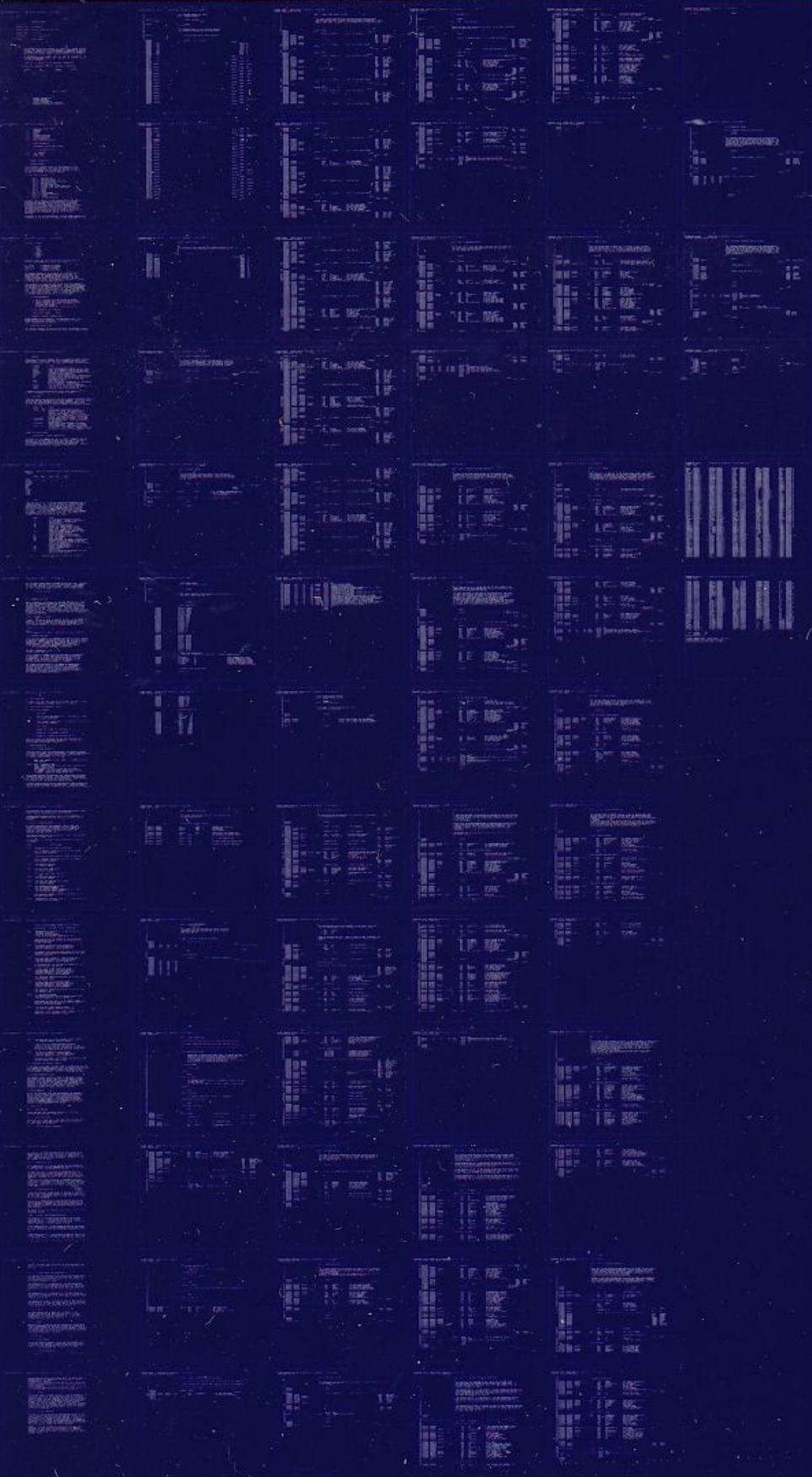


VTV31-K

VTV31-K VID INTF DIAG
CZVTVAO

AH-T061A-MC
FICHE 1 OF 1

APR 1982
COPYRIGHT © 1982
MADE IN USA



.REM 8

IDENTIFICATION

PRODUCT CODE: AC-T059A-MC
 PRODUCT NAME: CZVTVA0 VTV31K VID INTF DIAG
 PRODUCT DATE: 19-OCT-81
 MAINTAINER: C.S.S., READING, U.K.
 AUTHOR: DAVE HUNTER

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) 1982 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL
DEC

PDP
DECUS

UNIBUS
DECTAPE

MASSBUS

TABLE OF CONTENTS

1.0	GENERAL INFORMATION
1.1	PROGRAM ABSTRACT
1.2	SYSTEM REQUIREMENTS
1.3	RELATED DOCUMENTS AND STANDARDS
1.4	DIAGNOSTIC HIERARCHY PREREQUISITES

- 2.0 OPERATING INSTRUCTIONS
 - 2.1 COMMANDS
 - 2.2 SWITCHES
 - 2.3 FLAGS
 - 2.4 HARDWARE QUESTIONS
 - 2.5 SOFTWARE QUESTIONS
 - 2.6 CLOCK QUESTIONS
 - 2.7 QUICK STARTUP PROCEDURE
- 3.0 ERROR INFORMATION
- 4.0 PERFORMANCE AND PROGRESS REPORTS
- 5.0 DEVICE INFORMATION TABLES
- 6.0 TEST SUMMARIES
 - 6.1 LOGIC TESTS
 - 6.2 VISUAL TESTS

1.0 GENERAL INFORMATION

1.1 PROGRAM ABSTRACT

THIS DIAGNOSTIC PROGRAM CHECKS THE FUNCTIONALITY OF THE VTV31-K VIDEO INTERFACE. TEN TESTS ARE PROVIDED TO VERIFY THE VTV31-K HARDWARE, AND IN ADDITION, ONE TEST OUTPUTS PATTERNS FOR SETTING UP A DISPLAY MONITOR.

THE TESTS ARE AS FOLLOWS:

- TEST 1 - REGISTER NXM
- TEST 2 - REGISTER ACCESS
- TEST 3 - REGISTER ADDRESS UNIQUENESS
- TEST 4 - REGISTER BIT SET
- TEST 5 - PICTURE STORE DATA
- TEST 6 - PICTURE STORE COLUMN ADDRESSING
- TEST 7 - PICTURE STORE LINE ADDRESSING
- TEST 8 - 8-DOT
- TEST 9 - PRESET
- TEST 10 - VIDEO ENABLE
- TEST 11 - DISPLAY ADDRESSING
- TEST 12 - OFFSET
- TEST 13 - PATTERN GENERATOR

IN TESTS 1 TO 9, SPECIFIC VALUES ARE WRITTEN TO THE DEVICE REGISTERS AND THE CSR READ TO CONFIRM THAT THE REQUIRED OPERATION IS PERFORMED. TESTS 10 TO 13 ARE VISUAL ONLY. ALTHOUGH THEY WILL RUN WITHOUT A DISPLAY MONITOR, NO CHECKS ARE MADE BY THE PROGRAM. OBSERVATION BY THE OPERATOR IS THEREFORE REQUIRED TO FULLY CHECK THE VTV31K. IN TEST 13, THE OPERATOR CAN CONTROL THE RATE OF CHANGE OF THE DISPLAYED PATTERNS BY SELECTING MANUAL CONTROL IN THE STARTUP QUESTIONS.

IN SEVERAL OF THE TEST DESCRIPTIONS, PICTURE STORE CONTENTS ARE REFERRED TO BY DATA VALUE. THESE DATA VALUES CORRESPOND TO THE

FOLLOWING COLOURS:-

0 = BLACK
1 = RED
2 = GREEN
3 = YELLOW
4 = BLUE
5 = MAGENTA
6 = CYAN
7 = WHITE

EXECUTION TIMES VARY WITH THE CPU TYPE. THE FOLLOWING ARE TYPICAL TIMES OBSERVED ON A PDP-11/23 SYSTEM:

ALL TESTS	16 MINUTES 25 SECONDS
TESTS 1-9	14 MINUTES 40 SECONDS
TESTS 10-13	1 MINUTE 45 SECONDS
TEST 5	10 MINUTES 30 SECONDS

THE PROGRAM SUPPORTS UP TO 20 UNITS, ALL SELECTED TESTS BEING RUN ON ONE UNIT BEFORE PROCEEDING TO THE NEXT. IF DURING ONE PASS, THE NUMBER OF ERRORS DETECTED ON A UNIT EXCEEDS A MAXIMUM VALUE (REQUESTED AT STARTUP), THE UNIT IS DROPPED FROM TESTING. UNIT DROPPING IS PREVENTED IF EITHER THE IDU OR LOE FLAGS HAVE BEEN SELECTED.

THIS DIAGNOSTIC HAS BEEN WRITTEN FOR USE WITH THE DIAGNOSTIC RUNTIME SERVICES SOFTWARE (SUPERVISOR). THESE SERVICES PROVIDE THE INTERFACE TO THE OPERATOR AND TO THE SOFTWARE ENVIRONMENT. THIS PROGRAM CAN BE USED WITH XXDP+, ACT, APT, SLIDE AND PAPER TAPE. FOR A COMPLETE DESCRIPTION OF THE RUNTIME SERVICES, REFER TO THE XXDP+ USER'S MANUAL. THERE IS A BRIEF DESCRIPTION OF THE RUNTIME SERVICES IN SECTION 2 OF THIS DOCUMENT.

1.2 SYSTEM REQUIREMENTS

- A. PDP-11 PROCESSOR WITH 16K OR MORE OF MEMORY.
- B. CONSOLE TERMINAL WITH INTERFACE ADDRESS 777560.
- C. XXDP+ SUPPORTED LOAD DEVICE (RX, RM, RK, TM ETC.) OR PAPER TAPE READER.
- D. COLOUR DISPLAY MONITORS (1 FOR EACH VTV31-K).

1.3 RELATED DOCUMENTS AND STANDARDS

XXDP+ USER'S MANUAL CHQUS

1.4 DIAGNOSTIC HIERARCY PREREQUISITES

BEFORE RUNNING THIS DIAGNOSTIC, THE APPROPRIATE PDP-11 CPU, MEMORY AND PERIPHERAL DIAGNOSTICS SHOULD BE RUN TO VERIFY CORRECT OPERATION OF THE SYSTEM.

2.0 OPERATING INSTRUCTIONS

THIS SECTION CONTAINS A BRIEF DESCRIPTION OF THE RUNTIME SERVICES. FOR DETAILED INFORMATION, REFER TO THE XXDP+ USER'S MANUAL (CHQUS).

2.1 COMMANDS

THERE ARE ELEVEN LEGAL COMMANDS FOR THE DIAGNOSTIC RUNTIME SERVICES (SUPERVISOR). THIS SECTION LISTS THE COMMANDS AND GIVES A VERY BRIEF DESCRIPTION OF THEM. THE XXDP+ USER'S MANUAL HAS MORE DETAILS.

COMMAND	EFFECT
START	START THE DIAGNOSTIC FROM AN INITIAL STATE
RESTART	START THE DIAGNOSTIC WITHOUT INITIALIZING
CONTINUE	CONTINUE AT TEST THAT WAS INTERRUPTED (AFTER ^C)
PROCEED	CONTINUE FROM AN ERROR HALT
EXIT	RETURN TO XXDP+ MONITOR (XXDP+ OPERATION ONLY!)
ADD	ACTIVATE A UNIT FOR TESTING (ALL UNITS ARE CONSIDERED TO BE ACTIVE AT START TIME)
DROP	DEACTIVATE A UNIT
PRINT	PRINT STATISTICAL INFORMATION (IF IMPLEMENTED BY THE DIAGNOSTIC - SECTION 4.0)
DISPLAY	TYPE A LIST OF ALL DEVICE INFORMATION
FLAGS	TYPE THE STATE OF ALL FLAGS (SEE SECTION 2.3)
ZFLAGS	CLEAR ALL FLAGS (SEE SECTION 2.3)

A COMMAND CAN BE RECOGNIZED BY THE FIRST THREE CHARACTERS. SO YOU MAY, FOR EXAMPLE, TYPE "STA" INSTEAD OF "START".

2.2 SWITCHES

THERE ARE SEVERAL SWITCHES WHICH ARE USED TO MODIFY SUPERVISOR OPERATION. THESE SWITCHES ARE APPENDED TO THE LEGAL COMMANDS. ALL OF THE LEGAL SWITCHES ARE TABULATED BELOW WITH A BRIEF DESCRIPTION OF EACH. IN THE DESCRIPTIONS BELOW, A DECIMAL NUMBER IS DESIGNATED BY 'DDDD'.

SWITCH	EFFECT
/TESTS:LIST	EXECUTE ONLY THOSE TESTS SPECIFIED IN THE LIST. LIST IS A STRING OF TEST NUMBERS, FOR EXAMPLE - /TESTS:1:5:7-10. THIS LIST WILL CAUSE TESTS 1,5,7,8,9,10 TO BE RUN. ALL OTHER TESTS WILL NOT BE RUN.
/PASS:DDDD	EXECUTE DDDDD PASSES (DDDD = 1 TO 64000)
/FLAGS:FLGS	SET SPECIFIED FLAGS. FLAGS ARE DESCRIBED IN SECTION 2.3.
/EOP:DDDD	REPORT END OF PASS MESSAGE AFTER EVERY DDDDD PASSES ONLY. (DDDD = 1 TO 64000)
/UNITS:LIST	TEST/ADD/DROP ONLY THOSE UNITS SPECIFIED IN THE LIST. LIST EXAMPLE - /UNITS:0:5:10-12 USE UNITS 0,5,10,11,12 (UNIT NUMBERS = 0-63)

EXAMPLE OF SWITCH USAGE:

START/TESTS:1-5/PASS:1000/EOP:100

THE EFFECT OF THIS COMMAND WILL BE: 1) TESTS 1 THROUGH 5 WILL BE EXECUTED, 2) ALL UNITS WILL TESTED 1000 TIMES AND 3) THE END OF PASS MESSAGES WILL BE PRINTED AFTER EACH 100 PASSES ONLY. A SWITCH CAN BE RECOGNIZED BY THE FIRST THREE CHARACTERS. YOU MAY, FOR EXAMPLE, TYPE "/TES:1-5" INSTEAD OF "/TESTS:1-5".

BELOW IS A TABLE THAT SPECIFIES WHICH SWITCHES CAN BE USED BY EACH COMMAND.

	TESTS	PASS	FLAGS	EOP	UNITS
START	X	X	X	X	X
RESTART	X	X	X	X	X
CONTINUE		X	X	X	
PROCEED			X		
DROP					X
ADD					X
PRINT					
DISPLAY					X
FLAGS					
ZFLAGS					
EXIT					

2.3 FLAGS

FLAGS ARE USED TO SET UP CERTAIN OPERATIONAL PARAMETERS SUCH AS LOOPING ON ERROR. ALL FLAGS ARE CLEARED AT STARTUP AND REMAIN CLEARED UNTIL EXPLICITLY SET USING THE FLAGS SWITCH. FLAGS ARE ALSO CLEARED AFTER A START COMMAND UNLESS SET USING THE FLAG SWITCH. THE ZFLAGS COMMAND MAY ALSO BE USED TO CLEAR ALL FLAGS. WITH THE EXCEPTION OF THE START AND ZFLAGS COMMANDS, NO COMMANDS AFFECT THE STATE OF THE FLAGS; THEY REMAIN SET OR CLEARED AS SPECIFIED BY THE LAST FLAG SWITCH.

FLAG	EFFECT
HOE	HALT ON ERROR - CONTROL IS RETURNED TO RUNTIME SERVICES COMMAND MODE
LOE	LOOP ON ERROR
IER*	INHIBIT ALL ERROR REPORTS
IBE*	INHIBIT ALL ERROR REPORTS EXCEPT FIRST LEVEL (FIRST LEVEL CONTAINS ERROR TYPE, NUMBER, PC, TEST AND UNIT)
IXE*	INHIBIT EXTENDED ERROR REPORTS (THOSE CALLED BY PRINTX MACRO'S)
PRI	DIRECT MESSAGES TO LINE PRINTER
PNT	PRINT TEST NUMBER AS TEST EXECUTES
BOE	'BELL' ON ERROR
UAM	UNATTENDED MODE (NO MANUAL INTERVENTION)
ISR	INHIBIT STATISTICAL REPORTS (DOES NOT APPLY TO DIAGNOSTICS WHICH DO NOT SUPPORT STATISTICAL REPORTING)
IDU	INHIBIT PROGRAM DROPPING OF UNITS
ADR	EXECUTE AUTODROP CODE
LOT	LOOP ON TEST
EVL	EXECUTE EVALUATION (ON DIAGNOSTICS WHICH HAVE EVALUATION SUPPORT)

*ERROR MESSAGES ARE DESCRIBED IN SECTION 3.1

SEE THE XXDP+ USER'S MANUAL FOR MORE DETAILS ON FLAGS. YOU MAY SPECIFY MORE THAN ONE FLAG WITH THE FLAG SWITCH. FOR EXAMPLE, TO CAUSE THE PROGRAM TO LOOP ON ERROR, INHIBIT ERROR REPORTS AND TYPE A 'BELL' ON ERROR, YOU MAY USE THE FOLLOWING STRING:

/FLAGS:LOE:IER:BOE

2.4 HARDWARE QUESTIONS

WHEN A DIAGNOSTIC IS STARTED, THE RUNTIME SERVICES WILL PROMPT THE USER FOR HARDWARE INFORMATION BY TYPING "CHANGE HW (L) ?" YOU MUST ANSWER 'Y' AFTER A START COMMAND UNLESS THE HARDWARE INFORMATION HAS BEEN 'PRELOADED' USING THE SETUP UTILITY (SEE CHAPTER 6 OF THE XXDP+ USER'S MANUAL). WHEN YOU ANSWER THIS QUESTION WITH A 'Y', THE RUNTIME SERVICES WILL ASK FOR THE NUMBER OF UNITS (IN DECIMAL). YOU WILL THEN BE ASKED THE FOLLOWING QUESTION FOR EACH UNIT:

VTV31-K ADDRESS (O) 174000 ?

IN REPLY, YOU SHOULD ENTER AN ADDRESS IN OCTAL IN THE RANGE 160000 TO 177776. TO PREVENT THE QUESTIONS BEING OUTPUT FOR EACH UNIT, ALL ADDRESSES COULD BE ENTERED IN RESPONSE TO THE FIRST ADDRESS REQUEST AS IN THE FOLLOWING SAMPLE DIALOGUE:

UNITS (D) ? 4<CR>

UNIT 0

VTV31-K ADDRESS (O) 174000 ? 174000,174010,174020,174030<CR>

2.5 SOFTWARE QUESTIONS

AFTER YOU HAVE ANSWERED THE HARDWARE QUESTIONS OR AFTER A RESTART OR CONTINUE COMMAND, THE RUNTIME SERVICES WILL ASK FOR SOFTWARE PARAMETERS. THESE PARAMETERS WILL GOVERN SOME DIAGNOSTIC SPECIFIC OPERATION MODES. YOU WILL BE PROMPTED BY "CHANGE SW (L) ?" IF YOU WISH TO CHANGE ANY PARAMETERS, ANSWER BY TYPING 'Y'. THE FOLLOWING QUESTIONS WILL THEN BE ASKED:

RUN TEST 13 MANUALLY? (L) N ?
AUTODROP ERROR COUNT (D) 5 ?

TEST 13 OUTPUTS A NUMBER OF DISPLAY PATTERNS TO THE MONITOR. IF YOU ANSWER 'Y' TO THE FIRST QUESTION, EACH PATTERN WILL BE SHOWN UNTIL YOU PRESS THE CARRIAGE RETURN KEY. IF YOU RESPOND TO THE QUESTION BY TYPING 'N' OR CARRIAGE RETURN, EACH PATTERN WILL LAST APPROXIMATELY 5 SECONDS.

