Validation/Test Plan for UEFI Applications and Diagnostics

Kadin Brooks, Ivaylo Kozhuharov, Halla Tuaum

Sponsor: Intel

By Ivaylo Kozhuharov idk435@uw.edu

Revision History Block

Date	Version	Section(s)	Editor
10/23/2022	Version 0.1	All	Ivaylo Kozhuharov
10/30/2022	Version 0.2	2	Ivaylo Kozhuharov
11/13/2022	Version 0.3	2, 4, 5, 6, 8, 9	Ivaylo Kozhuharov
11/13/2022	Version 0.3	2, 4, 5, 6, 8, 9	Halla Tuaum
11/13/2022	Version 0.3	2, 4, 5, 6, 8, 9	Kadin Brooks
1/15/2023	Version 0.4	3, 7	Kadin Brooks
1/15/2023	Version 0.4	7	Ivaylo Kozhuharov
2/19/2023	Version 0.5	10, 11	Kadin Brooks
2/19/2023	Version 0.5	10, 11	Ivaylo Kozhuharov
4/9/2023	Version 0.6	2, 3, 4, 5	Kadin Brooks
4/19/2023	Version 0.6	6, 7, 8, 9, 10	Kadin Brooks

Table of Contents

1. Cover Sheet

- Test Plan:
- Project Title: UEFI Applications and Diagnostics
- Version: 0.6
- Date: 04/22/2023
- Authors: Ivaylo Kozhuharov, Halla Tuaum, Kadin Brooks
- Primary Point of Contact:

2. Introduction

The purpose of this document is to describe the testing methods that will be used in this project. Additionally this document will also contain the lists of the materials and software used to test each product. In addition, this document provides a schedule for when each element of the project will be tested.

This file contains the test plans for:

- CPU Diagnostics Tool:
    - The CPU diagnostics tool is a script that will use the CPUID instruction to gather information about the CPU and then display it in a human readable format in the UEFI shell.
    - First test will be to ensure that the base CPUID UEFI Shell executable will be run using the Simics simulator.
    - The second test will be to ensure that the base CPUID UEFI Shell executable will be run on the UP Xtreme board.
    - The final test for this tool will be conducted on the UP Xtreme board. The goal of the test will be to test the custom CPUID Shell application.

- Memory Diagnostics Tool:
    - The memory diagnostic tool will profile the memory reserved for system firmware and running Shell executables.
    - First test will be to report the memory allocated for system firmware. We will perform this test using Simics and UP Xtreme board.
    - Second test will be to ensure that the memory leak detection feature works for Shell executables. We will perform this test using Simics and UP Xtreme board.
    - Final tests will be conducted on the UP Xtreme board. The goal of the test will be to test the custom memory management Shell application.

- System Management Mode Optimization Tool:
    - The SMI handler profile feature will report the function name of the SMI handler, the name of the dispatcher the handler is registered with, the source file name and line number, and the SMI context.
    - First test will be to run the default SMI handler application using Simics.
    - Second test will be to run the default SMI handler application on the UP Xtreme board.

System Overview Diagrams

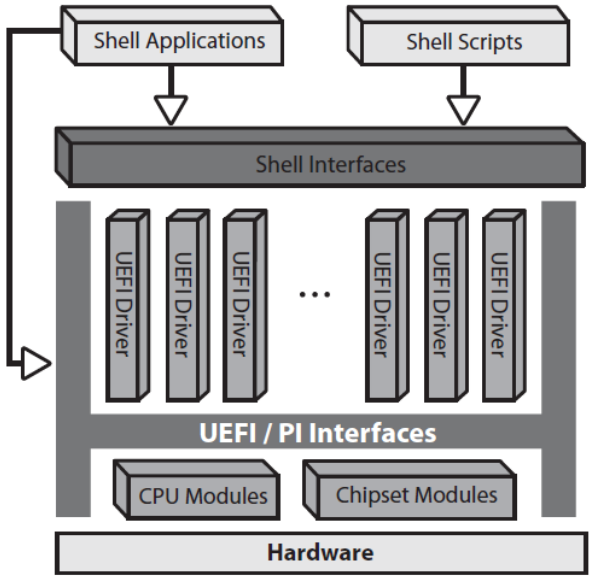

Figure 1. Architectural Relationship Between UEFI Shell and Underlying System Firmware and Hardware (2)

