

👋 Dasharo User Group #8 🎉
Dasharo Open Source Firmware Validation Status



Agenda

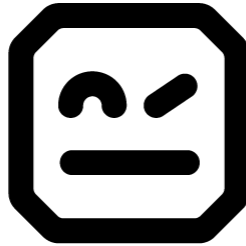
- Introduction to Dasharo Open Source Firmware Validation (OSFV)
- Current state
- Recent improvements
- Work in progress - current priorities
- Q&A

The background is a dark gray color with decorative circuit-like lines in a lighter gray color. These lines are located in the corners and form various geometric shapes, including straight lines, right angles, and small circles, resembling a printed circuit board (PCB) layout.

Introduction to Dasharo OSFV

Introduction to Dasharo OSFV

- Main purpose
 - validation of (open-source) firmware
 - can be used for any firmware, really
 - mainly Dasharo with UEFI payload right now
- Using Robot Framework as a base



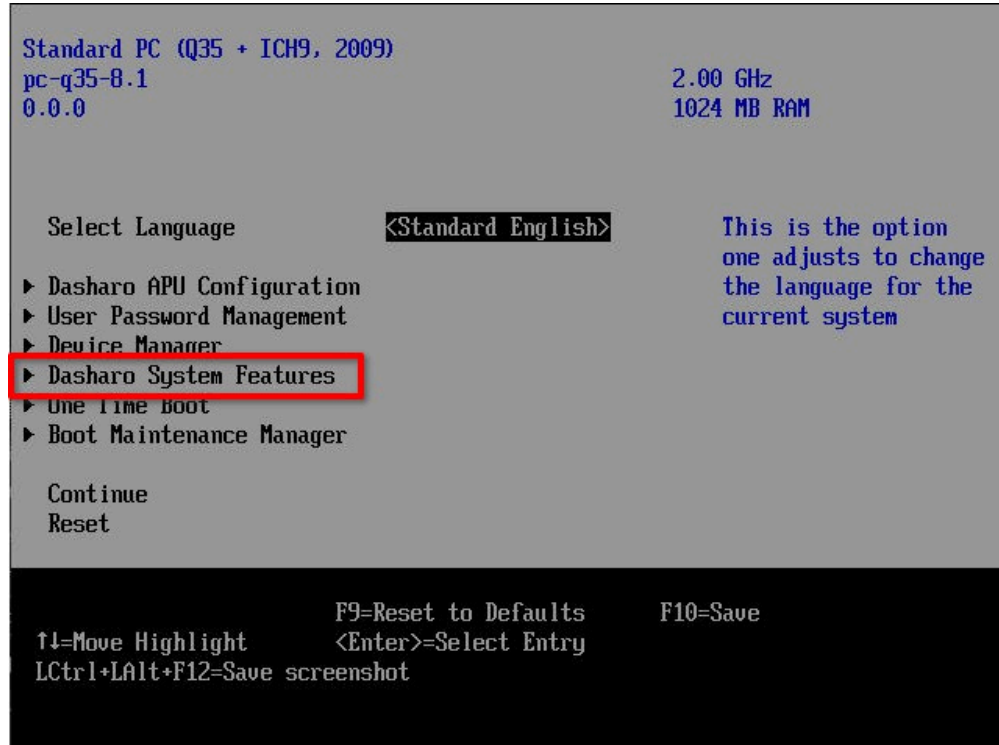
Introduction to Dasharo OSFV

- Use cases
 - testing Dasharo firmware releases
 - test-driven bug fixing (and adding new features)
 - regression testing
 - after introducing new features
 - after major changes (update base from upstream project)
 - validation of Dasharo-related tools (Dasharo Tools Suite, Dasharo Configuration Utility)
 - where possible, in QEMU

