Camera, Audio, Video UDL Development and Enablement in Pre-Silicon and Post-Silicon Environment

Major Project Report

Submitted in fulfillment of the requirements

for the degree of

Master of Technology

in

Electronics & Communication Engineering

(Embedded Systems)

By

Margi Soni

(16MECE23)

Electronics & Communication Engineering Department

Institute of Technology-Nirma University

Ahmedabad-382 481

May 2018

Camera, Audio, Video UDL Development and Enablement in Pre-Silicon and Post-Silicon Environment

Major Project Report

Submitted in fulfillment of the requirements

for the degree of

Master of Technology

in

Electronics & Communication Engineering

(Embedded Systems)

By

Margi Soni

(16MECE23)

Under the guidance of

External Project Guide:

Kishore Mottadi

Software Engineer Manager,CSS-BDV,

Intel Technology India Pvt. Ltd.,

Bangalore.

Internal Project Guide:

Dr. Tanish Zaveri

Professor, EC Department,

Institute of Technology,

Nirma University, Ahmedabad.

Electronics & Communication Engineering Department

Institute of Technology-Nirma University

Ahmedabad-382 481

May 2018

Declaration

This is to certify that

a. The thesis comprises of my original work towards the degree of Master of Technology in Embedded Systems at Nirma University and has not been submitted elsewhere for a degree.

b. Due acknowledgment has been made in the text to all other material used.

- Margi Soni

16MECE23

Disclaimer

"The content of this paper does not represent the technology, opinions, beliefs or positions of Intel Technology India Pvt. Ltd., its employees, vendors, customers or associates."

Certificate

This is to certify that the Major Project entitled **"Camera, Audio, Video UDL Development and Enablement in Pre-Silicon and Post-Silicon Environment"** submitted by **Margi Soni (16MECE23)**, towards the fulfillment of the requirements for the degree of Master of Technology in Embedded Systems, Nirma University, Ahmedabad is the record of work carried out by her under our supervision and guidance. In our opinion, the submitted work has reached a level required for being accepted for examination. The results embodied in this major project, to the best of our knowledge, haven't been submitted to any other university or institution for award of any degree or diploma.

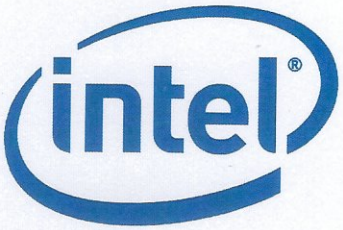
Date: Place: Ahmedabad

Dr.Tanish Zaveri
Internal Guide

Dr. N.P. Gajjar
Program Coordinator

Dr. D. K. Kothari
HOD, EC

Dr. Alka Mahajan
Director, IT



Certificate

This is to certify that the Major Project entitled **"Camera, Audio, Video UDL Development and Enablement in Pre-Silicon and Post-Silicon Environment"** submitted by **Margi Soni(16MECE23)**, towards the fulfillment of the requirements for the degree of Master of Technology in Embedded Systems, Nirma University, Ahmedabad is the record of work carried out by her under our supervision and guidance. In our opinion, the submitted work has reached a level required for being accepted for examination.

Kishore Mottadi
Software Engineer Manager,CSS-BDV,
Intel Technology India Pvt. Ltd.,
Bangalore.

Ms. Sneha Pingle
Software Engineer, CSS-BDV
Intel Technology India Pvt. Ltd.,
Bangalore.

Acknowledgements

I would like to express my gratitude and sincere thanks to **Dr. D. K. Kothari**, Head of Electronics and Communication Engineering Department, and **Dr. N. P. Gajjar**, PG Coordinator of M.Tech Embedded Systems program for allowing me to undertake this thesis work and for his guidelines during the review process.

I take this opportunity to express my profound gratitude and deep regards to **Dr. Tanish Zaveri**, guide of my major project for his exemplary guidance, monitoring and constant encouragement throughout the course of this thesis. The blessing, help and guidance given by her from time to time shall carry me a long way in the journey of life on which I am about to embark.

I would take this opportunity to express a deep sense of gratitude to **Kishore Mottadi**, Software Engineer Manager, Intel Technology India Pvt. Ltd. for his cordial support, constant supervision as well as for providing valuable information regarding the project and guidance, which helped me in completing this task through various stages. I would also like to thank **Ms. Sneha Pingle**, my Project Mentor for always helping, giving me good suggestions, solving my doubts and guide me to complete my project in better way.

I am obliged to **Mr. Jaimin Mody** my team member at Intel Technology India Pvt. Ltd. for the valuable information provided by him in respective fields. I am grateful for his cooperation during the period of my assignment.

- Margi Soni

16MECE23

Abstract

Intel is known for manufacturing of processors, so pre-processor and post-processor testing and validation is important task needed before putting its product in market. The processor has many functionalities and rigorous testing and validation of those functionality is required before the product goes into market.

Manual testing methods for validating of hundreds of features as well as functionalities for processor and reporting bugs is really time consuming and monotonous task which is prone to error. This can even delay the time to market and possibility of failure in market. In the recent years, Intel is focusing in atomization of these testing and verification methods to increase the robust and reliability of product in the market.

The aim of this project is to automate the test cases of camera, audio and video and validate the functionalities of Intel core processor platform. Automation to validate the features and functionalities and finding bugs for processor can reduce time to market product. Manual testing may not detect each and every bugs. So automation framework should be used to find all the major bugs. By using automation framework, good coverage of functionalities are possible which can identify each and every bugs of system under test.

In this project, camera, audio and video functionalities are validated and sub functionalities like bus, interrupt etc are also validated. One automation framework is developed by Intel for testing and validating of Intel core processor. So using that framework one single script is generated and using that script, features are validated.

Contents

List of Acronyms

UDL	User Defined Language
BIOS	Basic Input Output System
ROM	Read Only Memory
EEPROM	Electrically Erasable Programmable Read Only Memory
POST	Power on Self Test
SATA	Serial Advanced Technology Attachment
RAM	Random Access Memory
SOC	System On Chip
DMA	Direct Media Access
MSI	Message Signal Interrupt
UEFI	Unified Extensible Firmware Interface
ATMR	Automated Test Management Repository
EFT	Execution Framework Tool
ATK	Automatic Tool Kit
WMP	Windows Media Player
MIPI	Mobile Industry Processor Interface
CSI	Camera Serial Interface
I2C	Inter-Integrated Circuit

List of Figures

Chapter 1

Introduction

This chapter will explain the company and team profile, motivation of the thesis and possible solution and time line of the thesis.

1.1 Motivation

There are thousand of test cases available to validate on any system. And each test case has more than one step which is known has UDL. UDL is User Defined Language. To run a single test case on a system, it is a very prolonged task. If human efforts and time will be taken into consideration, it will be very tedious and time consuming. Also it is very difficult to find if there is any error occured. If those test cases will be automated then no need of human efforts.

So automation is key process for industry like Intel. Automation means avoiding manual task. Automation is not the big thing as that word indicates. By thinking of automation, first thought will come in the mind is like something related to robotics. But automation as simple as copying the file from one location to another location. So automation is tedious work initially but it will come with great future benefits. Automation work reduces manual efforts, increase efficiency and throughput.

It is good if the work is productive. Also the work must be beneficial for the team.

1.2 Company Profile

Intel Technologies is a multinational technology company established in America by two scientist, Gorden Moore and Robert Noyce in 1968. Its head quarter is in Santa Clara, in the Silicon Valley. Intel is one of the largest semiconductor chip makers company. First microprocessor was developed by Intel in 1971. Intel's x86 series processors are found in most of the PCs. Dell, Apple, HP, Lenovo are mainly customers to whom Intel supplies chipset, motherboards, memory etc. It has started with 4-bits microprocessors and right now it develops 64-bits microprocessors like Celeron, Xeon, Pentium, Atoms series.
Mainly six groups are there in Intel.