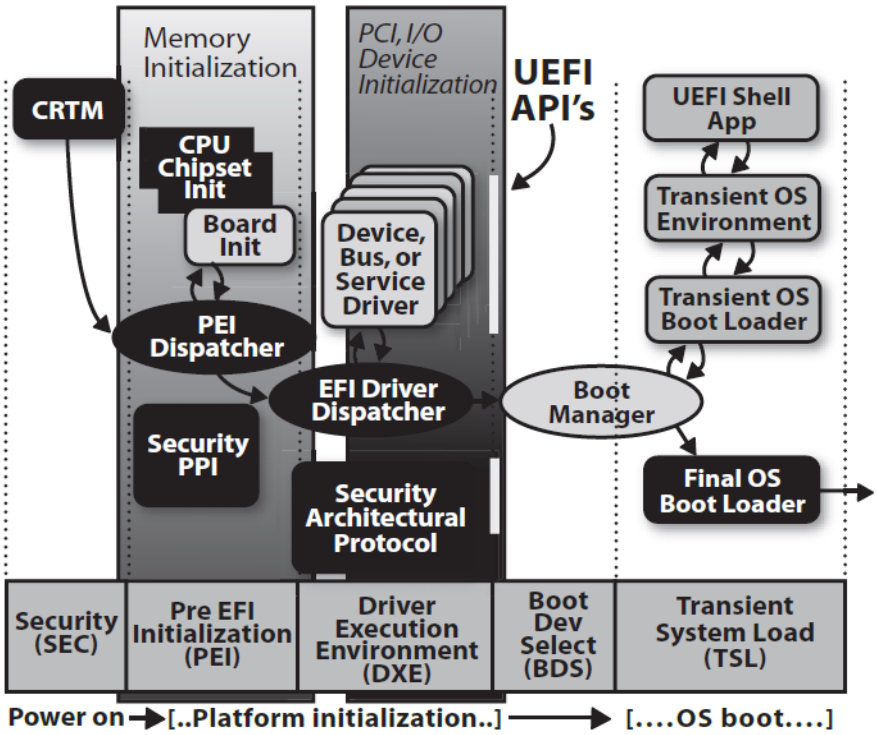

Figure 2. Role of UEFI in Platform Booting Sequence (2)

3. Configurations To Be Tested

The UEFI Shell .efi executables will be tested using the Simics simulator and the Whiskey Lake UP Xtreme board student kit provided by Intel. We will flash our custom firmware image to the board and execute .efi executable files from a USB flash drive. We will test the function of each Shell application separately and make modifications to the Shell executables iteratively. We will first build and test the base Shell applications included in the Intel training materials using the Simics simulator tool. After confirming functionality using Simics, we will proceed to build the base Shell application for the UP Xtreme board. After confirming functionality of the base applications on the UP Xtreme board, we will make custom modifications to the applications which we will then run from the UP Xtreme board.

4. Test Items

This section lists all tests that are going to be described in this document:

1. Hardware Test Items:
    a. For our project, we must validate our student firmware kit provided by Intel. We will follow a protocol provided by Intel for testing the function of our processor, flash programmer, and SSD storage before running our UEFI Shell applications.
    b. We will only test the features of the hardware needed for running our UEFI Shell scripts.

2. Software Test Items:
    a. As this project is primarily software based, tests will need to be performed for each of our three targeted deliverables:
        i. CPUID Output Processing Tool
            1. Can the tool extract and report CPU information on Simics and UP Xtreme board.
            2. We will limit our testing to CPU information related to the number of cores, the number of threads, processor name/specification, process length, instruction set support, and cache topology in the student firmware kit provided by Intel.
        ii. Memory Profiling Tool
            1. Can the tool report an accurate description of memory topology on Simics and UP Xtreme board.
            2. Can the tool detect memory leaks of Shell applications accurately.
            3. We will limit our testing to hardware memory reservation of the UEFI firmware implementation on the student firmware kit provided by Intel and memory leak detection of Shell applications.
        iii. SMI Handler Processing Tool
            1. Can the tool report the function name of the SMI handler, the name of the dispatcher the handler is registered with, the source file name and line number, and SMI context on Simics and UP Xtreme board.
            2. We will largely limit our testing to the base SMI handler application included with the Intel training materials.
3. System Test Items:
    a. We will need to test the UEFI Shell executables in Simics before testing on the UP Xtreme board.
    b. We will need to test the base Shell applications and the Shell applications after making custom modifications.

