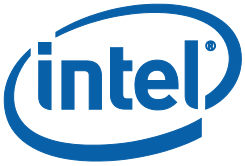


OEM Diagnostics Tool for Graphics Controllers

External Product Specification (EPS)

February 2011

Revision 1.0

Intel Confidential

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL® PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER, AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT. Intel products are not intended for use in medical, life saving, life sustaining, critical control or safety systems, or in nuclear facility applications.

Intel may make changes to specifications and product descriptions at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The information here is subject to change without notice. Do not finalize a design with this information.

The products described in this document may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

Intel and the Intel logo are trademarks of Intel Corporation or its subsidiaries in the U.S. and other countries.

*Other names and brands may be claimed as the property of others.

Copyright © 2011, Intel Corporation. All rights reserved.

Contents

Figures

Tables

Revision History

Revision Number	Description	Revision Date
1.0	• Initial release.	February 2011

§

1 Introduction

1.1 Purpose / Scope

The purpose of this document is to define the external product specification for the OEM Diagnostic Tool. The intended audience is OEM manufacturers who want to test the integrated graphics controller portion of the supported chipset devices.

1.2 Terminology

Term	Description
EPS	External Product Specification. Software interface specification from an external perspective. Typically outlines modules and their calling interface.
GC	Graphics Controller.
DVM	Dynamic Video Memory.
OEM	Original Equipment Manufacturer
PRD	Products Requirement Document

§

2 Supported Devices

The OEM Diagnostic Tool currently supports the following devices:

- Intel® SM35 Express Chipset

§

3 Hardware Requirements

The OEM Diagnostic Tool requires that the device be a supported Intel Graphics Controller.

Note: At this time the OEM Diagnostics utility does not support legacy free PC's. The DOS4GW DOS extender requires an active 8042-keyboard controller to operate properly. The DOS extender will therefore fail if used on a system that uses a USB keyboard, floppy drive and mouse. At this time there is no work around for this issue.

§

4 Software Requirements

Once the hardware requirements are met, the user must have the OEM Diagnostic Tool and the Tenberry* DOS4GW DOS extender. The OEM Diagnostic Tool comes with the DOS4GW DOS extender so there is no need to purchase a copy of DOS4GW. MS DOS 6.22 or later must be installed and all of the files included in the OEM Diagnostic Tool GC package must be in the same directory. A release of the OEM Diagnostic Tool GC will include all of the required software to test the supported hardware.

There is a unique OEM Diagnostic executable for each of the supported chipsets:

- IOAKDIAG.EXE for the Intel SM35 Express Chipset.
- This utility must be executed in “True MS DOS Mode” only. It will not execute in a “MS-DOS window”.
- Because of a conflict between the DOS4GW DOS extender and EMM386.EXE, need to remove f EMM386.EXE from the system.

4.1 Free DOS alternative to MS-DOS 6.22

The OEM Diagnostics utility suite is only validated and supported on MS DOS v6.22 or greater. However, the development team has performed informal testing with a DOS alternative called FreeDOS. This version is available on the internet at www.freedos.org .

If this version of DOS is used and issues are encountered, please verify the issue by running the utility on a supported version of Microsoft DOS before requesting support.

§

5 Installation

The OEM Diagnostics utility is distributed in a self-extracting ZIP file format. Once the distribution package has been downloaded, it MUST be executed under a Windows environment to obtain the actual binaries and documentation. The distribution package EXE will NOT run from DOS.

5.1 Installation Package Extraction

5.1.1 Overview:

- Download the distribution package to a machine running a Windows operating system.
- Execute the file named “iXXXdiag.exe” on the Windows machine to extract the contents of the self-extracting zip file.
- In the resultant dialog box, accept the default destination drive and directory or browse to another. Refer Figure 1
- Press the “Unzip” button to extract the DOS binaries and documents now**.**

OR

- If WinZip installed, press the “WinZip” button to perform the extraction using WinZip instead.
- Copy the extracted files from the destination media to the test machine running MS DOS 6.22 or greater.
- The extracted files can now be executed in a true MS DOS environment.

5.1.2 Details

Using a Windows based machine (Windows 95, 98, NT, 2000 or XP), execute the installation package named **“iXXXdiag.exe”.** The ‘X’ characters should be replaced with the chipset numbers or letters representing specific chipset in use. In this case it is “865”. Figure 1 shows a typical WinZip Self-Extractor dialog box. Use the **[Browse]** button to choose a destination drive and directory for the resultant files, if the default is unacceptable. The example shown will extract the package contents to a subdirectory called “**i865diag**” on the A: drive when the “Unzip” button is pressed.

Figure 1. WinZip Self Extractor Dialog

NOTE: Note: The distribution package CANNOT be extracted to the root directory of a drive by specifying **"A:\"** in the **[Unzip to folder:]** text box.

A subdirectory **MUST** be specified or an error will occur. This is an issue with the WinZip Self-Extractor application and not with the OEM Diagnostics utility. For the utility to run in the root directory, extract the contents to the default path and move the resultant files and directories manually, using Windows Explorer or use the full version of WinZip.

If WinZip is installed **[Run WinZip]** button will execute the full version of WinZip and display the distribution package contents. From the WinZip application, the user can then extract the package contents. The full WinZip application can extract the package contents to the root directory of any drive.

Note:

1. Most of the documentation provided with this utility requires a graphical user interface such as Windows* or Linux* to view.
2. HTML and PDF files may not be viewable in a true DOS environment and will require Windows or Linux to view them properly.
3. Text documents can be viewed directly in DOS using any appropriate text editor.
4. All files generated by this utility when running under DOS, such as LOG and INI files, can be viewed by any DOS, Windows, or Linux text editor.

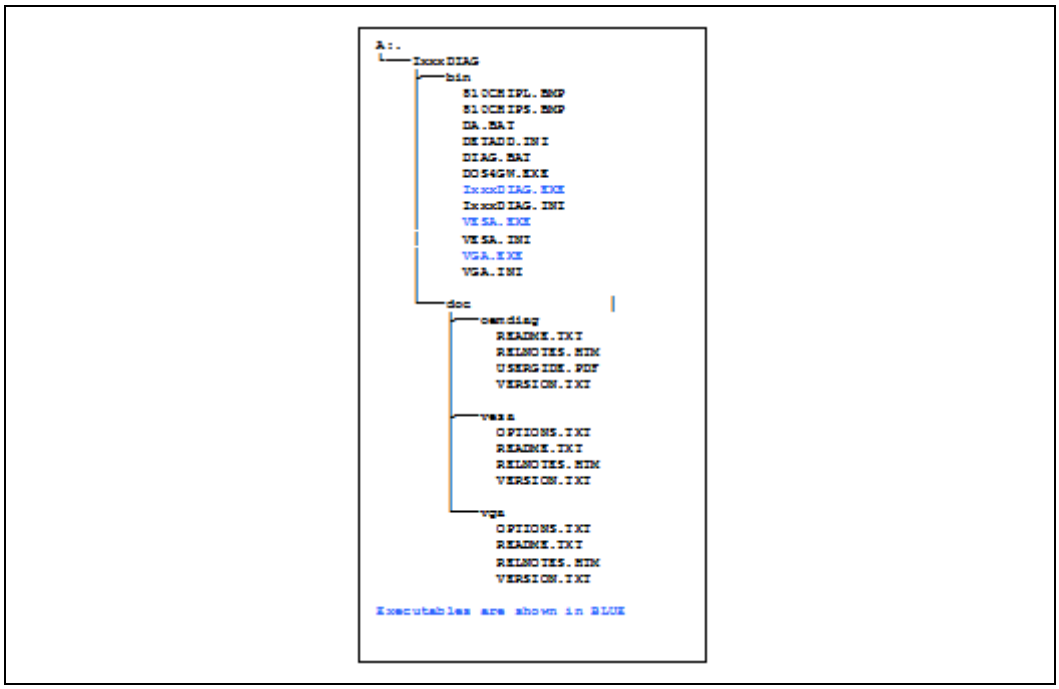
5.1.3 Installation Directory Structure

The distribution package contains a built-in directory structure which is recreated upon extraction to the destination media.