THE SECOND QUESTION REFERS TO THE NUMBER OF ERRORS WHICH MAY OCCUR ON A UNIT WITHIN A GIVEN PASS. IF THE NUMBER EXCEEDS THAT WHICH IS ENTERED (OR 5 IF THE RESPONSE IS A CARRIAGE RETURN), THE UNIT IS DROPPED FROM TESTING UNTIL A NEW COMMAND IS ENTERED. DROPPING IS PREVENTED IF EITHER THE IDU OR LOE FLAGS HAVE BEEN SELECTED.

2.6 CLOCK QUESTIONS

IF THERE IS NO LINE TIME CLOCK ON THE SYSTEM, THE USER IS ASKED TO TYPE 2 CHARACTERS 6 SECONDS APART ON THE CONSOLE.

2.7 QUICK START-UP PROCEDURE (XXDP+)

TO START-UP THIS PROGRAM:

1. BOOT XXDP+
2. GIVE THE DATE AND ANSWER THE LSI AND 50HZ (IF THERE IS A CLOCK) QUESTIONS
3. TYPE 'R NAME', WHERE NAME IS THE NAME OF THE BIN OR BIC FILE FOR THIS PROGRAM
4. TYPE "START"
5. ANSWER THE "CHANGE HW" QUESTION WITH 'Y'
6. ANSWER ALL THE HARDWARE QUESTIONS
7. ANSWER THE "CHANGE SW" QUESTION WITH 'N'

WHEN YOU FOLLOW THIS PROCEDURE YOU WILL BE USING ONLY THE DEFAULTS FOR FLAGS AND SOFTWARE PARAMETERS. THESE DEFAULTS ARE DESCRIBED IN SECTIONS 2.3 AND 2.5.

3.0 ERROR INFORMATION

3.1 TYPES OF ERROR MESSAGES

THERE ARE THREE LEVELS OF ERROR MESSAGES THAT MAY BE ISSUED BY A DIAGNOSTIC: GENERAL, BASIC AND EXTENDED. GENERAL ERROR MESSAGES ARE ALWAYS PRINTED UNLESS THE "IER" FLAG IS SET (SECTION 2.3). THE GENERAL ERROR MESSAGE IS OF THE FORM:

```
NAME TYPE NUMBER ON UNIT NUMBER TST NUMBER PC:XXXXXX
ERROR MESSAGE
```

WHERE; NAME = DIAGNOSTIC NAME
TYPE = ERROR TYPE (SYS FATAL, DEV FATAL, HARD OR SOFT)
NUMBER = ERROR NUMBER
UNIT NUMBER = 0 - N (N IS LAST UNIT IN PTABLE)
TST NUMBER = TEST AND SUBTEST WHERE ERROR OCCURRED
PC:XXXXXX = ADDRESS OF ERROR MESSAGE CALL

BASIC ERROR MESSAGES ARE MESSAGES THAT CONTAIN SOME ADDITIONAL INFORMATION ABOUT THE ERROR. THESE ARE ALWAYS PRINTED UNLESS THE "IER" OR "IBR" FLAGS ARE SET (SECTION 2.3). THESE MESSAGES ARE PRINTED AFTER THE ASSOCIATED GENERAL MESSAGE.

EXTENDED ERROR MESSAGES CONTAIN SUPPLEMENTARY ERROR INFORMATION SUCH AS REGISTER CONTENTS OR GOOD/BAD DATA. THESE ARE ALWAYS PRINTED UNLESS THE "IER", "IBR" OR "IXR" FLAGS ARE SET (SECTION 2.3).

THESE MESSAGES ARE PRINTED AFTER THE ASSOCIATED GENERAL ERROR MESSAGE AND ANY ASSOCIATED BASIC ERROR MESSAGES.

EXTENDED ERROR MESSAGES ARE NOT USED IN THIS DIAGNOSTIC, ALL SUPPLEMENTARY INFORMATION BEING OUTPUT IN THE BASIC ERROR MESSAGES.

3.2 SPECIFIC ERROR MESSAGES

ALL FAULTS DETECTED BY THE DIAGNOSTIC CAUSE A ONE LINE GENERAL ERROR MESSAGE TO BE PRINTED. THE TEST NUMBER IN WHICH THE FAULT OCCURS IS CONTAINED IN THE HIGHER THREE DIGITS OF THE ERROR NUMBER. THE LOWER TWO DIGITS ARE NUMBERED FROM ZERO FOR EACH TEST.

EACH GENERAL MESSAGE MAY BE FOLLOWED BY A BASIC ERROR MESSAGE CONSISTING OF SEVERAL LINES.

THE FOLLOWING LIST PROVIDES A BRIEF DESCRIPTION OF EACH OF THE ERRORS.

ERROR	DESCRIPTION
00100	'ADDRESSING THE CSR CAUSES AN NXM TRAP' THIS IS PROBABLY CAUSED BY THE DEVICE CSR ADDRESS BEING ENTERED INCORRECTLY.
00101	'ADDRESSING THE DBUFF REGISTER CAUSES AN NXM TRAP' THIS IS PROBABLY CAUSED BY THE DEVICE CSR ADDRESS BEING ENTERED INCORRECTLY.
00200	'CSR READY BIT NOT SET' 'CSR CONTENTS NNNNNN'
00201	'CSR BITS 3 OR 6 NOT CLEAR' 'CSR CONTENTS NNNNNN' BITS OTHER THAN 3 AND 6 MAY BE SET TO 0 OR 1.
00202	'CSR FIELD BIT DOES NOT GET SET' 'CSR CONTENTS NNNNNN'
00203	'CSR FIELD BIT DOES NOT GET CLEARED' 'CSR CONTENTS NNNNNN'
00300	'CSR CHANGED BY WRITING TO DBUFF' 'OLD CSR CONTENTS NNNNNN' 'NEW CSR CONTENTS NNNNNN (IGNORE BITS 0,1,2 AND 4)' ' DATA WRITTEN TO DBUFF NNNNNN' THIS INDICATES ADDRESS INTERACTION BETWEEN THE CSR AND DBUFF REGISTERS.
00400	'COMBINATION OF CSR BITS 5 AND 8 - 15 COULD NOT BE SET.' 'ATTEMPTED SETTING NNNNNN'

- 'ACTUAL SETTING NNNNNN (IGNORE BITS 0-4 AND 6-7)
INDICATES BIT 'STICKING' OR BIT INTERACTION.
- 00401 'COMBINATION OF CSR BITS 0 - 2 COULD NOT BE SET VIA
THE DBUFF REGISTER.'
'ATTEMPTED SETTING NNNNNN'
'ACTUAL SETTING NNNNNN (IGNORE BITS 3-15)'

COULD INDICATE BIT SET ERRORS IN THE CSR OR DBUFF
REGISTERS OR AN ERROR IN THE FIRST LOCATION OF THE
PICTURE STORE RAM.
- 00500 'BACKGROUND COLOUR READ BACK INCORRECTLY'
'LINE NNN (NNN OCTAL), DOT NNN (NNN OCTAL)'
'EXPECTED CONTENTS N, ACTUAL CONTENTS N'

THE ADDRESS LINES HAVE NOT BEEN TESTED AT THIS STAGE.
THEREFORE, THE LINE AND DOT NUMBERS MAY NOT BE THOSE
ACTUALLY WRITTEN TO.
- 00501 'DATA READ BACK INCORRECTLY FROM PICTURE STORE'
'LINE NNN (NNN OCTAL), DOT NNN (NNN OCTAL)'
'EXPECTED CONTENTS N, ACTUAL CONTENTS N'

AS FOR ERROR 500, THE LINE AND DOT NUMBERS MAY NOT
BE THOSE ACTUALLY WRITTEN TO.
- 00600 'PICTURE STORE COLUMN ADDRESSING ERROR'
'LINE NNN (NNN OCTAL), DOT NNN (NNN OCTAL)'
'EXPECTED CONTENTS N, ACTUAL CONTENTS N'
'ADDRESS BITS BEING TESTED NNNNNNNNN'

THIS INDICATES INTERACTION BETWEEN THE PICTURE STORE
COLUMN ADDRESS LINES.
- 00700 'PICTURE STORE LINE ADDRESSING ERROR'
'LINE NNN (NNN OCTAL), DOT NNN (NNN OCTAL)'
'EXPECTED CONTENTS N, ACTUAL CONTENTS N'
'ADDRESS BITS BEING TESTED NNNNNNNNN'

THIS INDICATES INTERACTION BETWEEN THE PICTURE STORE
LINE ADDRESS LINES.
- 00800 '8 - DOT WRITE FAILURE'
'DOT ADDRESS WRITTEN NNN, FAILING DOT ADDRESS NNN'
'EXPECTED CONTENTS N, ACTUAL CONTENTS N'

HAVING MADE AN 8-DOT WRITE TO THE ADDRESS SHOWN, THE
EXPECTED COLOUR WAS NOT READ BACK FROM THE ADDRESS
GIVEN AS THE FAILING ADDRESS.
- 00900 'CSR READY NOT CLEAR DURING A PRESET'
'CSR CONTENTS NNNNNN, EXPECTED COLOUR N'

IMMEDIATELY AFTER ISSUING A PRESET COMMAND, THE CSR
READY BIT DID NOT READ BACK AS ZERO.

00901 'CSR READY NOT SET 50 MILLISECONDS AFTER A PRESET'
'CSR CONTENTS NNNNNN, EXPECTED COLOUR'

AFTER A DELAY OF 50 MILLISECONDS FOLLOWING
A PRESET, THE CSR READY BIT WAS NOT SET. IF THE
PRESET DID ACTUALLY OCCUR, THE EXPECTED COLOUR SHOULD
BE SEEN ON THE MONITOR AND BE REFLECTED IN THE CSR.

00902 'DOT NOT SET CORRECTLY AFTER A PRESET'
'LINE NNN (NNN OCTAL), DOT NNN (NNN OCTAL)'
'EXPECTED CONTENTS N, ACTUAL CONTENTS N'

FOLLOWING THE PRESET, THE PICTURE STORE ADDRESS SHOWN
DID NOT CONTAIN THE PRESET COLOUR.

4.0 PERFORMANCE AND PROGRESS REPORTS

AT THE END OF EACH PASS, THE PASS COUNT IS GIVEN ALONG WITH THE
TOTAL NUMBER OF ERRORS REPORTED SINCE THE DIAGNOSTIC WAS STARTED.
THE 'EOP' SWITCH CAN BE USED TO CONTROL HOW OFTEN THE END
OF PASS MESSAGE IS PRINTED. SECTION 2.2 DESCRIBES SWITCHES.

5.0 DEVICE INFORMATION TABLES

THE HARDWARE PTABLE CREATED AT START UP CONTAINS A SINGLE
WORD ENTRY FOR EACH DEVICE UNDER TEST. THIS WORD IS LOADED
WITH THE DEVICE CSR ADDRESS AS THE HARDWARE QUESTIONS ARE
ANSWERED. AT THE END OF EACH SUBPASS, THE CSR ADDRESS OF THE
NEXT UNIT TO BE TESTED IS LOADED INTO LOCATION VTVCSR. THE
ADDRESS+2 IS LOADED INTO LOCATION DBUFF. THE TEST CODE
REFERENCES THE DEVICE THROUGH THESE TWO LOCATIONS.

6.0 TEST SUMMARIES

ALL OF THE FOLLOWING TESTS ARE RUN WHENEVER THE PROGRAM IS
RUN (BY DEFAULT) UNLESS SPECIFICALLY EXCLUDED BY THE TEST
SWITCH (SECTION 2.2). TESTS 1 TO 9 REQUIRE NO OPERATOR
INTERVENTION, THE DIAGNOSTIC VERIFYING THAT THE REQUIRED
ACTION IS PERFORMED CORRECTLY. THE REMAINING TESTS MUST BE
VERIFIED BY OPERATOR OBSERVATION, ALTHOUGH THEY WILL RUN
WITHOUT A DISPLAY MONITOR ON THE SYSTEM.

6.1 LOGIC TESTS

TEST 1 - REGISTER NXM

CHECKS ARE MADE THAT ACCESSING THE DEVICE CSR AND DBUFF
REGISTERS DOES NOT CAUSE AN NXM TRAP.

TEST 2 - REGISTER ACCESS

THE DEVICE CSR IS READ AND A CHECK MADE THAT THE READY BIT IS
SET, THAT BITS 3 AND 6 ARE CLEAR, AND THAT THE FIELD BIT IS
SET AND CLEARED BY THE DEVICE.

TEST 3 - REGISTER ADDRESS UNIQUENESS

ALL BITS IN THE DBUFF REGISTER ARE SET AND CLEARED IN ALL COMBINATIONS WITH THE EXCEPTION OF BIT 14 (READ PICTURE STORE), WHICH IS LEFT CLEAR. FOR EACH BIT COMBINATION, THE CSR IS READ AND CHECKED TO BE UNCHANGED. CSR BITS 0,1,2 AND 4 ARE IGNORED IN THIS CHECK.

TEST 4 - REGISTER BIT SET

BITS 5 AND BITS 8 TO 15 IN THE CSR ARE SET AND CLEARED IN ALL COMBINATIONS. FOR EACH COMBINATION, THE CSR IS READ AND A CHECK MADE THAT THE RELEVANT BITS ARE SET.

THE DBUFF REGISTER IS THEN USED IN CONJUNCTION WITH THE FIRST PICTURE STORE ELEMENT TO ENSURE THAT CSR BITS 0 - 2 CAN BE SET. CSR BITS 8 - 15 AND DBUFF BITS 0 - 8 ARE CLEAR THROUGHOUT, ENSURING THAT ONLY THE FIRST PICTURE ELEMENT IS USED.

A 'WRITE PICTURE STORE' OPERATION IS PERFORMED USING THE COLOUR BLACK. THE ELEMENT IS THEN READ AND A CHECK MADE THAT CSR BITS 0 - 2 READ ZERO. THIS PROCESS IS REPEATED FOR EACH COLOUR CODE, VERIFYING THAT THE CORRECT COLOUR IS REFLECTED IN THE CSR.

TEST 5 - PICTURE STORE DATA

A ZERO DATA PATTERN (COLOUR BLACK) IS WRITTEN TO THE FIRST PICTURE STORE LOCATION AND CHECKED TO READ BACK CORRECTLY IN THE CSR. THIS IS REPEATED FOR ALL PICTURE LOCATIONS TO FORM AN ALL BLACK BACKGROUND.

DATA IN ALL 8 COMBINATIONS IS THEN WRITTEN TO EACH PICTURE STORE LOCATION IN TURN AND AFTER EACH WRITE OPERATION, THE CSR IS EXAMINED TO ENSURE THAT THE DATA READS BACK CORRECTLY. EACH LOCATION IS RESET TO THE BACKGROUND COLOUR AFTER THE ABOVE OPERATION.

THE ENTIRE TEST IS THEN REPEATED USING WHITE (DATA 7) AS THE BACKGROUND COLOUR.

TEST 6 - PICTURE STORE COLUMN ADDRESSING

THIS VERIFIES THAT WRITING TO ANY COLUMN OF THE PICTURE STORE DOES NOT AFFECT THE CONTENTS OF ANY OTHER COLUMN. SINCE EACH LOCATION CAN HOLD A 3 BIT DATA VALUE, THE COLUMN ADDRESS LINES ARE CHECKED IN GROUPS OF THREE.

COLUMN ADDRESS BITS 0 - 2 ARE CHECKED BY WRITING THE VALUES 0 - 7 IN CONSECUTIVE LOCATIONS THROUGHOUT THE PICTURE STORE AND THEN CHECKING THAT THE DATA READS BACK CORRECTLY VIA THE CSR.

COLUMN ADDRESS BITS 3 - 5 ARE CHECKED BY WRITING ZERO TO THE FIRST 8 LOCATIONS, 1 TO THE NEXT 8 LOCATIONS ETC. UNTIL THE PICTURE STORE IS FULL. THE DATA IS THEN VERIFIED VIA THE CSR.

COLUMN ADDRESS BITS 6 - 8 ARE CHECKED BY WRITING ZERO TO THE

FIRST 64 LOCATIONS, ONE TO THE NEXT 64 LOCATIONS AND SO ON, UNTIL THE PICTURE STORE IS FULL. THE DATA IS THEN VERIFIED VIA THE CSR.

TEST 7 - PICTURE STORE LINE ADDRESSING

THIS VERIFIES THAT WRITING TO ANY LINE OF THE PICTURE STORE DOES NOT AFFECT THE CONTENTS OF ANY OTHER LINE. SINCE EACH LOCATION CAN HOLD A 3 BIT DATA VALUE, THE LINE ADDRESS LINES ARE CHECKED IN GROUPS OF THREE.

LINE ADDRESS BITS 0 - 2 (CSR BITS 8 - 10) ARE CHECKED BY WRITING ZERO TO ALL LOCATIONS IN THE FIRST LINE, ONE TO ALL LOCATIONS IN THE NEXT LINE AND SO ON, UNTIL ALL LINES ARE WRITTEN TO. THE DATA IS THEN READ BACK AND VERIFIED VIA THE CSR.

LINE ADDRESS BITS 3 - 5 (CSR BITS 11 - 13) ARE CHECKED BY WRITING ZERO TO THE FIRST EIGHT LINES, ONE TO THE NEXT EIGHT LINES ETC. TO THE END OF THE PICTURE STORE. THE DATA IS THEN VERIFIED VIA THE CSR.

LINE ADDRESS BIT 6 AND 7 (CSR BITS 14 AND 15) ARE CHECKED BY WRITING ZERO TO THE FIRST 64 LINES, ONE TO THE NEXT 64 LINES, TWO TO THE NEXT 64 LINES, AND THREE TO THE LAST 64 LINES. THE PICTURE STORE DATA IS THEN READ BACK AND VERIFIED.

TEST 8 - 8 DOT

THIS TEST CHECKS THAT IF DBUFF BIT 15 (8-DOT) IS SET, PICTURE STORE WRITES ARE MADE TO EIGHT DOTS AT A TIME, REGARDLESS OF THE VALUES OF DBUFF BITS 0 - 2. FOR EACH COMBINATION OF THE LOWER DOT ADDRESS BITS, A NEW COLOUR IS USED AND THE 8-DOT WRITE VERIFIED.

TEST 9 - PRESET

DBUFF BITS 10 - 12 ARE SET TO THE COLOUR BLACK (ZERO) AND A PRESET IS PERFORMED. THE READY BIT IN THE CSR IS CHECKED TO CLEAR DURING THE PRESET, AND AFTER 50 MILLISECONDS, TO BE SET. THE PICTURE STORE IS THEN READ TO CONFIRM THAT ALL LOCATIONS HAVE BEEN CORRECTLY SET. THE PROCESS IS THEN REPEATED FOR EACH OF THE EIGHT SELECTABLE COLOURS.

6.2 VISUAL TESTS

TEST 10 - VIDEO ENABLE

THE PICTURE STORE IS PRESET TO BLUE FOR FIVE SECONDS. BIT 5 OF THE CSR (VIDEO ENABLE) IS THEN CLEARED FOR FIVE SECONDS, DURING WHICH TIME THE MONITOR DISPLAY SHOULD BE DARK. THE DISPLAY IS THEN REENABLED FOR A FURTHER FIVE SECONDS.

TEST 11 - DISPLAY ADDRESSING

THE PICTURE STORE IS PRESET TO BLUE. LINES OF MAGENTA ARE THEN

WRITTEN FROM THE TOP OF THE DISPLAY UNTIL THE WHOLE SCREEN HAS CHANGED COLOUR. THE TRANSITION SHOULD BE SMOOTH, TAKING APPROXIMATELY 7 SECONDS. COLUMNS OF WHITE ARE THEN WRITTEN FROM THE LEFT SIDE OF THE DISPLAY UNTIL THE SCREEN IS ALL WHITE. AGAIN, THE TRANSITION SHOULD BE SMOOTH. IF THE DISPLAY DOES NOT APPEAR AS DESCRIBED HERE, AN ADDRESSING FAULT ON THE OUTPUT SIDE OF THE DEVICE IS INDICATED.