- Client Computing Group (CCG)
- Data Center Group (DCG)
- Internet of Things Group (IOTG)
- Software and Services Group (SSG)

1.2.1 Team Structure

Client Computing Group is the biggest group of Intel in terms of revenue generation(approx. 60%).CCG segment includes platforms designed for notebooks (including Ultrabook devices), 2 in 1 systems, desktops (including all-in-ones and personal computers (PCs)), tablets, phones, wireless and wired connectivity products, and mobile communication components. CCG team structure is shown in figure 1.1

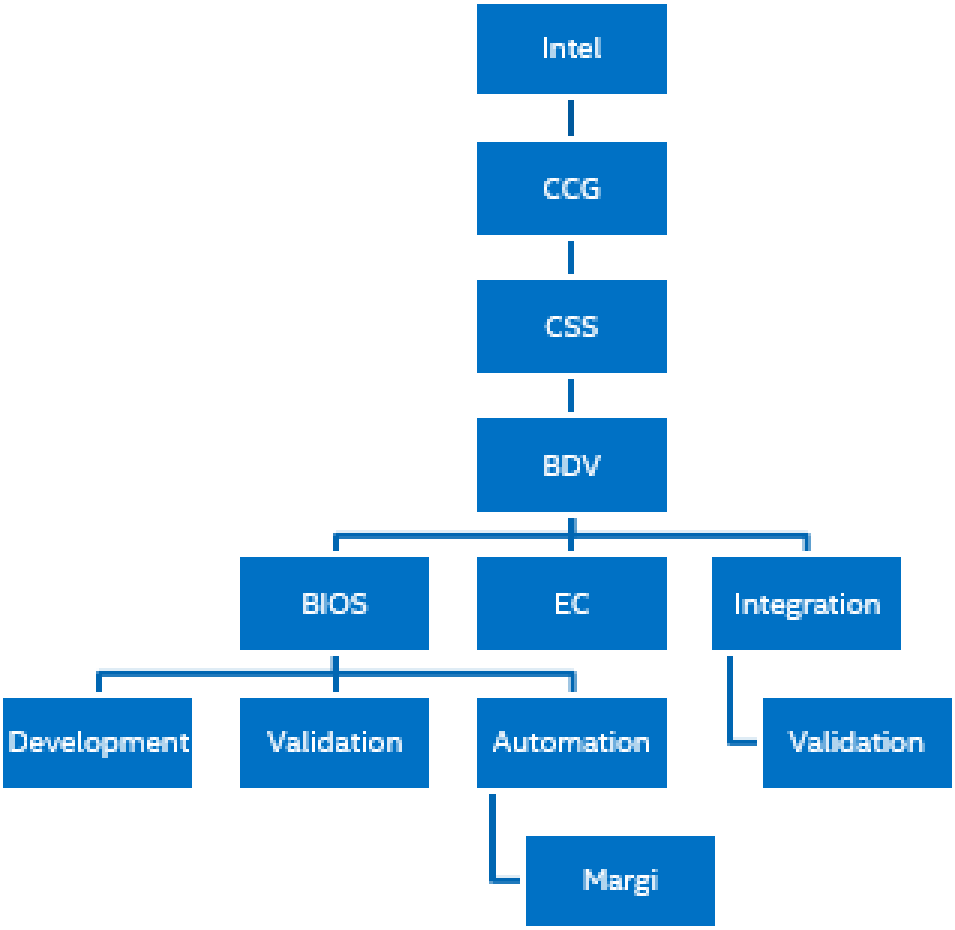


Figure 1.1: Team structure

1.3 Project Timeline

- During first month in Intel completed the mandatory training, did ramp up work on C programming and Python and also worked on one XML tool.
- During second quarter means during July to September developed different UEFI application at BIOS level.
- In October automation work was assigned. So understood automation framework, automation process and set up the board for automation.

- During November and December video part was covered.
- During January and February audio part was covered.
- During March and April camera part was covered.
- During May final thesis report was prepared.

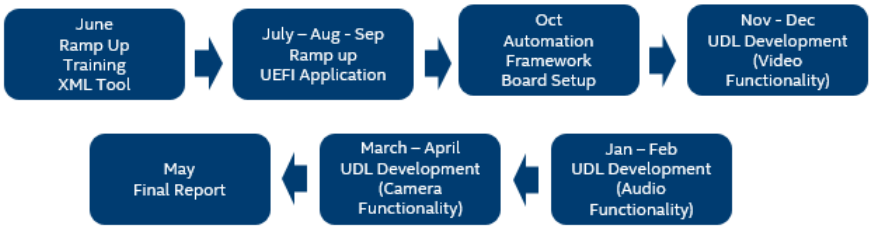


Figure 1.2: Time Line of Project

1.4 Thesis Organization

- Chapter 1 contains brief introduction company profile and project timeline.
- Chapter 2 describes Intel platform, legacy BIOS, platform initialization boot phases, boot flow as part of literature survey.
- Chapter 3 contains automation framework in detail and also describes the tools, which are used in automation.
- Chapter 4 contains hardware setup, process of UDL development and unit testing/
- Chapter 5 contains work done for fulfillment of this thesis.
- Chapter 6 describes conclusion and future scope for this thesis.
- At the end references are mentioned which are used for this thesis.

Chapter 2

Literature Survey

This chapter will give the overview of Intel's different platforms and architecture of Intel processor. It will describe SoC of Intel's chip. It will also describe the details of Pre-OS Firmware and the whole process from booting of pre-os firmware to loading the OS which is called BootFlow.

2.1 Intel Platform

Intel processors has been started from 4-bits and right now it is implemented on 64-bits. In that mainly two types of core are available. One is Big Core and another one is Small Core which is also known as Atom series processors. So basic difference in the platform is only the ingredient stuffing in the chipset. Legacy Intel processors are divided into North bridge and South bridge, which is called chipset. It is shown in figure 2.1. In recent platforms North bridge and South bridge are replaced with CPU and PCH. Recent Intel platforms have mainly two components:

- **CPU: Central Processing Unit**
  CPU has connection with memory controller and GPU.

- **PCH: Peripheral Control Hub**
  PCH have managibility engine, disk drives, PCI slots and ROM for pre-os

firmware.

Combination of CPU and PCH will decide the functionality of chipset and type of SoC.

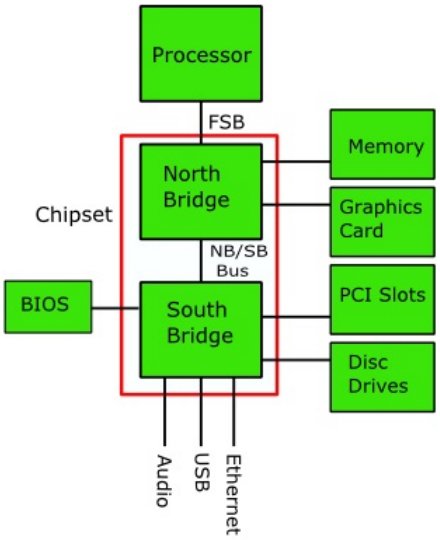


Figure 2.1: Chipset [8]

Different types of SoC Intel platform have

- Multichip Package: In this type of package CPU and PCH both are on same die. This kind of package makes one SoC. Multichip packages are used in low powered devices like ultrabooks.
- Two Chip Package: In this type of package CPU and PCH are two different chip. There are two different Socket on motherboard for CPU and PCH but single SoC for both. Two chip packages are used in desktops and high end loptops.
- Multi Socket Package: In multi Socket package more than one CPU inserted in it. In server machine this type of packages are used.

According to different die packages, there are different segments.

a. Ultra Thin Segment which is known as "U"

b. Ultra Light Segment which is known as "Y"

c. Halo Segment which is known as "H"

d. Desktop Segment which is known as "S"

Main hardware dependancy on these segments are CPU and PCH, motherboard, add on cards. Where as software dependancy are BIOS, device drivers, operating system and utility tools.

2.1.1 Intel Architecture: Skylake

Skylake is a 6th generation first 14nm micro architecture chip by Intel, which is replaced by Broadwell and Haswell. The Skylake family has main features are scalability, high performance, media graphics also noticeable changes compare to Haswell in terms of power management and power delivery. Skylake was introduced in all four segments "S", "U", "H", "Y".

Skylake has better instruction per clock using AVX2(Advanced Vector Extension) 512 bits. Skylake has TDP(Thermal Design Power) scaling is 20x. To improve memory performance, faster memory DDR4 and LPDDR3 is used. High resolution display upto 4K is used in Skylake which improves 3D gamming. For faster execution smarter branch predictor is used along with 3 level cache and better page miss handling can be possible. To improve over clocking, Intel separated PCIe clock and system clock. With enhancement of over clocking and liquid nitrogen cooling system upto 6.8 GHz frequency can be achieved. In this micro architecture there are two security technologies used. one is the Secure Guard Extension(SGX) and another one is Memory Protection Extension(MPX). [9]

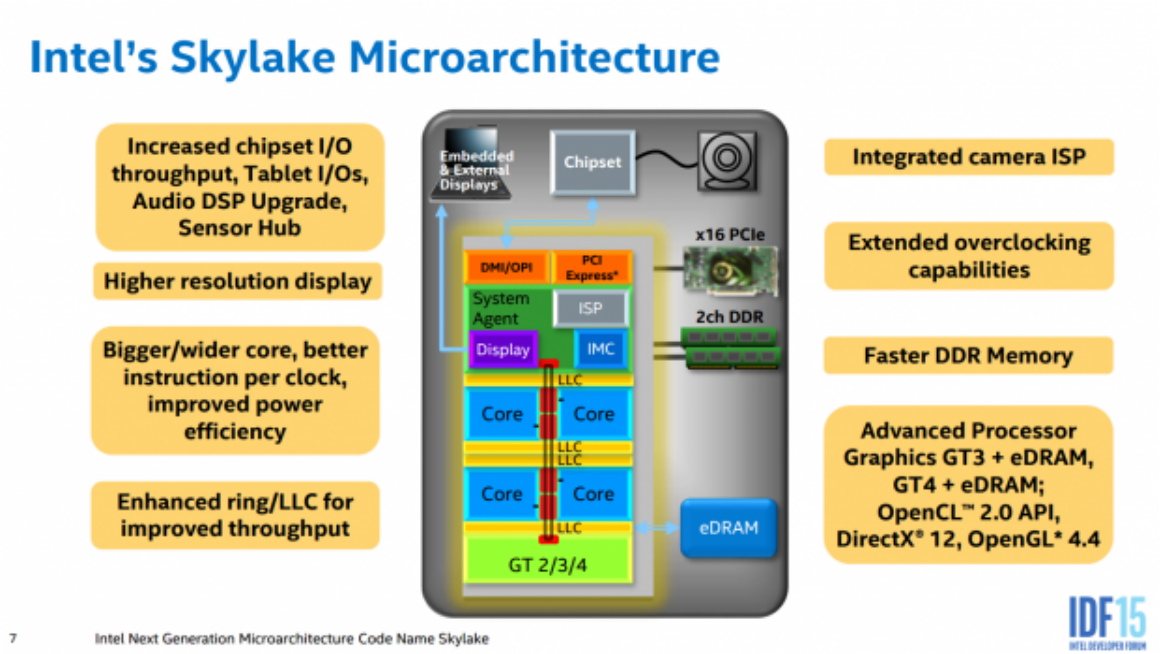


Figure 2.2: Skylake Micro Architecture [9]

