Date: 16 Apr 1986

Author: Karl Grabe

Subject: Vegas ROM Diagnostics

Document Version Number: 00:30

Test Document Number: TSKA 0047

Revision History

00:00 Initial Release (KRG)
00:10 Error codes changed for Alpha 2.0 (KRG)
Test pointer table described
Self Test System Beep shortened
Clock Interrupt, FPI Speed tests added for Alpha 2.0

00:11 Staggered screen error code for ram failures (16Mar86 KRG)
Rom Checksum "RM" error code added
"BP" Error message removed
00:30 Burn In Non Volatile Ram usage (16Apr86 KRG)
Burn In connector pin usage
Error Codes changed
FDE, Shadow tests added
Fail err code to port 1
Use of border colors for error detection
FDE uP fatal error code added
ROM checksum DD=01 error code added
Clock NVR DD=11 error code added

GENERAL

The Vegas ROM Diagnostics are used in the following environments:

1) User Self Test

2) Board level Burn-IN

3) Board Level test

4) Final System test

Brief Description of above:

1) User Self Test

This is the equivalent of the Ve Kernal Test. The tests run here are a subset of those run in Burn-In. When the test completes a "System Good" or System Bad Error code message is displayed in 40 column text. Non Volatile Ram (NVR) is not used for test status storage.

2) Board Level Burn In

Vegas boards will have a burn in period of approx 48 hours for initial production. During this time the boards are power and temperature cycled. During each power on cycle a group of tests are run several times. At the end of the cycles the test status is stored in Non Volatile Ram (NVR) and the board awaits power down. On each power up test status is read back from NVR. After burn in the post BI tester reads back NVR test results.

3) & 4) Board and Final System Test

In these environments, or anytime disk-based diagnostics are running, any or all of the built in ROM tests can be run. This is achieved using a table of pointers starting at the 3rd byte of the diagnostics.

The ROM Diagnostics are broken into two parts:

1) The Test Sequencer

2) The tests

The Test Sequencer is described next in detail. For a description of the individual tests see page 8.

1) The Test Sequencer

Overview

This piece of code is responsible for determining what the test environment is, running the appropriate tests and storing/displaying the test results. The sequencer supervises both Burn-In and User Self test. The sequencer is not used with disk based diagnostics (eg board functional test) but the tests themselves can be called if required.

The sequencer determines the test environment using signals brought out on the Burn-In connector on the front LHS of the logic board. The input signals are SW0, SW1, and DIAGSW. SW0 & 1 are the open and closed apple keys resp. and also button 1 & 2 resp. on the game I/O connector. DIAGSW is an unused Mega // input (originally intended as a mouse button down input).

The following table shows the different possibilities following a processor reset (including power up), note the User Self test is invoked exactly the same way as on the //e:

Soft Switches for Burn In/Self test.

DIAGTYPE+/-	SW1	SW0	MODE
0	0	0	Normal Power UP
0	0	1	Control Panel
0	1	0	Cold Restart
0	1	1	Self Diagnostic, no wr NVR
1	0	X	BI Diagnostic , wr NVR
1	1	X	Self Diagnostic, no wr NVR
BI.FLAG	BI.ALT	SELF	<-- Test s/w Equate Names
N/A	I= Key	CA Key	<-- From Keyboard
9/10	5 :	----	<-- BI Connector pins

Note: a. /wr NVR = write test STATUS registers to Non Volatile Ram.

b) DIAGTYPE (MSE $C046) is active lo eg MSE=0 is a hi on the Burn In connector (= DIAGSW on connector)

c) With no Keyboard plugged in SW0 & 1 are = 0 as opposed to 1 or a //e. A Vegas <u>will not</u> run Self Diagnostics or power up when the Keyboard is not attached.

d) DIAGSW is only available on the BI connector. The user can get into Burn In diagnostics.

