN LVP, XTERA , LONG VECTO
3$: LGOPTO 9$ : REPEAT...
```

```

39.      LCOM TO      79
        JSR          PC.LOOP

```

BCS 95

[illegible]

```
MOV     TEMP2,LOOPK      ;RESET LOOPER.
```

BIT LVX.#1777

```
BEQ      4$          ;BR IF DELTA X = 0.
```

```

SUB      #2,LVX          ;OTHERWISE, DECR DX...

```

```
ADD      #2,LVY      ;...AND INCR DY.
ADD      #2,DY        ;SET NEXT EXP. Y
```

ADD R2,R4

 *
 *
 * END TEST 20 *
 *

 *
 * BEGIN TEST 21 - SHORT VECTORS
 *

```

: VERIFY THAT WE CAN EXECUTE SHORT VECTORS FROM 1000,1000
: FOR ALL VALUES OF DELTA X AND Y.
: SWEEP VECTORS FROM MAX,MAX, CCW THRU 360 DEGREES.
:
: REPEAT EACH VECTOR 10 TIMES AFTER QUICK PASS.

```

```
SVTST: JSR PC,TSTGO ;TITLE
MOV #20$,EXTA ; SET ERROR EXTENSION ADDRESS.
JSR PC,LUMIN ; BLAST LUT.
TST DPUMOD
BNE .+6 ; SKIP NEXT IF DPU-ONLY.
JSR PC,CLRRW ; CLEAR AND SET R/W MODE.
MOV #1074,R4 ;INIT EXPECTED X...
MOV #1076,R5 ;...AND Y TOO.
MOV #17076,SVD ;INIT DX +74, DX +76.
BIS OPT!,SVD ;...INT OPTIONAL.
CLR TEMP1 ;USE AS XFER SWITCH.
MOV LOOPK,TEMP2 ;SAVE A COPY OF LOOPER.
```

1\$: BGNSEG ; <<<<<<<<<<<<<<<<<<<<<<<<<<<< BEGIN SEGMENT.

9\$: MOV #SVT,@DPC ;START DISPLAY
JSR PC,WAITF
CMP @DXR,R4 ;CHECK DXR = R4...
BNE 2\$
CMP @DYR,R5 ;...AND DYR = R5.
BEQ 3\$

2\$: HRDERR SVF,XVERR ;SHORT VECTOR FAILS.
BR 30\$

3\$: LOOPTO 9\$; REPEAT...
JSR PC,LOOP
BCS 9\$

30\$: ENDSEG ; <<<<<<<<<<<<<<<<<<<<<<<<<<<< END SEGMENT.

MOV TEMP2,LOOPK ;RESET LOOPER.
TST TEMP1 ;TEST XFER SWITCH...
BNE 7\$;...BR IF SET.
CMP R4,#1000-76
BEQ 5\$;BR IF X MIN.
SUB #2,R4 ;OTHERWISE, DECR X.
MOV R4,R0
JSR PC,SDX ;TRANSLATE TO DX.
BIC #37600,SVD ;CLEAR OLD...
BIS R0,SVD ;...SET NEW DX...
BR 1\$;...AND CONTINUE.

5\$: CMP R5,#1000-76

				Mov	R5,R1	; R1 = EXPECTED DXR.
5770	057124	010501		ASL	R1	:
5771	057126	006301		ASL	R1	:
5772	057130	006301		Mov	#0,R3	: R3 - EXPECTED DYR.
5773	057132	012703	000000	Mov	@DXR,R2	: READ DXR INTO R2.
5774	057136	017702	123674	Mov	@DYR,R4	: READ DYR INTO R4.
5775	057142	017704	123672	IERRX2	RLX2IE,RLXYE	:
5776	057146			CMP	R1,R2	: 1ST ITEM OK?
	057146	020102		BNE	67\$	: NO.
	057150	001002		CMP	R3,R4	: 2ND ITEM OK?
	057152	020304		BEQ	66\$	: YES.
	057154	001410		HRDERR	RLX2IE,RLXYE,	: NO.
	057156					
	057176					
5777	057176	005205		INC	R5	: INCREMENT X VALUE.
5778	057200	020527	000200	CMP	R5,#200	: DONE?
5779	057204	001340		BNE	3\$	: NO.
5780	057206			ENDSEG		; <<<<<<<<<<<<<<<<<<< END SEGMENT.
5781						
5782						
5783	057210			; TEST Y UP.		
5784	057212			RLNEWL:	BGNSEG	; <<<<<<<<<<<<<<<<<<< BEGIN SEGMENT.
	057212			DSTART	#YUP	
	057212	012777	057644	Mov	#YUP,@DPC	: START THE DPU.
	057220	004737	027454	JSR	PC,WAITF	: WAIT FOR DISPLAY STOP.
5785	057224	012701	000000	Mov	#0,R1	: R1 = EXPECTED DXR.
5786	057230	012703	000002	Mov	#2,R3	: R3 = EXPECTED DYR.
5787	057234	017702	123576	Mov	@DXR,R2	: READ DXR INTO R2.
5788	057240	017704	123574	Mov	@DYR,R4	: READ DYR INTO R4.
5789	057244			IERRX2	RLYEUE,RLXYE	:
	057244	020102		CMP	R1,R2	: 1ST ITEM OK?
	057246	001002		BNE	65\$	: NO.
	057250	020304		CMP	R3,R4	: 2ND ITEM OK?
	057252	001410		BEQ	64\$	: YES.
	057254			HRDERR	RLYEUE,RLXYE,	: NO.
	057274					
5790	057274			ENDSEG		; <<<<<<<<<<<<<<<<<<< END SEGMENT.
5791						
5792						
5793	057276			; TEST Y DOWN.		
5794	057300			BGNSEG		; <<<<<<<<<<<<<<<<<<< BEGIN SEGMENT.
	057300			DSTART	#YDN	
	057300	012777	057660	Mov	#YDN,@DPC	: START THE DPU.
	057306	004737	027454	JSR	PC,WAITF	: WAIT FOR DISPLAY STOP.
5795	057312	012701	000000	Mov	#0,R1	: R1 = EXPECTED DXR.
5796	057316	012703	000000	Mov	#0,R3	: R3 = EXPECTED DYR.
5797	057322	017702	123510	Mov	@DXR,R2	: READ DXR INTO R2.
5798	057326	017704	123506	Mov	@DYR,R4	: READ DYR INTO R4.
5799	057332			IERRX2	RLYDE,RLXYE	:
	057332	020102		CMP	R1,R2	: 1ST ITEM OK?
	057334	001002		BNE	67\$	: NO.
	057336	020304		CMP	R3,R4	: 2ND ITEM OK?
	057340	001410		BEQ	66\$	: YES.
	057342			HRDERR	RLYDE,RLXYE,	: NO.
	057362					
5800	057362			ENDSEG		; <<<<<<<<<<<<<<<<<<< END SEGMENT.
5801						
5802						
5803	057364			; TEST SKIP UP.		
5804	057366			BGNSEG		; <<<<<<<<<<<<<<<<<<< BEGIN SEGMENT.
				DSTART	#Y2UP	

Line	Address	Op Code	Op 1	Op 2	Op 3	Op 4	Op 5	Op 6	Op 7	Op 8	Op 9	Op 10	Op 11	Op 12	Op 13	Op 14	Op 15	Op 16	Op 17	Op 18	Op 19	Op 20	Op 21	Op 22	Op 23	Op 24	Op 25	Op 26	Op 27	Op 28	Op 29	Op 30	Op 31	Op 32	Op 33	Op 34	Op 35	Op 36	Op 37	Op 38	Op 39	Op 40	Op 41	Op 42	Op 43	Op 44	Op 45	Op 46	Op 47	Op 48	Op 49	Op 50	Op 51	Op 52	Op 53	Op 54	Op 55	Op 56	Op 57	Op 58	Op 59	Op 60	Op 61	Op 62	Op 63	Op 64	Op 65	Op 66	Op 67	Op 68	Op 69	Op 70	Op 71	Op 72	Op 73	Op 74	Op 75	Op 76	Op 77	Op 78	Op 79	Op 80	Op 81	Op 82	Op 83	Op 84	Op 85	Op 86	Op 87	Op 88	Op 89	Op 90	Op 91	Op 92	Op 93	Op 94	Op 95	Op 96	Op 97	Op 98	Op 99	Op 100	Op 101	Op 102	Op 103	Op 104	Op 105	Op 106	Op 107	Op 108	Op 109	Op 110	Op 111	Op 112	Op 113	Op 114	Op 115	Op 116	Op 117	Op 118	Op 119	Op 120	Op 121	Op 122	Op 123	Op 124	Op 125	Op 126	Op 127	Op 128	Op 129	Op 130	Op 131	Op 132	Op 133	Op 134	Op 135	Op 136	Op 137	Op 138	Op 139	Op 140	Op 141	Op 142	Op 143	Op 144	Op 145	Op 146	Op 147	Op 148	Op 149	Op 150	Op 151	Op 152	Op 153	Op 154	Op 155	Op 156	Op 157	Op 158	Op 159	Op 160	Op 161	Op 162	Op 163	Op 164	Op 165	Op 166	Op 167	Op 168	Op 169	Op 170	Op 171	Op 172	Op 173	Op 174	Op 175	Op 176	Op 177	Op 178	Op 179	Op 180	Op 181	Op 182	Op 183	Op 184	Op 185	Op 186	Op 187	Op 188	Op 189	Op 190	Op 191	Op 192	Op 193	Op 194	Op 195	Op 196	Op 197	Op 198	Op 199	Op 200	Op 201	Op 202	Op 203	Op 204	Op 205	Op 206	Op 207	Op 208	Op 209	Op 210	Op 211	Op 212	Op 213	Op 214	Op 215	Op 216	Op 217	Op 218	Op 219	Op 220	Op 221	Op 222	Op 223	Op 224	Op 225	Op 226	Op 227	Op 228	Op 229	Op 230	Op 231	Op 232	Op 233	Op 234	Op 235	Op 236	Op 237	Op 238	Op 239	Op 240	Op 241	Op 242	Op 243	Op 244	Op 245	Op 246	Op 247	Op 248	Op 249	Op 250	Op 251	Op 252	Op 253	Op 254	Op 255	Op 256	Op 257	Op 258	Op 259	Op 260	Op 261	Op 262	Op 263	Op 264	Op 265	Op 266	Op 267	Op 268	Op 269	Op 270	Op 271	Op 272	Op 273	Op 274	Op 275	Op 276	Op 277	Op 278	Op 279	Op 280	Op 281	Op 282	Op 283	Op 284	Op 285	Op 286	Op 287	Op 288	Op 289	Op 290	Op 291	Op 292	Op 293	Op 294	Op 295	Op 296	Op 297	Op 298	Op 299	Op 300	Op 301	Op 302	Op 303	Op 304	Op 305	Op 306	Op 307	Op 308	Op 309	Op 310	Op 311	Op 312	Op 313	Op 314	Op 315	Op 316	Op 317	Op 318	Op 319	Op 320	Op 321	Op 322	Op 323	Op 324	Op 325	Op 326	Op 327	Op 328	Op 329	Op 330	Op 331	Op 332	Op 333	Op 334	Op 335	Op 336	Op 337	Op 338	Op 339	Op 340	Op 341	Op 342	Op 343	Op 344	Op 345	Op 346	Op 347	Op 348	Op 349	Op 350	Op 351	Op 352	Op 353	Op 354	Op 355	Op 356	Op 357	Op 358	Op 359	Op 360	Op 361	Op 362	Op 363	Op 364	Op 365	Op 366	Op 367	Op 368	Op 369	Op 370	Op 371	Op 372	Op 373	Op 374	Op 375	Op 376	Op 377	Op 378	Op 379	Op 380	Op 381	Op 382	Op 383	Op 384	Op 385	Op 386	Op 387	Op 388	Op 389	Op 390	Op 391	Op 392	Op 393	Op 394	Op 395	Op 396	Op 397	Op 398	Op 399	Op 400	Op 401	Op 402	Op 403	Op 404	Op 405	Op 406	Op 407	Op 408	Op 409	Op 410	Op 411	Op 412	Op 413	Op 414	Op 415	Op 416	Op 417	Op
------	---------	---------	------	------	------	------	------	------	------	------	------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------	----

```
5839 057614 114000 000000 000000 X2INC: APNT,0,0 ; X2 INCREMENT.
5840 057622 173000 STOPN
5841 057624 114000 040000 000000 X2INCA: APNT,1!0,0
5842 057632 144100 RNLN,DBLPX
5843 057634 000777 X2INCB: 000777
5844 057636 173000 STOPN
5845 057640 160000 057624 DJMP,X2INCA
5846
5847 057644 114000 000000 000000 YUP: APNT,0,0 ; Y UP.
5848 057652 144000 000377 RNLN,377
5849 057656 173000 STOPN
5850
5851 057660 114000 000000 000002 YDN: APNT,0,2 ; Y DOWN.
5852 057666 144040 000377 RNLN,YDOWN,377
5853 057672 173000 STOPN
5854
5855 057674 114000 000000 000000 Y2UP: APNT,0,0 ; Y SKIP UP.
5856 057702 144400 000377 RNLN!RLSKIP,377
5857 057706 173000 STOPN
5858
5859 057710 114000 000000 000004 Y2DN: APNT,0,4 ; Y SKIP DOWN.
5860 057716 144440 000377 RNLN!RLSKIP!YDOWN,377
5861 057722 173000 STOPN
5862
5863 057724 END.TEST
```

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

5865

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

: (ITERATION COUNT = 3.)

5866
5867
5868
5869
5870
5871
5872
5873
5874
5875
5876
5877
5878
5879
5880
5881
5882
5883
5884
5885
5886
5887
5888
5889
5890
5891
5892
5893
5894
5895
5896
5897
5898
5899
5900

: 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 DYP) 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 LDCP (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.

5901
5902 060014 005737 002502
5903 060020 001402
5904 060022 000137 064660

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


5953

```
;WRITE 1'S INTO BOTH CURSOR REGISTERS AND VERIFY THAT ALL 1'S ARE RETURNED.
```

CSMT2: BGNSEG

```
;<<<<<<<<<<<BEGIN SEGMENT
```

```
MOV     #BIT15!^03776,R1 ; EXPECT CURSOR X = 3776 (WITH FLAG SET)
```

```
MOV      R1,R3          ; DITTO FOR CURSOR Y
```

```
MOV CSRETRY,R5 ; 000R
```

15: BISW3 CSCH.11, #WRTCPX!^03776, @DXR ; WRITE CURSOR X TO 3776.

MOV #WRTCPX!^03776,-(SP)

BIS CSCH.11.(SP)

```
MOV      (SP)+,ADXR
```

```
JSR PC, WAITF ; WAIT FOR READY.
```

BISW3 CSCH.11, WRTCPY!*03776, ADVR ; WRITE CURSOR Y TO 3776.

```
MOV     #WRTCPY!*03776,-(SP)
DEC     R12
```

BIS CSCH.11.(SP)
NOV (SP) 2000

```
MOV     (SP)+,ADYR
        DEC     DE,CF,IP,DP
```

```
JSR      PC,GETCURS
CMB      D1,D2
```

```
: READ CURSOR, GET DXR,DYR INTO R2,R4
```

CMP	R1,R2
DNF	28

; CHECK DXR (X-CURSOR)

BNE 25
GMD 03 04

; BR IF ERROR

CMP	R3,R4
REQ	36

```
; CHECK DYN (Y-CURSOR)
```

BEQ	SS
DEC	DE

; BR IF OK.

28: DEC RS
BPL 18

```

; DECREMENT THE RETRY COUNTER ;@@@R

```

BPL TS
UPDEB CRURO

```

; GO TRY AGAIN IF NOT NEGATIVE ;@@@R

```

ARDERR CRWRU,RLXYE
ENDSEG 122

; REPORT THE ERROR.

55: ENDSEG

```
;>>>>>>>>>>END SEGMENT
```



```
;CHECK DUAL ADDRESSING -- VERIFY THAT X NOT CHANGED BY WRITING Y.
```

```
CSM14:  MOV      #BIT15!^03776,R1 ; EXPECT CURSOR X TO BE ALL 1'S  
        MOV      #BIT15!0,R3       ; EXPECT CURSOR Y TO BE 0  
        BISW3    CSCH.11,#WRTCPX,R5 ; SET UP WRITE CURSOR W/ CHANNEL  
        MOV      #WRTCPX,R5  
        BIS      CSCH.11,R5
```

```
1$:     BGNSEG          ;<<<<<<<<<<<<<BEGIN SEGMENT  
        MOV      CSRETRY,R0         ; @@@R  
10$:    BISW3    R5,R1,@DXR         ; LOAD CURSOR X WITH A PATTERN.  
        MOV      R1,-(SP)  
        BIS      R5,(SP)  
        MOV      (SP)+,@DXR  
        JSR      PC,WAITF  
        BISW3    R5,R3,@DYR         ; LOAD OPPOSITE PATTERN INTO CURSOR Y.  
        MOV      R3,-(SP)  
        BIS      R5,(SP)  
        MOV      (SP)+,@DYR  
        JSR      PC,GETCURS          ; GET THE CURSOR INTO R2 & R4.  
        CMP      R1,R2              ; CHECK DXR (X-CURSOR)  
        BNE      2$                 ; BR IF ERROR  
        CMP      R3,R4              ; CHECK DYR (Y-CURSOR)  
        BEQ      3$                 ; BR IF OK.  
2$:     DEC      R0                  ; DECREMENT THE RETRY COUNTER ;@@@R  
        BPL      10$                ; GO TRY AGAIN IF NOT NEGATIVE ;@@@R  
        HRDEAR   CRDAY,RLXYE        ; REPORT THE ERROR.  
3$:     ENDSEG          ;>>>>>>>>>>>>>END SEGMENT
```

```
        BIT      #^03776,R3         ; DID WE WRITE 1'S INTO Y YET?  
        BNE      4$                 ; BR IF YES, SO WE'RE DONE.  
        MOV      R1,R3               ; NO, SO SWAP PATTERNS.  
        MOV      #BIT15,R1          ; ... SET UP TO WRITE X TO 0.  
        BR       1$                 ; LOOP BACK TO DO IT.
```

```
4$:     
```

: DO WRITE/READ TEST USING DISPLAY FILE.

