# LENOVO DIAGNOSTICS LINUX V4.57.0 **USER GUIDE**

*FIT – Instituto de Tecnologia*

# LENOVO DIAGNOSTICS LINUX **USER GUIDE**

| | |
|---|---|
| **Title:** | Lenovo Diagnostics User Guide |
| **Author:** | Lenovo Diagnostics Linux team |
| **Date:** | July 19th, 2024 |
| **Platform:** | Linux |

# REVISION **HISTORY**

| Revision | Author | Revision History | Date |
|---|---|---|---|
| 1.0 | Elsa Martins | Created according to Lenovo Diagnostics 4.29 | Mar 15, 2019 |
| 1.1 | Elsa Martins | Updated according to Lenovo Diagnostics 4.30 | May 23, 2019 |
| 1.2 | Helano Rocha | Updated according to Lenovo Diagnostics 4.31 | Jul 30, 2019 |
| 1.3 | Helano Rocha | Updated according to Lenovo Diagnostics 4.32 | Oct 15, 2019 |
| 1.4 | Helano Rocha | Updated according to Lenovo Diagnostics 4.33 | Jan 17, 2020 |
| 1.5 | Geisiane Almeida | Updated according to Lenovo Diagnostics 4.34 | Feb 20, 2020 |
| 1.6 | Júlio Oliveira | Updated according to Lenovo Diagnostics 4.35 | Apr 29, 2020 |
| 1.7 | Geisiane Almeida | Updated according to Lenovo Diagnostics 4.35.1 | Jun 08, 2020 |
| 1.8 | Geisiane Almeida | Updated according to Lenovo Diagnostics 4.36 | Jun 29, 2020 |
| 1.9 | Claudio Pereira | Updated according to Lenovo Diagnostics 4.37 | Oct 21, 2020 |
| 1.10 | Rafael Rodrigues | Updated according to Lenovo Diagnostics 4.38 | Dec 07, 2020 |
| 1.11 | Rafael Rodrigues | Updated according to Lenovo Diagnostics 4.39 | Mar 05, 2021 |
| 1.12 | Rafael Rodrigues | Updated according to Lenovo Diagnostics 4.40 | Jun 25, 2021 |
| 1.13 | Rafael Rodrigues | Updated according to Lenovo Diagnostics 4.41 | Sep 17, 2021 |
| 1.14 | Rafael Rodrigues | Updated according to Lenovo Diagnostics 4.42 | Dec 10, 2021 |
| 1.15 | Rafael Rodrigues | Updated according to Lenovo Diagnostics 4.43 | Feb 11, 2022 |
| 1.16 | Rafael Rodrigues | Updated according to Lenovo Diagnostics 4.44 | Apr 22, 2022 |
| 1.17 | Rafael Rodrigues | Updated according to Lenovo Diagnostics 4.45 | Jun 24, 2022 |
| 1.18 | Rafael Rodrigues | Updated according to Lenovo Diagnostics 4.46 | Jul 21, 2022 |
| 1.19 | Marcia Furlaneto | Updated according to Lenovo Diagnostics 4.47 | Oct 28, 2022 |
| 1.20 | Marcia Furlaneto | Updated according to Lenovo Diagnostics 4.48 | Dec 30, 2022 |
| 1.21 | Marcia Furlaneto | Updated according to Lenovo Diagnostics 4.49 | Mar 03, 2023 |
| 1.22 | Lucca Salvatore | Updated according to Lenovo Diagnostics 4.50 | May 02, 2023 |
| 1.23 | Lucca Salvatore | Updated according to Lenovo Diagnostics 4.51 | July 07, 2023 |
| 1.24 | Lucca Salvatore | Updated according to Lenovo Diagnostics 4.52 | Sep 05, 2023 |
| 1.25 | Lucca Salvatore | Updated according to Lenovo Diagnostics 4.53 | Sep 13, 2023 |
| 1.26 | Lucca Salvatore | Updated according to Lenovo Diagnostics 4.54 | Dec 14, 2023 |
| 1.27 | Lucca Salvatore | Updated according to Lenovo Diagnostics 4.55 | Mar 13, 2024 |
| 1.28 | Lucca Salvatore | Updated according to Lenovo Diagnostics 4.56 | May 17, 2024 |
| 1.29 | Lucca Salvatore | Updated according to Lenovo Diagnostics 4.57 | Jul 19, 2024 |

# TABLE OF **CONTENTS**

# LENOVO DIAGNOSTICS LINUX V4.57.0 **USER GUIDE**

**Note**
Before using this information, be sure to read and understand the Lenovo Privacy Statement.

# 1. LENOVO WRITTEN OFFER FOR **LENOVO DIAGNOSTIC FOR LINUX**

This Lenovo Diagnostics for UEFI (the “Product”) may include software made publicly available by Lenovo, including software licensed under the General Public License and/or the Lesser General Public License (the “open-source software”).

You may obtain a copy of the corresponding source code for any such open source software licensed under the General Public License and/or the Lesser General Public License (or any other license requiring us to make a written offer to provide corresponding source code to you) from Lenovo for a period of one year without charge except for the cost of media, shipping, and handling, upon written request to Lenovo. This offer is valid to anyone in receipt of this product. You may send your request in writing to the address below accompanied by a check or money order for $15 to:

Lenovo Legal Department
Attn: Open Source Team / Source Code Requests
8001 Development Dr.
Morrisville, NC 27560

Please include the “Product Name” and the “Product Version” associated with the software of this product as part of your request. Be sure to provide a return address.

The open source software is distributed in hope it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See for example the GNU General Public License and/or the Lesser General Public License for more information.

## 1.1 Technical Inquiries

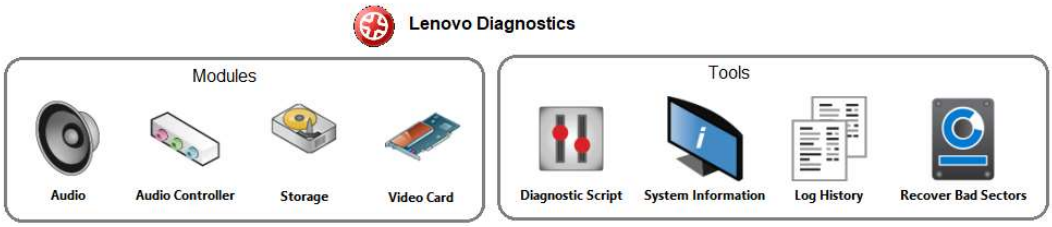
In case any doubts or strange behavior are risen during the use of Lenovo Diagnostics for Linux, please feel free to contact us by sending a message through Lenovo support channel, on **diagssupport@lenovo.com**.

When sending an inquiry, please inform a brief explanation of the problem encountered and any evidence that may help to visualize it.

# 2. LENOVO DIAGNOSTICS **OVERVIEW**

## 1.1 What is Lenovo Diagnostics?

Lenovo Diagnostics is a diagnostic tool that tests various devices in Lenovo computers providing feedback to the users about their machine's health. Lenovo Diagnostics is composed by Modules, that allow performing diagnostics for a group of devices, tools to create custom executions (diagnostic script). It is also possible to check detailed information about each device (system information) and consult the results for the tests performed in a machine (Log History).




## 1.2 Understanding the diagnostics

Each module contains one or more tests that may be performed under one or more devices resulting in a diagnostic. This structure is displayed in the image below:


When a diagnostic is finished, Lenovo Diagnostics displays the results for each performed test and creates two results codes resuming the test execution.

The tests on Lenovo Diagnostics may have the following statuses:

| | |
|---|---|
| **Passed** | When the test algorithm is executed and no failure is found. |
| **Failed** | When the test identifies the diagnosed device is defective. |
| **Warning** | When the test indicates the diagnosed device may have some defect, but the result is not conclusive. |
| **Canceled** | When the test is canceled in the middle of test execution. |

| | |
|---|---|
| **Not Applicable** | When the device does not meet a minimum requirement for the test to be executed. |

The generated codes are:

| | |
|---|---|
| **Result Code** | Contains information about the machine serial number, system platform and test execution status and date. This code is generated for each tested device. |
| **Final Result Code** | Contains information about the machine serial number, system platform and execution date. This code also reports the module where the tests have been performed and the tests with failed status. |

On the next section, you will learn how to use Lenovo Diagnostics to perform the diagnostics.

## 1.3 Anti-tamper protection

To protect users against possible security faults caused by using tampered versions of Lenovo Diagnostics Linux, the application performs a file checking before booting, in which the files on the bootable device are validated.
If it is detected that one or more files are not consistent with their expected format, the users will be prompted during the loading of the linux dependencies, which will read:

**"Checksum mismatch!"**
**"THIS DISTRO WAS MODIFIED AND IT IS NOT A LENOVO ORIGINAL."**
**"PLEASE CONFIRM, DO YOU WANT TO PROCEED WITH BOOT? (Yes, No, yes-Don't ask again) - [Yy/Nn/Dd]"**

In case this happens, there are three possibilities to the user, based on their choices of how to proceed:

- **The user may choose to do not load the application, by typing N or n.** In this case, the computing is simply going to restart.
- **The user may choose to ignore the warning, and proceed with the application, by typing Y or y.** In this case, the application is going to load normally, but a red label stating that the application was modified is going to appear to the right of the application's name.


- **It's also possible for the user to permanently allow the booting of the modified version, by typing D or d.** In this case, in future using, the user will not be prompted again. The effects are the same as the previous bullet, but no more warning will be made during linux loading. The red label, however, will still appear.

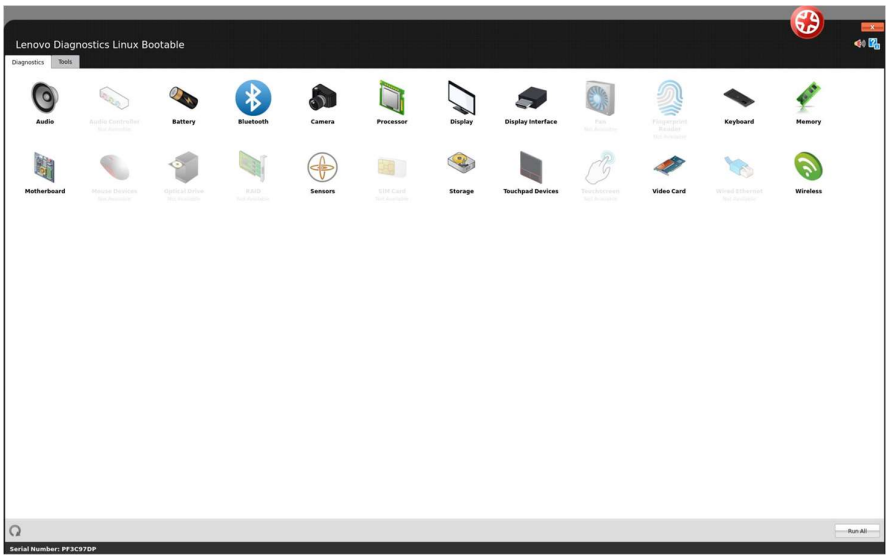
# 3. PERFORMING DIAGNOSTICS WITH **LENOVO DIAGNOSTICS**

## 3.1 Lenovo Diagnostics Main screen

By opening the Lenovo Diagnostics, you will see the main screen with the tabs Diagnostics and Tools.

**Note:** At the first time using the bootable version (without config.ini parameters on the USB flash drive) a popup will be shown informing about the sensitive data. When pressing the "OK" button, the information is memorized and the popup is no longer shown.

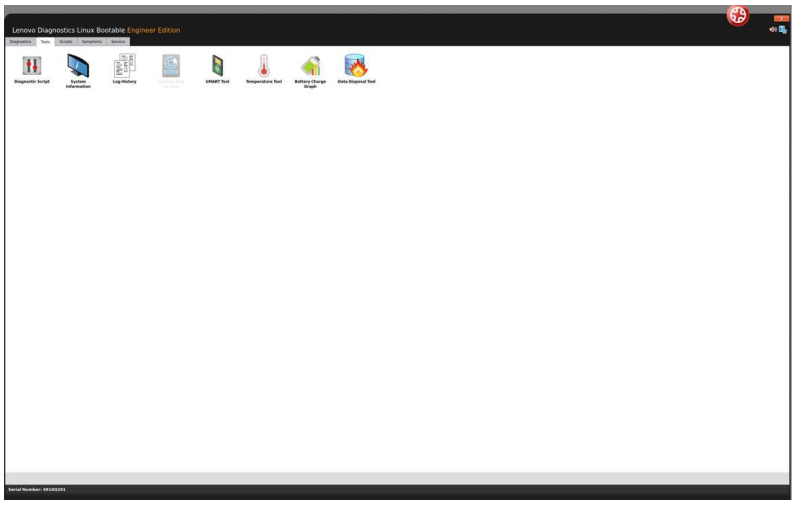

**Diagnostics**: Here it is possible to see the modules provided by Lenovo Diagnostics and select one of them to perform the tests. Modules with unavailable devices are shown as disabled with the information “Not Available” and modules with available but unsupported devices are shown disabled with a red “X” and the information “Not Supported”, as illustrated in the image below. On this tab, it is also possible to select the “Run All” option and perform the diagnostic for all available modules.




**Refresh button** : You must update the modules after plugging or unplugging any device.

**Sound Notice button** : You must choose if you want to be advised regarding the tests execution.

**Tools:** By selecting this option, you can access additional Lenovo Diagnostics features like Diagnostic Script, System Information, Log History, Recover Bad Sectors, SMART Tool, Temperature Tool, Battery Charge Graph and Data Disposal.

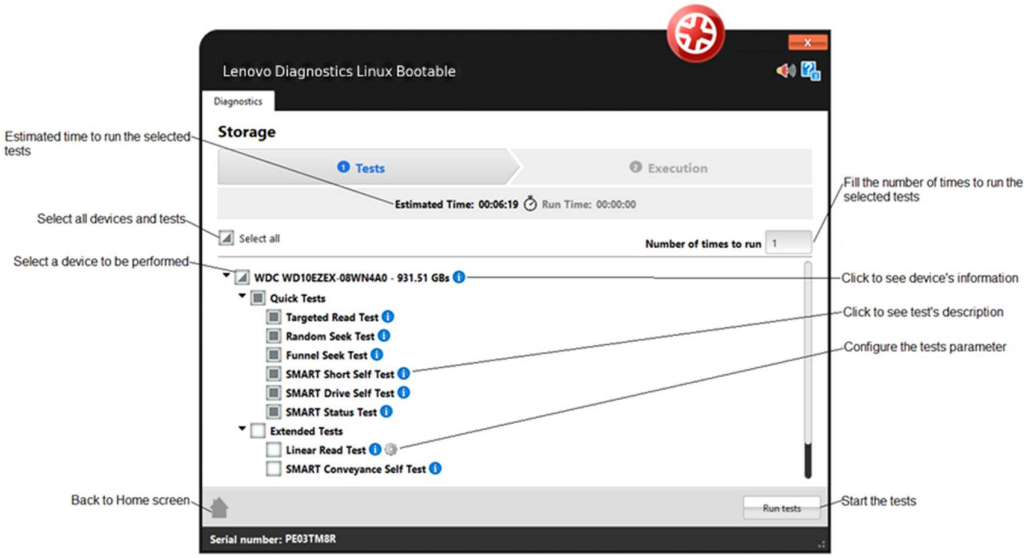


## 3.2 Run diagnostic for a module

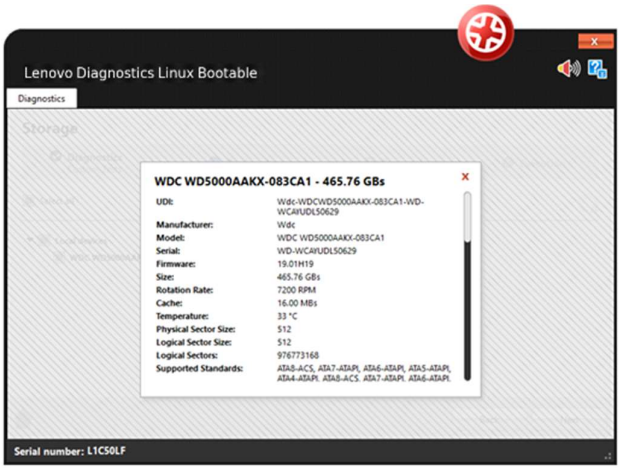
The diagnostic for a module in Lenovo Diagnostics is based on the following steps:

### 3.2.1 Select Devices and Tests

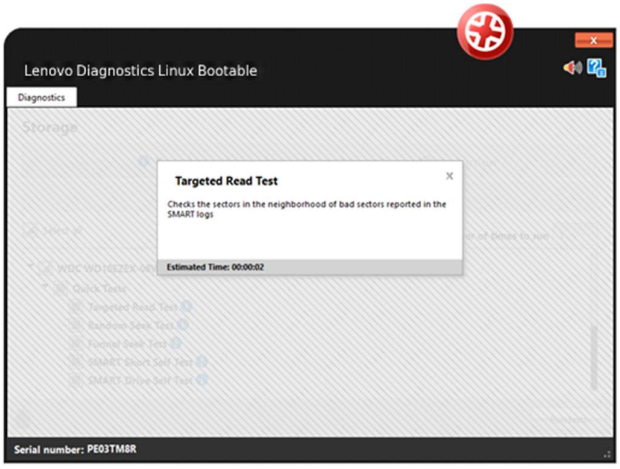
By selecting a module in the Diagnostics tab, you will be directed to a screen where you can select which devices and tests will be performed. On this step, all devices and tests supported by the selected module are displayed and you may select one or more of them to perform the tests. It is also possible to select the number of times to run the set of tests in a range from 1 to 999 times.