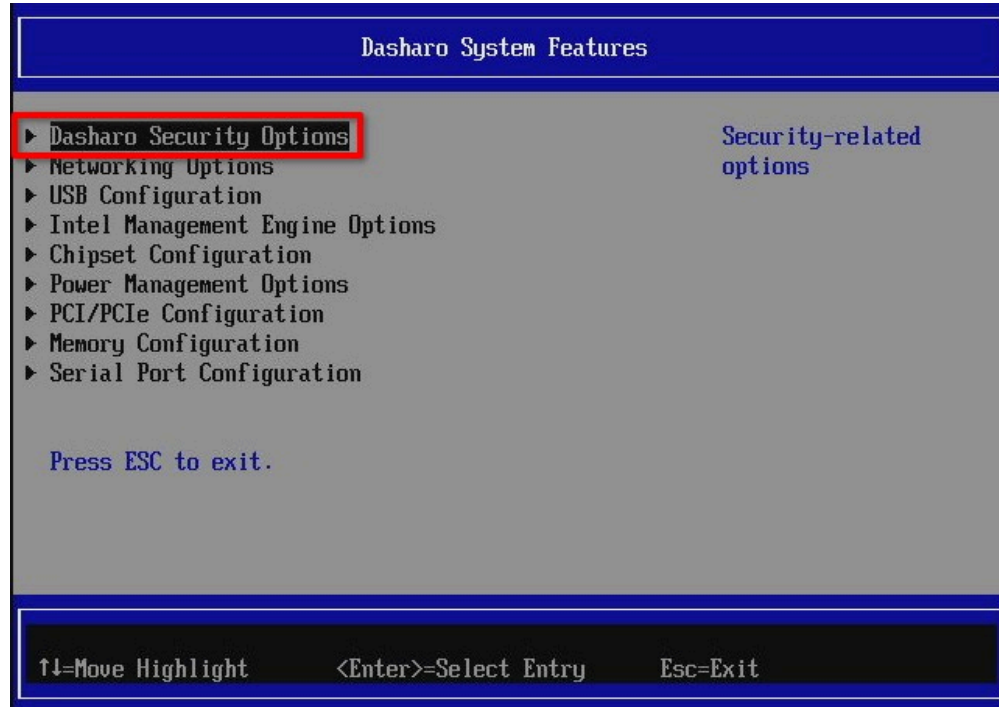
Typical setup

- Power control (power supply, power button)
- Serial console over telnet
- Move through firmware menus to switch certain options
- Check the result
 - In firmware
 - In OS
 - Dasharo Tools Suite - reference distribution for testing purposes
 - The latest Ubuntu LTS
 - The latest Windows 11
 - QubesOS for some tests

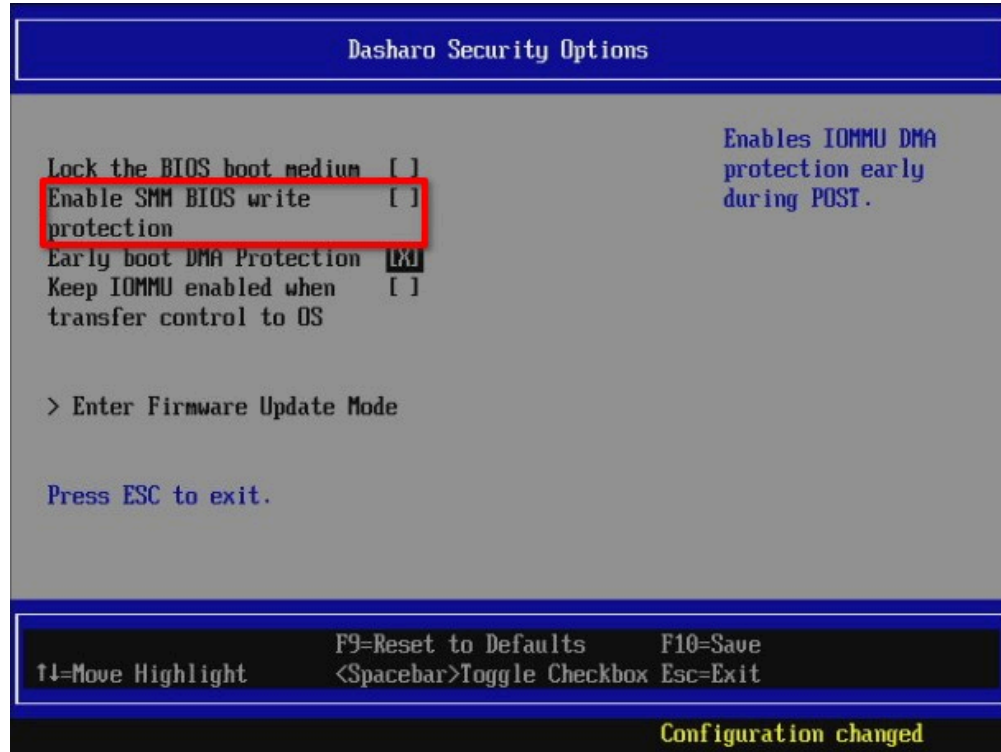
One test - step by step



One test - step by step



One test - step by step



One test - step by step

Power On

`${setup_menu}=` Enter Setup Menu Tianocore And Return Construction

`${dasharo_menu}=` Enter Dasharo System Features `${setup_menu}`

`${network_menu}=` Enter Dasharo Submenu `${dasharo_menu}` Dasharo Security Options