```

CSMT5: CLR      R1          ; INIT THE X PATTERN TO 0
        MOV      #^03776,R3    ; INIT THE Y PATTERN TO ALL 1'S
        BISW3    CSCH.6,#LDCP,CSMF1A ; SET UP THE DISPLAY FILE
        MOV      #LDCP,-(SP)
        BIS      CSCH.6,(SP)
        MOV      (SP)+,CSMF1A
        BISW3    CSCH.8,#CURD,CSMF1B ; ... BY INSERTING THE CHANNEL.
        MOV      #CURD,-(SP)
        BIS      CSCH.8,(SP)
        MOV      (SP)+,CSMF1B

1$:     MOV      R1,CSMF1A+2    ; INSERT X COORD. TO BE WRITTEN
        MOV      R3,CSMF1A+4    ; ... AND THE Y COORDINATE.
        MOV      CSRETRY,R5
        BGNSEG                      :<<<<<<<<<<BEGIN SEGMENT
2$:     DPSTART  #CSMF1A        ; EXECUTE LOAD THEN READ OF CURSOR.
        MOV      #CSMF1A,@DFC   ; START THE DPU.
        JSR      PC,WAITF       ; WAIT FOR DISPLAY STOP.
        BIS      #BIT15,R1      ; EXPECT THE CURSOR FLAG BIT IN DXR
        BIS      #BIT15,R3      ; ... AND DYR.
        JSR      PC,RDDXY       ; GET MASKED DXR AND DYR INTO R2 AND R4.
        CMP      R1,R2          ; CHECK DXR FOR X CURSOR.
        BNE      3$            ; BR IF INCORRECT.
        CMP      R3,R4          ; CHECK DYR FOR Y CURSOR.
        BEQ      4$            ; BR IF OK.
3$:     DEC      R5
        BPL      2$
        HRDERR   CRwRE,RLXYE    ; ERROR -- DXR OR DYR WRITE/READ ERROR.
4$:     ENDSEG                    ;>>>>>>>>>>END SEGMENT

        ADD      #2,BIT15,R1    ; INCREMENT THE X PATTERN.
        SUB      #2,BIT15,R3    ; DECREMENT THE Y PATTERN.
        BIT      #^03776,R1     ; DID WE INCREMENT X AROUND TO 0?
        BNE      1$            ; LOOP BACK IF NO.
30$:

```

```

; *** SPACE HERE FOR TESTING GENERAL DUAL ADDRESSING (ALL DBUS CODES & CHANNELS)
; VERIFY THAT THE SWITCH, MATCH AND CROSSHAIR ENABLES CAN BE CLEARED TO 0'S.

```

```
CSMT6: BGNSEG ;<<<<<<<<<<<<BEGIN SEGMENT
JSR PC,DSBINT ; DISABLE INTERRUPTS
MOV #BIT15,R1 ; EXPECT DXR TO BE 0
MOV R1,R3 ; DITTO FOR DYR.
BISW3 CSCH.8,#WJSS!^077,@DXR ; FIRST SET ALL ENABLES.
MOV #WJSS!^077,-(SP)
BIS CSCH.8,(SP)
MOV (SP)+,@DXR
JSR PC,WAITF ; WAIT FOR READY.
BISW3 CSCH.8,#WJSS.BIT5!BIT3!BIT1,@DXR ; THEN CLEAR THEM.
MOV #WJSS!BIT5!BIT3.BIT1,-(SP)
BIS CSCH.8,(SP)
MOV (SP)+,@DXR
MOV CSRETRY,R5 ; @@@@
1$: JSR PC,GETJSE ; GET THE ENABLES INTO R2 AND R4.
CMP R1,R2 ; CHECK DXR (X-CURSOR)
BNE 2$ ; BR IF ERROR
CMP R3,R4 ; CHECK DYR (Y-CURSOR)
BEQ 3$ ; BR IF OK.
2$: DEC R5 ; DECREMENT THE RETRY COUNTER ;@@@
BPL 1$ ; GO TRY AGAIN IF NOT NEGATIVE ;@@@
HRDERR JSEWRZ,RLXYE ; REPORT THE ERROR.
3$: ENDSEG ;>>>>>>>>>>>>END SEGMENT
```

: VERIFY THAT THE SWITCH, MATCH AND CROSSHAIR ENABLES CAN BE SET TO 1'S.

```
CSMT7: BGNSEG ;<<<<<<<<<BEGIN SEGMENT
MOV #BIT15!JSSIES!JSMIES,R1 ; EXPECT DXR TO HAVE SW & MAT SET.
MOV #BIT15!CHIE,R3 ; EXPECT DYR TO HAVE CROSSHAIR ENABLE SET.
BISW3 CSCH.8,#WJSS!BIT5!BIT3!BIT1,@DXR ; FIRST CLEAR THEM.
MOV #WJSS!BIT5!BIT3!BIT1,-(SP)
BIS CSCH.8,(SP)
MOV (SP)+,@DXR
JSR PC,WAITF ; WAIT FOR READY.
BISW3 CSCH.8,#WJSS!^077,@DXR ; THEN SET THEM.
MOV #WJSS!^077,-(SP)
BIS CSCH.8,(SP)
MOV (SP)+,@DXR
MOV CSRETRY,R5 ; @@@@
1$: JSR PC,GETJSE ; GET THE ENABLES INTO R2 AND R4.
CMP R1,R2 ; CHECK DXR (X-CURSOR)
BNE 2$ ; BR IF ERROR
CMP R3,R4 ; CHECK DYR (Y-CURSOR)
BEQ 3$ ; BR IF OK.
2$: DEC R5 ; DECREMENT THE RETRY COUNTER ; @@@@
BPL 1$ ; GO TRY AGAIN IF NOT NEGATIVE ; @@@@
HRDERR JS' WRO,RLXYE ; REPORT THE ERROR.
3$: ENDSEG ;>>>>>>>>>>END SEGMENT
```


[illegible]

[illegible]

6224
6225
6226
6227
6228
6229
6230
6231
6232
6233
6234
6235
6236
6237
6238
6239

```

:TEST THE SOFT JOYSTICK SWITCH INTERRUPT OPERATION:

```

- (1) CLEAR SWITCH ENABLE, SET SWITCH, SEE THAT NO INTERRUPT PENDING.
- (2) CLEAR SWITCH, SET SWITCH ENABLE, SEE THAT NO INTERRUPT PENDING.
- (3) SET BOTH SWITCH ENABLE AND SWITCH, SEE THAT FLAGS SHOWS PENDING INTERRUPT.
- (4) REPEAT (3), BUT ALLOW INTERRUPT AND CHECK RESULTS
- (5) REPEAT (4) BUT FROM DISPLAY FILE, SEE THAT SWITCH INTERRUPT STOPS PROCESSING AND GETS CURSOR.

;(1) TEST THAT NO INTERRUPT REQUEST OCCURS WITH ENABLE CLEAR AND SWITCH SET.

6240	062106		
6241	062110	004737	026176
6242	062114		
	062114	004737	025402
6243	062120		
	062120	012746	042002
	062124	053716	065006
	062130	012677	120702
6244	062134	004737	027454
6245	062140		
	062140	012746	160002
	062144	053716	065006
	062150	012677	120664
6246	062154	004737	027454
6247	062160	004737	023422
6248	062164		
	062164	012746	160000
	062170	053716	065006
	062174	012677	120640
6249	062200	004737	027454
6250	062204	012777	040000
6251	062212	004737	027454
6252	062216		
	062216	012777	000012
6253	062224		

```
CSMT10: BGNSEG ;<<<<<<<<<<<BEGIN SEGMENT
JSR PC,DSBINT ; DISABLE INTERRUPTS.
INITDP ; INITIALIZE THE DPU.
JSR PC,DPINIT ; GO DO SOFT INIT, CHECK THE STATE.
BISW3 CSCH.8,#WJSS!JSWE.0,@DXR ; CLEAR SWITCH ENABLE.
MOV #WJSS!JSWE.0,-(SP)
BIS CSCH.8,(SP)
MOV (SP)+,@DXR
JSR PC,WAITF
BISW3 CSCH.8,#WECC!SWCHON,@DYR ; SET THE SWITCH.
MOV #WECC!SWCHON,-(SP)
BIS CSCH.8,(SP)
MOV (SP)+,@DYR
JSR PC,WAITF
JSR PC,IIRCHK ; CHECK INTERNAL REGS FOR INIT STATE.
1$: BISW3 CSCH.8,#WECC,@DYR ;TURN OFF THE SWITCH
MOV #WECC,-(SP)
BIS CSCH.8,(SP)
MOV (SP)+,@DYR
JSR PC,WAITF
MOV #GTJSSW,@DXR ;DISMISS INTR REQ, IF ANY
JSR PC,WAITF
WTDSRA #0,FLG ;CLEAR PENDING INTERRUPT
MOV #0&^C17.SETFLG,@DSR
ENDSEG ;>>>>>>>>>>>END SEGMENT
```

```

6255                                     ;(2) TEST THAT NO INTERRUPT REQUEST OCCURS WITH ENABLE SET AND SWITCH CLEAR.
6256
6257
6258 062226 CSMT11: BGNSEG                                ;<<<<<<<<<<BEGIN SEGMENT
6259 062230      JSR      PC,DSBINI                      ; DISABLE INTERRUPTS.
6260 062234      INITDP                               ; INITIALIZE THE DPU.
        062234      004737      025402      JSR      PC,DPINIT          ; GO DO SOFT INIT, CHECK THE STATE.
6261 062240      BISW3    CSCH.8,#WECC!0,@DYR           ; CLEAR THE SWITCH.
        062240      012746      160000      MOV      #WECC.0,-(SP)
        062244      053716      065006      BIS      CSCH.8,(SP)
        062250      012677      120564      MOV      (SP)+,@DYR
6262 062254      004737      027454      JSR      PC,WAITF
6263 062260      BISW3    CSCH.8,#WJSS!JSWE!JSWD,@DXR     ; SET SWITCH ENABLE.
        062260      012746      042003      MOV      #WJSS!JSWE!JSWD,-(SP)
        062264      053716      065006      BIS      CSCH.8,(SP)
        062270      012677      120542      MOV      (SP)+,@DXR
6264 062274      004737      027454      JSR      PC,WAITF
6265 062300      004737      023422      JSR      PC,IIRCHK                ; CHECK INTERNAL REGS FOR INIT STATE.
6266 062304      1$:      INITDP
        062304      004737      025402      JSR      PC,DPINIT          ; GO DO SOFT INIT, CHECK THE STATE.
6267 062310      ENDSEG                                ;>>>>>>>>>>END SEGMENT

```

```

6269                                     ;(3) SET SWITCH AND ENABLE, SEE PENDING INTERRUPT
6270
6271
6272 062312 CSMT12: BGNSEG                               ;<<<<<<<<<<<<BEGIN SEGMENT
6273 062314 INITDP
        062314 004737 025402 JSR PC,DPINIT          ; GO DO SOFT INIT, CHECK THE STATE.
6274 062320 004737 026176 JSR PC,DSBINT
6275 062324 BISW3 CSCH.8,#WJSS!JSWE!JSWD,@DXR      ; ENABLE SWITCH INTERRUPT.
        062324 012746 042003 MOV #WJSS!JSWE!JSWD,-(SP)
        062330 053716 065006 BIS CSCH.8,(SP)
        062334 012677 120476 MOV (SP)+,@DXR
6276 062340 004737 027454 JSR PC,WAITF
6277 062344 BISW3 CSCH.8,#WECC!SWCHON,@DYR          ; TURN THE SWITCH ON.
        062344 012746 160002 MOV #WECC!SWCHON,-(SP)
        062350 053716 065006 BIS CSCH.8,(SP)
        062354 012677 120460 MOV (SP)+,@DYR
6278 062360 004737 027454 JSR PC,WAITF
6279 062364 012701 100016 MOV #CHAR!JSLCKO!PINTRB!PVEC2,R1    ; EXPECT FLAGS TO SHOW PENDING.
6280 062370 RDDSRA FLG,R2                                ; READ THE FLAGS REGISTER INTO R2.
        062370 012777 000002 120436 MOV #SELF LG,@DSR      ; READ FLG INTO R2 .
        062376 017702 120432 MOV @DSR,R2
6281 062402 IFERROR NOSWPI,EXPREC
        062402 020102 CMP R1,R2                        ; OK?
        062404 001410 BEQ 64$                          ; YES.
        062406 HRDERR NOSWPI,EXPREC,                    ; NO.
        062426 64$:
        062426 1$:
6282 062426 BISW3 CSCH.8,#WECC,@DYR                ; TURN OFF THE SWITCH.
6283 062426 MOV #WECC,-(SP)
        062432 053716 065006 BIS CSCH.8,(SP)
        062436 012677 120376 MOV (SP)+,@DYR
6284 062442 004737 027454 JSR PC,WAITF
6285 062446 BISW3 CSCH.8,#WJSS!JSWE!0,@DXR
        062446 012746 042002 MOV #WJSS!JSWE!0,-(SP)
        062452 053716 065006 BIS CSCH.8,(SP)
        062456 012677 120354 MOV (SP)+,@DXR
6286 062462 004737 027454 JSR PC,WAITF
6287 062466 012777 040000 120342 MOV #GTJSSW,@DXR      ;DISMISS INTR REQ, IF ANY
6288 062474 004737 027454 JSR PC,WAITF
6289
6290 062500 WTDSRA #0,FLG                              ; CLEAR THE FLAGS AND INTERRUPT.
        062500 012777 000012 120326 MOV #0&^C17!SETFLG,@DSR
6291 062506 004737 023422 JSR PC,IIRCHK          ; CHECK THE INTERNAL REGISTERS
6292 062512 2$: ENDSEG                                ;>>>>>>>>>>>>END SEGMENT
6293 062514 004737 026074 JSR PC,DPRESET          ; DO SOFT INITIALIZE.
```

```

6295                                     ;(4) SET SWITCH AND ENABLE, CHECK FOR PROPER INTERRUPT.
6296
6297
6298 CSMT13: BGNSEG                               ;<<<<<<<<<<BEGIN SEGMENT
6299      INITDP
        JSR     PC,DPINIT           ; GO DO SOFT INIT, CHECK THE STATE.
6300      JSR     PC,ENAINIT
6301      INTSET    JSS
        MOV     #^0177400!IOKCKIN!IOKJSS,INTMASK          ; PRIME FLAG, EXPECT JSS
6302      BISW3     CSCH.8,#WJSS!JSWE!JSWD,@DXR             ; ENABLE SWITCH INTERRUPT.
        MOV     #WJSS!JSWE!JSWD,-(SP)
        BIS     CSCH.8,(SP)
        MOV     (SP)+,@DXR
6303      JSR     PC,WAITF
6304      BISW3     CSCH.8,#WECC!SWCHON,@DYR                 ; TURN THE SWITCH ON.
        MOV     #WECC!SWCHON,-(SP)
        BIS     CSCH.8,(SP)
        MOV     (SP)+,@DYR
6305      JSR     PC,WAITF
6306      MOV     #100,R2
6307      TSTB     INTFLAG                     ; DID WE INTERRUPT?
6308      BPL     11$                          ; BR IF YES.
6309      DEC     R2
6310      BGT     10$
6311      TSTB     INTFLAG                     ; DID WE GET GOOD INTERRUPT?
6312      BEQ     2$                          ; BR IF YES
6313      MOV     #10$,R1                      ; GET CPU PC.
6314      HRDERR   NOSWI,INTERR
6315      JMP     3$                          ; MIGHT AS WELL QUIT.
6316
6317      2$:
6318
6319      3$: ENDSEG                               ;>>>>>>>>>END SEGMENT
6320      JSR     PC,DPRESET           ; DO SOFT INIT.

```

6322				: (5) CHECK SWITCH INTERRUPT OPERATION FROM DISPLAY FILE.
6323				
6324				
6325	062662			CSMT14: BGNSEG ; <<<<<<<<<< BEGIN SEGMENT
6326	062664			INITDP
	062664	004737	025402	JSR PC, DPINIT ; GO DO SOFT INIT, CHECK THE STATE.
6327	062670	004737	026114	JSR PC, ENAINT ; ENABLE INTERRUPTS.
6328	062674			BISW3 CSCH.6, #LDCP, CSMF3A ; SET UP DISPLAY FILE TO LOAD CURSOR,
	062674	012746	175000	MOV #LDCP, -(SP)
	062700	053716	065004	BIS CSCH.6, (SP)
	062704	012637	065072	MOV (SP)+, CSMF3A
6329	062710			BISW3 CSCH.8, #CUIS, CSMF3B ; ... AND SET SWITCH ENABLE
	062710	012746	146013	MOV #CUIS, -(SP)
	062714	053716	065006	BIS CSCH.8, (SP)
	062720	012637	065100	MOV (SP)+, CSMF3B
6330	062724			BISW3 CSCH.6, #LDECC!SWCHON, CSMF3C ; ... AND TURN ON SWITCH.
	062724	012746	175402	MOV #LDECC!SWCHON, -(SP)
	062730	053716	065004	BIS CSCH.6, (SP)
	062734	012637	065102	MOV (SP)+, CSMF3C
6331	062740	112737	000377	MOVB #^0377, INTFLAG ; PRIME THE INTERRUPT FLAG.
6332	062746			LPSTART #CSMF3A ; START THE DPU.
	062746	012777	065072	MOV #CSMF3A, @DPC ; START THE DPU.
	062754	004737	027454	JSR PC, WAITF ; WAIT FOR DISPLAY STOP.
6333	062760	004737	027454	JSR PC, WAITF ; WAIT FOR STOP.
6334	062764	017702	120042	MOV @DPC, R2 ; SAVE THE DPC.
6335	062770	105737	026107	TSTB INTFLAG ; TEST FOR GOOD INTERRUPT.
6336	062774	001412		BEQ 1\$; BR IF GOOD.
6337	062776	012701	062760	MOV #-16, R1
6338	063002			HRDERR NOSWI, INTERR
6339				
6340	063022	020227	065114	1\$: CMP R2, #CSMF3D ; DID THE DPC HIT THE STOPN?
6341	063026	103412		BLO 2\$; BR IF NO -- OK
6342	063030	012701	065106	MOV #CSMF3C+4, R1 ; EXPECT A DPC PRIOR TO STOPN.
6343	063034			HRDERR SWNOHLT, EXPREC
6344				
6345	063054			2\$: BISW3 #BIT15, CSMF3A+2, R1 ; EXPECT DXR TO HAVE X CURSOR.
	063054	013701	065074	MOV CSMF3A+2, R1
	063060	052701	100000	BIS #BIT15, R1
6346	063064			BISW3 #BIT15, CSMF3A+4, R3 ; EXPECT DYR TO HAVE Y CURSOR.
	063064	013703	065076	MOV CSMF3A+4, R3
	063070	052703	100000	BIS #BIT15, R3
6347	063074	005737	065012	TST CSRETRY
6348	063100	001416		BEQ 3\$
6349	063102	004737	065362	JSR PC, RDDXY ; GET MASKED DXR, DYR INTO R2,R4.
6350	063106			IFERRX2 SWNOCU, RLXYE
	063106	020102		CMP R1, R2 ; 1ST ITEM OK?
	063110	001002		BNE 65\$; NO.
	063112	020304		CMP R3, R4 ; 2ND ITEM OK?
	063114	001410		BEQ 64\$; YES.
	063116			65\$: HRDERR SWNOCU, RLXYE, ; NO.
	063136			64\$:
6351				
6352	063136	004737	026176	3\$: JSR PC, DSBINT
6353	063142			ENDSEG ; >>>>>>>>>> END SEGMENT
6354	063144	004737	026074	JSR PC, DPRESSET ; DO SOFT INIT.

[illegible]

6397	063470	013705	065012		MOV	CSRETRY,R5	
6398	063474	013701	065170		MOV	CSMF4X,R1	
6399	063500	013703	065172		MOV	CSMF4Y,R3	
6400	063504	052701	100000		BIS	#BIT15,R1	
6401	063510	052703	100000		BIS	#BIT15,R3	
6402	063514				BISW3	CSCH.8,#WECC,@DYR	; CLEAR THE SWITCH
	063514	012746	160000		MOV	#WECC,-(SP)	
	063520	053716	065006		BIS	CSCH.8,(SP)	
	063524	012677	117310		MOV	(SP)+,@DYR	
6403	063530	004737	027454		JSR	PC,WAITF	
6404	063534	112737	000377	026107 43\$:	MOVB	#^0377,INTFLAG	; PRIME THE INTERRUPT FLAG
6405	063542				DPSTART	#CSMF4D	
	063542	012777	065176	117262	MOV	#CSMF4D,@DPC	; START THE DPU.
	063550	004737	027454		JSR	PC,WAITF	; WAIT FOR DISPLAY STOP.
6406	063554	017702	117252		MOV	@DPC,R2	
6407	063560	105737	026107		TSTB	INTFLAG	; ANY INTERRUPTS?
6408	063564	002414			BLT	403\$	
6409	063566	010146			MOV	R1,-(SP)	
6410	063570	012701	063552		MOV	#-14,R1	
6411	063574				HRDERR	IFAUULT,INTERR	
6412	063614	012601			MOV	(SP)+,R1	
6413	063616			403\$:			
6414	063616	004737	065362		JSR	PC,RDDXY	
6415	063622	020102			CMP	R1,R2	
6416	063624	001002			BNE	44\$	
6417	063626	020304			CMP	R3,R4	
6418	063630	001414			BEQ	45\$	
6419	063632	005305		44\$:	DEC	R5	
6420	063634	100337			BPL	43\$	
6421	063636				HRDERR	CRWRE,RLXYE	
6422	063656	000137	064552		JMP	8\$	; GO TRY THE NEXT COORDINATES.
6423	063662			45\$:			
6424	063662	112737	000202	026106	MOVB	#I0KCKIN:I0KJSM,INTMASK	; SAY WE'LL CHECK FOR MATCH
6425	063670	112737	000377	026107	MOVB	#^0377,INTFLAG	; PRIME THE INTERRUPT FLAG.
6426	063676				DPCONT		
	063676	052777	000001	117126	BIS	#BIT0,@DPC	; CONTINUE THE DPU.
	063704	004737	027454		JSR	PC,WAITF	; WAIT FOR DISPLAY STOP.
6427	063710	012701	000100		MOV	#100,R1	; SETUP A WATCHDOG COUNTER.
6428	063714	105737	026107	40\$:	TSTB	INTFLAG	; DID WE GET AN INTERRUPT?
6429	063720	100002			BPL	41\$	; BR IF YES.
6430	063722	005301			DEC	R1	
6431	063724	003373			BGT	40\$	
6432	063726	017702	117100	41\$:	MOV	@DPC,R2	; GET DPC FOR ERROR.
6433	063732	017737	117100	003170	MOV	@DXR,SDXR	
6434	063740	017737	117074	003172	MOV	@DYR,SDYR	
6435	063746	105737	026107		TSTB	INTFLAG	
6436	063752	001417			BEQ	5\$	; WAS IT A GOOD INTERRUPT?
6437	063754	002434			BLT	50\$	; BR IF NO INTERRUPT OCCURRED.
6438	063756	010146			MOV	R1,-(SP)	
6439	063760	012701	063714		MOV	#40\$,R1	
6440	063764				HRDERR	NOCMI,INTERR	
6441	064004	012601			MOV	(SP)+,R1	
6442	064006	000137	064046		JMP	50\$	
6443							
6444	064012			5\$:			
6445	064012	027727	117014	065234	CMP	@DPC,#CSMF4K+2	; DID WE GET TO THE STOPN (NO MATCH)?
6446	064020	001040			BNE	51\$	; BR IF NO