The sequencer determines what tests to run from a 16 bit test mask. This test mask is fixed (in rom) for User Self test and programmable (using NVR set up by the Pre-Burn-In functional tester) in Burn-In. This allows the board manufacturing site to experiment with turning different tests, particularly ram tests, on and off to see which tests catch the most failures.

Test Sequencer: User Self Test

The tests run here are a subset of those run in Burn-In. User Self Test may be invoked in several ways:

1) Press Control-Open Apple-Closed Apple-Reset (retro only)
2) Press Control-Funct-Open Apple-Reset (Cortland only)

3) Press Open Apple-Closed Apple on power up (retro only)
4) Press Funct-Open Apple on power up (Cortland only)

5) Press Game I/O buttons BTN0, BTN1 on power up

The test takes approx 35 seconds to complete. During most of this time the test number being executed is visible on the bottom center of the screen followed by 6 zeros.

After all tests have been successfully completed a continuous 1 KHz 1-second beep is emitted and a 'System Good' message is displayed on the screen. The beep is used in manufacturing to test the speaker at final system test. The system can now be rebooted by pressing Control-Reset.

If any tests fail then a 6 KHz 1-second beep is emitted. A 'System Bad AAEEDDDD' message is displayed on the lower LHS of the screen and a staggered 'AAEEDDDD' is also displayed on the upper LHS to help reading the error code in the event of a ram failure. 'AA' is the test number that failed and EE-DD is the fail code. See individual tests for complete fail codes.

AN0/1 and disk port phases 1 & 2 are used to tell the outside world that the test sequence is ongoing/complete and and that the UUT has passed/failed the self test (see Annunciator table above). NVR is never read or written to by the sequencer. Some of the tests may use NVR but if so they will restore its contents after the test is complete.

Cortland's final system test is performed with the housing cover on so it is not possible to monitor Self Test on the burn-in connector. Instead the disk phase lines are used. The phase lines were carefully chosen so that it would not affect any peripheral disk devices including UniDisk 3.5 that the user may have connected during User Self test.

Error codes for Beta 2.0 and later rom releases:

Error code (AA)	Test
"RM"/01	Checksum
02	Ram Moving Inversions
03	Softswitch
04	Ram Address
05	FPI Speed
06	Serial I/O
07	Real Time Clock
08	Battery Ram
09	Front Desk Bus
0A	Shadowing
0E	Interrupts

For detailed error EEDDDD codes see individual test descriptions below.

The test number followed by 6 zeros is displayed just prior to calling each test and during execution of the test. This is done so that if the test hangs it is still possible to determine the most likely cause of the fault by reading the test number AA. If a test number followed by zeros is displayed on the screen for more than 35 seconds it is likely that that test has hung.

Note that the test number is not displayed during the Moving Inversions Ram test (instead you see the hires or super res screen).

Also for System Bad:

If the system fails self test the fail code bytes (6) preceded by a '*' are transmitted to the printer port. This allows automatic error logging in Cork's Final System test. (assuming the fault is not very serious and the serial ports are working). The control Panel settings are ignored and the default settings for port 1 are used instead (except for the baud rate which is set to 1200). This will also provide a means reading the error code if the video is not working properly (ie a Mega // softswitch problem).

Note: 'A System Bad: FFxxxxxx' message means the system went into Burn In diagnostics. This should not occur in Self Test but if it does then there is probably a hardware problem with the Mega //. This error message is given because the sequencer expects Battery Ram to be set up in a specific way (by the pre burn in tester) before commencing the diagnostics.

Interrupting Self Test

It is safe to interrupt the self test (by pressing Control Reset) only during the start of the Moving Inversions ram test (while screen displays Hires/Super Hires patterns). An interruption after this may cause the contents of Battery Ram or the Clock Time to be corrupted.

