# Design Verification of Power Management Scenarios in X86 Based SOC

Andluru Prathyusha Reddy
M.Tech in Embedded Systems,
JNTU College of Engineering Ananthapuramu, Andhra Pradesh, India

***Abstract:***

Power Management has fast become specialized in different spaces. There are different schools of thought and methodologies of verification, depending on whether you're looking at server, desktop, mobile or ultra-mobile spaces. For pre-silicon verification, the challenges are unique, and numerous tools and methodologies are used to ensure that as many bugs are found in pre-silicon. This paper outlines the verification of power management techniques such as clock gating and power gating in X86 based SOC which are equally challenging to produce very healthy and working silicon.

***IndexTerms* – Clock gating, Power gating, ACPI P-states and C-states verification**

## I. INTRODUCTION

Functional verification is a necessary step in the development of today's complex designs. Hardware complexity growth continues to follow Moore's law (Moore, 1965), but verification complexity is even more challenging. In fact, it theoretically rises exponentially with hardware complexity doubling exponentially with time (Dempster and Stuart, 2001). Functional verification is widely acknowledged as a major bottleneck in design methodology: up to 70% of the design development time and resources are spent of Functional verification.

Power Management Verification is the pre-silicon functional verification of these power management features, together with its impact on other parts and functions of the chip and platform. It involves testing the algorithms for functional correctness, attempting to emulate the low power states in a simulation environment and testing for correct power down and power up sequencing. It, however, is not responsible for estimating power numbers of blocks, nor does it analyze performance impacts … yet. As more techniques are used to save power and more low power emulation features are made available in EDA tools, the strategies for verifying power management logic change over time. The evolution of the tools requires verification collateral to keep up but allows the engineer to focus more on generating the right content, which improves the efficiency of the process. More complex designs are thus verifiable with smaller teams.

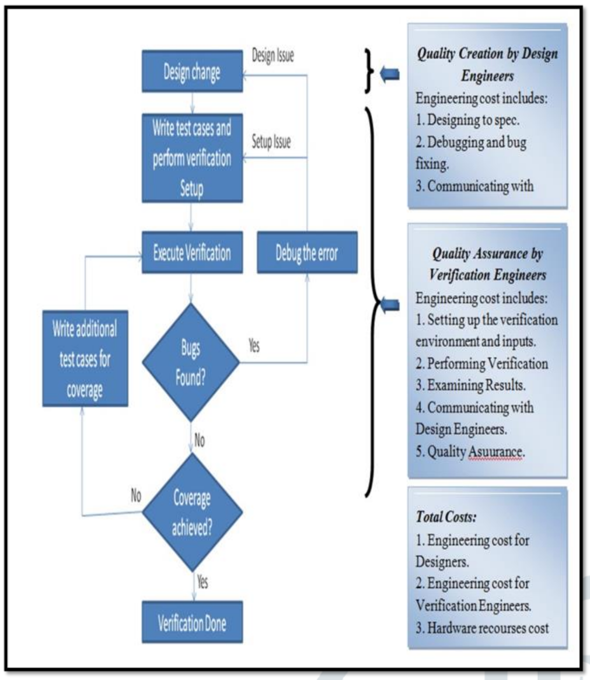
Design change
Design Issue
Write test cases and perform verification Setup
Setup Issue
Execute Verification
Debug the error
Write additional test cases for coverage
Bugs Found?
Yes
No
Coverage achieved?
No
Yes
Verification Done
Quality Creation by Design Engineers
Engineering cost includes:
1. Designing to spec.
2. Debugging and bug fixing.
3. Communicating with
Quality Assurance by Verification Engineers
Engineering cost includes:
1. Setting up the verification environment and inputs.
2. Performing Verification
3. Examining Results.
4. Communicating with Design Engineers.
5. Quality Asuurance.
Total Costs:
1. Engineering cost for Designers.
2. Engineering cost for Verification Engineers.
3. Hardware recourses cost

Fig.1. Typical Design and Verification Flow

## II. BASIC IDEA OF CLOCK GATING AND POWER GATING

**Dynamic Power reduction:** The dynamic power (switching power) dissipated per unit of time by a chip is $C \cdot V^2 \cdot A \cdot f$, where C is the capacitance being switched per clock cycle, V is voltage, A is the Activity Factor indicating the average number of switching events underwent by the transistors in the chip and f is the switching frequency (as a unit less quantity). The voltage required for stable operation is determined by the frequency at which the circuit is clocked. All the performance states are under C0 state.

