VERIFICATION OF ANALOG & MIXED SIGNAL IPs USING SV-UVM VERIFICATION METHODOLOGY

A thesis Submitted in Partial Fulfilment of the Requirements

for the Award of the Degree of

Master of Technology
In VLSI Design

Submitted By

SUBHRANSHU SHEKHAR PADHEE

601762017

Under Supervision of

Dr. ANIL ARORA

Assistant Professor

ELECTRONICS AND COMMUNICATION ENGINEERING DEPARTMENT

THAPAR INSTITUTE OF ENGINEERING & TECHNOLOGY

(A DEEMED TO BE UNIVERSITY), PATIALA, PUNJAB

JULY, 2019

STMICROELECTRONICS INDIA PVT. LTD.
Greater Noida, Uttar Pradesh 201308, India

Date: May 30, 2019

CERTIFTCATE

This is to certify that **Subhranshu Shekhar Padhee** (Regn. No. 601762017), a student of M.Tech.(VLSI Design), **Thapar Institute of Engineering and Technology, Patiala** has successfully completed one year (June 2018 – May 2019) internship program in **STMicroelectronics Pvt. Ltd, Greater Noida.** His title of dissertation is "**Verification of Analog and Mixed Signal IPs using SV-UVM verification Methodology**".

During the period of his internship program, he was punctual and hardworking. I wish his every success in life.

Mr. Amit Singh
Senior Staff Engineer
STMicroelectronics Pvt. Ltd.

DECLARATION

I ,Subhranshu Shekhar Padhee hereby declare that the work presented in this thesis entitled "**Verification of Analog and Mixed Signal IPs using SV-UVM verification Methodology**" in partial fulfillment of the requirement for the award of degree of Master of Technology (VLSI Design) submitted at Electronics and Communication Engineering Department , Thapar Institute of Engineering & Technology (Deemed to be University), Patiala is an authentic record of work carried out under supervision of Dr. Anil Arora (Assistant Professor, ECED) from May 2018 to May 2019. The matter presented in this has not been submitted either in part or full to any other university or institute for the award of any other degree.

Date 8th July 2019

SS Padhee

Subhranshu Shekhar Padhee
601762017

It is certified that the above statement made by the student is correct to the best of my knowledge and belief.

8/7/18

Date…….

Dr. Anil Arora
Assistant Professor
Department of Electronics and Communication Engineering
Thapar Institute of Engineering & Technology
(A deemed to be University), Patiala, Punjab

ACKNOWLEDGEMENT

I would like to convey my deep sense of gratitude to my project guide, **Dr. Anil Arora, Assistant Professor, ECED** who is a constant source of motivation and firm support in carrying out this project. The support and supervision that he gave has helped me to progress in the project. His co-operation is highly appreciated and I highly oblige to him for his valuable comments and moral support during this research period.

I would also like to thank my Mentor or my manager, **Mr. Amit Singh, Senior staff Engineer, STMicroelectronics Private Limited, Greater Noida** for their esteemed guidance, Valuable suggestions and time throughout my internship. His guidance and vast knowledge directed me to accomplish critical tasks smoothly.

Also, my special gratitude to my family and friends for their constant support and motivation.

DATE: SUBHRANSHU SHEKHAR PADHEE

PLACE: Roll No-601762017

ABSTRACT

UVM Based Verification (UVM) is one of the broadly utilized verification methodology to improve the verifying nature of Analog and Mixed flag IPs Design so as to accelerate the checking procedure. A confirmation domain to check the functionality of IP by utilizing System Verilog - UVM based approach. The functionality of Analog IPs was checked by Cadence Incisive and VCS tools. With Incisive and VCS Simulator, not only check the status of output pins by applying different input pattern but also reduce the overall debugging effort and shortened debug turnaround time. With the best possible test plan and verification plan checking the functionality of AMS IPs became simpler. The verification has been done at pre-silicon stage to make post silicon stage bug free. Because verifying a design at post silicon stage takes a lot time and costly re-spin process. As time to market has become a crucial factor, a solution must be there with a verifying IP having both analog and digital block with checking behavior of analog block in a real manner. Verification of analog blocks have been done in digital environment which make simulation faster and get better performance. Most frameworks on-chip (SoC) plans today are Mixed signal ones, and all SoCs will be mixed signal at cutting edge process hubs sooner rather than later. The fundamental objective of this project is to automate the UVM Environment and make the verification quick and simple.

TABLE OF CONTENTS

Name of the Chapters **Page No**

LISTS OF FIGURES

LIST OF ABBREVIATIONS

UVM	Universal Verification Methodology
SV	System Verilog
IP	Intellectual Property
SOC	System on Chip
RTL	Register-transfer level
AMS	Analog and Mixed Signal
LDO	Low Dropout Regulator
PLL	Phased Locked Loop
VCO	Voltage Control Oscillator

CHAPTER 1

INTRODUCTION

1.1 MOTIVATION

As per the current scenario complexity of digital systems grow rapidly, so the verification methodologies are more essential for VLSI industry. While in the beginning time, verification plans were confirmed by taking a look at the output waveforms with manual checks. The System Verilog language came to help numerous verification engineers. The language highlighted a few instruments, similar to classes, cover groups and limitations, that facilitated a few parts of confirming a digital design and after that, check philosophies began to show up. The Universal Verification Methodology is a gathering of API and demonstrated verification rules composed for System Verilog that help a designer to make a proficient check condition. Mixed signal IPs are made of both analog as well as digital block. Conventional verification of mixed signal IPs are being done by individual verification. This individual verification for an SoC level design becomes a humongous task. Thus, verification of mixed signal IPs in an efficient way has become much more important.

Verification of mixed signal IPs is happened at pre-silicon as well as post silicon level. Pre-silicon verification is the activity of verifying the task performed on simulation model. It is basically done before fabrication. Thus, pre-silicon verification helps verifying the correctness of the design with zero bugs before implementation on Silicon. This is important to verify in pre-silicon stage as detection and removal of bugs in post-silicon stage is very costly.

For mixed signal IPs, verification of analog part takes a lot of time in an analog solver. This creates a need of an environment through which a mixed signal IP can verify in least amount of time. A solution is to mimic the behavior of analog part in digital environment.

1.2 OBJECTIVE

The main objective of dissertation is to implement an efficient verification methodology for mixed signal IPs. This is divided into three parts,

1. To implement an UVM methodology such as to ensure that functionality should meet the design specification.

2. Using Behavioral Modelling is efficient and an accurate environment to verify mixed signal IPs is implemented for better performance.

3. Using SV-UVM methodology we can easily determine the functional coverage ofmixed signal IPs.

1.3 PROBLEM STATEMENT

Mixed signal IPs are those that have analog and digital block in same chip. Traditionally verification of analog and digital parts is verified separately. They follow by black box approach. In this approach verification of individual digital block is being done with the assumption of analog blocks are pre-verified and vice versa. Thus, for full chip SoC verification only concerned with connection between analog and digital blocks. As functionality and complexity of SoC design is increased day by day with deep submicron process comes into picture traditional approach becomes obsolete. Verification of analog blocks along with its digital counterpart is essential for proper verification. Because any bug found in post silicon verification could lead to re-spin, which costs a lot. Because any bug which get unavoidable could lead to loss in terms of cost and reputation of industry. Verification is very important process in VLSI Design Flow.

1.4 Analog & Mixed signal IPs:

Signal of both analog and digital types can be announced in a similar module. Beginning, dependably, and simple procedural squares can show up in a similar module. Both simple and advanced sign qualities can be gotten to from any unique situation (analog or digital) in a similar module. Advanced sign qualities can be set from any setting outside of a simple procedural block. Mixed signal IPs are made of both Analog as well as digital block. Conventional verification of mixed signal IPs are being done by individual verification. Thus, verification of mixed signal IPs in an efficient way has become much more important. Verification of mixed signal IPs is done at pre-silicon as well as post silicon level. Pre-silicon verification is the activity of verifying the task performed on simulation model. It is basically done before fabrication. Thus, pre-silicon verification helps verifying the correctness of the design with zero bugs before implementation on Silicon. This is important to verify in pre-silicon stage as detection and removal of bugs in post-silicon stage is very costly. Still there are some works needed to be done in post-silicon verification. Ideally all bugs are detected at pre-silicon stage. For mixed signal IPs, verification of analog part takes a lot of time in an analog solver. This creates a need of an environment through which a mixed signal IP can verify in least amount of time. A solution is to mimic the behavior of analog part in digital environment.

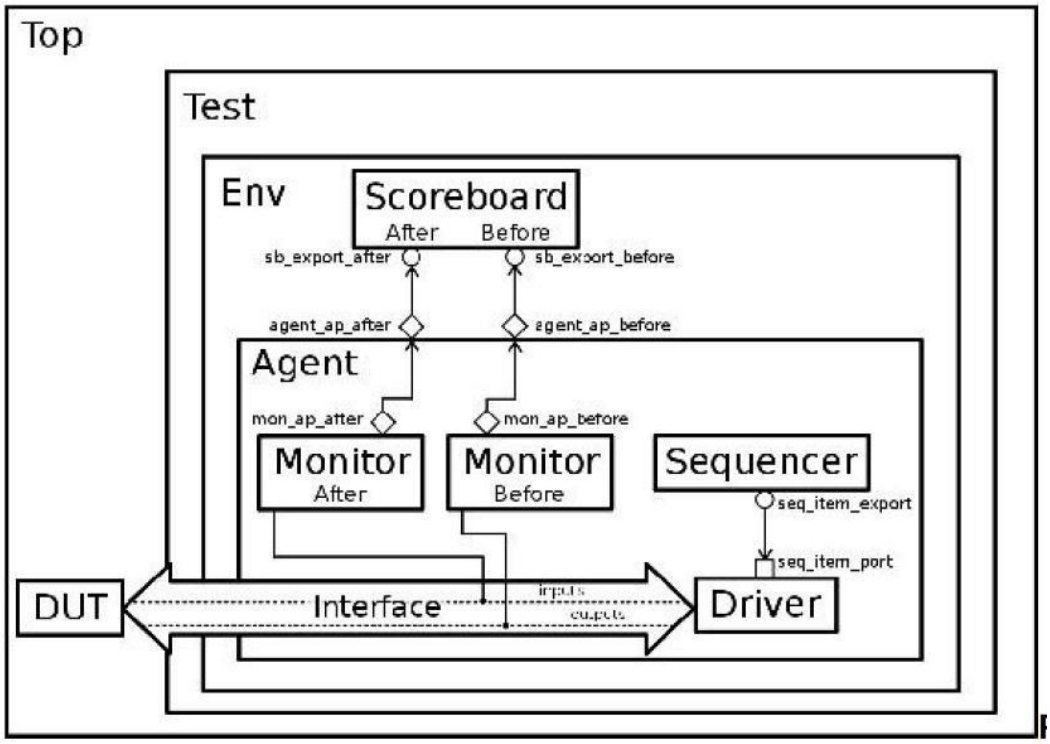

Figure 1.1 Final Testbench

1.5 Advantage of UVM based verification:

1. UVM is a System Verilog class library unequivocally intended to enable you to manufacture secluded reusable check segments and test-benches.

2. It is an industry standard so you can secure UVM IP from different sources and use them in your condition. On the off chance that you don't utilize UVM, you'll need to fabricate everything yourself starting with no outside help.

1.6 ORGANIZATION OF THESIS

The thesis has been divided into 8 chapters.

Chapter 2 Presents the literature survey related to verification and Mixed Signal IPs .

Chapter 3 Specifies UVM verification Methodology.

Chapter 4 Deals with the background of Analog & mixed signal IPs like LDO, PLL.

Chapter 5 IP level verification with Behavioral modelling.

CHAPTER 2

LITERATURE REVIEW

In this chapter, various research paper have been studied regarding verification of Analog and Mixed signal IPs and Universal Verification Methodology. Here authors are explained about the procedure, protocol and method to use for verification. Some authors are explained about the advantage of using LDO in various real-life application. Many authors are explained about the application and advantage of used of PLL in communication.

Juan Francesconi, et.al.[1] In this work, the Universal Verification Methodology (UVM) is examined through its application in the improvement of two test-benches for unit confirmation. The first focuses on a First Input-First Output (FIFO) support module and utilizes all the fundamental UVM parts; a scoreboard with a Reference Model and a Functional Coverage gatherer are additionally executed. The second one checks an I2C EEPROM slave module; a transport useful model for the I2C convention is characterized to encourage the driver usage, rising the dimension of reflection and permitting the reuse of the confirmation segment for other I2C device.

Chao Liang, et.al.[2] Analog-mixed signal(AMS), or or mixed-signal structure and applications are among the quickest developing need and market request. Most frameworks on-chip (SoC) structures today are mixed-signal ones, and all SoCs will be mixed-signal at cutting edge process hubs sooner rather than later. For cutting edge applications, for example, in auto-thought processes and Smart City frameworks high dependability of chip capacities are an absolute necessity. Mixed-signal confirmation technique has turned into a basic part to guarantee the accomplishment of a SoC plan at the top chip level.

Ken Kundert, et.al.[3] There are two reasons why digital architects are a long way in front of simple creators in improving their structure forms. To start with, digital designer went up against the need to structure exceptionally enormous and complex frameworks a lot sooner than simple originators. Think about that huge advanced chips today comprise of countless transistors, while complex simple chips contain just a huge number of devices. Second, the digital design structure issue is substantially more agreeable to digital design than the analog design issue.

Prince Gurha, et.al.[4] The time required for checking the structure is getting the opportunity to be dull as the complicacy of the chip design is extending exponentially. Nowadays, about 70% of the structure time is required for structure up the affirmation condition. Subsequently in this condition, it is basic to

diminish the affirmation time to quicken the whole improvement process. In any case, of course, the complicacy of the structure makes it difficult to cover all the corner cases in least time. Accordingly, to improve the structure detectable quality and to recognize and decode its inadequacies, Assertion Based Verification (ABV) is displayed.

Chai ; Zheng Xie, et.al.[5] At present, the conventional test seat is intended to a various leveled design, it contains three primary dimensions, to be specific Driver BFM, generator and the experiment segment. As is appeared in the below figure. The Fig 2.1 demonstrates that RM(reference model) and scoreboard may reuse in various undertakings or diverse test stages in a similar venture. In the event that the interface sign of DUT are the standard conventions, the screen and ace can be reused, for example the VIP got from EDA sellers.

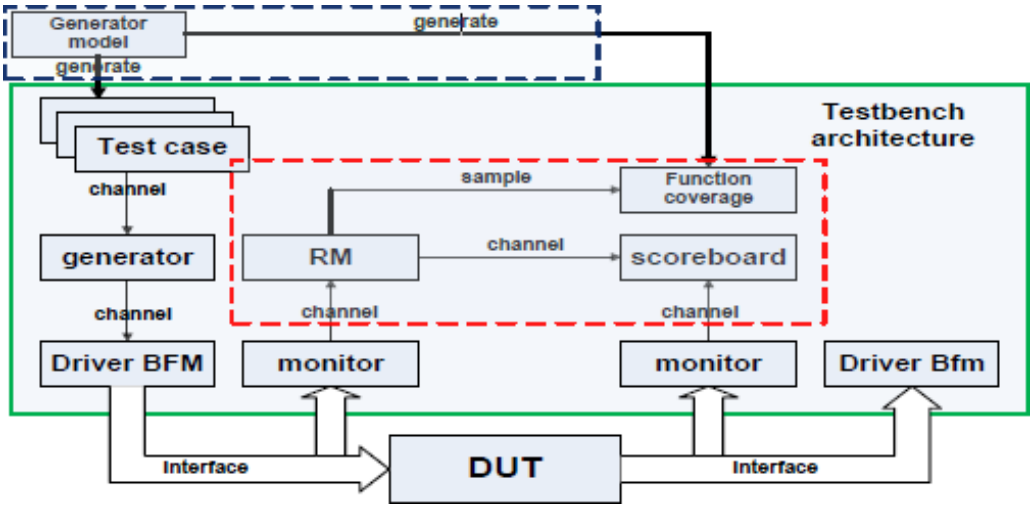

Figure 2.1 The complete and reusable test bench architecture

In the Fig2.1 the generator model that is the center of the reusable test seat was exhibited in the blue spotted edge. In the paper, the experiments generator model is disconnected from the DUT usefulness. This model composed by content language can consequently produce the experiment and capacity inclusion model. Owning to the DUT usefulness is the principles or the common circuits, the model might be reuse in various structures.