2.2 Legacy BIOS: Pre-OS Firmware

2.2.1 Legacy BIOS

BIOS stands for Basic Input Output System. Pre-os Firmware which is work as bridge between hardware and operating system. Now a days BIOS is switched with other pre-os firmware. BIOS is the most important part of pre-os firmware where as other part's requirement is based on for some particular features only.

BIOS is stored in non volatile memory ROM. ROM used to keep BIOS in EEPROM. ROM is not in large size as RAM, so BIOS is in size of few MBs. After pressing the power button of any laptop or desktop, first BIOS is booted and it will hand over the control to operating system.

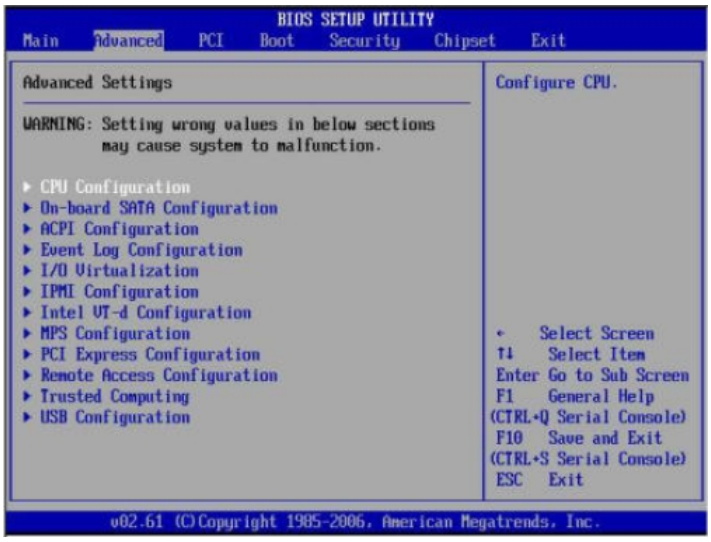


Figure 2.3: BIOS Configuration Screen [7]

Major tasks are done by BIOS are: [4]

- **Detecting Components:** BIOS is a software program which detects and enumerates components those are connected with computer. POST is performed by BIOS. BIOS checks configuration of external and internal components like keyboard, display, SATA, RAM, interrupt, DMA controller etc. It also checks those devices are working properly or not. If there is any failure occured it will show an error message or give a hex code and operating system will not be booted. Only after successful POST, computer will be able to fully boot up.

- **Booting Up:** After POST is completed, computer will boot. But before booting, BIOS has to check from where to boot. It will check all the memory devices which are connected to computer. There is a boot order to check the devices and if there is more than one boot order then BIOS will list out and first boot up with default one.

- **Gate Keeping:** After OS is loaded, still BIOS has task to do. It will proceed in background and provide support to operating system to handle major task

related to external peripheral. It will handle any MSI also which are generated by devices which are connected to computer. Take an example of hard drive which is connected with the computer and OS wants to some data from it. But OS doesn't know from where and how to get the data from hard drive. So CPU will raise a request and MSI will generate by hard drive and that MSI will handle by BIOS.

- **BIOS CMOS Setup:** BIOS is pre-configured and pre-installed. But the settings of BIOS can be changed by the button which is located on motherboard. It will help to keep the new or changed BIOS setting in CMOS RAM. If the RAM is cleared then BIOS setting will go to the default one. Changing CPU frequency, booting order, some security features which are basic setting that can be changed in BIOS.

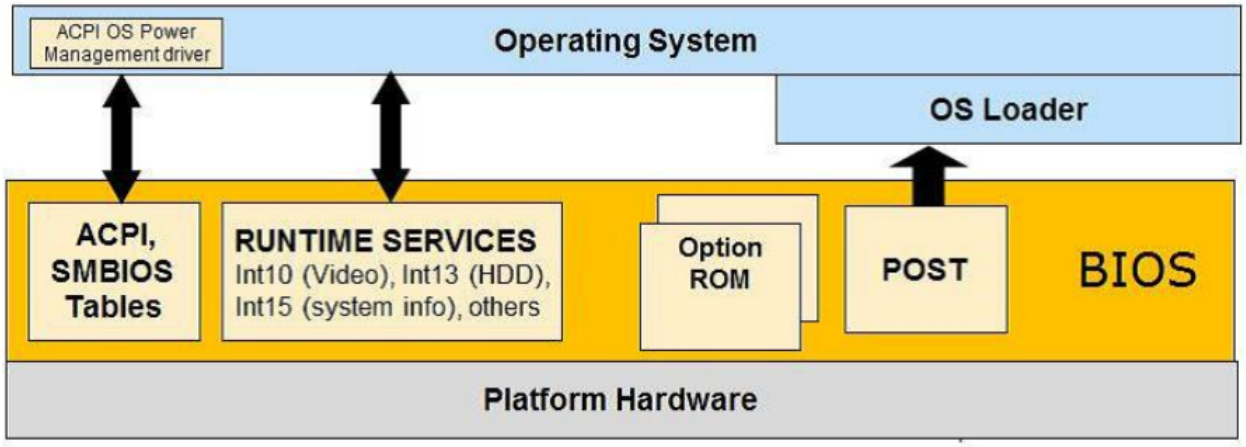


Figure 2.4: High Level Diagram of BIOS [4]

In computer just because of BIOS, communication can be possible between operating system and platform hardware. Only BIOS is the software who will know all the details about motherboard. POST will use the OS loader to load the operating

system, after completion of POST. ACPI and SMBIOS tables are used for power management and controlling the components which are connected to the computer. Run time services are the applications which are running after loading the operating system. Run time services will generate the interrupts for particular services and OS will return back acknowledgement after completion of services.

2.2.2 Pre-OS Firmware Components

- **BIOS Firmware:** POST operation, initialization of devices, boot loader operations are handled by BIOS firmware. BIOS is flashed on SPI-NOR chip which is connected with PCH.

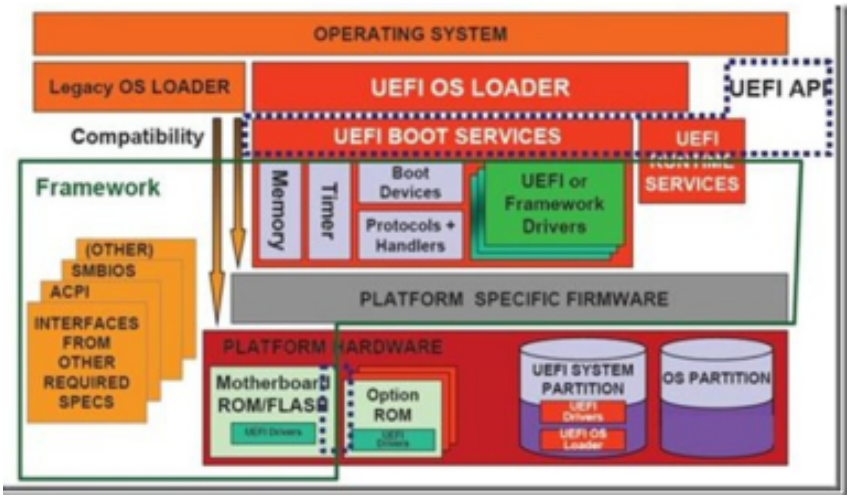


Figure 2.5: UEFI Architecture [4]

- **Processor Microcode:** High speed memory chip in the CPU contains micro code. Micro code is hard coded that means can't be altered after programmed.
- **PCH Patch:** All functionality of SoC can not be coded in silicon. So PCH patch is mostly written for PCH. PCH patch is some kind of micro code. Power management specific information is contained by PCH patch.