In C1 State, all the Cores are halted, and Core's clock can be gated for some time. If any interrupt comes, they can come back to the normal working state quickly.

**Dynamic Power reduction by Clock Gating:** Clock gating is a popular technique used in many synchronous circuits for reducing dynamic power dissipation. Clock gating saves power by adding more logic to a circuit to prune the clock tree. Clock signals are stopped for selected register banks during times when the stored logic values are not changing. Challenges to use are 1) finding the best places to use it and 2) creating the logic to shut off and turn on the clock at the proper times.

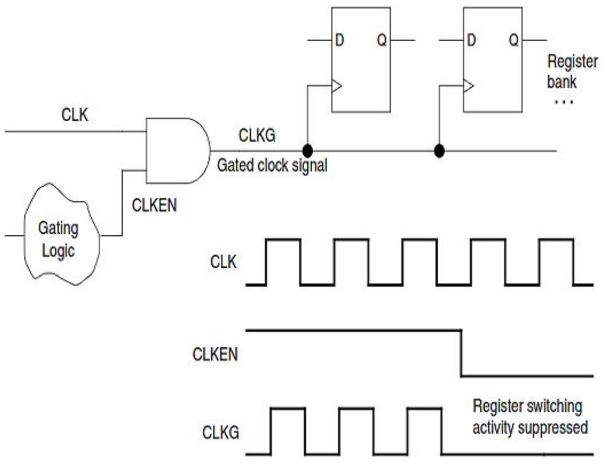

Fig.2. Example for Clock gating

C6 State is the more power saving state, where core's clock is gated and Core's power also gated.

**Static Power Reduction by Power gating:** Power gating is a technique used in integrated circuit design to reduce power consumption, by shutting off the current to blocks of the circuit that are not in use. Power gating affects design architecture more than clock gating. It increases time delays, as power gated modes must be safely entered and exited. Architectural trade-offs exist between designing for leakage power saving in low power modes and the energy dissipation to enter and exit the low power modes.

There are transistors connected from supply to the actual logic. These transistors are known as header and footer cells. We can use either header or footer cells in our design (Fig.2). Header cells are PMOS cells which are connected from VDD to the logic circuit. Footer cells are NMOS cells which are connected from logic circuit to the VSS. Header or Footer are controlled with either a Zero or negative VGS to cut off the cells from VDD and VSS. Often, multiple Header/Footer transistors are used in parallel, to ensure there is enough current delivery strength and capacity to withstand fluctuations in current [2].

These are high Vt cells to reduce the leakage. Even though power gating is done some leakage power is present due to usage of MOS switches. These switches are not idle so some leakage is present.

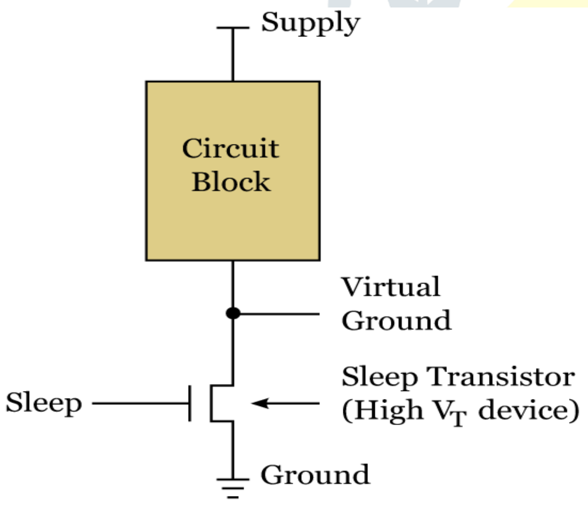

Fig.3.Example for Power gating

**Use of Isolation Cells:** Any use of power switching (Power shutting) requires isolation cells where signals leave a powered-down block and enter a block that is always on (or currently powered up). An isolation cell provides a known, constant logic value to an always-on block when the power-down block has no power, thereby preventing unknown or intermediate values that could cause crowbar currents.