```

6447 064022      HRDERR  NOMST,PNTCOOR
6448 064042 000137 064362      JMP      7$      ; GO SEE IF OTHER THINGS CHECK OUT.
6449
6450 064046      50$:      ; COME HERE IF NO MATCH INTERRUPT OCCURRED.
6451 064046 020227 065232      CMP      R2,#CSMF4K      ; DID WE DO A MATCH STOP?
6452 064052 001011      BNE      501$      ; BR IF NO
6453 064054      HRDERR  CMNOMI,PNTCOOR ; YES
6454 064074 000412      BR      51$      ; GO CHECK OTHER MATCH DATA.
6455 064076      501$:      HRDERR  NOMAT,PNTCOOR ; NO MATCH AT ALL.
6456 064116 000137 064362      JMP      7$      ; GO SEE IF IT MATCHES ELSEWHERE.
6457
6458
6459 064122      51$:      ; COME HERE IF WE THINK WE GOT A MATCH
6460
6461 064122 012701 065232      MOV      #CSMF4K,R1      ; EXPECT DPC TO BE JUST AFTER MATCH.
6462 064126 012703 113774      MOV      #LVEC!ALL,R3      ; EXPECT DSR TO SHOW LVEC & ALL PIX BITS
6463 064132 017702 116674      MOV      @DPC,R2      ; GET ACTUAL DPC.
6464 064136 017704 116672      MOV      @DSR,R4      ; AND THE DSR.
6465 064142      IFERRX2 MATPCS,EXPRC2
6466 064142 020102      CMP      R1,R2      ; 1ST ITEM OK?
6467 064144 001002      BNE      65$      ; NO.
6468 064146 020304      CMP      R3,R4      ; 2ND ITEM OK?
6469 064150 001410      BEQ      64$      ; YES.
6470 064152      65$:      HRDERR  MATPCS,EXPRC2, ; NO.
6471 064172      64$:
6472
6473 064172 013701 065170      MOV      CSMF4X,R1      ; GET EXPECTED MATCH X
6474 064176 013703 065172      MOV      CSMF4Y,R3      ; ... AND THE Y.
6475 064202 017702 116630      MOV      @DXR,R2      ; GET DXR, SHOULD HOLD MATCH X
6476 064206 017704 116626      MOV      @DYR,R4      ; GET DYR, SHOULD HOLD MATCH Y.
6477 064212      IFERRX2 MATXYE,EXPRC2
6478 064212 020102      CMP      R1,R2      ; 1ST ITEM OK?
6479 064214 001002      BNE      67$      ; NO.
6480 064216 020304      CMP      R3,R4      ; 2ND ITEM OK?
6481 064220 001410      BEQ      66$      ; YES.
6482 064222      67$:      HRDERR  MATXYE,EXPRC2, ; NO.
6483 064242      66$:
6484
6485 064242 012777 100000 116566      MOV      #RSTPOS,@DXP      ; RESTORE GRAPHIC POSITION TO DXR, DYR
6486 064250 004737 027454      JSR      PC,WAITF      ; WAIT FOR STOP.
6487 064254 062701 000004      ADD      #4,R1      ; X SHOULD BE X(MATCH)+4
6488 064260 017702 116552      MOV      @DXR,R2      ; GET ACTUAL DXR
6489 064264 017704 116550      MOV      @DYR,R4      ; ... AND DYR.
6490 064270      IFERRX2 MATPOS,EXPRC2
6491 064270 020102      CMP      R1,R2      ; 1ST ITEM OK?
6492 064272 001002      BNE      69$      ; NO.
6493 064274 020304      CMP      R3,R4      ; 2ND ITEM OK?
6494 064276 001410      BEQ      68$      ; YES.
6495 064300      69$:      HRDERR  MATPOS,EXPRC2, ; NO.
6496 064320      68$:
6497
6498 064320 012701 110000      6$:      MOV      #LVEC,R1      ; GET OP-CODE EXPECTED FOR FLAGS.
6499 064324      RDDSRA  FLG,R2      ; GET FLAGS.
6500 064324 012777 000002 116502      MOV      #SELFGL,@DSR ; READ FLG INTO R2.
6501 064332 017702 116476      MOV      @DSR,R2
6502 064336      IFERROR MATFLE,FLGER,7$
6503 064336 020102      CMP      R1,R2      ; OK?

```



```

    064650 000137 060046      64$: JMP CSMITR
    064654 004737 030072      65$: JSR PC,TSTEND ; PRINT ERROR SUMMARY, IF REQ'D.
6525 064660 CSMXIT: EXIT TST
6526
6527
6528
6529 ; TABLE OF MATCH POINTS -- ALL VALUES ARE USED FOR X AND Y:
6530
6531 064664 000000 CSM PX: .WORD 0
6532 064666 000000 CSM PY: .WORD 0
6533
6534 064670 000000 000002 000004 CSM PTB: .WORD 0,BIT1,BIT2,BIT3,BIT4 ; FIRST, FLOATING 1 ON 0'S, BITS 2
6535 064702 000040 000100 000200 .WORD BIT5,BIT6,BIT7,BIT8,BIT9 ; ... THRU 9.
6536 001776 A1=^01776 ; ALL 1'S FOR POSITION
6537 064714 001776 001774 001772 .WORD A1,A1&^CBIT1,A1&^CBIT2,A1&^CBIT3,A1&^CBIT4 ; THEN, FLOATING 0 ON 1'S
6538 064726 001736 001676 001576 .WORD A1&^CBIT5,A1&^CBIT6,A1&^CBIT7,A1&^CBIT8,A1&^CBIT9
6539 064740 001252 000524 001462 .WORD 1252,524,1462,636 ; ALTERNATING BITS AND PAIRS.
6540 064750 000006 000016 000036 .WORD 6,16,36,76,176,376
6541 064764 001770 001760 001740 .WORD 1770,1760,1740,1700,1600,1400
6542 065000 000000 CSM PTE: .WORD 0
6543
6544 065002 000000 CSCHAN: .WORD 0
6545 065004 000000 CSCH.6: .WORD 0
6546 065006 000000 CSCH.8: .WORD 0
6547 065010 000000 CSCH.11: .WORD 0
6548
6549 065012 000002 CSRETRY: .WORD 2
6550
6551 ; EXTENDED PRINTOUT FOR SYNC CHANNEL:
6552 065014 PRSYCH: PRINTX #FSYCHAN,CSCHAN
6553 065040 000207 RTS PC
6554
6555 ; DISPLAY FILES FOR CURSOR/SWITCH/MATCH TEST --
6556
6557 065042 175000 000000 000000 CSMF1A: .WORD LDCP,0,0 ; LOAD CURSOR POSITION REGISTERS
6558 065050 146100 CSMF1B: .WORD CURD ; READ THEM BACK
6559 065052 173000 .WORD STOPN
6560
6561 065054 146052 CSMF2A: .WORD CUIOFF!CUOFF ; CLEAR ALL ENABLES
6562 065056 146012 CSMF2B: .WORD CUIOFF ; DO GENERAL WRITE
6563 065060 146100 173000 CSMF2C: .WORD CURD,STOPN ; READ CURSOR (ENABLES COME TOO), THEN STOP
6564 065064 146077 CSMF2D: .WORD CUON!^077 ; SET ALL ENABLES
6565 065066 160000 065056 .WORD DJMP,CSMF2B ; THEN GO DO GENERAL WRITE AND READ.
6566
6567 065072 175000 001332 002464 CSMF3A: .WORD LDCP,^01332,^02464 ; LOAD CURSOR WITH SOME DATA
6568 065100 146013 CSMF3B: .WORD CUIS ; ENABLE SWITCH INTERRUPT
6569 065102 175402 CSMF3C: .WORD LDECC!SWCHON ; TURN ON THE SOFT SWITCH
6570 065104 164000 164000 164000 .WORD DNOP,DNOP,DNOP,DNOP ; DO SOME NO-OPS -- SHOULD STOP
6571 065114 173000 CSMF3D: .WORD STOPN ; STOP -- SWITCH DIDN'T WORK
6572
6573 ; DISPLAY FILE FOR CURSOR MATCH TEST:
6574
6575 065116 176074 176474 177074 CSMF4V: .WORD RDWRT!CHO,RDWRT.CH1,RDWRT!CH2,RDWRT.CH3 ; MEMORY SETUP, R/W.
6576 065126 160000 065142 .WORD DJMP,CSMF4A
6577
6578 065132 176034 176434 177034 CSMF4U: .WORD WRT!CHO,WRT.CH1,WRT.CH2,WRT!CH3 ; MEMORY SETUP, WRT-ONLY.
  
```

```

6579 065142 170140          CSMF4A: .WORD CLRMEM          ; CLEAR THE MEMORIES.
6580 065144 160001 065162          .WORD DJMS,CSMF4B      ; GO TURN ON CURSOR AND MATCH ENABLE
6581 065150 160001 065156 173000   .WORD DJMS,CSMF4C,STOPN ; TURN SWITCH ON, AND STOP
6582 065156 175402 165000   CSMF4C: .WORD LDECC!SWCHON,DPOP ; TURN SWITCH ON, AND RETURN.
6583
6584 065162 146076 165000   CSMF4B: .WORD CUON.CUIM,DPOP      ; ENABLE CURSOR AND MATCH.
6585 065166 175000          CSMF4LC: .WORD LDCP          ; LOAD CURSOR POSITION.
6586 065170 000000          CSMF4X: .WORD 0              ; X
6587 065172 000000          CSMF4Y: .WORD 0              ; Y
6588 065174 165000          .WORD DPOP                  ; RETURN.
6589
6590 065176 114000 160001 065170   CSMF4D: .WORD APNT,DJMS,CSMF4X      ; SET MATCH POSITION
6591 065204 160001 065166          .WORD DJMS,CSMF4LC      ; LOAD CURSOR POSITION.
6592 065210 146152 173000   CSMF4J: .WORD CURD.JSWE.JMWE.JCWE,STOPN ; STOP FOR CHECK.
6593 065214 160001 065162          .WORD DJMS,CSMF4B      ; GO TURN ON MATCH ENABLE
6594 065220 160001 065156          .WORD DJMS,CSMF4C      ; GO TURN ON SWITCH
6595 065224 113774          .WORD LVEC!ALL              ; SET TO WHITE.
6596 065226 040004 000000          .WORD I.4,0          ; DRAW, SHOULD MATCH
6597 065232 173000          CSMF4K: .WORD STOPN          ; ... WITH DPC AT F4K
6598 065234 114000 160001 065170   CSMF4E: .WORD APNT,DJMS,CSMF4X      ; RESTART. SET POINT
6599 065242 113600          .WORD LVEC!YELLOW          ; CHANGE COLOR
6600 065244 000002 000000          .WORD 2,0            ; ... THEN MOVE TO X+2, Y.
6601 065250 041776 000000 061776   .WORD I.MAXX,0,I.M.MAXX,0 ; DRAW OUT TO RIGHT AND BACK.
6602 065260 113300          .WORD LVEC!VIOLET
6603 065262 020002 000002          CSMF4F: .WORD M!2,2      ; GO TO ABOVE MATCH POINT
6604 065266 040000 001776 040000   .WORD I.0,MAYY,I.0,M!MAXY ; ... AND DRAW UP AND BACK.
6605 065276 113500          .WORD LVEC.EGGBL
6606 065300 020002 020002          CSMF4G: .WORD M!2,M.2
6607 065304 061776 000000 041776   .WORD I!M!MAXX,0,I.MAXX,0 ; DRAW LEFT AND BACK.
6608 065314 113200          .WORD LVEC!GOLD
6609 065316 000002 020002          CSMF4H: .WORD 2,M!2
6610 065322 040000 021776 040000   .WORD I!0,M!MAXY,I.0,MAYY
6611 065332 173000 173000   CSMF4I: .WORD STOPN,STOPN
6612
6613 ; SUBROUTINES:
6614 ; GET CURSOR COORDINATES, AND READ MASKED DXR, DYR INTO R2, R4 --
6615 065336 004737 027454   GETCURS: JSR PC,WAITF          ; WAIT FOR STOP BIT.
6616 065342          BICW3 CSCH.8,#WRTJSS!BIT6,@DXR      ; RETRIEVE CURSOR FROM SELECTED CHAN.
6617 065342 012746 042100          MOV #WRTJSS!BIT6,-(SP)
6618 065346 053716 065006          BIS CSCH.8,(SP)
6619 065352 012677 115460          MOV (SP)+,@DXR
6620 065356 004737 027454   JSR PC,WAITF          ; WAIT FOR STOP.
6621 065362          BICW3 #^C<BIT15!^03777>,@DXR,R2 ; DXR TO R2, LEAVE FLAG & COORD.
6622 065362 017702 115450          MOV @DXR,R2
6623 065366 042702 074000          BIC #^C<BIT15!^03777>,R2
6624 065372          BICW3 #^C<BIT15!^03777>,@DYR,R4 ; DYR TO R4, LEAVE FLAG & COORD.
6625 065372 017704 115442          MOV @DYR,R4
6626 065376 042704 074000          BIC #^C<BIT15!^03777>,R4
6627 065402 000207          RTS PC
6628
6629 ; GET JOYSTICK ENABLES-
6630 065404 004737 027454   GETJSE: JSR PC,WAITF          ; WAIT FOR STOP.
6631 065410          BICW3 CSCH.8,#WRTJSS!BIT6,@DXR      ; READ CURSOR
6632 065410 012746 042100          MOV #WRTJSS!BIT6,-(SP)
6633 065414 053716 065006          BIS CSCH.8,(SP)
6634 065420 012677 115412          MOV (SP)+,@DXR
6635 065424 004737 027454   JSR PC,WAITF

```