5. Testing Pass/Fail Criteria

Test Items	Pass/Fail Criteria
Hardware Test Items	We will follow the specific protocol provided by Intel for testing the student firmware kit. Differences from the specified protocol will indicate test failure.
CPU Diagnostic Tool	We will run our UEFI Shell application and then determine whether the output corresponds to the CPU in the student firmware kit.
Memory Topology Tool	We will run our UEFI Shell application and then determine whether the output matches the UEFI firmware implementation in the student firmware kit. We will also determine whether the memory leak detection feature correctly indicates allocated memory of Shell executables.
SMM Handler Tool	We will run our UEFI Shell application and then determine whether the SMI handler application reports the handler profile information.
System Test Items	We will test our deliverables on Simics before proceeding to testing on the UP Xtreme board. We will test the base applications before proceeding to our custom applications.

6. Procedures For Testing

- Materials:
    - A computer system that can run the UEFI shell (for our testing we will be using the student firmware kit provided by Intel and the EDK 2 Platform by Aaeon)
    - Intel Simics software and Intel Simics software packages for hardware simulation and prototyping of UEFI shell applications.
- Training:
    - Harry Hsiung will provide any special training needed before starting.
- Metrics:
    - Correctness. We need to establish that each UEFI shell executable functions properly based on the set of desired specifications specified in the PRD. These tests will be performed using the Intel Simics hardware simulation tool and then the physical UpXtreme board.
    - Portability. The UEFI shell executables should be able to be loaded onto a USB flash device and be able to be executed from computers that can boot to the UEFI shell.
    - Debugging records. We will perform tests that track the state of our applications during various points of testing to assure validity while using the Intel Simics hardware simulation tool and then testing on the physical UpXtreme board. We will keep records of the UEFI shell executables in documented output text files.
- Configurations:
    - The CPUID Shell executable will be tested using the Intel Simics hardware simulation tool and the UpXtreme board. We will only test the base features included in the Intel training materials.
    - For the memory profiling Shell executable will be tested using the Intel Simics hardware simulation tool and the UpXtreme board. We will enable the memory profile feature by configuring specific PCDs for UEFI memory profiling, EfiBootServicesData memory profiling, and UEFI shell memory profiling. If these features are functional, we will add memory profiling features for the DXE_CORE.
    - The SMI handler profile Shell executable will be tested using the Intel Simics hardware simulation tool and the UpXtreme board. We will only test the base features included in the Intel training materials.
- Regression Testing:
    - We will retest each feature of each application after every major change or addition. For the CPU feature enumeration, we will perform one regression test. For the memory topology feature, we will perform one regression test for the base features and one regression test for the additional features. For the SMI handler profile Shell executable, we will perform one regression test for the base features.
- Other Testing:

    - For all other features that must be tested, we will direct questions and follow up to our project sponsor Harry Hsiung and Laurie Jarlstrom.
- Testing Materials:
    - We will test each UEFI Shell executable using the Intel Simics hardware simulator tool and the physical UpXtreme student kit. We will also refer to the hardware specifications of the student firmware kit to validate certain results from the UEFI Shell executables.

7. Testing Deliverables

We will test any engineered firmware volumes using the Intel Simics hardware simulator and our UEFI shell executables before proceeding to testing the firmware volumes flashed onto the physical UpXtreme board and our UEFI shell executables. We will be able to provide output from the UEFI shell executables to the shell terminal and to output text files as evidence of the functionality of our engineered firmware volumes and UEFI shell executables.

For the CPUID UEFI shell executable, we will first use the default UEFI shell executable included with the Intel training materials using the Intel Simics hardware simulator. After confirming output from the Simics hardware simulator, we will proceed to build the default UEFI shell executable for the UpXtreme board for initial testing. After confirming output of the default UEFI shell executable via an output text file, we will make modifications to the default shell executable source code file to satisfy the requirements specified in the PRD documentation.