1. DOS executables and all files required for execution are extracted to the **"\BIN"** directory.
2. The DOC subdirectory is NOT required for the proper execution of the applications contained in "**\BIN**" directory.

3. If additional space is needed on the destination media, the entire **"\DOC"** directory can be removed. This is especially useful when the destination media is a standard 1.4 4MB floppy disk
4. All documentation files are extracted to the **"\DOC**" subdirectory and are further categorized into subdirectories based on the application name. For example, the documentation for the OEM Diagnostic utility will be located in **"\DOC\OEMDIAG"** directory and the **"VESA"** utility documents will be located in the **"\DOC\VESA"** directory. This example assumes the distribution package was extracted to the root directory of the media.
5. Figure below shows the resultant directory structure of an extraction to the A: drive. Notice all the binaries are located in **"A:\IxxxDIAG\bin"**. Change to this directory to run the any of the utility applications.
6. The **"DOS4GW.EXE"** file is not part of the utility suite and is actually a support file required by **"IxxxDIAG.EXE"**. It will not function without it.

Figure 2. Installation Directory Structure



The example directory structure will be similar across all versions of OEM Diagnostics. To obtain the actual distribution package contents, refer to "README.TXT" file located in the appropriate **"\DOC"** subdirectory.

Additional files may appear in the **"\BIN"** directory after any of the utilities have been run. Each utility will generate various output files for later review and editing. Generated files may have the following extensions: **LOG, BIN, CRC**. All generated files can be viewed with any DOS, Windows or Linux text editor.

5.1.4 Absolute Minimum File Requirements for Executables

This list is being provided for OEM's that require the maximum amount of disk space on the media where OEM Diagnostic utilities will reside. The list will be helpful in cases where a boot floppy has been created and other OS components or utilities need to be on the same media as OEM Diagnostics. By eliminating unnecessary files included with OEM Diagnostics additional disk space will be available.

Intel recommends at least 50K be available on the execution media to support the writing of LOG files that the OEM Diagnostics utilities automatically generate.

Below is a list of the utilities included with the OEM Diagnostic Utility suite. Also listed are the absolute minimum files required to execute each utility. Although all INI files can be excluded from the execution directory this is NOT recommended. Without an INI file for each utility, all test parameters that differ from the program's defaults must be specified on the command line. Since the command line has a set character limit, specifying all the required parameters on the command line may not provide the end user with the functionality desired.

Table 1. Minimum File Requirements for Executables table

Utility File Name	Description	Required Support Files	Optional Support Files
IxxxDIAG.EXE	Main diagnostic utility	DOS4GW.EXE 810CHIPS.BMP 810CHIPL.BMP	DIAG.BAT IxxxDIAG.INI
VESA.EXE	VESA utility	NONE	VESA.INI
VGA.EXE	VGA utility	NONE	VGA.INI

NOTES:

1. File sizes indicated are approximate and will vary slightly by platform and version.
2. INI file sizes represent the approximate size at the time of release. OEM modified INI files may increase or decrease INI file sizes.
3. "loop" parameter and "PASS/FAIL" screen displays will not function without this file.
4. Without the "INI" file, program defaults will be used. All test parameters must be submitted via command line. Missing "INI" file error message maybe displayed if missing, but basic execution not affected.
5. Program will not execute without this file.

§

6 Executing the OEM Diagnostic Tool

6.1 Execution with command line arguments

The OEM Diagnostic Tool accepts arguments given directly through the command line. The arguments are used to enable or disable specific tests as well as configure how the OEM Diagnostic Tool operates. See Section 6 for descriptions and correct syntax of the arguments.

Applies to All Versions

- The DOS/4GW extender limits the available command line length. Passing parameters on the command line where there are between 103 and 106 characters, will cause truncation of the commands after the 103rd character. This scenario will most likely cause OEM Diagnostics to generate an error due to the truncation of the last parameter.
- If more than 106 characters are passed, the DOS/4GW executable will generate an exception fault or an error message and then terminate. There is no work around for this issue besides limiting the maximum command parameters to 102 characters.
- This limit equates to approximately 6 average length commands composed of 3 values each such as "-t render_blt on". On Intel® 915G and later platforms, the DIAG.BAT file is modified to generate a DOS/4GW error message and terminate instead of generating an exception fault and terminating. This batch file will be included with when new releases are produced.
- The MS DOS batch file limitation of 9 parameters specified on the command line has been removed. The DIAG.BAT file was updated so that this limitation is no longer an issue except for the DOS/4GW limitation already mentioned. In addition the batch file supports a 'loop' parameter.
- This parameter causes the OEM Diagnostics to continually execute until there is a failure or until the user aborts testing using the CTRL-BREAK key sequence. The new batch file supports all platforms and all versions without editing. Just replace the current DIAG.BAT file on any supported system and execute it. The new batch file does not affect the functionality of the IxxxDIAG.INI file.

Syntax Examples:

- diag –t all on loop // Runs all tests until failure or user abort.
- diag loop –t all on // Same as above. Placement of 'loop' parameter irrelevant.
- diag loop // Runs all tests specified in the INI file until failure or user abort.
- diag –t all on LOOP // **ILLEGAL** – 'loop' parameter must be all lower case.

- ixxxdiag –t all on loop // **ILLEGAL** – 'loop' parameter is not supported in the executable.

This batch file will be included in all future releases of OEM Diagnostics.

Note: The MS DOS command line is typically limited to 128 characters in length. To specify more than 128 characters on the command line Intel recommends to use the configuration file. See next section for an explanation of this file.

6.2 Using a configuration file to pass arguments

The "IOAKDIAG.INI" configuration file may be used in place of or in conjunction with command line arguments. The syntax for passing arguments from the configuration file to the Diagnostic Tool is the same as using command line arguments. However, using the configuration file simplifies the process of multiple tests. See Section 6 for descriptions and correct syntax of the arguments. It is important to note that switches from the "IxxxDIAG.INI" configuration file (if present) will be read sequentially followed by arguments from the command line. So, the last setting of a particular test is the one that is used.

6.3 Execution without arguments

If no command line or configuration file arguments are used, all tests will be performed and all logs will be recorded in the "IOAKDIAG.LOG" file. All setting defaults will be used.

6.4 Optimizing Execution Speed

By default OEM Diagnostics will run all available tests with an approximate run time of 5 minutes. Several factors can cause OEM Diagnostics to execute more slowly than expected. The execution media, video BIOS and the tests themselves will cause significant changes in the total execution time of OEM Diagnostics. The following sections discuss various methods of decreasing execution time.

6.4.1 Improving Application Performance

In some situations, an OEM may want to decrease testing time, without significantly affecting the validity of the testing process. Below is a list of methods to decrease testing time without a significant impact to the validity of the testing process.

Please refer to Chapter 7 for more information about particular tests and options.

Do not execute the OEM Diagnostics from a floppy disk. OEM Diagnostics writes data to several log files after the completion of each subtest. Since writing to a floppy is very slow, changing the execution location to a hard drive or network share can cut the execution time in half. See Table 3 for a list of relative execution times versus the execution media.

Reduce the number of tested video modes tested using the "-o mode nnn" option. The 3 graphics tests, Hardware Cursor, 2D Blit and the 3D Render tests make up a significant part of the total test time. By default each test uses all applicable video modes, as reported by the video BIOS, to perform these tests. The OEM can decrease testing time by limiting which video modes are tested with the **"-o mode"** option.

A typical example would be to include the **"-o mode 111,114,117"** command in the INI file to limit the graphics tests. In this example only the 3 listed modes will be tested. When using this option, be careful not to limit testing to an unavailable video mode. At least 1, available and appropriate video mode MUST be specified. If a test is executed and there are no available modes, due to the use of the "-o mode" option, an error will be generated. Also note that the graphics UMA size setting, available in CMOS, will affect the number video modes available to the system. A setting of 16Meg will typically have more video modes available than a UMA size of 1Meg. Appropriate video modes are shown in the table below.