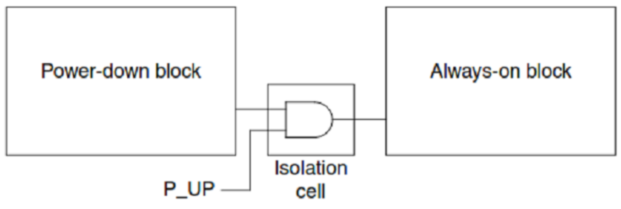

Fig.4. Isolation Cell example

## III. ABOUT CPU P-STATES AND C-STATES

The ACPI standard defines multiple "C-States" which are normal execution and lower-power modes for CPUs. C0 is the normal operating mode; all other C-states are low-power modes in which instruction execution is halted. Even though ACPI defined C-States are from C0-C3, it is not necessary to implement all the states at the hardware level. Hardware designers can implement their own industry specific C-states. Currently, industries are using more efficient CPU Power states are C0, C1 and C6.

The "P-States" are operating voltages/frequencies within C0 (the normal operating mode C-State). The ACPI standard defines P-States as well as C-States; P0 is the highest power, highest performance state; P7 is the lowest. A P-State is selected by the operating system on a per-CPU basis. ACPI based CPU P-states/C-states transits unused devices into lower power consumption states including placing the entire system in a low-power state (sleeping state) when possible.

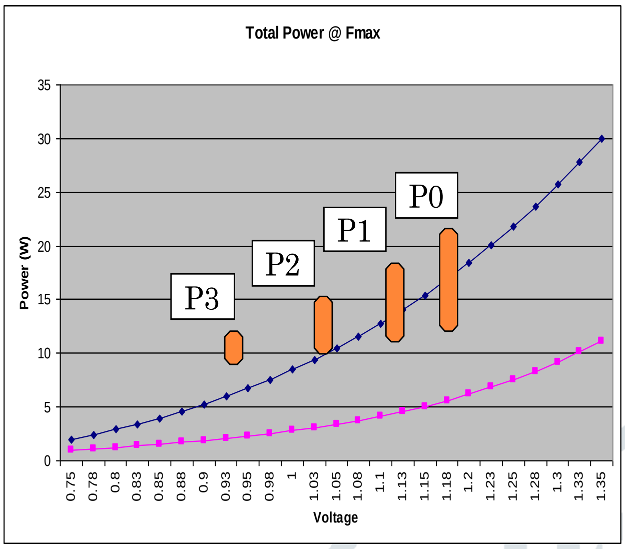

Fig.5.P-state change flow

## IV. CORE BASIC P-STATE TRANSITION

Objective of this test is to verify basic CPU P-state transition. Generated a random P-state between 0 to 7. Placed random request for P-state change in the request register.

Waits for control register monitor which initiates P-state change. Compare current P-state with the requested P-state value and trigger P-state change flow if required. Wait for P-state change i.e., update status register. Compare the requested P-state and updated status register for verifying the expected P-state change is done are not.

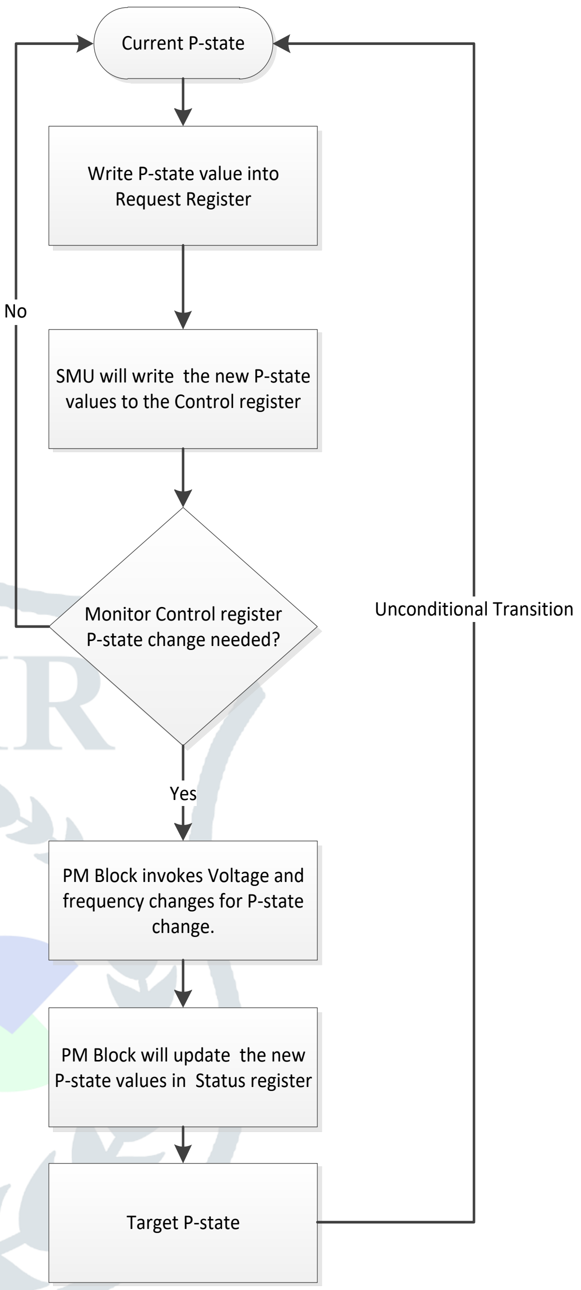
Current P-state
Write P-state value into Request Register
SMU will write the new P-state values to the Control register
Monitor Control register P-state change needed?
No
Yes
PM Block invokes Voltage and frequency changes for P-state change.
PM Block will update the new P-state values in Status register
Target P-state
Unconditional Transition