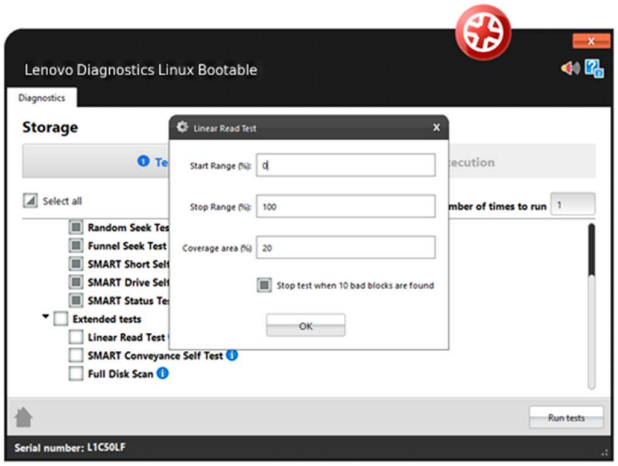

By clicking on **See Device Information**, you can view detailed information about the device. The properties displayed depend on the selected module.




By clicking on **See test description**, you can view a brief description of the test and the estimated time to run it, as can be seen in the screen below.




If a test supports parameters customization, the icon ⚙ is displayed next to the test name. By clicking on this icon, a popup is displayed in order to set the values. The image below shows the customization for the Linear Read Test from Storage module:

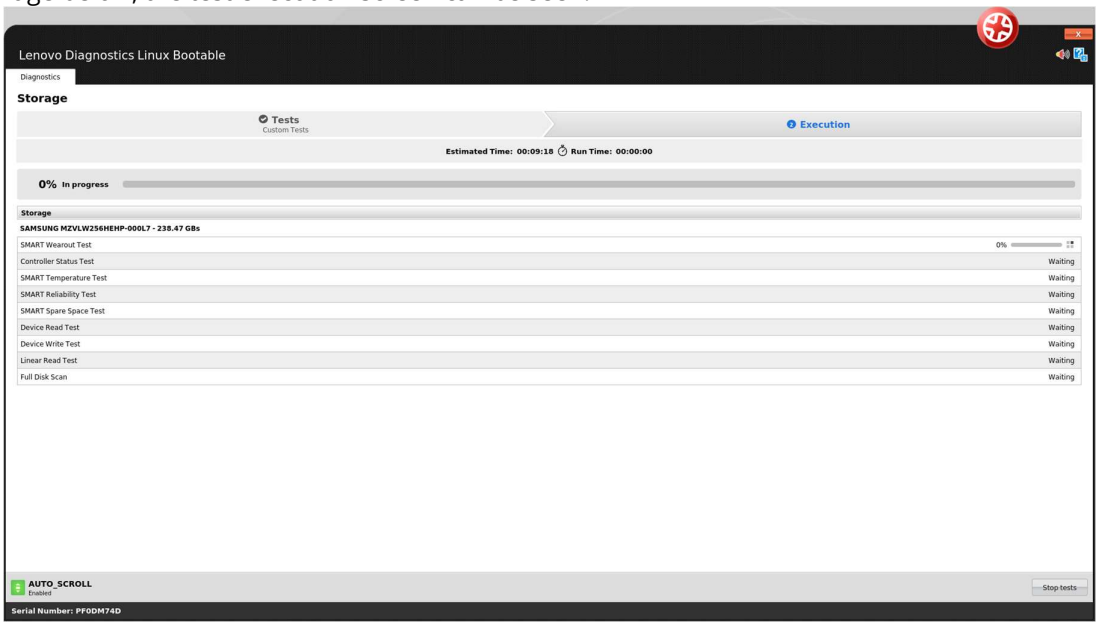


After the devices and test selection, the user is able to run the diagnostic. The test execution is detailed in the next section.

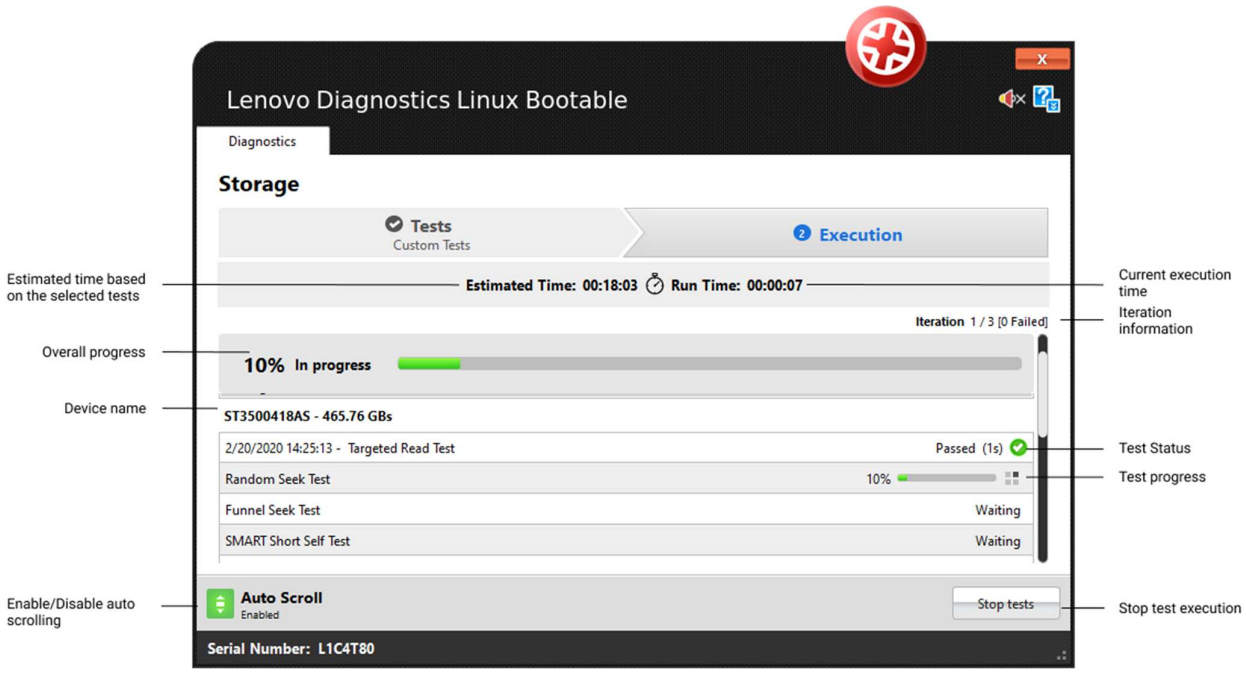
**Note:** When a second device is connected, in addition to the original one, the module screen is going to refresh and display both.

### 3.2.2 Run Tests

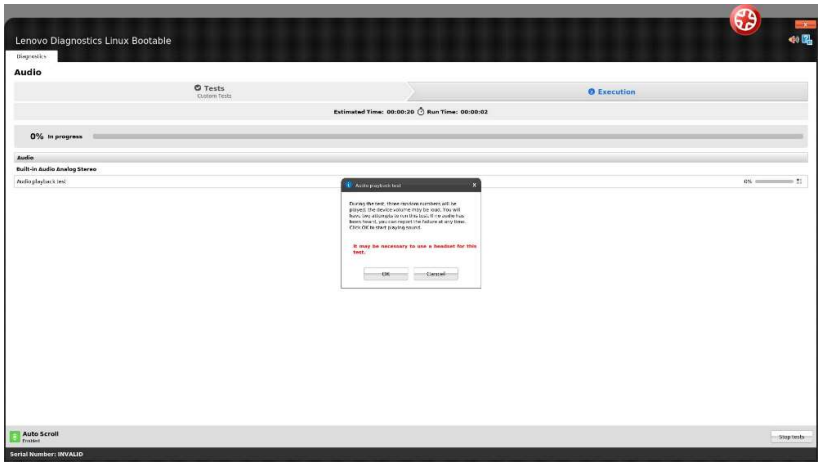
After configuring the tests execution and clicking on **Run Test,** the user will be directed to the Execution screen. This screen is displayed for both Recommended and Custom tests.
In the image below, the test execution screen can be seen:




Once you have pressed OK, the tests execution begins. See in the image below that all devices selected in the previous steps are displayed with their respective tests. The user can follow each test execution by tracking the individual test progress and watch the test status of each one individually. In addition, it is possible to see the overall test progress and time that report the progress for all devices and tests selected.
If the user has selected **Custom Tests** with more than one iteration, it will be possible to see the amount of iterations that failed and navigate around these iterations in order to see the executions that are already finished and the executions that have not been started yet.



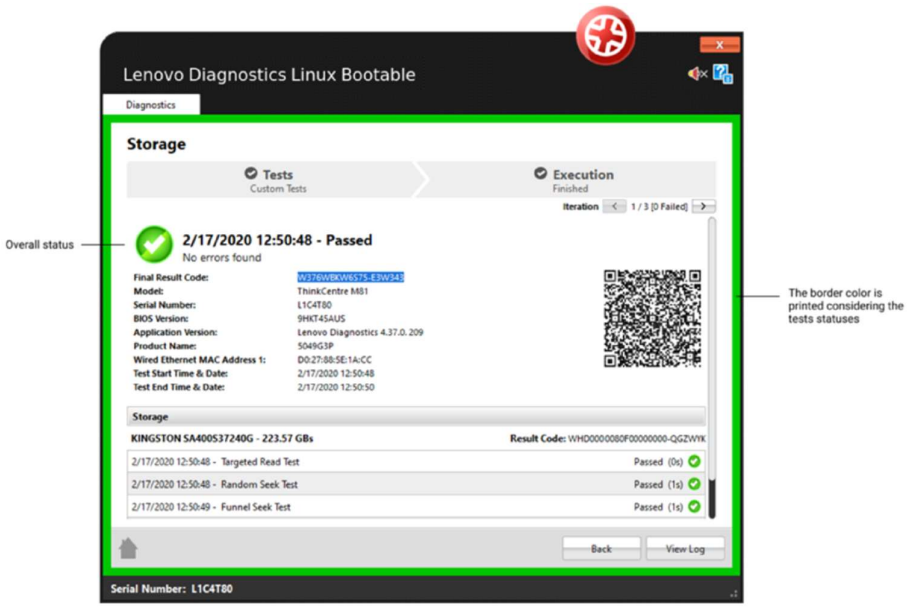

Some tests have specific guidelines that should be followed by the user to assure the correct execution. In this case, before beginning the execution of these tests, the application displays a popup with the test instructions. For instance, the Audio Playback test asks the user to make sure the output device is working correctly and advises that a probably loud sound will be played.




If the user choses to continue by pressing the **OK** button, the execution continues normally. If the user presses the **Cancel** button, the current test is canceled and the execution proceeds to the tests that have not been executed yet.

If you want to abort the whole test execution, you can click on **Stop tests.** In this case, the current test and all tests waiting for execution are canceled, including those from the next iterations. In the same way, the overall status for the current iteration and for all next iterations are changed to **Canceled**.

After all tests have been completed, the system will display the screen below, where it is possible to analyze the tests results.

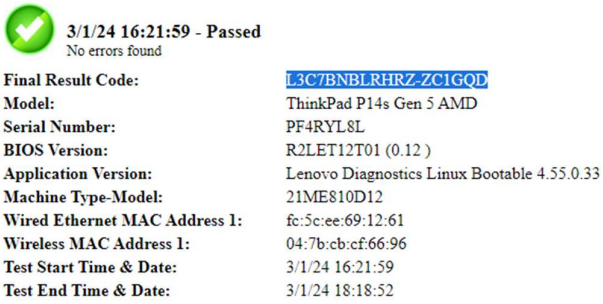



In the image above it is possible to see each test results, machine information, final result and test date. This section also displays a QR Code containing all these information, in addition to the machine's UUID.

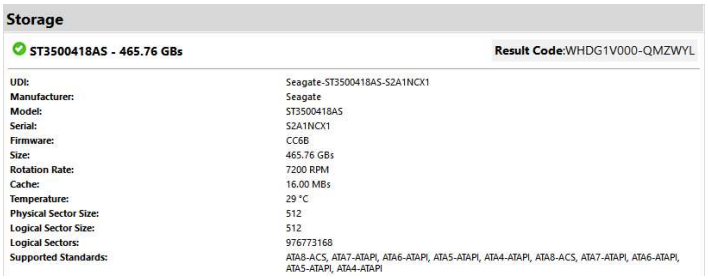
When the execution of a script starts, the application is going to automatically filter the diagnostics that are unsupported, and in case they were listed on the script, they are not going to be listed for execution, but will be logged on the JSON log, in the "unsupported_test" field.

### 3.3 See Execution Log

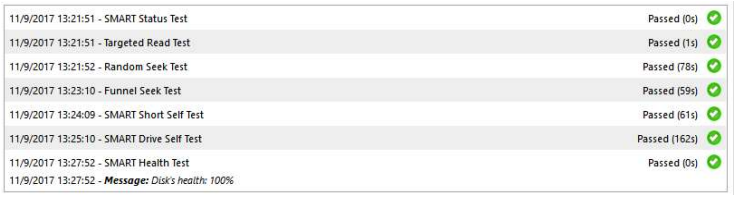
After a diagnostic execution, Lenovo Diagnostics generates a log with detailed information about the devices and their test results. This log is composed by the following sections:
**General information**: Contains information about the machine, test date and final result. This section also displays a QR Code containing this information.




**Device information**: Displays the technical details of each tested device and their respective result codes.




**Test Results**: Displays the results and execution time of each performed test.




### 3.3.1 QR Code

After a test cycle is completed, the user will be able to read on the log screen (and on the HTML log file, if exported), a QR code that contains the machine's general information and a summary of the diagnostics results.
By reading the QR code with a camera, the user will be met, on the screen of the device used to perform the reading, with the following information:

- **Final Result Code:** A code that identifies the diagnostics performed and their results **(internal use only).**
- **Model**: The tested machine's model.
- **Serial Number**: The tested machine's serial number.
- **BIOS Version**: The tested machine's BIOS version.
- **Device's UUID**: The tested machine's Universal Unique Identifier.
- **Application's mode**: Standard OR Engineer (when a config.ini file exists).
- **Application's Version**: The version of the application's build used during the diagnostics.
- **Machine Type-model**: A combination of type and model identifiers.
- **Wired MAC Address**: The machine's wired mac address. This value may be unavailable if the related hardware is not installed.
- **Wireless MAC Address**: The machine's wired mac address. This value may be unavailable if the related hardware is not installed.
- **Test Start Date**: The time & date register of the beginning of the diagnostic flow execution (YYYY-MM-DD format).
- **Test End Date**: The time & date register of the end of the diagnostic flow execution (YYYY-MM-DD format).
- **Scripts Name**: The name of the script used for the diagnostic flow, in case a script execution occurred.

### 3.4 About

In this function, the application will display a screen where we can find information related to the Lenovo Diagnostics Linux application.


- Application version.
- Copyright information.
- Linux and Kernel version.
- Privacy Policy: Link to Lenovo's website with related information, if the connection is available.
- License Agreement: Information about licenses of libraries and services used by the application.
- Open-Source Software Licenses: Information related to open source included in and distributed by Lenovo.

# 4. LENOVO DIAGNOSTICS **MODULES AND TESTS**

This section provides information about all modules available in Lenovo Diagnostics and their respective tests. Here, you will understand the approach implemented by each test and how these tests should be performed to assure the correct diagnostic of your machine.

**Note:** Many of the tested hardware, such as lid sensors for example, depend on the good communication with the hardware's firmware. If firmware problems exist, the results of some tests may be inaccurate, and side effects may happen. In this situations, we strongly recommend a BIOS update, if available.