Table 2. Appropriate Video Modes for Graphics Tests

OEM Diagnostics Tests	Tests all video modes with resolutions greater than or equal to:
Hardware Cursor (render_cursor)	640 x 400 pixels with color depths of **15 or 16 bits**.
2D Blit (render_blt)	640 x 400 pixels with a color depth of **16 bits**.
3D Render (render_3d)	640 x 400 pixels with a color depth of **16 bits**.

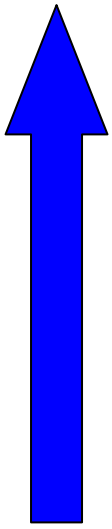
- **Reduce the number of available video modes with a CMOS setting.** As shown above, the number of video modes available will affect the number of graphics tests executed. Some BIOS's will allow the user to alter the amount of graphics memory allocated to graphics. This CMOS setting is typically called UMA Size, and has values such as 1M, 4M, 16M, 32M etc. Reducing the size of the UMA will, in most cases, reduces the number of available video modes and therefore decreases the number of graphics tests performed.
- **Limit the number of iterations that the Memory Burst uses with the "-o burst_mask nnn" option.** The memory burst test is the longest memory test executed. It executes 7 times with byte block sizes of 1, 2, 4, 8, 16, 32 and 64 bytes. A byte block size of 1 executes the slowest and a size of 64 executes the fastest. Removing the slower byte block sizes will decrease the memory burst test time. For example, using the **"-o burst_mask 96"** option, will only test using the byte block sizes 32 & 64 and skip the byte block sizes 1, 2, 4, 8 and 16.
- **Options to reduce the amount of memory tested during memory tests.** The two longest memory tests, memory stuck bit and memory burst tests are affected by several options. By changing the amount of memory tested, typical default is 12 Mbytes (0xBFFFFF), the testing time will be reduced.
  - **"memory_test_start" option:** Default is 0. Increase to a value less than 0xBFFFFF to decrease the amount of memory tested.
  - **"memory_test_end" option:** Default is 0xBFFFFF. Decrease to a value greater than 0 to decrease the amount of memory tested.
  - **"memory_test_percent" option:** Default is 100. Changing to 50 will only test half as much memory as the default.

- **Eliminate unnecessary tests.** Some OEM's use several tools to validate platforms besides OEM Diagnostics. Since all tools are not created equal, there may be some redundancy when it comes to testing memory or graphics. If for example another application is used to test memory, the OEM Diagnostic's memory tests could be disabled to save execution time. Simply edit the INI file or place the appropriate "-t *test_name* off" command on the command line to disable a test. To shut off more than a couple of tests, it is recommended that the INI file be edited instead of using the command line.
  The "INI" file lists all the available tests with each line preceded by a ';' semicolon. The semicolon is used to remark-out commands and options. To disable any test, simply locate the test name in the "INI" file and remove the semicolon remark character. See the examples below:
  ;-t pci_info off (command has no effect – uses the ';')
  -t pci_info off (pci_info test turned off)

6.4.2 Installation Media versus Execution Speed

The speed in which the OEM Diagnostic tool executes is directly related to the speed of the media where it is installed. Since the tool writes to a log file after each sub-test, the overall execution speed is determined by the relative speed of the media were the log files are written. The table below shows the relative execution speed based on the media in which the utility is installed.

Table 3. Relative Execution Times vs. Media table

Speed	Media	Notes	Limitations
	MS DOS Ram Drive	Typically offers the fastest execution speed.	Log files written to ram drive should be copied to a non-volatile media before system shut down.
	Hard Drive	Fastest non-volatile media	
	Bootable CD	**Limited Support**	May have to provide an alternate, writable location for main log file generation. Use the "-f logfile" option. See next section for more info.
	CD	**Not Supported**	Execution is not supported because the application will attempt to write LOG and CRC files to the READ-ONLY media. See next section for more info.
	External Hard Drive		Support based on system configuration.
	Network Share		Network setup dependant.
	USB 2.0 Flash Drive (not write protected)	Maybe faster than hard drive medias.	Support based on system configuration. Have not formally tested this media yet.

Speed	Media	Notes	Limitations
	USB 1.1 Flash Drive (not write protected)	Fairly slow execution due to USB 1.1 bandwidth.	Support based on system configuration.
	Floppy Drive	Lowest speed, but offers low cost, low maintenance testing.	Limited media space available.

6.4.2.1 CDR's and other READ-ONLY Medias

OEM Diagnostics must reside on a media that allows reading and writing. Although the main log file can be redirected to another writable location with the "-f" option, there are no options for redirecting the output to the METRICS.LOG and IxxxDIAG.CRC files. These files are by design, written to the execution directory.

Failures that will be encountered if attempting to run from READ-ONLY media include:

- PROGRAM ABORT – Unable to open metrics file
- ABNORMAL TERMINATION
- CRC Checksum error.

6.4.2.2 Bootable CD's

Bootable CD's will suffer from the same problems that regular CDR's do if created improperly. If the user simply takes a bootable CD project and adds the utility files to a directory on that CD, it will fail just as a regular CD would. This is because once the CD has booted the system; the media is still just another READ-ONLY media. OEM Diagnostics will fail as soon as it attempts to write its log files back to the device.

When a bootable CD starts a system, it performs a little bit of magic during the boot process. Typically the boot CD contains an image of a bootable floppy which is loaded into memory from the CD. This image boots just like a floppy drive, but off the CD itself. In fact, this boot image will appear to the system as the A: drive but in reality is a RAM Disk containing the boot image loaded from the CD. When the system finally completes its boot process, a quick investigation will reveal that the A: drive is in fact a RAM drive containing everything from the drive image on the CD. If the system had an actual floppy drive, which is usually the A: drive, it will be bumped up to become B: drive.

The trick to making OEM Diagnostics execute from a CD is to first create a complete bootable floppy containing the OEM Diagnostics utility and all the require boot files. You will then use this bootable floppy to create the bootable CD. This of course will require the use some third party software like Roxio's Easy CD Creator, or Nero. Once the floppy is created, attempt to boot a system with it. Once booted, verify you that you can execute the OEM Diagnostic utility without any errors from the floppy disk. If all goes well, you can now use your CD Authoring software to create the bootable CD from this floppy. Below are the steps involved.

- Start by creating a DOS 6.22 or greater boot disk using the DOS command "**format a: /s**"
- Copy the OEM Diagnostic binaries from the "**\BIN**" directory to the floppy drive.

- Edit the floppy boot files, “AUTOEXEC.BAT” and “CONFIG.SYS” to support any required functionality.
  You must add support for CD Rom access if you want to access any files on the CD Rom not contained on the floppy. Once the CD Rom boots the system, the floppy image will become the A:\ drive and the CD Rom will be in accessible unless you have loaded a driver in the “CONFIG.SYS” file and “MSCDEX” application in the “AUTOEXEC.BAT” file.
- Test your boot floppy to verify that everything works as intended. Also if you need CD Rom access, verify this is operational too.
  - Special Note about the “**iXXXdiag.crc**” file. While testing the floppy a “**CRC**” file will be created. Unless you intend to use this file as the reference CRC file, for future testing, it should be deleted from the floppy before creating the boot CD. If you do not delete the file, the values contained in it will be used to validate all systems you test with the bootable CD.
  - When you are done testing the floppy delete all the “***.LOG**” files from the “\BIN” directory on the floppy.
- Use your CD Authoring software to start a Bootable CD project. At some point, you should be prompted to insert the boot floppy you created earlier.
- Add any additional files to the CD, but remember you will not be able to access them unless your boot floppy contains the correct drivers for CD access under DOS.
- Burn the CD and test it.