```
6626 065430 RDJSE: BICW3 #^C<BIT15!JSSIES!JSMIES>,@DXR,R2 ;DXR TO R2, LEAVE ENABLES
      065430 017702 115402 MOV @DXR,R2
      065434 042702 063777 BIC #^C<BIT15!JSSIES!JSMIES>,R2
6627 065440 BICW3 #^C<BIT15!CHIE>,@DYR,R4 ;DYR TO R4.
      065440 017704 115374 MOV @DYR,R4
      065444 042704 067777 BIC #^C<BIT15!CHIE>,R4
6628 065450 000207 RTS PC
6629
6630 065452 END.TEST
```

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

6632

[illegible]

	065772	103722		BCS	1\$	
6680						
6681						
6682	065774	013737	003234	027700	MOV	TEMP2,LOOPK ;RESET LOOPER.
6683	066002	023727	066202	174102	CMP	YI,#GYI.2
6684	066010	001414			BEG	6\$;BR IF LAST INCR DONE.
6685	066012	162737	000002	066202	SUB	#2,YI ;OTHERWISE, SET NEXT INCR...
6686	066020	162737	000030	066214	SUB	#30,XA ;...AMPLITUDE...
6687	066026	162704	000030		SUB	#30,R4
6688	066032	162737	000030	066200	SUB	#30,GHXT+2 ;...AND BASE LINE.
6689	066040	000677			BR	1\$;DO NEXT SET.
6690						
6691	066042	005737	066174		6\$: TST	GHXTYP ;JUST DID GRAPH?
6692	066046	001004			BNE	30\$;...BR AHEAD IF NOT.
6693	066050	012737	040000	066174	MOV	#HST,GHXTYP ;YES. SET HISTO MODE
6694	066056	000652			BR	8\$;...AND GO DO IT.
6695	066060	023727	066174	060000	30\$: CMP	GHXTYP,#FHST ;JUST DID FILLED-HISTO?
6696	066066	001417			BEG	10\$;DONE IF YES.
6697	066070	162737	000400	066212	SUB	#BIT8,XA-2 ;IF NOT, DECREMENT INTENSITY.
6698	066076	023727	066174	040000	CMP	GHXTYP,#HST ;JUST DID HISTO.?
6699	066104	001004			BNE	31\$;...BR AHEAD IF NOT.
6700	066106	012737	020000	066174	MOV	#CGRPH,GHXTYP ;IF YES, NOW DO CONN-GRAPH
6701	066114	000633			BR	8\$;
6702	066116	012737	060000	066174	31\$: MOV	#FHST,GHXTYP ;DID C-GRPH, NOW DO F-HIST
6703	066124	000627			BR	8\$;....
6704						
6705	066126	00,737	027514		10\$: JSR	PC,PAUSE1
6706	066132	00,737	030072		JSR	PC,TSTEND
6707	066136				EXIT	TST ; EXIT THIS TEST >>>>>>>>>>>>>>>>>>>>>>
6708					:	
6709					: ON ERROR, SHOW X AMPLITUDE AND Y INCREMENT.	
6710					:	
6711	066142				20\$: PRINTX	#GHXYX,XA,YI ;ERROR EXTENSION.
6712	066172	000207			RTS	PC
6713					:	
6714					: DISPLAY CODE FOR GRAPH/HISTO X TEST.	
6715					:	
6716	066174	000000			GHXTYP:	0 ;FUNCTION TYPE
6717	066176	150000	000000		GHXT:	SETHB,0 ;SET BASE LINE.
6718	066202	174100			YI:	GYI.0 ;SET Y INCR.
6719	066204	114000			APNT	
6720	066206	000000	000000		O.O	
6721	066212	123774			GHX.ALL	
6722	066214	000000			XA:	0 ;X AMPLITUDE (BIT 14 = HST MODE).
6723	066216	173000			STOPN	
6724	066220	160000	066212		DJMP,XA-2	
6725						
6726	066224				END.TEST	

```
*****  
: *  
: *  
: *      END TEST 24  
: *  
: *  
: *****
```

6728

[illegible]

```

:*****
:
:
:
:
:
:*****
END TEST 25
:*****

```

```

*****
*
* BEGIN TEST 26 - BIT MAP (1)
*
*****

```

[illegible]

6828
6829
6830
6831
6832
6833
6834
6835

```
: EXECUTE BIT MAP MODE 1 WITH 4 AND 8 BIT PIXELS FOR ALL
: POSSIBLE RUN LENGTHS (0 THRU 777). TEST FINAL X/Y FOR EACH.
: IF VISUAL, DISPLAYS A WEDGE STARTING AT 0,0 WITH EVERY
: OTHER PIXEL (IN X) TURNED OFF.
: REPEAT AFTER QUICK AS BEFORE.
: PIXEL BUFFER SIZE = 256. WORDS (512. 8 BIT PIXELS MAX).
```

IN THE NEW DPU, A COUNT OF 0 DOES 512. PIXELS, SO IT'S LAST.

[illegible]

```

MOV      #SELP, @DSR      ;SAVE PCSAVE...
MOV      @DSR, IPCSAV     ;
MOV      #BM14+1, R1      ; 1ST OPCODE W/ COUNT=1.
MOV      FREE, BM1PIX     ; BUFFER ADDRESS.
1$:      JSR      PC, CLRRW ; INIT IN R/W MODE.
MOV      #2, R4           ; INIT EXPECTED X...
CLR      R5              ; ...AND Y.
2$:      MOV      R1, BM1PIX-2 ; SET 1ST/NEXT OPCODE...
MOV      R5, BM1Y        ; ...AND Y CO-ORD.

```

[illegible][illegible]

```

LOOP0 2$ ; REPEAT...
JSR PC,LOOP
BCS 2$

MOV TEMP2,LOOPK ;RESET LOOPER.
BIT #777,R1 ;DONE 512. PIXELS?

```


[illegible]

6955	067706	013737	003234	027700		MOV	TEMP2,LOOPK	;...AND RESTORE.
6956						:		
6957						:	NOW CHECK THE EXPANSION OF A 32 SQUARE ARRAY.	
6958						:		
6959	067714	012701	134000		4\$:	MOV	#BM04,R1	; 4 BIT, 32 SQUARE, NO EXP.
6960	067720	000402				SKP2		
6961	067722	012701	135000		5\$:	MOV	#BM08,R1	; 8 BIT, 32 SQUARE, NO EXP.
6962	067726	005037	070420			CLR	BMOP1+4	; 1ST BLOCK AT 0.0
6963	067732	005037	070422			CLR	BMOP1+6	
6964	067736	012704	000100			MOV	#100,R4	; SET EXP X AND Y.
6965	067742	010405				MOV	R4,R5	
6966	067744	010137	070424			MOV	R1,BMOP1+10	; SET OPCODE.
6967	067750	004737	026650			JSR	PC,CLRRW	
6968								
6969	067754				6\$:	BGNSEG		; <<<<<<<<<<<<<<<<<<<<<<<<<<<< BEGIN SEGMENT.
6970	067756	012777	070414	113046		MOV	#BMOP1,@DPC	
6971	067764	004737	070320			JSR	PC,BMOCHK	; ERROR CHECK.
6972	067770					ENDSEG		; <<<<<<<<<<<<<<<<<<<<<<<<<<<< END SEGMENT.
6973	067772					LOOPTO	6\$	; REPEAT...
	067772	004737	027646			JSR	PC,LOOP	
	067776	103766				BCS	6\$	
6974								
6975	070000	013737	003234	027700		MOV	TEMP2,LOOPK	
6976	070006	062737	000040	070420		ADD	#40,BMOP1+4	
6977	070014	062737	000040	070422		ADD	#40,BMOP1+6	
6978	070022	062704	000140			ADD	#140,R4	
6979	070026	010405				MOV	R4,R5	
6980	070030	042701	000777			BIC	#777,R1	
6981	070034	052701	000010			BIS	#EX2,R1	; EXPAND X2.
6982	070040	010137	070424			MOV	R1,BMOP1+10	
6983								
6984	070044				7\$:	BGNSEG		; <<<<<<<<<<<<<<<<<<<<<<<<<<<< BEGIN SEGMENT.
6985	070046	005277	112760			INC	@DPC	
6986	070052	004737	070320			JSR	PC,BMOCHK	; ERROR CHECK.
6987	070056					ENDSEG		; <<<<<<<<<<<<<<<<<<<<<<<<<<<< END SEGMENT.
6988	070060					LOOPTO	7\$	
	07006,	004737	027646			JSR	PC,LOOP	
	070064	103767				BCS	7\$	
6989								
6990	070066	013737	003234	027700		MOV	TEMP2,LOOPK	
6991	070074	062737	000100	070420		ADD	#100,BMOP1+4	
6992	070102	062737	000100	070422		ADD	#100,BMOP1+6	
6993	070110	062704	000300			ADD	#300,R4	
6994	070114	010405				MOV	R4,R5	
6995	070116	042701	000777			BIC	#777,R1	
6996	070122	052701	000020			BIS	#EX4,R1	; EXPAND X4.
6997	070126	010137	070424			MOV	R1,BMOP1+10	
6998								
6999	070132				8\$:	BGNSEG		; <<<<<<<<<<<<<<<<<<<<<<<<<<<< BEGIN SEGMENT.
7000	070134	005277	112672			INC	@DPC	
7001	070140	004737	070320			JSR	PC,BMOCHK	; ERROR CHECK.
7002	070144					ENDSEG		; <<<<<<<<<<<<<<<<<<<<<<<<<<<< END SEGMENT.
7003	070146					LOOPTO	8\$	; REPEAT
	070146	004737	027646					

7063	070426	000000		0	: BUFFER ADDRESS.
7064	070430	164000	173000	DNOP, STOPN	
7065	070434	160000	070416	DJMP, BMOP1+2	
7066					
7067	070440	170140		BMOP2: CLRMEM	
7068	070442	114000	000000 000000	APNT, 0, 0	
7069	070450	134000		BM04!0	
7070	070452	000000		0	
7071	070454	164000	173000	DNOP, STOPN	
7072	070460	160000	070440	DJMP, BMOP2	
7073					
7074	070464			END.TEST	

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

IMAGE MEMORY CLEAR-SET

```
*****  
* BEGIN TEST 28 - IMAGE MEMORY CLEAR-SET *  
*****  
  
; (ITERATION COUNT = 10.)  
  
TST DPJMOD ;DPU ONLY ??  
BEQ CLRTST ;NO, EXECUTE.  
EXIT TST ; EXIT THIS TEST >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>  
  
;'CLRMEM' TO ZERO, EXECUTE PIXEL-READ-BACK, AND TEST DATA.  
;'SETMEM' TO 1'S, AND REPEAT.  
DISPLAY MODE (INTERLACE/NON-INTERLACE) IS TRANSPARENT.  
  
IF SHORT MODE, CYCLE X FOR Y = 0 AND 2 (ODD VS EVEN),  
THEN CYCLE Y HOLDING X AT 0.  
REPEAT EACH PIXEL READ 10 TIMES AFTER QUICK PASS.  
IF LONG MODE, CYCLE X FOR EVERY Y (512 X 512 ADDRESSES),  
READING EACH PIXEL ONLY ONCE.  
  
CLRTST: JSR PC,TSTGO ;TITLE  
JSR PC,CONFIG ; CONFIGURE VSV SYSTEM.  
JSR PC,LUMIN ; BLAST LUT.  
BGNSUB  
CLR IMC.CH  
CLRT1: MOV IMC.CH,R1  
ASL R1  
MOV MEMTAB(R1),R1  
BNE .+6 ; BR IF MEMORY ON THIS CHANNEL  
JMP IIS$  
MOV R1,PDF  
COM R1  
MOV R1,PDM  
MOV IMC.CH,R1  
SWAB R1  
BISW3 #RDWRT,R1,CLR.ZA ; INSERT CHANNEL OF MEMORY TO TEST  
MOV R1,-(SP)  
BIS #RDWRT,(SP)  
MOV (SP)+,CLR.ZA  
TST IMTMO  
BEQ .+10 ;SKIP NEXT IF SHORT MODE...  
MOV #1,LOOPK ;...OTHERWISE, SET ONCE ONLY.  
MOV !OOPK,TEMP2 ;SAVE A COPY OF LOOPK.  
CLR R3 ;R3 = PATTERN (0'S FIRST).  
MOV #CLR.Z,@DPC ;XCT "CLEAR MEM TO 0".  
SKP2  
1$: INC @DPC ;RESUME, XCT "SETMEM TO 1'S".  
JSR PC,WAITF ;WAIT 'TIL DONE  
CLR R1 ;INIT X ADDRESS...  
CLR R2 ;....AND Y ADDRESS.  
  
2$: MOV R2,@DYR ;SET Y ADDRESS.  
JSR PC,WAITF
```


[illegible]

7236	071464				CLP.D	2	
7237	071466				CLP.D	1	
7238	071470				CLMSHT: CLP.D	14	
7239	071472				CLP.D	11	
7240	071474				CLP.D	12	
7241	071476				CLP.D	3	
7242	071500				CLP.D	5	
7243	071502				CLP.D	6	
7244	071504				CLP.D	7	
7245	071506	001774			1774		
7246	071510	177777			-1		
7247	071512	000000	000000	000000	0.0.0.0.0.0		
7248	071526	000000	000000	000000	0.0.0.0.0.0.0.0		
7249							
7250					;DISPLAY FILES FOR CLEARING MEMORY:		
7251	071546	176074	176474	177074	CLR.Z2: RDWRT.CH0,RDWRT.CH1,RDWRT.CH2,RDWRT.CH3		
7252	071556	170140	173000		CLRMEM,STOPN		
7253							
7254	071562	117774			CLR.D2: APNT.ALL		
7255	071564	176074	176474	177074	RDWRT.CH0,RDWRT.CH1,RDWRT.CH2,RDWRT.CH3		
7256	071574	170100	173000		SETMEM,STOPN		
7257							
7258							
7259							
7260					; SUBROUTINE - READ & VERIFY 1 LINE OF IMAGE MEMORY.		
7261					; ENTER WITH:		
7262					IMC.CH = MEMORY CHANNEL TO READ (0, 1, 2, OR 3)		
7263					IMC.Y = Y POSITION OF LINE TO BE READ		
7264					IMC.RA = ADDRESS OF 512.-BYTE BUFFER TO READ DATA INTO		
7265					IMC.TA = ADDRESS OF 512.-BYTE BUFFER TO TEST AGAINST		
7266					(DATA MUST INCLUDE 1'S IN NON-EXISTENT BIT POSITIONS TO		
7267					CATCH BAD DRIVERS & DBUS LINES)		
7268							
7269							
7270	071600	013700	072414		IMCHK:	MOV	IMC.CH,R0
7271	071604	000300				SWAB	R0
7272	071606	042737	001400	072430		BIC	#BIT9:BIT8,IMC.ME
7273	071614	050037	072430			BIS	R0,IMC.ME
7274	071620	013737	072352	072354		MOV	IMCTRY,IMCRCT
7275	071626	012777	072420	111176	IMCHKR:	MOV	#IMC.SR,@DPC
7276	071634	004737	027454			JSR	PC,WAITF
7277	071640	013702	072450			MOV	IMC.RA,R2
7278	071644	062702	001000			ADD	#512.,R2
7279	071650	013703	072440			MOV	IMC.IR,R3
7280	071654	027702	111152			CMP	@DPC,R2
7281	071660	001003				BNE	1\$
7282	071662	027703	111146			CMP	@DSR,R3
7283	071666	001410				BEQ	2\$
7284	071670				1\$:		
7285	071670					HRDERR	IMRF,PCSERR
7286	071710	005004			2\$:	CLR	R4
7287	071712	013705	072436			MOV	IMC.Y,R5
7288	071716	062705	000002			ADD	#2,R5
7289	071722	027704	111110			CMP	@DXR,R4
7290	071726	001003				BNE	3\$
7291	071730	027705	111104			CMP	@DYR,R5
7292	071734	001410				BEQ	4\$

;GET CHANNEL #
 ;... INTO HIGH BYTE OF R0.
 ;CLEAR OLD CHANNEL # FROM ENABLING INSTR.
 ;SET IN THE NEW CHANNEL
 ;START THE READ.
 ;... WAIT FOR IT TO FINISH.
 ;R2 GETS FINAL ADDRESS
 ;R3 GETS WHAT DSR SHOULD BE.
 ;IS DPC CORRECT?
 ;IS DSR CORRECT?
 ;PRINT IF NOT.
 ;DXR SHOULD BE 0.
 ;DYR SHOULD BE Y+2
 ;TEST DXR
 ;TEST DYR

7293 071736				3\$:	HRDERR	IMRF,XYERR	
7294 071736				4\$:	MOV	#0,R1	:R1 = X ADDRESS UNDER TEST.
7295 071756	012701	000000			MOV	IMC.TA,R4	:R4 POINTS TO EXPECTED DATA
7296 071762	013704	072416			MOV	IMC.RA,R5	:R5 POINTS TO RECEIVED DATA
7297 071766	013705	072450		5\$:	CMP	(R4)+,(R5)+	:CHECK THE DATA
7298 071772	022425				BNE	10\$	
7299 071774	001006			6\$:	ADD	#4,R1	:BUMP THE X ADDRESS
7300 071776	062701	000004			CMP	R1,#MAXX+2	
7301 072002	020127	002000			BLT	5\$	
7302 072006	002771				BR	9\$	:EXIT WHEN DONE WITH LINE.
7303 072010	000471						
7304							
7305 072012	005337	072354		10\$:	DEC	IMCRCT	
7306 072016	100402				BMI	.+6	
7307 072020	000137	071626			JMP	IMCHKR	
7308 072024	010137	072346			MOV	R1,IMCBDA	:GET POSITION OF BAD DATA
7309 072030	12 465	177776	177776		CMPB	-2(R4),-2(R5)	:LOW BYTE BAD?
7310 072036	001440				BEQ	12\$	:BR IF NOT.
7311 072040	116437	177776	023056		MOVB	-2(R4),GDDAT	:IF YES, GET THE DATA
7312 072046	116537	177776	023060		MOVB	-2(R5),BADDAT	
7313 072054	006337	023056		11\$:	ASL	GDDAT	:SHIFT IT TO LOOK LIKE DBUS DATA
7314 072060	006337	023056			ASL	GDDAT	
7315 072064	006337	023060			ASL	BADDAT	
7316 072070	006337	023060			ASL	BADDAT	
7317 072074	042737	176003	023056		BIC	#^C1774,GDDAT	
7318 072102	042737	176003	023060		BIC	#^C1774,BADDAT	
7319 072110					HRDERR	IMDI,IMCERR	
7320 072130	032737	000002	072346		BIT	#2,IMCBDA	:DONE HIGH BYTE?
7321 072136	001317				BNE	6\$	:IF YES, GO BACK TO MAIN LOOP.
7322 072140	062737	000002	072346	12\$:	ADD	#2,IMCBDA	
7323 072146	126465	177777	177777		CMPB	-1(R4),-1(R5)	:COMPARE HIGH BYTES
7324 072154	001710				BEQ	6\$	
7325 072156	116437	177777	023056		MOVB	-1(R4),GDDAT	
7326 072164	116537	177777	023060		MOVB	-1(R5),BADDAT	
7327 072172	000730				BR	11\$	
7328							
7329 072174	000207			9\$:	RTS	PC	
7330							
7331 072176				BGNMSG	IMCERR		
7332 072176					PRINTX	TNAM	
7333 072216	013737	072346	072350		MOV	IMCBDA,IMCRAM	
7334 072224	006237	072350			ASR	IMCRAM	
7335 072230					PRINTX	#IMDIXC,IMCBDA,IMC.Y,PDF,GDDAT,BADDAT	
7336 072274					PRINTX	#IMDIX2,IMCRAM,IMC.CH	
7337 072324					PRINTX	#NULCR	:PRINT BLANK LINE
7338 072344				ENDMSG			
7339							
7340 072346	000000				IMCBDA:	.WORD 0	:X POSITION FOR PRINTOUT
7341 072350	000000				IMCRAM:	.WORD 0	:SUSPECT RAM
7342 072352	000000				IMCTRY:	.WORD 0	
7343 072354	000000				IMCRCT:	.WORD 0	
7344							
7345 072356	040	040	104	IMRF:	.ASCIZ /	DMA PIXEL READBACK FAILURE/	
7346					.EVEN		
7347							
7348	163000				IMR8=	163000	
7349 072414	000000			IMC.CH:	0		

7350	072416	000000			IMC.TA: 0	: ADDRESS OF TEST BUFFER
7351	072420	176000	176400	177000	IMC.SR: PROTEC!CH0,PROTEC!CH1,PROTEC!CH2,PROTEC!CH3	
7352	072430	176074			IMC.ME: RDWRT	: ENABLE SELECTED MEMORY
7353	072432	114000	000000		APNT,0	: START AT X=0
7354	072436	000000			IMC.Y: 0	: Y
7355	072440	163006			IMC.IR: IMR8!6	: READ, 2 8-BIT PIXELS/WORD, ALL BITS
7356	072442	001000	000001		512.,1	: 1 FULL LINE
7357	072446	000400			256.	: WORD COUNT = 256.
7358	072450	160000			IMC.RA: .WORD 1600C0	: READ BUFFER
7359	072452	173000			STOPN	: (SHOULD STOP BY ITSELF)
7360	072454				END.TEST	

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

7362

7363	072536	005737	002502
7364	072542	001402	
7365	072544		
7366			
7367			
7368			
7369			
7370			
7371			
7372	072550		
	072550	004737	027702
	072554	004737	026074
7373	072560	004737	030414
7374	072564	005005	
7375	072566	004737	026650
7376			
7377	072572		
7378	072574	010504	
7379	072576	006304	
7380	072600	016401	003256
7381	072604	010103	
7382	072606	001450	
7383	072610	010504	
7384	072612	000304	
7385	072614	052704	176034
7386	072620	010437	072776
7387	072624		
	072624	012777	072766
	072632	004737	027454
7388	072636	032737	020000
7389	072644	001401	
7390	072646	005001	
7391	072650	012777	000002
7392	072656	004737	027454
7393	072662	012777	000000
7394	072670	017702	110140
7395	072674	042702	176003
7396	072700	074203	
7397	072702	040302	
7398	072704		
	072704	020102	
	072706	001410	
	072710		
	072730		
7399	072730	005205	
7400	072732	020527	000004
7401	072736	001316	
7402			

[illegible]

IMAGE MEMORY PATTERNS

7417

[illegible]

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

[illegible]