Board Level Rework:

Because the diagnostics require some of the system to be functional before they will run properly it is likely that they may crash/hang on boards that have serious hardware problems. The rom diagnostics sequencer has several features built in to help in these situations:

a) The test number is printed on the screen BEFORE the test is executed. If the test hangs then it is most likely that the last test number printed on the screen is causing the problem.

b) The sequencer changes the border colour each time a new test is run. By noting how the colours change on a good system it is possible to determine where a problem is occurring on a bad system in cases where screen text is not working. The colours start at 0 and increment once for each test with the exception of the moving inversions ram test which increments a further 4 times by itself. See Bank## equates files for the various border colours.

c) The fail error code is sent to port 1. Again this is useful if video is not functional.

d) The error code is printed on the top LHS of the screen 3 time and staggered. This helps reading the error code in the event of partial ram failures.

e) The first part of the rom checksum test (which is the very first test) does a register based ram test for $0000 to $0400 in bank zero. This prevents a roms error code being presented when there is reallv a ram problem (the rom checksum routine uses some ram).

```
.................................................................
;
; Self Test Annunciator/Disk Port Truth Table:
;
; Note: Use an LEDs connected via a buffer to game I/O connector or
; disk port to verify UUT without VDU.
;
;_____________________________________________________________
;  |   AN0   |   AN1   |          MODE                        |
;=============================================================
;  |    1    |    0    | --------                             |
;-------------------------------------------------------------
;  |    0    |    0    | Test running                         |
;-------------------------------------------------------------
;  |    1    |    1    | Testing complete, all tests pass     |
;-------------------------------------------------------------
;  |    0    |    1    | Testing complete, a test failed      |
;-------------------------------------------------------------
;  | Phase 1 | Phase 2 | <-- Final System Test with cover on  |
;-------------------------------------------------------------
;  | 13/14   | 17/18   | <-- BI Connector Pins                |
;-------------------------------------------------------------
; Note: Cork uses dish port phase lines P1 & P2 to match AN0 & AN1
```

Test Sequencer: Burn In

In Burn In the number of test cycles the board performs before awaiting power down is programmable (by the Pre-Burn-In Functional tester). Thus altering the power cycling time becomes easy. The following is a list of NVR registers used by the Pre-Burn In tester, BI rom diagnostics, and the Post BI tester:

```
BI.STATUS    DS 4    ;Test Results -) Stored by ROM diagnostics
BI.COUNTER   DS 2    ;Counts the number of test cycles done -) ROM Diag
BI.PASSES    DS 2    ;Number of cycles passed -) ROM diagnostic
BI.FAILS     DS 2    ;Number of cycles failed -) ROM diagnostic
BI.LASTF     DS 2    ;The cycle the last fail occurred on -) ROM diagnostic
CYCLES.MAX   DS 2    ;# Test cycles per power cycle -)  Pre BI Tester
VALID.CHK    DS 3    ;3 byte check signature -) Pre BI Tester
BI.ALT.MASK DS 2     ;What tests to run -) Pre BI tester.
```

These registers work in the manufacturing process in the following manner:

a) The Pre-BI board tester tests the board and if it passed sets up the following NVR registers: CYCLES.MAX, VALID CHECK, BI.ALT.MASK. Each bit in the 16 bit BI.ALT.MASK represents a test in rom and may be turned on or off.

b) When the board is in Burn In the sequencer looks for a certain sequence in the VALID.CHECK bytes. If this sequence is not found then the Pre-BI tester failed to set up NVR or the UUT/NVR is bad so the test halts setting the FAIL.LED(D). IF this sequence is correct then the tests mask BI.ALT.MASK is used instead of the default mask. The entire test sequence is repeated for a number of cycles = CYCLES.MAX. When the cycles are complete the power off annunciator is activated to tell the BI controller that the UUT is inactive and ready to power down. Test fail status (if any) is stored in NVR to be read by the Post-BI tester. Annunciators AN0 & AN1 are toggle so that the Burn In controller can monitor all boards in the BI oven (see Annunciator table later on).

c) The Post BI tester reads all NVR registers from which it can determine the following:

1. If the board executed correct # of test cycles (BI.COUNTER)
2. Whether the board failed at any time during BI

If it failed:

3) How often it failed (BI.FAILS)
4) When the fail s occurred (BI.LASTF)
5) What test last failed and its error code (BI.STATUS)

The tester additionally checks that BI.COUNTER = BI.PASSES + BI.FAILS and also that VALID.CHK still contains the check bytes put there by the Pre-BI tester. If these check out then the above results 1-5 are valid. The board is then functionally tested again.