TEST 12 - OFFSET

THE PICTURE STORE IS PRESET TO BLUE. A LINE OF MAGENTA IS THEN WRITTEN TO THE TOP OF THE DISPLAY AND THE OFFSET REGISTER CHANGED TO MOVE THE LINE DOWN ONE POSITION. THIS IS REPEATED UNTIL THE WHOLE DISPLAY IS MAGENTA, WHICH SHOULD TAKE ABOUT 7 SECONDS. THE DISPLAY IS THEN CHANGED TO WHITE FROM THE BOTTOM USING THE SAME PROCESS, UNTIL THE SCREEN IS ALL WHITE. IN BOTH CASES, THE LINE BETWEEN COLOURS SHOULD MOVE SMOOTHLY ACROSS THE SCREEN. ANY UNEVEN MOVEMENT INDICATES A PROBLEM IN THE OFFSET REGISTER.

TEST 13 - PATTERN GENERATOR

THIS TEST IS FOR SETTING UP AND CHECKING THE OUTPUT TO THE DISPLAY MONITOR. NINE PATTERNS ARE PROVIDED - EIGHT TO PRESET THE SCREEN TO EACH POSSIBLE COLOUR, AND ONE TO OUTPUT A CROSS HATCH OF 32 COLUMN, 16 LINE RECTANGLES. THE NUMBER OF RECTANGLES WHICH ARE DISPLAYED DEPENDS ON THE CONFIGURATION BEING USED - 12/16 ACROSS THE SCREEN FOR 384/512 DOT VERSIONS, AND 15/16 DOWN THE SCREEN FOR 60/50 HZ VERSIONS.

NORMALLY, THE NINE PATTERNS ARE DISPLAYED IN TURN, EACH REMAINING ON THE SCREEN FOR APPROXIMATELY FIVE SECONDS. HOWEVER, IF MANUAL SELECTION IS SELECTED AT START UP, EACH PATTERN IS KEPT DISPLAYED UNTIL THE RETURN KEY IS HIT ON THE OPERATOR CONSOLE.


```

12      .TITLE PROGRAM HEADER AND TABLES
13      .SBTTL PROGRAM HEADER
39
41 000000      .ENABL ABS,AMA
42      002000      =      2000
44
45 002000      BGNMOD
46
47      :++
48      : THE PROGRAM HEADER IS THE INTERFACE BETWEEN
49      : THE DIAGNOSTIC PROGRAM AND THE SUPERVISOR.
50      :--
51
52 002000      POINTER BGNSW,BGNSFT,BGNDU,BGNSETUP
53
70
71 002000      HEADER CZVTV,A,0,660.,0

```

```

LSNAME::      .ASCII /C/
              .ASCII /Z/
              .ASCII /V/
              .ASCII /T/
              .ASCII /V/
              .BYTE 0
              .BYTE 0
              .BYTE 0
LSREV::      .ASCII /A/
LSDEPO::      .ASCII /0/
LSUNIT::      .WORD TSPTHV
LSTIML::      .WORD 660.
LSHPCP::      .WORD LSHARD
LSSPCP::      .WORD LSSOFT
LSHPTP::      .WORD LSHW
LSSPTP::      .WORD LSSW
LSLADP::      .WORD LSLAST
LSSTA::      .WORD 0
LSCO::      .WORD 0
LSDTYP::      .WORD 0
LSAPT::      .WORD 0
LSDTP::      .WORD 0
LSPRIO::      .WORD LSDISPATCH
LSENV1::      .WORD 0

```

PROGRAM HEADER AND TABLES
PROGRAM HEADER

```

002044 000000
002046
002046 000000
002050
002050 003
002051 003
002052
002052 000000
002054 000000
002056
002056 000000
002060
002060 002210
002062
002062 000000
002064
002064 000000
002066
002066 000000
002070
002070 000000
002072
002072 005562
002074
002074 000000
002076
002076 002220
002100
002100 104035
002102
002102 000000
002104
002104 004722
002106
002106 005510
002110
002110 005424
002112
002112 004714
002114
002114 000000
002116
002116 000000
002120
002120 000000

```

72

```

L$EXP1:: .WORD 0
L$MREV:: .WORD 0
          .BYTE CSREVISION
L$EF::   .BYTE CS$EDIT
          .WORD 0
          .WORD 0
L$SPC::  .WORD 0
L$DEVP:: .WORD LSDVTYP
L$REPP:: .WORD 0
L$EXP4:: .WORD 0
L$EXP5:: .WORD 0
L$AUT::  .WORD 0
L$DUT::  .WORD LSDU
L$LUN::  .WORD 0
L$DESP:: .WORD L$DESC
L$LOAD:: .WORD L$LOAD
L$ETP::  .WORD 0
L$ICP::  .WORD L$INIT
L$CCP::  .WORD L$CLEAN
L$ACP::  .WORD L$AUTO
L$PRT::  .WORD L$PROT
L$TEST:: .WORD 0
L$DLY::  .WORD 0
L$HIME:: .WORD 0
          .WORD 0

```


1
2
3
4
5
6
7
8

.SBTTL DISPATCH TABLE

:+
: THE DISPATCH TABLE CONTAINS THE STARTING ADDRESS OF EACH TEST.
: IT IS USED BY THE SUPERVISOR TO DISPATCH TO EACH TEST.
:--

DISPATCH 13

002122
002122 000015
002124
002124 005640
002126 006140
002130 006452
002132 006606
002134 007200
002136 007632
002140 010164
002142 010520
002144 010714
002146 011316
002150 011424
002152 011610
002154 012030

.WORD 13
LSDISPATCH::
.WORD T1
.WORD T2
.WORD T3
.WORD T4
.WORD T5
.WORD T6
.WORD T7
.WORD T8
.WORD T9
.WORD T10
.WORD T11
.WORD T12
.WORD T13

9

.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 HARDWARE P-TABLES,
: AND IS USED AS A "TEMPLATE" FOR BUILDING THE P-TABLES.
:--

```
1  
2  
3  
4  
5  
6  
7  
8  
9  
10 002156          BGNHW  DFPTBL  
    002156 000001  
    002160  
    002160          LSHW:: .WORD L10000-LSHW/2  
                                DFPTBL::  
11 002160 174000    .WORD 174000    ; VTV31-K CSR ADDRESS  
12  
13 002162          ENDDHW  
    002162          L10000:
```



```
1      .SBTTL  SOFTWARE P-TABLE
2
3
4      :++
5      : THE SOFTWARE TABLE CONTAINS VARIOUS DATA USED BY THE
6      : PROGRAM AS OPERATIONAL PARAMETERS.  THESE PARAMETERS ARE
7      : SET UP AT ASSEMBLY TIME AND MAY BE VARIED BY THE OPERATOR
8      : AT RUN TIME.
9      :--
10     BGNSW  SFPTBL
11     002162 000002
12     002162
13     002164
14     002164
15
16     LSSW:: .WORD  L10001-LSSW/2
17     SFPTBL::
18
19     002164 000000
20     MANVEN:: .WORD  0      ; TEST 13 MANUAL INTERVENTION FLAG
21     ; 0 FOR NO INTERVENTION (DEFAULT)
22     ; 1 FOR RUN MANUALLY
23     002166 000005
24     MAXERR:: .WORD  5      ; AUTODROP ERROR COUNT
25     ENDSW
26
27     002170
28     ENDMOD
29
30     L10001:
```

12
13
41
51
52 002170
53
54
55
56
57
58
73
74 002170

.TITLE GLOBAL AREAS
.SBTTL GLOBAL EQUATES SECTION

BGNMOD

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

EQUALS

:
: 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

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34

.SBTTL GLOBAL DATA SECTION

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

: UUT REGISTER ADDRESS STORAGE

UUT::	.WORD	0	: CURRENT UNIT UNDER TEST
VTCSR::	.WORD	0	: CSR ADDRESS
DBUFF::	.WORD	0	: DBUFF ADDRESS
ERRCNT::	.WORD	0	: ERROR COUNT FOR CURRENT UNIT
DOT::	.WORD	0	: DOT POSITION FOR ERROR REPORTS
NXMFLG::	.WORD	0	: SET IF NXM ERROR OCCURS
ABITS::	.WORD	0	: BIT MASK STORE FOR ADDRESS TESTS
COUNT::	.WORD	0	: COUNTER FOR WAIT25 SUBROUTINE

002170	000000
002172	000000
002174	000000
002176	000000
002200	000000
002202	000000
002204	000000
002206	000000


```

1      .SBTTL GLOBAL TEXT SECTION
2
3
4      :++
5      : THE GLOBAL TEXT SECTION CONTAINS FORMAT STATEMENTS,
6      : MESSAGES, AND ASCII INFORMATION THAT ARE USED IN
7      : MORE THAN ONE TEST.
8      :--
9
10     :
11     : NAMES OF DEVICES SUPPORTED BY PROGRAM
12     :
12     DEVTYP <VTV31-K>
12     002210
12     002210
12     002210      126      124      126
12     002213      063      061      055
12     002216      113      000
12
12     LSDVTYP::
12     .ASCIZ /VTV31-K/
12
12     .EVEN
13
19
20     : TEST DESCRIPTION
21     :
22     DESCRIPT      <VTV31-K INTERFACE DIAGNOSTIC>
22     002220
22     002220
22     002220      126      124      126
22     002223      063      061      055
22     002226      113      040      111
22     002231      116      124      105
22     002234      122      106      101
22     002237      103      105      040
22     002242      104      111      101
22     002245      107      116      117
22     002250      123      124      111
22     002253      103      000
22
22     L$DESC::
22     .ASCIZ /VTV31-K INTERFACE D
22
22     .EVEN
23
23     .EVEN
24
31
    
```

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

.SBTTL GLOBAL SUBROUTINES

: THE GLOBAL SUBROUTINES ARE CALLED BY MORE THAN ONE TEST. *

++
: FUNCTIONAL DESCRIPTION:

SUBROUTINE TO UPDATE UNIT ERROR COUNT. IF DURING ONE PASS, THE
ERROR COUNT EXCEEDS MAXERR (THE AUTODROP COUNT REQUESTED AT
STARTUP), THE UNIT IS DROPPED FROM THE TEST CYCLE.
IF THE PROGRAM IS LOOPING ON AN ERROR OR IF THE FLAG IDU HAS
BEEN SELECTED, THE SUBROUTINE DOES NOTHING.

: CALLING SEQUENCE:

JSR PC,CHKMAX

: INPUTS:

NONE.

: IMPLICIT INPUTS:

THE VARIABLE MAXERR CONTAINS THE NUMBER OF ERRORS AFTER WHICH
TO DROP THE UNIT.
UT CONTAINS THE NUMBER OF THE UNIT CURRENTLY BEING TESTED.

: OUTPUTS:

NONE.

: IMPLICIT OUTPUTS:

THE SUBPASS ERROR COUNT (ERRCNT) IS INCREMENTED.

: SUBORDINATE ROUTINES USED:

NONE.

: FUNCTIONAL SIDE EFFECTS:

NONE.

--
CHKMAX::INLOOP ; LOOPING ON ERROR? TRAP C\$INLP
BCOMplete 1\$; IF YES, EXIT BCS 1\$
RFLAGS R0 ; GET OPERATOR FLAGS TRAP CSRFLA
BIT #IDU,R0 ; IS DROPPING INHIBITED?
BNE 1\$; IF YES, EXIT

002256
002256 104420
002260
002260 103432
002262
002262 104421
002264 032700 000040
002270 001026


```

55
56 002272 005267 177700      INC      ERRCNT      : UPDATE THE ERROR COUNT
57 002276 026767 177674 177662  CMP      ERRCNT,MAXERR : TOO MANY ERRORS?
58 002304 003420      BLE      1$          : IF NOT, JUMP
59 002306      PRINTF   #NERRS,MAXERR,UUT : 'TOO MANY ERRORS'
      002306 016746 177656      MOV      UUT,-(SP)
      002312 016746 177650      MOV      MAXERR,-(SP)
      002316 012746 002350      MOV      #NERRS,-(SP)
      002322 012746 000003      MOV      #3,-(SP)
      002326 010600      MOV      SP,R0
      002330 104417      TRAP    C$PNTF
      002332 062706 000010      ADD     #10,SP
60 002336      DODU     UUT          : DROP THE UNIT
      002336 016700 177626      MOV     UUT,R0
      002342 104451      TRAP   C$DODU
61
62 002344      DOCLN          : END THE SUBPASS
      002344 104444      TRAP   C$DCLN
63
64 002346 000207      1$:   RTS     PC
65
66
67 002350      045      116      045  NERRS: .NLIST  BEX
      .ASCIZ  /%N%AMORE THAN%D3%A ERRORS ON UNIT%D2/
68      .LIST  BEX
69      .EVEN
70
71
72

```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27

```
..++  
: FUNCTIONAL DESCRIPTION:  
:     SUBROUTINE TO WAIT FOR 25 MILLISECONDS  
: CALLING SEQUENCE:  
:     JSR     PC, WAIT25  
: INPUTS:  
:     NONE.  
: IMPLICIT INPUTS:  
:     THE VARIABLE COUNT MUST HAVE BEEN SET UP BY ROUTINE SETCLK.  
: OUTPUTS:  
:     NONE.  
: IMPLICIT OUTPUTS:  
:     NONE.  
:--
```

28 002416 016700 177564
29 002422 005300
30 002424 001376
31 002426 000207

```
WAIT25: :MOV     COUNT, R0      : GET WAIT COUNT  
1$:     :DEC     R0           : ALL DONE?  
        :BNE    1$          : IF NOT, WAIT SOME MORE  
        :RTS    PC           : ELSE RETURN
```


1
2
3
4
5
6
7
8
9

.SBTTL GLOBAL INTERRUPT SERVICE ROUTINES

:*****
: THESE ROUTINES ARE USED IN MORE THAN ONE PLACE IN THE PROGRAM. *
:*****

10 002430 012737 000001 002202
11 002436 012702 000001
12 002442
002442
002442 000002

BGNSRV NXMTRP

MOV #1,@NXMFLG
MOV #1,R2

: FLAG UNIT NXM ERROR
: FLAG REGISTER NXM ERROR

NXMTRP::

ENDSRV

L10002:
RTI

.SBTTL GLOBAL ERROR REPORT SECTION

:++
: THE GLOBAL ERROR REPORT SECTION CONTAINS MESSAGE PRINTING AREAS
: USED BY MORE THAN TEST TO OUTPUT ADDITIONAL ERROR INFORMATION. PRINTB
: (BASIC) AND PRINTX (EXTENDED) CALLS ARE USED TO CALL PRINT SERVICES.
:--

1
2
3
4
5
6
7
8
9
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39

002444
002444
002444
002446 010146
002446 012746 003636
002452 012746 000002
002456 010600
002460 104414
002462 062706 000006
002466 004767 177564
002472
002472 104423
002474
002474
002474
002474 010346
002476 010146
002500 012746 003663
002504 012746 000003
002510 010600
002512 104414
002514 062706 000010
002520
002520 010246
002522 012746 003776
002526 012746 000002
002532 010600
002534 104414
002536 062706 000006
002542 004767 177510
002546
002546 104423
002550
002550
002550
002550 010446
002552 010146
002554 012746 004036
002560 012746 000003
002564 010600
002566 104414
002570 062706 000010
002574 004767 177456

BGNMSG ER200
PRINTB #E200B,R1 ; PRINT CSR CONTENTS
ER200::
JSR PC,CHKMAX ; CHECK FOR TOO MANY ERRORS
ENDMSG
L10003:
BGNMSG ER300
PRINTB #E300B,R1,R3 ; PRINT REGISTER CONTENTS
PRINTB #E300C,R2 ;
JSR PC,CHKMAX ; CHECK FOR TOO MANY ERRORS
ENDMSG
L10004:
BGNMSG ER400
PRINTB #E400B,R1,R4 ; PRINT GOOD AND BAD
ER400::
JSR PC,CHKMAX ; CHECK FOR TOO MANY ERRORS

MOV R1,-(SP)
MOV #E200B,-(SP)
MOV #2,-(SP)
MOV SP,R0
TRAP C\$PNTB
ADD #6,SP
TRAP C\$MSG
MOV R3,-(SP)
MOV R1,-(SP)
MOV #E300B,-(SP)
MOV #3,-(SP)
MOV SP,R0
TRAP C\$PNTB
ADD #10,SP
MOV R2,-(SP)
MOV #E300C,-(SP)
MOV #2,-(SP)
MOV SP,R0
TRAP C\$PNTB
ADD #6,SP
TRAP C\$MSG
MOV R4,-(SP)
MOV R1,-(SP)
MOV #E400B,-(SP)
MOV #3,-(SP)
MOV SP,R0
TRAP C\$PNTB
ADD #10,SP


```

40 002600          ENDMSG                                L10005: TRAP CSMSG
    002600          104423
41 002602          BGNMSG ER401                          ER401::
42 002602          PRINTB #E401B,R3,R5 ; PRINT GOOD AND BAD
43 002602          010546                                MOV R5,-(SP)
    002604          010346                                MOV R3,-(SP)
    002606          012746 004152                          MOV #E401B,-(SP)
    002612          012746 000003                          MOV #3,-(SP)
    002616          010600                                MOV SP,R0
    002620          104414                                TRAP C$PNTB
    002622          062706 000010                          ADD #10,SP
44 002626          JSR PC,CHKMAX ; CHECK FOR TOO MANY ERRORS
45 002632          ENDMSG                                L10006: TRAP CSMSG
    002632          104423
46 002634          BGNMSG ER500                          ER500::
47 002634          SWAB R3 ; GET LINE NUMBER
48 002634          000303                                BIC #177400,R3 ; CLEAR OTHER BITS
49 002636          042703 177400                          PRINTB #LINDOT,R3,R3,R1,R1 ; LINE AND DOT NUMBER
50 002642          010146                                MOV R1,-(SP)
    002644          010146                                MOV R1,-(SP)
    002646          010346                                MOV R3,-(SP)
    002650          010346                                MOV R3,-(SP)
    002652          012746 004503                          MOV #LINDOT,-(SP)
    002656          012746 000005                          MOV #5,-(SP)
    002662          010600                                MOV SP,R0
    002664          104414                                TRAP C$PNTB
    002666          062706 000014                          ADD #14,SP
51 002672          PRINTX #GOOBAD,R2,R3 ; GOOD AND BAD
    002672          010346                                MOV R3,-(SP)
    002674          010246                                MOV R2,-(SP)
    002676          012746 004571                          MOV #GOOBAD,-(SP)
    002702          012746 000003                          MOV #3,-(SP)
    002706          010600                                MOV SP,R0
    002710          104415                                TRAP C$PNTX
    002712          062706 000010                          ADD #10,SP
52 002716          JSR PC,CHKMAX ; CHECK FOR TOO MANY ERRORS
53 002722          ENDMSG                                L10007: TRAP CSMSG
    002722          104423
54 002724          BGNMSG ER501                          ER501::
55 002724          SWAB R4 ; GET THE LINE NUMBER
56 002724          000304                                BIC #177400,R4 ; CLEAR THE OTHER BITS
57 002726          042704 177400                          MOV R1,@#DOT ; GET THE DATA PATTERN
58 002732          010137 002200                          BIC #177000,@#DOT ; GET THE DOT NUMBER
59 002736          042737 177000 002200                    PRINTB #LINDOT,R4,R4,DOT,DOT ; LINE AND DOT NUMBER
60 002744          016746 177230                                MOV DOT,-(SP)
    002750          016746 177224                                MOV DOT,-(SP)
    002754          010446                                MOV R4,-(SP)
    002756          010446                                MOV R4,-(SP)