```

7573 074054 023737 072436 003250      CMP      IMC.Y,YMAX      ;DONE LAST LINE?
7574 074062 101711                      BLOS      5$              ;IF NOT, GO FOR MORE
7575 074064 005237 072414                INC      IMC.CH      ;BUMP THE CHANNEL NUMBER.
7576 074070 023727 072414 000004      8$:    CMP      IMC.CH,#4    ;DONE ALL CHANNELS?
7577 074076 002634                      BLT       3$              ;IF NOT, DO THEM.
7578 074100                      ENDSUB
7579
7580 074102 004737 030072                JSR      PC,TSTEND
7581 074106                      EXIT      TST
7582 074112 000000                      PATS:    0
7583 074114 000524                      0524
7584 074116 001250                      1250
7585 074120 001774                      1774
7586 074122 177777                      -1
7587
7588 074124 116000                      WRT.D:    APNT.L0      ;PIXEL DATA SET ON THE FLY.
7589 074126 040000                      WRT.X:    1.0        ;LIKEWISE ADDRESSES.
7590 074130 000000                      WRT.Y:    0
7591 074132 164000 164000 164000        DNOP,DNOP,DNOP      ;IN CASE WE WANT TO CHANGE...
7592 074140 164000 164000 164000        DNOP,DNOP,DNOP      ;...MODE FOR P-R-B DEBUG.
7593 074146 173000                      STOPN      ;STOP AND TEST THIS PIXEL.
7594
7595      ;+++++
7596      .MACRO PAT.D MDX
7597      <MDX*BIT8>!<MDX*BIT4>.<-MDX-1&17*BIT12>.<-MDX-1&17>
7598      .ENDM
7599
7600 074150                      MPAT2:    PAT.D    0      ;PUT 4-BIT PATTERN INTO BITS <7:4> AND <11:8>.
7601 074152                      PAT.D    1      ;...AND ITS COMPLEMENT INTO <15:12> AND <3:0>
7602 074154                      PAT.D    2      ;USE FLOATING 1'S AND 0'S
7603 074156                      PAT.D    4
7604 074160                      PAT.D    10
7605 074162                      PAT.D    7
7606 074164                      PAT.D    13
7607 074166                      PAT.D    15
7608 074170                      PAT.D    16
7609                      000011      MLEN=-MPAT2/2      ;USE ODD # OF WORDS TO MAXIMIZE CHANCES OF FINDING
7610                      ;          DUAL-ADDRESSING PROBLEMS.
7611
7612 074172 000000                      MPOFF:    0      ;OFFSET FOR WRITING AND COMPARING
7613 074174 000000                      MPCBAS:    0      ;BASE ADDRESS OF COMPARE BUFFER
7614
7615      ;COMMANDS FOR WRITING THE PATTERN INTO THE LINE
7616 074176 176000 176400 177000      MPT.ST:    PROTEC.CH0,PROTEC.CH1,PROTEC.CH2,PROTEC.CH3
7617 074206 176034                      MPT.WE:    WRT!CH0      ;ENABLE A CHANNEL FOR WRITING
7618 074210 114000 000000                      APNT,0      ;START AT X=0
7619 074214 000000                      MPT.WY:    0
7620 074216 137000                      BM18      ;USE BIT-MAP-1, WITH A COUNT OF 512. PIXELS
7621 074220 000000                      MPT.WB:    0
7622 074222 173000                      STOPN
7623
7624      ;+++++
7625 074224                      END.TEST

;*****
;
;
;      END TEST 30

```

.....

7639

```
*****  
* BEGIN TEST 32 - SYSTEM VERIFICATION DISPLAY *  
*****
```

; (ITERATION COUNT = 1.)

```
TST    DPJMOD      ;DPU ONLY ??  
BEQ    VIS1        ;NO, EXECUTE.  
EXIT   TST         ; EXIT THIS TEST >>>>>>>>>>>>>>>>>>>>>>>>>>>>>
```

:
; SMORGASBORD, ONE FRAME THAT HAS A BIT OF EVERYTHING.
; FIRST, A DEC LOGO (SORT OF), THEN VIOLATE THE EDGES
; AT LOWER LEFT AND UPPER RIGHT.

```
VIS1: JSR    PC,TSTGO  
JSR    PC,CLRRW     ; CLEAR.  
JSR    PC,DBLINK    ; DISABLE BLINKING.  
JSR    PC,LUMIN     ; BLAST THE LUT.
```

```
1$: MOV    #LOGO,@DPC ; START WITH THE LOGO.  
JSR    PC,WAITF  
;  
JSR    PC,PAUS.5  
BR     VIS2         ;...AND CONTINUE.
```

LOGO:

```
APNT, 540, 100  
SETCB,ACAT  
CHAR!ALL  
.ASCIZ /DIGITAL/  
  
APNT,140,140       ;EXCEED EDGES AT LOWER LEFT...  
LVEC  
LXY    I,-200,0  
LXY    I,0,-200  
LXY    I,200,0  
LXY    I,0,200  
APNT,1640,1540    ;...AND AT UPPER RIGHT.  
LVEC  
LXY    I,200,0  
LXY    I,0,300  
LXY    I,-200,0  
LXY    I,0,-300  
STOPN
```

```

7677
7678
7679
7680 074556 012777 074616 106246 VIS2: MOV #LOCT,@DPC ;1 LARGE ONE...
7681 074564 004737 027454 JSR PC,WAITF
7682 074570 012777 074670 106234 MOV #SOCT1,@DPC ;...AND 2 SMALL ONES.
7683 074576 004737 027454 JSR PC,WAITF
7684 074602 012777 074702 106222 MOV #SOCT2,@DPC
7685 074610 004737 027454 JSR PC,WAITF
7686 JSR PC,PAUS.5
7687 074614 000456 BR VIS3 ;...AND CONTINUE.
7688
7689 074616 114000 000500 000002 LOCT: APNT,500.2 ; 1 LARGE OCTAGON...
7690 074624 113774 LVEC!ALL
7691 074626 LXV 1,576.0
7692 074632 LXV 1,436.436
7693 074636 LXV 1,0,576
7694 074642 LXV 1,-436.436
7695 074646 LXV 1,-576.0
7696 074652 LXV 1,-436,-436
7697 074656 LXV 1,0,-576
7698 074662 LXV 1,436,-436
7699 074666 173000 STOPN
7700
7701 074670 114000 000220 000620 SOCT1: APNT,220.620 ;...AND 2 SMALL ONES.
7702 074676 160000 074710 DJMP,+.10
7703 074702 114000 001460 000620 SOCT2: APNT,1460.620
7704 074710 107774 SVEC.ALL
7705 074712 SXY 1,76.0
7706 074714 SXY 1,56.56
7707 074716 SXY 1,0,76
7708 074720 SXY 1,-56.56
7709 074722 SXY 1,-76.0
7710 074724 SXY 1,-56,-56
7711 074726 SXY 1,0,-76
7712 074730 SXY 1,56,-56
7713 074732 130000 RPNT ;5 REL POINTS IN CENTER.
7714 074734 SXY U,20.50 ;CENTER.
7715 074736 SXY 1,20.50
7716 074740 SXY 1,20,0
7717 074742 SXY 1,-20.20
7718 074744 SXY 1,-20,-20
7719 074746 SXY 1,20,-20
7720 074750 173000 STOPN
  
```

```

7722
7723      ; NEXT, SHOW GRAPH/HISTO Y IN UPPER LEFT.
7724      ;
7725 074752 012737 011240 075124 VIS3: MOV #1240!GHIINH,GY+2 ;PLOT GRAPH Y STARTING AT 1240.GHIINH.
7726 074760 012777 075100 106044      MOV #GRY,@DPC ;START GRAPH Y.
7727 074766 000402      SKP2
7728 074770 005277 106036      18: INC @DPC
7729 074774 004737 027454      JSR PC,WAITF
7730 075000 042737 010000 075124      BIC #GHIINH,GY+2
7731 075006 062737 000004 075124      ADD #4,GY+2 ;INCR Y AMPLITUDE...
7732 075014 023727 075124 001604      CMP GY+2,#1604
7733 075022 001362      BNE 18 ;LOOP UNTIL DONE.
7734
7735 075024 012737 051240 075152      MOV #HST!1240!GHIINH,HY+2 ;NOW, HISTO Y STARTING AT 1240.GHIINH.
7736 075032 012777 075134 105772      MOV #HSY,@DPC
7737 075040 000402      SKP2
7738 075042 005277 105764      28: INC @DPC
7739 075046 004737 027454      JSR PC,WAITF
7740 075052 042737 010000 075152      BIC #GHIINH,HY+2
7741 075060 062737 000010 075152      ADD #10,HY+2 ;INCR Y AMPLITUDE...
7742 075066 023727 075152 041550      CMP HY+2,#HST!1550
7743 075074 001362      BNE 28 ;...AND LOOP TIL DONE.
7744
7745      ; JSR PC,PAUS.5
7746 075076 000431      BR VIS4 ;...AND CONTINUE.
7747
7748 075100 114000 000100 001240 GRV: APNT,100,1240
7749 075106 113774      LVEC,ALL ;DRAW BASE LINE.
7750 075110      LXV 1,340,0
7751 075114      LXV U,-340,0
7752 075120 174104      GXI:4 ;X INCR = 4 FOR GRAPH.
7753 075122 127774 001240      GY: GHY:ALL,0!1240
7754 075126 173000 160000 075122      STOPN,DJMP,GY
7755
7756 075134 150000 001240      HSY: SETHB,1240 ;HISTO BASE LINE.
7757 075140 114000 000140 001240      APNT,140,1240
7758 075146 174110      GXI:10 ;X INCR = 10 FOR HISTO.
7759 075150 127774 041240      HY: GHY:ALL,HST!1240
7760 075154 173000 160000 075150      STOPN,DJMP,HY
  
```

```

7762
7763      : NEXT, GRAPH/HISTO X IN LOWER RIGHT.
7764      :
7765 075162 012737 011340 075334 VIS4: MOV      #1340!GHIINH,GX+2 ;PLOT GRAPH X STARTING AT 1340!GHIINH.
7766 075170 012777 075310 105634      MOV      #GRX,@DPC      ;START GRAPH X.
7767 075176 000402      SKP2
7768 075200 005277 105626      1$: INC      @DPC
7769 075204 004737 027454      JSR      PC,WAITF
7770 075210 042737 010000 075334      BIC      #GHIINH,GX+2
7771 075216 062737 000004 075334      ADD      #4,GX+2      ;INCR X AMPLITUDE...
7772 075224 023727 075334 001704      CMP      GX+2,#1704
7773 075232 001362      BNE      1$      ;LOOP UNTIL DONE.
7774
7775 075234 012737 051340 075362      MOV      #HST!1340!GHIINH,HX+2 ;NOW, HISTO X STARTING AT 1340!GHIINH.
7776 075242 012777 075344 105562      MOV      #HSX,@DPC
7777 075250 000402      SKP2
7778 075252 005277 105554      2$: INC      @DPC
7779 075256 004737 027454      JSR      PC,WAITF
7780 075262 042737 010000 075362      BIC      #GHIINH,HX+2
7781 075270 062737 000010 075362      ADD      #10,HX+2      ;INCR X AMPLITUDE...
7782 075276 023727 075362 041650      CMP      HX+2,#HST.1650
7783 075304 001362      BNE      2$      ;...AND LOOP TIL DONE.
7784
7785      : JSR      PC,PAUS.5
7786 075306 000431      BR      VIS5      ;...AND CONTINUE.
7787
7788 075310 114000 001340 000100 GRX:  APNT,1340,100
7789 075316 113774      LVEC!ALL      ;DRAW BASE LINE.
7790 075320      LXV      1,0,340
7791 075324      LXV      0,0,-340
7792 075330 174104      GYI!4      ;Y INCR = 4 FOR GRAPH.
7793 075332 123774 001340      GX:  GHX!ALL,0!1340
7794 075336 173000 160000 075332      STOPN,DJMP,GX
7795
7796 075344 150000 001340      HSX:  SETHB,1340      ;HISTO BASE LINE.
7797 075350 114000 001340 000140      APNT,1340,140
7798 075356 174110      GYI!10      ;Y INCR = 10 FOR HISTO.
7799 075360 123774 041340      HX:  GHX!ALL,HST!1340
7800 075364 173000 160000 075360      STOPN,DJMP,HX
  
```

```

7802
7803      ; NEXT SHOW BIT MAP MODE 1 AT BOTTOM CENTER.
7804      ; AND BIT MAP MODE 0 AT TOP CENTER.
7805
7806 075372 012700 177777      VIS5:      MOV      #-1,R0      ; SET PIXEL DATA PATTERN...
7807 075376 004737 027020      JSR      PC,FILL2      ;...AND FILL BUFFER WITH...
7808                                     ;...ALTERNATE WORDS.
7809
7810 075402 013737 003240 075552      MOV      FREE,B148      ; SET BUFFER ADDRESS.
7811 075410 012737 136340 075550      MOV      #BM14!340,B148-2 ; SET OPCODE = BM14 X 340
7812 075416 012777 075530 105406      MOV      #B14,@DPC      ; XCT BM14, DRAWS A DASHED...
7813 075424 004737 027454      JSR      PC,WAITF      ;...LINE (4 ON, 4 OFF)...
7814                                     ;...FOR 340 PIXELS.
7815 075430 012737 137400 075550      MOV      #BM18!400,B148-2 ; SET OPCODE = BM18 X 400
7816 075436 012777 075542 105366      MOV      #B18,@DPC      ; XCT BM18, DRAWS A DASHED...
7817 075444 004737 027454      JSR      PC,WAITF      ;...LINE (2 ON, 2 OFF)...
7818                                     ;...FOR 400 PIXELS.
7819
7820 075450 012700 000377      MOV      #377,R0
7821 075454 004737 027014      JSR      PC,FILL1
7822
7823 075460 013737 003240 075600      MOV      FREE,B048      ; SET BUFFER ADDRESS.
7824                                     ;
7825 075466 012737 134010 075576      MOV      #BM04!M64,B048-2 ; 4 BIT X 64 SQUARE.
7826 075474 012777 075556 105330      MOV      #BM04!M32!EX2,B048-2 ; 4 BIT X 64 SQUARE.
7827 075502 004737 027454      MOV      #B04,@DPC      ; DO IT.
7828                                     ;
7829 075506 012737 135010 075576      MOV      #BM08!M64,B048-2 ; 8 BIT X 64 SQUARE.
7830 075514 012777 075570 105310      MOV      #BM08!M32!EX2,B048-2 ; 8 BIT X 64 SQUARE.
7831 075522 004737 027454      MOV      #B08,@DPC      ; AND DO THAT TOO.
7832      JSR      PC,WAITF
7833      1$:      JSR      PC,PAUS.5
7834 075526 000430      BR      VIS5A      ;...AND CONTINUE.
7835
7836 075530 114000 000400 000240 B14:      APNT,400,240      ; 4 ON 4 OFF FOR 340 PIXELS.
7837 075536 160000 075550      DJMP, .+10
7838 075542 114000 000300 000340 B18:      APNT,300,340      ; 2 ON 2 OFF FOR 400 PIXELS.
7839 075550 136340      BM14!340
7840 075552 000000      B148:      0
7841 075554 173000      STOPN
7842
7843 075556 114000 000540 001340 B04:      APNT,540,1340
7844 075564 160000 075576      DJMP, .+10
7845 075570 114000 001040 001340 B08:      APNT,1040,1340
7846 075576 134001      BM04!M64
7847 075600 000000      B048:      0
7848 075602 164000 164000      DNOP, DNOP
7849 075606 173000      STOPN
  
```