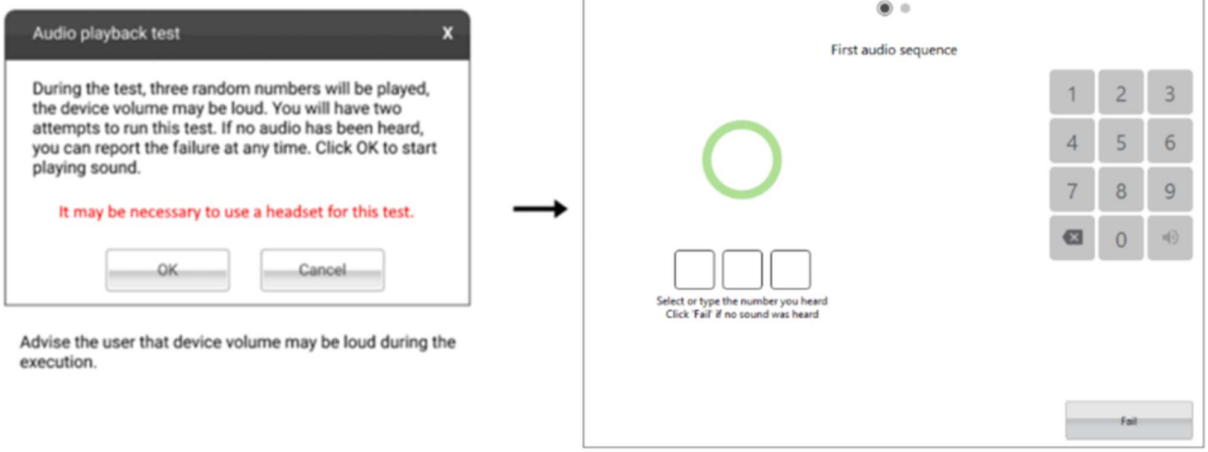
## 4.1 Audio

The Audio module is composed by the following tests:

| Test | Test type | Attendance |
|---|---|---|
| Audio Playback Test | Quick | Attended |
| Microphone Interactive Test | Quick | Attended |

**Audio Playback Test**

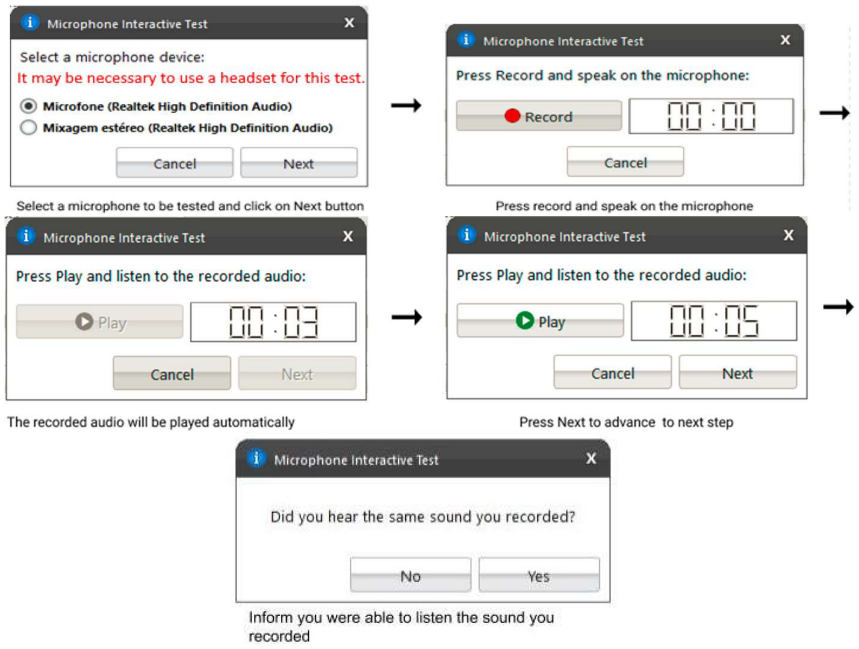
The audio playback test tries to play random numbers through the audio hardware and asks the user in what order the numbers were played.




**Note**: A second sequence of numbers will start if the first one ends.

**Microphone Interactive Test**

This test helps to identify if the microphone is capable of capturing sound properly.
The microphone interactive test is performed according to the following workflow:




**Note**: If the Microphone test does not work properly, please try using a headset.

## 4.2 Audio Controller

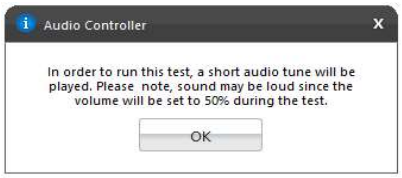
The Audio Controller module is composed by the following tests:

| Test | Test type | Attendance |
|---|---|---|
| CORB Status Test | Quick | Unattended |
| Output Stream Test | Quick | Unattended |
| Input Stream Test | Quick | Unattended |
| Bidirectional Input Test | Quick | Unattended |

**CORB Status Test**

This test checks the status of the Command Outbound Ring Buffer (CORB) mechanism.

**Output Stream Test**
This test checks the status of the output streams. Before starting the test, Lenovo Diagnostics advises the user that a short audio tune will be played, as displayed in the image below.




**Input Stream Test**
This test checks the status of the input streams.
**Bidirectional Stream Test**
In this test all the available bidirectional streams are checked, as a short audio tune is played for five seconds on all possible audio devices.

## 4.3 Battery

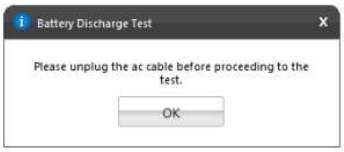
The Battery module is composed by the following tests:

| Test | Test type | Attendance |
|---|---|---|
| Battery Health Test | Quick | Unattended |
| Battery Temperature Test | Quick | Unattended |
| Battery Discharge Test | Extended | Attended |
| Battery Charge Test | Extended | Attended |

**Battery Health Test**
Battery Health Test checks the device charge capacity and other important battery properties in order to evaluate device's health.

**Battery Temperature Test**
Temperature test evaluates if the battery temperature is too high, which may prevent it from charging properly. This diagnostic is only available for SMART batteries.

**Battery Discharge Test**
Battery Health Test checks the device charge capacity and other important battery properties in order to evaluate device's health. If there is an AC cable plugged the user will have to unplug it before proceeding to the test as displayed below. It is only available if the battery is more than 20% charged.



**Battery Charge Test**

The test checks if the battery charge increases while the AC cable is connected. If there is no AC cable plugged you should connect it before proceeding to the test.


## 4.4 Bluetooth

The Bluetooth module is composed by the following test:

| Test | Test type | Attendance |
|---|---|---|
| Scan Test | Quick | Unattended |

**Scan Test**

Scan for nearby active Bluetooth devices.

The test starts by asking the user to make sure that the Bluetooth device is enabled and there is another Bluetooth close and active.


## 4.5 Camera

The Camera module is composed by the following tests:

| Test | Test type | Attendance |
|---|---|---|
| Camera Capture Test | Quick | Attended |
| Camera Barcode Test | Quick | Attended |

**Camera Capture Test**

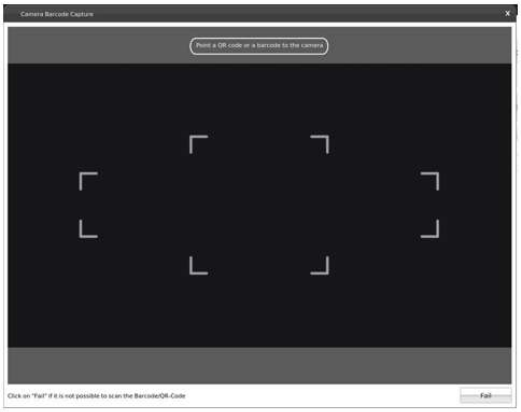
It verifies if the camera device is working properly based on the user's feedback for the captures images. This test is performed according to the following workflow:


Inform the user the video from camera will be displayed

Display the video from camera

Ask the user if the images was properly displayed

**Camera Barcode Test**
Verifies if the camera device can properly read the content of barcodes and QR codes. This test is performed according to the following workflow:




The user must point a QR code or a barcode to the camera, until it is centralized on the middle screen square

If the application is not able to read the barcode / QR code contents, the user must click the fail button or cancel the test via the “x” button on the window’s top right screen, or the ESC button.

## 4.6 Processor

The Processor module is composed by the following tests:

| Test | Test type | Attendance |
|---|---|---|
| BT Instruction Test | Quick | Unattended |
| x87 Floating Point Test | Quick | Unattended |
| MMX Test | Quick | Unattended |
| SSE Family Test | Quick | Unattended |
| AES Test | Quick | Unattended |
| Stress Test | Extended | Unattended |

**BT Instruction Test**
The test checks the processor support for BT instruction.

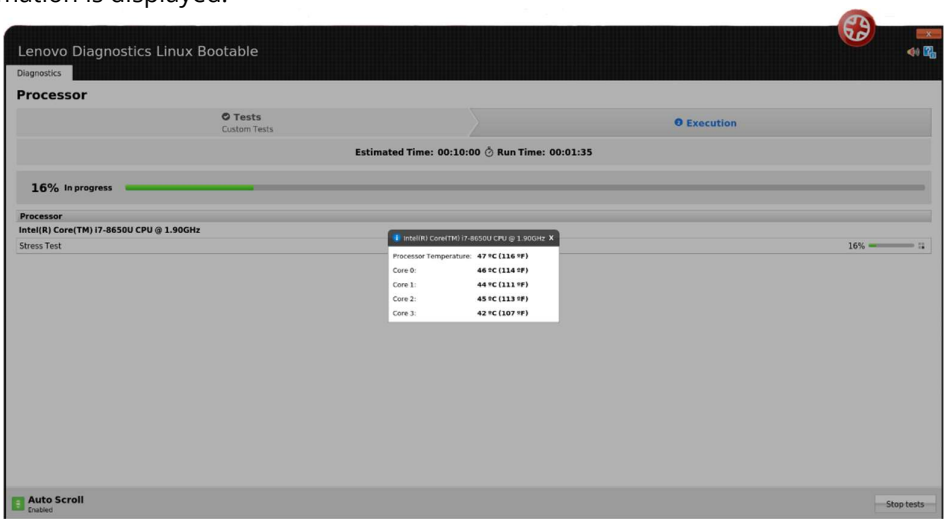
**x87 Floating Point Test**
The test checks the processor support for x87 Floating Point instructions. If the processor does not support such feature, the test returns unsupported.

**MMX Test**
The test checks the processor support for MMX instructions. If the processor does not support such feature, the test returns unsupported.

**SSE Test**
The test checks the processor support for SSE Family (SSE, SSE2, SSE3, SSSE3, SSE4.1, SSE4.2) instructions. If the processor does not support such feature, the test returns unsupported.

**AES Test**
The test checks the processor support for AES instructions. If the processor does not support such feature, the test returns unsupported.

**Stress Test**
The stress test performs a sequence of continuous check on all processor cores for 10 minutes. While running this test, the CPU temperature can increase considerably, to check the temperature a popup screen with this information is displayed.



## 4.7 Display

The Display module is composed by the following tests:

| Test | Test type | Attendance |
|---|---|---|
| Red Purity test | Quick | Attended |
| Green Purity test | Quick | Attended |
| Blue Purity test | Quick | Attended |
| Black Purity test | Quick | Attended |
| White test | Quick | Attended |
| Color Transition Test | Quick | Attended |
| Monochromatic Mesh Test | Quick | Attended |
| Inverted Monochromatic Mesh Test | Quick | Attended |
| Sharpness Test | Quick | Attended |
| Resolution Fitting Test | Quick | Unattended |
| Display Interactive Test | Quick | Attended |

**Red Purity test**
This test identifies any dead pixel or burn-in problem within the red channel.

**Green Purity test**
This test identifies any dead pixel or burn-in problem within the green channel.

**Blue Purity test**
This test identifies any dead pixel or burn-in problem within the blue channel.

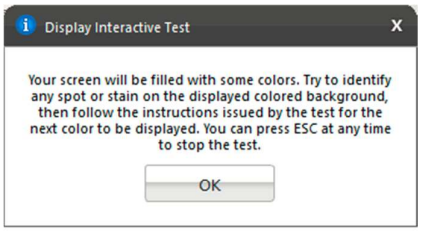
**Black Purity test**
This test identifies any dead pixel or burn-in problem within the black channel.

**White Purity test**
This test identifies any dead pixel or burn-in problem within the white channel.

**Color Transition Test**
This test identifies any problem with the display color distinction.

**Resolution Fitting Test**
This test checks if the system can take full advantage of the display native resolution.

**Display Interactive Test**
The Display Interactive Test is the combination of all purity tests. The purity tests aim to identify dead pixels or burn-in problems in the channels: red, green, blue, black and white.




Advise the user that her screen will be filled with some colors.

Fill the screen with a color.


Ask the user which number is being displayed.

All Purity tests from Display module are performed according to the following workflow:


Advise the user that her screen will be filled with the color

Fill the screen with the color

Ask the user if the color was properly displayed

## 4.8 Display Interface

The Display module is composed by the following tests:

| Test | Test type | Attendance |
|---|---|---|
| EDID Checksum Test | Quick | Unattended |
| Display Communication Test | Quick | Unattended |

**EDID Checksum Test**
This test checks the integrity of the Extended Display Identification Data (EDID) checksum provided by the monitor.

**Display Communication Test**
This test checks the communication with the monitor.

## 4.9 Fan

The Fan module is composed by the following test:

| Test | Test type | Attendance |
|---|---|---|
| Control Test | Quick | Unattended |

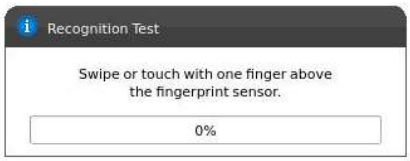
**Control Test**
Check if the fan controller is able to work on higher speeds according to predefined levels (0-7) in the firmware.

## 4.10 Fingerprint Reader

The Fingerprint Reader module is composed by the following test:

| Test | Test type | Attendance |
|---|---|---|
| Recognition Test | Quick | Attended |

**Recognition Test**
Check if the Fingerprint Reader device is able to detect a swipe or touch of a finger.
To do this, the user will be asked to swipe or touch with one finger above the fingerprint sensor to perform the operation required for the test.




**Note:** At the beginning of the test, a pop-up informing the user of the timeout will appear.

## 4.11 Keyboard

The Keyboard module is composed by the following tests:

| Test | Test type | Attendance |
|---|---|---|
| PS2 Keyboard Test | Quick | Unattended |
| USB Keyboard Test | Quick | Unattended |
| Keycode Verification Test | Quick | Attended |
| Advanced Test | Quick | Attended |
| Keyboard Backlight Test | Quick | Attended |

**PS2 Keyboard Test**
This test tries to identify any defective PS/2 keyboard detected on this machine.

**USB Keyboard Test**
This test tries to identify any defective USB keyboard detected on this machine.

**Keycode Verification Test**
It presents the latest pressed key by the user in a legible format and the current state of the toggle keys.
If the user confirms that all pressed keys were displayed, the test is finished as “Passed”. Otherwise, the test is finished as “Failed”.

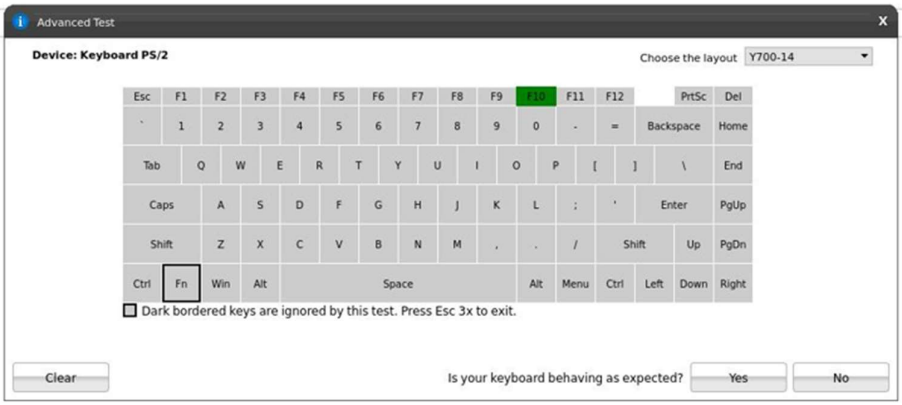

When you cancel the Keycode Verification Test, a popup will be displayed asking if you really want to exit the test, allowing you to return to where you had stopped by pressing “No” or finish the test by pressing “Yes”.


**Note:** If the user wants to cancel the test by using the keyboard, it’s possible to do it by pressing the ESC key three times.

**Note²:** The test identifies, with a **"Device"** label, the keyboard being tested at the time, and won't accept input from other ones.

### Advanced Test

It verifies the status of the keyboard keys.
The test will be showing the pressed keys until all keys are tested. The user can select the most appropriate keyboard layout.




When the user cancels the Advanced Test, a popup will be displayed asking if the user really wants to exit the test, allowing him to return to where he had stopped by pressing "No" or finish the test by pressing "Yes".