Chia-Min Chen, et.al.[6] The present compact gadgets need an effective power the board framework so as to keep going extended periods. Likewise, these gadgets normally need numerous on-chip voltage levels, so low dropout (LDO) voltage controllers are generally utilized in the versatile electronic gadgets, for example, MP3 players, mobile phones, and computerized cameras. It is particularly appropriate for

applications like RF IC and sound IC which require low commotion. Another favorable position of the direct controller is the low reserve current because of the nonappearance of exchanging. In compact gadgets, structure thought is to lessen quiet current to boost the lifetime of the battery.

Ujwala Ghodeswar, et.al.[7] Low dropout (LDO) voltage controller is a significant square in power official framework. Low dropout controllers (LDO) are generally utilized in electronic gadgets due to their accuracy yield voltage and a littler measure of commotion. It gives consistent yield voltage regardless of the adjustment in the heap current. Low dropout controller can be utilized as a straight controller and exchanging controller. Voltage controller with direct sort works at a little information yield differential voltage is named as low dropout voltage controller.

Young - Jun Park, et.al.[8] These days, the Internet of Thing (IoT) and wearable gadgets are rising to impacts on our day by day lives. The interest of raising the practical coordinated into cell phones is persistently expanded. The main power source supply to these portable devises is battery. To delay the battery life while the put away power in the battery is limited, the control utilization should be decreased. Voltage dimension of the battery isn't steady over working time, in this manner the yield voltage should be kept up with a decent guideline and high soundness. Low-dropout (LDO) controllers are generally utilized in battery fueled portable frameworks where little size, low power, and stable yield voltage are the basic determination. When all is said in done, a bandgap reference, an error amplifier (EA) and pass gadgets make behind an LDO.

YAO HAN, et.al.[9] Radio-recurrence (RF) generators are basic parts in the remote media transmission space, all things considered gadgets are crucial for giving exceedingly stable RF transporters and reference clock signals. The RF signal from a PLL generator demonstrates a high soundness and guarantees a very low float of the created recurrence.

S. Aditya, et.al.[10] PLL is an input control framework which bolts the reference sign to the VCO yield signal when they got sign is well inside the working scope of the PLL. PLL is utilized as on chip clock generator, recurrence synthesizer and as clock and information recuperation framework in PC, radio and media transmission framework. A PLL with wide tuning extent and low jitter is wanted. As the innovation scale down, PLL working at high frequencies are preferred. It is the VCO which chooses the acquisition scope of PLL. For a PLL with wide obtaining extent a wide tuning VCO is required.

By analyses of the various research journals, conference papers and IEEE transection papers it has been concluded that Verification is one of the emerging fields in the VLSI industries. Now a days the complexity of IPs are increases day by day. So, more protocol and data types are adding to the language reference manual of the system Verilog standardized by IEEE. Some of the authors are explain about the LDO and PLL application and how its change the trend of real-life application successfully.

CHAPTER 3

SPECIFIES UVM VERIFICATION METHODOLOGY

Verification is the way toward deciding if the actualized structure is right. verification is done to ensure that it fills in true to form necessity's i.e., it meets the required introductory detail of the item with the end goal that to guarantee that it will be utilized productively by proposed client or clients. In other word, to guarantee that item is exact and right or data gave is valid on giving a correlation. Check become progressively significant step by step as the intricacy increments so any items if have any kind of bug makes return and effect cost additionally to avert time to market delay. So, it is imperative to confirm the plan in configuration stream. As quick a bug is identified quicker, additional time which could get squandered in re-turn will be spared. UVM is one of the methodologies to develop generic verification environments were portable from one project to another project.

3.1 Need for verification

Consider a case in industry where item is conveyed to client without appropriate guaranteeing that the item was filling in as planned. In the event that could be a plausibility that item which is conveyed is having some issue which will prompt deficient misfortune as return costs much additionally will affect notoriety. Endeavors taken in confirmation is more than plan exertion, the structure is develop dependent on determinations and after that that particular is changed over to by configuration engineer into usage which has designer details which is then should be checked by a plan architect including all cases and in this way it is crafted by Verification specialist to demonstrate that the structure which is executed is really fill in as indicated by determinations. Check exertion ought to be free of structure exertion. In the event that both structure and Verification engineer both pursues same guideline of plan they may submit same blunder. Likewise, can't keep away from check as it cost excessively high and takes an excessive amount of time. In this manner, it is perceived as significant part in ventures.

3.2 UNIVERSAL VERIFICATIO METHODOLOGY

Universal Verification Methodology (UVM) created to automate the verification. The UVM Methodology is a basic industry standard written by System Verilog Language to create an efficient verification Environment.

3.2.1 UVM Structure

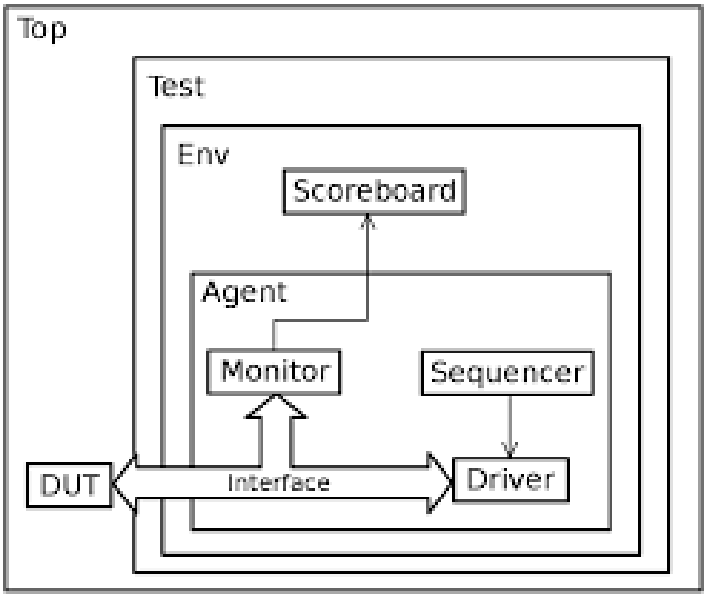

Figure 3.1 Typical UVM Testbench

3.2.2 UVM Sequence Item

The succession thing is composed by expanding the UVM_sequence_item, UVM_sequence_item acquires from the UVM_object by means of the UVM_transaction class. hence UVM_sequence_item is of item type. Arrangement thing consist of information fields required for producing the. stimulus. In request to create the upgrade, the succession things are randomized in groupings. Subsequently information properties in arrangement thing ought to for the most part be announced as rand and can have requirements characterized.

3.2.3 UVM Sequence

Succession creates arrangement of sequence item's and send to the driver by means of sequencer, Sequence is composed by expanding the UVM_sequence. An UVM_sequence_item is gotten

from an UVM_sequence_item and it is parameterized with the kind of sequence_item, this characterizes the sort of thing arrangement will send/receive to/from the driver.

3.2.4 UVM Sequencer

The sequencer control the progression of solicitation and reaction grouping things among arrangements and the driver. Sequencer and driver utilizes TLM Interface to impart transactions.UVM_sequencer and UVM_driver base classes has seq_item_export and seq_item_port characterized individually. Client need to interface them utilizing TLM associate method. Sequencer can be composed by expanding the UVM_sequencer parameterized with seq_item type.

3.2.5 UVM Driver

The driver is a block whose job is to collaborate with the DUT. The driver pulls exchanges from the sequencer and sends them monotonously to the sign dimension interface. This cooperation will be watched and assessed by another square, the screen, and therefore, the driver's usefulness should just be constrained to send the vital information to the DUT. Driver is composed by broadening UVM_driver. UVM_driver is acquired from UVM_component, Methods and TLM port (seq_item_port) are characterized for correspondence among sequencer and driver. The UVM_driver is a parameterized class and it is parameterized with the sort of the solicitation sequence_item and the kind of the reaction sequence_item .

3.2.6 UVM Monitor

The screen is an independent model that watches the correspondence of the DUT with the Test-bench. At most, it ought to watch the yields of the structure and, if there should be an occurrence of not regarding the convention's standards, the screen must restore an error. The screen is a detached segment, it doesn't drive any sign into the DUT, its motivation is to concentrate signal data and make an interpretation of it into important data to be assessed by different parts. A confirmation situation isn't constrained to only one screen, it can have numerous of them. The methodology that will be pursued for this confirmation plan is: test the two data sources, make an expectation of the normal outcome and contrast it and the consequence of the DUT. The screens will gather exchanges from the virtual interface and utilize the investigation ports to send those exchanges to the scoreboard. The code for the run stage can be planned a similar route with respect to the driver however it was overlooked in this section. A specific normal for this sort of

correspondence is that a port must be associated with a solitary fare. However, there are situations when we may be keen on having an uncommon port that can be connected to a few fares. A third sort of TLM port exists to cover these sorts of cases: the examination port. An investigation port works precisely like an ordinary port however it can identify the quantity of fares that are associated with it and each time a required capacity is asked through this port, every single other segment whose fares are associated with an examination port will be activated.

3.2.7 UVM Agent

User-defined agent is extended from UVM_agent, UVM_agent is acquired by UVM_component. An specialist ordinarily contains: a driver, sequencer, and screen. An operator doesn't require a run stage, there is no reenactment code to be executed in this square however there will be an interface stage, other than of the manufacture stage. The screens, the sequencer and the driver will be developed in the assemble stage. There will be need to make two examination ports; these ports will go about as intermediaries for the screens to be associate with an outside scoreboard through the operator's ports. The screens will gather exchanges from the virtual interface and utilize the examination ports to send those exchanges to the scoreboard. The code for the run stage can be planned a similar path with respect to the driver however it was precluded in this segment.

3.2.8 UVM Scoreboard

The scoreboard is a urgent component in a self-checking condition, it confirms the best possible task of a structure at a useful dimension. This part is the most troublesome one to compose, it changes from task to extend and from originator to designer. In this case, it has been chosen to make the forecast of the DUT usefulness in the screens and let the scoreboard contrast the expectation and the DUT's reaction. Be that as it may, there are creators who like to leave the expectation to the scoreboard. So, the usefulness of the scoreboard is abstract. In the specialist, two screens were made, subsequently, we should make two investigation sends out in the scoreboard that will be utilized to recover exchanges from the two screens. From that point forward, a technique thinks about () will be executed in the run stage and analyze the two exchanges. On the off chance that they coordinate, it implies that the testbench and the DUT both concur in the usefulness and it will restore an "alright" message. But we have an issue: we have two exchange streams originating from two screens and we have to ensure they are synchronized. This should be possible physically by composing appropriated compose () works however there is a simpler and cleaner method for doing this by utilizing UVM Fifo.

3.2.9 UVM Environment

The environment is an exceptionally straightforward class that instantiates the operator and the scoreboard and interfaces them together. User-characterized condition is gotten from UVM_env, UVM_env is acquired from UVM_component. Environment is the compartment class, It contains at least one specialists, just as different segments, for example, scoreboard, top dimension screen, and checker.

3.2.10 UVM Test

Finally, one more block has to be made: the test. This block will get from the UVM_test class and it will have two purposes: Connect the sequencer to the grouping and characterizes the test situation for the test-bench. It may be asked for what reason are we interfacing the sequencer and the succession in this square, rather than the specialist square or the arrangement block. The reason is straightforward: by determining in the test class which grouping will go be produced in the sequencer, we can undoubtedly change the sort of information is transmitted to the DUT without upsetting the operator's or grouping's code. User-characterized test is gotten from UVM_test, UVM_test is acquired from UVM_component. The UVM test-bench is enacted when the run_test() technique is called, the worldwide run_test() assignment ought to be indicated inside an underlying block.

3.2.11 UVM Top

The top square will make examples of the DUT and of the test-bench and the virtual interface will go about as an extension between them. The interface is a module that holds every one of the sign of the DUT. The screen, the driver and the DUT are largely going to be associated with this module.

3.3 THE VERIFICATION PROCESS

Our motivation as a verification engineer is to ensure that the device can achieve that errand effectively that is, the structure is an exact portrayal of the detail. Bugs are what you get when there is an inconsistency. The procedure of confirmation parallels the plan creation process. A fashioner peruses the equipment determination for a square, translates the human language depiction, and makes the relating rationale in a machine-learning structure, generally RTL code. To do this, the person needs to comprehend the info design, the change work, and the configuration of the yield. Along these lines, as

a confirmation engineer, in addition to the fact that you have to comprehend the structure and its purpose, yet additionally, you need to consider all the corner experiments that the creator probably won't have pondered. By having more than one individual play out a similar elucidation, you have added excess to the plan procedure. As the verification engineer, your responsibility is to peruse a similar equipment particular and make an autonomous appraisal of what they mean. Your tests at that point practice the RTL to demonstrate that it coordinates your understanding.

3.4 CHECKING METHODOLOGY

Methodology used in this project is Universal Verification Methodology. The UVM structure follow the top module with sequence items, sequence lib, sequencer, driver, monitor, agent, scoreboard, environment and test_lib. Sequence_lib synchronize with test lib so, that the test cases run perfectly. The checking methodology used in verification is self-checking. A self-checking Test-bench checks expected outcomes against genuine outcomes got from the simulation. For initial test-bench writing Self-checking testbenches require more effort. This procedure can drastically Reduce the measure of exertion expected to re-check a structure after a change has been made to the DUT. Investigating time is fundamentally abbreviated by valuable blunder following data that can be incorporated with the Test-Bench to demonstrate where a structure comes up short. A self-checking Test-Bench has two noteworthy parts, the information squares and yield squares. Info square comprise of improvement and driver to drive the boost to DUT. The yield square comprises of screen to gather the DUT yields and confirm them.

CHAPTER 4

BACKGROUND OF ANALOG & MIXED SIGNAL IPs

In this chapter, the basic details of Analog and Mixed signal IPs like LDO and PLL has been discussed. LDO is one type of Linear Regulator used in the real-life application when minimum drop of output voltage required with respect to input voltage. PLL is used to generate a fixed output high frequency with respect to input low frequency. Output frequency have high noise immunity.

4.1 LINEAR REGULATOR

IC linear voltage regulator have been around for a considerable length of time. These easy to-utilize gadgets show up in almost every kind of electronics equipment, where they produce a perfect, exact yield voltage utilized by touchy parts. A circuit which can keep up the DC yield voltage of a power supplies consistent regardless of change in line voltage and variety in burden current is called Voltage Regulator. Key controller advantages and applications incorporate Accurate supply voltage, Active commotion sifting, Protection from overcurrent deficiencies, Inter-organize disconnection (decoupling), Generation of numerous yield voltages from a solitary source, Useful in consistent current sources.