- **Option ROM:** Some devices like video card, RAM, SATA have their own ROM chip. Option ROM has great flexibility to enable the features.
- **Security Patch:** Security is required for protecting the system from virus as well as malfunctioning. Security will authenticate the system at BIOS level itself. It will check that any fraud device will not be connected with the system and ensure the correct path of devices. There are many straps available for security in CPU which can be enable or disable using BIOS.
- **Graphics Drivers:** GOP drivers are used for the graphics. Legacy video BIOS is replaced by GOP drivers which are written in UEFI. GOP drivers can be 64-bits or 32-bits.
- **Thunderbolt / Type C Support:** USB type C and Thunderbolt are enabled in UEFI. Thunderbolt has to support many functions at firmware level. So Thunderbolt must be enabled in pre-os.
- **ME Drivers:** ME drivers provide the features of power management, security, remote management.

2.2.3 BIOS vs UEFI

Legacy BIOS is replaced by UEFI. UEFI is a common standard developed by Intel for pre-os firmware. It is software interface between platform firmware and OS. UEFI allows user to run application on command line. Even if OS is not installed UEFI will support remote diagnostics repair the computer. UEFI has faster booting time. UEFI supports different boot flows like measured boot, secure boot, verified boot which will enable the feature to prevent malware at low level and upper stack also.

Legacy BIOS	UEFI BIOS
This is the traditional BIOS	New architecture based on EFI spec
Written in assembly code; initially designed for IBM PC-AT	C based; initially designed for Itanium server systems
Interface is per-BIOS "spaghetti" code, not modular	Well defined module environment and interface based on EFI specification
Lives within the first 1MB of system memory	Can live anywhere in the 4GB system memory space
Uses 16bit memory access, requires hacks to access above 1MB memory	Allows direct access of all memory via (32-bit and/or 64 bit) pointers
Supports 3rd party modules in the form of 16 bit Option ROMS	Supports 3rd party 32/64 bit drivers
No built in boot/test environment	Built-in boot/test via EFI - Shell
Only supports 16-bit runtime services such as INT10, INT13, etc	New runtime interfaces and supports legacy OSs and 16-bit legacy devices

Figure 2.6: Difference between Legacy BIOS and UEFI [4]

2.3 Platform Initialization(PI) Boot Phases

- **Security(SEC) Phase:** First phase of platform initialization is SEC. Temporary memory is created in this phase. It will secure the system and act as root of trust for the system. SEC will ensure about firmware integrity.
- **Pre-EFI Initialization(PEI) Phase:** It will initialize permanant memory. It will also initialize RAM and chipset. It will pass the control to driver execution environment phase.
- **Driver eXecution Environment (DXE) Phase:** Booting process is done during this stage. Devices are enumerated and initialized in this stage. Protocols and drivers are implemented during this stage.
- **Boot Device Selection (BDS) Phase:** This phase will decide how and where OS will be booted. Example Boot to OS or Boot to BIOS.

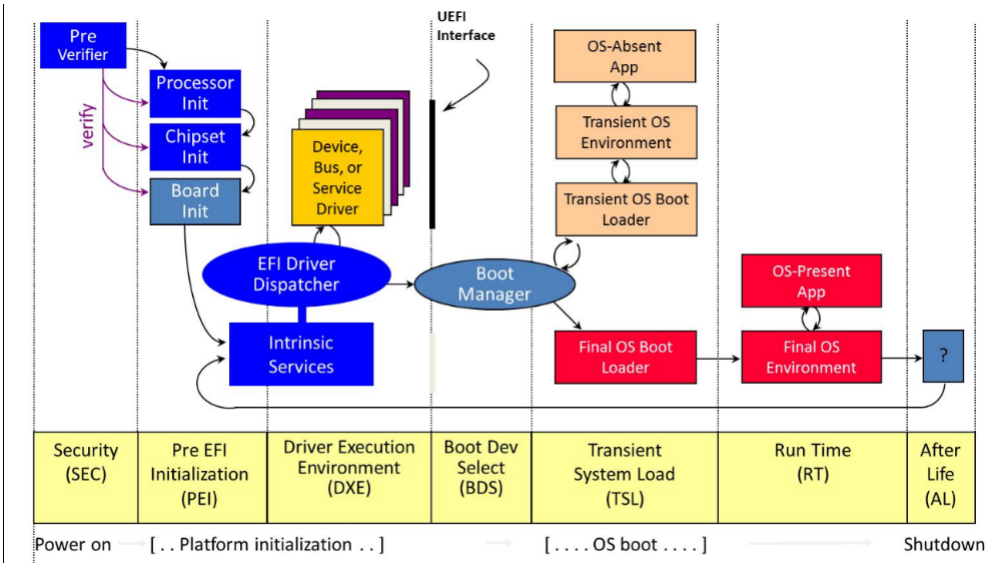


Figure 2.7: PI Boot Phases [6]

- **Transient System Load (TSL) Phase:** First stage of boot process where OS loader is in UEFI shell.
- **Run Time (RT) Phase and After Life (AF):** These two phases are come after system fully booted up to OS. IN RT and AF application can be run.

2.4 Boot Flow

- After pressing the power button of the system, power is taken from the battery or AC supply.
- First embedded controller will wake up
- Power management unit will trigger to PCH and SPI controller will wake up.
- SPI controller has SPI-NOR chip, which has ROM file of the firmware.
- Firmware image will initialize all the ingredients which are connected with PCH.

- Then CPU cores will powered on. CPU will wake up with 16-bit and that code is written in assembly language. After 32-bit and 64-bit core will come in the picture and at that time memory segment will created in the cache. That means cache will initialize.
- RAM will initialize.
- System will be fully booted.

Chapter 3

Automation Framework

This chapter will explain the requirement of framework, how it works and how it helps in validation or executing the test cases. Automation framework is well defined structure which is developed for validation and execution.

3.1 Introduction

An automation framework is well defined approach, which provides execution environment to execute the test scripts. It is specially structured to run the automation scripts. There are more than one developer work on automation and try to enable functionality on one single system, for them one common approach is required. If there is no common approach is available then developers will start developing their own approach. This method becomes very difficult in terms of communication and also sharing the result and knowledge to each other is very difficult. So use of automation framework, execution task becomes very easy. There are many advantages of framework like scalability, re-usability, cheaper in cost and also in maintenance. Thus automation framework makes task easy.

3.2 Requirement of Framework

Importance of automation framework is pretty much clear in above section. This section will explain greater advantages of automation framework.

- Re-Usability of Code: There is one library which stores the code (scripts). If new code will be written for existing feature then that previous code is re-usable from library.
- Maximum Coverage: Centralized tracking system is there. So it is to maintain the records of automation and this records help to get maximum coverage.
- Recovery Scenario: Any crash on the SUT or Host can damage the test cases. But the test cases will be regenerated from the library of framework. Automation framework works as database, there is no lost of data possible.
- Low Cost Maintenance: Because of centralized framework, result and knowledge sharing is easy. Easy controlling and monitoring of validation process. With individual approach collaboration process is difficult and also costly in teams of time.
- Minimal Manual Intervention: End to end solution is possible in automation framework. So manual efforts are not required and that is why manual error can be possible to reduce.
- Easy Reporting: Result and logs of test case can be maintain at same place. So it helps in fixing the bugs and issues.

One step of test case may be used in other test case. Then no need to write the same script again and again. It is reusable and it is maintain by automation framework. Thus in framework just needs to give correct input, rest of the work is handled by framework. Correct inputs are like SUT and Host ID, test case parameters which are passed through functions written in the scripts, location to store the result, etc.

3.3 Hybrid Automation Framework

There are many types of automation frameworks are available. Here one of the well known framework **Hybrid Automation Framework** is explained. In validation there are many types of testing can be done. Number of functionality and features are to be tested on any platform in proper way. Number of BIOS functionality must be validated and for that too many test cases are available. But not all test cases are unique, many are repeated steps. For example..
6th generation onward Intel's platforms have many power options like shutdown, hybernate, sleep, connected modern standby, etc.. These power options come under one common functionality power down state, which is tested on system under test. So major steps of these testing would be common. Possible steps are following:

- Boot to OS
- Check whether system is up or not
- Put the system into one of the power state
- Reboot the system up to OS
- Again check system is up or not
- Results

Above test case steps are mostly common for all test cases regarding power down state. So instead of writing same script again and again for all power options, scripts can be split in the functions. And functions can be reusable up to great extend. One centralized library can be created by redundant functions. And the libraries are accessible across all over the framework.
Automation is not possible by using single script or tool. There can be multiple way of performing validation on particular platform. Some test case may be data dependent. With the help of data driven test automation, the existing automation

framework can be more flexible. By selecting a correct data input, libraries and other required utilities, package can be generated. Input should be in proper way that framework can understand. For passing the parameters to framework, proper syntax is defined. So parameters must be match with the parameter syntax to avoid miss match error. If there is any change required in the function then need to change only at one place in the library. This is a main advantage of the framework.

3.4 Automation Team Process Flow

.