Fig.6. Pstate_basic_trans flow chart

## V. CORE C1 ENTRY AND EXIT PROCESS

Objective of this test is to verify basic entry and exit sequence for C1. Enter Core C1 state by executing Halt instruction. Wait for CC1 counter timeout signal. Then request for clock reduction to clocking block if there is no Intr Pending. Exit from Core C1 State by wake up event NMI Interrupt. Then request for clock enhancement to clocking block. Performance counter guarantees that Core entered C1 state.

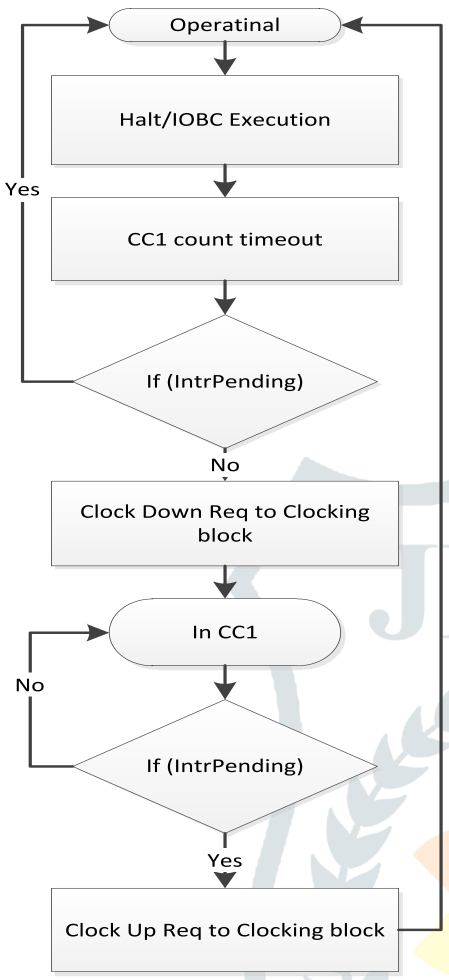

Fig.7.C1 Entry-Exit flow chart

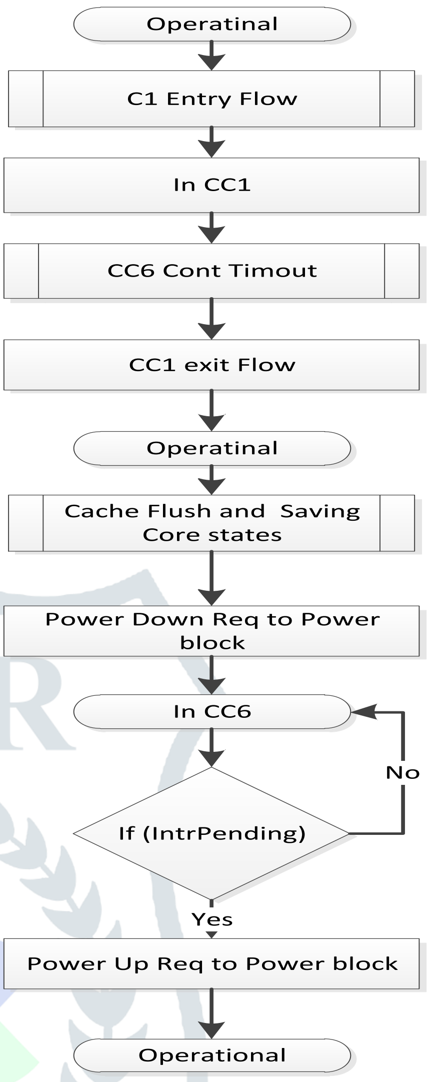

Fig.8. C6 Entry – Exit flow chart

## VI. CORE C6 ENTRY AND EXIT PROCESS

When a halt occurs, before entering to power gated state i.e. c6, the core must be clock gated i.e. core c1 state. To see whether the core may enter into c6 or not there are many algorithms used which may involve different monitors and timers. Based on these algorithms, the decision is taken whether the core should enter power gating state or not.

In c6 entry process first the core architectural states are saved into next level of cache, here it is l3 cache. Then the caches are flushed that means the modified lines are invalidated, the cache flush can take variable time according to number of modified lines present in the cache. The cache flush stage is interruptible stage [4].

Then the power down sequence starts: the system is put into the reset, then the isolations are turned on, then the power to the core is turned off through the switches now the core is in c6 state.

Now when any wake-up event happens than the power up sequence starts which is exactly reverse of the power down sequence, then the architectural state of the core is restored. So that the core starts the execution from the place it got halted.

## VII. POWER MANGEMENT VERIFICATION & RESULTS

There are aspects of power management that can be validated at RTL. How much can be validated at RTL also depends upon design methodology being used by a design team.

We can find instantiation of power management cells such as isolation, retention, switches, and level shifter at RTL in more power management aware design team where as some team may bring these elements late in the flow.

In some cases, we can even find power and ground connectivity in the power management elements at RTL itself. Lack of tool support have taken design teams though different routes and that's what we see today; but there are common grounds resulting in common needs. A generic verification infrastructure should be able to help verify most of these variations.