4.2 LOW-DROPOUT LINEAR REGULATOR

A low dropout controller is a class of linear regulator that is intended to limit the immersion of the yield pass transistor and its drive requirements. A low-dropout direct controller will work with info voltages just marginally higher than the ideal yield voltage. For instance, "exemplary" straight controllers, for example, the 7805 need about 2.5 to 3V higher information voltage for a given yield voltage. For a 5V yield, these more established gadgets need a 8V input. We see that value touchy applications incline toward straight controllers over their examined time counterparts. The structure choice is particularly obvious for producers of correspondences equipment, small devices, battery worked systems, low current devices, high execution chip with rest mode (quick transient recuperation required). Linear controllers are less vitality proficient than exchanging regulators. Depending upon the application, direct controllers have a few recovering highlights like , lower yield clamor is significant for radios and different interchanges equipment, faster reaction to information and yield transients, easier to utilize in light of the fact that they require just channel capacitors for task , by and large littler in size and more affordable.

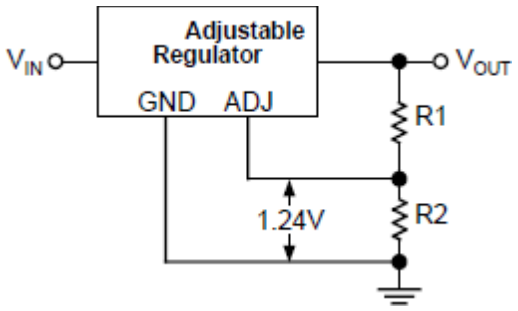

Figure 4.1 An adjustable LDO

utilizes the proportion of two resistors to decide its yield voltage. Adjustable controllers utilize the proportion of two resistors to increase the reference voltage to deliver the ideal yield voltage (see Figure 4.1).

4.3 LDO OUTPUT VOLTAGE ACCURACY

LDO Regulators are high precision gadgets with yield voltages processing plant cut to much superior to 1% exactness. Over the working temperature, input voltage, and burden current ranges, their most pessimistic scenario exactnesses are still superior to ±2%. For flexible controllers, the yield likewise relies on the precision of two programming resistors. A few frameworks require supply voltage exactnesses superior to ±2.5% including clamor and homeless people. While clamor is commonly not a noteworthy supporter of yield incorrectness, load drifters brought about by quickly fluctuating burdens, (for example, rapid microchips), are critical, notwithstanding when utilizing quick transient-reaction LDO controllers and superb channel capacitors.

$$V_{OUT} = V_{REF}\left(1+\frac{R1}{R2}\right)$$

4.4 IMPROVING REGULATOR ACCURACY

Accomplishing a most pessimistic scenario mistake of ±2.5%, including all D/C and A/C blunder terms, is conceivable by expanding the essential precision of the controller itself, yet this is costly since high current controllers have noteworthy self-warming. Its inside reference must keep up exactness over a wide temperature run. Testing for this dimension of execution is tedious and raises the expense of the controller, which is unsuitable for incredibly value touchy commercial centers. A few frameworks require superior to ±2% exactness.

4.5 PHASE-LOCKED LOOPS

A phase-locked loop (PLL) consist of a phase comparator and voltage controlled oscillator (VCO) .PLL connected with an oscillator maintains a constant phase angle relative to a reference signal. Phase-locked loops can be used, to generate stable output high frequency signals from a fixed low-frequency signal.

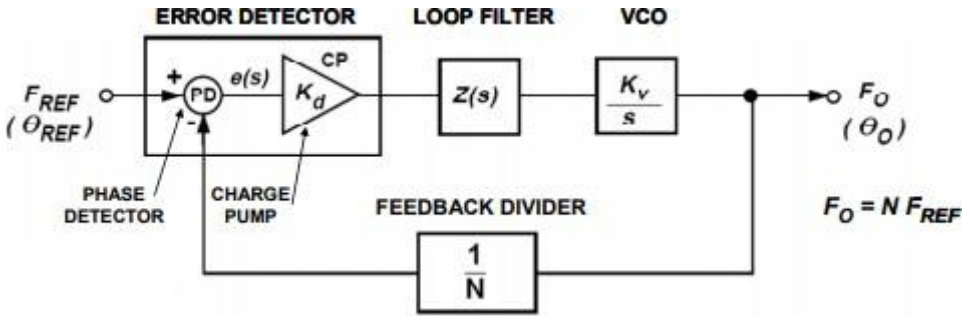

Figure 4.2 Basic Block Diagram of PLL

The essential block of the PLL are the Error Detector, Loop Filter, VCO, and a Feedback Divider. Negative input powers the mistake signal, e(s), to approach zero so, all in all the criticism divider divide the output frequency and the reference recurrence are in stage and recurrence lock, and FO = NFREF. An PLL, comprises of three fundamental components:

- Phase comparator/detector: As the name infers, this circuit obstruct inside the PLL thinks about the period of two signal and creates a voltage as indicated by the stage contrast between the two sign. This circuit can take an assortment of structures.
- Voltage controlled oscillator, VCO: The voltage-controlled oscillator is the circuit obstruct that produces the radio frequency signal that is ordinarily considered as the yield of the circle. Its recurrence can be controlled over the operational recurrence band required for the circle.
- Loop filter: This block is utilized to filter the output from the stage comparator in the phase locked loop, PLL. It is utilized to expel any parts of the signal of which the stage is being looked at from the VCO line, for example the reference and VCO input. It likewise administers huge numbers of the attributes of the circle including the circle solidness, speed of lock, and so on.

4.6 NEED OF PLL

A Phase locked loop used in real life application for Skew suppression, jitter reduction, Frequency Synthesis, Clock Recovery.

4.6.1 Jitter Reduction

Signals go through jitter when travel through the communication channel. Jitter is due to device and external noise added to the signal. Timing jitter timing jitters are retrieved from a storage medium. Jitter define itself as variation of the period of a waveform, a type of corruption that cannot be removed by amplification and clipping even if the signal is binary. A PLL can be used to reduce the jitter.

4.6.2 Skew Suppression

Skew Suppression is a basic issue in rapid digital system frameworks. Here, a framework clock, CKs, enters a chip from a printed-circuit (PC) load up and is cradled (in a few phases) to hone its edges and drive the heap capacitance with insignificant postponement. The vital trouble in such a course of action is, that the on-chip clock, regularly drives a few nano farads of gadget and interconnect capacitance, showing huge delay concerning CKs. The subsequent slant lessens the planning spending plan for on-chip and between chip tasks.

4.6.3 Frequency Synthesis

Numerous applications require recurrence augmentation of intermittent sign. For instance, in the advanced arrangement of the data transmission constraint of PC loads up compels the recurrence of CKs, though the on-chip clock recurrence may should be a lot higher. As another precedent, remote handsets utilize a nearby oscillator whose yield recurrence must be changed in little, exact strides, for instance, from 900 MHz to 925 MHz in.steps of 200 kHz.These epitomize the issue of "recurrence combination," an undertaking performed productively utilizing stage bolted frameworks.

4.6.4 Clock Recovery

In numerous frameworks, information is transmitted or recovered with no extra planning reference. In optical interchanges, for instance, a flood of information streams over a solitary fiber with no going with clock, yet the recipient should in the long run procedure the information synchronously. Accordingly, the planning data (e.g., the clock) must be recuperated from the information at the get end. Most clock recuperation circuits utilize stage locking.

4.7 APPLICATION OF PLL

Some phase lock loop applications include:

- **FM demodulation:** One major application of PLL is that of a FM demodulator.In this PLL applications enables high quality audio to be demodulated from an FM signal.
- **AM demodulation:** Phase locked loops used in the synchronous demodulation of amplitude modulated signals. Using this approach, the PLL locks onto the carrier so that a reference within the receiver can be generated. As this corresponds exactly to the frequency of the carrier, it can be mixer with the incoming signal to synchronous demodulate the AM.
- **Indirect frequency synthesizers:** Use within a frequency synthesizer is one of the most important phase locked loop applications. Although direct digital synthesis is also used, indirect frequency synthesis forms one of the major phase locked loop applications.
- **Signal recovery:** The fact that the phase locked loop is able to lock to a signal enables it to provide a clean signal, and remember the signal frequency if there is a short interruption. This phase locked loop application is used in a number of areas where signals may be interrupted for short periods of time, for example when using pulsed transmissions.
- **Timing distribution:** Another PLL application is in the distributtion of precise timed clock pulses in digital logic circuits and system, for example within a microprocessor system.

4.8 CHARGE PUMP BASED PLL

The Charge Pump PLL (CPPLL) is an extension of the basic PLL which requires the addition of a CP between the phase detector and loop-filter. The CP converts the voltage fluctuation in the Phase detector to corresponding current signal thereby reduces the static error. Figure 4.3 shows the usage of CP between Phase Detector and Loop Filter (LF).

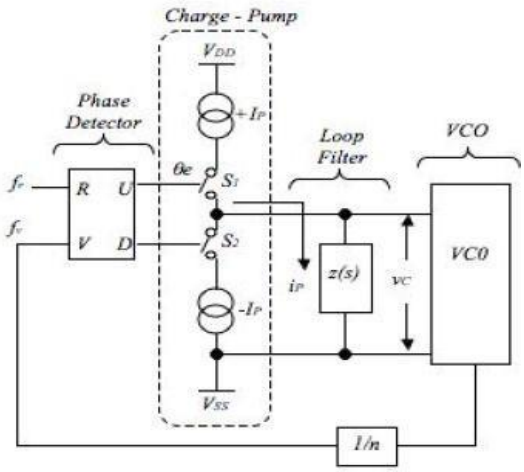

Figure 4.3 Charge Pump Based PLL

The CP shown in Figure 4.4 consists of a set of current sources with magnitudes of I_{P1} and I_{P2} amps respectively. In most cases, the current sources are symmetrical. Thus $I_{P1} = I_{P2} = I_P$.

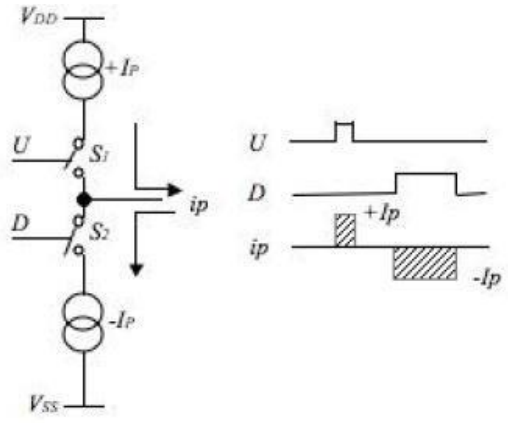

Figure 4.4 Charge Pump

The reference information is connected to the one of the PFD and VCO yield is given to another information. This usage detects the progress at the info and yield identifies stage or recurrence distinction and enacts the CP as needs be. At the point when circle is turned on, out might be far in, and the PFD and CP differ the control voltage with the end goal that out methodologies in. Whenever info and yield frequencies are adequately close, the PFD works as stage identifier, performing stage lock. Presently think about a case, that out – in drops to zero. For this situation PFD just produce QA = QB = 0. The CP accordingly stays inactive and CP supports a steady control voltage. However, this does not imply that PFD and CP are never again required. In the event that Vconst stays consistent for quite a while, the VCO recurrence and stage start to float. Specifically, the VCO make arbitrary varieties in the

swaying recurrence that can result in huge collection of stage blunder. At that point, the PFD identifies the stage distinction, produces restorative heartbeats on QA or QB that modifies the VCO recurrence through CP and filter. Also, as stage correlation is performed in each cycle, the VCO stage and recurrence can't float considerably.

CHAPTER 5

IP LEVEL VERIFICATION WITH BEHAVIORAL MODELLING

To verify mixed signal IPs various technique has been introduced. This section covers all methods to verify Analog and Mixed signal design using different methods.

5.1 BEHAVIORAL MODELLING

Describing an analog circuit in higher abstraction level makes simulation of this model more effective thus behavioral modelling is a key component in verification methodology for mixed signal. Behavioral modelling is where actually one mimics the analog circuit behavior. While creating model for such analog and mixed signal model is not an easy task there are several challenges:

The analog circuit which is implemented should be verified to make sure that the implemented model is working as specification with high accuracy. If there is any specification changes it must be synchronized with circuit. For any issue while simulating, the circuit should be written in such way that it doesn't not converge any issues. The main purpose of model must be understood and according to that model, or architecture must be chosen. Models are developed before circuit availability in a top-down approach. While in bottom-up approach for verification model needs to be verified with an already implemented block. For implementing such model must have good understanding of analog and mixed signals, coding and debugging

5.1.1 Types of Modelling

It is important to use different modelling schemes for verification of a large integrated chips. Modelling can be done in different manner depending upon the application are and time to market need of industry.

- **Device based design (Spectre, SPICE) –** in this circuit is build using standard transistor level technique.
- **Analog modelling (Verilog-A) –** Verilog-A is the subset of Verilog-AMS which describes the current and voltage relation of analog part.
- **Mixed Signal modeling (Verilog-AMS, VHDL-AMS) -** it is the extension of VHDL known as IEEE 1076 VHDL. It describes the behavior of analog and digital blocks.

- **Real Value modelling (VHDL, System Verilog) –** in this analog block is model as a discrete real number data. It provides extension to support full chip verification of SoC having analog blocks. The analog block can be abstracted using transistor level netlist or FAST SPICE depending on application and according to what form of abstraction has been used the accuracy and performance and effort taken while modelling an analog circuit varies.

5.2 ANALOG MODELLING

For modelling electrical behavior of analog circuit pure analog languages like SPICE are used. Thus, for modelling analog behavior Verilog-A is a standard language. Verilog-A language creates the relation between voltage and current of the system. Can directly write impedance characteristics and integral and derivatives using this language. Then Verilog-A model converts this in some set of equations whether linear or differential which is suitable for simulator.

On comparing to a transistor level this well-defined model can sped up to 10 to 70 times of SPICE simulation. But speed of simulation also depends upon the application for which the equations are formed. By using behavioral modelling of analog circuit performance is increased.

5.3 DIGITAL MODELLING

For a digital circuit with digital input and output characteristics a pure digital solver is used. VHDL, Verilog and System Verilog are the language through which a digital circuit is modelled. These are efficient in handling logic and timing relationships of input/output using a digital simulation environment. Although this approach is not for analog signals. Thus, this model is used for an entirely digital model where an analog block is assumed to be black box means, assuming that analog block is pre-verified. But this digital approach is extended where an analog bock can be models in a digital environment taking advantage of modelling.

5.4 MIXED SIGNAL MODELLING

In a mixed signal modelling and analog and digital model solver come in single simulator. For modelling mixed signal languages like Verilog-AMS, VHDL-AMS can be used. These languages allow modelling of mixed signal in a simple manner as analog block can be model through its electrical behavior modelling approach and digital block can be modelled using discrete or event modelling technique. With this two simulation algorithm data can be transparently passed. And is very efficient way to write test benches for mixed signal signals. For analog and mixed signal, Verilog-AMS can be used to write read both analog and digital blocks which makes it an efficient environment in which one can implement an

entire system. And further real number modelling is a technique used to write mixed signal models. The simulation speed for AMS mode depends on application and replacing transistor level circuit.

5.5 REAL VALUE MODELLING

Real Value Modelling (RNM) is a modelling technique which actually mimic the analog circuit behavior. This is a special technique where electrical signals of analog blocks are represented using time varying real values. For analog simulator, some set of equations are defined in a model. And this model which is defined by equations are enforced by simulator by adding constraint by applying Kirchhoff's law and then solve this constraint of equations at every step to determine the voltage and current from those equations.