```

```

002760 012746 004503
002764 012746 000005
002770 010600
002772 104414
002774 062706 000014
61 003000          PRINTX #GOOBAD,R2,R3 ; GOOD AND BAD
003000 010346
003002 010246
003004 012746 004571
003010 012746 000003
003014 010600
003016 104415
003020 062706 000010
62 003024 004767 177226
63 003030          ENDMSG
003030
003030 104423          L10010: TRAP C$MSG
64
65 003032          BGNMSG ER600
003032
66 003032 000305          SWAB R5 ; GET THE LINE NUMBER ER600::
67 003034 042705 177400 BIC #177400,R5 ; CLEAR OTHER BITS
68 003040 010137 002200 MOV R1,@#DOT ; GET THE DOT NUMBER
69 003044 042737 177000 002200 BIC #177000,@#DOT ; CLEAR THE OTHER BITS
70 003052          PRINTB #LINDOT,R5,R5,DOT,DOT ; LINE AND DOT NUMBER
003052 016746 177122
003056 016746 177116
003062 010546
003064 010546
003066 012746 004503
003072 012746 000005
003076 010600
003100 104414
003102 062706 000014
71 003106          PRINTX #GOOBAD,R4,R3 ; GOOD AND BAD
003106 010346
003110 010446
003112 012746 004571
003116 012746 000003
003122 010600
003124 104415
003126 062706 000010
72 003132 016767 004754 177044 MOV NCOLS,ABITS ; GET NO. OF COLUMNS PER COLOUR
73 003140 006367 177040 ASL ABITS ; SHIFT TO FORM
74 003144 006367 177034 ASL ABITS ; BIT MASK OF
75 003150 006367 177030 ASL ABITS ; ADDRESS LINES
76 003154 166767 004732 177022 SUB NCOLS,ABITS ; BEING TESTED
77 003162          PRINTX #EADD,ABITS ; AND ADDRESS LINES BEING TESTED
003162 016746 177016
003166 012746 004652
003172 012746 000002
003176 010600
003200 104415
003202 062706 000006
78 003206 004767 177044          JSR PC,CHKMAX ; CHECK FOR TOO MANY ERRORS
79 003212          ENDMSG
003212          L10011:

```

```

MOV #LINDOT, -(SP)
MOV #5, -(SP)
MOV SP,R0
TRAP C$PNTB
ADD #14,SP
MOV R3, -(SP)
MOV R2, -(SP)
MOV #GOOBAD, -(SP)
MOV #3, -(SP)
MOV SP,R0
TRAP C$PNTX
ADD #10,SP
TRAP C$MSG
ER600::
MOV DOT, -(SP)
MOV DOT, -(SP)
MOV R5, -(SP)
MOV R5, -(SP)
MOV #LINDOT, -(SP)
MOV #5, -(SP)
MOV SP,R0
TRAP C$PNTB
ADD #14,SP
MOV R3, -(SP)
MOV R4, -(SP)
MOV #GOOBAD, -(SP)
MOV #3, -(SP)
MOV SP,R0
TRAP C$PNTX
ADD #10,SP
MOV ABITS, -(SP)
MOV #EADD, -(SP)
MOV #2, -(SP)
MOV SP,R0
TRAP C$PNTX
ADD #6,SP

```


80	003212	104423					TRAP	C\$MSG
81	003214		BGNMSG	ER700				
	003214						ER700::	
82	003214	000305		SWAB	R5	: GET THE LINE NUMBER		
83	003216	042705	177400	BIC	#177400,R5	: CLEAR OTHER BITS		
84	003222	010137	002200	MOV	R1,a#DOT	: GET THE DOT NUMBER		
85	003226	042737	177000	BIC	#177000,a#DOT	: CLEAR THE OTHER BITS		
86	003234		002200	PRINTB	#LINDOT,R5,R5,DOT,DOT	: LINE AND DOT NUMBER		
	003234	016746	176740				MOV	DOT,-(SP)
	003240	016746	176734				MOV	DOT,-(SP)
	003244	010546					MOV	R5,-(SP)
	003246	010546					MOV	R5,-(SP)
	003250	012746	004503				MOV	#LINDOT,-(SP)
	003254	012746	000005				MOV	#5,-(SP)
	003260	010600					MOV	SP,R0
	003262	104414					TRAP	C\$PNTB
	003264	062706	000014				ADD	#14,SP
87	003270			PRINTX	#GOOBAD,R4,R3	: GOOD AND BAD		
	003270	010346					MOV	R3,-(SP)
	003272	010446					MOV	R4,-(SP)
	003274	012746	004571				MOV	#GOOBAD,-(SP)
	003300	012746	000003				MOV	#3,-(SP)
	003304	010600					MOV	SP,R0
	003306	104415					TRAP	C\$PNTX
	003310	062706	000010				ADD	#10,SP
88	003314	016767	005130	176662	MOV	NLINS,ABITS	: GET NO. OF LINES PER COLOUR	
89	003322	006367	176656		ASL	ABITS	: SHIFT TO FORM	
90	003326	006367	176652		ASL	ABITS	: MASK OF ADDRESS	
91	003332	006367	176646		ASL	ABITS	: LINES BEING TESTED	
92	003336	166767	005106	176640	SUB	NLINS,ABITS		
93	003344	042767	177400	176632	BIC	#177400,ABITS	: CLEAR THE TOP BYTE	
94	003352				PRINTX	#EADD,ABITS	: AND ADDRESS LINES BEING TESTED	
	003352	016746	176626				MOV	ABITS,-(SP)
	003356	012746	004652				MOV	#EADD,-(SP)
	003362	012746	000002				MOV	#2,-(SP)
	003366	010600					MOV	SP,R0
	003370	104415					TRAP	C\$PNTX
	003372	062706	000006				ADD	#6,SP
95	003376	004767	176654		JSR	PC,CHKMAX	: CHECK FOR TOO MANY ERRORS	
96	003402			ENDMSG				
	003402						L10012:	
	003402	104423					TRAP	C\$MSG
97								
98	003404		BGNMSG	ER800				
	003404						ER800::	
99	003404	010167	176570		MOV	R1,DOT	: GET THE DOT NUMBER	
100	003410	042767	177000	176562	BIC	#177000,DOT	: CLEAR THE OTHER BITS	
101	003416				PRINTB	#E800B,R3,DOT	: DOT ADDRESS,GOOD,BAD	
	003416	016746	176556				MOV	DOT,-(SP)
	003422	010346					MOV	R3,-(SP)
	003424	012746	004257				MOV	#E800B,-(SP)
	003430	012746	000003				MOV	#3,-(SP)
	003434	010600					MOV	SP,R0
	003436	104414					TRAP	C\$PNTB
	003440	062706	000010				ADD	#10,SP
102	003444			PRINTB	#E800C,R3,R4	:		

GLOBAL AREAS MACRO V03.01 18-NOV-81 11:10:45 PAGE 14-5
 GLOBAL ERROR REPORT SECTION

122					.NLIST	BEX
123	003636	045	101	103	E200B: .ASCIZ	/%ACSR CONTENTS %06%N/
124	003663	045	101	117	E300B: .ASCII	/%AOLD CSR CONTENTS %06/
125	003711	045	116	045	.ASCIZ	/%N%ANew CSR CONTENTS %06%A (IGNORE BITS 0,1,2 AND 4)/
126	003776	045	116	045	E300C: .ASCIZ	/%N%ADATA WRITTEN TO DBUFF %06%N/
127	004036	045	101	101	E400B: .ASCII	/%AATTEMPTED SETTING %06/
128	004065	045	116	045	.ASCIZ	/%N%AACTUAL SETTING %06%A (IGNORE BITS 0-4 AND 6-7)%N/
129	004152	045	101	101	E401B: .ASCII	/%AATTEMPTED SETTING %06/
130	004201	045	116	045	.ASCIZ	/%N%AACTUAL SETTING %06%A (IGNORE BITS 3-15)%N/
131	004257	045	101	104	E800B: .ASCIZ	/%ADOT ADDRESS WRITTEN %01%A, FAILING DOT ADDRESS %01/
132	004344	045	116	045	E800C: .ASCIZ	/%N%AEEXPECTED CONTENTS %01%A, ACTUAL CONTENTS %01%N/
133	004427	045	101	103	E900B: .ASCIZ	/%ACSR CONTENTS %06%A, EXPECTED COLOUR %01%N/
134	004503	045	101	114	LINDOT: .ASCIZ	/%ALINE %D3%A (%03%A OCTAL), DOT %D3%A (%03%A OCTAL)%N/
135	004571	045	101	105	GOOBAD: .ASCIZ	/%AEEXPECTED CONTENTS %01%A, ACTUAL CONTENTS %01%N/
136	004652	045	101	101	EADD: .ASCIZ	/%AADDRESS BITS BEING TESTED %B9%N/
137					.LIST	BEX
138					.EVEN	
139	004714				ENDMOD	


```
11          .TITLE MISCELLANEOUS SECTIONS
12          .SBTTL  REPORT CODING SECTION
40
41 004714          BGNMOD
42          .SBTTL  PROTECTION TABLE
43
44          :++
45          : THIS TABLE IS USED BY THE RUNTIME SERVICES
46          : TO PROTECT THE LOAD MEDIA.
47          :--
48
49 004714          BGNPROT
49 004714          L$PROT::
50
51 004714 000000          0          :OFFSET INTO P-TABLE FOR CSR ADDRESS
52 004716 177777          -1          :OFFSET INTO P-TABLE FOR MASSBUS ADDRESS
53 004720 177777          -1          :OFFSET INTO P-TABLE FOR DRIVE NUMBER
54
55 004722          ENDPROT
56
```

```

1      .SBTTL  INITIALIZE SECTION
2
3      :++
4      : THE INITIALIZE SECTION CONTAINS THE CODING THAT IS PERFORMED
5      : AT THE BEGINNING OF EACH PASS.
6      :--
7
8      004722      BGNINIT
9      004722
10     004722      LSINIT::
11     33 004722      READEF #EF.START      ; START COMMAND?
12     004722      012700 000040          ;
13     004726      104447          ;
14     34 004730      BNCOMPLETE CONT      ; IF NOT, BRANCH
15     004730      103002          ;
16     35 004732      004767 000114      JSR   PC,SETCLK      ; SET UP CLOCK COUNTER
17     36 004736      012700 000036      READEF #EF.CONTINUE ; CONTINUE COMMAND?
18     004742      104447          ;
19     37 004744      BCOMPLETE END        ; IF YES, NO P TABLE
20     004744      103441          ;
21     38 004746      READEF #EF.NEW       ; NEW PASS?
22     004746      012700 000035      ;
23     004752      104447          ;
24     39 004754      BNCOMPLETE NEXT      ; IF NOT, SKIP SETUP
25     004754      103003          ;
26     40 004756      012767 177777 175204 SETUP: MOV  #-1,UUT      ; INITIALISE LOGICAL UNIT NUMBER
27     41 004764      005267 175200      NEXT:  INC  UUT        ; POINT TO NEXT LOGICAL UNIT
28     42 004770      026767 175174 175014  CMP   UUT,LSUNIT    ; ALL DONE?
29     43 004776      001421          ;
30     44 005000      016700 175164      BEQ   ABORT        ; IF YES, END OF PASS
31     005004      104442          ;
32     005006      010002          ;
33     45 005010      BNCOMPLETE NEXT      ; IF NOT AVAILABLE, GET NEXT
34     005010      103365          ;
35     46 005012      011267 175154      MOV   @R2,VTVCSR    ; SAVE NEW CSR ADDRESS
36     47 005016      016767 175150 175150 MOV   VTVCSR,DBUFF ; AND DBUFF ADDRESS
37     48 005024      062767 000002 175142 ADD   #2,DBUFF     ;
38     49 005032      005067 175140      CLR   ERRCNT       ; CLEAR SUBPASS ERROR COUNT
39     50 005036      104432          ;
40     005040      000010          ;
41     51 005042      DOCLN                ; CLEAN UP AND ABORT THE PASS
42     005042      104444          ;
43     53 005044      EXIT  INIT           ; AND EXIT
44     005044      104432          ;
45     005046      000002          ;
46     65
47     66      .EVEN
48     67
49     68 005050      END:  ENDINIT
50     005050
51     005050      104411          ;
52     69
53     L10017: TRAP  CSINIT

```

```

1
2
3
4
5
6
7
8
9
10
11
12
13
14 005052 005004
15 005054 005005
16 005056
   005056 104440
   005060 010002
17 005062
   005062 012700 000114
   005066 104462
   005070 010001
18 005072 103052
19
20
21
22 005074
   005074 012746 000340
   005100 012746 005162
   005104 016146 000004
   005110 012746 000003
   005114 104437
   005116 062706 000010
23 005122 012703 000005
24 005126 026127 000006 000062
25 005134 001401
26 005136 005203
27
28 005140
   005140 012700 000000
   005144 104441
29 005146 012771 000100 000000
30
31 005154 005204
32 005156 001376
33 005160 000417
34
35 005162 005303
36 005164 100401
37 005166 000002
38
39 005170
   005170 010200
   005172 104441
40 005174 022626
41 005176 005071 000000
42

```

```

:++
: THIS ROUTINE SETS UP A DELAY VARIABLE CALLED COUNT. THIS GIVES A
: DELAY OF APPROXIMATELY 25 MILLISECONDS IF USED AS FOLLOWS:-
:
:   MOV     COUNT,R0
1$: DEC     R0
:   BNE    1$
:
: COUNT IS DERIVED FROM THE L CLOCK IF THERE IS ONE. OTHERWISE, THE
: OPERATOR IS ASKED TO TYPE 2 CHARACTERS ON THE CONSOLE 6 SECONDS
: APART.
:--

```

```

SETCLK: CLR     R4           ; CLEAR THE COUNTERS
        CLR     R5
        GETPRI  R2         ; SAVE CURRENT PRIORITY IN R2
:
:   TRAP    C$GPRI
:   MOV     R0,R2
:
:   CLOCK   L,R1          ; IS THERE AN L CLOCK?
:
:   MOV     #'L,R0
:   TRAP    C$CLCK
:   MOV     R0,R1
:
:   BCC     NOCLOCK       ; IF NOT, BRANCH
:
:   USE THE L CLOCK
LCLOCK: SETVEC 4(R1),#KLINT,#340 ; SET UP THE CLOCK VECTOR
:
:   MOV     #340,-(SP)
:   MOV     #KLINT,-(SP)
:   MOV     4(R1),-(SP)
:   MOV     #3,-(SP)
:   TRAP    C$SVEC
:   ADD     #10,SP
:
:   MOV     #5,R3          ; IF 50 HZ, 100 MS = 5 INTERRUPTS
:   CMP     6(R1),#50.    ; 50 HZ CORRECT?
:   BEQ     10$           ; IF YES, BRANCH
:   INC     R3            ; ELSE ALLOW 6 INTERRUPTS
:
10$: SETPRI #0           ; NOW DROP THE PRIORITY
:
:   MOV     #0,R0
:   TRAP    C$SPRI
:
:   MOV     #100,@(R1)    ; ALLOW INTERRUPTS
:
30$: INC     R4           ; COUNT THE DELAY
:   BNE    30$
:   BR     NOCLOCK       ; IF TOO LONG, ASSUME NO CLOCK
:
KLINT: DEC     R3         ; 5 OR 6 INTERRUPTS?
:   BMI    40$           ; IF YES, TIDY UP
:   RTI
:   ELSE KEEP COUNTING
:
40$: SETPRI  R2         ; RESTORE THE PRIORITY
:
:   MOV     R2,R0
:   TRAP    C$SPRI
:
:   CMP     (SP)+,(SP)+  ; TIDY UP THE STACK
:   CLR     @(R1)        ; DISABLE CLOCK INTERRUPTS

```


MISCELLANEOUS SECTIONS
INITIALIZE SECTION

```

43 005202 006204          ASR      R4          ; DIVIDE THE 100 MILLISECOND COUNTER
44 005204 006204          ASR      R4          ; BY 4 TO GIVE 25 MILLISECONDS
45 005206 042704 140000   BIC      #140000,R4   ; ENSURE TOP BITS ARE CLEAR
46 005212 010467 174770   MOV      R4,COUNT    ; AND SAVE THE COUNTER
47 005216 000207          RTS      PC          ; RETURN
48
49
50           ; USE THE CONSOLE FOR TIMING
51           ;
52           TKS=177560   ; KEYBOARD STATUS REGISTER
53           TKB=177562   ; KEYBOARD DATA BUFFER
54           TPS=177564   ; PRINTER STATUS REGISTER
55           TPB=177566   ; PRINTER DATA BUFFER
56
57 NOCLOCK: SETVEC #60,#TTINT,#340 ; SET UP INTERRUPT VECTOR
          MOV      #340,-(SP)
          MOV      #TTINT,-(SP)
          MOV      #60,-(SP)
          MOV      #3,-(SP)
          TRAP     C$$VEC
          ADD      #10,SP
58           PRINTF #TIMMSG          ; 'TYPE 2 CHARACTERS 6 SECONDS APART'
          MOV      #TIMMSG,-(SP)
          MOV      #1,-(SP)
          MOV      SP,R0
          TRAP     C$PNTF
          ADD      #4,SP
59
60 005266 105767 172266   10$:   TSTB     TKS          ; IS FIRST CHARACTER READY?
61 005272 100375          BPL      10$          ; IF NOT, WAIT
62
63 005274 016767 172262 172264   MOV      TKB,TPB    ; NOW ECHO THE CHARACTER
64 005302          SETPRI   #0          ; DROP THE PRIORITY
          MOV      #0,R0
          TRAP     C$$PRI
65 005310 012767 000100 172242   MOV      #100,TKS   ; ALLOW INTERRUPTS
66
67 005316 105205          20$:   INCB     R5          ; START COUNTING
68 005320 001376          BNE     20$          ;
69 005322 005204          INC      R4          ; UPDATE THE COUNTER
70 005324 000774          BR      20$          ;
71
72 005326          TTINT: SETPRI   R2          ; RESTORE THE PRIORITY
          MOV      R2,R0
          TRAP     C$$PRI
73 005332 016767 172224 172226   MOV      TKB,TPB    ; ECHO THE CHARACTER
74 005340 022626          CMP     (SP)+,(SP)+ ; TIDY UP THE STACK
75 005342 005067 172212          CLR     TKS          ; DISABLE INTERRUPTS
76
77 005346 010467 174634          MOV      R4,COUNT    ; SAVE THE COUNTER
78                                     ; 6 SECONDS/256 = 25 MILLISECONDS
79 005352 000207          RTS      PC          ; RETURN
80
81
82 005354      045      116      045 TIMMSG: .NLIST BEX
83                                     .ASCIZ /%N%ATYPE 2 CHARACTERS 6 SECONDS APART >/
84                                     .LIST BEX
85                                     .EVEN

```

.SBTTL AUTODROP SECTION

```

:++
: THIS CODE IS EXECUTED IMMEDIATELY AFTER THE INITIALIZE CODE IF
: THE "ADR" FLAG WAS SET. THE UNIT(S) UNDER TEST ARE CHECKED TO
: SEE IF THEY WILL RESPOND. THOSE THAT DON'T ARE IMMEDIATELY
: DROPPED FROM TESTING.
:--
    
```

```

1
2
3
4
5
6
7
8
9
10 005424          BGNAUTO
    005424
11
18
19 005424          SETVEC #4,#NXMTRP,#PRI07 ; SET UP NXM TRAP VECTOR
    005424 012746 000340
    005430 012746 002430
    005434 012746 000004
    005440 012746 000003
    005444 104437
    005446 062706 000010
20 005452 005037 002202
21 005456 005777 174510
22 005462 005737 002202
23 005466 001404
24 005470
    005470 016700 174474
    005474 104451
25 005476
    005476 104444
26 005500          10$: CLRVEC #4
    005500 012700 000004
    005504 104436
27
28 005506          ENDAUTO
    005506
    005506 104461
    
```