### A. Simulation

RTL verification should be Power Management IC (PMIC) aware in the context of power managed designs. A model for PMIC along with support for simulation features incorporating effects of voltage changes such as power down is required. PMICs typically implement voltage variation in steps indicated by a value through the bus; this can be leveraged in the context of simulation-based verification flow. Simulation can help validate the following key aspects of power management architecture:

- Validate chip functionality in these new power states of the system.
- Validate sequencing among these power states i.e., you will typically power off in a sequence from a given active state and this aspect of power controller should be validate through test cases.
- Power state coverage indication; an additional coverage measure.
- Assertions about power management architecture can be automatically generated from a UPF like description and validated during the course of simulation. Some specific assertion about special power management signals can be written with assertion languages.
- A common recurring problem with respect to Power Gating is that of a proper reset occurring upon power up at the end of a sleep state. This can be validated with a simulator supporting the power down feature.
- Validate retention under power down condition when SRPG and S&RPG techniques are used.
- Improper isolation can be flagged by the simulator; it will show up in the simulator supporting a power down feature.
- Voltage variation related issues lend themselves better for a gate-level simulator solution assuming the design methodologies continue to prefer higher level of description for faster speeds;

- Clock gating happens before power gating i.e., not wasting clock power during power down.
- With power management, what you have here is an added dimension to already exploding verification state space; efficiency and efficacy should be looked into made part of the overall solution.

**B. Formal**

Formal tools can be leveraged to validate some aspects of power management architecture at RTL. Both assertion-based and equivalence checking tools can play useful roles here utilizing UPF like description of power architecture as an input.

- There can be a hierarchical relationship among the power sequencing of islands. In a simpler case, situations such as when an island A is off then island B is also off but when A is on then B could be off. This directly implies a logical constraint on the switch enable signals for island A and island B that can automatically generated as an assertion and validated.
- State retention signal will have a relationship with power gating enable.

These are all valid examples of power management assertion in Verification.

The simulation results of my project are shown below.

Initial P-state is 4 and we requested P-state is 3 so final updated P-state is 3 now. This is shown in figure below.