But in a digital environment there is no set of equations no Kirchhoff's law. In this output is directly evaluated using input disregarding feedback or any voltage or current which can create interdependencies . Real value model procedure is easy where just using input/ output transfer function can write mathematic algorithm and check whether the corresponding output is changes on change of input. Thus, it is a straight forward concept. Also, the inputs like voltage, current and power are pass as a real value in model and later check whether they are coming in with some specified tolerance. Thus, biasing should be proper ad checking of biasing is easy task. And it is not difficult to create behavior model of analog circuit as an analog circuit verification on transistor level to behavior in higher level of abstraction is already a known process using VHDL-AMS thus using Real model is not a difficult task.

For Real Number Modelling languages like Verilog, System Verilog is used which has the property where a signal can be represented as a real i.e., floating point value which makes them behaves more like an analog circuit values as in digital actually deal with four vales that is 0, 1, x and z but not with the real modelling case where actual float values called real values are used.

CHAPTER 6

FUNCTIONALITY OF ANALOG & MIXED SIGNAL IPs

In this chapter, the functionality of AMS IPs like LDO and PLL. Basic functionality of LDO design ,Bandgap model design have been discussed, Voltage-Regulator design and PLL design are explain below. Designer design these IPs according to the certain specification given by the customer. Verification engineer write the verification plan to verify these design and made the design bug free. Verification Engineer main aim to verify the design in such a way that functionality of the design meet the specification given by the designer.

6.1 FUNCTIONALITY OF LDO MODEL

According to the Figure 6.1 functionality of LDO depends on the Supply pins, enable pin and Select pin.When Vdd pin is logic high and Ground pins are logic low ,status of the supply pins are valid .Supply pins have highest priority according to the behaviour of LDO Model .Enable pin has second most priority among all the input pins . Select pin has lowest priority among all the input pins.When all the supply pins and Enable pin are Valid then what ever the valid status of Select pin Output must be logic high after some delay .This delay is nothing but the start_up delay of LDO Model.When select pin is logic High then the output voltage (VOUT1) of LDO is 2V and similarly for logic low the output voltage (VOUT1) is 1.5 V .When Enable is Logic high then after some delay the outputs are active .Start_up delay of the LDO IS 100 us .In case of invalid scenario when all the supply pins are invalid,Vdd supply is logic low and Ground Supply is Logic high then outputs are corrupted. If supplies are valid and Enable pin status invalid then then also outputs are corrupted. When all the input supplies and Enable pin are valid but select pin is invalid then also outputs are corrupted. If within the start_up time inputs are toggle then also outputs are corrupted.

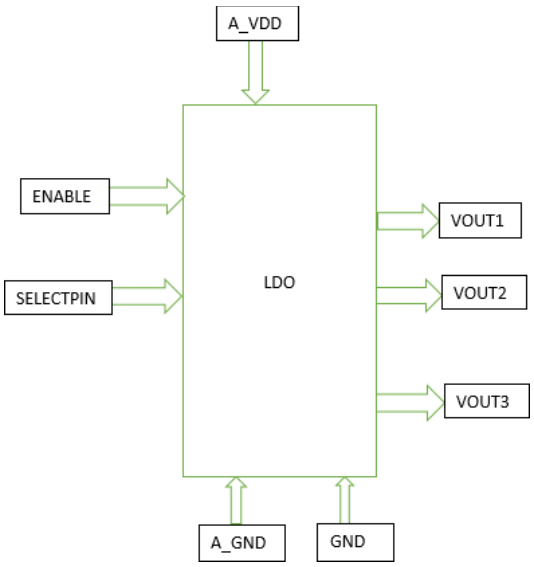

Figure 6.1 LDO MODEL

After getting the specification of LDO IP from the designer verification engineer write the verification plan to verify that particular IP. In Test plan write some test cases for verification of particular IP. After verification plan verification engineer write the Test bench with UVM Methodology. Reference model is written by the verification engineer with System-Verilog Language. With the help of self-checking test bench verify that whether the observed outputs from DUT matched with Expected outputs from reference model. If any mismatched between the observed and expected outputs, then UVM error reflects on the terminal.

6.2 FUNCTIONALITY OF BANDGAP MODEL

According to the Figure 6.2 functionality of BANDGAP Model depends on the Supply pins and Enable pin .When Vdd pin is logic high , Ground pin is logic low and SUB pin is logic low ,status of the supply pins are valid .Supply pins have highest priority according to the behavior of LDO Model .Enable pin has second most priority among all the input pins .When all the supply pins and Enable pin have valid supply then the outputs are logic high after some delay. This delay is called as start-up delay of BANDGAP Model. BANDGAP Model has 80us start-up delay. In case of invalid scenario when all the supply pins are invalid,Vdd supply is logic low , GND Supply is Logic high and SUB supply is logic high then outputs are corrupted. If supplies are valid and Enable pin status is invalid then then also outputs are corrupted. If any supply pins and Enable pin are toggle in between start-up time then also outputs are corrupted.

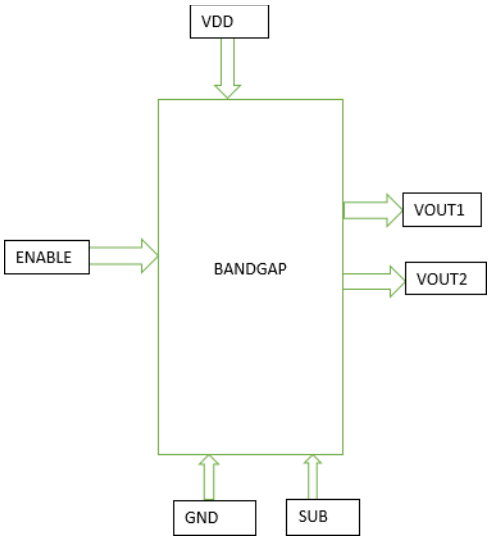

Figure 6.2 BANDGAP MODEL

After getting the specification of BANDGAP IP from the designer verification engineer write the verification plan to verify that particular IP. In Test plan write some test cases for verification of particular IP. After verification plan verification engineer write the Test bench with UVM Methodology. Reference model is written by the verification engineer with System-Verilog Language. With the help of self-checking test bench verify that whether the observed outputs from DUT matched with Expected outputs from reference model. If any mismatched between the observed and expected outputs, then UVM error reflects on the terminal.

6.3 FUNCTIONALITY OF VOLTAGE REGULATOR MODEL

According to the Figure 6.3 functionality of voltage regulator depend on the 7 supply pins and 10 input pins.IN1 is power on reset signal ,high when VDD3 reach voltage regulator start up value.IN2 is high when reference voltage and biasing are on and steady state reached .This signal gates the regulator enable.IN3 is few micro ampere biasing current generated from PMOS transistor.IN4 pin deter-mind the reference voltage.IN5 pin apply for place regulation in IP.IN6 is high then regulate the enable signal.IN7 pin control the regulated output voltage.IN8 pin regulate the output to pull-down mode. IN9 regulate the additional DC-load control to improve the load transient, gain margin and phase margin. VOUT1 regulate the output supply. VOUT2 and VOUT3 are power on reset signal, high when LDO output reached steady state. VOUT4 analogue test outputs. VDD1 regulator output power supply. VDD2 main regulated input supply. VDD3 digital core supply. GND1 main ground supply. GND4 P Substrate.

After getting the specification of VOLTAGE REGULATOR IP from the designer verification engineer write the verification plan to verify that particular IP. In Test plan write some test cases for verification of particular IP. After verification plan verification engineer write the Test bench with UVM Methodology. Reference model is written by the verification engineer with System-Verilog Language. With the help of self-checking test bench verify that whether the observed outputs from DUT matched with Expected outputs from reference model. If any mismatched between the observed and expected outputs, then UVM error reflects on the terminal.

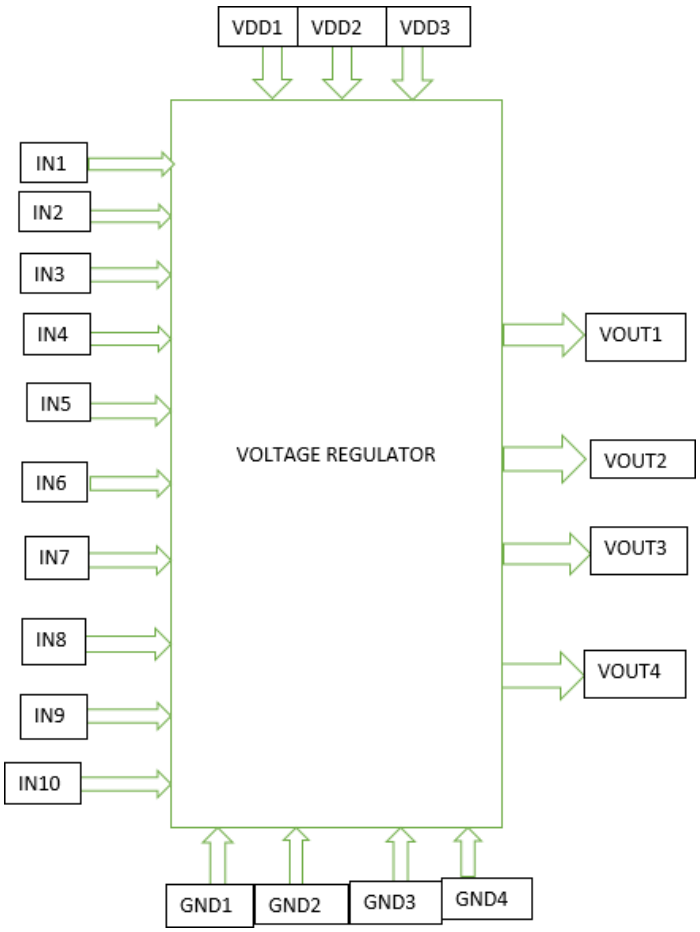

Figure 6.3 Voltage Regulator Model

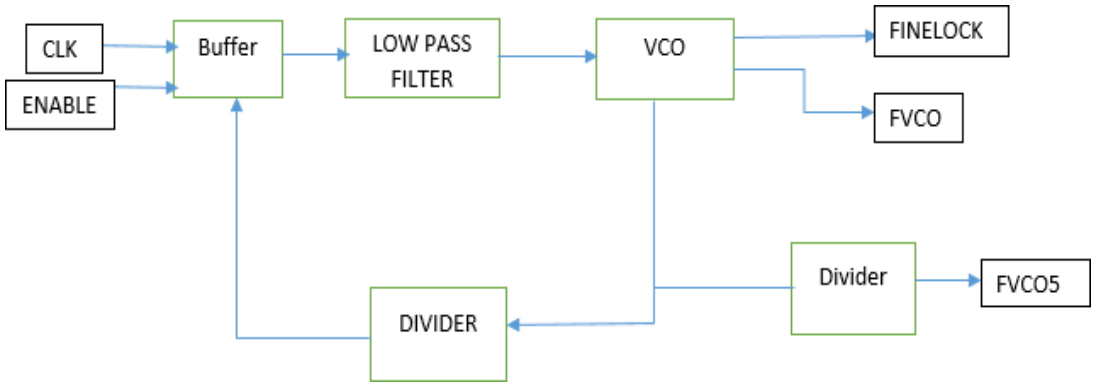

Figure 6.4 PLL MODEL

After getting the specification of PLL IP from the designer verification engineer write the verification plan to verify that particular IP. In Test plan write some test cases for verification of particular IP. After verification plan verification engineer write the Test bench with UVM Methodology. Reference model is written by the verification engineer with System-Verilog Language. With the help of self-checking test bench verify that whether the observed outputs from DUT matched with Expected outputs from reference model. If any mismatched between the observed and expected outputs, then UVM error reflects on the terminal.

6.4 VERIFICATION PLAN OF LDO

VP NO	Feature	TYPE	Feature TYPE	Verification Criterion
F1	LDO Normal mode of operation .	Valid	Logic, Transition, Temporal (Delay)	Coverage Point Coverage Assumed : Reference generator giving desired o/p
F2	POWER down mode with ENABLE goes LOW	valid	Logic,transition	Coverage Point Coverage Assumed : Reference generator giving desired o/p
F3	Supply incorrect	Invalid	Logic	Coverage point
F4	Supply OFF	OFF	Logic	Coverage point
F5	input ENABLE changes before output VOUT1 comes	Invalid	Logic,Temporal (delay)	Coverage Point Coverage Assumed : Reference generator giving desired o/p
F6	VOUT1 goes high after some delay when AvddPOKHV goes Low->high	valid	Logic,Transition,Temporal(delay)	Coverage point
F7	SELECTPIN LOW 1.5V mode ofoperation	Valid	Logic ,transition	Coverage Point Coverage Assumed : Reference

				generator giving desired o/p
F8	SELECTPIN HIGH 1.8V mode of operation	Valid	Logic ,transition	Coverage Point Coverage Assumed : Reference generator giving desired o/p
F9	ENABLE becomes invalid L->x or H->x	invalid	Logic,transition	Coverage point
F10	SELECTPIN in invalid case L->X or H->X	invalid	Logic ,transition	Coverage point
F11	VOUT1 becomes 1.5 V when SELECTPIN=LOW	valid	Logic,transition	Coverage point
F12	VOUT1becomes 2V when SELECTPIN=HIGH	valid	Logic,transition	Coverage point
F13	VOUT1becomes X invalid	invalid	Logic,transition	Coverage point
F14	Check status of VOUT3	Valid/invalid	Temporal delay	Coverage point
F15	Check status of band gap voltage VOUT2	Valid/invalid	Logic,transition	Coverage point

Verification Plan is playing the major role during verification. Verification plan written by the verification engineer to verify the model design by the designer. Verification plan is the superset of test-plan developed by the designer. Verification plan defined all the features of the Model and also defined the cross coverage of the model or IP design by the designer. By the help of cross coverage, we are easily defining the functionality of IPs and conclude that all the specification meets by the design which are design by the designer.

CHAPTER 7

RESULTS AND DISCUSSION

In this chapter, the simulation results of all the IPs in Incisive and VCS tools have been explained. Results are taken according to various input condition of supply pins and input pins. Outputs are changes accordingly when input pins are changes .We discussed about the changes of outputs with different simulation time .