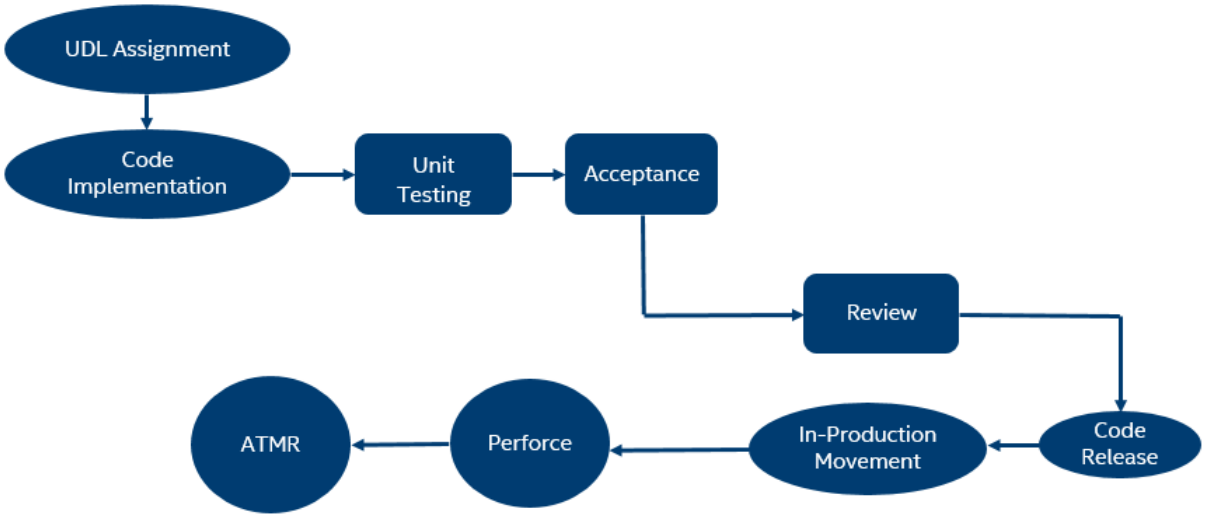


Figure 3.1: Automation Framework

The following steps shows the whole process of automation framework.

- **UDL Assignment:** Step of test cases are known as UDL or grammar name. Here UDL means one step of test case. Those UDLs are first assigned to the developers. According to priority UDLs are distributed.

- **Code Implementation:** Developers develop the scripts. They develop proper libraries, code gen, dictionaries etc
- **Unit Testing:** After development of code, developers only do unit testing. Unit test includes code level checking across different platforms and for all parameters. And result and logs of unit test is shared for future reference in shared folder.
- After unit testing, peer review is happened. Peer review is done by another developer at coding level like code optimization, error reduction.
- **Acceptance:** Acceptance is based on UDL and Test case. Need to share all the information like H/w, S/W, tools to execution people. Acceptance is done carefully, because it should not affect to the test case expectation.
- **Review:** L-1 review is done by team lead. Code optimization is taken care by team lead. Overall syntax error and prep-8 standard everything will be reviewed.
- **Code release and In-Production Movement:** Perforce is database for scripts. After completion of review code must be released in Perforce as a centralize database so that everyone can access it.

3.5 Introduction of Tools

3.5.1 JAMA: Product Management Platform

JAMA Product Management Platform is a web based application. JAMA is basically defined for project management. JAMA is used as test case repository. Test case id, H/W and S/W details, test case steps, platform details, tools etc.. Complete information regarding test cases can be found in JAMA.
JAMA successfully manages the test management process in the form of complete

detail about that test case. JAMA is mainly use by manual validation team. But in automation, during execution of test cases JAMA comes in the picture. During execution ATMR fetches all information about test case.

3.5.2 ATMR: Automated Test Management Repository

ATMR stands for Automated Test Management Repository. ATMR is the database where all the software validations information is stored in executable copy.
ATMR tool in which whole automation task can be possible in fully automated. Just need to give the proper information and select correct option for the test case. And in one click package will be generated and as well result also generated.

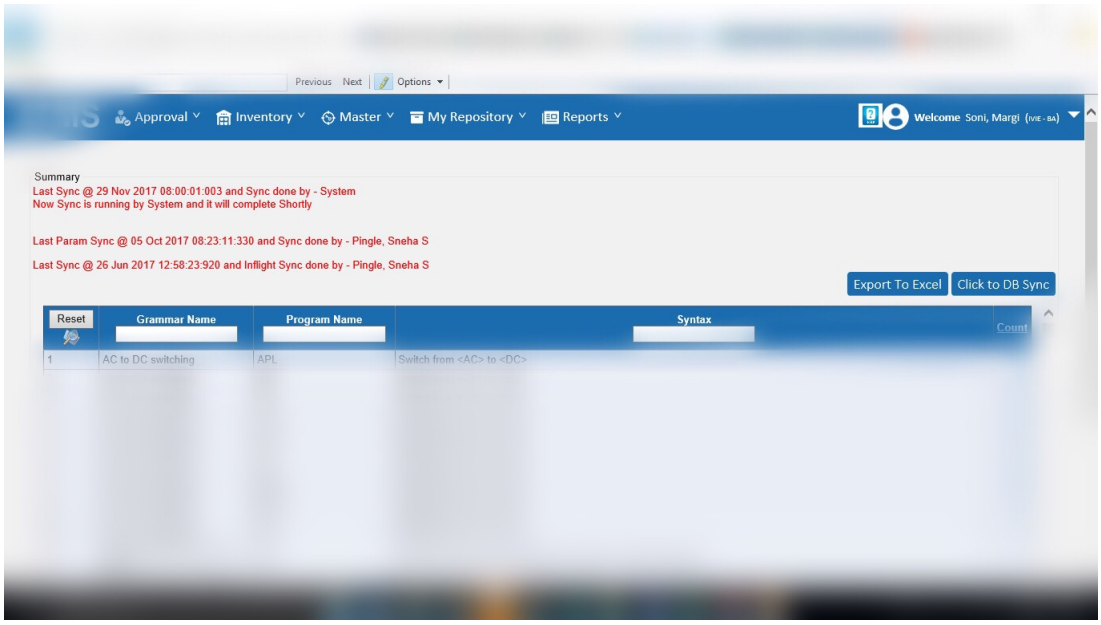


Figure 3.2: ATMR UI

3.5.3 Perforce: Code Integration and Version Control

There are many aspect of automation framework like code recovery, code integration, revision control etc. To maintain these aspects Perforce is used. Perforce act as a permanent database. If there is any change required in the code then that code should be changed in local and then should be uploaded on Perforce. So in Perforce at any instance latest version of scripts can be available.
This system offers the hybrid of merge and lock concurrency models. Perforce is a central location to upload the files. So whoever has access of Perforce they can access all the files. By mistake if any one overwrite the code, backup will be available. The server keeps the files in master repository and that contents are not accessible for all users except the superuser.

3.5.4 EFT: Execution Framework Tool

EFT stands for Execution Framework Tool. EFT is an execution framework. EFT is a collection of web based services which executing test scenarios. SUT and Host management can be done by EFT. In this framework every application does only one thing and does it well. It supports Windows, Linux, MAC OS and Android.

Some scripts should be run in SUT and some scripts should be run HOST. Connection between SUT and HOST is made by EFT execution framework. One JSON file should be uploaded to run the scripts through EFT. JSON file will indicate the flow of scripts. And at the end result will be generated in EFT.

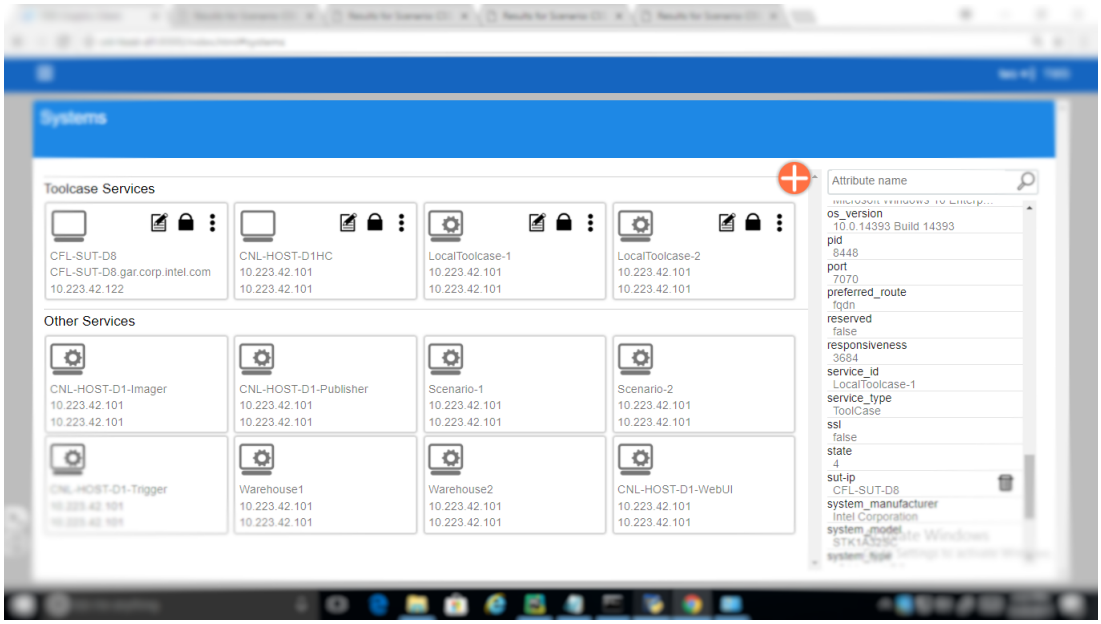


Figure 3.3: EFT

Chapter 4

UDL Development

This chapter will explain end to end process of UDL development. In this chapter get to know which technologies are used in the UDLs, different stages of UDL, etc.