Set Option State `${network_menu}` Enable SMM BIOS write `${TRUE}`

Save Changes And Reset

Boot System Or From Connected Disk ubuntu

Login To Linux

Switch To Root User

Get Flashrom From Cloud

`${out_flashrom}=` Execute Command In Terminal `flashrom -p internal`

Should Contain `${out_flashrom}` SMM protection is enabled

The image features a dark gray background with decorative circuit-like lines in the corners. These lines are light gray and consist of straight segments connected at right angles, ending in small circular nodes. The lines are positioned in the top-left, top-right, and bottom-right corners, framing the central text.

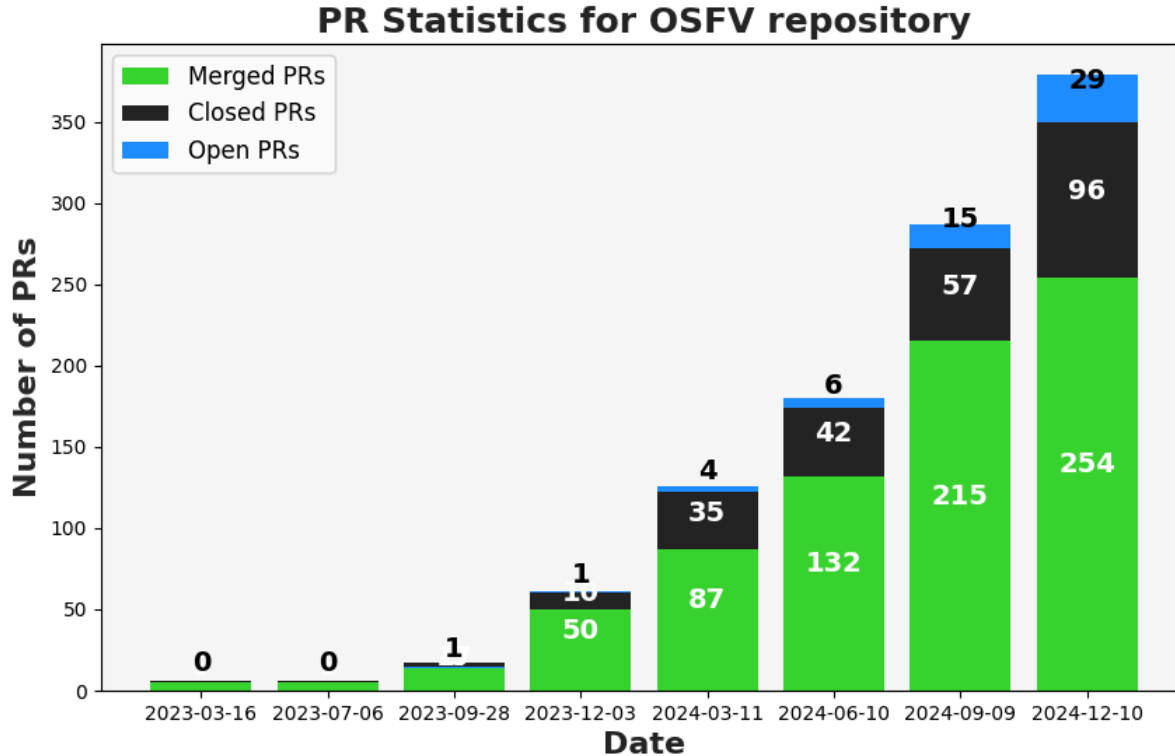
Current state

Current state

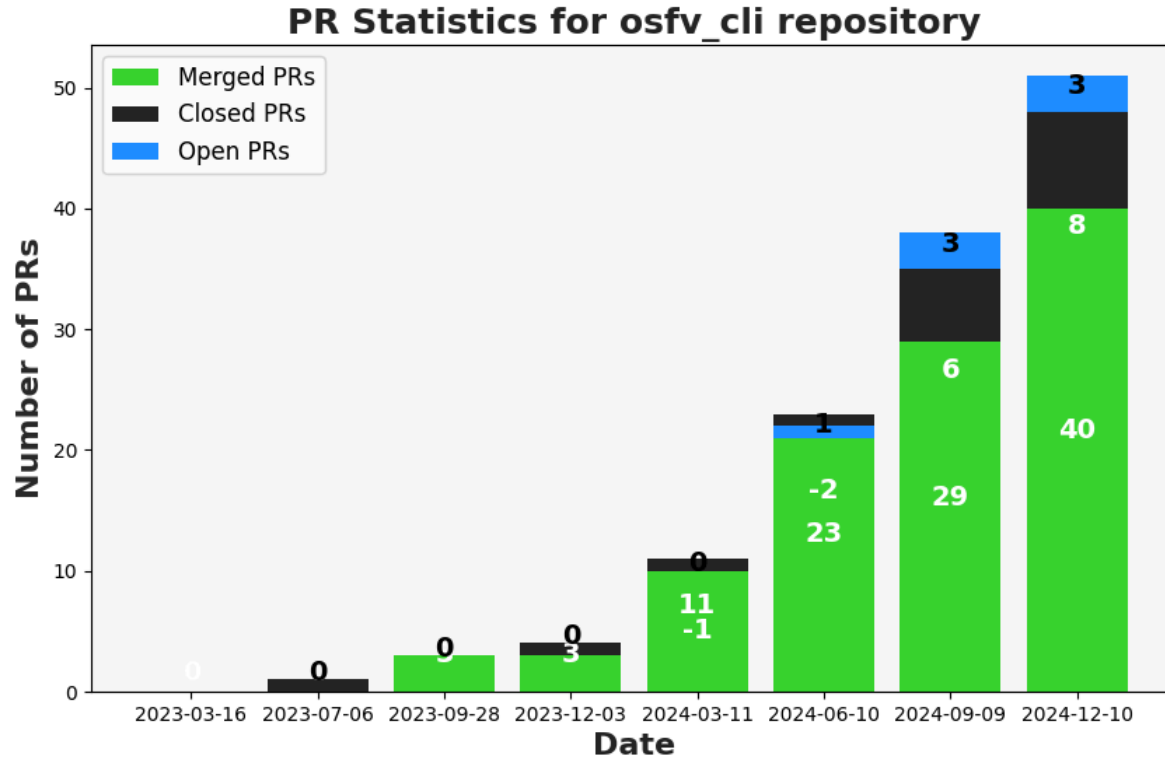
- 😞 No release 😞
- We are still failing to define (and reach) criteria for tagged release of test environment
- Right now it worked more like a rolling release in develop branch
- We wanted to merge develop into main once we are "ready" but it never happens
- It is confusing to see the default branch (main) with very little updates and outdated information
- Merged develop branch into main branch yesterday
 - we might use it as the default target branch until we figure it out