7.1 SIMULATION RESULTS OF LDO MODEL IN INCISIVE TOOL

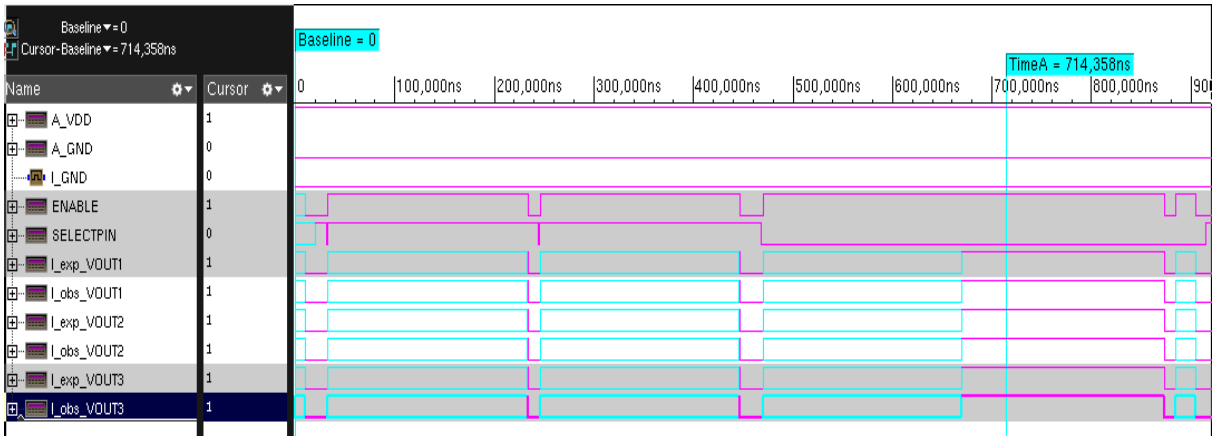

Figure 7.1 Simulation Result of LDO with all valid inputs

From the above figure it's clearly reveals that when all the inputs are valid supply then outputs are logic high after the start-up delay of LDO. If Enable pin is logic low, then the LDO goes to power down mode. At time 714358ns when all the power supplies are valid and Enable pin is logic high then outputs are valid shown in figure. If one of the supplies pins are corrupted than outputs are corrupted .Supplies pins have the highest priority among all the input pins. If Enable pin is goes low or invalid than output pins are corrupted. Enable pins has second highest priority among all the input pins. If select pin is corrupted than outputs are corrupted.

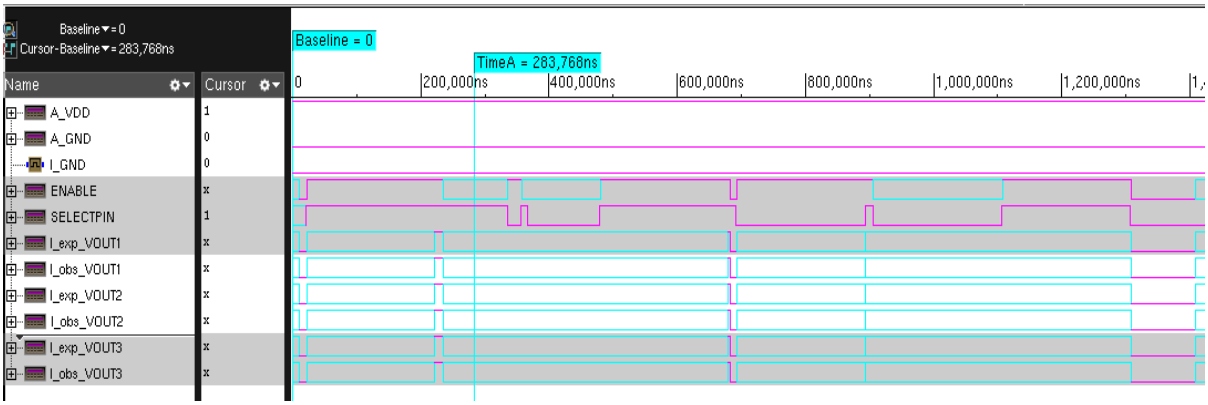

Figure 7.2 Simulation Result of LDO with ENABLE pin as invalid inputs

From the above figure its clearly reveals that when all the inputs are valid supply and Enable pin is invalid then outputs are corrupted.If Enable pin is logic low then the LDO goes to power down mode.At time 283768ns when all the power supplies are valid and Enable pin is invalid mode then all outputs are corrupted shown in figure.

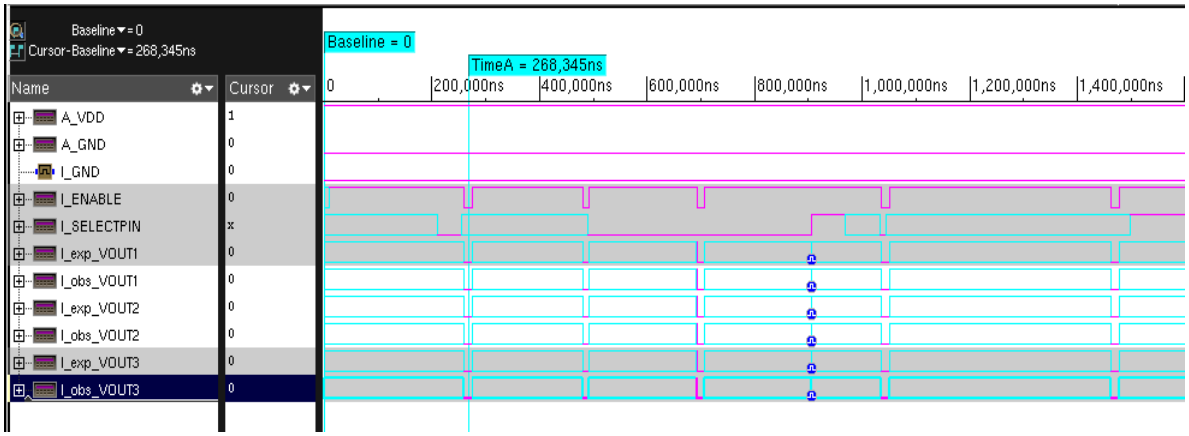

Figure 7.3 Simulation Result of LDO with SELECT pin as invalid inputs

From the above figure its clearly reveals that when all the inputs are valid supply and SELECT pin is invalid then outputs are corrupted.If Enable pin is logic low then the LDO goes to power down mode.At time 210768ns when all the power supplies are valid and SELECT pin is invalid mode then all outputs are corrupted shown in figure.

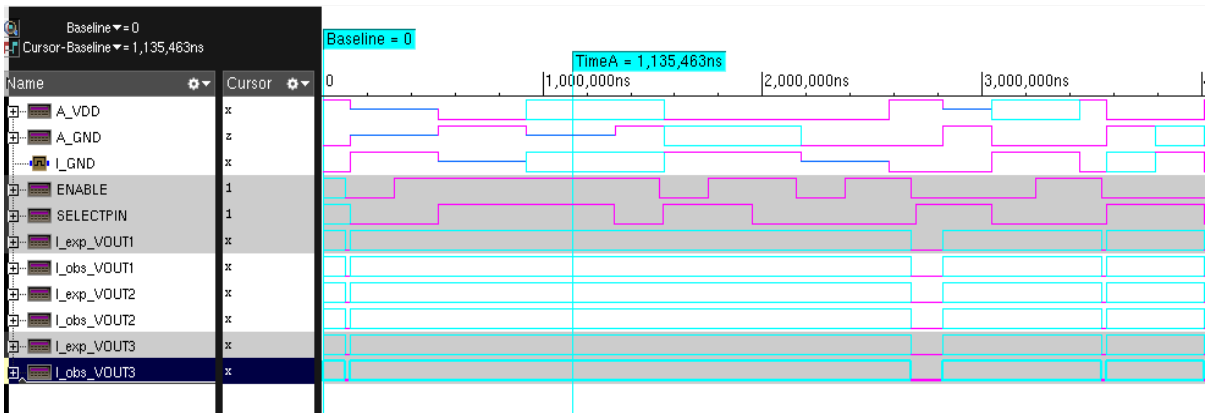

Figure 7.4 Simulation Result of LDO with all power supplies are invalid and other inputs are valid.

From the above figure it's clearly reveals that when all the power supplies are invalid then outputs are corrupted. If Enable pin is logic low, then the LDO goes to power down mode. At time 1135463ns when all the power supplies are invalid then all outputs are corrupted shown in figure.

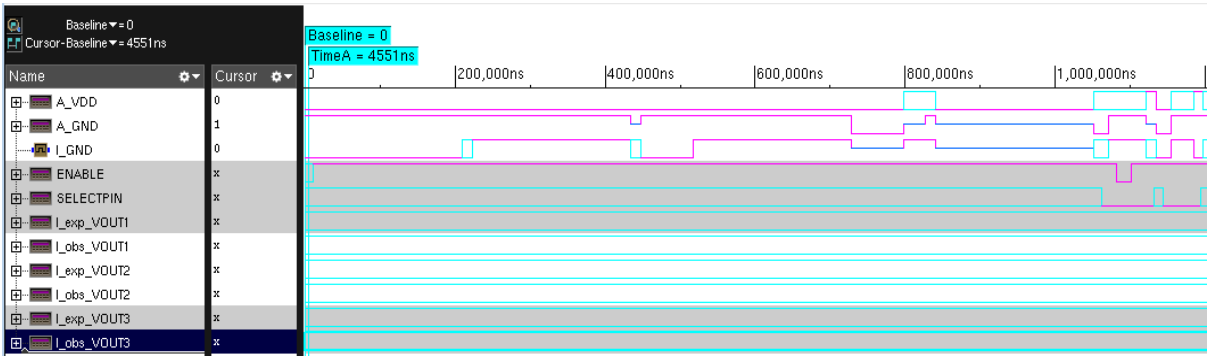

Figure 7.5 Simulation Result of LDO with all input supplies are invalid

From the above figure it's clearly reveals that when all the input supplies are invalid then outputs are corrupted. If Enable pin is logic low, then the LDO goes to power down mode. At time 4551ns when all the input supplies are invalid then all outputs are corrupted shown in figure.

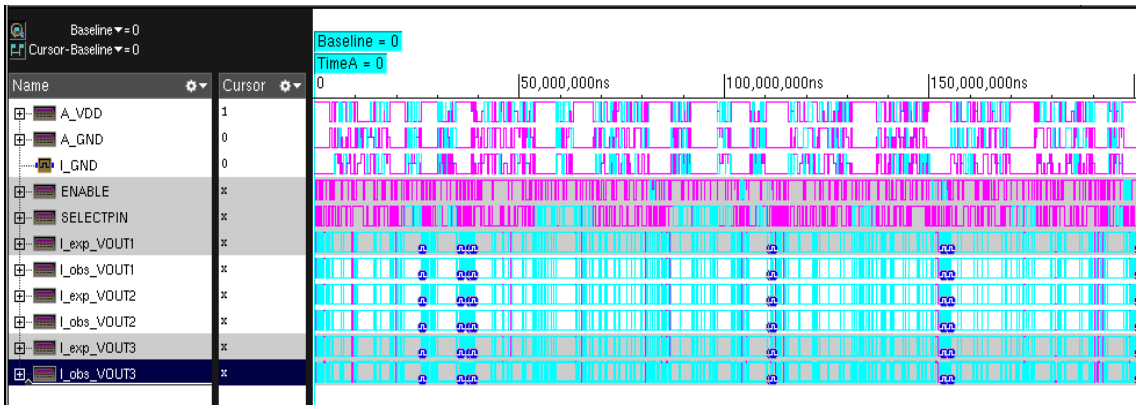

Figure 7.6 Simulation Result of LDO with all the combine random sequence

From the above figure it's clearly reveals that by simulating all the sequences combinely then the observed and expected outputs are matched. Total UVM errors are zero.

7.2 SIMULATION RESULTS OF LDO MODEL IN VCS TOOL

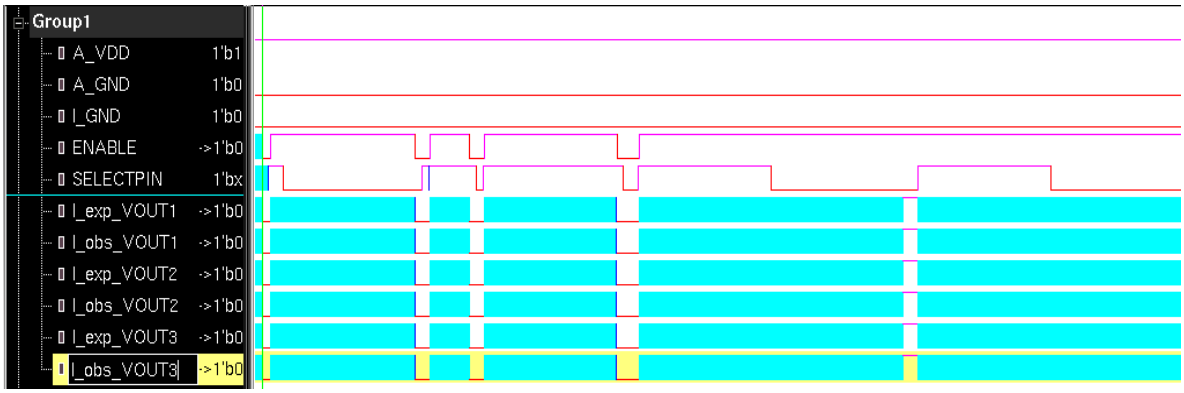

Figure 7.7 Simulation Result of LDO with all valid inputs

From the above figure it's clearly reveals that when all the inputs are valid supply then outputs are logic high after the start-up delay of LDO. If Enable pin is logic low, then the LDO goes to power down mode.

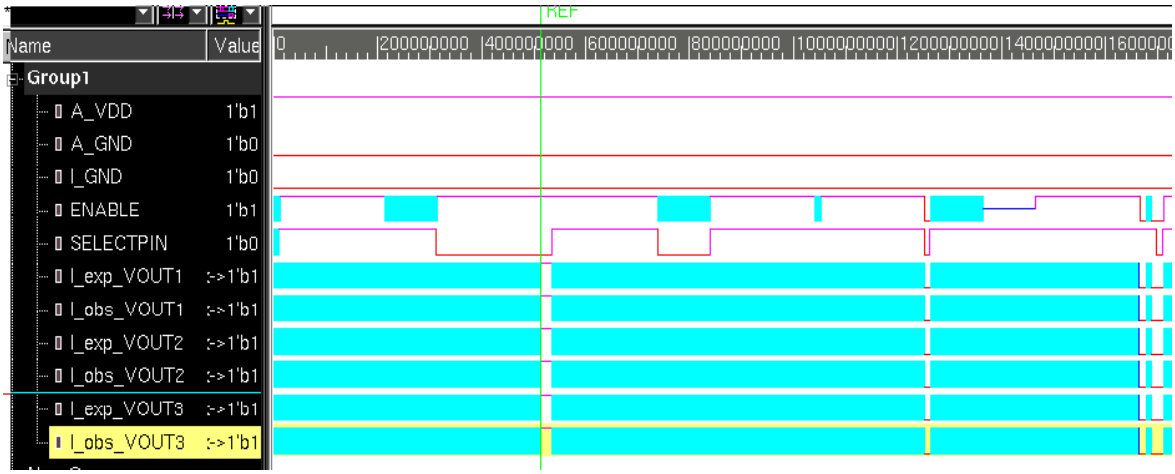

Figure 7.8 Simulation Result of LDO with ENABLE pin as invalid inputs

From the above figure it's clearly reveals that when all the inputs are valid supply and Enable pin is invalid then outputs are corrupted. If Enable pin is logic low, then the LDO goes to power down mode.