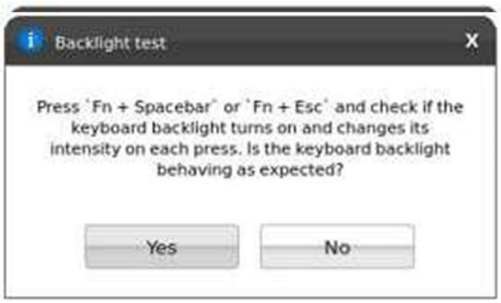

**Note:** If the user wants to cancel the test by using the keyboard, it's possible to do it by pressing the ESC key three times.

**Note²:** The test identifies, with a **"Device"** label, the keyboard being tested at the time, and won't accept input from other ones.

### Backlight Test

The backlight test gives instructions to the user on how to activate the keyboard's backlight, and checks the result of the activation.

The test starts by asking the user whether the backlight symbol is present in the keyboard or not.



If the user responds that it has, the test will give instructions on how to activate the backlight and proceeds to ask if it turned on or not.

**Obs.:** The keyboard module lists the available keyboards separately, each with a list of the supported tests.

## 4.12 Memory

The Memory module is composed by the following tests:

| Test | Test type | Attendance |
|---|---|---|
| Quick Random Pattern Test | Quick | Unattended |
| Advanced Integrity Test | Extended | Unattended |
| Address Test | Extended | Unattended |
| Bit Low Test | Extended | Unattended |
| Bit High Test | Extended | Unattended |
| Walking Ones Left Test | Extended | Unattended |
| Walking Ones Right Test | Extended | Unattended |
| Modulo-20 Test | Extended | Unattended |
| Moving Inversions 8 Bit Test | Extended | Unattended |
| Moving Inversions 32 Bit Test | Extended | Unattended |
| Random Pattern Test | Extended | Unattended |
| Random Number Sequence Test | Extended | Unattended |
| Block Move Test | Extended | Unattended |
| Nibble Move Test | Extended | Unattended |

**Quick Random Pattern Test**
The test consists of filling the memory with a random generated pattern and then checking if the pattern was correctly written. When checking, it writes the pattern binary complement and checks again. The test is repeated twice. By default, 15 random patterns are used, therefore, the test runs once for each of these patterns.

**Advanced Integrity Test**
The test is based on the March C- enhanced algorithm. This test consists of filling the accessible memory with a pattern, checking it, and writing its complement in an 8 bytes block size (64 bits) and then checking it again. This procedure is repeated twice, the first one addresses the accessible memory from the highest position to the lowest and the second time by doing the inverse path. This test is intended to cover Stuck-At Faults and some Coupling Faults and Transition Faults.

**Address Test**
This test consists of writing each memory address its own address. After that, the algorithm reads the memory previously written and checks if they still store their own address. This test is intended to cover any addressing fault in the accessible memory range.

**Bit Low Test**
This test consists of filling the memory buffer with a pattern where all bits are 0 and then checking it. When checking for this pattern, it writes its binary complement, and finally checks if the complement was stored accordingly. Such process is repeated 4 times. This test is intended to identify the most serious Stuck-At Faults, some cases of Transition Faults and some cases of Read Random Faults.

**Bit High Test**
This test consists of filling the memory buffer with a pattern where all bits are 1 and then checking it. When checking for this pattern, it writes its binary complement, and finally checks if the complement was stored accordingly. Such process is repeated 4 times. This test is intended to identify the most serious Stuck-At Faults, some cases of Transition Faults and some cases of Read Random Faults.

**Walking Ones Left Test**
The Walking Ones Left Test consists of writing a pattern where only the rightmost bit is set (e.g. 00000001), then shift this pattern to the left (e.g. 00000010) until the end of the size of a byte, writing it again at the same memory address each time such pattern is shifted. Therefore, the test is intended to cover most of the Stuck-At Faults and some cases of Coupling Faults, and also testing the data bus by confirming that every bit can be written.

**Walking Ones Right Test**
The Walking Ones Right Test consists of writing a pattern where only the leftmost bit is set (e.g. 10000000), then shift this pattern to the right (e.g. 01000000) until the end of the size of a byte, writing it again at the same memory address each time such pattern is shifted. Therefore, such test is intended to cover most of the Stuck-At Faults and some cases of Coupling Faults, and also testing the data bus by confirming that every bit can be written.

**Modulo-20 Test**
The test consists of writing into an interval of 20 memory locations for each block with a pattern and filling all other locations with its complement 6 times. Unlike the other tests, the Modulo-20 test is not affected by buffering or caching, so it is able to detect most of the Stuck-At Faults, Coupling Faults, Transition Faults and Read Random Faults that are not detected by other testing approaches.

**Moving Inversions 8 Bit Test**
The test consists of filling the memory with the 8-bit wide pattern: 10000000 and then checking that the pattern was correctly written. When checking, it writes the pattern binary complement (01111111) and checks it again. The procedure described earlier is repeated 8 times, one for each pattern right shifted: 10000000, 01000000, 00100000, 00010000, 00001000, 00000100, 00000010, 00000001.

**Moving Inversions 32 Bit Test**
This test fills all the accessible memory with a shifting pattern, that is, a value which is binary left shifted as it is written out through the accessible memory of every memory block. Once the pattern reaches 0x80000000 (a value with the left most bit set to 1 only) then the pattern is reset to 0x00000001. After that, it checks the written values and writes their binary complements, starting from the first memory address to the last one. Finally, the algorithm checks the memory for the complements written in the previous step, this checking starts from the last element down to the first one. Such process is repeated 2 times. This test presents a

more thorough approach intended to cover most of the Stuck-At Faults and Transition Faults and some cases of Coupling Faults and Read Random Faults...

**Random Pattern Test**
The test consists of filling the memory with a random generated pattern and then checking if the pattern was correctly written. When checking, it writes the pattern binary complement and checks it again. This process is repeated twice. By default, 50 random patterns are used, therefore the test runs once for each of these patterns.

**Random Number Sequence Test**
The test consists of filling the memory with one different random generated pattern for each memory address and then checking if the pattern was correctly written. In order to check it, the test must generate these numbers based on a seed that may be reset to reproduce the sequence. When checking, it writes the pattern binary complement, and it checks again. Such process is repeated several times. This test is intended to cover most of the Stuck-At Faults. Coupling Faults, and some cases of Transition Faults and Read Random Faults.

**Block Move Test**
The test consists of moving memory data around within memory blocks. It repeats the movements described above 80 times. Finally, the test checks every memory address to verify if it is consistent.

**Nibble Move Test**
This test consists of writing to a nibble (a nibble is a group of four bits) a pattern value in each memory address, then it validates every nibble individually. It repeats this process until all nibbles in every address are checked.

## 4.13 Motherboard

The Motherboard module is composed by the following tests:

| Test | Test type | Attendance |
|---|---|---|
| Chipset Test | Quick | Unattended |
| PCI/PCI-e Test | Quick | Unattended |
| RTC Test | Quick | Unattended |
| USB Test | Quick | Unattended |
| CMOS Pattern Test | Quick | Unattended |

**Chipset Test**
The test checks the status registers of the controllers that form the foundation of the motherboard chipset. These controllers are: EHCI, OHCI, xHCI and SATA.

**PCI/PCI-e Test**
The PCI/PCI-e Test checks the status registers of the PCI Express onboard devices for unexpected errors or power failure.

**RTC Test**
The test checks the following RTC (Real Time Clock) properties: accuracy and rollover. The test attempts to guarantee the correct operation of these properties.

**USB Test**
The test checks the status of USB devices. If any errors are indicated, the test fails.

**CMOS Pattern Test**
The test saves the CMOS current value and attempts to write predefined pattern values on CMOS, verifying if the written values are consistent, and then the backup is restored

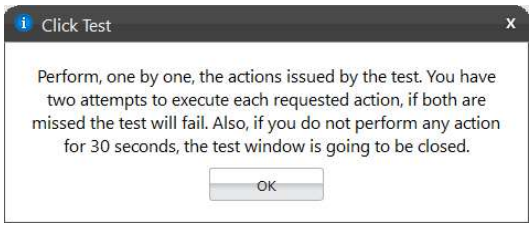
**Note:** The use of HUBs is not fully supported by the application, and the diagnostic results may be inaccurate when this kind of device is used. As such, we strongly discourage their use when performing diagnostics.

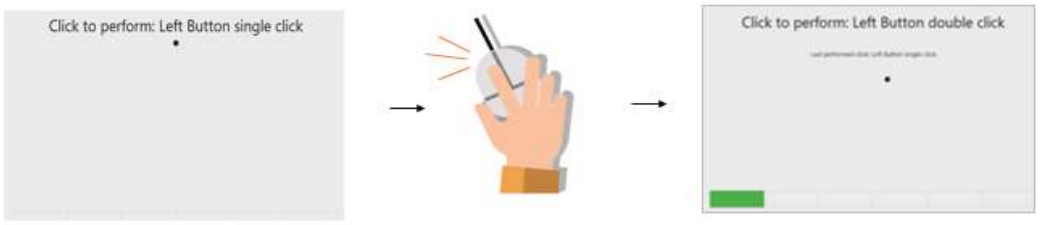
## 4.14 Mouse Devices

The Mouse Devices module is composed by the following tests:

| Test | Test type | Attendance |
|---|---|---|
| Click Test | Quick | Attended |
| Precision Test | Quick | Attended |
| Press Precision Test | Quick | Attended |

**Click Test**
It check the device buttons, issuing the user to perform some actions regarding them. The Click Test execution is based on the workflow below.



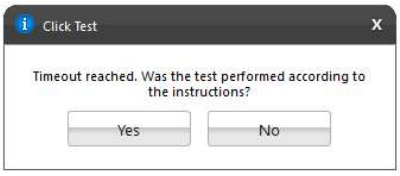



Application requests the user to perform some action with the tested device

User performs the requested action

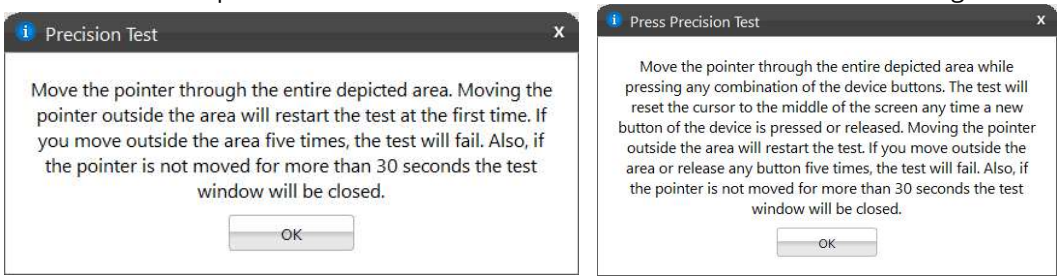
Application detects if the action was performed successfully or not and request another action

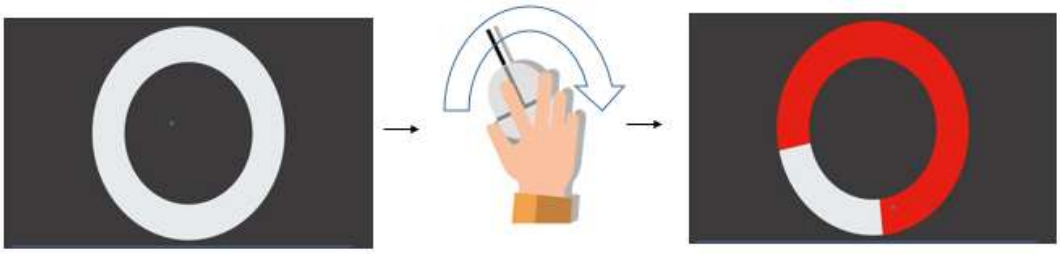
The user must perform one by one, the actions issued by the test. Note that the actions requested depends on the numbers and types of buttons present in your pointing device.
The user has two attempts to execute each requested action. If the requested action was detected by the application, it is considered as passed. Otherwise the action is considered as “Failed”. The Click Test is considered as “Passed” if all actions required are passed.
If the user does not perform any action for 30 seconds, the following popup is displayed:




If the user select “Yes” reporting that the test was performed according to the instructions the test will fail. In this case, the application assumes that it was not possible to detect the device, indicating a bad working of it.
If the user selects “No”, the test is canceled because the actions required were not properly executed.

**Precision Test and Press Precision Test**
Tests the device movement precision. The Precision Test execution is based on the following workflow:



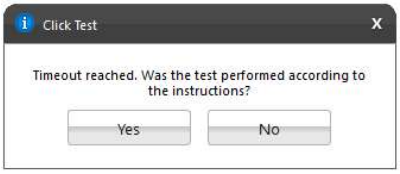


Application requests the user to move the pointer through the depicted area.

User performs the requested action

Application detects if the pointer was moved without to reach the outside area

If the user gets to move the pointer through the entire depicted area without reaching the outside area, the test is finished as passed. If the outside area was reached twice, the test will fail. If the pointer is not moved for more than 30 seconds the test will be closed and the following message will be displayed to the user:




If the user selects “Yes”, reporting the test was performed according to the instructions, the test will be finished as “failed”. In this case, the application assumes that it was not possible to detect the device movement, indicating a bad working of the device. If the user selects “No”, the test is canceled because the required actions were not properly executed.

**Note:** To execute the Press Precision test, you must press any button of the mouse and move the pointer.

## 4.15 Optical Drive

The Optical Drive module is composed by the following tests:

| Test | Test type | Attendance |
|---|---|---|
| Media-Less Optical Self-Test | Quick | Attended |
| Linear Seek Test | Quick | Attended |
| Random Seek Test | Quick | Attended |
| Funnel Seek Test | Quick | Attended |
| Read and Compare Test | Quick | Attended |
| Write Test | Quick | Attended |

**Media-Less Optical Self-Test**

Checks the optical drive internal components. This test requires that no media is inserted into the drive. Therefore, if any media was detected into the drive the application displays the following message:


**Linear Seek Test**
Checks the integrity of the optical drive mechanism bt continuously moving the driver head all around the media. To execute this test the user must have a media containing at least the following amount of date written on it according to its type: CD (210 MB), DVD (1000 MB), Blu-Ray (4000 MB).

**Random Seek Test**
Checks the integrity of the optical drive transmission mechanism by moving the driver head through random positions on the media. In order to execute this test, the user must have a media containing at least the following amount of date written on it according to its type: CD (210 MB), DVD (1000 MB), Blu-Ray (4000 MB).

**Funnel Seek Test**
Checks the integrity of the optical drive mechanism by moving the driver head in a funnel pattern. For executing this test, the user must have a media containing at least the following amount of date written on it according to its type: CD (210 MB), DVD (1000 MB), Blu-Ray (4000 MB).

**Read and Compare Test**
Checks the driver ability to make correct read operations. It performs two linear read operations and compares the information obtained from the two readings. For executing this test, the user must have a media containing at least the following amount of date written on it according to its type: CD (210 MB), DVD (1000 MB), Blu-Ray (4000 MB).

**Write Test**
Checks the capability of the drive to write correctly to an optical media. For executing this test, you must have a blank media (CDR, CD-RW, DVD-R, DVD-RW, BD-R, BD-RE).

**Linear Seek Test, Random Seek Test, Funnel Seek Test, Read and Compare Test and Write Test** are performed according the following workflow:


Application opens the optical drive tray and request the user to insert a valid media

User inserts a valid media and click on "OK" button

Application validates the media and continues the test

## 4.16 RAID

The RAID module is composed by the following tests:

| Test | Test type | Attendance |
|---|---|---|
| Battery Status Test | Quick | Unattended |
| Enclosure Status Test | Quick | Unattended |
| Logical Drive Status Test | Quick | Unattended |
| RAID Status Test | Quick | Unattended |
| Consistency Check | Extended | Attended |

**Battery Status Test**
Checks the health of the controller BBU (Battery Backup Unit).

**Enclosure Status Test**
Checks the status of the enclosures used by a RAID controller.

**Logical Drive Status Test**
Checks the status of each logical drive in the RAID controller.

**RAID Status Test**
Checks the status of each RAID volume in the RAID controller.

**Consistency Check**
Checks the consistency of each controller logical drive. This test might take a few hours to run depending on how many logical and physical drives are present in the RAID controller.

## 4.17 Sensors

The Sensors module is composed by the following test:

| Test | Test type | Attendance |
|---|---|---|
| Accelerometer Noise Test | Quick | Unattended |
| Gyrometer Noise Test | Quick | Unattended |
| Compass Noise Test | Quick | Unattended |
| Accelerometer Interactive Test | Quick | Attended |
| Lid Closure Test | Quick | Attended |

**Accelerometer Noise Test**
The Accelerometer Noise Test will verify the variation of the readings from the accelerometer over time and will also check if the noise detected is acceptable.