If the board passes, the tester resets NVR to it's default contents as set up by the firmware on a system powering up for the first time.

Note that it is now possible to determine (using BI.LASTF w . . . . . temperature . during a test failed (assuming NVR hasn't failed).

In Burn in the annunciators reflect what the rom diagnostics are doing as follows:

```
;.....................................................
;
; Burn In Annunciator/Disk Port Truth Table:
;
; Note: The PASS/FAIL.LED (AN0 & Phase 1) signal also operates
;       in self test. Use an LED connected via a buffer to gate
;       I/O connector/disk port to verify ULT without VDU.
;
;  ---------------------------------------------------------------
;  |   AN0   |   AN1   |          MODE                            |
;  ===============================================================
;  |    1    |    0    | Passed last cycle, continue              |
;  |---------|---------|------------------------------------------|
;  |    0    |    0    | Failed last cycle, continue              |
;  |---------|---------|------------------------------------------|
;  |    1    |    1    | Pass, ready to power down                |
;  |---------|---------|------------------------------------------|
;  |    0    |    1    | Fail, ready to power down                |
;  |---------|---------|------------------------------------------|
;  |PASS.LED |POWER.ON | <-- Equate Names to turn on  (1)         |
;  |FAIL.LED |POWER..  |     -- Equate Names to turn off (0)      |
;  |---------|---------|------------------------------------------|
;  | Phase 1 | Phase 2 | <-- Final System Test with cover on      |
;  |---------|---------|------------------------------------------|
;  |  13/14  |  17/18  | <-- BI Connector Pins                    |
;  ---------------------------------------------------------------
; Note: Core uses disk port phase lines P1 & P2 to match AN0 & AN1
```

ROM Tests available:

This is a list of tests in ROM:

```
 PAGE
;.....
;
; Tests available:
;
; Note: Each TEST NO corresponds to a bit in the 16 bit
;       C register on calling EXT.SEQ. If the bit is true
;       then the test is run, else it is skipped.
;       The test subroutine must clear the carry if the
;       test passed, else set the carry. The sequencer
;       automatically  passes a skipped test. Two different
;       groups of tests can be run as defined in the equates file
;       (a third set can be defined in Battery backed up ram):
;
;_____________________________________________________________
;   | TST NO  |  Tests performed                              |
;=============================================================
;   | LSB: 0  |  Bank $FE/$FF Roms Checksum                   |
;-------------------------------------------------------------
;   |      1  |  Ram:  Moving Inversions                      |
;-------------------------------------------------------------
;   |      2  |  Softswitch/STATEREG test                     |
;-------------------------------------------------------------
;   |      3  |  Ram: Addressing                              |
;-------------------------------------------------------------|
;   |      4  |  FPI/Video Counters Speed verification        |
;-------------------------------------------------------------
;          5  |  Serial: Internal Loopback                    |
;-------------------------------------------------------------
;   |      6  |  Real Time Clock                              |
;-------------------------------------------------------------
;   |      7  |  Battery Ram                                  |
;-------------------------------------------------------------
;          8  |  Front Desk Bus Processor & Rom               |
;-------------------------------------------------------------
;         09  |  Shadow register                              |
;-------------------------------------------------------------
;   |     10  |  Interrupts                                   |
;-------------------------------------------------------------
;   |     11  |  -                                            |
;-------------------------------------------------------------
;   |     12  |  -                                            |
;-------------------------------------------------------------
;   |     13  |  -                                            |
;-------------------------------------------------------------
;   |     14  |  -                                            |
;-------------------------------------------------------------
;     MSB:15  |  -                                            |
;-------------------------------------------------------------
 PAGE
```