LSAUTO::

```

MOV #PRI07,-(SP)
MOV #NXMTRP,-(SP)
MOV #4,-(SP)
MOV #3,-(SP)
TRAP C$SVEC
ADD #10,SP
    
```

```

CLR @#NXMFLG ; INITIALISE NXM FLAG
TST @VTVCSR ; TEST THE CSR ADDRESS
TST @#NXMFLG ; WAS THERE A TRAP?
BEQ 10$ ; IF NOT, BRANCH
DODU UUT ; ELSE DROP THE UNIT
    
```

```

MOV UUT,R0
TRAP C$DODU
TRAP C$DCLN
MOV #4,R0
TRAP C$CVEC
    
```

```

; AND TIDY UP END OF PASS
; RESTORE THE DRS NXM TRAP
    
```

```

L10020: TRAP C$AUTO
    
```

.SBTTL CLEANUP CODING SECTION

```

:++
: THE CLEANUP CODING SECTION CONTAINS THE CODING THAT IS PERFORMED
: AFTER THE HARDWARE TESTS HAVE BEEN PERFORMED OR IF CONTROL/C IS
: TYPED DURING TESTING.
: A PRESET IS PERFORMED TO SET THE SCREEN TO ALL BLACK, THE OFFSET
: REGISTER IS CLEARED, AND THE ERROR COUNT ZEROED.
:--

```

```

1
2
3
4
5
6
7
8
9
10
11 005510          BGNCLN
12 005510
13
14
15
16
17
18
19
20
21 005510 005737 002202          TST    @#NXMFLG      : HAS NXM ERROR OCCURRED?
22 005514 001015          BNE    2$           : IF YES, SKIP CLEAN UP
23 005516 005077 174452          CLR    @DBUFF       : SET DBUFF COLOUR TO BLACK
24 005522 012777 000150 174442  MOV    #150,@VTVCSR : RESET OFFSET AND DO PRESET
25
26 005530 012701 177777          MOV    #-1,R1      : SET UP WAIT COUNTER
27 005534 105777 174432          TSTB  @VTVCSR     : PRESET FINISHED?
28 005540 100403          BMI    2$         : IF YES, BRANCH
29 005542          BREAK          : ALLOW OPERATOR BREAK IN
30 005544 005301          DEC    R1          : WAITED TOO LONG?
31 005546 001372          BNE    1$         : IF NOT, WAIT LONGER
32 005550 005067 174422          CLR    ERRCNT     : CLEAR SUBPASS ERROR COUNT
33
34 005554          EXIT    CLN
35 005554 104432          TRAP  C$BRK
36 005556 000002          .WORD  L10021-.
37
38
39
40
41
42
43
44
45
46
47
48          .EVEN
49
50 005560          ENDCLN
51 005560
52 005560 104412          L10021: TRAP  C$CLEAN

```



```
1  
2  
3  
4  
5  
6  
7  
8 005562          BGNDU  
   005562  
9  
18 005562          PRINTF #DROPD,RO      ; 'UNIT DROPPED'  
   005562 010046  
   005564 012746 005610  
   005570 012746 000002  
   005574 010600  
   005576 104417  
   005600 062706 000006  
19  
20 005604          EXIT  DU  
   005604 000167  
   005606 000026  
21  
23  
33 005610          045      116      045  DROPD: .NLIST BEX  
34                                     .ASCIZ  /%N%AUNIT%D2%A DROPPED/  
35                                     .LIST  BEX  
36  
37                                     .EVEN  
38  
39 005636          ENDDU  
   005636  
   005636 104453  
40 005640          ENDMOD  
41
```

LSDU::
MOV R0,-(SP)
MOV #DROPD,-(SP)
MOV #2,-(SP)
MOV SP,R0
TRAP C\$PNTF
ADD #6,SP

.WORD JSJMP
.WORD L10022-2-

L10022: TRAP C\$DU

```

2         .TITLE HARDWARE TESTS
10
35
36 005640         BGNMOD
37
38         .SBTTL TEST 1: REGISTER NXM
39
40         :++
41         : CHECKS ARE MADE THAT ACCESSING THE DEVICE CSR AND DBUFF REGISTERS
42         : DOES NOT CAUSE AN NXM TRAP.
43         :--
44
45 005640         BGNTST
46 005640                                         T1::
47 005640         SETVEC #4,#NXMTRP,#PRI07 ; SET UP NXM TRAP VECTOR
48 005640 012746 000340         MOV #PRI07,-(SP)
49 005644 012746 002430         MOV #NXMTRP,-(SP)
50 005650 012746 000004         MOV #4,-(SP)
51 005654 012746 000003         MOV #3,-(SP)
52 005660 104437         TRAP C$SVEC
53 005662 062706 000010         ADD #10,SP
54 005666 005037 002202         CLR @#NXMFLG ; INITIALISE NXM ERROR FLAG
55         :
56         : CHECK THE CSR
57         :
58         :***** BEGIN SUBTEST
59         :***** T1.1:
60 005672 104402         CLR R2 ; INITIALISE CSR NXM ERROR FLAG
61 005674 005002         TST @VTVCSR ; TEST THE ADDRESS
62 005676 005777 174270     TST R2 ; WAS THERE A TRAP?
63 005702 005702         BEQ 10$ ; IF NOT, BRANCH
64 005704 001404         ERRDF 100,E100 ; ELSE REPORT THE ERROR
65 005706 104455         TRAP C$ERDF
66 005710 000144         .WORD 100
67 005712 006006         .WORD E100
68 005714 000000         .WORD 0
69 005716 104403     10$: ENDSUB ;***** END SUBTEST
70 005716 104403         L10024: TRAP C$ESUB
71 005720 005720         BGNSUB ;***** BEGIN SUBTEST
72 005720 104402         T1.2: TRAP C$BSUB
73 005722 005002         CLR R2 ; INITIALISE DBUFF NXM ERROR FLAG
74 005724 005777 174244     TST @DBUFF ; TEST THE ADDRESS
75 005730 005702         TST R2 ; WAS THERE A TRAP?
76 005732 001404         BEQ 20$ ; IF NOT, BRANCH
77 005734 001404         ERRDF 101,E101 ; ELSE REPORT THE ERROR
78 005734 104455         TRAP C$ERDF
79 005736 000145         .WORD 101
80 005740 006054         .WORD E101
81 005742 000000         .WORD 0
82 005744 005744     20$: ENDSUB ;***** END SUBTEST
83 005744 104403         L10025: TRAP C$ESUB
84
85
86
```



```

1          .SBTTL TEST 2: REGISTER ACCESS
2
3          :++
4          : THE DEVICE CSR IS READ AND CHECKS ARE MADE THAT THE READY BIT IS
5          : SET, THAT BITS 3 AND 6 ARE ZERO, AND THAT THE FIELD BIT (4) IS
6          : SET AND CLEARED BY THE DEVICE.
7          :--
8
9
10         006140          BGNTST
11         006140
12         006140          BGNSUB          ;***** BEGIN SUBTEST
13         006140 104402          ;***** T2.1:
14         006142 017701 174024          TRAP          CSBSUB
15         006146 105701          MOV          @VTVCSR,R1          ; READ THE CSR
16         006150 100404          TSTB          R1          ; READY BIT SET?
17         006152          BMI          1$          ; IF YES, BRANCH
18         006152 104456          ERRHRD 200,E200,ER200 ; ELSE REPORT THE ERROR
19         006154 000310          ; TRAP          CSERHRD
20         006156 006266          ; .WORD          200
21         006160 002444          ; .WORD          E200
22         006162 032701 000110          ; .WORD          ER200
23         006166 001404          1$:          BIT          #110,R1          ; CHECK BITS 3 AND 6
24         006170          BEQ          2$          ; IF CLEAR, JUMP
25         006170 104456          ERRHRD 201,E201,ER200 ; ELSE REPORT THE ERROR
26         006172 000311          ; TRAP          CSERHRD
27         006174 006314          ; .WORD          201
28         006176 002444          ; .WORD          E201
29         006200 005002          ; .WORD          ER200
30         006202 017701 173764          2$:          CLR          R2          ; ZERO WAIT COUNTER
31         006206 032701 000020          3$:          MOV          @VTVCSR,R1          ; READ THE CSR
32         006212 001006          BIT          #20,R1          ; FIELD BIT SET?
33         006214 005302          BNE          4$          ; IF YES, BRANCH
34         006216 001371          DEC          R2          ; ELSE WAITED TOO LONG?
35         006220          BNE          3$          ; IF NOT, KEEP WAITING
36         006220 104456          ERRHRD 202,E202,ER200 ; ELSE REPORT THE ERROR
37         006222 000312          ; TRAP          CSERHRD
38         006224 006346          ; .WORD          202
39         006226 002444          ; .WORD          E202
40         006230 005002          ; .WORD          ER200
41         006232 017701 173734          4$:          CLR          R2          ; ZERO WAIT COUNTER
42         006236 032701 000020          5$:          MOV          @VTVCSR,R1          ; READ THE CSR
43         006242 001406          BIT          #20,R1          ; FIELD BIT CLEAR?
44         006244 005302          BEQ          6$          ; IF YES, BRANCH
45         006246 001571          DEC          R2          ; ELSE WAITED TOO LONG?
46         006250          BNE          5$          ; IF NOT, KEEP WAITING
47         006250 104456          ERRHRD 203,E203,ER200 ; ELSE REPORT THE ERROR
48         006252 000313          ; TRAP          CSERHRD
49         006254 006405          ; .WORD          203
50         006256 002444          ; .WORD          E203
51         006260          ; .WORD          ER200
52         006260          6$:          ENDSUB          ;***** END SUBTEST
    
```

006260
006260 104403
39 006262
006262 104432
006264 000164

EXIT TST

L10027: TRAP C\$ESUB
TRAP C\$EXIT
.WORD L10026-

40
41
42 006266 103 123 122 E200:
43 006314 103 123 122 E201:
44 006346 103 123 122 E202:
45 006405 103 123 122 E203:

.NLIST BEX
.ASCIZ /CSR READY BIT NOT SET/
.ASCIZ /CSR BITS 3 OR 6 NOT CLEAR/
.ASCIZ /CSR FIELD BIT DOES NOT GET SET/
.ASCIZ /CSR FIELD BIT DOES NOT GET CLEARED/
.LIST BEX
.EVEN
ENDTST

46
47
48 006450
006450
006450 104401
49

L10026: TRAP C\$ETST

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39

.SBTTL TEST 3: REGISTER ADDRESS UNIQUENESS

```

:++
: ALL BITS IN THE DBUFF REGISTER ARE SET AND CLEARED IN ALL
: COMBINATIONS WITH THE EXCEPTION OF BIT 14 (READ PICTURE STORE),
: WHICH IS LEFT CLEAR. FOR EACH BIT COMBINATION, THE CSR IS READ
: AND CHECKED TO BE UNCHANGED. BITS 0,1,2 AND 4 ARE IGNORED IN THIS
: CHECK.
:--
  
```

```

12 006452          BGNTST
13 006452 017701 173514          MOV @VTVCSR,R1      ; SAVE CSR CONTENTS      T3::
14 006456 042701 000027          BIC #27,R1         ; CLEAR BIT 4
15 006462 005002          CLR R2              ; INITIALISE DBUFF TEST DATA
17 006464 010277 173504          1$: MOV R2,@DBUFF     ; WRITE DBUFF TEST DATA
18 006470 017703 173476          MOV @VTVCSR,R3     ; READ CSR
19 006474 042703 000027          BIC #27,R3         ; CLEAR BIT 4
20 006500 020301          CMP R3,R1          ; CSR UNCHANGED?
21 006502 001405          BEQ 2$            ; IF YES, NEXT PATTERN
22 006504          ERRHRD 300,E300,ER300 ; ELSE REPORT THE ERROR
                                TRAP C$ERHRD
                                .WORD 300
                                .WORD E300
                                .WORD ER300
23 006514 010301          MOV R3,R1          ; SAVE NEW CSR CONTENTS
25 006516 005202 040000          2$: INC R2          ; NEXT DATA PATTERN
26 006520 020227          CMP R2,#40000     ; BIT 14 SET?
27 006524 001002          BNE 3$            ; IF NOT, JUMP
28 006526 006302          ASL R2            ; SKIP SOME PATTERNS
29 006530 000755          BR 1$             ; NEXT PATTERN
31 006532 020227 140000          3$: CMP R2,#140000 ; ALL DONE?
32 006536 001352          BNE 1$            ; IF NOT, NEXT PATTERN
33 006540          EXIT TST
                                TRAP C$EXIT
                                .WORD L10030-.
36 006544 103 123 122 E300: .NLIST BEX
                                .ASCIZ /CSR CHANGED BY WRITING TO DBUFF/
                                .LIST BEX
                                .EVEN
39 006604          ENDTST
                                L10030: TRAP C$ETST
                                104401
  
```



```

1      .SBTTL TEST 4: REGISTER BIT SET
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18 006606          BGNTST
19 006606          BGNSUB          ;***** BEGIN SUBTEST
20 006606          ;***** T4::
21 006606 104402   ;***** T4.1: TRAP C$BSUB
22
23 006610 005001   ; DIRECT BIT SET
24
25 006612          T4A:  BGNSSEG   ;----- BEGIN SEGMENT
26 006612 104404   ; TRAP C$BSEG
27 006614 010177 173352  MOV R1,@VTVCSR ; WRITE TEST PATTERN TO CSR
28 006620 017703 173346  MOV @VTVCSR,R3 ; READ THE CSR
29 006624 010304          MOV R3,R4      ; SAVE THE CONTENTS
30 006626 042703 000337  BIC #337,R3   ; CLEAR UNWANTED BITS
31 006632 020301          CMP R3,R1      ; CORRECT BITS WRITTEN?
32 006634 001404          BEQ 1$          ; IF YES, BRANCH
33 006636          ERRHRD 400,E400,ER400 ; ELSE REPORT THE ERROR
34 006636 104456   ; TRAP C$ERHRD
35 006640 000620   ; .WORD 400
36 006642 006774   ; .WORD E400
37 006644 002550   ; .WORD ER400
38 006646          1$:  ENDSEGE   ;----- END SEGMENT
39 006646          ;***** 10000$:
40 006646 104405   ; TRAP C$ESEG
41
42 006650 062701 000400  ADD #400,R1   ; NEXT PATTERN
43 006654 103356          BCC T4A        ; IF MORE, BRANCH
44 006656 020127 000040  CMP R1,#40   ; ELSE SECOND TIME THROUGH?
45 006662 001403          BEQ 2$          ; IF YES, BRANCH
46 006664 012701 000040  MOV #40,R1   ; ELSE SET BIT 5
47 006670 000750          BR T4A          ; AND REPEAT PATTERNS
48 006672          2$:  ENDSUB   ;***** END SUBTEST
49 006672          ;***** L10032:
50 006672 104403   ; TRAP C$ESUB
51
52 006674          BGNSUB          ;***** BEGIN SUBTEST
53 006674          ;***** T4.2:
54 006674 104402   ; TRAP C$BSUB
    
```

```

44
45      : BIT SET VIA DBUFF AND PICTURE STORE
46      :
47 006676 005077 173270      CLR    @VTVCSR      : CLEAR THE CSR
48 006702 005001              CLR    R1              : INITIALISE TEST DBUFF PATTERN
49 006704 005003              CLR    R3              : INITIALISE EXPECTED CSR PATTERN
50
51 006706      104404      1$:  BGNSEG      :----- BEGIN SEGMENT
52 006710 010177 173260      MOV    R1,@DBUFF      : WRITE DBUFF PATTERN
53 006714 012777 040000 173252  MOV    #40000,@DBUFF : LATCH PATTERN INTO THE CSR
54 006722 017704 173244      MOV    @VTVCSR,R4     : READ THE CSR
55 006726 010405              MOV    R4,R5         : SAVE THE CONTENTS
56 006730 042704 177770      BIC    #177770,R4    : CLEAR UNWANTED BITS
57 006734 020403              CMP    R4,R3         : EXPECTED PATTERN?
58 006736 001404              BEQ    2$,              : IF YES, BRANCH
59 006740              ERRHRD 401,E401,ER401 : ELSE REPORT THE ERROR
60 006740 104456              TRAP   C$ERHRD
61 006742 000621              .WORD 401
62 006744 007067              .WORD E401
63 006746 002602              .WORD ER401
64 006750      104405      2$:  ENDSEG      :----- END SEGMENT
65 006750              10000$: TRAP   C$ESEG
66 006752 005203              INC    R3            : NEXT EXPECTED CSR PATTERN
67 006754 062701 002000      ADD    #2000,R1      : NEXT DBUFF WRITE PATTERN
68 006760 020127 020000      CMP    R1,#20000    : ALL PATTERNS DONE?
69 006764 001350              BNE    1$,          : IF NOT, DO NEXT
70
71 006766      104432      EXIT TST      : ELSE END OF TEST
72 006766 104432              TRAP   C$EXIT
73 006770 000206              .WORD L10031-
74 006772      104403      ENDSUB      :***** END SUBTEST
75 006772              L10033: TRAP   C$ESUB
76
77 006774      103      117      115  E400: .NLIST BEX
78 007057      102      105      040  .ASCII /COMBINATION OF CSR BIT 5 AND BITS 8 - 15 COULD NOT /
79 007067      103      117      115  E401: .ASCIZ /BE SET./
80 007152      124      110      105  .ASCII /COMBINATION OF CSR BITS 0 - 2 COULD NOT BE SET VIA /
81              110      105  .ASCIZ /THE DBUFF REGISTER./
82              .LIST BEX
83              .EVEN
84
85 007176              ENDTST
86 007176              L10031: TRAP   C$ETST
87 007176 104401

```

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

.SBTTL TEST 5: PICTURE STORE DATA

```

:++
: A ZERO DATA PATTERN (COLOUR BLACK) IS WRITTEN TO THE FIRST PICTURE
: STORE LOCATION AND CHECKED TO READ BACK CORRECTLY IN THE CSR. THIS
: IS REPEATED FOR ALL OF THE PICTURE STORE LOCATIONS TO FORM AN ALL
: BLACK BACKGROUND.

```

```

: DATA IN ALL 8 COLOUR COMBINATIONS IS THEN WRITTEN TO EACH PICTURE
: STORE LOCATION IN TURN AND AFTER EACH WRITE OPERATION, THE CSR IS
: EXAMINED TO ENSURE THAT THE DATA READS BACK CORRECTLY. EACH
: LOCATION IS RESET TO THE BACKGROUND COLOUR AFTER THE ABOVE
: OPERATION.

```

```

: THE ENTIRE TEST IS THEN REPEATED USING WHITE (DATA 7) AS THE
: BACKGROUND COLOUR.