For the memory profiling and memory leak detection feature UEFI shell application, we will first proceed to build the default UEFI shell executable for the UpXtreme board for initial testing. After confirming the functioning of the default UEFI shell executable via an output text file, we will make modifications to the default shell executable source code files and metadata files to expand the functionality of the memory profiling features and add memory leak detection functionality. We will then proceed to build the modified UEFI shell executable for the UpXtreme board for testing via an output text file.

For the SMI handler profiling feature UEFI shell executable, we will first use the default UEFI shell executable included with the Intel training materials using the Intel Simics hardware simulator. After confirming output from the Simics hardware simulator, we will then proceed to build the default UEFI shell executable for the UpXtreme board for initial testing. We will then confirm output of the default UEFI shell executable via an output text file.

8. Roles/Responsibilities

CPUID Output Processing Testing

- This test requires 1 person.
- A tester will be responsible for validating the default and final UEFI shell executable on the Intel Simics hardware simulator and the UpXtreme board.

Memory Profiling Feature Testing

- This test requires 1-2 people.
- A tester will be responsible for validating the default UEFI shell executable on the UpXtreme board.
- A tester will be responsible for validating the final UEFI shell executable on the UpXtreme board.

SMI Handler Profiling Testing

- This test requires 1 person.
- A tester will be responsible for validating the default UEFI shell executable on the UpXtreme board.

9. Schedule

Testing will start after building the default CPUID processing UEFI shell application for the Intel Simics hardware simulator. We will test the validity of the shell executable before proceeding to make any modifications for further testing. We will then proceed to build and test the modified UEFI shell executable on the Intel Simics hardware simulator before proceeding to building and testing the modified shell executable on the UpXtreme board.

After validating the CPUID processing UEFI shell executable, we will then build the default memory profiling UEFI shell application using the Intel Simics hardware simulator if possible. After confirming functionality using the Intel Simics hardware simulator, we will then proceed to build and test the default memory profiling UEFI shell application for the UpXtreme board. After confirming the functionality of the default memory profiling UEFI shell application for the UpXtreme board, we will proceed to test the final version of the memory profiling UEFI shell application on the UpXtreme board.

Finally, we will build and test the default SMI handler profiler UEFI shell executable using the Intel Simics hardware simulator. After confirming functionality of the default shell executable, we will proceed to test the default SMI handler profiler UEFI shell executable for the UpXtreme board.

All testing will be performed sequentially and iteratively.

10. Testing Risks And Mitigation

Booting errors and other bugs - There is the risk that the changes we make to the firmware image that is loaded to the board may cause errors to occur in the boot sequence. This problem can be mitigated by keeping records of previous versions of our code and previous firmware images loaded before any modifications. We will preserve a copy of a functioning firmware image from the previous year's senior design group to use as backup in case any of our custom firmware images result in booting errors.

Scheduling and Deadlines - There are risks associated with the scheduling of this design project. Before starting work on the main deliverables, we must complete several training modules provided by Intel for simulating the boot environment using a Windows based application and a proprietary hardware simulator called Simics. The proprietary Simics simulator is used to build and simulate the boot environment and hardware of various Intel based chips. Since the training modules should be completed before working on the main deliverables, we must account for the training time allotted for our scheduling and deadlines. In order to mitigate risks associated with scheduling and deadlines, we have decided to make our final deliverable, the SMI handler profile feature shell application an optional deliverable to complete if time permits.

11. Approvals

Harry Hsiung and Laurie Jarlstrom will approve all test plans and project decisions.

References

1. "IEEE Standard for Software and System Test Documentation," in IEEE Std 829-2008 , vol., no., pp.1-150, 18 July 2008, doi: 10.1109/IEEESTD.2008.4578383.
2. Rothman, Michael, et al. *Harnessing the UEFI Shell*. De Gruyter, 2017.
3. *Unified Extensible Firmware Interface (UEFI) Specification*, Release 2.10. UEFI Forum, Inc., August 29, 2022
4. V. Zimmer, M. Rothman, and S. Marisetty, *Beyond bios: Developing with the Unified Extensible Firmware Interface*. De G Press, 2017.