§

7 OEM Diagnostic Tool GC Arguments

7.1 General

Note: When specifying hex numbers as an input parameter to the arguments, use of a lagging 'h', (for example: 1234h) is invalid and will generate an invalid option failure message. Use a leading '0x' instead (0x1234).

-i[ni] <ini_file_name>

- Specify name of INI file. Default is "IxxxDIAG.INI". Specifying this inside an INI file will have no effect.

-f[ilename] <log_file_name>

- Specifies the name of the text file in which the diagnostic's output will be logged. If this switch is not used, the log file name defaults to "IOAKDIAG.LOG"

  Syntax Example: -f OUTPUT.LOG

-? or –h[elp]

- This switch brings up the command line help. All of the commands are listed with brief descriptions.

  Syntax Example: -?

; <comment>

- A ';' at the beginning of a line in the IxxxDIAG.INI configuration file will cause the diagnostic to skip the remainder of that line. This allows you to include comments in the configuration file.

  Syntax Example: ;Below `test inserted on 2/1/99`

7.2 Tests

The `–t <test_name> on|off` selectively enables or disables the various OEM Diagnostics Tool tests. By default, all tests are enabled. Below are the test switches that can be used with the `–t` switch. Tests are listed in the order that they are executed.

all

Using this will tell the Diagnostic Tool to use all of the tests on your Intel hardware. This is very useful, allowing a user to run all of the tests without having to specify each individual test through switches. By default, this option is turned on.
Syntax Example: `-t all on`

pci_info

This test scans the PCI bus and reports all of the devices that are present.
Syntax Example: `-t pci_info on`

cpu_info

This test reports the processor's family, model, stepping, as well as the amount of system memory.
Syntax Example: `-t cpu_info on`

memory_info

This test gives detailed information about the graphics device's graphics memory.
Syntax Example: `-t memory_info on`

bios_info

This option reports information about the video BIOS, including version number and supported video modes.
Syntax Example: `-t bios_info on`

ddc_info

This tests the graphics adapter and monitor's DDC (Display Data Channel) information. You must have a monitor that supports DDC in order to execute this test and receive a PASS indication.
Syntax Example: `-t ddc_info on`

bios_crc

This test calculates a checksum of the adapter video BIOS and compares it to the value stored in the shadowed BIOS space.
Syntax Example: `-t bios_crc on`

memory_data_line

This test checks the graphics memory data path by performing a walking 1s and 0s test on the last address in the memory range. This address will be either DVM or local memory, if installed.
The test performs the following (this description valid starting with version 2.3 of utility):

Set pattern equal to 0x00000001.
Loop until pattern equals zero.
Write pattern to address, then read and compare.
Write ones complement of pattern to address, then read and compare.
Shift pattern left one place, filling places right of the one with zero.
Syntax Example: -t memory_data_line on

memory_address

This test performs a check on the graphics memory address bus. The range of addresses tested will be 4 Mbyte if local memory is installed or 12 Mbyte if local memory is not installed. Memory at address offsets 0, 1, 2, 4, … (end of memory) is first initialized to zero. Then a pattern of 0xFF is walked through the above listed offsets and tested.
The test performs the following (this description valid starting with version 2.3 of utility):

```
#define PATTERN 0xff
#define PASS 0
#define FAIL 1
unsigned long Add1, Add2, dwTestSize, dwStartAddr, fault = PASS;
unsigned char cv, *pc = (unsigned char *) dwStartAddr;

// Initialize tested locations
for( Add1 = 0; Add1 < dwTestSize; Add1 = (Add1) ? (Add1 << 1) : 1 )
   pc[Add1] = 0;

// Walk the address bus
for( Add1 = 0; (Add1 < dwTestSize) && !fault; Add1 = (Add1) ? (Add1 <<
1) : 1 )
{
   pc[Add1] = PATTERN;    // Set address to test to pattern.
   for( Add2 = 0; (Add2 < dwTestSize) && !fault; Add2 = (Add2) ? (Add2
<< 1) : 1 )
   {
      cv = (Add1 == Add2) ? PATTERN : 0;
      fault = (cv == pc[Add2] ) ? PASS : FAIL;
   }
   pc[Add1] = 0; // Set address to test back to zero.
}
```

memory_stuck_bit

This test checks the display adapter's graphics memory by writing a series of patterns to the memory, and making sure the values are properly written to

memory. This test checks all installed local memory on the adapter and up to 12MB of DVM, when applicable.
The test performs the following for each address (this description valid starting with version 2.3 of utility):

write 0xAAAAAAA, then read and compare.
write 0x55555555, then read and compare.
write 0xFFFFFFFF, then read and compare.
write 0x00000000, then read and compare.

Syntax Example: -t memory_stuck_bit on

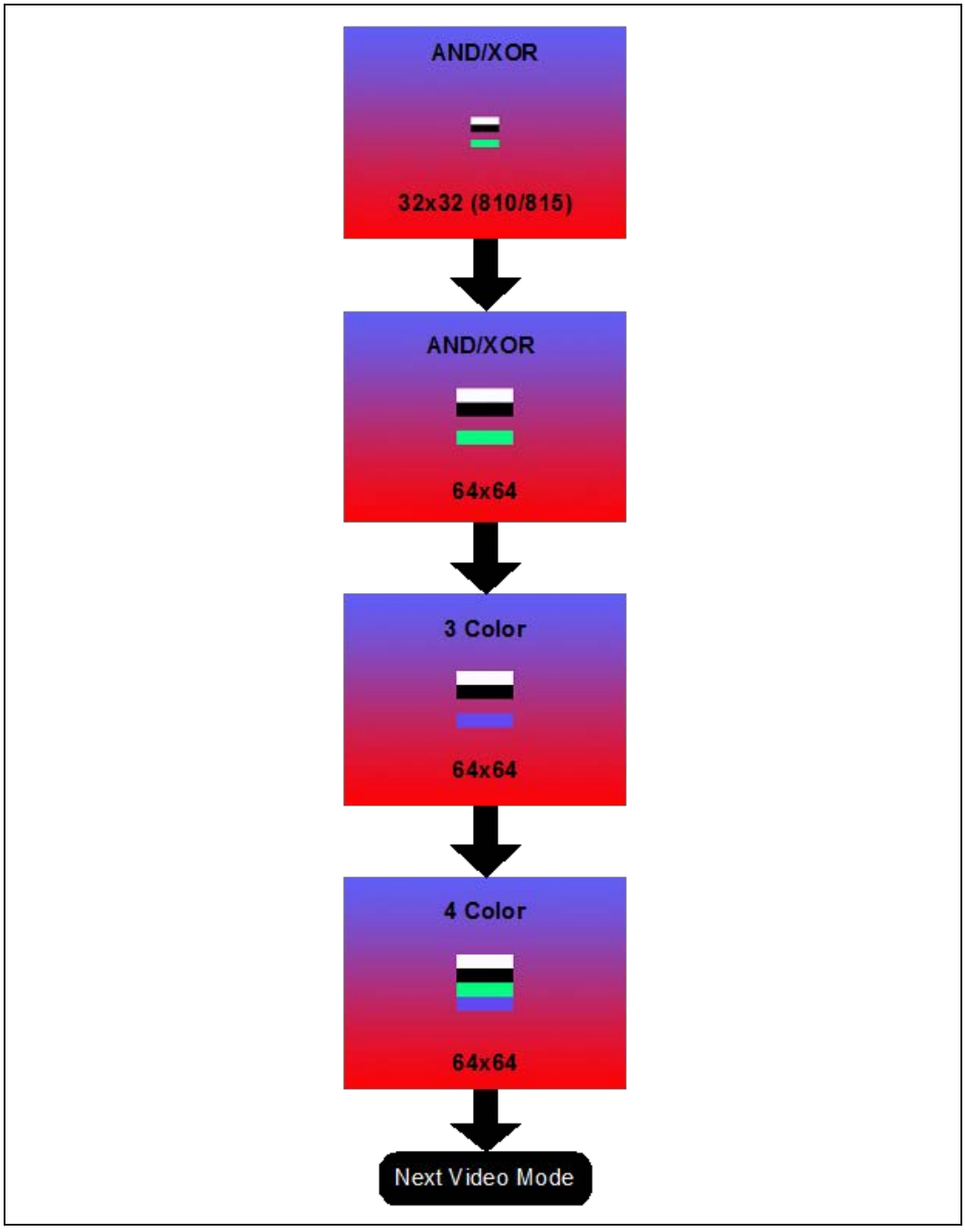
Figure 3. Typical Memory Stuck Bit / Burst Test Display

memory_burst

This test checks the local memory by writing in blocks of 1, 2, 4, 8, 16, 32, and 64 bytes, then verifying that the block was written correctly. This test may take several minutes to complete. This test checks all installed memory on the adapter and up to 12MB of DVM, when applicable.
The test performs the following for each address:

For each block size, where block size = 1, 2, 4,. . .,64; throughout the range of memory
Initialize block with 0x5A, read and compare
Initialize block with 0xA5, read and compare
Syntax Example: `-t memory_burst on`

Figure 4. Hardware Cursor Test Display



registers

This test checks the display adapter's registers by writing 1s and 0s to verify they are working correctly. It also takes into account any side effects one register has on another one.
Syntax Example: `-t registers on`

render_cursor

This test exercises the Intel devices' hardware cursor overlaying functionality. For each supported video mode, a gradient background will be displayed, blue at the top to red at the bottom. The cursor will be rendered as a four-color rectangle in the middle of the screen. One rectangle is possible (three on IxxxDIAG.EXE). These rectangles are described below:

32 x 32 x 2bpp AND/XOR 2-plane mode:

32 pixels wide by 32 scan lines in the center of the screen. The first 8 scan lines of the rectangle will be displayed as cursor color 0 (white). The second 8 scan lines will be displayed as cursor color 1 (black). The third set of 8 scan lines will be transparent (pixels of the main display image behind the cursor shows through. The fourth set of 8 scan lines will be transparent but inverted (pixels of the main display image behind the cursor shows through with inverted color.

64 x 64 x 2bpp AND/XOR 2-plane mode:

64 pixels wide by 64 scan lines in the center of the screen. The first 16 scan lines of the rectangle will be displayed as cursor color 0 (white). The second 16 scan lines will be displayed as cursor color 1 (black). The third set of 16 scan lines will be transparent (pixels of the main display image behind the cursor shows through. The fourth set of 16 scan lines will be transparent but inverted (pixels of the main display image behind the cursor shows through with inverted color.

64 x 64 x 2bpp 3-color and transparency mode:

64 pixels wide by 64 scan lines in the center of the screen. The first 16 scan lines of the rectangle will be displayed as cursor color 0 (white). The second 16 scan lines will be displayed as cursor color 1 (black). The third set of 16 scan lines will be transparent (pixels of the main display image behind the cursor shows through. The fourth set of 16 scan lines will be cursor color 3 (blue).

64 x 64 x 2bpp 4-color mode:

64 pixels wide by 64 scan lines in the center of the screen. The first 16 scan lines of the rectangle will be displayed as cursor color 0 (white). The second 16 scan lines will be displayed as cursor color 1 (black). The third set of 16 scan lines will be cursor color 2 (green). The fourth set of 16 scan lines will be cursor color 3 (blue).

Syntax Example: -t render_cursor on

7.3 Settings

The `–o[ther] <setting_name> <value(s)>` selectively enables or disables the various OEM Diagnostic Tool settings. Below are the switch settings that can be used with the –o argument. Underlined options represent the default setting.

burst_mask <decimal value>

This setting specifies which byte block sizes should be tested. The default for this option is 127, which specifies all byte block sizes. To determine the value, simple add up the block sizes to test. For example, if only block sizes 4 and 8 are to be tested, use 12 as the decimal value. Only values from 1 to 127 are valid. Any value outside this range will generate an error.

In the below example since 96 equals 32 + 64, only the 32 and 64 byte block sizes will execute.

Syntax Example: `-o burst_mask 96`

color on|off

This setting specifies whether the diagnostic should change the background color before exiting. If this option is set to on and all of the desired tests pass, the background will change to green. If this option is set to on, and any test fails, the background will change to red. If this option is set to off or the user aborts the testing, the background will be black. If an error has been detected on the command line or in the INI file, the exiting screen color will change only if this option has been parsed before the error occurs. The default for this option is "`off`".

Syntax Example: `-o color on`

crc_file create|add|read|auto

This setting allows the user to create or add to the file that contains the reference checksum values for rendering tests. Note that CRC values are only written for tests and video modes that are selected by the user during that execution of the diagnostic. The default for this option is "`auto`" on 830M platforms and later. The default is "read".

halt_on_error on|off

This setting specifies whether the diagnostic should halt immediately upon encountering an error, or attempt to complete all desired testing and report all encountered errors at the end of testing. If all of the desired tests pass, "PASS" will be displayed to the screen. If any test fails, "`FAIL`" is displayed on the screen. Furthermore, all test successes and failures will be noted in the log file ("`IxxxDIAG.LOG`" by default). The default for this option is "`on`".

Syntax Example: `-o halt_on_error on`

local_memory on|off

This setting specifies whether the diagnostic should used display cache (local) memory as graphics memory. If this option is set to off, then this forces system memory to be used in place of local memory.

The default for this option is "on" for 810 versions prior to v2.8, and "`off`" for v2.8 and later.
The default for this option is "on" for 815 versions prior to v1.3, and "`off`" for v1.3 and later.

The default for this option is “on” for 830M versions prior to v0.5.2, and “off” for v0.5.2 and later.
All future platforms the default is “off”

Note: NOTE:

7. Set this option to off if there is a chipset that supports the use of display cache, but do not have the memory installed. Else, the memory info test will fail.
8. Local memory usage on 830M differs from previous 81x platforms. If local memory is installed and activated in BIOS then it will be mapped to the beginning of the display cache. On systems with 16Meg of local memory approximately 11 meg is available for display cache. Approximately 1 meg of system memory will be mapped to the end of the display cache, for a total display cache size of 12Meg. For systems with more than 16Meg of local memory, the entire display cache should be mapped to local memory. Specifying the **local_memory off** when it is present and active prevents the use of local memory for the display cache and does not generate an error. In this case 12Meg of system memory will be used for display cache.

   If the BIOS is set to use UMA 512K, 1Meg, or 8Meg instead of local memory, the **local_memory off** option must be used to prevent test failure as described previously.

   Syntax Example: -o local_memory on

memory_test_start <hex address>

This setting specifies the address (relative to the start of graphics memory) of the first byte to test when doing memory testing. Valid addresses range from 0 hex to BFFFFF hex. If no address is specified, or if the address is outside of the valid range, this setting will default to 0 hex. The value specified is always interpreted as a hex value. The leading 0x is optional. This setting is used for the stuck bit and burst tests only.

Syntax Example: `-o memory_test_start 0 or –o memory_test_start 0x0`

memory_test_end <hex address>

This setting specifies the address (relative to the start of video memory) of the last byte to test when doing memory testing. Valid addresses range from 0 hex to BFFFFF hex, depending on the amount of local memory installed. If no address is specified, or if the address is outside of the valid range, this setting will default to the address of the last byte of graphics memory. The value specified is always interpreted as a hex value. The leading 0x is optional. This setting is used for the stuck bit and burst tests only.

Syntax Example: `-o memory_test_end BFFFFF or –o memory_test_end 0xBFFFFF`

memory_test_percent < decimal value>

This setting specifies how much of the memory that the memory tests will check. The default value is 100. The value must be from 1 to 100. The value specified is always interpreted as a decimal value. This setting is used for the stuck bit and burst tests only.