```

--

BGNTST

T5::

```

MOV #50,@VTVCSR ; SET VIDEO ENABLE AND ZERO OFFSET
CLR BKGND ; SET BACKGROUND TO BLACK
CLR R4 ; AND EXPECTED BACKGROUND

: SET PICTURE STORE TO BACKGROUND COLOUR
T5A: MOV BKGND,R1 ; INITIALISE DOT NO. AND COLOUR
MOV #40000,R5 ; READ PICTURE STORE MASK

1$: BGNSEG ;----- BEGIN SEGMENT
TRAP CSBSEG

MOV R1,@DBUFF ; WRITE BACKGROUND TO DBUFF
MOV R5,@DBUFF ; LATCH COLOUR INTO CSR
MOV @VTVCSR,R2 ; READ THE CSR
MOV R2,R3 ; SAVE THE CONTENTS
BIC #177770,R2 ; GET THE COLOUR
CMP R2,R4 ; BACKGROUND COLOUR?
BEQ 2$ ; IF YES, BRANCH
ERRHRD 500,E500,ER500 ; ELSE REPORT THE ERROR

TRAP CSERHRD
WORD 500
WORD E500
WORD ER500

2$: ENDSEG ;----- END SEGMENT
10000$: TRAP CSESEG

INC R5 ; NEXT DOT TO READ
INC R1 ; AND TO WRITE
BIT #777,R1 ; ALL DOTS DONE?
BNE 1$ ; IF NOT, DO NEXT

ADD #400,@VTVCSR ; NEXT LINE
BCC T5A ; IF MORE LINES, GO BACK

: SET ALL OTHER BIT COMBINATIONS

```

H
T


```

50
51 007310 012777 000040 172654 ; MOV #40,@VTVCSR ; SET VIDEO ENABLE ONLY
52 007316 005002 CLR R2 ; FIRST EXPECTED COLOUR
53
54 007320 005001 3$: CLR R1 ; INITIALISE DOT NO. AND DATA
55 007322 012705 040000 MOV #40000,R5 ; READ PICTURE STORE MASK
56
57 007326 4$: BGNSEG ;----- BEGIN SEGMENT
007326 104404 TRAP C$BSEG
58
59 007330 010177 172640 MOV R1,@DBUFF ; WRITE COLOUR TO DBUFF
60 007334 010577 172634 MOV R5,@DBUFF ; LATCH COLOUR INTO CSR
61 007340 017703 172626 MOV @VTVCSR,R3 ; READ THE CSR
62 007344 010304 MOV R3,R4 ; SAVE THE CONTENTS
63 007346 042703 177770 BIC #177770,R3 ; GET THE COLOUR
64 007352 020302 CMP R3,R2 ; EXPECTED VALUE?
65 007354 001404 BEQ 5$ ; IF YES, BRANCH
66 007356 ERRHRD 501,E501,ER501 ; ELSE REPORT THE ERROR
007356 104456 TRAP C$ERHRD
007360 000765 .WORD 501
007362 007552 .WORD E501
007364 002724 .WORD ER501
67 007366 5$: ENDSEG ;----- END SEGMENT
007366 104405 10001$: TRAP C$ESEG
68
69 007370 005202 INC R2 ; NEXT EXPECTED COLOUR
70 007372 062701 002000 ADD #2000,R1 ; NEXT DBUFF WRITE COLOUR
71 007376 032701 020000 BIT #20000,R1 ; ALL COLOURS DONE?
72 007402 001751 BEQ 4$ ; IF NOT, DO NEXT
73
74 007404 005002 CLR R2 ; ELSE RESET EXPECTED COLOUR
75 007406 042701 020000 BIC #20000,R1 ; CLEAR COLOUR OVERFLOW BIT
76 007412 056701 000062 BIS BKGND,R1 ; SET BACKGROUND COLOUR
77 007416 010177 172552 MOV R1,@DBUFF ; WRITE BACKGROUND COLOUR
78 007422 042701 016000 BIC #16000,R1 ; CLEAR COLOUR BITS
79
80 007426 005205 INC R5 ; NEXT READ DOT POSITION
81 007430 005201 INC R1 ; AND WRITE DOT POSITION
82 007432 032701 001000 BIT #1000,R1 ; ALL DOTS DONE?
83 007436 001733 BEQ 4$ ; IF NOT, DO NEXT
84
85 007440 062777 000400 172524 ADD #400,@VTVCSR ; ELSE NEXT LINE
86 007446 103324 BCC 3$ ; IF MORE, GO BACK
87
88 007450 005767 000024 TST BKGND ; WHITE BACKGROUND DONE?
89 007454 001007 BNE 6$ ; IF YES, BRANCH
90 007456 012767 016000 000014 MOV #16000,BKGND ; ELSE FLAG WHITE BACKGROUND
91 007464 012704 000007 MOV #7,R4 ; AND CSR COLOUR
92 007470 000167 177520 JMP T5A ; AND REPEAT THE TEST
93
94 007474 6$: EXIT TST ; END THE TEST
007474 104432 TRAP C$EXIT
007476 000132 .WORD L10034-.
95
96 007500 000000 BKGND: .WORD 0 ; BACKGROUND COLOUR TO WRITE
97

```

98						.NLIST	BEX	
99	007502	102	101	103	E500:	.ASCIZ	/BACKGROUND COLOUR READ BACK INCORRECTLY/	
100	007552	104	101	124	E501:	.ASCIZ	/DATA READ BACK INCORRECTLY FROM PICTURE STORE/	
101						.LIST	BEX	
102						.EVEN		
103	007630					ENDTST		
	007630							L10034:
	007630	104401						TRAP C\$ETST

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56

.SBTTL TEST 6: PICTURE STORE COLUMN ADDRESSING

```

:++
: THIS VERIFIES THAT WRITING TO ANY COLUMN OF THE PICTURE STORE
: DOES NOT AFFECT THE CONTENTS OF ANY OTHER COLUMN. SINCE EACH
: LOCATION CAN HOLD A 3 BIT DATA VALUE, THE COLUMN ADDRESS LINES ARE
: CHECKED IN GROUPS OF 3.

: COLUMN ADDRESS BITS 0 - 2 ARE CHECKED BY WRITING THE VALUES 0 - 7 IN
: CONSECUTIVE LOCATIONS THROUGHOUT THE PICTURE STORE AND CHECKING THAT
: THE DATA READS BACK CORRECTLY VIA THE CSR.

: COLUMN ADDRESS BITS 3 - 5 ARE CHECKED BY WRITING ZERO TO THE FIRST
: 8 LOCATIONS, 1 TO THE NEXT 8 LOCATIONS ETC. UNTIL THE PICTURE STORE
: IS FULL. THE DATA IS THEN VERIFIED VIA THE CSR.

: COLUMN ADDRESS BITS 6 - 8 ARE CHECKED BY WRITING ZERO TO THE FIRST
: 64 LOCATIONS, ONE TO THE NEXT 64 LOCATIONS AND SO ON, UNTIL THE
: PICTURE STORE IS FULL. THE DATA IS THEN VERIFIED VIA THE CSR.
    
```

--

BGNTST

T6::

```

007632      BGNTST
007632
24 007632 012767 000001 000252      MOV      #1,NCOLS      : INITIALISE 1 COLUMN PER COLOUR
25
26
27
28
29 007640 005002      T6A:    CLR      R2      : DOT COUNTER FOR CURRENT COLOUR
30 007642 005001      CLR      R1      : INITIALISE DOT NO. AND COLOUR
31 007644 012777 000050 172320      MOV      #50,@VTVCSR  : INITIALISE LINE NUMBER
32
33 007652 010177 172316      1$:    MOV      R1,@DBUFF  : WRITE COLOUR
34 007656 005202      INC      R2      : INCREMENT DOT COUNTER
35 007660 020267 000226      CMP      R2,NCOLS  : READY FOR NEXT COLOUR?
36 007664 103410      BLO     2$      : IF NOT, BRANCH
37 007666 005002      CLR      R2      : ELSE CLEAR DOT COUNTER
38 007670 062701 002000      ADD     #2000,R1   : NEXT COLOUR
39 007674 032701 020000      BIT     #20000,R1  : ALL COLOURS USED?
40 007700 001402      BEQ     2$      : IF NOT, BRANCH
41 007702 042701 020000      BIC     #20000,R1  : ELSE CLEAR COLOUR BITS
42
43 007706 005201      2$:    INC      R1      : NEXT DOT POSITION
44 007710 032701 000777      BIT     #777,R1   : ALL DOTS DONE?
45 007714 001356      BNE     1$      : IF NOT, NEXT DOT
46 007716 162701 001000      SUB     #1000,R1  : ELSE CLEAR DOT NUMBER
47
48 007722 062777 000400 172242      ADD     #400,@VTVCSR : NEXT LINE
49 007730 103350      BCC     1$      : IF MORE LINES, BRANCH
50
51
52
53
54
55
56 007732 005002      : DOT COUNTER FOR CURRENT COLOUR
57 007734 012701 040000      MOV     #40000,R1  : DOT 0, READ PICTURE STORE
58 007740 005004      CLR     R4      : INITIALISE EXPECTED COLOUR
59 007742 012777 000040 172222      MOV     #40,@VTVCSR : INITIALISE LINE NUMBER
    
```



```

57
58 007750          T6B:  BGNSEG          ;----- BEGIN SEGMENT
    007750 104404          ;                               TRAP      C$BSEG
59 007752 010177 172216    MOV      R1,@DBUFF      ; LATCH COLOUR INTO CSR
60 007756 017703 172210    MOV      @VTVCSR,R3    ; READ THE CSR
61 007762 010305          MOV      R3,R5        ; SAVE THE CONTENTS
62 007764 042703 177770    BIC      #177770,R3    ; GET THE COLOUR
63 007770 020304          CMP      R3,R4        ; AS EXPECTED?
64 007772 001404          BEQ      1$           ; IF YES, BRANCH
65 007774          ERRHRD 600,E600,ER600 ; ELSE REPORT THE ERROR
    007774 104456          ;                               TRAP      C$ERHRD
    007776 001130          ;                               .WORD    600
    010000 010114          ;                               .WORD    E600
    010002 003032          ;                               .WORD    ER600
66 010004          1$:  ENDSEG          ;----- END SEGMENT
    010004          ;                               10000$: TRAP      C$ESEG
    010004 104405          ;
67
68 010006 005202          INC      R2           ; INCREMENT DOT COUNTER
69 010010 020267 000076    CMP      R2,NCOLS     ; READY FOR NEXT COLOUR?
70 010014 103406          BLO     2$           ; IF NOT, BRANCH
71 010016 005002          CLR     R2           ; ELSE CLEAR DOT COUNTER
72 010020 005204          INC     R4           ; EXPECT NEXT COLOUR
73 010022 032704 000010    BIT     #10,R4        ; ALL COLOURS USED?
74 010026 001401          BEQ     2$           ; IF NOT, BRANCH
75 010030 005004          CLR     R4           ; ELSE NEXT COLOUR
76
77 010032 005201          2$:  INC     R1           ; NEXT DOT PATTERN
78 010034 032701 000777    BIT     #777,R1       ; ALL DOTS DONE?
79 010040 001343          BNE     T6B          ; IF NOT, NEXT DOT
80 010042 162701 001000    SUB     #1000,R1      ; ELSE CLEAR DOT NUMBER
81
82 010046 062777 000400 172116 ADD     #400,@VTVCSR   ; NEXT LINE
83 010054 103335          BCC     T6B          ; IF MORE LINES, BRANCH
84
85 010056 006367 000030    ASL     NCOLS         ; FLAG NEXT 3
86 010062 006367 000024    ASL     NCOLS         ; ADDRESS LINES
87 010066 006367 000020    ASL     NCOLS         ; TO BE TESTED
88 010072 042767 177400 000012 BIC     #177400,NCOLS  ; STOP IF TOP BYTE USED
89 010100 001402          BEQ     3$           ; IF FINISHED, BRANCH
90 010102 000167 177532    JMP     T6A          ; ELSE NEXT ADDRESS LINES
91
92 010106          3$:  EXIT    TST          ; END OF TEST
    010106 104432          ;                               TRAP      C$EXIT
    010110 000052          ;                               .WORD    L10035-.
93
94 010112 000000          NCOLS: .WORD    0          ; NO. OF COLUMNS OF CURRENT COLOUR
95
96 010114 120 111 103 E600: .NLIST  BEX          ;
    .ASCIZ /PICTURE STORE COLUMN ADDRESSING ERROR/
97
98 .LIST  BEX
    .EVEN
99 010162          .ENDTST
    010162          ;                               L10035: TRAP      C$ETST
    010162 104401          ;
  
```

.SBTTL TEST 7: PICTURE STORE LINE ADDRESSING

..++
 : THIS VERIFIES THAT WRITING TO ANY LINE OF THE PICTURE STORE
 : DOES NOT AFFECT THE CONTENTS OF ANY OTHER LINE. SINCE EACH
 : LOCATION CAN HOLD A 3 BIT DATA VALUE, THE LINE ADDRESS LINES ARE
 : CHECKED IN GROUPS OF THREE.

..: LINE ADDRESS BITS 0 - 2 (CSR BITS 8 - 10) ARE CHECKED BY WRITING ZERO
 : TO ALL LOCATIONS IN THE FIRST LINE, ONE TO ALL LOCATIONS IN THE
 : NEXT LINE AND SO ON, UNTIL ALL LINES ARE WRITTEN TO. THE DATA IS THEN
 : READ BACK AND VERIFIED VIA THE CSR.

..: LINE ADDRESS BITS 3 - 5 (CSR BITS 11 - 13) ARE CHECKED BY WRITING
 : ZERO TO THE FIRST EIGHT LINES, ONE TO THE NEXT EIGHT LINES ETC.,
 : TO THE END OF THE PICTURE STORE. THE DATA IS THEN VERIFIED VIA THE
 : CSR.

..: LINE ADDRESS BITS 6 AND 7 (CSR BITS 14 AND 15) ARE CHECKED BY WRITING
 : ZERO TO THE FIRST 64 LINES, ONE TO THE NEXT 64 LINES, TWO TO THE NEXT
 : 64 LINES, AND THREE TO THE LAST 64 LINES. THE PICTURE STORE DATA IS
 : THEN READ BACK AND VERIFIED.

..--

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56

```

010164
010164
010164 012767 000001 000256
010172 005002
010174 005001
010176 012777 000050 171766
010204 010177 171764
010210 005201
010212 032701 000777
010216 001372
010220 162701 001000
010224 062777 000400 171740
010232 103415
010234 005202
010236 020267 000206
010242 103760
010244 005002
010246 062701 002000
010252 032701 020000
010256 001752
010260 042701 020000
010264 000747
010266 005002
010270 012701 040000
    
```

```

          BGNTST
                                     T7::
MOV      #1,NLINS      ; INITIALISE ONE LINE PER COLOUR
: WRITE TO THE PICTURE STORE
17A:    CLR      R2      ; LINE COUNTER FOR CURRENT COLOUR
        CLR      R1      ; INITIALISE DOT NO. AND COLOUR
        MOV      #50,@VTVCSR ; INITIALISE LINE NUMBER
1$:     MOV      R1,@DBUFF ; WRITE COLOUR
        INC      R1      ; NEXT DOT POSITION
        BIT      #777,R1  ; ALL DOTS DONE?
        BNE     1$      ; IF NOT, NEXT DOT
        SUB     #1000,R1  ; ELSE CLEAR DOT NUMBER
        ADD     #400,@VTVCSR ; NEXT LINE
        BCS     2$      ; IF ALL LINES DONE, BRANCH
        INC     R2      ; ELSE INCREMENT LINE COUNTER
        CMP     R2,NLINS ; READY FOR NEXT COLOUR?
        BLO     1$      ; IF NOT, DO NEXT LINE
        CLR     R2      ; ELSE CLEAR LINE COUNTER
        ADD     #2000,R1  ; NEXT COLOUR
        BIT     #20000,R1 ; ALL COLOURS USED?
        BEQ     1$      ; IF NOT, BRANCH
        BIC     #20000,R1 ; ELSE CLEAR COLOUR BITS
        BR      1$      ; AND DO NEXT LINE
: READ BACK THE DATA
2$:     CLR      R2      ; LINE COUNTER FOR CURRENT COLOUR
        MOV     #40000,R1 ; DOT 0, READ PICTURE STORE
    
```



```

57 010274 005004          CLR      R4          ; INITIALISE EXPECTED COLOUR
58 010276 012777 000040 171666  MOV     #40,@VTVCSR ; INITIALISE LINE NUMBER
59
60 010304          T7B:   BGNSEG          ;----- BEGIN SEGMENT
    010304 104404          TRAP          CSBSEG
61 010306 010177 171662    MOV     R1,@DBUFF    ; LATCH COLOUR INTO CSR
62 010312 017703 171654    MOV     @VTVCSR,R3  ; READ THE CSR
63 010316 010305          MOV     R3,R5        ; SAVE THE CONTENTS
64 010320 042703 177770    BIC     #177770,R3  ; GET THE COLOUR
65 010324 020304          CMP     R3,R4        ; AS EXPECTED?
66 010326 001404          BEQ     1$          ; IF YES, BRANCH
67 010330          ERRHRD 700,E700,ER700 ; ELSE REPORT THE ERROR
    010330 104456          TRAP          C$ERHRD
    010332 001274          .WORD       700
    010334 010452          .WORD       E700
    010336 003214          .WORD       ER700
68 010340          1$:   ENDSEG          ;----- END SEGMENT
    010340          10000$: TRAP          C$ESEG
    010340 104405
69
70 010342 005201          INC     R1          ; NEXT DATA PATTERN
71 010344 032701 000777    BIT     #777,R1     ; ALL DOTS DONE?
72 010350 001355          BNE     T7B        ; IF NOT, NEXT DOT
73 010352 162701 001000    SUB     #1000,R1    ; ELSE CLEAR DOT NUMBER
74
75 010356 062777 000400 171606  ADD     #400,@VTVCSR ; NEXT LINE
76 010364 103413          BCS     2$          ; IF ALL LINES DONE, BRANCH
77
78 010366 005202          INC     R2          ; ELSE INCREMENT LINE COUNTER
79 010370 020267 000054    CMP     R2,NLINS    ; READY FOR NEXT COLOUR?
80 010374 103743          BLO     T7B        ; IF NOT, READ NEXT LINE
81
82 010376 005002          CLR     R2          ; ELSE CLEAR LINE COUNTER
83 010400 005204          INC     R4          ; EXPECT NEXT COLOUR
84 010402 032704 000010    BIT     #10,R4      ; ALL COLOURS USED?
85 010406 001736          BEQ     T7B        ; IF NOT, DO NEXT LINE
86 010410 005004          CLR     R4          ; ELSE RESET COLOUR
87 010412 000734          BR     T7B        ; AND READ NEXT LINE
88
89 010414 006367 000030          2$:   ASL     NLINS    ; FLAG NEXT 3
90 010420 006367 000024          ASL     NLINS    ; ADDRESS LINES
91 010424 006367 000020          ASL     NLINS    ; TO BE TESTED
92 010430 026727 000014 001000  CMP     NLINS,#1000 ; ALL FINISHED?
93 010436 001402          BEQ     3$          ; IF YES, BRANCH
94 010440 000167 177526          JMP     T7A        ; ELSE NEXT ADDRESS LINES
95
96 010444          3$:   EXIT     TST          ; END OF TEST
    010444 104432          TRAP          C$EXIT
    010446 000050          .WORD       L10036-.
97
98 010450 000000          NLINS: .WORD     0          ; NO. OF LINES PER COLOUR
99
100
101 010452          120      111      103  E700: .NLIST  BEX
102                                     .ASCIZ  /PICTURE STORE LINE ADDRESSING ERROR/
103                                     .LIST   BEX
104 010516          .EVEN
                                     .ENDTST
    
```


HARDWARE TESTS MACRO V03.01 18-NOV-81 11:10:45 PAGE 40-2
TEST 7: PICTURE STORE LINE ADDRESSING

010516
010516 104401

L10036: TRAP C\$ETST

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

.SBTTL TEST 8: 8-DOT

```

:++
: THIS TEST CHECKS THAT IF DBUFF BIT 15 (8-DOT) IS SET, PICTURE
: STORE WRITES ARE MADE TO EIGHT DOTS AT A TIME REGARDLESS OF THE
: VALUES OF DBUFF BITS 0 - 2. FOR EACH COMBINATION OF THE LOWER DOT
: ADDRESS BITS, A NEW COLOUR IS USED AND THE 8-DOT WRITE VERIFIED.
:--

```

BGNTST

T8::

```

10 010520
   010520
11
12 010520 005002
13 010522 005003
14 010524 012777 000050 171440

```

```

CLR R2 ; INITIALISE DBUFF WRITE COLOUR
CLR R3 ; INITIALISE CSR COLOUR AND DOT NO.
MOV #50,@VTVCSR ; SET VIDEO ENABLE AND ZERO OFFSET

```

INITIALISE FIRST 8 DOTS

```

19 010532 012701 016000
20 010536 010177 171432
21 010542 005201
22 010544 032701 000007
23 010550 001372

```

```

1$: MOV #16000,R1 ; SET COLOUR TO WHITE
    MOV R1,@DBUFF ; WRITE DOT
    INC R1 ; NEXT DOT
    BIT #7,R1 ; 8 DOTS DONE?
    BNE 1$ ; IF NOT, DO MORE

```

DO 8 DOT TEST

```

27 010552
   010552 104404
28 010554 012701 100000
29 010560 050201
30 010562 050301
31 010564 010177 171404
32 010570 052701 040000
33 010574 042701 100000

```

```

T8A: BGNSEG ; ----- BEGIN SEGMENT
      TRAP CSBSEG
      MOV #100000,R1 ; SET 8-DOT
      BIS R2,R1 ; SET THE COLOUR
      BIS R3,R1 ; AND DOT NUMBER
      MOV R1,@DBUFF ; WRITE TO DBUFF
      BIS #40000,R1 ; SET READ PICTURE STORE
      BIC #100000,R1 ; CLEAR 8-DOT

```

```

35 010600 010177 171370
36 010604 017704 171362
37 010610 010405
38 010612 042704 177770
39 010616 020403
40 010620 001404

```

```

T8B: MOV R1,@DBUFF ; LATCH COLOUR INTO CSR
      MOV @VTVCSR,R4 ; READ THE CSR
      MOV R4,R5 ; SAVE THE CONTENTS
      BIC #177770,R4 ; GET THE COLOUR
      CMP R4,R3 ; EXPECTED COLOUR
      BEQ 1$ ; IF YES, BRANCH
      ERRHRD 800,E800,ER800 ; ELSE REPORT THE ERROR

```

```

TRAP CSERHRD
.WORD 800
.WORD E800
.WORD ER800

```

```

41 010622
   010622 104456
   010624 001440
   010626 010664
   010630 003404
42 010632 005201
43 010634 032701 000010
44 010640 001757
45 010642
   010642
   010642 104405

```

```

1$: INC R1 ; NEXT DOT ADDRESS
    BIT #10,R1 ; 8 DOTS READ?
    BEQ T8B ; IF NOT, READ NEXT
    ENDSEG

```

```

----- END SEGMENT
10000$: TRAP CSESEG

```

```

46
47 010644 062702 002000
48 010650 005203
49 010652 032703 000010

```

```

ADD #2000,R2 ; ELSE NEXT COLOUR
INC R3 ; NEXT READ COLOUR, DOT NO.
BIT #10,R3 ; ALL COLOURS DONE?