4.1 Hardware Setup

To run the scripts on any platform, board setup is main aspect. There are many components are used, which are described below:

- **SUT:** SUT stands for System Under Test. The validation testing is done on SUT. SUT contains silicon and PCH. Main validation testings are related to BIOS, different functionality of silicon and PCH. During testing EFT should be run in SUT.

- **Host:** Host works as server. Host and SUT are connected by LAN connection. Sometimes it is necessary to run the scripts through HOST. EFT must be run in HOST. Example
  If SUT is in the S3(sleep) state and that time need to run the scripts. In this example scripts should be run in Host.

- **ATK:** ATK stands for Automatic Tool Kit. ATK provides solution to execute and debug the functionality remotely. It has a single UI which works across multiple hardware solutions. ATK remote capabilities include: power cycling control, front panel power On/Off control RESET, real time postcode, voltages LEDs Monitoring, Jumper Settings (GPIO)

- **Relay:** Relay is connected with SUT. Relay is used to power and reset button which are located on the SUT. Relay is reworked with SUT for power button, reset button, volume up-down button. Relay is handled by ATK software. High and Low signals can be sent through relay.

- **Brain Box:** Brain box is an external keyboard. If any input is required from user then brain box is used.

4.2 Process of UDL Development

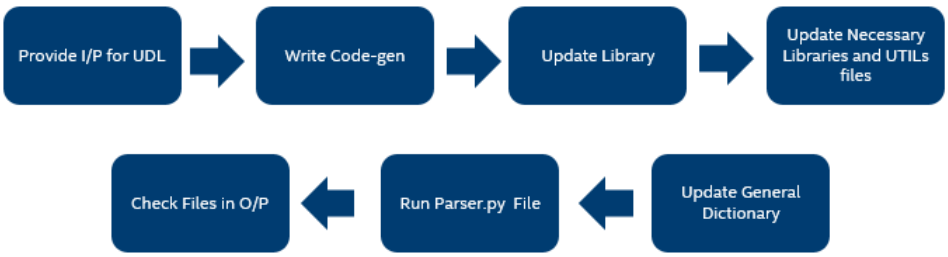


Figure 4.1: UDL Development Process

- Provide UDL name in input.txt file

- Write code-gen for the UDL which will generate the scripts. Code-gen will generate the scripts for SUT and Host. Code-gen also contains calling part functions.

- Create the library. One UDL has only one library and one code-gen. Library contains functions.
- Update necessary libraries and UTILs files. If there are any tool is going to be used then it should be mention in the proper library.
- One general dictionary is there which contains regex. Regex will find the one particular code-gen for the UDL.
- After completion of all above steps, one Parser.py file is there. Parser.py will generate the scripts for SUT and Host from the code-gen.
- Generated scripts are stored in the output folder.

4.3 Development of UDL and Unit Testing

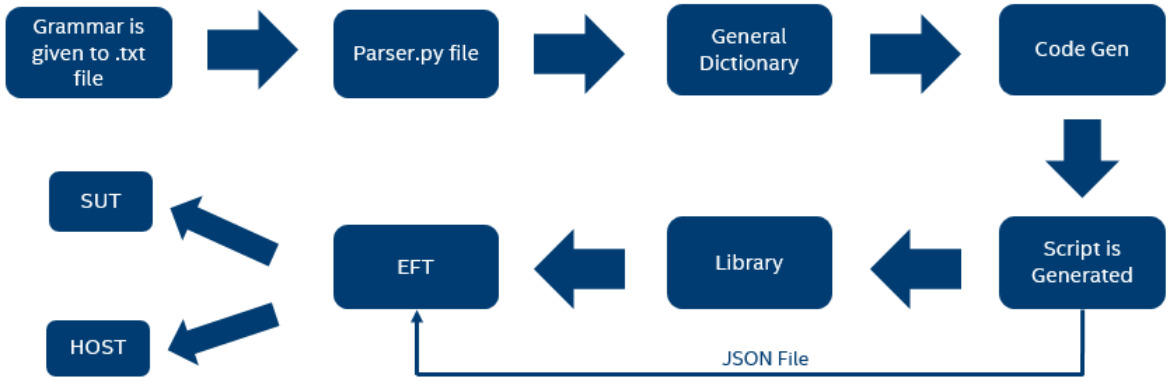


Figure 4.2: Development of UDL and Unit Testing

UDL name should be given to .txt or .xml file. Praser.py file should be run. First it will check general dictionary which contains Regex. Regex should match to code gen. Parser.py file will generate scripts using code gen. Generated scripts have

calling function. Those functions should be in library. One JSON creator file will be there, which generate JSON file. JSON file should be updated in EFT. EFT will run the scripts in SUT and Host. At the end of this process EFT will generate one result file.

This procedure should be done by developer. This is called development of UDL and unit testing.

Chapter 5

Developed UDL

Assigned UDLs are related to Camera, Audio and Video. Here in this chapter UDLs are explained in detail like how it is developed, which technology is used or which scripting language is used etc.

5.1 Scripting Language

Python is used as scripting language. Python is widely used as interpreted, high-level, general purpose language. Python is an open source programming language, which is compatible with most of the operating systems. Python contains very easy and effective approach of OOP.

Python has more English key words, so it makes easy to understand. Arrangement of words is called syntax. Every language has own syntax and as a syntax python has indentation. Python has many flow control statements like if else, for, while, etc in that try and exception for better handling of statements.

The biggest advantage of python is its vast library collection. It is an open source programming language, so if there is no existing file available then one can create library and add up in the stock.

Few of the Python features are:

- **Readability:** Python has English key words. So python syntax are easily

readable and easy to understand.

- **Quick to Implement and Learn:** Python code is most of the time same as pseudo code. Many modules are inbuilt as standard Python library functions.
- **Portability:** Python provides same interface for different platform. So Python language is compatible with most of the operating systems like Windows, Linux, MAC OS, Android, etc.

5.2 Developed UDL

5.2.1 UDL: initialize ONLINEVIDEO and perform Sx

Initialize online video and perform Sx UDL comes under the functionality of interrupts, Bus and video. By perform Sx cycles interrupts are given to the system and during this systems how system going to be worked that need to be check. And after interrupting the system how video functionality will response that need to be check. So main purpose of this UDL is to verify functionality of Bus, Video and Interrupts.
Here ONLINEVIDEO and Sx are parameter1 and parameter2 respectively. To initialize online video Selenium package of python is used and for other functionality, Python language is used. To perform Sx power states ATK tool is used, which reads the postcodes and perform power states act accordingly.

Sx States

Sx states are different states for system. They are also known as system states. X might be 3, 4 or 5.

- **S3(Sleep):** All the process will be stored in the volatile memory(RAM) and during particular period of time it will be refreshed so that it will be maintained the sleep state. When the input is given from external devices like LAN, USB,

Keyboard etc. computer can wake up from the sleep states because of these external devices are remain powered.

- **S4(Hibernate):** System will act as it is off. Power consumption is at lowest level in this state. Content of all the processes will be stored in non-volatile memory(hard drive). After waking up or coming in the normal state, system will take more time than S3(sleep state).
- **S5(Shutdown):** In this state system has no power. It acts to be off.

To implement these system states, need to do setting in control panel. After restarting the system, system will act accordingly. ATK is connected with SUT which will read the post codes and tell about the correct system states. To automate this process python scripting language as well ATK(Automation Tool Kit) is used.

Technology Used

To initialize online video, Selenium package of python is used. Using Selenium package website going to be opened and video will be launched. Python script will maximize the screen on which video is playing. Also it verify rather video is playing or not. Result of this UDL is written in log file by using Python script only.

Function Wrapper: To verify video is continuously playing or not, there is YouTube API is published by Google. In this document different states of YouTube are mentioned. According to this document video is continuously playing or not that is going to check.
"player.getPlayerState()" is a function for JAVA language. But by using function wrapper of Python it is used in this UDL to know video's state. It will return particular value according to video's state.
Possible values are [9]
-1 – unstarted
0 – ended

1 – playing

2 – paused

3 – buffering

5 – video cued

Selenium: Selenium is an open source testing framework for web applications. Selenium provides wrappers for different languages like Java, Perl, PHP, Python, Rubby.

Architecture of Selenium consist mainly three parts:

1) Language Level Binding: Selenium has set of functionality and commands for different languages like Java, Python, PHP etc. These languages can interact with the selenium web driver and perform various action on different browsers.

2) Selenium Web Driver API: Selenium web driver API sends the commands and those commands are interpreted by web drivers.

3) Drivers: Drivers provide different web browsers like Firefox, Google chrome, Internet Explorer, etc. These drivers has functionality to drive the web browsers.