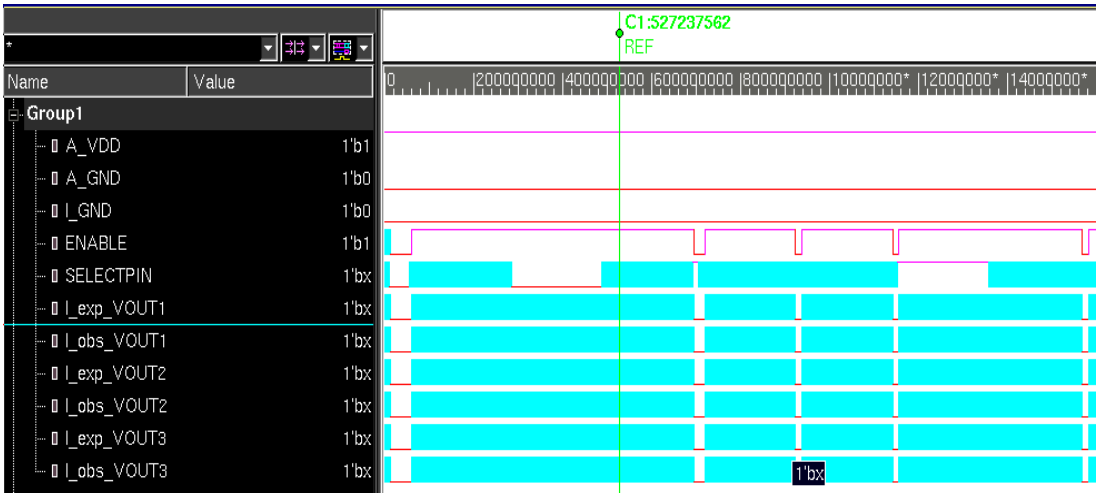

Figure 7.9 Simulation Result of LDO with SELECT pin as invalid inputs

From the above figure it's clearly reveals that when all the inputs are valid supply and SELECT pin is invalid then outputs are corrupted. If Enable pin is logic low, then the LDO goes to power down mode.

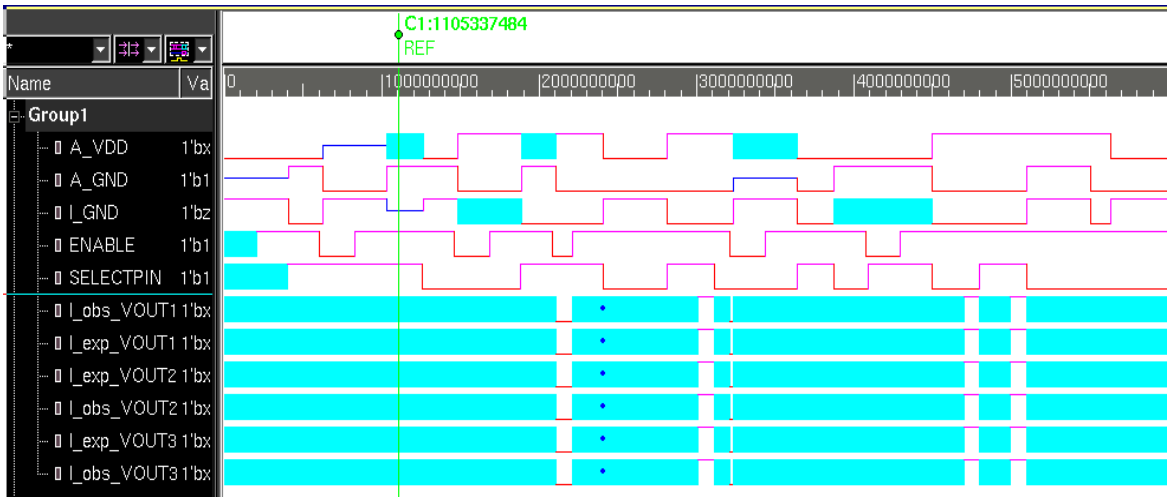

Figure 7.10 Simulation Result of LDO with all power supplies are invalid and other inputs are valid

From the above figure it's clearly reveals that when all the power supplies are invalid then outputs are corrupted. If Enable pin is logic low, then the LDO goes to power down mode.

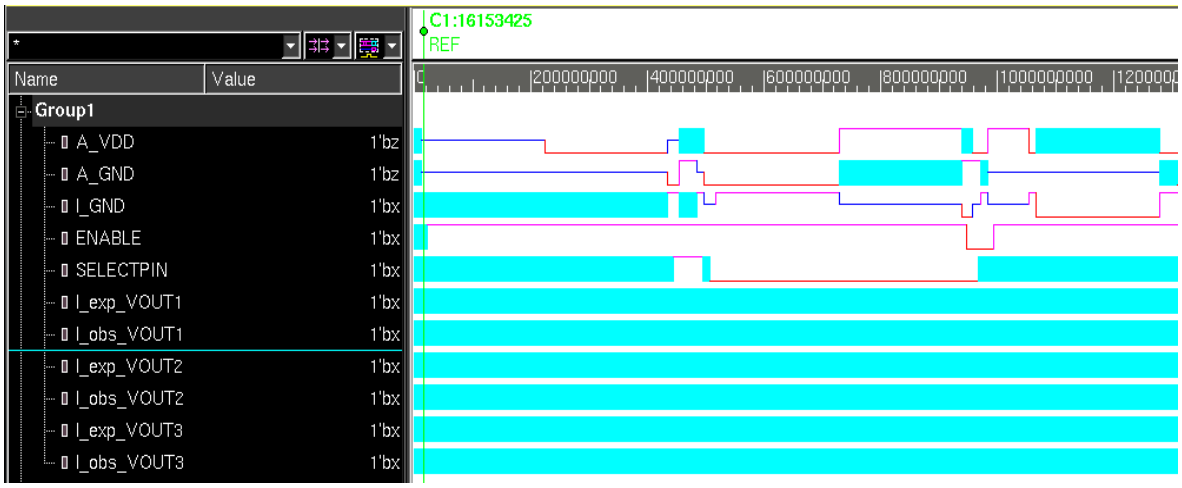

Figure 7.11 Simulation Result of LDO with all input supplies are invalid

From the above figure its clearly reveals that when all the input supplies are invalid then outputs are corrupted.If Enable pin is logic low then the LDO goes to power down mode.

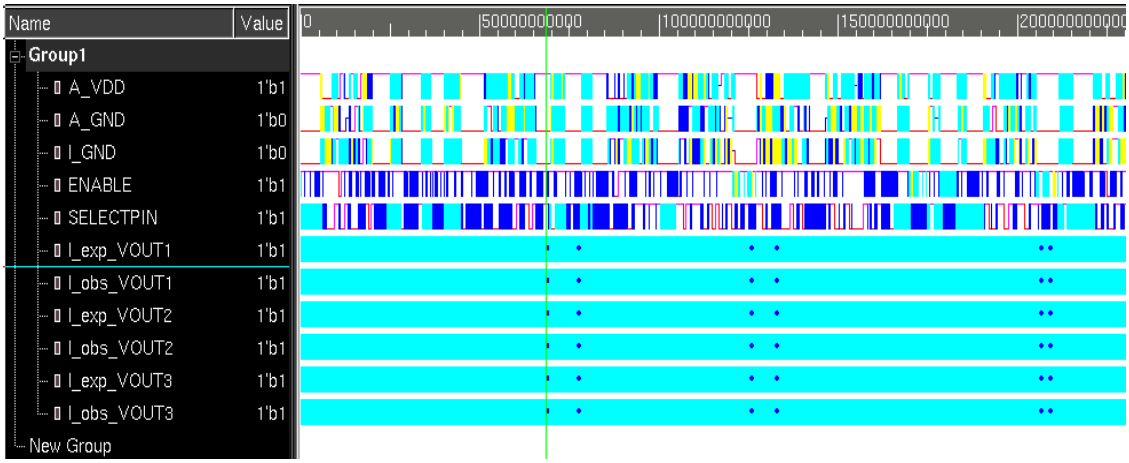

Figure 7.12 Simulation Result of LDO with all the combine random sequence

From the above figure its clearly reveals that by simulating all the sequences combinely then the observed and expected outputs are matched.Total UVM errors are zero.

7.3 SIMULATION RESULTS OF BANDGAP MODEL IN INCISIVE TOOL

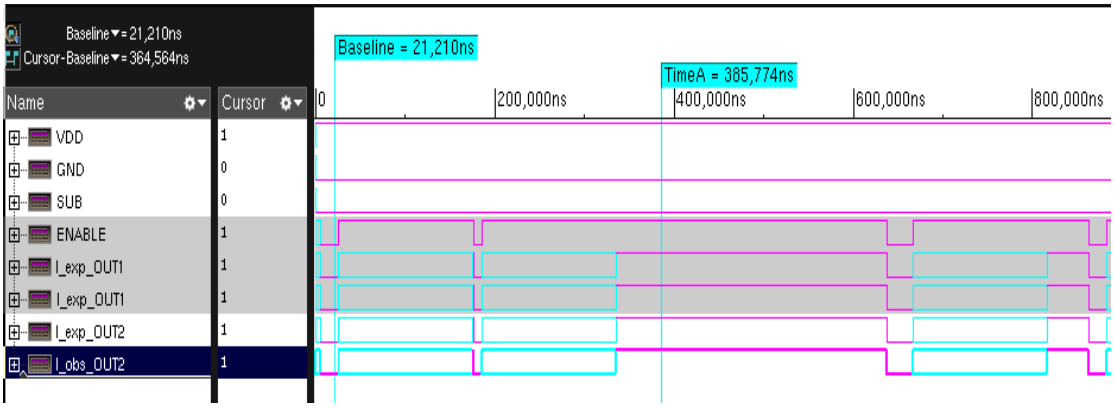

Figure 7.13 Simulation Result of BANDGAP with all valid inputs

From the above figure its clearly reveals that when all the inputs are valid supply then outputs are logic high after the start-up delay of BANDGAP.If Enable pin is logic low then the BANDGAP goes to power down mode .At time 385,774ns when all the power supplies are valid and Enable pin is logic high then outputs are valid shown in figure.

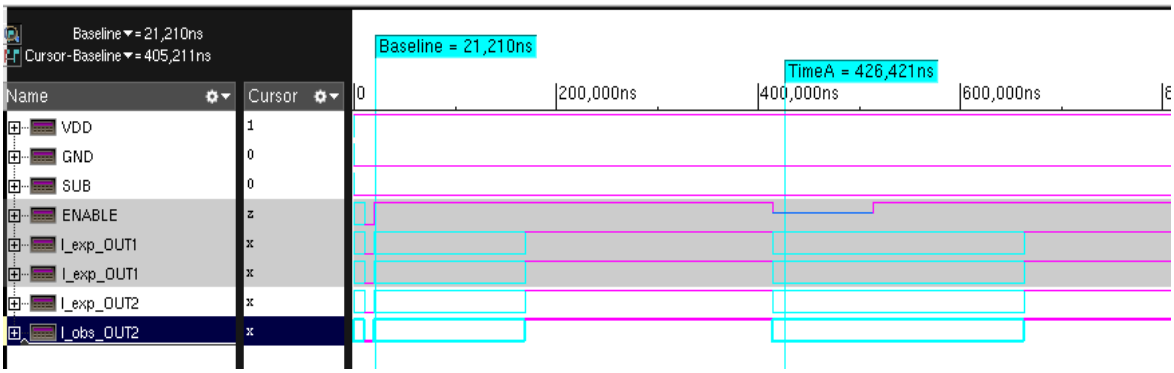

Figure 7.14 Simulation Result of BANDGAP with ENABLE pin as invalid inputs

From the above figure it's clearly reveals that when all the inputs are valid supply and Enable pin is invalid then outputs are corrupted. If Enable pin is logic low then the BANDGAP goes to power down mode. At time 426,421ns when all the power supplies are valid and Enable pin is invalid mode then all outputs are corrupted shown in figure.

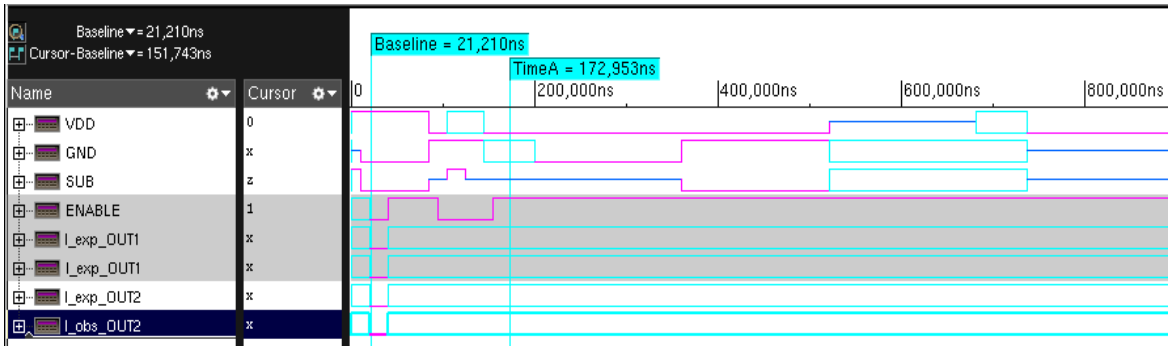

Figure 7.15 Simulation Result of BANDGAP with all power supplies are invalid and other inputs are valid

From the above figure its clearly reveals that when all the power supplies are invalid then outputs are corrupted.If Enable pin is logic low then the BANDGAP goes to power down mode. At time 172,953ns When all the power supplies are invalid then all outputs are corrupted shown in figure.

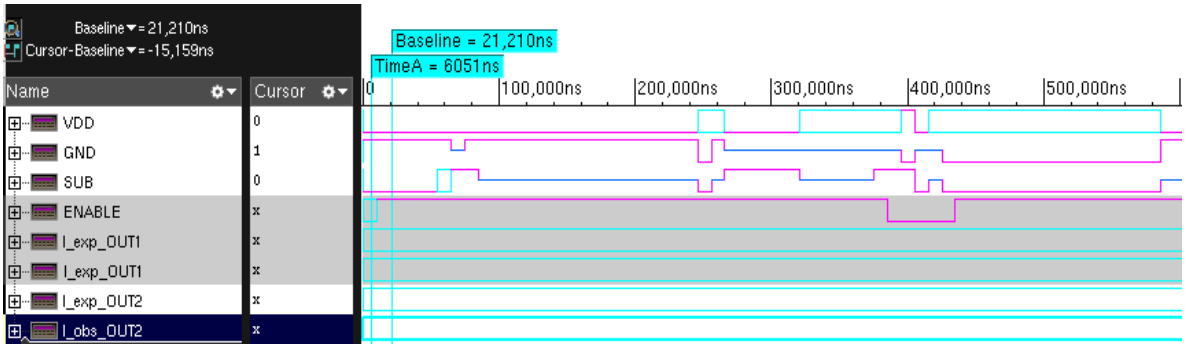

Figure 7.16 Simulation Result of BANDGAP with all input supplies are invalid

From the above figure its clearly reveals that when all the input supplies are invalid then outputs are corrupted. If Enable pin is logic low then the BANDGAP goes to power down mode. At time 6051ns When all the input supplies are invalid then all outputs are corrupted shown in figure.

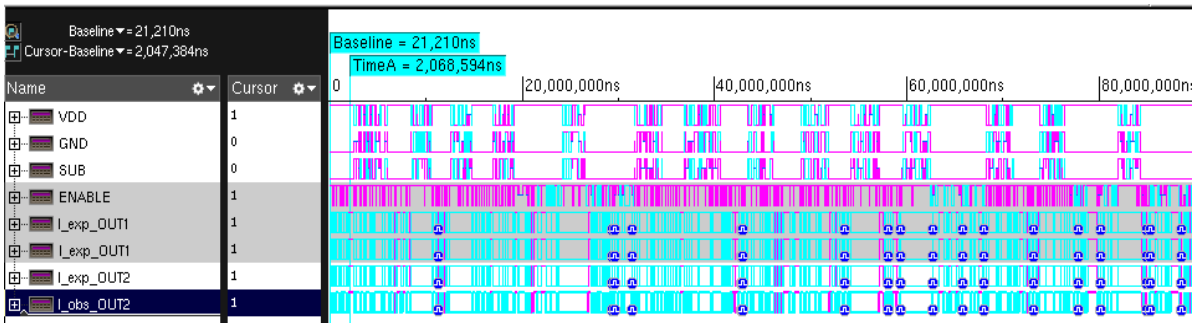

Figure 7.17 Simulation Result of BANDGAP with all the combine random sequence

From the above figure its clearly reveals that by simulating all the sequences combinely then the observed and expected outputs are matched.Total UVM errors are zero.