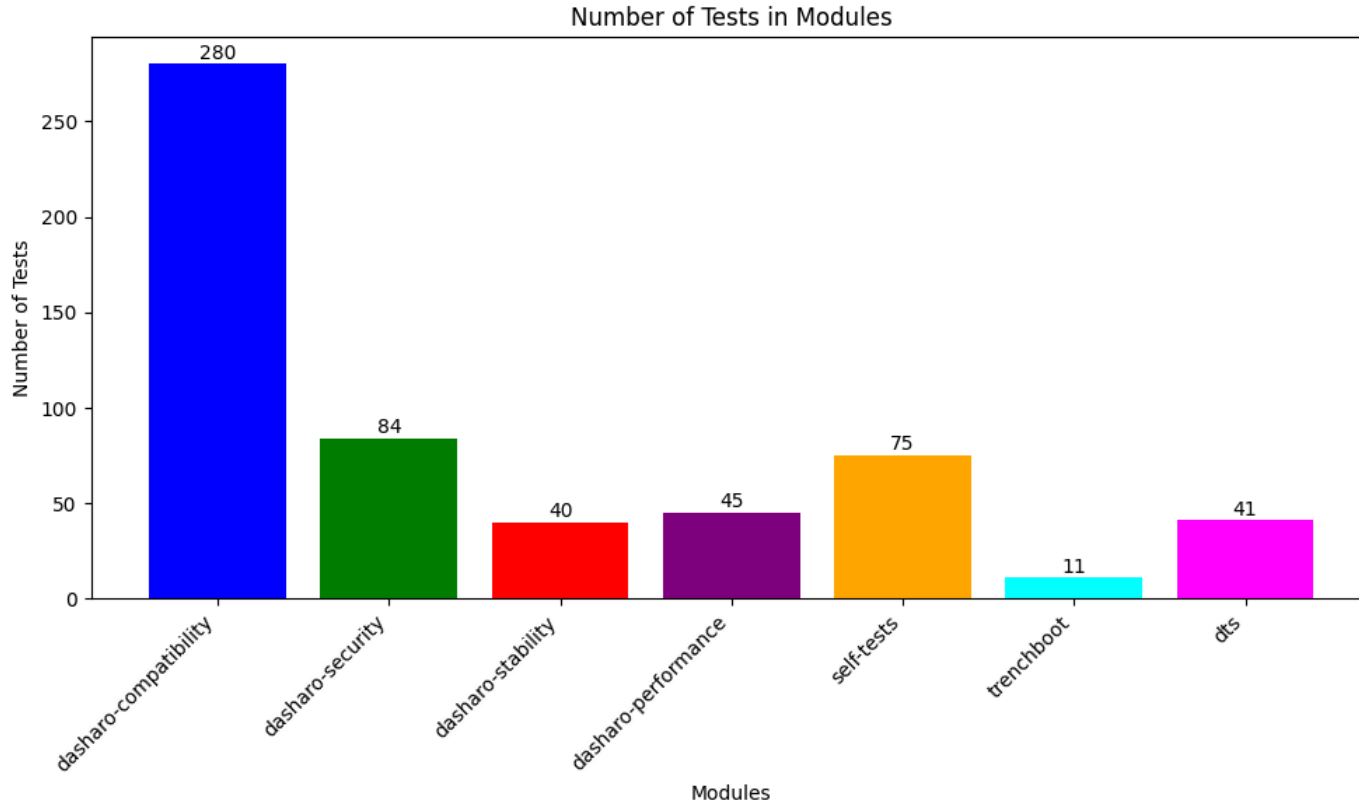
Statistics - Pull Requests



Statistics - Pull Requests



Statistics - tests



The background features a dark gray color with light gray circuit-like lines in the corners. These lines consist of straight segments connected by right-angle turns, with small circular nodes at the junctions and endpoints. The lines are most prominent in the top-left and top-right corners, with some extending towards the bottom corners.

Recent improvements

New tests

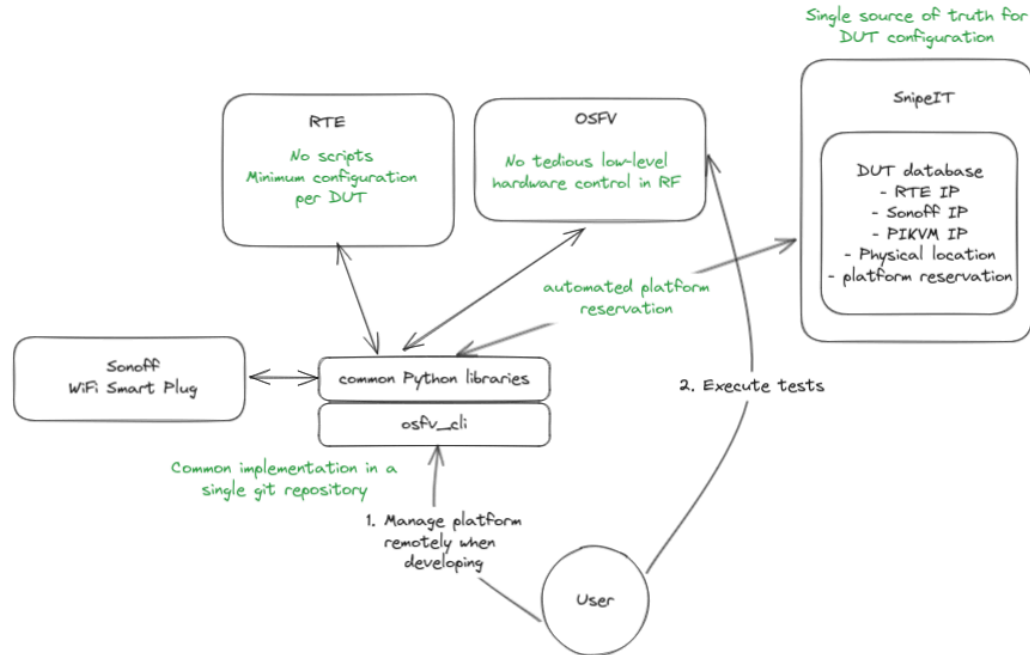
- Extended CPU suite (P/E cores, HT)
- Extended DCU tests
- Introduced more thorough testing of DTS in QEMU
- Suite for TrenchBoot development
 - Booting Linux and Xen from meta-trenchboot
- Suite for Capsule Updates
- More