```

HARDWARE TESTS MACRO V03.01 18-NOV-81 11:10:45 PAGE 41-1
TEST 8: 8-DOT

```

50 010656 001735          BEQ      T8A      : IF NOT, DO NEXT
51 010660          EXIT     TST      : ELSE END OF TEST
   010660 104432
   010662 000030          TRAP     CSEXIT
                              .WORD   L10037-.
52
53
54 010664 070 040 055 E800: .NLIST  BEX
   .ASCIZ /8 - DOT WRITE FAILURE/
55 .LIST  BEX
56 .EVEN
57 010712          .NDTST
   010712
   010712 104401          L10037: TRAP  CSETST

```



```

1          .SBTTL TEST 9: PRESET
2
3          :++
4          : DBUFF BITS 10 - 12 ARE SET TO THE COLOUR BLACK (ZERO) AND A PRESET
5          : IS PERFORMED. THE READY BIT IN THE CSR IS CHECKED TO CLEAR AND
6          : THEN TO BE SET AFTER 50 MILLISECONDS. THE PICTURE STORE IS THEN READ
7          : TO CONFIRM THAT ALL LOCATIONS HAVE BEEN CORRECTLY SET. THE PROCESS
8          : IS THEN REPEATED FOR EACH OF THE EIGHT SELECTABLE COLOURS.
9          :--
10
11         010714          BGNTST
12         010714
13         010714 005001          CLR      R1          ; INITIALISE PRESET COLOUR
14         010716 005002          CLR      R2          ; AND EXPECTED CSR COLOUR
15
16
17         010720          T9A:  BGNSEG          ;----- BEGIN SEGMENT
18         010720 104404          TRAP      CSBSEG
19
20         : DO THE PRESET
21         :
22         010722          BGNSEG          ;----- BEGIN SEGMENT
23         010722 104404          TRAP      CSBSEG
24         010724 010177 171244          MOV      R1,@DBUFF          ; WRITE COLOUR TO DBUFF
25         010730 012777 000150 171234          MOV      #150,@VTVCSR          ; PERFORM THE PRESET
26         010736 017703 171230          MOV      @VTVCSR,R3          ; READ THE CSR
27         010742 105703          TSTB     R3          ; READY BIT CLEAR?
28         010744 100004          BPL      1$          ; IF YES, BRANCH
29         010746          ERRHRD 900,E900,ER900 ; ELSE REPORT THE ERROR
30
31         010746          TRAP      CSERHRD
32         010750          .WORD    900
33         010752          .WORD    E900
34         010754          .WORD    ER900
35
36
37         010756 004767 171434          1$:     JSR      PC,WAIT25          ; WAIT FOR 50 MILLISECONDS
38         010762 004767 171430          JSR      PC,WAIT25
39         010766 017703 171200          MOV      @VTVCSR,R3          ; READ THE CSR
40         010772 105703          TSTB     R3          ; READY SET?
41         010774 100404          BMI      2$          ; IF YES, BRANCH
42         010776          ERRHRD 901,E901,ER900 ; ELSE REPORT THE ERROR
43
44         010776          TRAP      CSERHRD
45         011000          .WORD    901
46         011002          .WORD    E901
47         011004          .WORD    ER900
48
49         011006          2$:     ENDSEG          ;----- END SEGMENT
50         011006          TRAP      10001$:  CSESEG
51         011006 104405
52
53         : CHECK THE PICTURE STORE
54         :
55         011010 012703 040000          MOV      #40000,R3          ; FIRST DOT POSITION
56         011014 012777 000040 171150          MOV      #40,@VTVCSR          ; FIRST LINE
57
58         011022          T9B:  MOV      R3,@DBUFF          ; LATCH COLOUR INTO CSR
59         011026 017704 171140          MOV      @VTVCSR,R4          ; READ THE CSR
60         011032 010405          MOV      R4,R5          ; SAVE THE CONTENTS

```



```

1      .SBTTL TEST 10: VIDEO ENABLE
2
3
4      :++
5      : THE PICTURE STORE IS PRESET TO BLUE FOR FIVE SECONDS. BIT 5 OF
6      : THE CSR (VIDEO ENABLE) IS THEN CLEARED FOR FIVE SECONDS, DURING
7      : WHICH TIME THE MONITOR DISPLAY SHOULD BE DARK. THE DISPLAY IS THEN
8      : REENABLED FOR A FURTHER FIVE SECONDS.
9      :--
10     BGNTST
11
12     T10::
13     : PRESET TO WHITE, VIDEO ENABLED
14     011316 012777 010000 170650      MOV    #10000,@DBUFF      : SET PRESET COLOUR BLUE
15     011324 012777 000140 170640      MOV    #140,@VTVCSR     : PERFORM THE PRESET
16     011332 012701 000310              MOV    #200.,R1         : WAIT FOR 5 SECONDS
17     011336 004767 171054      1$:   JSR    PC,WAIT25      : 25 MILLISECOND DELAY
18     011342                                BREAK                       : ALLOW OPERATOR BREAK-IN
19     011342 104422                                DEC    R1                 : TIME UP?
20     011344 005301                                BNE   1$                 : IF NOT, WAIT LONGER
21
22     : CLEAR VIDEO ENABLE
23     :
24     011350 005077 170616      2$:   CLR    @VTVCSR       : CLEAR VIDEO ENABLE
25     011354 012701 000310      MOV    #200.,R1         : WAIT FOR 5 SECONDS
26     011360 004767 171032      JSR    PC,WAIT25      : 25 MILLISECOND DELAY
27     011364                                BREAK                       : ALLOW OPERATOR BREAK-IN
28     011364 104422                                DEC    R1                 : TIME UP?
29     011366 005301                                BNE   2$                 : IF NOT, WAIT LONGER
30
31     : REENABLE VIDEO
32     :
33     011372 012777 000040 170572      3$:   MOV    #40,@VTVCSR     : SET VIDEO ENABLE
34     011400 012701 000310      MOV    #200.,R1         : WAIT FOR 5 SECONDS
35     011404 004767 171006      JSR    PC,WAIT25      : 5 MILLISECOND DELAY
36     011410                                BREAK                       : ALLOW OPERATOR BREAK-IN
37     011410 104422                                DEC    R1                 : TIME UP?
38     011412 005301                                BNE   3$                 : IF NOT, WAIT LONGER
39
40     011416                                EXIT   TST                : END OF TEST
41     011416 104432                                TRAP  .WORD              TRAP  CSEXIT
42     011420 000002                                L10041-
43     011422                                TRAP  CSETST
44     011422 104401                                L10041:
  
```


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

.SBTTL TEST 11: DISPLAY ADDRESSING

:++
 : THE PICTURE STORE IS PRESET TO BLUE. LINES OF MAGENTA ARE THEN
 : WRITTEN FROM THE TOP OF THE DISPLAY UNTIL THE WHOLE SCREEN HAS
 : CHANGED COLOUR. THE TRANSITION SHOULD BE SMOOTH, TAKING APPROXIMATELY
 : 7 SECONDS.
 : COLUMNS OF WHITE ARE THEN WRITTEN FROM THE LEFT SIDE OF THE DISPLAY
 : UNTIL THE SCREEN IS ALL WHITE. AGAIN THE TRANSITION SHOULD BE SMOOTH.
 : IF THE DISPLAY DOES NOT APPEAR AS DESCRIBED HERE, AN ADDRESSING
 : FAULT ON THE OUTPUT SIDE OF THE DEVICE IS INDICATED.
 :--

BGNTST

T11::

: PRESET SCREEN TO BLUE

: MOV #10000,@DBUFF : SET PRESET COLOUR BLUE
 : MOV #150,@VTVCSR : PERFORM THE PRESET
 : MOV #-1,R1 : SET UP WAIT COUNTER
 1\$: TSTB @VTVCSR : READY SET?
 BMI 2\$: IF YES, BRANCH
 BREAK : ELSE ALLOW OPERATOR BREAK IN TRAP CSBRK
 DEC R1 : WAITED TOO LONG?
 BNE 1\$: IF NOT, WAIT LONGER

: MOVE MAGENTA DOWN THE DISPLAY

2\$: MOV #40,@VTVCSR : START AT LINE 0
 MOV #12000,R1 : SET WRITE COLOUR TO MAGENTA
 3\$: MOV R1,@DBUFF : WRITE DOT
 INC R1 : NEXT DOT ADDRESS
 BIT #777,R1 : LINE FINISHED?
 BNE 3\$: IF NOT, DO NEXT DOT
 JSR PC,WAIT25 : WAIT FOR 25 MILLISECONDS
 BREAK : ALLOW OPERATOR BREAK IN TRAP CSBRK

BIC #1000,R1 : RESET TO START OF LINE
 ADD #400,@VTVCSR : AND NEXT LINE
 BCC 3\$: IF MORE LINES, GO BACK

: MOVE WHITE ACROSS THE DISPLAY

: MOV #40,@VTVCSR : START AT LINE 0
 : MOV #16000,R1 : SET WRITE COLOUR TO WHITE
 5\$: MOV R1,@DBUFF : WRITE DOT
 ADD #400,@VTVCSR : NEXT LINE ADDRESS
 BCC 5\$: IF MORE LINES, GO BACK
 MOV COUNT,R0 : GET 25 MILLISECOND COUNT
 ASR R0 : MAKE IT 12 MILLISECONDS

011424 012777 010000 170542
 011432 012777 000150 170532
 011440 012701 177777
 011444 105777 170522
 011450 100403
 011452
 011452 104422
 011454 005301
 011456 001372
 011460 012777 000040 170504
 011466 012701 012000
 011472 010177 170476
 011476 005201
 011500 032701 000777
 011504 001372
 011506 004767 170704
 011512
 011512 104422
 011514 042701 001000
 011520 062777 000400 170444
 011526 103361
 011530 012777 000040 170434
 011536 012701 016000
 011542 010177 170426
 011546 062777 000400 170416
 011554 103372
 011556 016700 170424
 011562 006200

```
55 011564 005300          6$: DEC      R0          : WAIT 12 MILLISECONDS
56 011566 001376          BNE     6$          :
57 011570          BREAK          : ALLOW OPERATOR BREAK IN          TRAP    C$BRK
   011570 104422
58
59 011572 005201          INC     R1          : NEXT COLUMN
60 011574 032701 000777  BIT     #777,R1    : ALL COLUMNS DONE?
61 011600 001360          BNE     5$          : IF NOT, DO NEXT
62
63 011602          EXIT TST          : ELSE END OF TEST
   011602 104432          TRAP    C$EXIT
   011604 000002          .WORD  L10042-.
64 011606          ENDTST
   011606          L10042: TRAP    C$ETST
   011606 104401
```

```

1      .SBTTL TEST 12: OFFSET
2
3
4      :++
5      : THE PICTURE STORE IS PRESET TO BLUE. A LINE OF MAGENTA IS THEN WRITTEN
6      : TO THE TOP OF THE DISPLAY AND THE OFFSET REGISTER CHANGED TO MOVE
7      : THE LINE DOWN ONE POSITION. THIS IS REPEATED UNTIL THE WHOLE
8      : DISPLAY IS MAGENTA, WHICH SHOULD TAKE ABOUT 7 SECONDS. THE DISPLAY IS
9      : THEN CHANGED TO WHITE FROM THE BOTTOM USING THE SAME PROCESS,
10     : UNTIL THE SCREEN IS ALL WHITE.
11     : IN BOTH CASES, THE LINE BETWEEN THE COLOURS SHOULD MOVE SMOOTHLY
12     : ACROSS THE SCREEN. ANY UNEVEN MOVEMENT INDICATES A PROBLEM
13     : IN THE OFFSET REGISTER.
14     :--
15     BGNTST
16
17     : PRESET SCREEN TO BLUE
18
19     011610 012777 010000 170356      MOV #10000, @DBUFF      : SET PRESET COLOUR BLUE
20     011616 012777 000140 170346      MOV #140, @VTVCSR      : PERFORM THE PRESET
21     011624 012701 177777              MOV #-1, R1             : SET UP WAIT COUNTER
22     011630 105777 170336      1$: TSTB @VTVCSR         : READY SET?
23     011634 100403              BMI 2$                  : IF YES, BRANCH
24     011636 104422              BREAK                    : ELSE ALLOW OPERATOR BREAK IN
25     011640 005301              DEC R1                   : WAITED TOO LONG?
26     011642 001372              BNE 1$                   : IF NOT, WAIT LONGER
27
28     : MOVE MAGENTA DOWN THE DISPLAY
29
30     011644 012701 177450      2$: MOV #177450, R1         : OFFSET LAST LINE TO TOP
31     011650 010177 170316      3$: MOV R1, @VTVCSR        : WRITE THE OFFSET
32     011654 012702 112000      MOV #112000, R2         : SET FIRST 8 DOTS MAGENTA
33     011660 010277 170310      5$: MOV R2, @DBUFF         : WRITE 8 DOTS
34     011664 062702 000010      ADD #10, R2             : NEXT DOT
35     011670 032702 000777      BIT #777, R2           : LINE WRITTEN?
36     011674 001371              BNE 5$                  : IF NOT, DO MORE DOTS
37     011676 162702 001000      SUB #1000, R2          : ELSE CLEAR DOT NUMBER
38     011702 004767 170510      JSR PC, WAIT25         : WAIT FOR 25 MILLISECONDS
39     011706 104422              BREAK                    : ALLOW OPERATOR BREAK-IN
40
41     011710 162701 000400      SUB #400, R1           : REDUCE THE OFFSET
42     011714 103355              BCC 3$                   : IF MORE LINES, BRANCH
43
44     : MOVE WHITE UP THE DISPLAY
45
46     011716 012701 000450      MOV #450, R1           : OFFSET FIRST LINE TO BOTTOM
47     011722 012703 000040      MOV #40, R3            : FIRST WRITE LINE
48     011726 032777 000020 170236 7$: BIT #20, @VTVCSR        : FIELD BIT CLEAR?
49     011734 001374              BNE 7$                  : IF NOT, WAIT
50     011736 010177 170230      MOV R1, @VTVCSR        : WRITE THE OFFSET
51     011742 010377 170224      MOV R3, @VTVCSR        : SET UP TO WRITE LINE
52     011746 012702 116000      MOV #116000, R2        : SET FIRST 8 DOTS WHITE
53     011752 010277 170216      8$: MOV R2, @DBUFF         : WRITE 8 DOTS
54     011756 062702 000010      ADD #10, R2            : NEXT DOTS
  
```

TRAP CSBRK

TRAP CSBRK

55	011762	032702	001000	BIT	#1000,R2	:	LINE WRITTEN?		
56	011766	001771		BEQ	8\$:	IF NOT, DO MORE DOTS		
57	011770	162702	001000	SUB	#1000,R2	:	ELSE CLEAR THE DOT NUMBER		
58	011774	004767	170416	JSR	PC,WAIT25	:	WAIT FOR 25 MILLISECONDS		
59	012000			BREAK		:	ALLOW OPERATOR BREAK-IN		
	012000	104422						TRAP	C\$BRK
60									
61	012002	062703	000400	ADD	#400,R3	:	NEXT WRITE LINE		
62	012006	062701	000400	ADD	#400,R1	:	INCREASE THE OFFSET		
63	012012	103345		BCC	7\$:	IF MORE LINES, BRANCH		
64	012014	012777	000010 170150	MOV	#10,@VTVCSR	:	CLEAR THE OFFSET REGISTER		
65									
66	012022			EXIT	TST				
	012022	104432						TRAP	C\$EXIT
	012024	000002						.WORD	L10043-
67	012026			ENDTST					
	012026							L10043:	
	012026	104401						TRAP	C\$ETST

.SBTTL TEST 13: PATTERN GENERATOR

```

:++
: THIS TEST IS FOR SETTING UP AND CHECKING THE OUTPUT TO THE DISPLAY
: MONITOR. NINE PATTERNS ARE PROVIDED - EIGHT TO PRESET THE SCREEN
: TO EACH POSSIBLE COLOUR, AND ONE TO OUTPUT A CROSS HATCH OF 32
: COLUMN BY 16 LINE RECTANGLES. THE NUMBER OF RECTANGLES DISPLAYED
: WILL DEPEND ON THE DEVICE CONFIGURATION.