Each test is followed by a brief description on the next page:

ROM Tests:

System Self Test errors are in the format: 'System Bad: AABBCCDD'

Serial Tests (J. Reynolds)

The serial chip registers are tested for Read/Write. Then an internal loop back test is performed.
Error code AA= 06. BB is as follows:

BB= 01: Register R/W
04: Tx Buffer empty status
05: Tx Buffer empty failure
06: All sent status fail
07: Rx char available
08: Bad data

Ram 1 (R. Carr) Moving Inversions

This checks bank 0 ram pages 0 thru' 4 non-destructively and then tests the remaining ram in bank 0. This is repeated for banks $01, $E0, $E1.
Error code AA= 03. BBCC= Address

ROM Checksum (KRG)

This computes the checksum of Banks $FE & $FF and compares it against a known good value.
For a fail "R.M" appears on top LHS of screen. This is done as there is a reasonable chance that the system will crash before printing the error code.
Error code AA= 01. BB= Failed checksum. If DD=1 the the test encountered bad ram and the error code is a ram error code similar to the MOVIRAM errorcodes.

Speed (KRG)

Here the relative speed of the video counters is compared to the system speed in fast and slow modes. It checks that the system is capable of switching speed, that the FPI slows down when accessing the Mega // and that the video counters are working.
Error code AA= 05. BB=1 speed stuck slow, BB=2 speed stuck fast.

Battery Ram or NVR (KRG)

This tests the 255 byte Non Volatile Ram non destructively. An address uniqueness test is followed by a pattern test.
Error code AA= 08.
BB= 01 is address test and CC= bad address value
BB= 02 is memory fail and CC= pattern, DD= address

Soft Switches/STATEREG (KRG)

This tests all soft switches by setting/testing and clearing/testing. Eight of the softswitches have an equivalent bit in the STATEREG and these are also tested. All combinations of setting/clearing a softswitch directly and with the STATEREG are tested. Error code AA= 02, BB= STATEREG bit, CC= Read softswitch address.

Front Desk Bus (KRG)

Reads the entire FDB processor rom into Vegas memory and computes a checksum. If this is as expected then the FDB processor is functional and all the language layouts are correct. Test works with REV 2 and subsequent revs of FDB processor.
Error code AA= 09, BBCC = Bad checksum found. If DD=01 the the FDB toolcode encountered a fatal error and no checksum was computed.

Ram Address (RCarr)

This is a ram test testing unique addressing capability of Vegas ram. Address uniqueness between banks is also tested.
Error code AA= 04, BB= Failed Bank No, CC= Failed bit.

Clock Test (KRG)

Performs a R/W test on the 32 bit clock register. The time is restored after the test (to within a second).
Error code AA= 06, BBCCDD not used. If DD=01 then a fatal error occurred and the test was aborted.

Shadow Register (J. Reynolds)

Tests the functionality of the shadow register.
Error code AA= 0A.

Interrupts (J. Reynolds)

Tests Mega II and VGC capability of generating interrupts.
Error code AA= 0E.

BB= 01: VBL interrupt timeout
BB= 02: VBL IRQ status fail
BB= 03: 1/4 SEC INTERRUPT
BB= 04: 1/4 SEC INTERRUPT
BB= 05: ---
BB= 06: VGC IRQ
BB= 07: SCAN LINE

Burn In error

Errcode = AA= FF, BBCCDD not defined

This error code should only occur when running burn in diagnostics (ie when pins 9 or 10 on the burn in connector are grounded). This code means that NVR was not correctly set up (by the pre BI tester) for Burn In or that the Battery ram is bad.