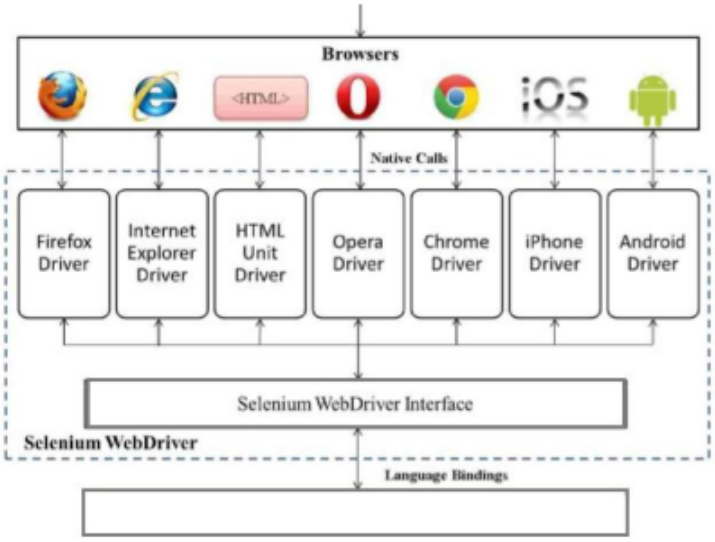


Figure 5.1: Selenium Web Driver Architecture [4]

Flow of Developed UDL

Figure 5.2 shows the flow of UDL.

- First script will set the option in control panel that after pressing the power button, system should be perform S3(sleep), S4(Hibernate) or S5(Shutdown).
- Second script will restart the system to save the changes.
- Third script will open the google chrome, initialize the video, maximize the screen and check whether video is playing or not.
- Forth script will run in the back ground to read the post code using ATK tool.
- Fifth script will press the power button using relay. Power button will press when proper post code will be read by ATK. ATK will send the High signal through relay and power button will be pressed and power state will be performed.

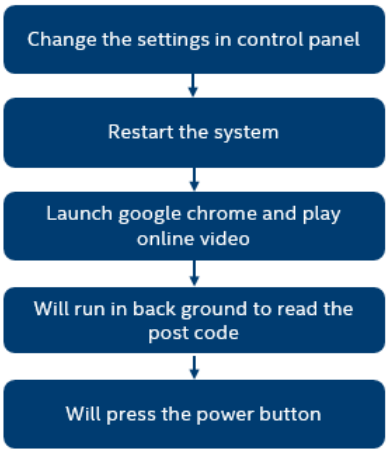


Figure 5.2: Flow of UDL: initialize ONLINEVIDEO and perform S3

NOTE: Here only "initialize ONLINEVIDEO and perform S3" UDL is explained. But for development of other UDLs with different parameters need to change some logic. For an example..

"UDL: initialize VIDEO and perform S3"
For this UDL need to give a path for video where it is stored and for verification video, it is explained in later this chapter only. For perform S4 instead of S3 need to check different post code.

5.2.2 UDL: play VIDEO and verify path SDCARD and app = WMP

This UDL comes under the category of play audio/video files and verify the functionality. That means need to verify the audio/video functionality. It also verify the Bus because in this UDL need to verify path also. That means file must be read from given path only. And here given path is sdcard.

UDL name is **play <param1>and verify path <param2>and app = <param3>**
Parameter 1 can be AUDIO/VIDEO
Parameter 2 can be SDCARD/Intranet Path
Parameter 3 can be WMP/VLC/Groove Music
Here VIDEO, SDCARD and WMP are parameter1, parameter2 and parameter3 respectively. To check availability of sdcard, open WMP and launch the video in WMP app, python scripting language is used.

First need to check sdcard is connected with the system or not. It is checked into device manager. If device manager is showed availability of sdcard then need to check is there any yellow bang along with sdcard. If there is yellow bang that means driver is not installed correctly or may be any issue with hardware. So need to fix

it for further process. This whole process is done by python script and one Intel internal tool is used to check device manager. That tool is also written in python scripting language. By using that tool one log file is going to be generated which contains detail of all the devices which are connected with the system.

If sdcard is properly connected with the system then next step is to be launched the Windows Media Player. WMP.exe file need to be executed so that WMP application going to be launched. WMP application is properly launched or not that has to be checked. To check WMP is working properly or not, need to check in task manager. Need to check .exe file of WMP is running in the task manager or not. If WMP.exe file is existing in task manager that means WMP is running properly. This also done by python scripting language.

If sdcard is detected correctly and WMP.exe file is running task manager then next step is to launch the video in WMP. For that need to give particular path from where video needs to be played. Here need to give path for sdcard.

If video is launched, then need to check video is continuously playing or not. To verify video, need to take screen shots of the screen at particular time period and compare it with the sub sequence screen shots. To capture the screen shots python scripting language is used and to compare the image opencv module of python is used.

To verify video is playing or not, first need to take screen shots. Then each screen shots are compared with it's sub sequence screen shots. But for comparing first need to covert screen shots in to black and white image. And then comparison can be done. Result of this comparison will be a matrix. So all the component of matrix is "0" that means all the images are same and video is not playing. Vice versa if any matrix is not zero that means video is playing. For this opencv module is used in

Python scripting language.

Flow of UDL

Figure 5.3 shows the flow of this UDL.

- The system is going to be restarted.
- Below tasks are going to be done in one script but each task have separate function.
  - Check device manager for detection of sdcard.
  - Launch Windows Media Player.
  - Initialize video.
  - Verify whether video is continuously playing or not.
- Result is going to be stored in the log file.

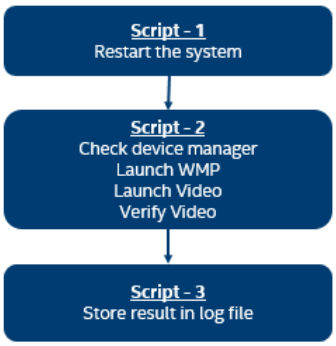


Figure 5.3: Flow of UDL: play VIDEO and verify path SDCARD and app = WMP

NOTE: Here "play VIDEO and verify path SDCARD and app = WMP" UDL is only explained. But for development of other UDLs with different parameters need to change the path only. let's take an example..

"UDL: play VIDEO and verify path INTRANET and app = VLC"
For this UDL no need to check the connection of SDCARD with the system only need to change the location and here VLC app is going to be used. So instead of WMP.exe file, VLC.exe file is going to be launched.

For audio there is different logic from video. It is explained in the later section of this chapter only. For audio there is different logic from video.

5.2.3 UDL: play AUDIO and verify path INTRANET and app = GROOVEMUSIC

This UDL comes under the functionality of audio and bus. Here need to check audio file is playing without any nose or any without any corruption. Also need to check internet connection like wifi module or LAN connection as audio file is playing from intranet. Here intranet is a shared location for Intel where necessary files are stored, so that everyone can access it or someone can use it later also.

UDL name is **"initialize <param1>and verify path <param2>and app <param3>"**
This UDL is sub part of above UDL. In this UDL, only parameters are changed. In this UDL the process is explained for verification audio. Here AUDIO, INTRANET and GROOVEMUSIC are parameter1, parameter2 and parameter3 respectively.

For this UDL first need to check internet connection. For internet LAN cable is used. In device manager, internet connectivity is going to be checked. To check

device manager, Intel internal tool is used. For this tool need to run .exe file so that it will generate one log file which will contain detail of the devices which are connected with the system.

If network connection is good then need to launch groove music application. And then need to check in task manager that groove music.exe file is running or not. Groove music application is launched properly or not that must be checked. If groove music.exe file is running in task manager that means this app working properly.

If groove music app is launched then initialize audio file from shared location. And need to verify audio is continuously playing without any noise or not. To verify audio file, need to check it's amplitude and generate a log file. And this amplitude is going to be compared with the amplitude of the original audio file which is generated previously. For verification of audio file, one threshold value is decided. If amplitude is crossed that threshold value that means noise is present and audio functionality can't be verified.
This whole process is done by python scripting language and pyaudio module is used for getting the amplitude of audio file.

Flow of UDL

Figure 5.4 shows the flow of UDL.

- The system is going to be restarted.
- Below tasks are done in one single script but they have different methods.
  - Check internet connection
  - Launch Groove Music Application
  - Initialize audio

  - Verify audio file that it is playing without noise or not

- Result is going to be stored in the log file.

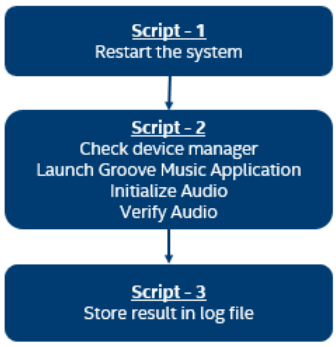


Figure 5.4: Flow of UDL: play AUDIO and verify path INTRANET and app = GROOVEMUSIC

5.2.4 UDL: capture PHOTO using FRONTCAMERA

This UDL comes under the functionality of camera and bus. In this UDL need to check different functionality of camera and also camera is an external device which is connected through bus. So need to verify bus functionality also.