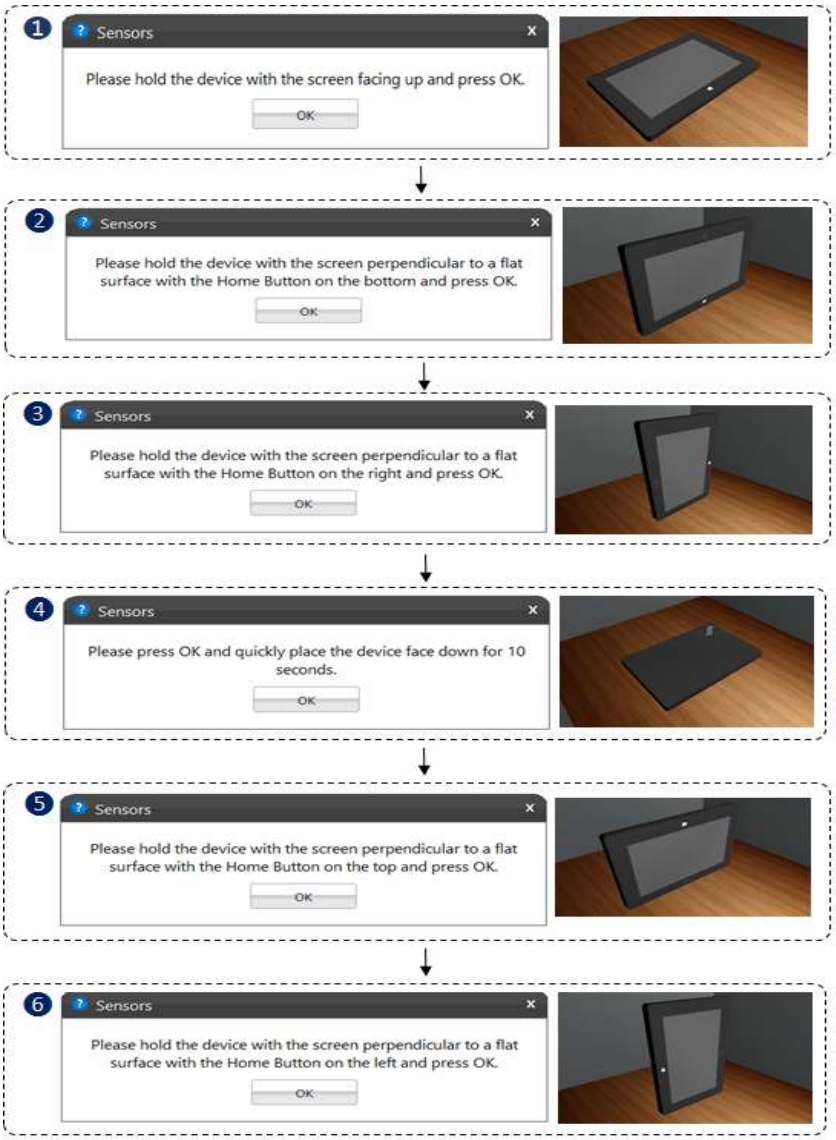
**Gyrometer Noise Test**
The Gyrometer Noise Test will verify the variation of the readings from the gyrometer over time and will also check if the noise detected is acceptable.

**Compass Noise Test**

The Compass Noise Test will verify the variation of the readings from the compass over time and will also check if the noise detected is acceptable.

**Accelerometer Interactive Test**

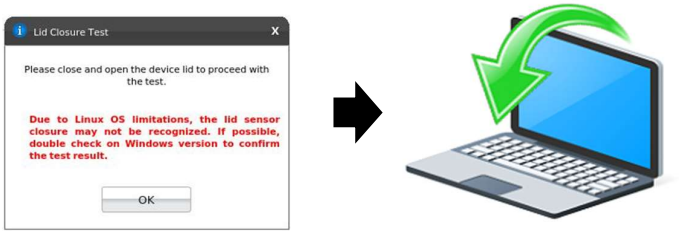
The Accelerometer Interactive Test will check if the values returned by the accelerometer in different positions are correct.



**Lid Closure Test**

This test verifies if the lid sensor is capturing the events of closing and opening the lid. The test will fail if these events are not captured.

The test will ask to the user open and close the notebook lid.



## 4.18 Storage

The Storage module is composed by the following tests:

| Test | Test type | Attendance |
|---|---|---|
| SMART Status Test | Quick | Unattended |
| Targeted Read Test | Quick | Unattended |
| Random Seek Test | Quick | Unattended |
| Funnel Seek Test | Quick | Unattended |
| SMART Short Self Test | Quick | Unattended |
| SMART Drive Self Test | Quick | Unattended |
| Default Self Test | Quick | Unattended |
| Device Read Test | Quick | Unattended |
| SMART Wearout Test | Quick | Unattended |
| Controller Status Test | Quick | Unattended |
| SMART Temperature Test | Quick | Unattended |
| SMART Reliability Test | Quick | Unattended |
| SMART Spare Space Test | Quick | Unattended |
| NVME Short Self-Test | Quick | Unattended |
| Extended Random Seek Test | Extended | Unattended |
| Extended Funnel Seek Test | Extended | Unattended |
| Extended Self Test | Extended | Unattended |
| Device Write Test | Extended | Unattended |
| Linear Read Test | Extended | Unattended |
| SMART Conveyance Test | Extended | Unattended |
| Full Disk Scan Test | Extended | Unattended |

**SMART Status Test**
Checks the status reported by SMART to quickly identify whether a device is defective or not.

**Targeted Read Test**
Checks the sectors in neighborhood of bad sectors reported in the SMART logs.

**Random Seek Test**
Checks the integrity of the Servomechanism of a device by checking sectors at several randomly chosen addresses.

**Funnel Seek Test**
Checks the integrity of the Servomechanism of a device by checking sectors following a "funnel" or "butterfly" pattern.

**SMART Short Self Test**
Checks electrical and mechanical component status as well as the reading ability of the device.

**SMART Drive Self Test**
Proprietary Lenovo Drive Self-Test (DST) that mixes sequential and random reads to the disk.

**Default Self Test**
Vendor specific test that runs a quick check.

**Device Read Test**
Tests if it is possible to correctly read sectors in different areas of the storage device.

**SMART Wearout Test**
SMART Wearout Test checks the wearout level of the attached SSD device by reading SMART attributes and informs whether the device is in good condition or has reached its wearout limit.

**Controller Status Test**
This test detects if the device behaves as expected.
**Note:** If the tested device is an NVMe, the test name will be preceded by the “NVME” word.

**SMART Temperature Test**
This test detects if the current temperature for the device is in critical state.
**Note:** If the tested device is an NVMe, the test name will be preceded by the “NVME” word.

**SMART Reliability Test**
This test detects if the device is still reliable based on SMART metrics.
**Note:** If the tested device is an NVMe, the test name will be preceded by the “NVME” word.

**SMART Spare Space Test**
This test detects if the spare space in the device is critically low.
**Note:** If the tested device is an NVMe, the test name will be preceded by the “NVME” word.

**NVME Short Self-Test**
A device self-test operation is a diagnostics test sequence that tests the integrity and functionality controller and can include testing the media associated with the namespaces.

**Extended Random Seek Test**
Works similar to quick random test, but the number of checked sectors is larger and it does not stop when the first bad sector is found.

**Extended Funnel Seek Test**
Works similar to quick funnel test, but the number of checked sectors is larger and it does not stop when the first bad sector is found.

**Extended Self Test**
Works similar to a quick short self test, but checks more sectors.
**Note:** If the tested device is an NVMe, the test name will be preceded by the “NVME” word.

**Device Write Test**
The Storage Device Write Test will verify if it is possible to write data on different areas of the device and then read the data correctly.

**Note:** Device Write Test requires the existence of, at least, one ext3, ext4 or ntfs type partition, with at least, 1.1gb of free space. Without this partition(s), the test won’t be able to be executed.

**Note²:** If the tested machine has Windows 8 or higher installed, please turn off the Fast Startup feature, as this option, if enabled, can prejudice the outcomes of Device Write Test.

To turn off the feature, on windows OS, go to Hardware and Sound > Power Options > System Settings, then uncheck the "Turn on fast startup" checkbox (admin access may be required).

**Note³:** When **BitLocker encryption is enabled**, the **partition isn't marked read-only** as it's with the "Fast Boot" feature. Instead, the partition remains as NTFS, read/write mode, but **outside of Windows it looks unformatted** (because the BitLocker feature is proprietary to Windows and scrambles the contents of the formatted partition). So **outside Windows it's not possible to mount** the partition. This may cause changes to the test's results. Bellow, is a table of the possible results, with each of the possible combinations of BitLocker/Fast-Boot, all of which are acceptable, due to the limitation of the features:

| Number | Fast Boot | Bitlocker | Windows Shutdown Mode | Device Write Test |
|---|---|---|---|---|
| 1 | Disabled | Disabled | Irrelevant | PASS<br>*Partition 1* |
| 2 | Enabled | Disabled | Shutdown | NOT APPLICABLE<br>*The test is not able to be executed, because the partition is locked by Windows. Please, disable Windows Fast Boot, Bitlocker and System Protection, do a clean shutdown with Windows (not hibernate) and run the test again.* |
| 3 | Enabled | Disabled | Restart | PASS<br>*Partition 1* |
| 4 | Disabled | Enabled (Partial) | Irrelevant | NOT APPLICABLE<br>*The available volumes do not fulfill the test requirements. Make sure to have at least one valid volume or partition with more than 1.1GB of free space and formatted as NTFS or ext4 or ext3.* |
| 5 | Enabled | Enabled (Partial) | Irrelevant | NOT APPLICABLE<br>*The available volumes do not fulfill the test requirements. Make sure to have at least one valid volume or partition with more than 1.1GB of free space and formatted as NTFS or ext4 or ext3.* |
| 6 | Disabled | Enabled (Full) | Irrelevant | NOT APPLICABLE<br>*The available volumes do not fulfill the test requirements. Make sure to have at least one valid volume or partition with more than 1.1GB of free space and formatted as NTFS or ext4 or ext3.* |
| 7 | Enabled | Enabled (Full) | Irrelevant | NOT APPLICABLE<br>*The available volumes do not fulfill the test requirements. Make sure to have at least one valid volume or partition with more than 1.1GB of free space and formatted as NTFS or ext4 or ext3.* |

As the mounting of this partition is technically challenging to overcome on linux, and not always we have access to the OS installed on the machine, we suggest this test to be skipped, in case BitLocker encryption is encountered.

**Linear Read Test**
Checks the integrity of the storage device by reading its sectors following a linear pattern.

**SMART Conveyance Test**
Checks the device integrity through the status returned by SMART Conveyance test.

**Full Disk Scan Test**
This test performs a full verification of the disk.

### Tests supported by device

It is important to emphasize that not all of the tests are supported by all of the storage device types. Bellow, is a relation of tests supported by device, in which we can see what test can be properly executed by each:

| Quick Tests | | | | | |
|---|---|---|---|---|---|
| **Test** | **SATA** | **SSD SATA** | **SSD NVMe** | **SSD EMMC** | **UFS** |
| SMART Status Test | X | X | - | - | X |
| Target Read Test | X | X | - | - | - |
| Random Seek Test | X | X | - | - | X |
| Funnel Seek Test | X | X | - | - | X |
| SMART Short Self Test | X | X | - | - | - |
| SMART Drive Self Test | X | - | - | - | - |
| Device Read Test | - | - | X | X | X |
| SMART Wearout Test | | X | X | - | - |
| SMART Health Test | X | X | - | - | - |
| NVMe Controller Status Test | - | - | X | - | - |
| NVMe SMART Temperature Test | - | - | X | - | - |
| NVMe SMART Reliability Test | - | - | X | - | - |
| NVMe SMART Spare Space | - | - | X | - | - |
| NVMe Health Prediction Test | - | - | X | - | - |
| **Extended Tests** | | | | | |
| **Test** | **SATA** | **SSD SATA** | **SSD NVMe** | **SSD EMMC** | **UFS** |
| Device Write Test | - | - | X | X | X |
| Linear Read Test | X | X | X | X | X |
| SMART Conveyance Test | X | X | - | - | - |
| NVMe Extended Self Test | - | - | X | - | - |
| Extended Random Seek | - | - | - | - | X |
| Extended Funnel Seek | - | - | - | - | X |
| Full Disk Scan | X | X | X | X | X |

**RST support**

Due to the lack of support for Intel's Rapid Store Technology (RST) on linux platforms, the storage module will not be enabled for testing if the option is active, and upon starting the application, users will be met with a warning, picturing the situation:


To mitigate this situation, please, disable the feature on BIOS, by setting the related configuration to "AHCI".

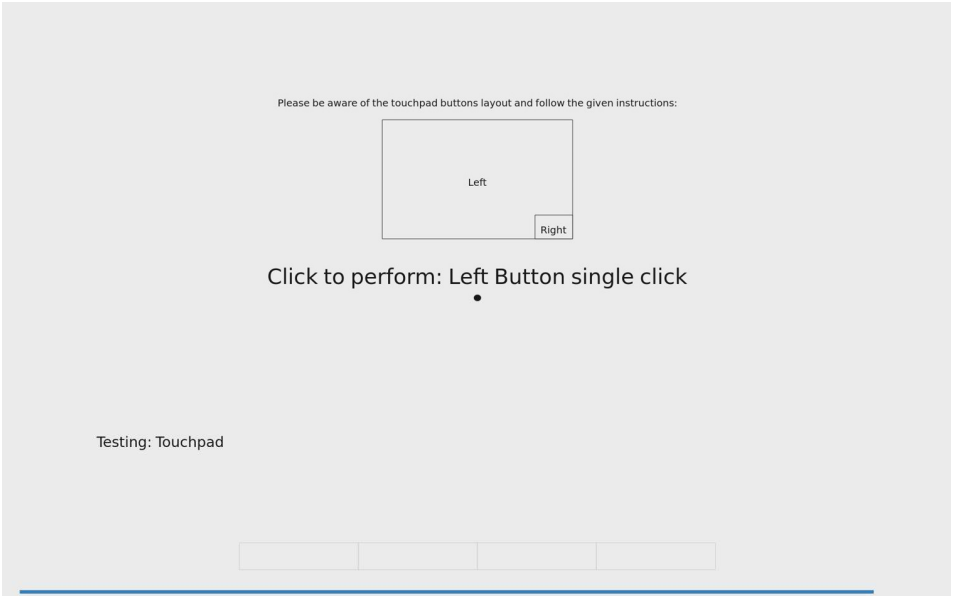
## 4.19 Touchpad Devices

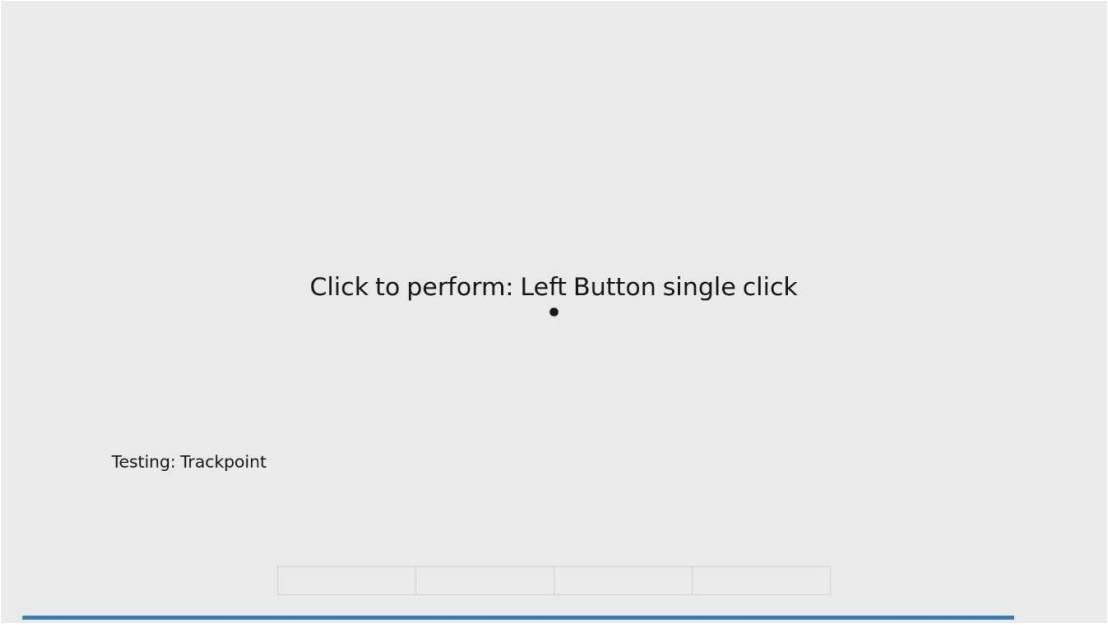
The Touchpad Devices module is composed by the following tests:

| Test | Test type | Attendance |
|---|---|---|
| Click Test | Quick | Attended |
| Precision Test | Quick | Attended |
| Press Precision Test | Quick | Attended |