Calling rom tests using Test Pointer Table:

The test pointer table resides at DIAGNOSTICS+2. Currently the diagnostics begin at $FF7400 (this is not expected to change) so the table begins at $FF7402. The first byte is the size of the table followed by the 2 byte pointers themselves. All tests pointed to are in bank $FF. For Alpha 2.0 the pointer table starts at $FF7430. For Beta 1.0 and later it's at $FF7402.

To call a diagnostic routine simply JSL to the address pointed to by the table. On return the carry flag is clear if the test passes or set if it fails. For a fail, the error status is stored in the 5 TST.STATUS registers - see source code for location (currently at $000315 and is not expected to change). The error codes are the same as those in self test. You should clear these registers before calling the test routine. Enter with Data Bank = $00, Native mode and 8 bit data and index. Return will be in Native mode with the M, X and data bank in unknown states. Note that the main ram tests are destructive above $0400 in all banks ! Outside of the ram tests all other tests use memory in bank $00 from $0000 to $1FFF.

```
 SKP 2
*****************************************************************>
*
* Test pointer table
* Points to tests in Bank $FF
* Tests must return with RTL instruction
* ( This allows disk s/w to look up the pointer table and
*  call the tests from any bank ).
*
*****************************************************************
 SKP 2
TST.TAB DFB TST.TAB.E-*-3 ;Number of pointers times 2
 SKP 1
 DW ROM.CHECKSUM  ;Test 1  128K rom
 DW MOVIRAM  ;Test 2  Ram: Moving Inversions
 DW SOFT.SW  ;Test 3  Mega II & Statereg softswitch test
 DW RAM.ADDR  ;TEST 4  Ram: Addressing
 DW FPI.SPEED  ;Test 5  Fpi fast, slow mode check
 DW SER.TST  ;Test 6  Serial Chip
 DW CLOCK  ;Test 7  Real Time Clock
 DW BAT.RAM  ;Test 8  Battery ram
 DW FDE  ;Test 9 Front Desk Bus
 DW SHADOW.TST  ;Test 0A Shadow
 DO SEQ.DEBUG  ;Do following test for debug only
 DW TEST1  ;Fails if a key is pressed
 FIN
 DW CUSTOM.IRQ  ;Test 0B Interrupts
 SKP 1
TST.TAB.E EQU *  ;End of test pointer table
 DW EXT.SEQ  ;Pointer for Disk s/w
 SKP 1
EI.MASK EQU %1111111111111111  ;Tests to run if in EI and BTN0 = 1
SELF.MASK EQU %1111111111111111  ;Tests to run in self test
EI.SELF.MASK EQU %1111111100111111  ;Tests to run in EI and BTN0 =0
 SKP 1
```

Using Battery Ram (NVR) for Burn In

The pre Burn In tester is required to set up the following NVR registers:

BI.STATUS : Set to zero
BI.COUNTER : Set to zero
BI.PASSES : Set to zero
BI.FAILS : Set to zero
BI.LASTF : Set to zero
CYCLES.MAX : Set to number of BI test cycles. Currently this should by 26 for a 20 minute power on cycle time.
VALID.CHK : Set to $CBD2C7
BI.ALT.MASK : Set what tests to run. Normally $FFFF for all tests.

Writing to NVR (Pre Burn-In tester)

The registers above start in location $A1 in NVR. The easiest way to write to them is to call the TOWRITEBR hook. This copies a page of main ram starting at $E102C0 to NVR. The first register, BI.STATUS, is at location $E102C0 + $A1 = $E10361.

Reading NVR (Post Burn-In tester)

Use TOREADBR to retrieve the contents of these registers after Burn In. This routine loads NVR into a page in main ram starting at location $E102C0 so the first register is at $E10361 as before.

The locations of TOREADBR and TOWRITEBR can be found in the file BANKFF.EQUATES1 used in the bank $FF rom source. Currently TOREADBR is at $E10084 and TOWRITEBR at $E10090.