New platforms

- Protectli VP2430
- Protectli VP3210, VP3230
- Odroid H4
- Bring back Dell Optiplex - both UEFI and SeaBIOS

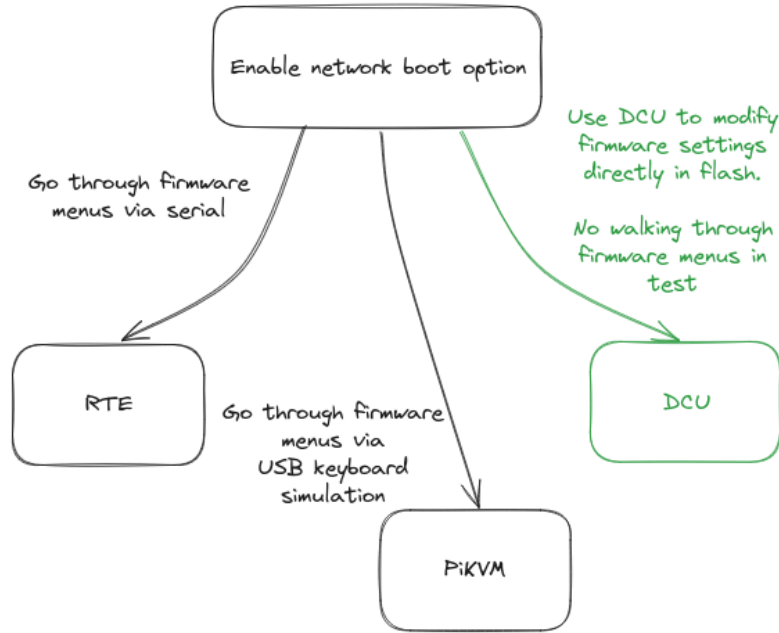
osfv_cli integration

- Integrate low-level hardware operations into Python libraries
- Reuse the same libraries by test framework and CLI tool



dcu integration

- Alternative interface for changing fw settings*
- Instead of manual steps, we can modify SMMSTORE variables directly



Keywords documentation

The screenshot displays a web interface for keyword documentation. On the left, a sidebar titled 'keywords' contains a search bar and a list of 150 keywords. The main content area on the right shows the documentation for five specific keywords, each with a title, a 'Documentation' section, and a brief description.

keywords

Search

Keywords (150)

- Login To Linux Via SSH Without Password
- Login To Linux With Root Privileges
- Login To Windows
- Login To Windows Via SSH
- OBMC Power Cycle Off
- OBMC Power Cycle On
- Open Connection And Log In
- Open Connection And Log In OpenBMC
- Power Cycle Off
- Power Cycle On
- Prepare Lm-sensors
- Prepare Test Suite
- Prepare To OBMC Connection
- Prepare To PiKVM Connection
- Prepare To Serial Connection
- Prepare To SSH Connection
- Read System Information In Petitboot
- Reboot In OPNsense
- Reboot In PiSense
- Reboot Via Linux On USB
- Reboot Via OS Boot By Petitboot
- Reboot Via Ubuntu By Tianocore
- Refresh Serial Screen In BIOS Editable Settings M

Reboot system with system installed on the DUT while already logged into Petitboot.

Reboot Via Ubuntu By Tianocore

Documentation

Reboot system with Ubuntu installed on the DUT while already logged into Tianocore.

Refresh Serial Screen In BIOS Editable Settings Menu

Documentation

This keyword tries to refresh the screen while inside the BIOS setting menu - to be specific while in a screen where you can press F10 to save the changes. Opening save windows and closing it should refresh the screen, but it is not guaranteed.

Remap Keys Variables From PiKVM

Documentation

Updates keys variables from PiKVM ones to the ones as defined in keys.robot

Remap Keys Variables To PiKVM

Documentation

Updates keys variables from keys.robot to be compatible with PiKVM

Remove Entry From List

<https://dasharo.github.io/open-source-firmware-validation/>

Next steps

- Finalize SeaBIOS support
- Need to focus more on repeatability and reliability than adding new tests/features
 - resolve Power On reliability: <https://github.com/Dasharo/open-source-firmware-validation/issues/607>
 - identify other areas generating problems, implement stress testing of these areas
 - add some HW into CI loop next to the QEMU tests
 - serial + relay - one of the Protectli boards
 - serial + PiKVM + Sonoff - one of the MSI boards

Questions?