Syntax Example: `-o memory_test_percent 50`

memory_test_distrib <decimal value>

This setting specifies how the memory test is distributed across the memory. memory_test_distrib will take the desired memory_test_percent and distribute it into segments of X% where X% is the value of memory_test_distrib. This is only useful if the memory_test_percent is set to less than 100. The default value of memory_test_distrib is 100. The value specified is always interpreted as a decimal value. This setting is used for the stuck bit and burst tests only.

Syntax Example: `-o memory_test_distrib 10`

metrics on|off

The metrics setting will record the time taken for each test and output the results to the metrics log file, "`metrics.log`. See Appendix B for information on how the metrics log file organizes it's results. The default for this option is "`on`".

Syntax Example: `-o metrics on`

mode <hex mode number> [,<hex mode number>,...]

This setting allows the user to specify one or more modes in which to conduct the rendering tests. Note that the 2d and 3d tests currently support 16-bit extended modes only, the hardware cursor test may also support 15-bit extended modes in addition to 16-bit modes. Also, each mode requires a minimal amount of VGA memory and must be supported by the video BIOS. If no modes are specified, all modes supported by a test will be used. The value specified is always interpreted as a hex value. A leading 0x (as in 0x127) is allowed. The default for this option is to use modes automatically detected as supported by the video BIOS.

The Valid modes are:

Table 4. Video Mode Table

VESA Mode	Resolution		Color Depth	VGA Memory Requirements
0x101	640	480	8	0.29 MB
0x103	800	600	8	0.46 MB
0x105	1024	768	8	0.75 MB
0x107	1280	1024	8	1.25 MB
0x111	640	480	16	0.59 MB

VESA Mode	Resolution		Color Depth	VGA Memory Requirements
0x112	640	480	32	1.17 MB
0x114	800	600	16	0.92 MB
0x115	800	600	32	1.83 MB
0x117	1024	768	16	1.50 MB
0x118	1024	768	32	3.00 MB
0x11A	1280	1024	16	2.50 MB
0x11B	1280	1024	32	5.00 MB
0x13A	1600	1200	8	1.83 MB
0x13C	1920	1440	8	2.64 MB
0x14B	1600	1200	16	3.66 MB
0x14D	1920	1440	16	5.27 MB
0x15A	1600	1200	32	7.32 MB
0x15C	1920	1440	32	10.55 MB

NOTES:

1. Available modes may vary by vender and video BIOS versions.
2. The mode values must be surrounded by double quotes when this option is specified on the command line.

INI file syntax example: –o mode 117,111 or –o mode 0x117,0x111

Command line syntax example: -o mode “111,117” or -o mode “0x111,0x117”

oem std|pcd (VALID WITH I830DIAG VERSION 0.5.1 OR GREATER, AND ALL LATER PLATFORMS)

The oem setting has been added to support various special OEM functional requests. It is only supported on the 830M and later platforms. Two functions are supported, STD and PCD (Standard, PC-Doctor®).

STD: Standard mode is the default and disables all other “oem” functions. Standard mode is activated automatically and is the program’s default setting. Command syntax checking is on and active.

PCD: The PCD function supports the PC-Doctor® application by changing the return codes that OEM Diagnostics reports back to the operating system. When in PCD mode, a screen message is displayed upon exiting the application in the form “PCD ssss (n)” where ‘ssss’ is either Pass, Failed, or User Break. The ‘n’ value represents the PC-Doctor® return code. There are no log entries to indicating this mode. The PC-Doctor® verses the Standard return codes are shown below. Only the return codes shown in bold will be returned for any particular mode and are fully supported. It is recommended that when using PCD mode, the OEM Diagnostic (i8xxdiag.exe) utility be called directly without using the DIAG.BAT file, since the BAT file does not support the PCD return codes.

It is particularly important to verify the command syntax within the INI file and on the command line when using this option. When running OEM Diagnostics under the PC Doctor® application, all command syntax checking is disabled following the use of the "-o oem pcd" option. This is required because the PC Doctor® application may attempt to pass incompatible parameters to OEM Diagnostic application, causing it to fail. Any unknown commands contained in the INI file following the "-o oem pcd" option switch will be ignored. Furthermore, if an error exists in the INI file, legal commands following it may not execute properly or at all.

For proper execution it is recommended that the "–o oem pcd" option be placed at the top of the OEM Diagnostic INI file, and NOT as a command line switch passed from within the PC Doctor® application itself.

Table 5. Return Code Table

OEMDiags Result	Standard (STD)	PC-Doctor® (PCD)	PC-Doctor® defines
PASS	0	1	PCDR_RESULT_PASSED
FAIL	2	2	PCDR_RESULT_FAILED
N/A	N/A	3	PCDR_RESULT_NA
N/A	N/A	4	PCDR_RESULT_ABORTED
N/A	N/A	5	PCDR_RESULT_UNDEFINED
USER ABORT	**1**	**6**	**PCDR_RESULT_USERBREAK**

Syntax Example: `-o oem pcd`

subjective on|off

If this switch is set to "on", the diagnostic will pause after every graphical rendering to wait for the user's subjective judgment of the rendering. Pressing the spacebar or the "y" key passes the test, and pressing the "n" key fails it. If this switch is set to "off", the diagnostic will display the rendering for a minimum of 3 seconds, then move on to the next test. CRC tests of the rendered images will be performed regardless of this switch setting. The default for this option is "off".
Syntax Example: `-o subjective off`

§

Appendix A : Known Issues

Following tests fails currently:

- memory_info
- memory_address
- memory_burst
- render_cursor

§

Appendix B : Sample Configuration File

```
$Workfile: IOAKDIAG.INI $
;
;   Purpose:
;       Intel(r) Graphics Controller OEM diagnostics option file.
;
;   Environment:
;       MSDOS 6.22, DOS protected mode.
;
;   Warnings and Restrictions:
;       Copyright (c) 2010 Intel Corporation, all rights reserved.
;
;       INTEL MAKES NO WARRANTY OF ANY KIND REGARDING THE CODE.  THIS CODE
;       IS LICENSED ON AN "AS IS" BASIS AND INTEL WILL NOT PROVIDE ANY
;       SUPPORT, ASSISTANCE, INSTALLATION, TRAINING OR OTHER SERVICES.
;       INTEL DOES NOT PROVIDE ANY UPDATES ENHANCEMENTS OR EXTENSIONS.
;       INTEL SPECIFICALLY DISCLAIMS ANY WARRANTY OF MERCHANTABILITY,
;       NONINFRINGEMENT, FITNESS FOR ANY PARTICULAR PURPOSE, OR ANY OTHER
;       WARRANTY.  INTEL DISCLAIMS ALL LIABILITY, INCLUDING LIABILITY FOR
;       INFRINGMENT OF ANY RIGHTS, RELATING TO USE OF THE CODE.  NO
;       LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY
;       INTELLECTUAL PROPERTY RIGHTS IS GRANTED HEREIN.
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
-t all on
-t pci_info on
-t cpu_info on
-t bios_info on
-t ddc_info on
-t bios_crc on
-t memory_data_line on
-t memory_stuck_bit on
-t registers on
;-o burst_mask 127
;-o crc_file create
;-o halt_on_error on
;-o local_memory on
;-o memory_test_start 0
;-o memory_test_end bfffff
;-o memory_test_percent 100
;-o memory_test_distrib 1
;-o oem std
;-o metrics on
;-o subjective on
;-o mode 111
;-f ioakdiag.log
```