```

7851
7852
7853
7854
7855
7856
7857 075610 112737 000004 075671
7858 075616 012777 075660 105206
7859 075624 000403
7860 075626 012777 000001 105176
7861 075634 004737 027454
7862 075640 123727 075671 000024
7863 075646 103003
7864 075650 105237 075671
7865 075654 000764
7866 075656
7867 075656 000430
7868
7869
7870 075660 114000 000300 000320
7871 075666 144040
7872 075670 000 004
7873 075672 006020 006040 006060
7874 075700 006100 006120 006140
7875 075710 006200 006220 006240
7876 075720 006300 006320 006340
7877 075730 173000
7878 075732 000000
7879 075734 160000 075670

; 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
        MOV     #1,@DPC        ;RESUME THE DISPLAY
        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.
        JSR     PC,PAUS.5       ;PAUSE
        BR      VIS6           ;...THEN CONTINUE.
        3$:
;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

```

```

7881      : FINALLY, AT DEAD-CENTER:
7882      : A CIRCULAR SWEEP FROM 12 O'CLOCK THRU 360 DEGREES.
7883      : INCREMENT INTENSITY/COLOR LEVEL SUCH THAT ALL LEVELS
7884      : ARE DISPLAYED IN 360 DEGREES.
7885      : 3 VECTORS PER LEVEL * 256 LEVELS = 768 VECTORS TOTAL.
7886      : (OCTAL 3 * 400 = 1400)
7887      : ACTUAL COLOR/INTENSITY SPECTRUM DEPENDS UPON HOW MANY
7888      : PIXEL BITS ARE AVAILABLE, AND HOW THE LOOK-UP-RAM IS
7889      : PROGRAMMED (GREY-SCALE BY DEFAULT, IF INSTALLED).
7890      :
7891      :     2 BITS <9:8> = 4 LEVELS.
7892      :     4 BITS <9:6> = 16 LEVELS (4 PER QUADRANT).
7893      :     6 BITS <9:4> = 64 LEVELS (LUT REQ'D).
7894      :     8 BITS <9:2> = 256 LEVELS (LUT REQ'D).
7895
7896 075740 012737 112000 076400 VIS6:    MOV        #LVEC!LO,RBL      ;INIT DISPLAY FILE.
7897 075746 012737 040000 076402          MOV        #I!0,RBX
7898 075754 012737 000300 076404          MOV        #300,RBY
7899 075762 012701 000003              : SET FOR 3 VECTORS PER LEVEL...
7900 075766 012703 000004              : ...AND INCR LEVEL BY 'L1'.
7901 075772 010102              1$:      MOV        R1,R2           ; SCRATCH COPY LEVEL COUNTER.
7902 075774 012777 076372 105030          MOV        #RBO,@DPC       ; START, QUAD 1.
7903 076002 004737 027454              JSR         PC,WAITF
7904 076006 023727 076404 000000 2$:      CMP        RBY,#0         ; DELTA Y MIN ??
7905 076014 001403              BEQ         3$             ; DONE QUAD 1 IF SO.
7906 076016 004737 076230              JSR         PC,Q1
7907 076022 000771              BR         2$
7908 076024 052737 020000 076404 3$:      BIS        #MXY,RBY       ; QUAD 2, DX POS, DY NEG.
7909 076032 023727 076402 040000 4$:      CMP        RBX,#I!0       ; DELTA X MIN ??
7910 076040 001403              BEQ         5$             ; DONE QUAD 2 IF SO.
7911 076042 004737 076266              JSR         PC,Q2
7912 076046 000771              BR         4$
7913 076050 052737 020000 076402 5$:      BIS        #MXY,RBX       ; QUAD 3, DX NEG, DY NEG.
7914 076056 023727 076404 020000 6$:      CMP        RBY,#MXY.0   ; DELTA Y -MIN ??
7915 076064 001403              BEQ         7$             ; DONE QUAD 3 IF SO.
7916 076066 004737 076230              JSR         PC,Q3
7917 076072 000771              BR         6$
7918 076074 042737 020000 076404 7$:      BIC        #MXY,RBY       ; QUAD 4, DX NEG, DY POS.
7919 076102 023727 076402 060000 8$:      CMP        RBX,#I.MXY!0   ; DELTA X -MIN ??
7920 076110 001403              BEQ         9$             ; DONE QUAD 4 IF SO.
7921 076112 004737 076266              JSR         PC,Q4
7922 076116 000771              BR         8$
7923
7924 076120 012737 004000 003220 9$:      MOV        #4000,SHADLY    ; FINALLY, SET SHADING DELAY...
7925 076126 012705 110670              MOV        #NTSC8+2,R5      ; ...AND CYCLE THRU 16 SHADES...
7926 076132 012501              10$:     MOV        (R5)+,R1      ; ...OF EACH BASIC (NTSC) COLOR.
7927 076134 001403              BEQ         11$            ; BR WHEN ALL DONE.
7928 076136 004737 027172              JSR         PC,SHAD16
7929 076142 000773              BR         10$
7930
7931 076144 004737 104626              11$:     JSR         PC,OUTLIN    ; OUTLINE THE SCREEN.
7932 076150 004737 027514              JSR         PC,PAUSE1
7933 076154 004737 027514              JSR         PC,PAUSE1
7934 076160 004737 027514              JSR         PC,PAUSE1      ; PAUSE 3 SECONDS...
7935 076164 004737 105174              JSR         PC,EBLINK     ; ENABLE BLINKING.
7936 076170 004737 027514              JSR         PC,PAUSE1
7937 076174 004737 027514              JSR         PC,PAUSE1      ; PAUSE 3 SELONS...

```

VISXIT:

```
JSR      PC,PAUSE1
JSR      PC,DBLINK
JSR      PC,PAUSE1
JSR      PC,PAUSE1
JSR      PC,TSTEND
EXIT     TST
```

; DISABLE BLINKING.

[illegible]

```

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:  1!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
;
;*****
  
```

```
;*****  
; BEGIN TEST 33 - JOY-STICK VERIFICATION (STAND-ALONE)  
;*****  
  
; (ITERATION COUNT = 1.)  
  
TST      TESTK  
BEQ      JSV          ; BR IF CLEAR TO RUN.  
EXIT     TST         ; EXIT THIS TEST >>>>>>>>>>>>>>>>>>>>>>  
  
; THIS TEST WILL RUN ONLY IF IT IS SPECIFIED BY ITSELF, AND IF MANUAL  
; INTERVENTION IS ALLOWED (UAM=0), AND, OBVIOUSLY, ONLY IF A JOY-STICK  
; IS AVAILABLE.  
  
; ONCE STARTED, THERE IS NO EXIT, SAVE <^C>.  
  
JSV:    JSR        PC,TSTGO           ; TITLE.  
  
MANUAL  
BCOMLETE                JSVA1  
PRINTF #NOMAN  
EXIT   TST              ; EXIT THIS TEST >>>>>>>>>>>>>>>>>>>>>>  
JSVA1: PRINTF #SCOUT  
JSVA:  JSR        PC,CONFIG           ; FIND THE CHANNELS  
CLR     SELCHA          ; SET SELECTED CHANNEL TO 0 (DEFAULT).  
GMANID MCHAN,SELCHA,0,1400,0,3,YES  
MOV     #CUON!CUIOFF,JCXAX  
BIS     SELCHA,JCXA  
MOV     #CUIM,JCXB  
BIS     SELCHA,JCXB  
MOV     #CUIS,JCXC  
BIS     SELCHA,JCXC  
MOV     #CURD,JCXD  
BIS     SELCHA,JCXD  
MOV     SELCHA,JSPIXA  
ASR     JSPIXA  
ASR     JSPIXA  
BIS     #CUWT,JSPIXA  
MOV     #CURD,10$  
BIS     SELCHA,10$  
MOV     UNITN,R1  
MOV     SELCHA,R0            ; GET CHANNEL NO. (BITS 9-8)  
SWAB    R0  
ASL     R0                  ; MAKE WORD OFFSET  
TST     SYCTAB(R0)          ; IS THERE A JOYSTICK HERE?  
BMI     3$  
BR      2$  
; TSTB STIK(R1)             ; STICK AVAILABLE THIS UNIT ??  
; BGT JSTST                 ; YES, GET ON WITH IT.  
; *****  
; BEQ 2$                    ; NO, CAN'T RUN.  
; 240, 240  
; *****  
MOV     #1$,@DPC           ; DON'T KNOW, LET'S FIND OUT.
```

```

8058 077146 000403
8059 077150 164000
8060 077152 146100 173000
8061 077156 004737 027454
8062 077162 103416
8063
8064
8065
8066
8067 077164 004737 026756
8068 077170 006200
8069 077172
8070 077214 000137 076744
8071
8072 077220
8073 077220 000431
8074
8075 077222 045 101 040 NOSTIK: .ASCIZ /%A -- NON-EXISTENT SYNC CHANNEL : %01/
8076 .EVEN
8077
8078
8079
8080
8081 077270 103 110 101 MCHAN: .ASCIZ /CHANNEL /
8082 .EVEN
8083 077302 000000 SELCHA: 0

1$: SKP3
DNOP ;TRY TO READ CURSOR POSITION.
10$: CURD,STOPN
JSR PC,WAITF
BCS 3$ ;BR IF DPU RESPONDS TO STICK.
;*****
; CLRB STIK(R1) ;OTHERWISE, SET FLAG = NO STICK.
; 240, 240
;*****
JSR PC,RELEAS ;RELEASE 'HUNG' DPU...
2$: ASR R0
PRINTF #NOSTIK,R0 ;...TELL THE MAN...
JMP JSVA ;KEEP TRYING (FOREVER) !!

3$: ; MOVB #1,STIK(R1) ;SET FLAG = STICK AVAILABLE...
BR JSTST ;...AND START 'EM UP.

; CHANNEL SELECTION QUESTION AND ANSWER.
;
MCHAN: .ASCIZ /CHANNEL /
.EVEN
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
      077410 004737 077752
8106 077410 023737 100004 003170
8107 077414
8108 077422 001103
8109 077424 023737 100006 003172
8110 077432 001077
8111 077434 005277 103372
8112 077440 000002
8113
8114 077442 052737 001777 100470
8115 077450 017737 103362 003170
8116 077456 100404
8117 077460 017737 103354 003172
8118 077466 100010
8119 077470
8120 077510 000433
8121 077512 042737 001777 100470
8122 077520 004737 023524
8123 077524 017737 103306 003170
8124 077532 100004
8125 077534 017737 103300 003172
8126 077542 100410
8127 077544
8128 077564 042737 174000 003170
8129 077572 042737 174000 003172
8130 077600
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
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$: BREAK ; ON 'STOP', BREAK...
        JSR PC,GT CURS ; GET & MASK THE CURSOR DATA
        CMP MKDXR,SDXR ; ...AND CHECK STICK MOTION.
        BNE 6$ ; BR IF STICK IS MOVING.
        CMP MKDYR,SDYR
        BNE 6$
        INC @DPC ; DITTO
        RTI ; 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 JSDXYPE,JXYPE ; 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      62$:
8141 077654 013737 100414 100336      MOV      JSX,EX
8142 077662 013737 100416 100340      MOV      JSX+2,EX+2
8143 077670 013737 100434 100352      MOV      JSY,EY
8144 077676 013737 100436 100354      MOV      JSY+2,EY+2      ;ERASE OLD X/Y.
8145 077704 013701 003170      MOV      SDXR,R1
8146 077710 004737 027324      JSR      PC,CNVRT      ;CONVERT X TO ASCII...
8147 077714 010137 100414      MOV      R1,JSX
8148 077720 010237 100416      MOV      R2,JSX+2      ;...UPDATE READ-OUT.
8149 077724 013701 003172      MOV      SDYR,R1
8150 077730 004737 027324      JSR      PC,CNVRT      ;CONVERT Y TO ASCII...
8151 077734 010137 100434      MOV      R1,JSY
8152 077740 010237 100436      MOV      R2,JSY+2      ;...UPDATE READ-OUT.
8153 077744 012716 077376      MOV      #1$, (SP)      ; SET PC TO RESTART...
8154 077750 000002      RTI      ;...AND DISMISS.
8155 077752 017737 103060 100004      GTCURS: MOV      @DXR,MKDXR
8156 077760 017737 103054 100006      MOV      @DYR,MKDYR
8157 077766 042737 174000 100004      BIC      #174000,MKDXR
8158 077774 042737 174000 100006      BIC      #174000,MKDYR
8159 100002 000207      RTS      PC
8160 100004 000000      MKDXR: 0
8161 100006 000000      MKDYR: 0
8162
8163      ;ERROR MESSAGES FOR JOYSTICK OPERATIONS:
8164 100010      040      040      127      JSWTEX: .ASCIIZ / WAITED TOO LONG FOR J.S. STATUS RETRIEVAL/
8165 100064      040      040      104      JSDXYE: .ASCIIZ / DXR OR DYR SIGN NOT SET ON J.S. RETRIEVAL/
8166 100140      040      040      104      DXYME: .ASCIIZ / DXR OR DYR SIGN SET ON J.S. MATCH INTERRUPT/
8167 100216      045      101      040      JXYPX: .ASCIIZ /%A DXR: %06%A DYR: %06%A%N/
8168      .EVEN
8169 100254      BGNMSG      JXYPE
8170 100254      PRINTX      #JXYPX,@DXR,@DYR
8171 100304      ENDMSG
8172

```

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		APNT,1540,1500	
8216	100470	102000			MATCH:	CHAR!LO	: 'CHAR!ALL' IF MATCH INTERRUPT
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.,HAFY50-64.	
8281	101060	134001			BM04!M64	: A 64 SQUARE BIT MAP AT CENTER.
8282	101062	000000			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 ***
JCXD: CUIS,SYNC+2 ;SWITCH ON, MATCH OFF.
CURD,STOPI ;READ JSX/JSY, STOP, INTERRUPT.
DNOP,DNOP,DNOP ;*** FOR DEBUG ***
DJMP,BOXES ;LOOP

END.TEST

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

END TEST 33

```
*****
```

[illegible]

```

8347
8348
8349
8350
8351 101342
8352 101362
8353 101402 000410
8354 101404
8358 101424 005077 101404
8359
8360
8361
8362
8363 101430 012701 000050
8364 101434 004737 027454
8365 101440 103402
8366 101442 005301
8367 101444 001373
8368 101446
8369 101446 013700 101470
8370 101452 001743
8371 101454 005037 101470
8372 101460 006300
8373 101462 004770 101236
8374 101466 000746
8375
8376 101470 000000
8377
8378 101472 040 040 060
8379 101513 015 012
8380 101515 040 040 061
8381 101543 015 012
8382 101545 040 040 062
8383 101574 015 012
8384 101576 040 040 063
8385 101616 015 012
8386 101620 040 040 064
8387 101640 015 012
8388 101642 040 040 065
8389 101671 015 012
8390 101673 040 040 066
8391 101722 015 012
8392 101724 040 040 067
8393 101750 015 012
8394 101752 040 040 070
8395 101767 015 012
8396 101771 040 040 071
8397 102017 015 012
8398 102021 040 061 060
8399 102103 015 012
8400 102105 040 061 061
8401 102131 015 012
8402 102133 040 061 062
8403 102160 015 012
8404 102162 040 061 063
8405 102210 015 012
8406 102212 015 012

; IDLE LOOP. WAIT FOR KBD ENTRY TO CHANGE MODES.
IDLEX: PRINTF #SCOUT
IDLE: GMANID SDLIST,DISSEL,D,-1,0,DTHIL,YES
      BR IDLEB
IDLEA: GMANID SDASK,DISSEL,D,-1,0,DTHIL,YES
IDLEB: CLR @DSR ;STOP THE DPU (IE., FOR BLOOM)
;***B JSR PC,WAITF
;***B JSR PC,WAITF
;***B JSR PC,WAITF
;***B JSR PC,WAITF
      MOV #40,R1 ;WAIT FOR STOP
      JSR PC,WAITF ;SET UP A TIMEOUT COUNTER. ;***B
1$: JSR PC,WAITF ;WAIT FOR THE STOP BIT. ;***B
    BCS 2$ ;BR IF STOPPED. ;***B
    DEC R1 ;NOT STOPPED. BUMP COUNTER. ;***B
    BNE 1$ ;LOOP IF NOT STOPPED. ;***B
2$: MOV DISSEL,R0
    BEQ IDLE
    CLR DISSEL
    ASL R0
    JSR PC,@DTAB(R0)
    BR IDLEA

DISSEL: 0 ; DISPLAY CURRENTLY SELECTED.