UDL name is **capture <param1>using <param2>**
Here parameter1 can be PHOTO/VIDEO
And parameter2 can be FRONTCAMERA/REARCAMERA
Here PHOTO and FRONTCAMERA are parameter1 and parameter2 respectively. To check bus and camera functionality here also python scripting language is used. First need to check availability of camera in device manager. If camera is properly connected with the system that it will be detected in device manager. Then need

to check if any yellow bang is there or not. If yellow bang is there it means there is some problem with driver or there might be hardware problem. So first need to solve it.

If camera is correctly detected in device manager then one Intel internal tool is used to operate the camera. By using that tool camera can be switched on and shutter button can be pressed. This whole process is handle through python scripting language.

To capture the photo shutter button must be pressed through the internal tool. And after pressing the shutter button photo must be captured and it must be saved for later use.

After capturing the image need to be checked whether it is bad image or good image. Sometimes after capturing the photo it might be possible that image is going to be corrupted so black image is going to be stored. So this need to be checked. And this is checked by subtraction of image from the black image. After subtraction of image, one matrix will be generated of the size of pixels. If all the component of matrix is zero that means image is corrupted.

By using Intel internal tool camera is going to be switched off.

Hardware Description

16 MP MIPI camera module is used to capture the photo or video. MIPI is stands for Mobile Industry Processor Interface. Actually MIPI alliance is an organization which is mainly developed interface specific for the mobile eco system.
Camera module is connected with host device through CSI-2 interface. CSI-2 interface is developed by MIPI alliance. CSI stands for Camera Serial Interface. For point-to-point image transmission and video transmission between camera module

and host device, a widely adopted, simple, high-speed protocol primarily intended.

- MIPI camera modules are widely used in mobile devices and automotive that is the main advantage of MIPI camera module.
- Fundamental features of MIPI camera modules are high performance, low power etc..
- Main use cases are imaging, vision, contextual awareness, biometric recognition, surveillance

Figure 5.5 shows interface between MIPI camera module and host device.

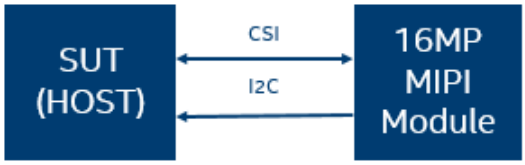


Figure 5.5: MIPI Camera Module Interface with Host

CSI stands for Camera Serial Interface. CSI is bidirectional control lines. By CSI controls are transferred from host to MIPI camera module as well MIPI camera module to host. And by I2C data is transferred from MIPI camera module to host device.

There are two physical layers are implemented for the transmission. MIPI C-PHY and MIPI D-PHY. For upto 24gbps data transferred three lane(nine wire) C-PHY interface is used. And for upto 18gbps data transferred four lane(ten wire) interface is used. [10]

Figure 5.6 shows actual MIPI camera module interface with the SUT. And Figure 5.7 shows MIPI camera module.

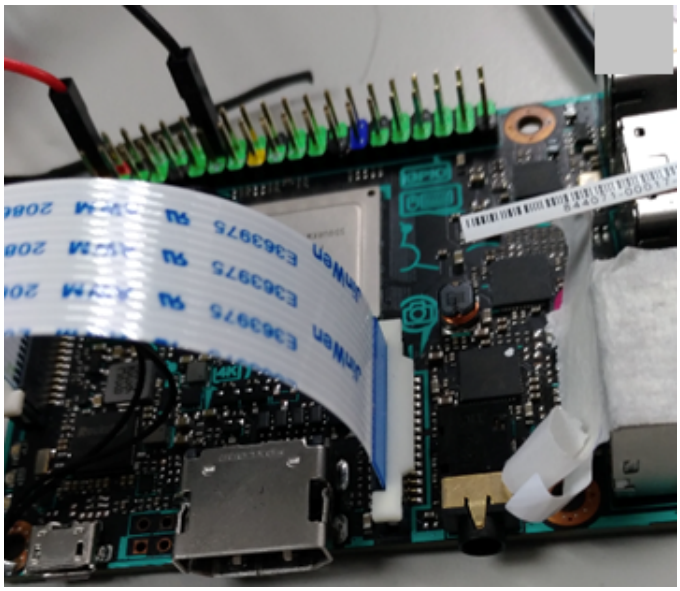


Figure 5.6: MIPI Camera Module on SUT

MIPI Camera Specification:

- Type: CMOS Camera Module
- Supply voltage: 2.6 to 3.0 v
- Image sensor: CMOS
- Pixels: 16mp

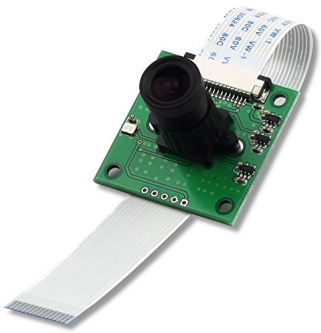


Figure 5.7: MIPI Camera Module

Flow of UDL

- Restart the system.
- Below tasks are going to be done in one script but each task have separate function.
  - Check device manager for detection of camera module
  - Switch on the camera using Intel internal tool
  - Press shutter button
  - Save the image
  - Check whether saved image is bad image or not
  - Switch of the camera
- Result will be stored in the log file.

Figure 5.7 shows flow of UDL.

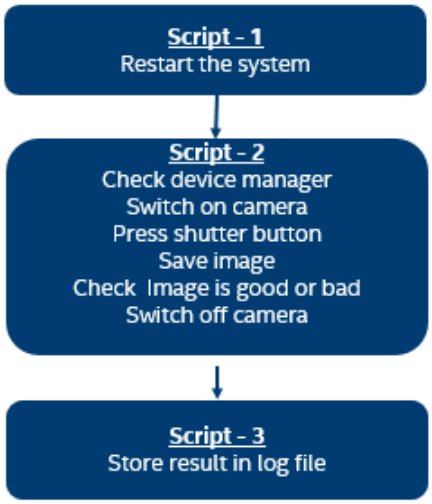


Figure 5.8: Flow of UDL: capture PHOTO using FRONTCAMERA

NOTE: Here "capture PHOTO using FRONTCAMERA" is only explained. But for development of other UDLs with different parameters need to change the some settings in Intel internal tool.

"UDL: capture PHOTO using REARCAMERA"

For this UDL only switch on the rear camera instead of front camera and that setting can be done through Intel internal tool.

Chapter 6

Conclusion and Future Scope

6.1 Conclusion

In this thesis, legacy BIOS and UEFI explored in detail. Also flashed BIOS on the Intel core processor using different methods. Developed different UEFI applications at BIOS level.

In manual testing there are different tasks involved. For example.. system settings, file editing, giving commands as an input, generate result in log file etc. So this is very lengthy process and human efforts are required. But in this thesis these type of all the task are done by one single script and validate the functionalities of camera, audio and video. Automation framework is efficient in all terms. Automation framework can reduce manual intervention up to great extend. And also reduce manual errors. Automation is cheaper in maintenance cost as well in time.

Using different technologies and hardware, almost 11 UDLs are developed related to camera, audio and video and automate the 150 around test cases for different platforms. And by this almost 80% coverage can be achieved of camera, audio and video functionalities.

6.2 Future Scope

In this project Camera, Audio and Video functionality must be enabled and tested by automation. For that there are too many UDLs and test cases. Those all test cases must be automated using this automation framework.

In next step Camera functionality for video is remaining and many other functionality for camera, audio and video like clarity of photo/video, resolution of photo, etc are not covered. So for that may be different UDLs are possible.

Bibliography

[1] Learning Python, Mark Lutz, Programming Language, 5th Edition

[2] Intel internal documents

[3] Software Testing Studio, 'Selenium', 2016 [Online].
Available: http://www.softwaretestingstudio.com/selenium-webdriver/ [Accessed: December2,2017]

[4] Khursid.Usman, 'Difference between Legacy BIOS and UEFI', 2013, [Online].
Available: https://www.maketecheasier.com/differences-between-uefi-and-bios/ [Accessed:December1,2017]

[5] Brian.Richardson,'Platform Initialization Phases', 2017, [Online].
Available: https://github.com/tianocore/tianocore.github.io/wiki/PI-Boot-Flow [Accessed: December1,2017]

[6] Intel Pvt.Ltd, 'Advanced BIOS Option', 2013, [Online].
Available: https://docs.oracle.com/cd/E19269-01/820-5830-13/app$_b$$ios.html[Accessed : December2, 2017]

[7] 'Chipset', [Online].
Available: http://bucarotechelp.com/computers/architecture/86081601.asp [Accessed: December2,2017]

[8] Steven.Bassiri, 'Skylake Architecture', 2015, [Online].
Available: https://www.tweaktown.com/articles/7309/intel-skylake-microarchitecture-high-level-info-idf-2015/index.html [Accessed: December2,2017]

[9] Google, 'YouTube API', [Online].
Available: https://developers.google.com/youtube/iframe$_api_reference[Accessed : April28, 2018]$

[10] MIPI Alliance, 'MIPI alliance', 2015, [Online].
Available: https://www.mipi.org/specifications/csi-2 [Accessed: May1,2018]

[11] Mikko.Muukki, 'MIPI alliance webinar', 2015, [Online].
Available: https://www.mipi.org/sites/default/files/MIPI