§

Appendix C : Sample Log File

The below sample log file was generated using an Intel® SM35 Chipset and the corresponding OEM Diagnostic tool using the configuration file specified in Appendix A. The actual output may vary depending on individual hardware and software configuration:

```
##############################################################################
##                            Oaktrail Platform                             ##
##                 Graphics Controller Diagnostic Utility                   ##
##                                Version 2.0.1                             ##
##                   Copyright(c) 2010 Intel Corporation                    ##
##############################################################################

##############################################################################
PCI Device Information (pci_info)
##############################################

Setting          Value     Description
------------------------------------------------------------------------
Class:           0x06      Bridge devices
Sub-Class:       0x00      Host/PCI bridge
Interface:       0x00      Host Bridge
Device ID:       0x4110    Whitney Point PCI Host bridge
Vendor ID:       0x8086    Intel Corporation
Subsystem ID:    0x4102    Unknown
Subsystem Vendor ID:    0x8086    Intel Corporation
Revision ID:     0x04
Interrupt Line:     0x00      IRQ0
Base Addr. Registers:   0x00000000    Not Implemented
          0x00000000    Not Implemented
          0x00000000    Not Implemented
          0x00000000    Not Implemented
          0x00000000    Not Implemented


Setting          Value     Description
------------------------------------------------------------------------
Class:           0x03      Display controllers
Sub-Class:       0x00      VGA controllers
Interface:       0x00      VGA-compatible controllers
Device ID:       0x4102    Intel(R) Graphics Media Accelerator 600
Vendor ID:       0x8086    Intel Corporation
Subsystem ID:    0x4102    Unknown
Subsystem Vendor ID:    0x8086    Intel Corporation
Revision ID:     0x04
Interrupt Line:     0x0B      IRQ11
Base Addr. Registers:   0xFF200000    Memory-Mapped Register
          0x00007FF9    I/O Register
          0xD0000000    Memory-Mapped Register
          0xCFFC0000    Memory-Mapped Register
```

```
          0x00000000    Not Implemented
          0x00000000    Not Implemented
PCI Express Device: No     0 PCI Express devices found
PCI Cap List ID's:  None      Capabilities list not implemented


Setting          Value     Description
-------------------------------------------------------------------------
Class:           0x06      Bridge devices
Sub-Class:       0x01      PCI/ISA bridge
Interface:       0x00      ISA bridge
Device ID:       0x0817    Whitney Point LPC Interface Controllers
Vendor ID:       0x8086    Intel Corporation
Subsystem ID:    0xD6FE    Unknown
Subsystem Vendor ID:    0x8086    Intel Corporation
Revision ID:     0x01
Interrupt Line:     0x00      IRQ0
Base Addr. Registers:   0x00000000    Not Implemented
          0x00000000    Not Implemented
          0x00000000    Not Implemented
          0x00000000    Not Implemented
          0x00000000    Not Implemented
          0x00000000    Not Implemented
PCI Express Device: No     0 PCI Express devices found
PCI Cap List ID's:  None      Capabilities list not implemented


Setting          Value     Description
-------------------------------------------------------------------------
Class:           0x03      Display controllers
Sub-Class:       0x80      Other
Interface:       0x00      Other display controller
Device ID:       0x080D    Video Controller
Vendor ID:       0x8086    Intel Corporation
Subsystem ID:    0x080D    Unknown
Subsystem Vendor ID:    0x8086    Intel Corporation
Revision ID:     0x01
Interrupt Line:     0x7E      IRQ126
Base Addr. Registers:   0xFF010000    Memory-Mapped Register
          0x00002101    I/O Register
          0x00000000    Not Implemented
          0x00000000    Not Implemented
          0x00000000    Not Implemented
          0x00000000    Not Implemented
PCI Express Device: No     0 PCI Express devices found
PCI Cap List ID's:  N/A    CAP ID's not accessable


Setting          Value     Description
-------------------------------------------------------------------------
Class:           0x08      Base system peripheral
Sub-Class:       0x80      Other
Interface:       0x00      Other system peripheral
Device ID:       0x0813    Whitney Point SCU DMA
Vendor ID:       0x8086    Intel Corporation
Subsystem ID:    0x0813    Unknown
Subsystem Vendor ID:    0x8086    Intel Corporation
Revision ID:     0x01
Interrupt Line:     0x0A      IRQ10
```

```
Base Addr. Registers:  0xFF11F000   Memory-Mapped Register
          0x00000000   Not Implemented
          0x00000000   Not Implemented
          0x00000000   Not Implemented
          0x00000000   Not Implemented
          0x00000000   Not Implemented
PCI Express Device: No     0 PCI Express devices found
PCI Cap List ID's:  0xF8      Not Defined
          0xF8      Not Defined
          0xF8      Not Defined
          0xF8      Not Defined
          0xF8      Not Defined


Setting          Value     Description
-------------------------------------------------------------------------
Class:           0x07      Simple communications controller
Sub-Class:       0x80      Other
Interface:       0x00      Other communication device
Device ID:       0x0801    Whitney Point SPI Ctrl 1
Vendor ID:       0x8086    Intel Corporation
Subsystem ID:    0x0801    Unknown
Subsystem Vendor ID:   0x8086    Intel Corporation
Revision ID:     0x01
Interrupt Line:     0x05      IRQ5
Base Addr. Registers:  0xFF129000   Memory-Mapped Register
          0x00000000   Not Implemented
          0x00000000   Not Implemented
          0x00000000   Not Implemented
          0x00000000   Not Implemented
          0x00000000   Not Implemented
PCI Express Device: No     0 PCI Express devices found
PCI Cap List ID's:  N/A    CAP ID's not accessable


Setting          Value     Description
-------------------------------------------------------------------------
Class:           0x07      Simple communications controller
Sub-Class:       0x80      Other
Interface:       0x00      Other communication device
Device ID:       0x0802    Whitney Point I2C 0
Vendor ID:       0x8086    Intel Corporation
Subsystem ID:    0x0802    Unknown
Subsystem Vendor ID:   0x8086    Intel Corporation
Revision ID:     0x01
Interrupt Line:     0x0B      IRQ11
Base Addr. Registers:  0xFF12A000   Memory-Mapped Register
          0x00000000   Not Implemented
          0x00000000   Not Implemented
          0x00000000   Not Implemented
          0x00000000   Not Implemented
          0x00000000   Not Implemented
PCI Express Device: No     0 PCI Express devices found
PCI Cap List ID's:  None      Capabilities list not implemented


Setting          Value     Description
-------------------------------------------------------------------------
```

```
Class:           0x07      Simple communications controller
Sub-Class:       0x80      Other
Interface:       0x00      Other communication device
Device ID:       0x0803    Whitney Point I2C 1
Vendor ID:       0x8086    Intel Corporation
Subsystem ID:    0x0803    Unknown
Subsystem Vendor ID:    0x8086    Intel Corporation
Revision ID:     0x01
Interrupt Line:     0x05      IRQ5
Base Addr. Registers:   0xFF130000    Memory-Mapped Register
          0x00000000    Not Implemented
          0x00000000    Not Implemented
          0x00000000    Not Implemented
          0x00000000    Not Implemented
          0x00000000    Not Implemented
PCI Express Device: No     0 PCI Express devices found
PCI Cap List ID's:  N/A    CAP ID's not accessable


Setting          Value     Description
-------------------------------------------------------------------------
Class:           0x07      Simple communications controller
Sub-Class:       0x80      Other
Interface:       0x00      Other communication device
Device ID:       0x0804    Whitney Point I2C 2
Vendor ID:       0x8086    Intel Corporation
Subsystem ID:    0x0804    Unknown
Subsystem Vendor ID:    0x8086    Intel Corporation
Revision ID:     0x01
Interrupt Line:     0x03      IRQ3
Base Addr. Registers:   0xFF131000    Memory-Mapped Register
          0x00000000    Not Implemented
          0x00000000    Not Implemented
          0x00000000    Not Implemented
          0x00000000    Not Implemented
          0x00000000    Not Implemented
PCI Express Device: No     0 PCI Express devices found
PCI Cap List ID's:  None      Capabilities list not implemented


Setting          Value     Description