SDLIST: .ASCII / 0 = 'TYPE THIS'/
        .BYTE 15,12
061 .ASCII / 1 = TURN BLINKING ON/
        .BYTE 15,12
062 .ASCII / 2 = TURN BLINKING OFF/
        .BYTE 15,12
063 .ASCII / 3 = COLOR BARS/
        .BYTE 15,12
064 .ASCII / 4 = 7 X 7 DOTS/
        .BYTE 15,12
065 .ASCII / 5 = 7 X 7 CROSS-HATCH/
        .BYTE 15,12
066 .ASCII / 6 = PERIMETER OUTLINE/
        .BYTE 15,12
067 .ASCII / 7 = BASIC COLOR ID/
        .BYTE 15,12
070 .ASCII / 8 = GUNS ID/
        .BYTE 15,12
071 .ASCII / 9 = ALL WHITE SCREEN/
        .BYTE 15,12
060 .ASCII / 10 = ALTERNATING WHITE SCREEN & PERIMETER OUTLINE/
        .BYTE 15,12
061 .ASCII / 11 = ALL RED SCREEN/
        .BYTE 15,12
062 .ASCII / 12 = ALL BLUE SCREEN/
        .BYTE 15,12
063 .ASCII / 13 = ALL GREEN SCREEN/
        .BYTE 15,12
SDASK: .BYTE 15,12

```

VSV11/VS11 -- GRAPHIC SYS. DIAG MACRO M1113 28-OCT-81 11:23 PAGE 114-2
*** TEST 34 SELECTED DISPLAYS (STAND-ALONE)

SEQ 0220

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


```

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 ; RE URN 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
8522 102670 130000      RPNT
8526 102700 173000      STOPN
  
```

; *** NOP TO ENLARGE THE DOTS ***

```

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 103016 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$      ;LOOP FOR 7 VERT VECTORS.
8555 103034 012737 000001 105070  4$:  MOV    #1,OUTFLG      ; SAY WE WANT DOTS ON THE OUTLINE.
8556 103042 004737 104762      JSR    PC,OUTLIN      ; OUTLINE THE SCREEN.
8557 103046 000207      RTS    PC      ; RETURN TO IDLE LOOP.
8558
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      .ASCIIZ /BLUE      RED      LSB-GRN      MSB-GRN /
8616
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 130 130 130 24$: .ASCIZ /XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX/
8690 103770 173000 25$: STOPN
8691 103772 160000 103706 DJMP,21$
8692
8693
8694
8695 ;
8696 ; MESSAGES FOR DISPLAY.
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      ;
8717      ; DISPLAYS FOR FILLING SCREEN WITH SOLID COLOR
8718      ;
8719      ;
8720      ;
8721      ;
8722      ;
8723      ;
8724      ;
8725      ;
8726      ;
8727      ;
8728      ;
8729      ;
8730      ;
8731      ;
8732      ;
8733      ;
8734      ;
8735      ;
8736      ;
8737      ;
8738      ;
8739      ;

104526 012746 003774
104532 000410
104534 012746 002200
104540 000405
104542 012746 002100
104546 000402
104550 012746 003400
104554 004737 026650
104560 004737 027066
104564 012737 114000 104612
104572 052637 104612
104576 012777 104612 076226
104604 004737 027454
104610 000207
104612 117774
104614 170100
104616 173000

SCRWHT: MOV #ALL,-(SP)
        BR   SCRCOM      ; GO TO COMMON SECTION

SCRRED: MOV #LO!200,-(SP) ; GET RED ONTO STACK,
        BR   SCRCOM      ; GO TO COMMON

SCRBLU: MOV #LO!100,-(SP) ; GET BLUE ONTO STACK,
        BR   SCRCOM      ; GO TO COMMON

SCRGRN: MOV #LO!1400,-(SP) ; GET FULL GREEN ONTO STACK,

SCRCOM: JSR PC,CLRRW      ; CLEAR FOR READ/WRITE
        JSR PC,NTSC      ; SETUP COLOR TABLE (IF AVAILABLE)
        MOV #APNT,10$    ; INIT PIXEL DATA SELECTION
        BIS (SP)+,10$    ; INSERT THE COLOR
        MOV #10$,@DPC    ; START THE DPU
        JSR PC,WAITF     ; WAIT FOR DONE
        RTS PC          ; RETURN TO IDLE

10$:    APNT!ALL
        SETMEM           ; SET THE MEMORY
        STOPN
  
```

```

8741
8742      : 'BLOOM' TEST DISPLAY
8743      : THIS DISPLAY ALTERNATELY DISPLAYS A FULL WHITE SCREEN AND A PERIMETER OUTLINE.
8744      : THE DISPLAY CHANGES ONCE EVERY 2 SECONDS.
8745
8746 104620 004737 025402      BLOOM: JSR      PC,DPINIT      ;INIT THE DPU
8747 104624 004737 102320      JSR      PC,TBOFF      ;TURN BLINKING OFF
8748 104630 004737 027066      JSR      PC,NTSC      ;SET UP THE COLOR TABLE (IF AVAILABLE)
8749 104634 013737 003250 104730      MOV      YMAX,12$      ;SET UP THE MAX Y VALUES, BASED ON FREQUENCY.
8750 104642 013737 003250 104740      MOV      YMAX,13$
8751 104650 052737 020000 104740      BIS      #MAXY,13$
8752 104656 012777 104666 076146      MOV      #10$,ADPC      ;START THE DISPLAY & LET IT RUN.
8753 104664 000207      RTS      PC      ;RETURN TO THE IDLE LOOP
8754
8755
8756      : DISPLAY FILE:
8757
8758 104666 176034      10$: WRT!CH0      ;SET ALL CHANNELS TO WRITE-ONLY, SWITCH-ENABLED.
8759 104670 176434      WRT!CH1
8760 104672 177034      WRT!CH2
8761 104674 177434      WRT!CH3
8762
8763 104676 117774 000000 000000      :LOOP TO HERE:
8764 104704 170100      11$: APNT!ALL,0,0      ;SET WHITE PIXEL DATA
8765 104706 170200      SETMEM      ;SET MEMORIES TO ALL WHITE
8766 104710 164170      SWCH      ;SWITCH TO READ-ONLY TO DISPLAY WHITE SCREEN
8767 104712 170200      DNOP+120.      ;SHOW IT FOR 2 SECONDS.
8768 104714 170140      SWCH      ;SWITCH TO WRITE-ONLY MODE.
8769 104716 117774 000000 000000      CLRMEM      ;CLEAR MEMORIES.
8770 104724 110000      APNT!ALL,0,0      ;START PERIMETER OUTLINE AT <0,0>
8771 104726 040000      LVEC      ;ENTER LONG-VECTOR MODE.
8772 104730 001776      I!0      ;DRAW LEFT EDGE
8773 104732 041776      12$: MAXY      ;... TO TOP OF VISIBLE AREA.
8774 104734 000000      I!MAXX      ;DRAW TOP EDGE
8775 104736 040000      0      ;... TO RIGHT SIDE.
8776 104740 021776      I!0      ;DRAW RIGHT EDGE ...
8777 104742 061776      13$: MAXY!MAXY      ;... TO BOTTOM.
8778 104744 000000      I!MAXY!MAXX      ;DRAW BOTTOM EDGE ...
8779 104746 170200      0      ;... BACK TO <0,0>
8780 104750 164170      SWCH      ;SWITCH TO READ-ONLY TO DISPLAY PERIMETER
8781 104752 170200      DNOP+120.      ;SHOW IT FOR 2 SECONDS.
8782 104754 160000 104676      SWCH      ;SWITCH BACK TO WRITE-ONLY.
8783      DJMP,11$      ;LOOP BACK TO DO IT ALL OVER AGAIN.
8784

```

```

:*****
:
:      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
      105012 004737 027454
8797 105016 005737 105070
8798 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
      105126 004737 027454
8828 105132 000207
8829
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
        DJMS,OUTDFX    ; ...AND AT 1000,0
        DJMS,OUTDFX    ; ... AND AT 1400,0.
        APNT!LO
        0
OUTDT1: MAXY60 ; GETS YMAX
        DJMS, OUTDFX   ; DO DOT AT 400,YMAX
        DJMS, OUTDFX   ; ... AND AT 1000, YMAX
        DJMS, OUTDFX   ; ... AND AT 1400,YMAX
  
```

```

8841 105200 116000 000000 000000      APNT.L0,0,0      : 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.L0,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      STOPN
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!L0
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$,@DPC      : START THE DPU.
8876 105336 004737 027454      JSR      PC,WAITE      : WAIT FOR DISPLAY STOP.
8877 105342 000207      RTS      PC
8878
8879      : DISPLAY CODE.
8880 105344 175401      10$: 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,WAITE      : WAIT FOR DISPLAY STOP.
8894 105370 000207      RTS      PC

```

8894			:	
8895			:	DISPLAY CODE.
8896			:	
8897	105372	175400	10\$:	NBLINK
8898	105374	175500		NBLINK+100
8899	105376	175600		NBLINK+200
8900	105400	175700		NBLINK+300
8901	105402	173000		STOPN

B 3

B 3

B 3

B 3

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 CHAN 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	#SCSYN,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	11\$		
9012	106242	013704	003022	12\$:	MOV	MFGMO,R4		GET TABLE ENTRY FOR THIS MEM IN R4.
9013	106246				PRINTF	#SCMMMO,R4,R3		PRINT DATA FOR THIS MEM.
9014	106272	005705			TST	R5		NON-STANDARD?
9015	106274	001410			BEQ	13\$		
9016	106276				PRINTF	#SCNSTD		YES. TELL THE WORLD!
9017	106316	013702	003024	13\$:	MOV	MFGSO,R2		GET MFG MASTER SYNC CHAN 0 IN R2.
9018	106322				PRINTF	#SCMMSO,R2		PRINT DATA FOR THIS SYNC CHAN.
9019	106344	032702	020000		BIT	#BIT13,R2		INTERLACED?
9020	106350	001011			BNE	14\$		YES.
9021	106352				PRINTF	#SCNINT		NO.
9022	106372	000410			BR	15\$		
9023								
9024	106374			14\$:	PRINTF	#SCINT		
9025	106414	013701	002012	15\$:	MOV	L\$UNIT,R1		GET NUMBER OF UNITS
9026	106420	162701	000001		SUB	#1,R1		
9027	106424	006301			ASL	R1		MAKE A WORD OFFSET
9028	106426	005761	003302		TST	ERTABL(R1)		DID LAST UNIT RUN?
9029	106432	001002			BNE	16\$		BR IF YES
9030	106434				EXIT	TST		NO -- EXIT TO DO ANOTHER.
9031	106440			16\$:	PRINTF	#SCOUT		
9032	106460			17\$:	BREAK			WAIT FOR USER TO <^C>.
9033	106462	000776			BR	17\$		
9034								
9035	106464	045	116	045	SCMEM:	.ASCIZ	/XNZAMEMORY: %D1XA, MEMTAB VAL: %06XA, %D1XA BITS/	
9036	106552	045	101	054	SCNSTD:	.ASCIZ	/XA, (NON-STD)/	
9037	106571	045	116	045	SCSYC:	.ASCIZ	/XNZASYNC CHAN: %01XA, SYCTAB VAL: %06/	
9038	106640	045	101	054	SCINT:	.ASCIZ	/XA, INTERLACED/	
9039	106660	045	101	054	SCNINT:	.ASCIZ	/XA, NON-INTERLACED/	
9040	106704	045	116	045	SCMMMO:	.ASCIZ	/XNZAMFG MASTER MEM 0, MEMTAB VAL: %06XA, %D1XA BITS/	
9041	106772	045	116	045	SCMMSO:	.ASCIZ	/XNZAMFG SYNC CHAN 0, SYCTAB VAL: %06/	
9042	107040	045	116	045	SCOUT:	.ASCIZ	/XNZNZA (<^C> TO GET BACK TO 'DR>')XN/	
9043						.EVEN		
9044								
9045	107110							END.TEST

```

*****
:
:
:   END TEST 35
:
:
*****

```

9047	.SBTTL
9048	.SBTTL SUPERVISORS DISPATCH TABLE
9049	
9050	:++
9051	: THIS TABLE HOLDS THE STARTING ADDRESS OF EACH TEST
9052	: FOR THE SUPERVISOR'S DISPATCHER.
9053	:--
9054	

```

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

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

BLU= 17*BIT8
 RED= 17*BIT0
 GRN= 17*BIT4

NTSC8: 0 ; BLACK (NO COLOR AT ALL).

BLU
 RED
 RED+BLU ; MAGENTA
 GRN
 GRN+BLU ; CYAN
 GRN+RED ; YELLOW
 GRN+RED+BLU ; WHITE

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

EXPRM: 0
 43 ; BROWN (3RD LEVEL YELLOW).
 0
 RED ; RED
 0
 117 ; ORANGE (RED + HI GREEN)
 0
 RED+GRN ; YELLOW
 0
 GRN ; GREEN
 0
 BLU ; BLUE
 0
 RED+BLU ; VIOLET (MAGENTA)
 0
 BLU+RED+GRN ; WHITE.

9145	110242	000000
9146	110244	000043
9147	110246	000000
9148	110250	000017
9149	110252	000000
9150	110254	000117
9151	110256	000000
9152	110260	000377
9153	110262	000000
9154	110264	000360
9155	110266	000000
9156	110270	007400
9157	110272	000000
9158	110274	007417
9159	110276	000000
9160	110300	007777

9162			.SBTTL *** ASCII CHARACTER ADDRESS TABLE AND SUBPIX	
9163			:THE FOLLOWING 128. WORDS CONTAIN THE CHARACTER SUBPIX	
9164			:STARTING ADDRESSES. 0-37 AND 140-177 ARE NULL FOR NOW.	
9165			: (EXCEPT FOR 12 AND 15)	
9166				
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      ;...OK, EXECUTE THE <CRLF>
9277 110722 000000 000000      ..CRX: .WORD 0,0
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 DJMP,...CC ;FINISH USING 'C'.
9648	112272	104000	..HH:	SVEC
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
9727	112532	104000		..RR:		DJMP,...OO	SVEC
9731	112542	160000	112474				.WORD
9732	112546	104000		..SS:		DJMP,...PP	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		..BSLM:			SVEC

;FINISH USING 'O'.

;FINISH USING 'P'.