```

```

: NORMALLY, THE NINE PATTERNS ARE DISPLAYED IN TURN, EACH REMAINING
: ON THE SCREEN FOR APPROXIMATELY FIVE SECONDS. HOWEVER, IF MANUAL
: SELECTION IS SELECTED AT START UP, EACH PATTERN IS KEPT DISPLAYED
: UNTIL THE RETURN KEY IS HIT ON THE OPERATOR CONSOLE.
:--

```

```

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16 012030          BGNTST
   012030
17
18 012030 005001          CLR    R1          ; FIRST PRESET COLOUR
19
20 012032 010177 170136  T13A: MOV    R1, @DBUFF      ; SET THE PRESET COLOUR
21 012036 012777 000150 170126  MOV    #150, @VTVCSR    ; DO THE PRESET
22
23 : WAIT FOR NEXT PATTERN
24
25 012044 005767 170114  2$:  TST    MANVEN      ; RUN MANUALLY?
26 012050 001413          BEQ    3$          ; IF NOT, BRANCH
27 012052          MANUAL      ; MANUAL ALLOWED?
   012052 104450
28 012054          BNCOMPLETE 3$      ; IF NOT, BRANCH
   012054 103011
29 012056          GMANID  READY, FLAG, A, 377, 0, 1, YES ; WAIT FOR <CR>
   012056 104443
   012060 000406
   012062 012302
   012064 000152
   012066 012304
   012070 000377
   012072 000000
   012074 000001
   012076
30 012076 000407          BR     5$          ; CONTINUE
31
32 012100 012704 000310  3$:  MOV    #200, R4      ; WAIT FOR 5 SECONDS
33 012104 004767 170306  4$:  JSR    PC, WAIT25
34 012110          BREAK
   012110 104422
35 012112 005304          DEC    R4          ; TIME UP?
36 012114 001373          BNE    4$          ; IF NOT, WAIT LONGER
37
38 : NEXT PATTERN
39
40 012116 032701 020000  5$:  BIT    #20000, R1    ; ALL PATTERNS DONE?
41 012122 001065          BNE    12$         ; IF YES, END TEST
42 012124 062701 002000  ADD    #2000, R1    ; ELSE NEXT COLOUR
43 012130 032701 020000  BIT    #20000, R1    ; ALL PRESETS DONE?
44 012134 001736          BEQ    T13A        ; IF NOT, NEXT COLOUR

```

T13::

```

TRAP  C$MANI
BCC   3$
TRAP  C$GMAN
BR    10000$
.WORD FLAG
.WORD T$CODE
.WORD READY
.WORD 377
.WORD T$LOLIM
.WORD T$HILIM

```

10000\$:

TRAP C\$BRK

```

45
46
47
48 012136 005077 170032
49 012142 012777 000140 170022
50 012150 012701 177777
51 012154 105777 170012
52 012160 100403
53 012162
54 012164 104422
55 012166 005301
56 012166 001372
57
58
59 012170 012702 116000
60 012174 010277 167774
61 012200 062702 000010
62 012204 032702 001000
63 012210 001771
64 012212 000414
65
66
67
68 012214 012702 016000
69 012220 010277 167750
70 012224 062702 000037
71 012230 010277 167740
72 012234 005202
73 012236 032702 001000
74 012242 001766
75
76 012244 062777 000400 167720
77 012252 103674
78 012254 017703 167712
79 012260 042703 170377
80 012264 001741
81 012266 020327 007400
82 012272 001736
83 012274 000747
84
85 012276
86 012276 104432
87 012302 000000
88
89
90 012304 120 122 105
91
92
93 012344
012344
012344 104401

: GENERATE CROSS HATCH
:
: CLR @DBUFF : SET PRESET COLOUR BLACK
: MOV #140,@VTVCSR : DO THE PRESET
: MOV #-1,R1 : SET UP WAIT COUNTER
6$: TSTB @VTVCSR : READY SET?
: BMI 7$ : IF YES, BRANCH
: BREAK : ELSE ALLOW OPERATOR BREAK IN TRAP CSBRK
: DEC R1 : WAITED TOO LONG?
: BNE 6$ : IF NOT, WAIT LONGER

: WRITE FULL LINE
7$: MOV #116000,R2 : INITIALISE DOT NO.,WHITE
8$: MOV R2,@DBUFF : WRITE 8 DOTS
: ADD #10,R2 : NEXT DOTS
: BIT #1000,R2 : LINE WRITTEN?
: BEQ 8$ : IF NOT, DO MORE DOTS
: BR 11$ : ELSE DO NEXT LINE

: WRITE PARTIAL LINE
9$: MOV #16000,R2 : SET DOT 0, COLOUR WHITE
10$: MOV R2,@DBUFF : WRITE FIRST DOT
: ADD #31,R2 : END OF SQUARE
: MOV R2,@DBUFF : WRITE PENULTIMATE DOT
: INC R2 : NEXT DOT
: BIT #1000,R2 : LINE FINISHED?
: BEQ 10$ : IF NOT, GO BACK
11$: ADD #400,@VTVCSR : NEXT LINE
: BCS 2$ : IF FINISHED, GO AND WAIT
: MOV @VTVCSR,R3 : ELSE READ CSR
: BIC #170377,R3 : GET LINE NO. LOW BITS
: BEQ 7$ : IF ZERO, DO FULL LINE
: CMP R3,#7400 : LAST LINE OF SQUARE?
: BEQ 7$ : IF YES, DO FULL LINE
: BR 9$ : ELSE PARTIAL LINE

12$: EXIT TST TRAP CSEXIT
: .WORD L10044-.

FLAG: .WORD 0
: .NLIST BEX
90 READY: .ASCIZ /PRESS 'RETURN' FOR NEXT PATTERN/
: .LIST BEX
: .EVEN
: ENDTST

L10044: TRAP CSETST

```


1 012346
2

ENDMOD

1
12
13
42
43
44
45
46
47
48
49
50
51
52
53
54
55
65
66
67
68
69
76
77

.TITLE PARAMETER CODING

.SBTTL HARDWARE PARAMETER CODING SECTION

BGNMOD

;++

: THE HARDWARE PARAMETER CODING SECTION CONTAINS MACROS
: THAT ARE USED BY THE SUPERVISOR TO BUILD P-TABLES. THE
: MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE
: INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES. THE
: MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS
: WITH THE OPERATOR.
:--

BGNHRD

012346

012346
012346
012350

000004

.WORD L10045-LSHARD/2
LSHARD::

GPRMA GETADR,0,0,160000,177776,YES

012350

012350
012352
012354
012356

000031
012360
160000
177776

.WORD TSCODE
.WORD GETADR
.WORD T\$LOLIM
.WORD T\$HILIM

ENDHRD

012360

012360

.EVEN
L10045:

012360
012363
012366
012371
012374
012377

126
063
113
104
105
000

124
061
040
104
123

126
055
101
122
123

GETADR: .ASCIZ /VTV31-K ADDRESS/

1
2
3
4
5
6
7
8
9
10
11
12
13
22
23
24
25
26
27
28
29
36
37
38
39
40
41
42
49
50
51
52
53

.SBTTL SOFTWARE PARAMETER CODING SECTION

```

:++
: THE SOFTWARE PARAMETER CODING SECTION CONTAINS MACROS
: THAT ARE USED BY THE SUPERVISOR TO BUILD P-TABLES. THE
: MACROS ARE NOT EXECUTED AS MACHINE INSTRUCTIONS BUT ARE
: INTERPRETED BY THE SUPERVISOR AS DATA STRUCTURES. THE
: MACROS ALLOW THE SUPERVISOR TO ESTABLISH COMMUNICATIONS
: WITH THE OPERATOR.
:--

```

```

012400          BGNSFT
012400 000010
012402
                                .WORD L10046-L$SOFT/2
                                L$SOFT::

012402          GPRML MANINT,0,1,YES
                                .WORD T$CODE
                                .WORD MANINT
                                .WORD 1

012406 000001
                                .WORD T$CODE
                                .WORD GETMAX
                                .WORD 377
                                .WORD T$LOLIM
                                .WORD T$HILIM

012410          GPRMD GETMAX,2,D,377,0,255.,YES
                                .WORD T$CODE
                                .WORD GETMAX
                                .WORD 377
                                .WORD T$LOLIM
                                .WORD T$HILIM

012410 001052
012412 012450
012414 000377
012416 000000
012420 000377

                                .EVEN

012422          ENDSFT
                                L10046: .EVEN

                                .NLIST BEX
012422 122 125 116 MANINT: .ASCIZ /RUN TEST 13 MANUALLY?/
012450 101 125 124 GETMAX: .ASCIZ /AUTODROP ERROR COUNT/
                                .LIST BEX
                                .EVEN

$PATCH::

                                .BLKW 50. : PATCH AREA
                                .BLKB 400-<.&377> : LASTAD SHIFT FOR LSI BUG
                                LASTAD

013000
                                .EVEN
                                .WORD T$FREE
                                .WORD T$SIZE

013000 013012
013002 000003
                                L$LAST::
013004
                                ENDMOD

```


PARAMETER CODING
SOFTWARE PARAMETER CODING SECTION

MACRO V03.01 18-NOV-81 11:10:45 PAGE 52

E 6

```
1  
2  
15  
16 013004          BGNSETUP          1  
17 013004          BGNPTAB  
   013004 000000  
   013006 000001  
   013010  
18 013010 174000    .WORD 174000  
19 013012          ENDPTAB  
   013012  
20 013012          ENDSETUP  
21          .END  
  
L10047: .WORD 0  
        .WORD L10051-./2-1  
  
L10051:
```

PARAMETER CODING
SYMBOL TABLE

ABITS 002204 G
 ABORT 005042
 ADR = 000020 G
 ASSEMB= 000010
 BIT0 = 000001 G
 BIT00 = 000001 G
 BIT01 = 000002 G
 BIT02 = 000004 G
 BIT03 = 000010 G
 BIT04 = 000020 G
 BIT05 = 000040 G
 BIT06 = 000100 G
 BIT07 = 000200 G
 BIT08 = 000400 G
 BIT09 = 001000 G
 BIT1 = 000002 G
 BIT10 = 002000 G
 BIT11 = 004000 G
 BIT12 = 010000 G
 BIT13 = 020000 G
 BIT14 = 040000 G
 BIT15 = 100000 G
 BIT2 = 000004 G
 BIT3 = 000010 G
 BIT4 = 000020 G
 BIT5 = 000040 G
 BIT6 = 000100 G
 BIT7 = 000200 G
 BIT8 = 000400 G
 BIT9 = 001000 G
 BKGND 007500
 BOE = 000400 G
 CHKMAX 002256 G
 CONT 004736
 COUNT 002206 G
 CSAU = 000052
 CSAUTO= 000061
 CSBRK = 000022
 CSBSEG= 000004
 CSBSUB= 000002
 CSCEFG= 000045
 CSCCLK= 000062
 CSCLEA= 000012
 CSCLOS= 000035
 CSCLP1= 000006
 CSCVEC= 000036
 CSDCLN= 000044
 CSDODU= 000051
 CSDRPT= 000024
 CSDU = 000053
 CSEDIT= 000003
 CSERDF= 000055
 CSERHR= 000056
 CSERRO= 000060
 CSERSF= 000054
 CSERSO= 000057
 CSESCA= 000010

CSESEG= 000005
 CSESUB= 000003
 CSETST= 000001
 CSEXIT= 000032
 CSGETB= 000026
 CSGETW= 000027
 CSGMAN= 000043
 CSGPHR= 000042
 CSGPLO= 000030
 CSGPRI= 000040
 CSINIT= 000011
 CSINLP= 000020
 CSMANI= 000050
 CSMEM = 000031
 CSMSG = 000023
 CSOPEN= 000034
 CSPNTB= 000014
 CSPNTF= 000017
 CSPNTS= 000016
 CSPNTX= 000015
 CSQIO = 000377
 CSRDBU= 000007
 CSREFG= 000047
 CSRESE= 000033
 CSREVI= 000003
 CSRFLA= 000021
 CSRPT = 000025
 CSSEFG= 000046
 CSSPRI= 000041
 CSSVEC= 000037
 CSTPRI= 000013
 DBUFF 002174 G
 DFPTBL 002160 G
 DIAGMC= 000000
 DOT 002200 G
 DROPD 005610
 EADD 004652
 EF.CON= 000036 G
 EF.NEW= 000035 G
 EF.PWR= 000034 G
 EF.RES= 000037 G
 EF.STA= 000040 G
 END 005050
 ERRCNT 002176 G
 ER200 002444 G
 ER300 002474 G
 ER400 002550 G
 ER401 002602 G
 ER500 002634 G
 ER501 002724 G
 ER600 003032 G
 ER700 003214 G
 ER800 003404 G
 ER900 003476 G
 ER902 003530 G
 EVL = 000004 G
 E\$END = 002100

E\$LOAD= 000035
 E100 006006
 E101 006054
 E200 006266
 E200B 003636
 E201 006314
 E202 006346
 E203 006405
 E300 006544
 E300B 003663
 E300C 003776
 E400 006774
 E400B 004036
 E401 007067
 E401B 004152
 E500 007502
 E501 007552
 E600 010114
 E700 010452
 E800 010664
 E800B 004257
 E800C 004344
 E900 011122
 E900B 004427
 E901 011166
 E902 011247
 FLAG 012302
 FSAU = 000015
 FSAUTO= 000020
 FSBGN = 000040
 F\$CLEA= 000007
 F\$DU = 000016
 F\$END = 000041
 F\$HARD= 000004
 F\$HW = 000013
 F\$INIT= 000006
 F\$JMP = 000050
 F\$MOD = 000000
 F\$MSG = 000011
 F\$PROT= 000021
 F\$PWR = 000017
 F\$RPT = 000012
 F\$SEG = 000003
 F\$SOFT= 000005
 F\$SRV = 000010
 F\$SUB = 000002
 F\$SW = 000014
 F\$TEST= 000001
 GETADR 012360
 GETMAX 012450
 GOOBAD 004571
 G\$CNTO= 000200
 G\$DELM= 000372
 G\$DISP= 000003
 G\$EXCP= 000400
 G\$HILI= 000002
 G\$LOLI= 000001

G\$NO = 000000
 G\$OFFS= 000400
 G\$OFSI= 000376
 G\$PRMA= 000001
 G\$PRMD= 000002
 G\$PRML= 000000
 G\$RADA= 000140
 G\$RADB= 000000
 G\$RADD= 000040
 G\$RADL= 000120
 G\$RADO= 000020
 G\$XFER= 000004
 G\$YES = 000010
 HELP = 000000
 HOE = 100000 G
 IBE = 010000 G
 IDU = 000040 G
 IER = 020000 G
 ISR = 000100 G
 IXE = 004000 G
 ISAU = 000041
 ISAUTO= 000041
 ISCLN = 000041
 ISDU = 000041
 ISHRD = 000041
 ISINIT= 000041
 ISMOD = 000041
 ISMSG = 000041
 ISPROT= 000040
 ISPTAB= 000041
 ISPWR = 000041
 ISRPT = 000041
 ISSEG = 000041
 ISSETU= 000041
 ISSFT = 000041
 ISSRV = 000041
 ISSUB = 000041
 ISTST = 000041
 JSJMP = 000167
 KLINT 005162
 LCLOCK 005074
 LINDOT 004503
 LOE = 040000 G
 LOT = 000010 G
 LSACP 002110 G
 LSAPT 002036 G
 LSAUT 002070 G
 LSAUTO 005424 G
 LSCCP 002106 G
 LSCLEA 005510 G
 LSCO 002032 G
 LSDEPO 002011 G
 L\$DESC 002220 G
 L\$DESP 002076 G
 L\$DEVP 002060 G
 L\$DISP 002124 G
 L\$DLY 002116 G

LSDTP 002040 G
 LSDTYP 002034 G
 LSDU 005562 G
 LSDUT 002072 G
 LSDVTY 002210 G
 LSEF 002052 G
 L\$ENVI 002044 G
 L\$ETP 002102 G
 L\$EXP1 002046 G
 L\$EXP4 002064 G
 L\$EXP5 002066 G
 LSHARD 012350 G
 LSHIME 002120 G
 LSHPCP 002016 G
 LSHPTP 002022 G
 LSHW 002160 G
 LSICP 002104 G
 L\$INIT 004722 G
 L\$LADP 002026 G
 L\$LAST 013004 G
 L\$LOAD 002100 G
 L\$LUN 002074 G
 L\$MREV 002050 G
 L\$NAME 002000 G
 L\$PRIO 002042 G
 L\$PROT 004714 G
 L\$PRT 002112 G
 L\$REPP 002062 G
 L\$REV 002010 G
 L\$SOFT 012402 G
 L\$SPC 002056 G
 L\$SPCP 002020 G
 L\$SPTP 002024 G
 L\$STA 002030 G
 L\$SW 002164 G
 L\$TEST 002114 G
 L\$TML 002014 G
 L\$UNIT 002012 G
 L1000 002162
 L10001 002170
 L10002 002442
 L10003 002472
 L10004 002546
 L10005 002600
 L10006 002632
 L10007 002722
 L10010 003030
 L10011 003212
 L10012 003402
 L10013 003474
 L10014 003526
 L10015 003634
 L10017 005050
 L10020 005506
 L10021 005560
 L10022 005636
 L10023 006136

PARAMETER CODING
SYMBOL TABLE

MACRO V03.01 18-NOV-81 11:10:45 PAGE 52-2

L10024 005716
L10025 005744
L10026 006450
L10027 006260
L10030 006604
L10031 007176
L10032 006672
L10033 006772
L10034 007630
L10035 010162
L10036 010516
L10037 010712
L10040 011314
L10041 011422
L10042 011606
L10043 012026
L10044 012344
L10045 012360
L10046 012422
L10047 013010
L10051 013012
MANINT 012422
MANVEN 002164 G
MAXERR 002166 G
NCOLS 010112
NERRS 002350
NEXT 004764
NLINS 010450
NOCLOK 005220
NXMFLG 002202 G
NXMTRP 002430 G

ONEFIL= 000001
OSAPTS= 000000
OSAU = 000000
OSBGNR= 000000
OSBGNS= 000001
OSDU = 000001
OSERRT= 000000
OSGNSW= 000001
OSPOIN= 000001
OSSETU= 000001
PNT = 001000 G
PRI = 002000 G
PRI00 = 000000 G
PRI01 = 000040 G
PRI02 = 000100 G
PRI03 = 000140 G
PRI04 = 000200 G
PRI05 = 000240 G
PRI06 = 000300 G
PRI07 = 000340 G
READY 012304
SETCLK 005052
SETUP 004756
SFPTBL 002164 G
SVCGBL= 000000
SVCINS= 000001
SVCSUB= 000001
SVCTAG= 000001
SVCTST= 000001
SSLSYM= 010000
TIMMSG 005354

TKB = 177562
TKS = 177560
TPB = 177566
TPS = 177564
TTINT 005326
TSARGC= 000002
TSCODE= 001052
TSERRN= 001606
TSEXCP= 000000
TSFLAG= 000040
TSFREE= 013012
TSGMAN= 000000
TSHILI= 000377
TSLAST= 000001
TSLOLI= 000000
TSLSYM= 010000
TSLTNO= 000015
TSNEST= 177777
TSNS0 = 000000
TSNS1 = 000005
TSNS2 = 000003
TSNS3 = 000003
TSPCNT= 000000
TSPTAB= 010050
TSPTHV= 000001
TSPTNU= 000001
TSSAVL= 177777
TSSEGL= 177777
TSSEK0= 010000
TSSEK1= 010001
TSSIZE= 000003

TSSUBN= 000000
TSTAGL= 177777
TSTAGN= 010052
TSTEMP= 000000
TSTEST= 000015
TSTSTM= 177777
TSTSTS= 000001
TSSAUT= 010020
TSSCLE= 010021
TSSDAT= 010051
TSSDU = 010022
TSSHAR= 010045
TSSHW = 010000
TSSINI= 010017
TSSMSG= 010015
TSSPC = 000001
TSSPRO= 010016
TSSPTA= 010050
TSSSEG= 010000
TSSSOF= 010046
TSSSRV= 010002
TSSSUB= 010033
TSSSW = 010001
TSSTES= 010044
T1 005640 G
T1.1 005672
T1.2 005720
T10 011316 G
T11 011424 G
T12 011610 G
T13 012030 G

T13A 012032
T2 006140 G
T2.1 006140
T3 006452 G
T4 006606 G
T4A 006612
T4.1 006606
T4.2 006674
T5 007200 G
T5A 007214
T6 007632 G
T6A 007640
T6B 007750
T7 010164 G
T7A 010172
T7B 010304
T8 010520 G
T8A 010552
T8B 010600
T9 010714 G
T9A 010720
T9B 011022
UAM = 000200 G
UUT 002170 G
VTVCSR 002172 G
WAIT25 002416 G
XSALWA= 000000
XSFALS= 000040
XSOFFS= 000400
XSTRUE= 000020
\$PATCH 012476 G

. ABS. 013012 000
000000 001

ERRORS DETECTED: 0

VIRTUAL MEMORY USED: 20500 WORDS (81 PAGES)
DYNAMIC MEMORY AVAILABLE FOR 70 PAGES
ZVTVA0.BIN,ZVTVA0.LST=RT.MAC/M,ZVTVA0.SRC