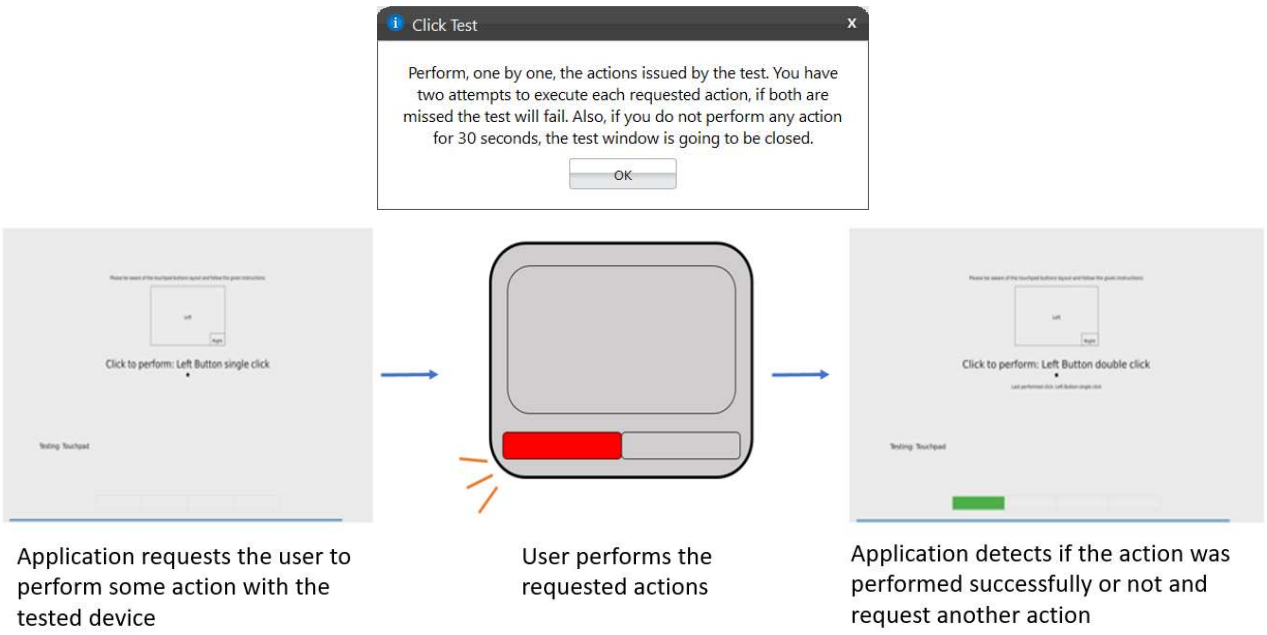
**Note:** If the machine has the Trackpoint device, the same tests will be performed separately. Also, all of the tests will be accompanied of a label, on the screen's left bottom area, informing which device is being currently tested.

**Click Test**




It does a check on the device buttons, issuing the user to perform some actions regarding them. The Click Test execution is based on the workflow below.

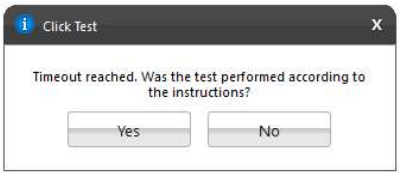



Application requests the user to perform some action with the tested device

User performs the requested actions

Application detects if the action was performed successfully or not and request another action

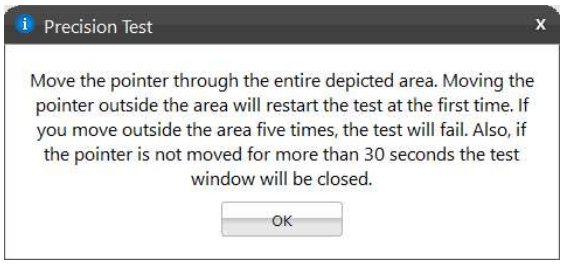
The user must perform one by one, the actions issued by the test. Note that the actions requested depend on the numbers and types of buttons present in your touchpad device.
The user has two attempts to execute each requested action. If the requested action was detected by the application, it is considered as passed. Otherwise the action is considered as “Failed”. The Click Test is considered as “Passed” if all actions required are passed.
If the user does not perform any action for 30 seconds, the following popup is displayed:

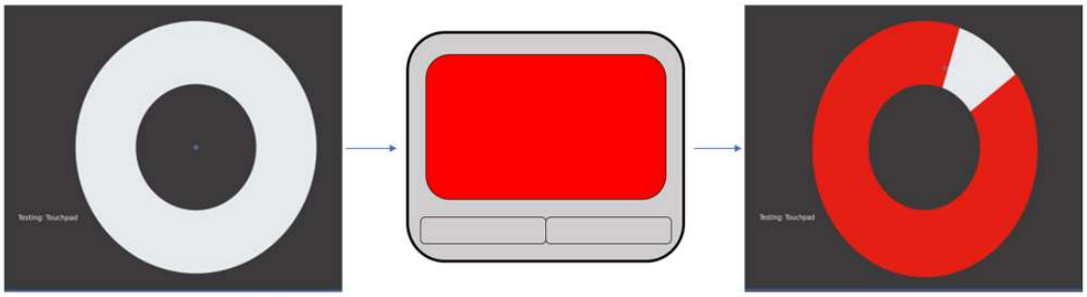



If the user selects “Yes” reporting that the test was performed according to the instructions, the test will fail. In this case, the application assumes that it was not possible to detect the device, indicating it is not working properly. If the user selects “No”, the test is canceled because the actions required were not properly executed.

**Note:** On the test’s background, a picture depicting the current layout of a touchpad device, in relation to the location of the area of each button, will be available to the user, with the objective of guiding it through the execution.

**Precision Test and Press Precision Test**
Tests the device movement precision. The Precision Test execution is based on the following workflows:

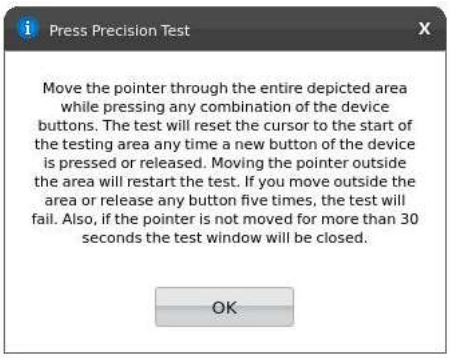


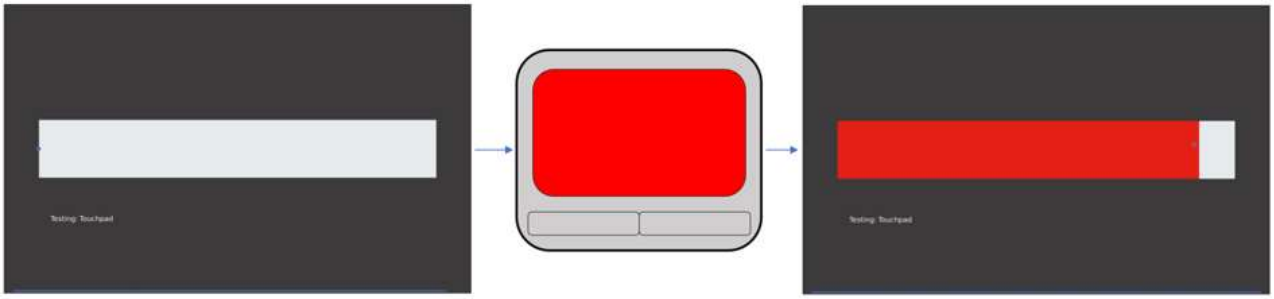



Application requests the user to move the pointer through the depicted area

User performs the requested action

Application detects if the pointer was moved without reaching the outside area



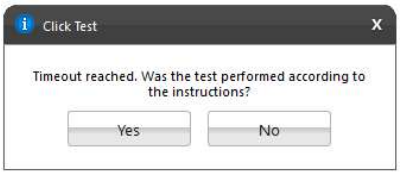



Application requests the user to move the pointer through the depicted area

User performs the requested action

Application detects if the pointer was moved without reaching the outside area

If the user gets to move the pointer through the entire depicted area without reaching the outside area, the test is finished as passed. If the outside area is reached two times the test will fail. If the pointer is not moved for more than 30 seconds the test windows is going to be closed and the following message is displaying to the user:




If the user selects “Yes”, reporting the test was performed according to the instructions, the test will be finished as “failed”. In this case, the application assumes that it was not possible to detect the device movement, indicating that the device is not working properly. If the user selects “No”, the test is canceled because the required actions were not properly executed.

**Note:** For executing Press Precision test, you must press any button from touchpad and move the pointer.

## 4.20 Touchscreen

The Touchscreen module is composed by the following tests:

| Test | Test type | Attendance |
|---|---|---|
| Basic Touch Test | Quick | Attended |
| Accuracy Test | Quick | Attended |
| Diagonal Test | Quick | Attended |
| Grid Test | Quick | Attended |
| Multi-touch Test | Quick | Attended |

### Basic Touch Test

The Basic touch test will verify if the system is receiving touch events. This test is based on the following workflow:


Display the test execution instructions → Display a full screen to be touched → User must touch in any area of the screen to complete the test

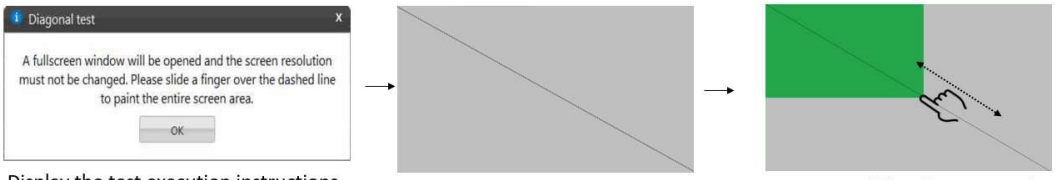
### Accuracy test

The Accuracy test will verify if the touchpoints are accurate with the screen mapping. This test is based on the following workflow:


Display the test execution instructions → Display a full screen to be touched in specific points → User must touch each point displayed in the screen to complete the test

### Diagonal Test

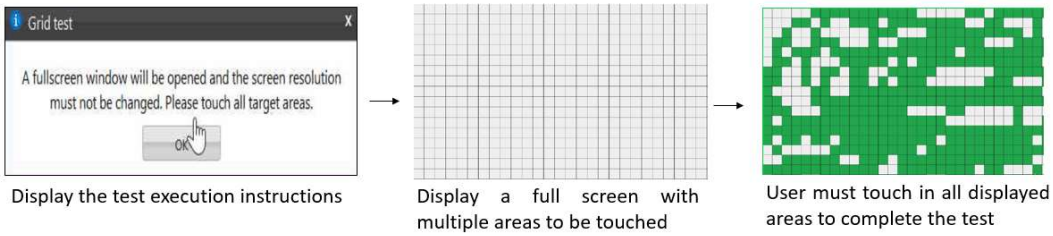
The Diagonal test will verify if rows and columns of the touchscreen are sensing through a diagonal gesture on the screen. This test is based on the following workflow:




Display the test execution instructions → Display a full with a dashed line → User must slide a finger over the dashed line to paint the entire screen area

**Grid test**

The Grid test will verify if any areas of touchscreen are not able to receive touch events.




Display the test execution instructions

Display a full screen with multiple areas to be touched

User must touch in all displayed areas to complete the test

**Multi-touch Test**

The Multi-touch test will verify if the system is receiving multi-touch events.


Display the test execution instructions

Display a full screen to be touched

User must make two or more touches simultaneously

## 4.21 Video Card

The Video Card module is composed by the following tests:

| Test | Test type | Attendance |
|---|---|---|
| Video Memory Test | Quick | Unattended |
| Texture Pipeline Test | Quick | Unattended |
| Mathematical Operations Test | Quick | Unattended |
| CUDA Modulo X Test | Quick | Unattended |
| CUDA Memory Fault Test | Quick | Unattended |
| CUDA Moving Inversion Test | Quick | Unattended |
| CUDA Address Test | Quick | Unattended |
| CUDA Block Move Test | Quick | Unattended |
| CUDA Random Number Sequence Test | Quick | Unattended |
| CUDA Standard Mathematical Operations Test | Quick | Unattended |
| CUDA Advanced Mathematical Operations Test | Quick | Unattended |
| Framebuffer Address Test | Quick | Unattended |
| Framebuffer Block Moving Test | Quick | Unattended |
| Framebuffer Walking Ones Test | Quick | Unattended |
| Framebuffer Deterministic Filling Test | Quick | Unattended |
| Framebuffer Random Filling Test | Quick | Unattended |
| Extended Video Memory Test | Extended | Unattended |
| Stress Test | Extended | Unattended |

**Video Memory Test**
Verifies if some data patterns are consistently read from and written to video card memory.

**Texture Pipeline Test**
Sends texture patterns to be rendered by the graphics pipeline and checks for loss of data when comparing input and output.

**Mathematical Operations Test**
Consists in basic mathematical operations such as addition, subtraction, multiplication, division.

**CUDA Modulo X Test**
This test tries to fill the video card every 20th memory location with a generated pattern, and the other locations with the pattern's complement, then it verifies if the values were correctly written.

**CUDA Memory Fault Test**
This test fills the memory with values alternating between 0 and 1 six times, and checking if the memory was correctly written.

**CUDA Moving Inversion Test**
This test writes a pattern to the entire memory, then proceeds to read it and write it's complement.

**CUDA Address Test**
This test writes to every memory location it's own address, and then proceed to verify them to check it's consistency.

**CUDA Block Move Test**
This test creates blocks of 8 bits, that are moved through the memory.

**CUDA Random Number Sequence Test**
This test fills the GPU memory with multiple random sequencies of numbers, and checks their consistency.

**CUDA Standard Mathematical Operations Test**
This test executes a series of standard mathematical operations on the video card, to ensure that the GPU's processing units are working as expected.

**CUDA Advanced Mathematical Operations Test**
This test executes a series of advanced mathematical operations on the video card, to ensure that the GPU's processing units are working as expected

**Framebuffer Address Test**
This test consists in writing to each memory address its own address, for each memory block.

**Framebuffer Block Moving Test**
This test fills all the accessible memory with a shifting pattern, that is, a value which is binary left shifted as it is written out through the accessible memory of every memory block.

**Framebuffer Walking Ones Test**
Consists on writing a pattern where only the rightmost bit is set (e.g.00000001), then shift this pattern to the left (e.g.00000010) until the end of the size of a byte, writing it again at the same memory address each time such pattern is shifted

**Framebuffer Deterministic Filling Test**
Consists in filling the memory buffer from every block with a pattern where all bits are 1 and then checking it. When checking for this pattern, it writes its binary complement was stored accordingly.

**Framebuffer Random Filling Test**
Consists in basic mathematical operations such as addition, subtraction, multiplication, division.

**Extended Video Memory Test**
Similar to Video Memory Test, but performs an extended analysis with more data patterns

**Stress Test**
Executes heavy operations on the video card for the purpose of stressing the GPU and verifying that the results remain reliable under stress.


**Note**: If your video card is manufactured by Nvidia, run the test with the power plug connected. Also, CUDA tests are only available for this vendor.

**Note²:** At the beginning of the module's execution, a pop-up will appear, informing the user to connect the charger, as the tests highly increase the energy consumption. This pop-up will be available when executing any test of the module directly through its own menu, through "Run All Tests" and through scripts.


**Note³:** At the beginning of the execution of any framebuffer tests, a pop-up warning the user of the flickering of the screen is going to be displayed.

## 4.22 Wired Ethernet

The Wired Ethernet module is composed by the following test:

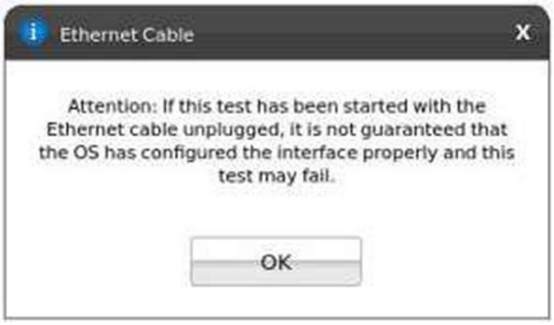
| Test | Test type | Attendance |
|---|---|---|
| Self-Test | Quick | Unattended |
| Internet Connection Test | Quick | Attended |

**Self-Test**
Checks the device's integrity by performing its driver's built-in self-tests.

Before starting, the Self-Test has a single pop-up informing the user that the test will reset it's connection.


**Internet Connection Test**
This test aims to verify if the ethernet port, when connected to a cable, can correctly access the internet.

Before starting, the Internet Connection Test has a single pop-up asking the user to connect an ethernet cable.