-------------------------------------------------------------------------
Class:           0x04      Multimedia device
Sub-Class:       0x03      Unknown
Device ID:       0x080A    High Definition Audio Controller
Vendor ID:       0x8086    Intel Corporation
Subsystem ID:    0xD6FE    Unknown
Subsystem Vendor ID:    0x8086    Intel Corporation
Revision ID:     0x00
Interrupt Line:     0x7E      IRQ126
Base Addr. Registers:   0xFFAF4004    Memory-Mapped Register
          0x00000000    Not Implemented
          0x00000000    Not Implemented
          0x00000000    Not Implemented
          0x00000000    Not Implemented
          0x00000000    Not Implemented
PCI Express Device: No     0 PCI Express devices found
PCI Cap List ID's:  N/A    CAP ID's not accessable
```

```
Setting          Value     Description
-------------------------------------------------------------------------
Class:           0x0C      Serial bus controller
Sub-Class:       0x03      USB (Universal Serial Bus)
Interface:       0x20      USB2 (Intel enhanced host controller
interface)
Device ID:       0x0806    Whitney Point USB2 EHCI Controller MPH
Vendor ID:       0x8086    Intel Corporation
Subsystem ID:    0x0806    Unknown
Subsystem Vendor ID:    0x8086    Intel Corporation
Revision ID:     0x01
Interrupt Line:     0x0A      IRQ10
Base Addr. Registers:  0xFFA80000    Memory-Mapped Register
          0x00000000    Not Implemented
          0x00000000    Not Implemented
          0x00000000    Not Implemented
          0x00000000    Not Implemented
          0x00000000    Not Implemented
PCI Express Device: No     0 PCI Express devices found
PCI Cap List ID's:  None      Capabilities list not implemented


Setting          Value     Description
-------------------------------------------------------------------------
Class:           0x0C      Serial bus controller
Sub-Class:       0x03      USB (Universal Serial Bus)
Interface:       0x20      USB2 (Intel enhanced host controller
interface)
Device ID:       0x0811    Whitney Point USB2 EHCI Controller SPM
Vendor ID:       0x8086    Intel Corporation
Subsystem ID:    0x0811    Unknown
Subsystem Vendor ID:    0x8086    Intel Corporation
Revision ID:     0x01
Interrupt Line:     0x7E      IRQ126
Base Addr. Registers:  0xFFA60000    Memory-Mapped Register
          0x00000000    Not Implemented
          0x00000000    Not Implemented
          0x00000000    Not Implemented
          0x00000000    Not Implemented
          0x00000000    Not Implemented
PCI Express Device: No     0 PCI Express devices found
PCI Cap List ID's:  N/A    CAP ID's not accessable


Setting          Value     Description
-------------------------------------------------------------------------
Class:           0x08      Base system peripheral
Sub-Class:       0x05      Unknown
Device ID:       0x0807    Whitney Point SD Ctrl 0
Vendor ID:       0x8086    Intel Corporation
Subsystem ID:    0x0807    Unknown
Subsystem Vendor ID:    0x8086    Intel Corporation
Revision ID:     0x01
Interrupt Line:     0x0B      IRQ11
Base Addr. Registers:  0xFFA50000    Memory-Mapped Register
          0x00000000    Not Implemented
```

```
          0x00000000    Not Implemented
          0x00000000    Not Implemented
          0x00000000    Not Implemented
          0x00000000    Not Implemented
PCI Express Device: No     0 PCI Express devices found
PCI Cap List ID's:  None      Capabilities list not implemented


Setting          Value     Description
-------------------------------------------------------------------------
Class:           0x08      Base system peripheral
Sub-Class:       0x05      Unknown
Device ID:       0x0808    Whitney Point SD Ctrl 1
Vendor ID:       0x8086    Intel Corporation
Subsystem ID:    0x0808    Unknown
Subsystem Vendor ID:    0x8086    Intel Corporation
Revision ID:     0x01
Interrupt Line:     0x05      IRQ5
Base Addr. Registers:   0xFFA58000    Memory-Mapped Register
          0x00000000    Not Implemented
          0x00000000    Not Implemented
          0x00000000    Not Implemented
          0x00000000    Not Implemented
          0x00000000    Not Implemented
PCI Express Device: No     0 PCI Express devices found
PCI Cap List ID's:  N/A    CAP ID's not accessable


Setting          Value     Description
-------------------------------------------------------------------------
Class:           0x08      Base system peripheral
Sub-Class:       0x05      Unknown
Device ID:       0x0812    Whitney Point SDIO Ctrl 2
Vendor ID:       0x8086    Intel Corporation
Subsystem ID:    0x0812    Unknown
Subsystem Vendor ID:    0x8086    Intel Corporation
Revision ID:     0x01
Interrupt Line:     0x03      IRQ3
Base Addr. Registers:   0xFFA5C000    Memory-Mapped Register
          0x00000000    Not Implemented
          0x00000000    Not Implemented
          0x00000000    Not Implemented
          0x00000000    Not Implemented
          0x00000000    Not Implemented
PCI Express Device: No     0 PCI Express devices found
PCI Cap List ID's:  None      Capabilities list not implemented


Setting          Value     Description
-------------------------------------------------------------------------
Class:           0x01      Mass-storage controller
Sub-Class:       0x06      Unknown
Device ID:       0x0816    Whitney Point SATA AHCI Controller
Vendor ID:       0x8086    Intel Corporation
Subsystem ID:    0x0816    Unknown
Subsystem Vendor ID:    0x8086    Intel Corporation
Revision ID:     0x01
Interrupt Line:     0x7E      IRQ126
Base Addr. Registers:   0x00000000    Not Implemented
```

```
          0x00000000    Not Implemented
          0x00000000    Not Implemented
          0x00000000    Not Implemented
          0x00002001    I/O Register
          0xFF009000    Memory-Mapped Register
PCI Express Device: No     0 PCI Express devices found
PCI Cap List ID's:  N/A    CAP ID's not accessable


###############################################################################
CPU Information (cpu_info)
###################################################

   Vendor ID:       GenuineIntel
   Type:        0
   Family:          6
   Model:       6
   Stepping:    1

   CPU Brand String:   Genuine Intel(R) CPU        @ 1.20GHz
   CPU Frequency:      600 MHz
   Core Ratio:      6
   Bus Speed:       100 MHz
   Front Side Bus:     400 MHz


###############################################################################
Video BIOS Information (bios_info)
###########################################

General Information:
  Build:     3034
  Date:          12/10/2010
  Size:          65536
  Checksum:      0x8D

VESA BIOS Extension Information:
  Version:       01.00
  PM Version:    1.0

(VBE) Supported Power Management States:
  Reduced On:    Yes
  Off:       Yes
  Suspend:       Yes
  Standby:       Yes

Video BIOS Supported VESA Modes:
Mode   Width  Height Depth  Frequencies (Hz)
-------------------------------------------------
0x101  640 480 8bpp   60
0x103  800 600 8bpp   60
0x105  1024   768 8bpp   60
0x107  1280   1024   8bpp   60
0x111  640 480 16bpp  60
0x112  640 480 32bpp  60
0x114  800 600 16bpp  60
0x115  800 600 32bpp  60
```

```
0x117  1024   768 16bpp  60
0x118  1024   768 32bpp  60
0x11A  1280   1024   16bpp  60
0x11B  1280   1024   32bpp  60
0x160  1024   600 8bpp   60
0x161  1024   600 16bpp  60
0x162  1024   600 32bpp  60


################################################################################
Video BIOS Tests (bios_crc)
################################################

Video BIOS checksums match
Calculated checksum is        0x8D
Video BIOS stored checksum is 0x8D


################################################################################
Display Data Channel Information (ddc_info)
#################################

DDC information not available.
CRT display does not support DDC, or is not connected.


################################################################################
Graphics Memory Data Line (memory_data_line)
#################################

Performed graphics memory data line test on address 00bffffb


################################################################################
Graphics Memory Stuck Bit (memory_stuck_bit)
#################################

Performed graphics memory stuck bit test on range 00000000-00bfffff


################################################################################
Register Read/Write Test (registers)
########################################

Register Read/Write test passed


################################################################################
Final report
################################################################


All the selected tests passed with no failures.
```

§