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      .SBTTL *** DEVELOPMENT VERSIONS CONTROL
9842
9843
9844      ; THE FOLLOWING NUMBERS GET CHANGED AT EACH NEW ASSEMBLY, THEREFORE:
9845      ; IF THESE NUMBERS DON'T MATCH WHAT'S IN CORE (OCTALLY) THEN THIS
9846      ; LISTING DOESN'T MATCH THE LOADED PROGRAM!
9847
9848 113074 021557      DDMY: 09071.
9849 113076 002165      HMM: 1141.
9850
9851
9852
9853      .SBTTL *** PATCH AREA
9854      .SBTTL
9855
9856      ;
9857      ; FINALLY A GENEROUS PATCH AREA.
9858      ;
9859      ; AND AN ADJUSTMENT TO ACCOUNT FOR THE 'LASIAD BIT7' HACK
9860      ; DESCRIBED IN 'SUPPRG.MEM' (FOR REV C).
9861      ;
9862
9863 113100      PATCH: .BLKW 16.
9864
9865      : .BLKB <<.400>8^C377>-.
9866      :
9867      : =.377+1
9868      113400
9869
9870 113404      LAST::
9871
  
```

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	

ABP	055140	BM1	067532	CMIVB	015430	CSMT1	060314	C\$EDIT=	000003
ABPX	055146	BM1PIX	067550	CMNOMI	015371	CSMT10	062244	C\$ERDF=	000055
ABPY	055150	BM1TST	067214	CMPE	016224	CSMT11	062364	C\$ERHR=	000056
ACAT	110302	BM1X	067542	CNVRT	027324	CSMT12	062450	C\$ERRO=	000060
ADR	= 000020 G	BM1Y	067544	COLID	103106	CSMT13	062656	C\$ERSF=	000054
ALL	= 003774 G	BM14	= 136000 G	CONFIG	030414	CSMT14	063020	C\$ERSO=	000057
ALLB	= 003774 G	BM18	= 137000 G	CONMEM	030414	CSMT15	063306	C\$ESCA=	000010
ALLNB	= 003374 G	BOE	= 000400 G	CONSYC	030760	CSMT2	060426	C\$ESEG=	000005
APF	005047	BOXES	100732	CRDAX	013734	CSMT3	060540	C\$ESUB=	000003
APNT	= 114000 G	BRINIT	007030	CRDAY	014017	CSMT4	060672	C\$ETST=	000001
APTST	054566	BV	102466	CRWRE	014102	CSMT5	061024	C\$EXIT=	000032
ASSEMB=	000010	B04	075714	CRWRO	013664	CSMT6	061202	C\$GETB=	000026
AUXSEG=	000001 G	B048	075736	CRWRZ	013613	CSMT7	061320	C\$GETW=	000027
A1	= 001776	B08	075726	CSCHAN	065140	CSMT8	061434	C\$GMAN=	000043
BADDAT	023060	B14	075666	CSCH.1	065146	CSMT9	062070	C\$GPHR=	000042
BARS	102356	B148	075710	CSCH.6	065142	CSMXIT	065016	C\$GPLO=	000030
BGEN	102460	B18	075700	CSCH.8	065144	CSPCE	010411	C\$GPRI=	000040
BIT0	= 000001 G	CBAER	021422 G	CSDSE	010351	CSRCHK	022440	C\$INIT=	000011
BIT00	= 000001 G	CBAERA	013507	CSFLE	010451	CSRER	021166 G	C\$INLP=	000020
BIT01	= 000002 G	CBCHK	022624	CSIFC	014630	CSRERA	013165	C\$MANI=	000050
BIT02	= 000004 G	CESWE	014326	CSMEND	064776	CSRETR	065150	C\$MEM =	000031
BIT03	= 000010 G	CEZWE	014255	CSMF1A	065200	CSRGEN	014543	C\$MSG =	000023
BIT04	= 000020 G	CGRPH	= 020000 G	CSMF1B	065206	CSRGSN	014461	C\$OPEN=	000034
BIT05	= 000040 G	CHAR	= 100000 G	CSMF2A	065212	CSXPE	010651	C\$PNTB=	000014
BIT06	= 000100 G	CHDISP	054122	CSMF2B	065214	CSXRE	010611	C\$PNTF=	000017
BIT07	= 000200 G	CHIE	= 010000 G	CSMF2C	065216	CT	054070	C\$PNTS=	000016
BIT08	= 000400 G	CHPROT=	000200 G	CSMF2D	065222	CTAB	003256 G	C\$PNTX=	000015
BIT09	= 001000 G	CHRF	005475	CSMF3A	065230	CTABE	003302 G	C\$QIO =	000377
BIT1	= 000002 G	CHRFX	005521	CSMF3B	065236	CTABM	003256 G	C\$RDBU=	000007
BIT10	= 002000 G	CHRTST	053374	CSMF3C	065240	CTABS	003270 G	C\$REFG=	000047
BIT11	= 004000 G	CHSEG	054066	CSMF3D	065252	CT.ASC	054104	C\$RESE=	000033
BIT12	= 010000 G	CHV	103060	CSMF4A	065300	CT.BAS	054116	C\$REVI=	000003
BIT13	= 020000 G	CHX	103052	CSMF4B	065320	CT.RET	054106	C\$RFLA=	000021
BIT14	= 040000 G	CHY	103054	CSMF4C	065314	CT.SUB	054110	C\$RPT =	000025
BIT15	= 100000 G	CH0	= 000000 G	CSMF4D	065334	CUIM	= 146016 G	C\$SEFG=	000046
BIT2	= 000004 G	CH1	= 000400 G	CSMF4E	065372	CUIOFF=	146012 G	C\$SPRI=	000041
BIT3	= 000010 G	CH2	= 001000 G	CSMF4F	065420	CUIS	= 146013 G	C\$SVEC=	000037
BIT4	= 000020 G	CH3	= 001400 G	CSMF4G	065436	CUOFF =	146040 G	C\$TPRI=	000013
BIT5	= 000040 G	CKDROP	030354	CSMF4H	065454	CUON	= 146060 G	DAVTO =	144003 G
BIT6	= 000100 G	CKEMAX	030254	CSMF4I	065470	CURD	= 146100 G	DBLINK	105356
BIT7	= 000200 G	CLFLGS=	000012 G	CSMF4J	065346	CUWT	= 175000 G	DBLPX=	000100 G
BIT8	= 000400 G	CLMFCN	026754	CSMF4K	065370	CVC	= 000040 G	DDMMY	113074
BIT9	= 001000 G	CLMPTR	071610	CSMF4L	065324	C\$AU	= 000052	DEVcnt	032726
BLINK	= 175401 G	CLMSHT	071626	CSMF4U	065270	C\$AUTO=	000061	DEVDR0	034004
BLOOM	104620	CLMTAB	071612	CSMF4V	065254	C\$BRK =	000022	DEVNRD	033723
BLU	= 007400	CLRMEM=	170140 G	CSMF4X	065326	C\$BSEG=	000004	DEVNRR	033641
BLUC	= 000004 G	CLRRW	026650	CSMF4Y	065330	C\$BSUB=	000002	DEVONL	033571
BLUEG	= 002100 G	CLRRW1	026640	CSMITR	060204	C\$CEFG=	000045	DEVSUM	033534
BMF	005431	CLRTST	070716	CSMLOO	060210	C\$CLCK=	000062	DFPTBL	002224 G
BMX	005453	CLRT1	070740	CSMNEX	064756	C\$CLEA=	000012	DGEN	102660
BMXX	101062	CLRW	026632	CSMPE	010551	C\$CLOS=	000035	DGX	102662
BMOCHK	070456	CLR.D	071602	CSMPTB	065026	C\$CLP1=	000006	DGY	102664
BMOP1	070552	CLR.D2	071720	CSMPTC	065136	C\$CVEC=	000036	DIAGMC=	000000
BMOP2	070576	CLR.2	071562	CSMPX	065022	C\$DCLN=	000044	DISSEL	101470
BMOTST	067642	CLR.2A	071572	CSMPY	065024	C\$DODU=	000051	DJM	026274
BM04	= 134000 G	CLR.22	071704	CSMRE	010511	C\$DRPT=	000024	DJMP	= 160000 G
BM08	= 135000 G	CLTNEW	071272	CSMTST	060164	C\$DU	= 000053	DJMS	= 160001 G

SYMBOL TABLE

DJS	026326	ECMPE4	052452	FCOORD	016377	GHYTST	066450	HUE	003222	G
DNOP =	164000	ECRSV1	052466	FERCM	024030	GHYTYP	067104	HX	075516	
DOTS	102556	ECRSV2	052472	FHST =	060000	G	GOLD =	003200	G	
DPC	003032	ECRSV3	052502	FILL1	027014		GRN =	000360		
DPCER	020732	ECSEQ1	052512	FILL2	027020		GRNC =	000010	G	
DPCERA	012643	ECSEQ2	052530	FLGCHK	022410		GRN1 =	002400	G	
DPDSE	007611	ECSEQ3	052576	FLGER	021134	G	GRN2 =	003000	G	
DPINIT	025402	ECSEQ4	052630	FLGERA	013122		GRN3 =	003400	G	
DPOP =	165000	ECSEQ5	052654	FNOCSR	023772		GRX	075446		
DPRESE	026074	ECSEQ6	052676	FNOINT	004315		GRY	075236		
DPRI	003050	ECSEQ7	052726	FORCER	003020		GTCURS	077752		
DPSINI	025540	ECSQ7B	052750	FORCET	032776		GTJSSW=	040000	G	
DPUMOD	002502	ECSTO	052754	FORCSI=	000040	G	GUNID	103360		
DPUSAV	026360	EF.CON=	000036	FREE	003240	G	GX	075470		
DRDYTO=	150003	EF.NEW=	000035	FRESIZ	003242	G	GXI =	174100	G	
DRR	003034	EF.PWR=	000034	FSYCHA	016500		GY	075260		
DSBINT	026176	EF.RES=	000037	FUMI	004206		GYI =	174100	G	
DSCSE	010111	EF.STA=	000040	FUSI	003752		G\$CNTO=	000200		
DSDPE	007651	EGGBL =	003500	FUSWI	004251		G\$DELM=	000372		
DSDXE	007711	EMAXDU	030151	FUTO	004066		G\$DISP=	000003		
DSDYE	007751	EMK	100360	FSAU =	000015		G\$EXCP=	000400		
DSFLE	010051	EN =	006654	FSAUTO=	000020		G\$HILI=	000002		
DSMPE	010211	ENAINI	026114	F\$BGM =	000040		G\$LOLI=	000001		
DSMRE	010151	ENECHK=	000100	F\$CLEA=	000007		G\$NO =	000000		
DSPCE	010011	ENVIRN	031476	F\$DU =	000016		G\$OFFS=	000400		
DSR	003034	ERCM	024041	F\$END =	000041		G\$OF SI=	000376		
DSRCHK	022332	ERCNAM	024264	F\$HARD=	000004		G\$PRMA=	000001		
DSRER	020764	ERCOD	024074	F\$HW =	000013		G\$PRMD=	000002		
DSRERA	012706	ERCODS	047466	F\$INIT=	000006		G\$PRML=	000000		
DSTP	026242	ERCTBL	024266	F\$JMP =	000050		G\$RADA=	000140		
DSXPE	010311	ERICHK	023516	F\$MOD =	000000		G\$RADB=	000000		
DSXRE	010251	ERR =	100000	F\$MSG =	000011		G\$RADD=	000040		
DTAB	101236	ERRCHK	023524	F\$PROT=	000021		G\$RADL=	000120		
DTABE	101272	ERRCK1	023526	F\$PWR =	000017		G\$RADO=	000020		
DTHIL =	000015	ERRK	030130	F\$RPT =	000012		G\$XFER=	000004		
DTO	026210	ERRVEC=	000004	F\$SEG =	000003		G\$YFS =	000010		
DUADR	035704	ERTABE	003502	F\$SOFT=	000005		HAFX =	000776	G	
DUADRE	021454	ERTABL	003302	F\$SRV =	000010		HAFY =	000776	G	
DUAD12	017105	ESF	004612	F\$SUB =	000002		HAFY50=	000776	G	
DUFLG	003224	ESUM	030132	F\$SW =	000014		HAFY60=	000736	G	
DUMMY	003076	EVL =	000004	F\$TEST=	000001		HATCH	102702		
DX =	000002	EX	100336	GCOFF =	000100	G	HBAER	021370	G	
DXR	003036	EXPGOT	017051	GDBAD	022736		HBAERA	013444		
DXRER	021016	EXPGT2	016322	GDDAT	023056	G	HBCHK	022526		
DXRERA	012751	EXPR	027100	GERRMA	002516	G	HCOPY =	176051	G	
DXYME	100140	EXPRC2	022074	GETCUR	065474		HCPY =	000001	G	
DYI =	000002	EXPREC	020700	GETJSE	065542		HGEN	103050		
DYNI =	000004	EXPRM	110242	GHIINH=	010000	G	HMM	113076		
DYR	003040	EXSM2 =	000050	GHX =	120000	G	HOE =	100000	G	
DYRER	021050	EXSM4 =	000060	GHXF	005314		HPM1	002310		
DYRERA	013014	EXTA	022300	GHXT	066334		HPM2	002340		
EBLINK	105330	EXTEND	022276	GHXTST	065676		HPM3	002370		
ECE	013552	EX2 =	000010	GHXTYP	066332		HPM4	002420		
ECISVC	052416	EX4 =	000020	GHXYX	005374		HPM5	002450		
ECMPE1	052420	EY	100352	GHY =	124000	G	HST =	040000	G	
ECMPE2	052424	F\$END =	002100	GHYF	005344		HSX	075502		
ECMPE3	052436	F\$LOAD=	000035	GHYT	067106		HSY	075272		

IMRF	072514	JMSPOP	044222	LDCP	= 175000 G	L\$EXP1	002046 G	L10031	021452
IMRRE	007205	JMWD	= 000004 G	LDECC	= 175400 G	L\$EXP4	002064 G	L10032	021506
IMR8	= 163000	JMWE	= 000010 G	LDI	= 002000 G	L\$EXP5	002066 G	L10033	021646
IMTMOD	002504 G	JPDPC	007525	LDINC	006662	L\$HARD	002240 G	L10034	021760
INCCBA	044014	JPPCSE	007557	LDJSS	= 146000 G	L\$HIME	002120 G	L10035	022072
INCERK	030216	JSDXYE	100064	LDR	003044 G	L\$HPCP	002016 G	L10036	022130
INCHBA	043544	JSEWRO	014211	LDUN	026602	L\$HPTP	002022 G	L10037	022176
INCREG	041232	JSEWRZ	014145	LERRMA	002514 G	L\$HW	002224 G	L10040	022224
INLAC	072706	JSF	004566	LINTER	020610 G	L\$ICP	002104 G	L10041	022274
INLE	017207	JSLCKO	= 000010 G	LINTX	006540	L\$INIT	032014 G	L10042	023322
INLEX	022226 G	JSMIES	= 010000 G	LMR	003046 G	L\$LADP	002026 G	L10043	024262
INLEXF	017235	JSMV	003054 G	LOCT	074754	L\$LAST	113404 G	L10044	025740
INTCPC	026112	JSPIX	100316	LOE	= 040000 G	L\$LOAD	002100 G	L10046	032724
INTERR	020226 G	JSPIXA	100306	LOGO	074600	L\$LUN	002074 G	L10047	033106
INTFLA	026107	JSSIES	= 004000 G	LONGX	005121	L\$MREV	002050 G	L10050	033214
INTLAC	003230 G	JSSV	003060 G	LOOP	027646	L\$NAME	002000 G	L10051	033216
INTMAS	026106	JSTST	077304	LOOPFL	023054 G	L\$PRIO	002042 G	L10052	033270
INTVEC	026110	JSV	076670	LOOPK	027700	L\$PROT	032004 G	L10053	033532
INTX	004400	JSVA	076744	LOT	= 000010 G	L\$PRT	002112 G	L10054	034272
IOKCKI	= 000200	JSVA1	076724	LREAD	= 010000 G	L\$REPP	002062 G	L10055	035042
IOKERR	= 000004	JSWD	= 000001 G	LRWERR	020642 G	L\$REV	002010 G	L10056	035620
IOKJSM	= 000002	JSWE	= 000002 G	LRWX	006707	L\$RPT	033272 G	L10057	041150
IOKJSS	= 000010	JSWTEX	100010	LSET	= 010000 G	L\$SOFT	002522 G	L10060	042136
IOKSTP	= 000001	JSX	100414	LSR	003042 G	L\$SPC	002056 G	L10061	042564
IPCSAV	003146 G	JSY	100434	LUMIN	027054	L\$SPCP	002020 G	L10062	043446
IREGCK	023066	JSO	= 000000 G	LUMTBL	107222	L\$SPTP	002024 G	L10063	043716
ISDSR	003144 G	JS1	= 000400 G	LUTAV	003216 G	L\$STA	002030 G	L10064	044156
ISR	= 000100 G	JS2	= 001000 G	LUTSAV	026606	L\$SW	002502 G	L10065	044652
IXE	= 004000 G	JS3	= 001400 G	LUTV	003062 G	L\$TEST	002114 G	L10066	045366
IXPME	007355	JXYPE	100254 G	LV	055606	L\$TIML	002014 G	L10067	046036
IXPWE	007404	JXYPX	100216	LVEC	= 110000 G	L\$UNIT	002012 G	L10070	046766
IXRRE	007326	JSJMP	= 000167	LVF	005073	LO	= 002000 G	L10071	047334
ISAU	= 000041	KIPAR0	= 172340	LVTST	055224	L10000	002236	L10072	052774
ISAUTO	= 000041	KIPAR1	= 172342	LVX	055616	L10001	002500	L10073	053326
ISCLN	= 000041	KIPAR2	= 172344	LVY	055620	L10002	002520	L10074	054514
ISDU	= 000041	KIPAR3	= 172346	LSACP	002110 G	L10003	003020	L10075	055154
ISHRD	= 000041	KIPAR4	= 172350	LSAPT	002036 G	L10004	020062	L10076	055624
ISINIT	= 000041	KIPAR5	= 172352	LSAU	033012 G	L10005	020124	L10077	056314
ISMOD	= 000041	KIPAR6	= 172354	LSAUT	002070 G	L10006	020166	L10100	057014
ISMSG	= 000041	KIPAR7	= 172356	LSAUTO	033216 G	L10007	020224	L10101	060062
ISPROT	= 000040	KIPDR0	= 172300	LSCCP	002106 G	L10010	020504	L10102	065610
ISPTAB	= 000041	KIPDR1	= 172302	LSCLEA	033220 G	L10011	020606	L10103	066362
ISPR	= 000041	KIPDR2	= 172304	LSCO	002032 G	L10012	020640	L10104	067134
ISRPT	= 000041	KIPDR3	= 172306	L\$DEPO	002011 G	L10013	020676	L10105	067562
ISSEG	= 000041	KIPDR4	= 172310	L\$DESC	002122 G	L10014	020730	L10106	070622
ISSETU	= 000041	KIPDR5	= 172312	L\$DESP	002076 G	L10015	020762	L10107	072612
ISSFT	= 000041	KIPDR6	= 172314	L\$DEVP	002060 G	L10016	021014	L10110	071270
ISSRV	= 000041	KIPDR7	= 172316	L\$DISP	107114 G	L10017	021046	L10111	071550
ISSUB	= 000041	KLUDGE	031572	L\$DLY	002116 G	L10020	021100	L10112	072502
ISTST	= 000041	KTBKGD	031322	L\$DTP	002040 G	L10021	021132	L10113	073146
JCWD	= 000020 G	KTFLG	003244 G	L\$DTYP	002034 G	L10022	021164	L10114	074362
JCWE	= 000040 G	KTINIT	031656	L\$DU	033110 G	L10023	021216	L10115	073660
JCXA	100320	KTOFF	031306	L\$DUT	002072 G	L10024	021250	L10116	074236
JCXB	100740	KTON	031270	L\$DVTY	002162 G	L10025	021302	L10117	074444
JCXC	101110	K100US	027564	L\$EF	002052 G	L10026	021334	L10120	076556
JCXD	101114	LAIER	006555	L\$ENVI	002044 G	L10027	021366	L10121	101132
JENOCL	014376	LAIEW	006617	L\$ETP	002102 G	L10030	021420	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	
MATXE	015547	MAXY =	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	NOCSE	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	PDJ	003070	G	RLMOD	057062	SELCBA=	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	NOSWI	014712	PINTRA=	000001	G	RLXIE	016552	SELFLG=	000002	G
MMMACA	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
MMNEXE	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	NTSC8	110222	PRBH	005574		RLY2UE	016751	SELXRR=	000006	G
MPAT2	074306	NUL	016546	PRBHG	005556		RND8 =	000400	SEQERR=	120003	G
MPCBAS	074332	NULCR	016547	PRGS17=	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	NXR	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

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	XRMKE 012011
SIMSG 023012	SYCFLG 003254 G	TSSHW = 010000	USWI 004255	XRPCE 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
SIOFF = 171000 G	SYNCTO= 140003 G	TSSPC = 000002	VIOLET= 003300 G	XXRPE 012111
SION = 171400 G	SYSCON 021510 G	TSSPRO= 010045	VISXIT 076356	XTBL 102516
SKP0 = 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	VIS5A 075746	XYX 003657
SOCT2 075040	TIDFLG 002506 G	TSSTES= 010124	VIS6 076076	XSALWA= 000000
SPARTS 074434	TINERR 022775	T1 034054 G	VRLF 076016	XSALS= 000040
SPCSAV 003176 G	TN = 000043	T10 044160 G	VRLF1 076024	X\$OFFS= 000400
SPM1 002600	TNAM 030066	T11 044654 G	VRLF2 076026	X\$TRUE= 000020
SPM2 002630	TNUM 030064	T12 045370 G	VSHGE 026010	X2INC 057752
SPM4 002660	TOTV 003056 G	T13 046040 G	VSHNG 025720 G	X2INCA 057762
SPM5 002710	TOTVST 047050	T14 046770 G	VSHUNG 025742	X2INCB 057772
SPM6 002740	TSTEND 030072	T15 047422 G	WAITF 027454	YA 067124
SPM7 002770	TSTGO 027702	T16 052776 G	WE = 000010 G	YDN 060016
SR0 = 177572	T\$ARGC= 000001	T17 053330 G	WECC = 160000 G	YDOWN = 000040 G
SR1 = 177574	T\$CODE= 000052	T18 054516 G	WHITE = 003774 G	YELLOW= 003600 G
SR2 = 177576	T\$ERRN= 006346	T19 055156 G	WJSS = 042000 G	YI 066340
SR3 = 172516	T\$EXCP= 000000	T2 034274 G	WRT = 176034 G	YMAX 003250 G
SSDSR 003174 G	T\$FLAG= 000040	T20 055626 G	WRTCPX= 140000 G	YUP 060002
SSF 004503	T\$FREE= 113440	T21 056316 G	WRTCPY= 140000 G	Y2DN 060046
SSTST 042210	T\$GMAN= 000000	T22 057016 G	WRTCSR= 000013 G	Y2UP 060032
SST1 042516	T\$HILI= 000015	T23 060064 G	WRTJSS= 042000 G	Y50TBL 102516
SST2 042526	T\$LAST= 000001	T24 065612 G	WRTMSR= 040000 G	Y60TBL 102536
SST3 042550	T\$LOLI= 000000	T25 066364 G	WRT.D 074262	\$SVPC = 110402
STOP = 172000 G	T\$LSYM= 010000	T26 067136 G	WRT.X 074264	.DX = 002000
STOP1 = 173400 G	T\$LTNO= 000043	T27 067564 G	WRT.Y 074266	.DY = 000000
STOPN = 173000 G	T\$NEST= 177777	T28 070624 G	WRT1 = 176036 G	.I = 000000
STPTST 046110	T\$NS0 = 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 G	XINC 057722	..AND 111112
STP2 046746	T\$PCNT= 000000	T3 035044 G	XINCA 057732	..AST 111220
STP3 046752	T\$PTAB= 010130	T30 073150 G	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	..BSLM 113014
SUPVSR= 026050	T\$SAVL= 177777	T31 074364 G	XMMACA 045530	..CC 112160
SVCGBL= 000000	T\$SEGL= 177777	T32 074446 G	XMMACE 012545	..CMA 111260
SVCINS= 177777	T\$SEKO= 010000	T33 076560 G	XMMAE 021762 G	..COLN 111716
SVCSUB= 177777	T\$SIZE= 000016	T34 101134 G	XPCSE 012311	..CR 110714
SVCTAG= 177777	T\$SUBN= 000000	T35 105404 G	XPDSSE 012151	..CRT 113050
SVCTST= 177777	T\$TAGL= 177777	T4 035622 G	XPFLF 012251	..CRX 110722
SVD 057010	T\$TAGN= 010132	T5 041152 G	XPMER 021336 G	..DD 112206
SVEC = 104000 G	T\$TEMP= 000044	T6 042140 G	XPMERA 013401	..DOL 111020
SVF 005221	T\$TEST= 000043	T7 042566 G	XPMPE 012411	..DOT 111304
SVT 057000	T\$TSTM= 177777	T8 043450 G	XPMRE 012351	..EE 112230
SVTST 056364	T\$TSTS= 000001	T9 043720 G	XPPCE 012211	..EQ 111766

..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