## 4.23 Wireless

The Wireless module is composed by the following tests:

| Test | Test type | Attendance |
|---|---|---|
| Radio Enabled Test | Quick | Unattended |
| Network Scan Test | Quick | Unattended |
| Signal Strength Test | Quick | Unattended |

**Radio Enabled Test**
Verifies if the wireless is turned on.

**Network Scan Test**
Verifies if the wireless adapter can detect available networks nearby. Make sure that there is a properly configured router or access point nearby before running this test.

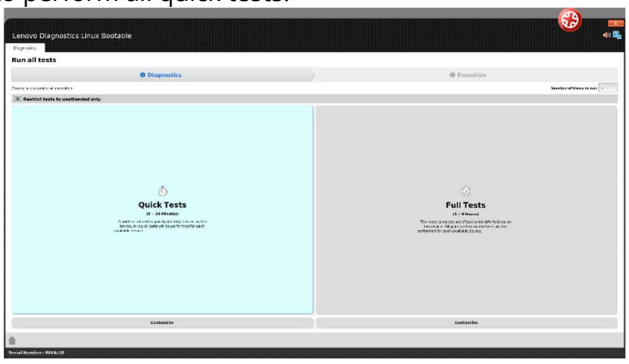
**Signal Strength Test**
Verifies the device scanning capabilities by trying to identify any connectable network nearby that has a signal quality of 20% or more. Make sure that there is a properly configured router or access point nearby before running this test.

# 5. EXPLORING LENOVO **RUN ALL OPTION**

Run All option allows to perform all supported tests from all supported modules at the same execution. In this flow, it is not possible to select devices, thus all devices will be tested. It is possible to choose if the Attended tests should be performed or not.
When the checkbox is marked, the application will run Unattended + Attended tests from the test execution option selected.

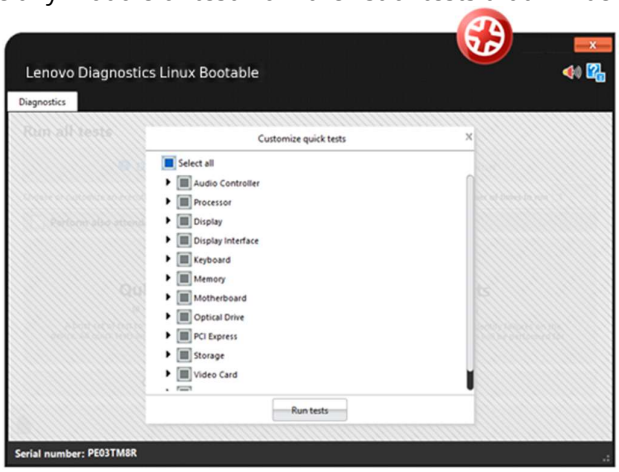
## 5.1 Quick tests

Click on Quick tests button to perform all quick tests.



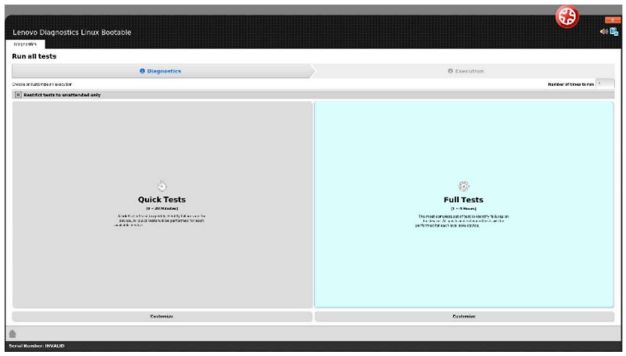

## 5.2 Quick tests (customized option)

Click to Customize to remove any module or test from the list of tests that will be performed.



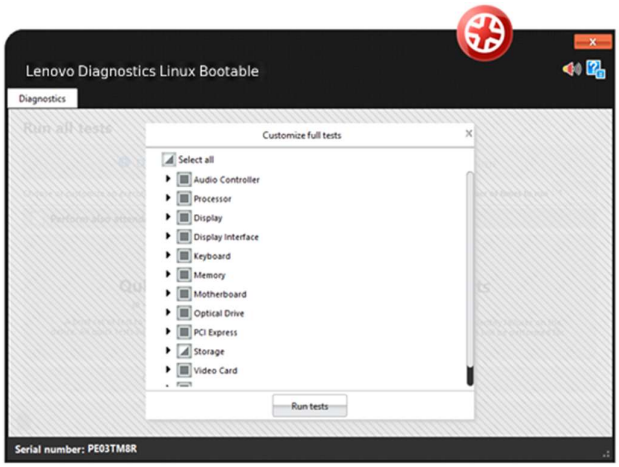

## 5.3 Full tests

Click on Full Tests button to perform all quick and extended tests.



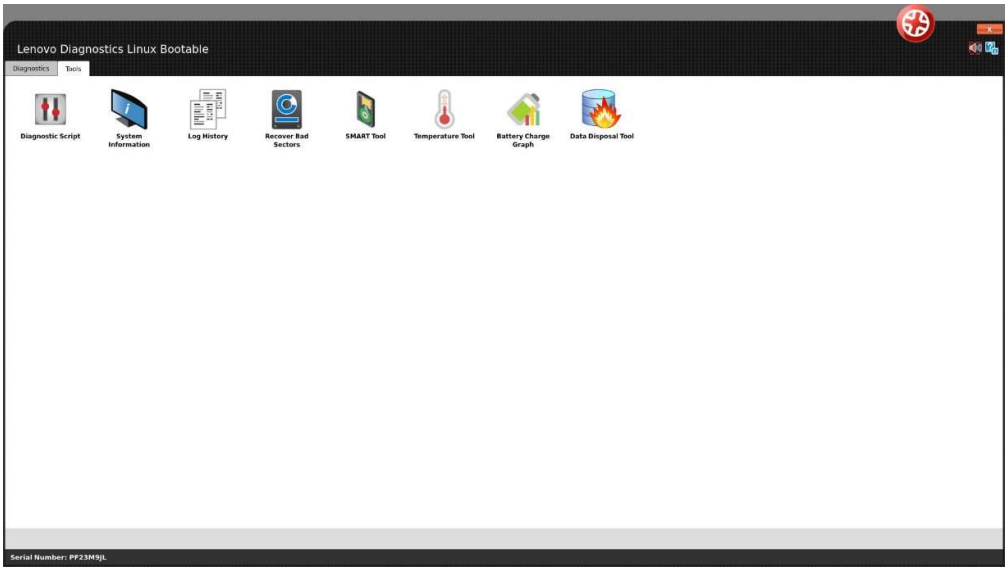

### 5.4 Full tests (customized option)

Click to Customize to remove any module or test from the list of tests that will be performed.



# 6. EXPLORING LENOVO **DIAGNOSTICS TOOLS**

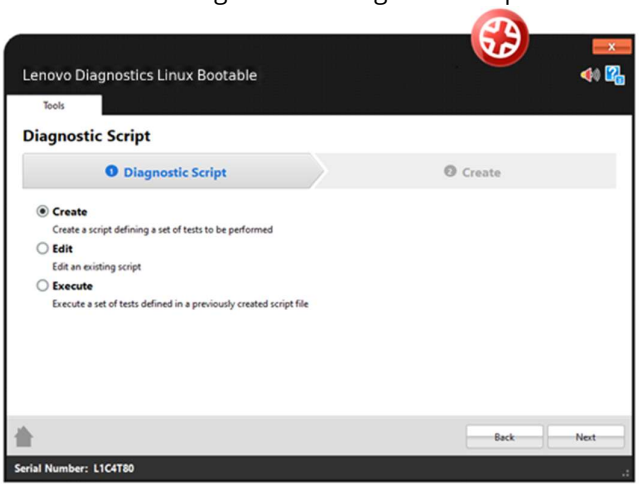
This section provides information about Tools:




## 6.1 Diagnostic Script

The Diagnostic Script is a tool that allows user to create a custom list of tests from any module. By selecting this tool, the following options are displayed:

- **Create**: allows to create a new diagnostic script.
- **Edit**: allows editing a diagnostic script saved previously.
- **Execute**: allows to perform the tests configured in a diagnostic script.

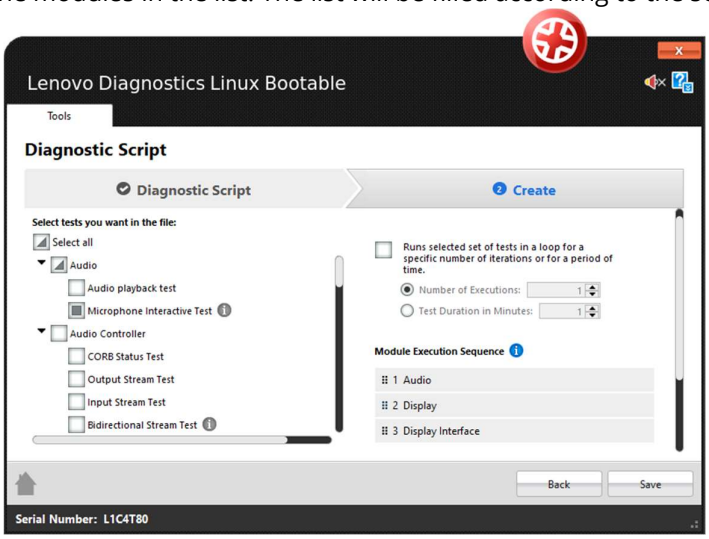


### 6.1.1 Create a diagnostic script

By selecting the option “Create” and clicking on the Next button, the screen below will be displayed. This screen allows selecting a set of tests to be performed from a list with all tests present in Lenovo Diagnostics. The tests not supported by the tested machine are marked with the icon .
It is also possible to configure the execution of this tests according one of the following parameters:

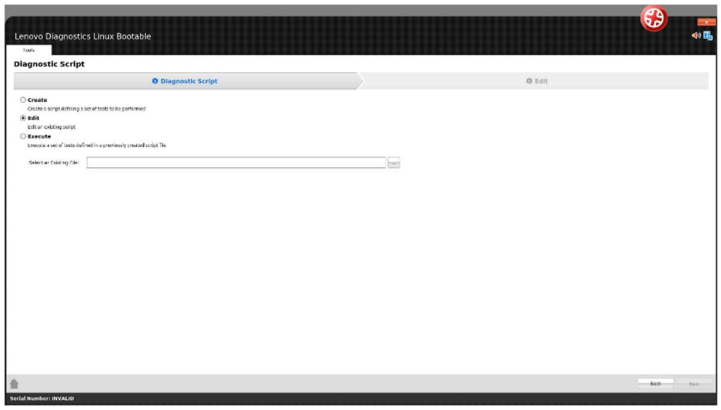
**Number of executions:** allows performing the tests according to a specific number of executions in a range from 1 to 999999999. In this case, the diagnostic will be finished when all iterations are completed.

**Test duration in minutes:** allows performing the tests according to a specific number of minutes in a range from 1 to 999999999. In this case, the diagnostic will be finished when this time is reached and all tests from the current iteration are finished. Notice that even if the time is reached the tests will be performed until the end in the current iteration.
**Module Execution sequence:** allows you to select the order in which the modules will be executed by dragging and dropping the modules in the list. The list will be filled according to the selected modules.

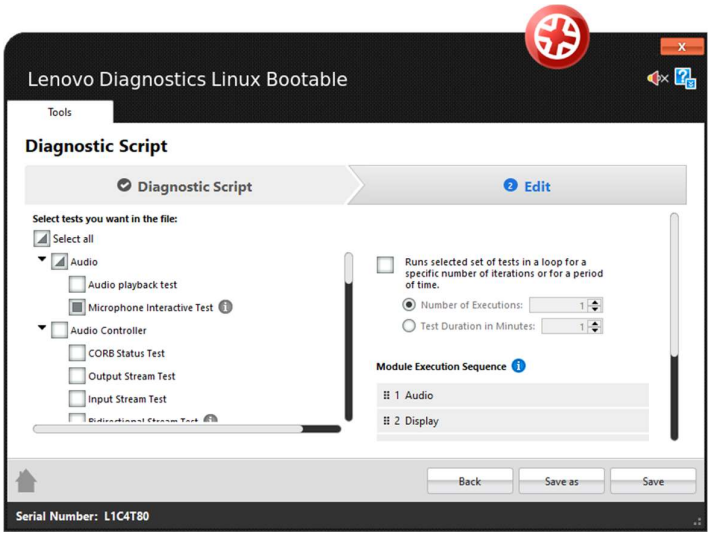



### 6.1.2 Edit a diagnostic script

By selecting the option “Edit”, the Lenovo Diagnostics displays a field where is possible to select an existing diagnostic script to be edited.

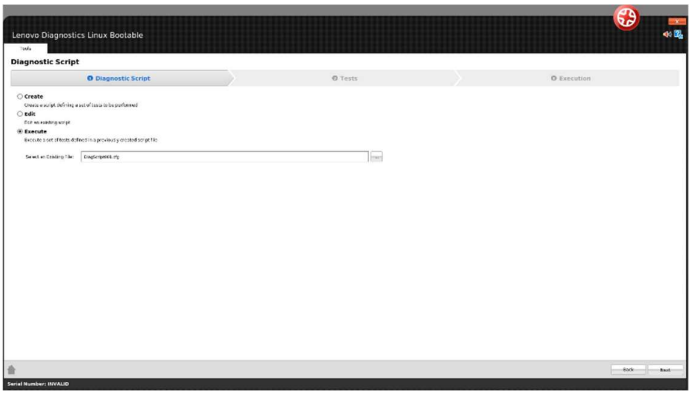



By selecting this file and clicking on the Next button, a screen is displayed with the configuration from the selected diagnostic script. Here it is possible to modify this configuration by changing the list of tests, modify the number of executions or duration minutes and change the order of modules execution. By clicking on Save button all changes are saved in the current file and by clicking on Save, it is possible to create a new file with the current configuration.

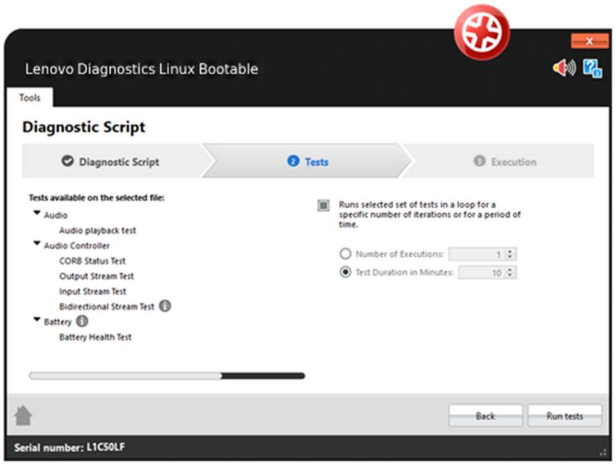



### 6.1.3 Execute a diagnostic script

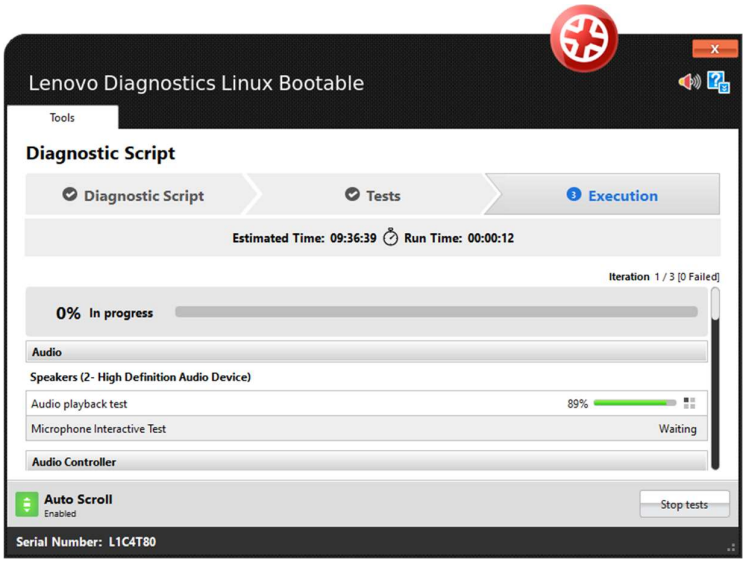
By selecting the option “Execute”, the Lenovo Diagnostics displays a field where is possible to select an existing diagnostic script to be performed.



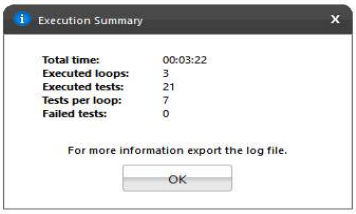

By selecting a valid diagnostic script file and clicking on the Next button, a screen is displayed with the configuration from the selected diagnostic script. All selected tests are listed selected and the not supported one are marked with the icon.