7.4 SIMULATION RESULTS OF BANDGAP MODEL IN VCS TOOL

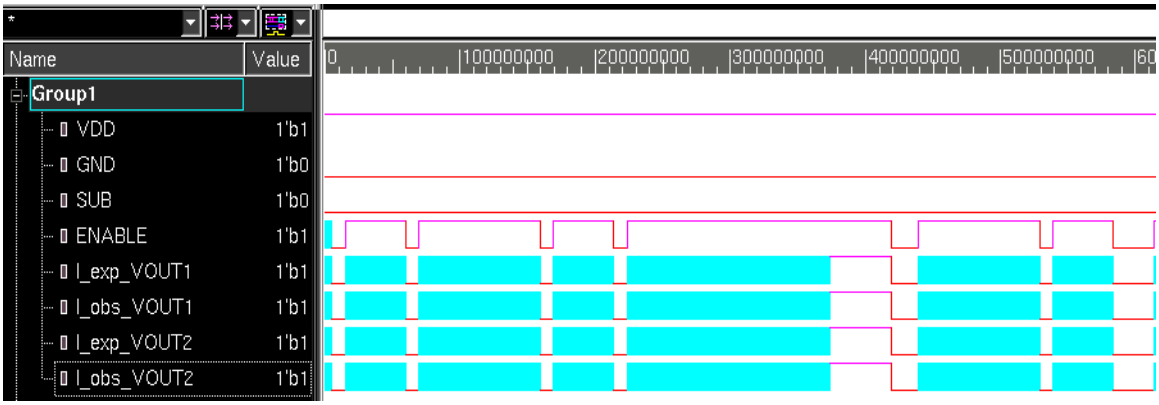

Figure 7.18 Simulation Result of BANDGAP with all valid inputs

From the above figure its clearly reveals that when all the inputs are valid supply then outputs are logic high after the start-up delay of BANDGAP.If Enable pin is logic low then the BANDGAP goes to power down mode.

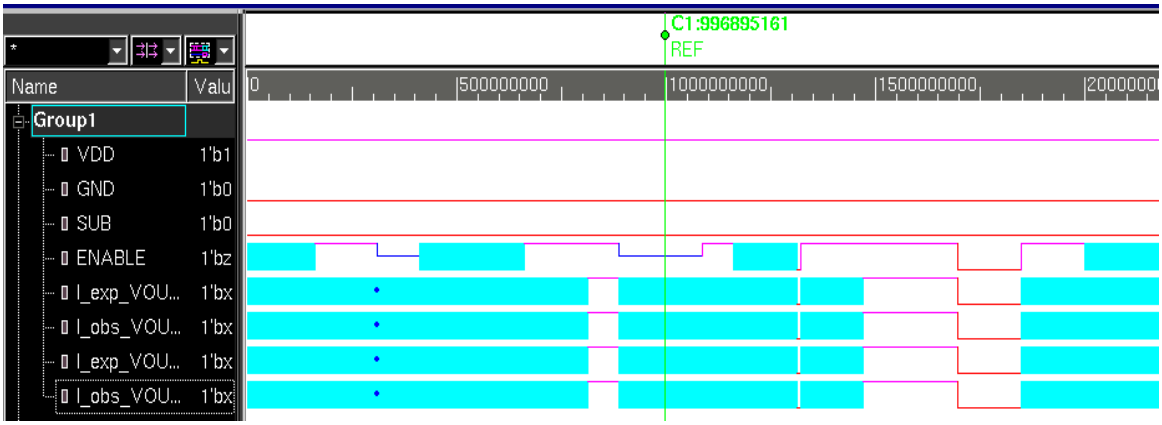

Figure 7.19 Simulation Result of BANDGAP with ENABLE pin as invalid inputs

From the above figure its clearly reveals that when all the inputs are valid supply and Enable pin is invalid then outputs are corrupted.If Enable pin is logic low then the BANDGAP goes to power down mode.

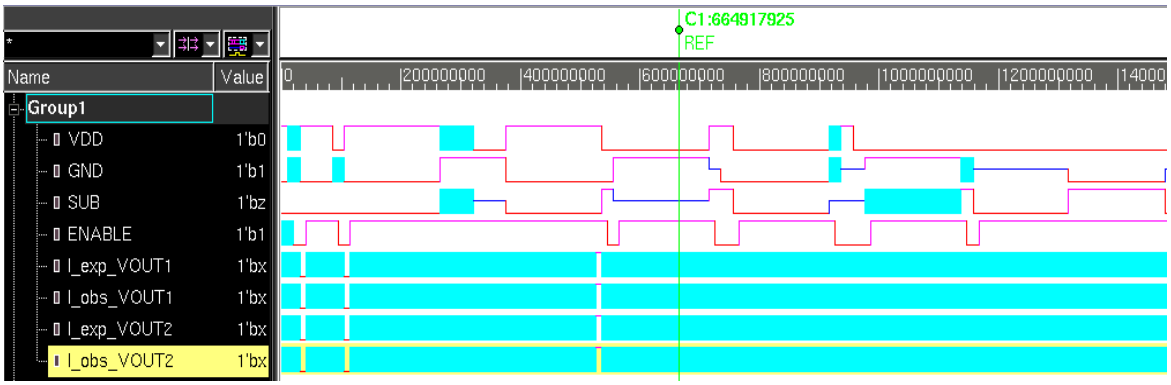

Figure 7.20 Simulation Result of BANDGAP with all power supplies are invalid and other inputs are valid

From the above figure it's clearly reveals that when all the power supplies are invalid then outputs are corrupted. If Enable pin is logic low, then the BANDGAP goes to power down mode.

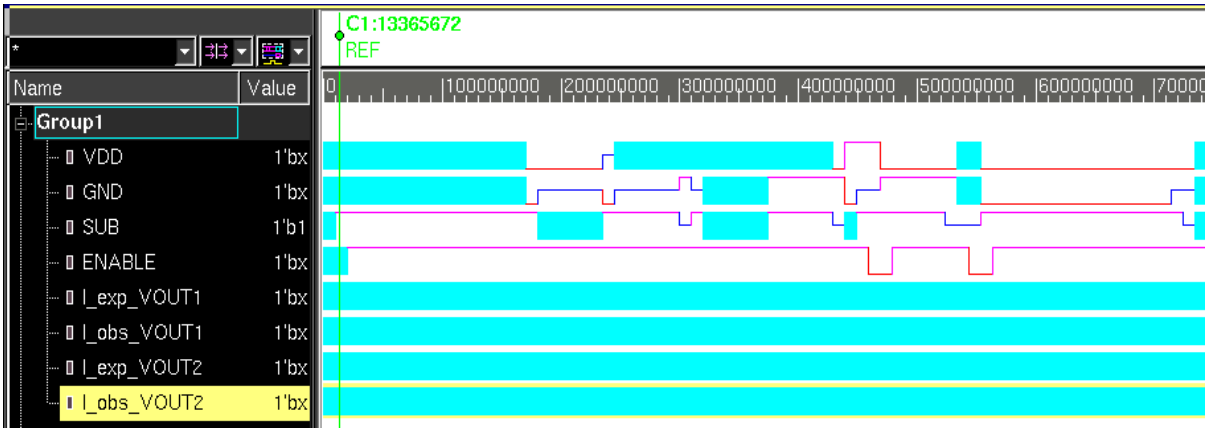

Figure 7.21 Simulation Result of BANDGAP with all input supplies are invalid

From the above figure it's clearly reveals that when all the input supplies are invalid then outputs are corrupted. If Enable pin is logic low, then the BANDGAP goes to power down mode.

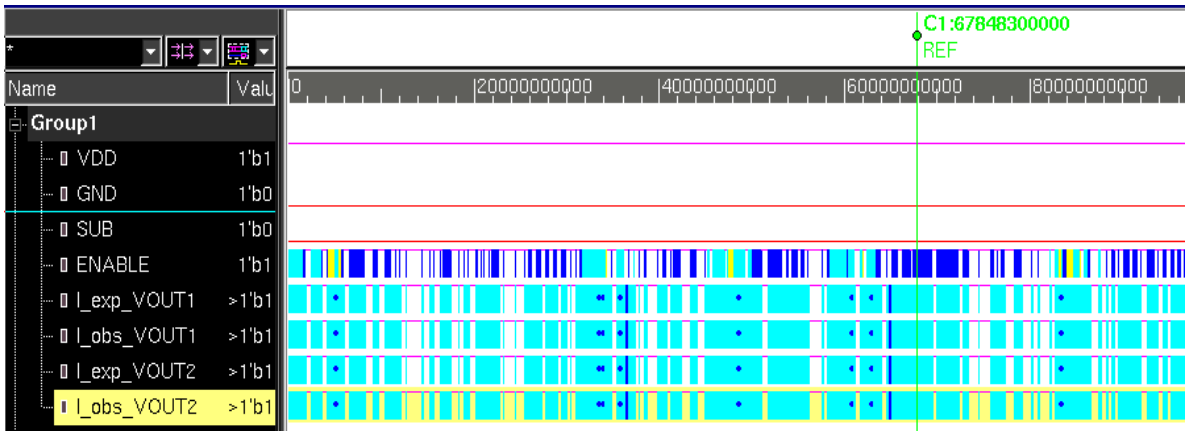

Figure 7.22 Simulation Result of BANDGAP with all the combine random sequence

From the above figure its clearly reveals that by simulating all the sequences combinely then the observed and expected outputs are matched. Total UVM errors are zero.

7.5 SIMULATION RESULTS OF VOLTAGE REGULATOR MODEL IN INCISIVE TOOL

Figure 7.23 Simulation result of Voltage Regulator Model with all valid random sequence

From the above figure it's clearly reveals that when all the supply pins are valid then only we check the status of others input pins.IN1 pin is logic zero or '?', whatever the status of others input pins then VOUT1 is high impedance state and VOUT2, VOUT3 are logic zero state. Any of the power supply pins are logic zero, x and? state then VOUT1 is 'z' state, VOUT2, VOUT3 are '0' state.

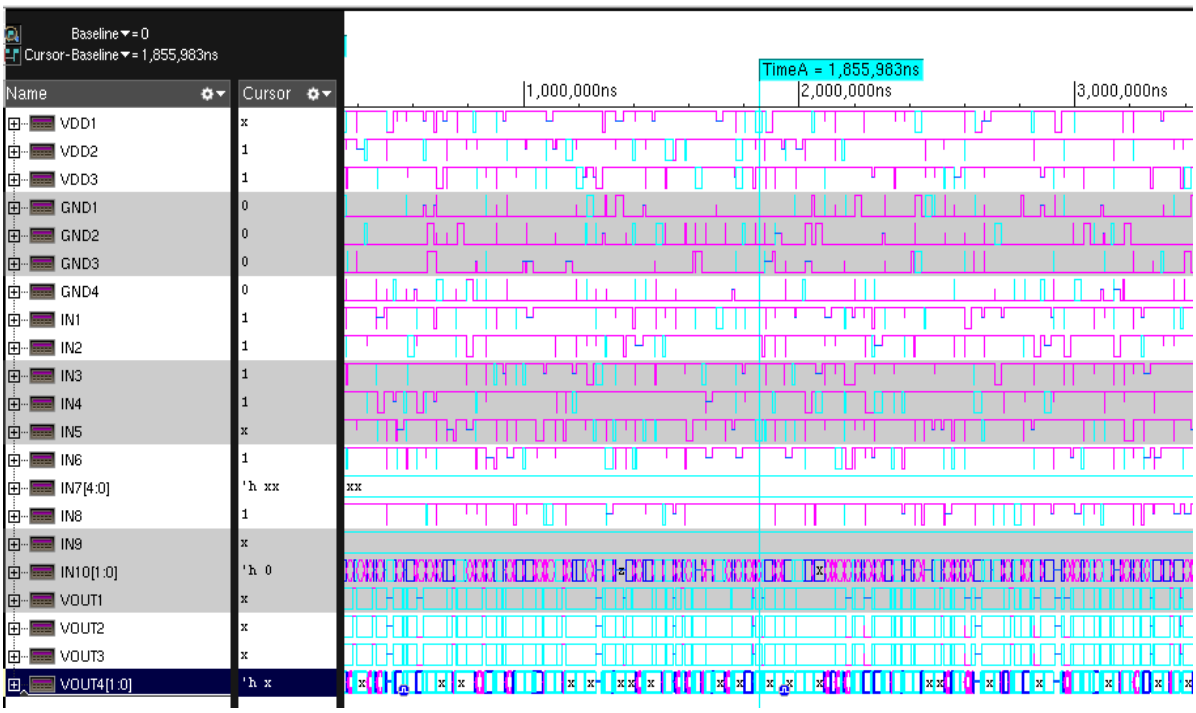

Figure 7.24 Simulation result of Voltage Regulator Model with all invalid random sequence

From the above figure it's clearly reveals when all the power supplies have invalid state then the outputs are in high impedance state.

7.6 SIMULATION RESULTS OF PLL MODEL IN INCISIVE TOOL

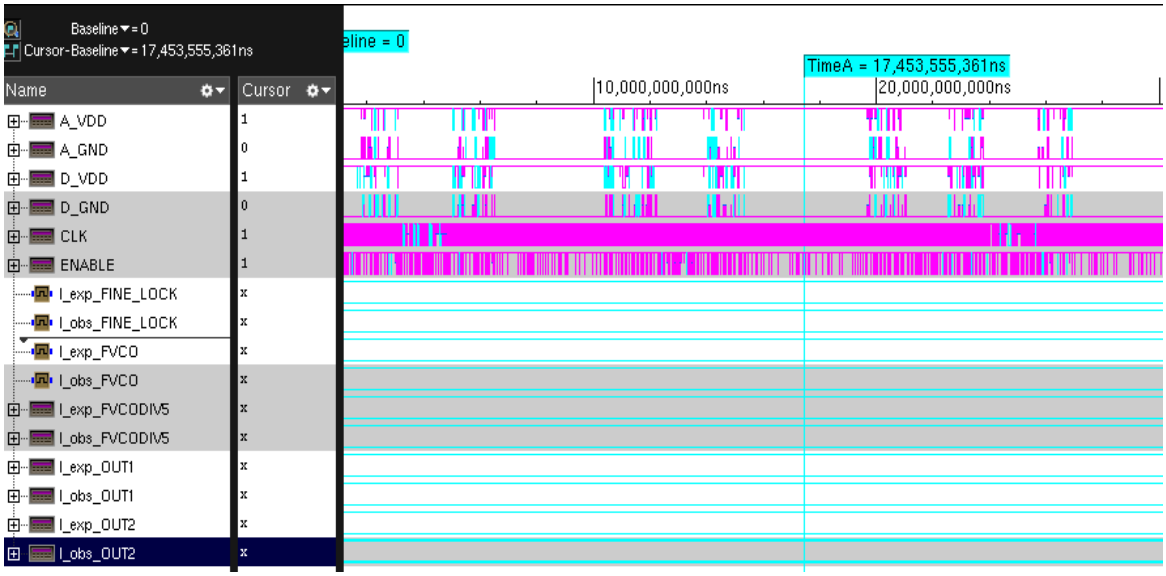

Figure 7.25 Simulation result of PLL Model with all random sequences

From the above figure it's clearly reveals that when all the Analog and Digital power supply pins are valid with valid CLK input. When Enable pin is logic high then with in 10ms FINE_LOCK is logic high. FVCO and FVCODIV5 outputs are gives the desired clock outputs.