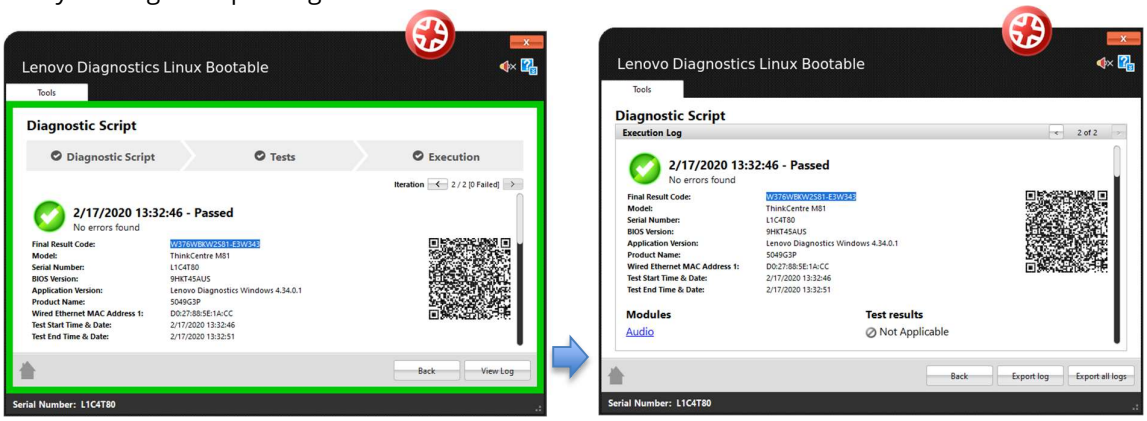

By clicking to Run Tests, the diagnostic script execution screen is displayed, and all supported tests are performed. The unsupported tests that do not have an associated device are filtered on this execution. It is possible to finish the execution anytime by clicking on Stop Test.




When the diagnostic is finished, the following popup is displayed with the execution summary.




The user can view the execution log by clicking View log.
It is also possible to export all executions log to a PDF file by clicking Export all log or export a specific iteration by clicking on Export log.




**Note:** When a script is generated with diagnostics not currently supported by the tested machine, they will be filtered automatically to do not integrate the execution.
The filtered diagnostics can be consulted on the JSON log generated by the execution, under the "unsupported_tests" key.

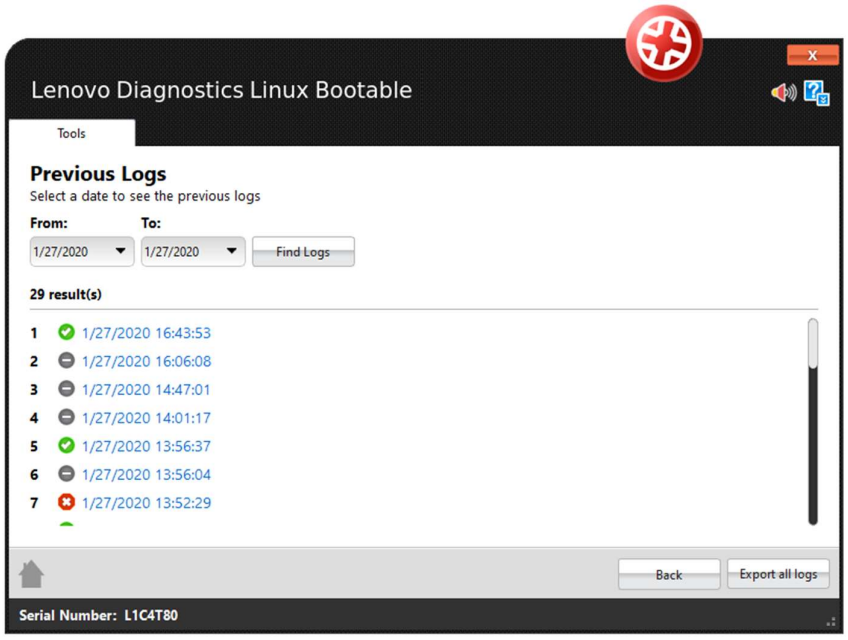
## 6.2 System Information

The system information tool allows the user to see general information about the system and the available modules devices. See in the screen below that is possible navigating between the modules and export the information by module or export all information at the same time.


**Note:** if the user connects an external display device, the tool is going to refresh the screen and make it available under the Display and Display Interface categories.

## 6.3 Log History

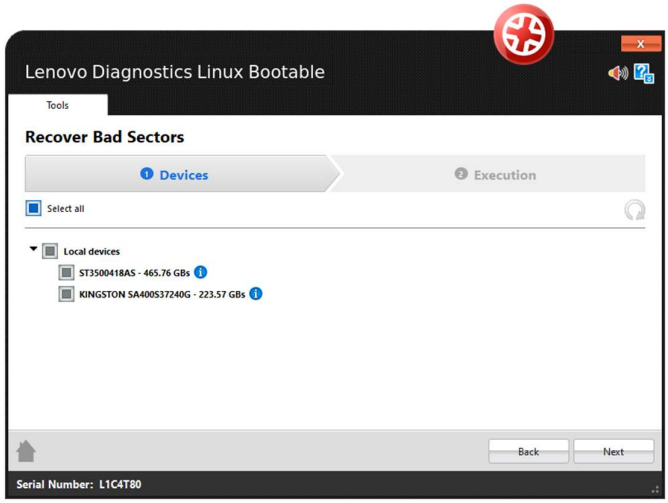
The Log History tool allows the user to see the logs of all executions performed in a machine. See in the image below it is possible to find logs by informing a period and export all logs.



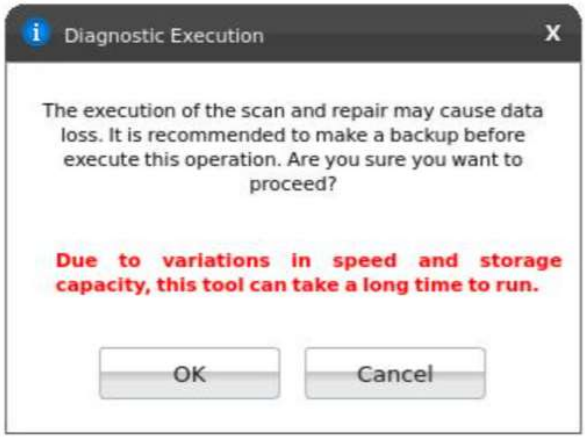

**Note**: It is not possible to remove the flash drive when running this tool.

## 6.4 Recover Bad Sectors

The Recover Bad Sectors tool allows the user to scan HDD/SSD/NVMe devices for bad sectors and fix them whenever possible.



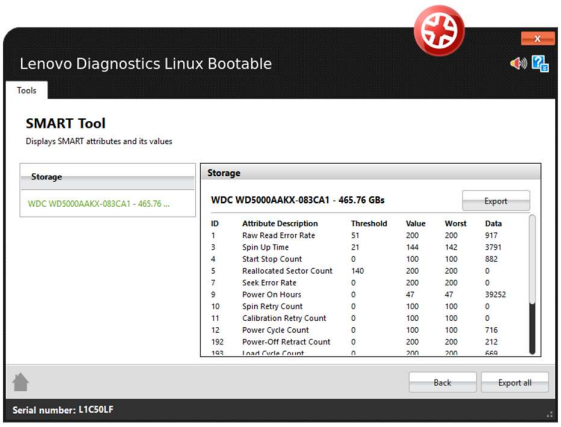

Be aware to perform a backup of your disk before performing this operation. The recovery operation can cause data loss and requires your confirmation.




## 6.5 SMART Tool

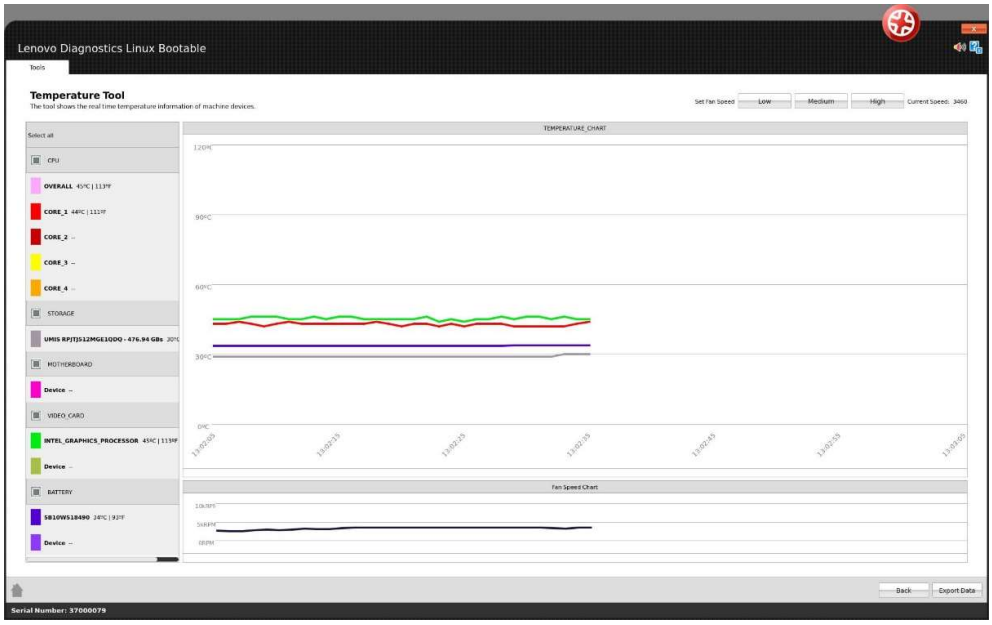
SMART tool provides a list of storage devices and shows, for each one, the SMART attributes and its information.

You are able to export that information to a PDF or HTML file. It is possible to export information for each storage device separately or for all available devices in the same file.




## 6.6 Temperature Tool

The Temperature Tool shows in real time the temperature information of machine devices.




It is possible to export the information on this screen.

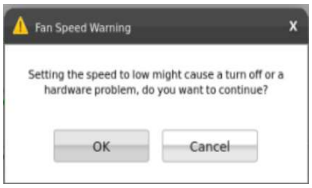

Once exported, a CSV file will be created on the USB flash drive.

### 6.6.1 Fan Speed Tool

It is possible to set the Fan Speed and compare Temperature X Speed.

Fan speed control buttons and RPM value at real time:

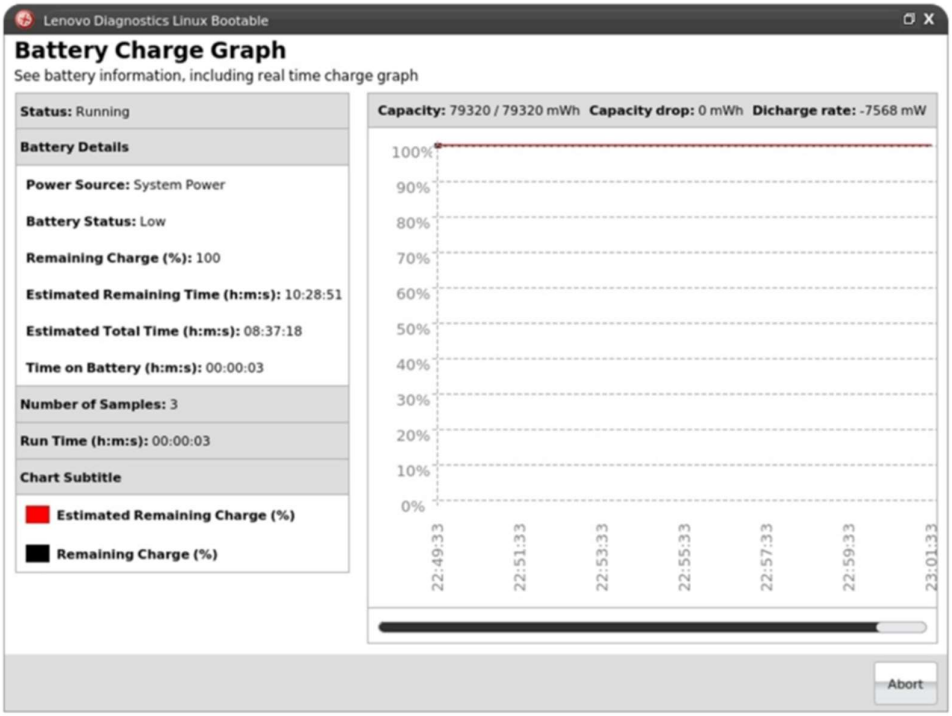

If the low speed is chosen, a popup warns the user about possible risks, once the user confirms his option, low speed is set.




**Note:**
1) Even on dual fan machines, only one line will be shown on chart, because the same speed level is always set on both fans.
2) Fan Speed Tool is not available for all machines.

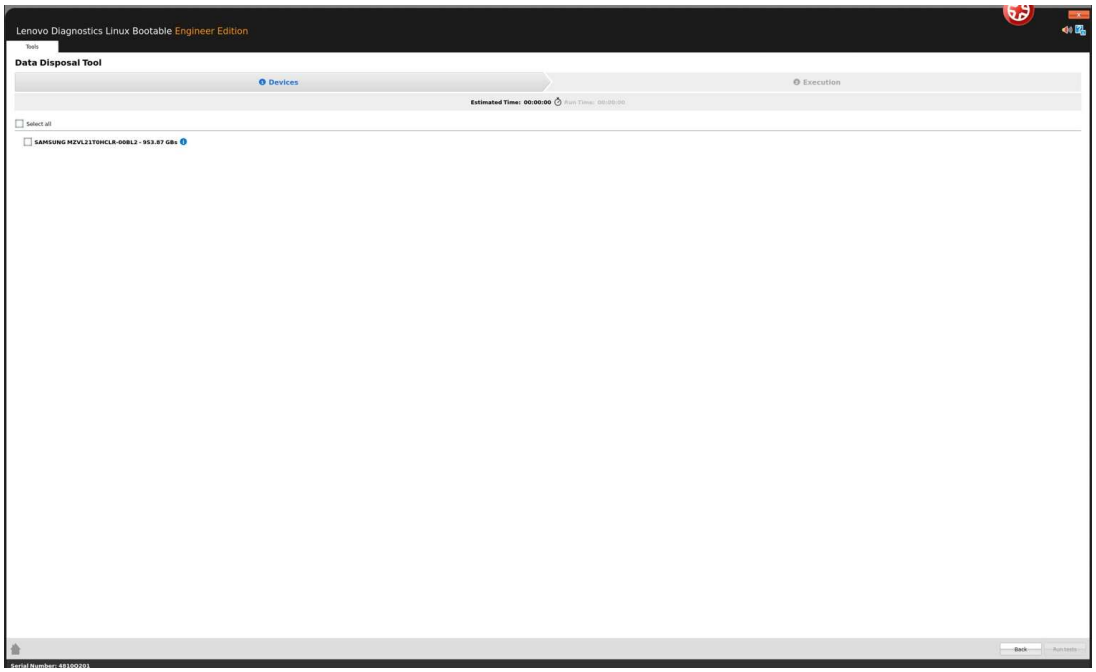
## 6.7 Battery Charge Graph

Battery information, including real time charge graph. It is possible to minimize this tool screen and navigate through the application running other tests and check the Battery Charge Graph behavior at the same time.

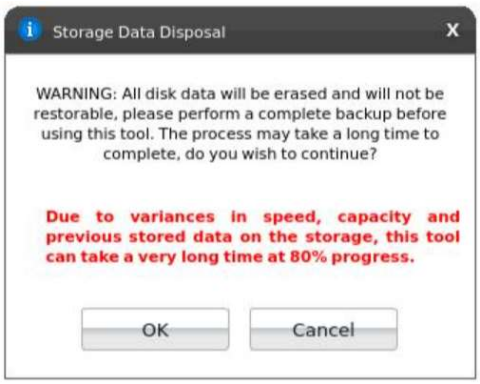


## 5.8 Data Disposal Tool

The data disposal tool allows the user to choose a storage device and completely erase all the data contained on it.




Upon requesting the execution of the tool, the user will be met with a warning stating the complete deletion of all of the drive's content, as well as a reminder of the possible long time the execution can take.




**Notes:**

1) Be sure to make a backup of any data you may need in the future, as the device will be completely erased.
2) Due to variances in speed, capacity and previous stored data on the storage, this tool can take a very long time after 80% progress.

# 7. GLOSSARY

**Extended Test**: Type of test that is performed in some hours.
**Quick test**: Type of test that is performed in some minutes.
**Unattended test:** A test that does not depend on user actions to be executed. All steps are performed automatically by the application.
**Attended test:** A test that depends on some user action to be executed.
**Module**: A module contains a set of tests that can be performed for a type of devices. It is enabled in the application only if the tested machine has at least one device supported by the module.
**Tool:** A functionality with a more complex and specific purpose, not necessarily related to diagnostics.