CHAPTER 8

CONCLUSION AND FUTURE WORK

8.1 CONCLUSION

By the help of UVM methodology successfully verified the functionality of Analog and mixed signal IPs design and declare our design bug free. Constraint randomization concept of system Verilog helpful to verify the complex design. So, our model meets all the design specification written on the model spec. Functionality of LDO, VOLTAGE REGULATOR, BANDGAP, PLL model has verified successfully with SV-UVM Methodology. We conclude that UVM is one of the best methodologies provide by the industry. Self-checking Test-bench is reduced the effort of verification as compare to directed Test-Bench. System Verilog is the heart of the UVM Methodology . Language written in UVM is System Verilog, Which has more advance datatype and function to make the coding easier. Perl scripting is used to automate the various industry project. In my project Perl scripting is helpful to Automate the AMS IPs Verification environment. So, its reduce the verification time and deliver our project to the customer with in the time period.

8.2 FUTURE WORK

The UVM has become a de facto standard in today's functional verification of digital design, now it is extended into mixed signal design**.** One of our interests in the verification environment to introduce UVM and assertion-based verification to improve our methodology to verify the Analog and Mixed Signal IPs.This is left to the future investigation. Assertion Based Verification is one of the recommended verification techniques to enhance the verification quality and reduce the debugging time of analog and mixed signal IPs.

REFERENCES

[1] Francesconi, J., Rodriguez, J.A. and Julian, P.M., 2014, July. UVM based testbench architecture for unit verification. In 2014 Argentine Conference on Micro-Nanoelectronics, Technology and Applications (EAMTA) (pp. 89-94). IEEE.

[2] Liang, C., Fang, Z. and Chen, C.Z., 2015, March. Method for analog-mixed signal design verification and model calibration. In 2015 China Semiconductor Technology International Conference (pp. 1-4). IEEE.

[3] K. Kundert, "Principles of Top-Down Mixed-Signal Design," IEEE, 2003.

[4] Gurha, P. and Khandelwal, R.R., 2016, September. SystemVerilog Assertion Based Verification of AMBA-AHB. In 2016 International Conference on Micro-Electronics and Telecommunication Engineering (ICMETE) (pp. 641-645). IEEE.

[5] Chai, L., Xie, Z. and Wang, X.A., 2014, June. A verification methodology for reusable test cases and coverage based on system verilog. In 2014 IEEE International Conference on Electron Devices and Solid-State Circuits (pp. 1-2). IEEE.

[6] Liang, C., 2013, October. Mixed-signal verification methods for multi-power mixed-signal System-on-Chip (SoC) design. In 2013 IEEE 10th International Conference on ASIC (pp. 1-4). IEEE.

[7] Liang, C., Zhong, G., Huang, S. and Xia, B., 2014, October. UVM-AMS based sub-system verification of wireless power receiver SoC. In 2014 12th IEEE International Conference on Solid-State and Integrated Circuit Technology (ICSICT) (pp. 1-3). IEEE.

[8] Wong, W., Gao, X., Wang, Y. and Vishwanathan, S., 2004, October. Overview of mixed signal methodology for digital full-chip design/verification. In Proceedings. 7th International Conference on Solid-State and Integrated Circuits Technology, 2004. (Vol. 2, pp. 1421-1424). IEEE.

[9] B. Razavi, Design of Analog CMOS Integrated Circuits, Mc Graw Hill Education.

[10] CHRISSPEAR, SYSTEMVERILOG FOR VERIFICATION A Guide to Learning the Testbench Language Features, springer

[11] P. Araujo, UVM Guide for Beginners, Readlists.

[12] G.Bonfini and R.Mariani, "A Mixed-Signal Verification Kit for Verification of Analog-Digital Circuits," *IEEE*, 2006.

[13] Chia-minchen, "A Capacitor-Free CMOS Low-Dropout Voltage," *IEEE*, 2009.

[14] Motkurwar, S. and Ghodeswar, U., 2016, April. Design and power noise cancellation analysis of low dropout voltage regulator. In *2016 International Conference on Communication and Signal Processing (ICCSP)* (pp. 1282-1285). IEEE.

[15] Y.-J.-Y. Lee, "A Sub-threshold Ultra-Low Power Low-Dropout Voltage Regulator," *IEEE*, 2017.

[16] Y. Han and Qiz-Hang, "ANALYSIS OF THE INFLUENCE OF THE LOOP FILTER IN THE PHASE LOCKED LOOP ON THE OUTPUT PHASE NOISE," *IEEE*, 2019.

[17] L. xue, "Differential Charge Pump Circuit for High Speed PLL Aplication," *IEEE*, 2009.

APPENDIX(A)

PERL SCRIPTING TO AUTOMATE THE VERIFICATION OF AMS IPS

Perl scripting is used to automate the verification of UVM environment. Verification engineer write the generic script to made the verification faster and easier. According to the below code it's clearly reveals that user gives their choice of verification of AMS IPs, technology name , model name ,version name. After selection of version name a directory was created by the name of the user's Unix Id . Inside the User name directory regression file, testbench file, model file, test plan file ,doc file create automatically. Inside the testbench file all the UVM templates are created automatically in the name of TB_SEQ_ITEM.sv, TB_INTERFACE.sv , TB_SEQ_LIB.sv, TB_DRIVER.sv, TB_MONITOR.sv, TB_TEST_LIB.sv, TB_AGENT.sv, TB_ENV.sv, TB_SCOREBOARD.sv, TB_CLK_CHECKER.sv, TB_SUBSCRIBER.sv .

BASIC CODE :

```
#!usr/bin/perl
use Cwd;
our $OWNDIR=cwd;
unlink .dmgroup_activity.sh;
print ("WELCOME TO ANALOG_IPs VERIFICATION ENVIRONMENT \n");
#$a=common path of our project;
$a="/prj/femodel6/FEM_VERIF/ANALOG_IPs/";
print "$a\n";
#$d_list=list of all IPs inside ANALOG_IPs directory;
$d_list=`ls $a/;`;
@analogs_ips=split(/\s+/,$d_list);
$cnt=@analogs_ips;
for($i=0;$i<$cnt;$i++)
 { $c=$i;
 print ("$c @analogs_ips[$i]\n");
 }
START:
#selection of analog_ips according to user choice
print "Enter the choice of your above analog_ips--\n";
 $choice=<STDIN>;
```

```
 chomp($choice);
 if($choice<$cnt)
{
 print "$a@analogs_ips[$choice]/\n";
 $dir="$a@analogs_ips[$choice]/";
  }
 #choice of creation of new directory inside analog_ips directory
elsif($choice == $cnt) {
 print "write the name of new directory..in small letter like (abc)....\n";
$d =<STDIN>;
chomp($d);
print "welcome to dmgroup login :- to creat new directory..\n";
#supporting .dmgroup_activity1.sh file to create new directory inside analog_ips directory.
open (dmgrp_file_handler1 ,"> .dmgroup_activity1.sh") or die (" - Couldn't Create
.dmgroup_activity1.sh :$! ");
print dmgrp_file_handler1 "mkdir $dir/$d\n";
print dmgrp_file_handler1 "cd $dir/$d\n";
print dmgrp_file_handler1 ("chmod 755 $dir/$d\n");
print dmgrp_file_handler ("mkdir development production ; chmod development production;\n") ;
print dmgrp_file_handler1 ("exit\n");
close(dmgrp_file_handler1);
system ("chmod 755 .dmgroup_activity1.sh");
system ("su dmgroup -c /bin/sh -c .dmgroup_activity1.sh"); #dmgroup login to create new directory
system ("rm -rf .dmgroup_activity1.sh");
}
else {
print "Wrong ip( not exist ): \n";
goto START;
}
##-------------------development area------------------##
$d_list1=`ls $dir/;`;
@new_d=split(/\s+/,$d_list1);
$cnt1=@new_d;
for($j=0;$j<=$cnt1-2;$j++)
```

```
 { $b=$j;
 #print ("$b @new_d[$j]\n");print ("$dir@new_d[$j]/\n");
 $dir1="$dir@new_d[$j]/";
 }
#list of all model present inside the development area according to the choice of ANALOG_IPs;
$d_list2=`ls $dir1/;`;
@new_d1=split(/\s+/,$d_list2);
$cnt2=@new_d1;
for($k=0;$k<$cnt2;$k++)
 { $e=$k;
 print ("$e @new_d1[$k]\n");
 }
START2:
#choose your model according to your verification plan;
print "Enter the choice of your above direcotory--\n";
 $choice2=<STDIN>;
 chomp($choice2);
 if($choice2<$cnt2 ) {
 print "$dir1@new_d1[$choice2]/\n";
 $dir2="$dir1@new_d1[$choice2]/";
}
#create a new directory of "new model" inside the development area;
elsif  ( $choice2 == $cnt2  )                    {
print "....write the name of your new directory in CAPITAL letter like (ABC)....(including numeric
character also....)\n" ;
$f=<STDIN>;
chomp($f);
print "welcome to dmgroup login :- to creat new directory..\n";
#supporting .dmgroup_activity2.sh file to create new directory inside development area;
open (dmgrp_file_handler2 ,"> .dmgroup_activity2.sh") or die (" - Couldn't Create
.dmgroup_activity2.sh :$! ");
print dmgrp_file_handler2 "mkdir $dir2/$f\n";
print dmgrp_file_handler2 "cd $dir2/$f\n";
print dmgrp_file_handler2 ("chmod 755 $dir2/$f\n");
```

```
print dmgrp_file_handler2 ("exit\n");
close(dmgrp_file_handler2);
system ("chmod 755 .dmgroup_activity2.sh");
system ("su dmgroup -c /bin/sh -c .dmgroup_activity2.sh"); #dmgroup login to create new directory.
system ("rm -rf .dmgroup_activity2.sh");
}
else
{
print("Are you sleepy ? directory will not creat:\n");
goto START2;
}
# Below code describe about the incremental version of model choose by the user.
# Previously v1.0 present inside a model then automatically v1.1 will be create.
# If no version present intially then v1.0 will automatically create.
$in = $dir2;
$in_len=length($in);
$v="*";
$path=$in.$v;
@list= glob($path);
$v1="V1.0";
$coun_t=0;
for($i=0;$i<=$#list;$i++)
{
        if(-d $list[$i])
        {
                @dir_names[$coun_t]=$list[$i];
                $coun_t++;
        }
}
@sorted_dirs=sort(@dir_names);
$count_sorted=0;
foreach(@sorted_dirs)
{
        $sot_cond=$sorted_dirs[$count_sorted];
```

```
        @pvts[$count_sorted]=substr($sot_cond,$in_len);
        $count_sorted++;
}
$last_dir=pop @pvts;

if ($last_dir=~/(\S+\.)(\d+)/)
{
$name=$1;
$no=$2;
$new_no=$no+1;
$new_name=$name.$new_no;
$full_path=$in.$new_name;
print("$full_path\n");
}
else
{
  $full_path=$in.$v1;
#Full path of directories save by the user upto incremental version of model.
#like:- /prj/femodel6/FEM_VERIF/ANALOG_IPs/IP/development/model/v1.1/ (previously v1.0
present).
  print("$full_path\n");
}
#directory will create according to user name.
$dir_1= $full_path;
$usr_name= `whoami`;chomp($usr_name);
print" create a new directory:-$usr_name \n";
print "Enter the model name going to verify :";
$module=<STDIN>;
chomp($module);

#".dmgroup_activity.sh" supporting file to create various subdirectories && files inside "user_name"
directory.
open (dmgrp_file_handler ,"> .dmgroup_activity.sh") or die (" - Couldn't Create .dmgroup_activity.sh
:$! ");
```

```
print dmgrp_file_handler ("mkdir $full_path\n");
print dmgrp_file_handler ("cd $full_path\n");
print dmgrp_file_handler ("chmod 755 $full_path\n");
print dmgrp_file_handler   "mkdir $usr_name\n";
print dmgrp_file_handler "cd $usr_name\n" ;
print dmgrp_file_handler ("mkdir reg tb model doc testplan ; chmod 755 reg tb model doc
testplan;\n")  ;
print dmgrp_file_handler " cd tb \n ";
#print dmgrp_file_handler    ("touch driver.sv sequence.sv monitor.sv agent.sv scoreboard.sv \n") ;
print dmgrp_file_handler (" cd $full_path/$usr_name/reg \n ");
print dmgrp_file_handler    ("touch file1.sh \n") ;
print dmgrp_file_handler (" cd $full_path/$usr_name/model \n ");
print dmgrp_file_handler    ("touch dut.v \n") ;
print dmgrp_file_handler (" cd $full_path/$usr_name/doc \n ");
print dmgrp_file_handler ("touch file.pdf \n");
print dmgrp_file_handler (" cd $full_path/$usr_name/testplan \n ");
print dmgrp_file_handler    ("touch file.xlsx \n") ;
print dmgrp_file_handler ("cp -rf $OWNDIR/$TB\t$full_path/$usr_name/tb \n ");
print dmgrp_file_handler ("cp -rf $OWNDIR/$TB1\t$full_path/$usr_name/tb \n ");
print dmgrp_file_handler ("cp -rf $OWNDIR/$TB2\t$full_path/$usr_name/tb \n ");
print dmgrp_file_handler ("cp -rf $OWNDIR/$TB3\t$full_path/$usr_name/tb \n ");
print dmgrp_file_handler ("cp -rf $OWNDIR/$TB4\t$full_path/$usr_name/tb \n ");
print dmgrp_file_handler ("cp -rf $OWNDIR/$TB5\t$full_path/$usr_name/tb \n ");
print dmgrp_file_handler ("cp -rf $OWNDIR/$TB6\t$full_path/$usr_name/tb \n ");
print dmgrp_file_handler ("cp -rf $OWNDIR/$TB7\t$full_path/$usr_name/tb \n ");
print dmgrp_file_handler ("cp -rf $OWNDIR/$TB8\t$full_path/$usr_name/tb \n ");
print dmgrp_file_handler ("cp -rf $OWNDIR/$TB9\t$full_path/$usr_name/tb \n ");
print dmgrp_file_handler ("cp -rf $OWNDIR/$TB10\t$full_path/$usr_name/tb \n ");
print dmgrp_file_handler ("cp -rf $OWNDIR/$TB11\t$full_path/$usr_name/tb \n ");
print dmgrp_file_handler ("cp -rf $OWNDIR/$TB12\t$full_path/$usr_name/tb \n ");
print dmgrp_file_handler ("cp -rf $OWNDIR/$TB13\t$full_path/$usr_name/tb \n ");
print dmgrp_file_handler ("cp -rf $OWNDIR/$TB14\t$full_path/$usr_name/tb \n ");
print dmgrp_file_handler ("cp -rf $OWNDIR/$TB15\t$full_path/$usr_name/tb \n ");
print dmgrp_file_handler ("cp -rf $OWNDIR/$TB16\t$full_path/$usr_name/tb \n ");
```

```
print dmgrp_file_handler ("exit\n");
close(dmgrp_file_handler);
system ("chmod 755 .dmgroup_activity.sh");
system ("su dmgroup -c /bin/sh -c .dmgroup_activity.sh"); #dmgroup login to create subdirectory.
#system ("rm -rf .dmgroup_activity.sh");
```