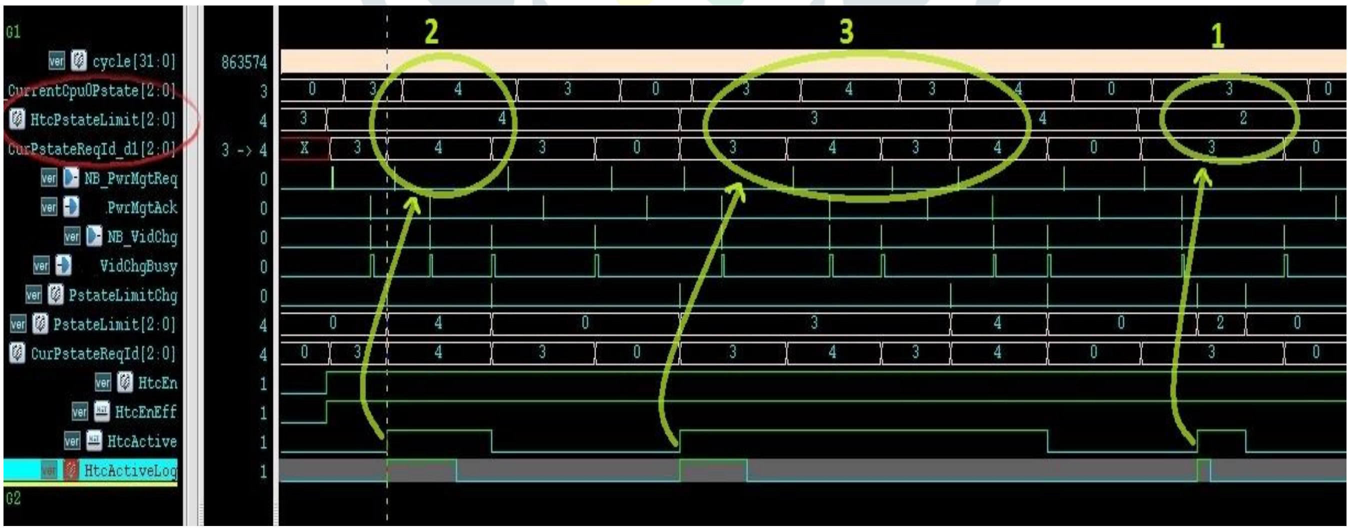

Fig.9. CPU P-State_basic_trans waveform

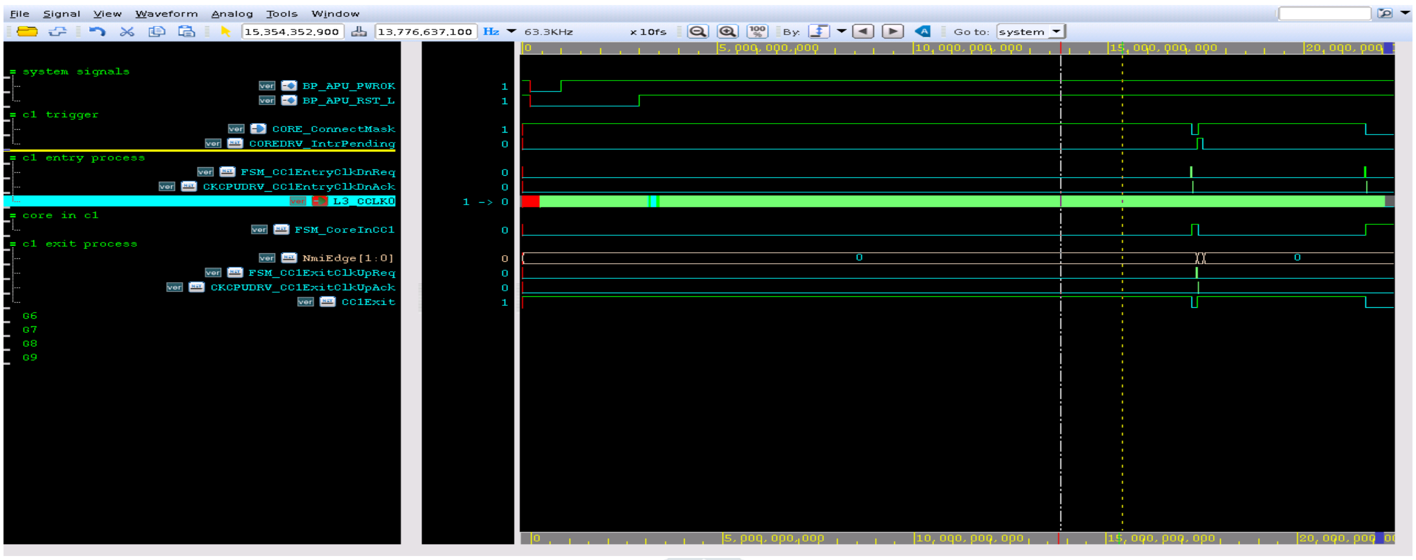

Fig.10. C1 Entry-Exit waveform

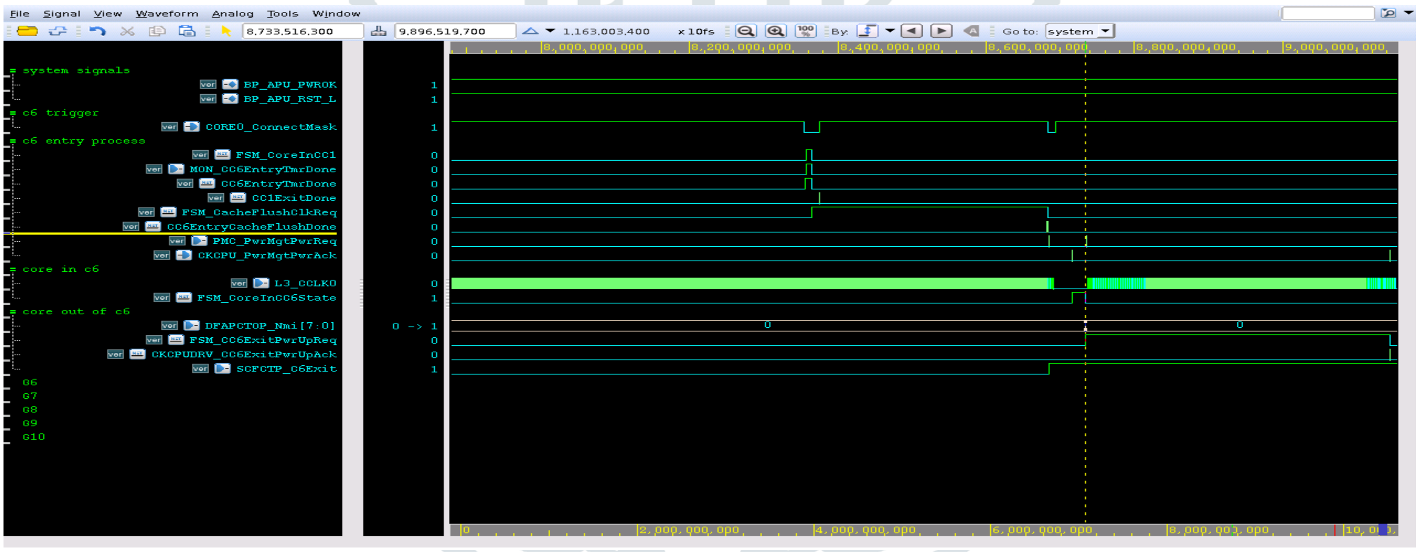

Fig.11. C6 Entry-Exit Waveform

## VIII. CONCLUSIONS AND FUTURE SCOPE

SoC level Verification of P-States ensures the efficient performance of the CPU by changing the frequency and voltage of the processor, according to the workload and obeying the P-State Limits.

Verification of C-States ensures the effective Power savings of the CPU by intelligent shutting down of the Clock and Voltage of the processor. This verification is done by considering various possible real time scenarios for the processor so that the end users should not get any bugs further while using the device or processor.

The Proposed Front-End Verification of CPU P-States and CPU C-States at SoC level is a promising step in ensuring correctness of RTL that will save the designs from multiple spins thus reduces the Time-to-Market and production cost.

Future scope of this project can involve the verification of Device States, Sleep States for the complete SoC.

## REFERENCES

[1] Alan Gibbons, David Flynn, Kaijian Shi, Michael Keating, and RobAitken, Low Power Methodology Manual For System On Chip design,2007

[2] Srikanth Jadcherla, Janick Bergeron, Yoshio Inoue,David Flynn,Verification Methodology Manual for Low Power,2009

[3] Cui, Shu Ping, and Chuang Xie. "Research of Low Power Design Strategy Based on IEEE 1801 Unified Power Format", Advanced Materials Research, 2014

[4] M. Arora, S. Manne, I. Paul, N. Jayasena and D. M. Tullsen, "Understanding idle behavior and power gating mechanisms in the context of modern benchmarks on CPU-GPU Integrated systems," 2015 IEEE 21st International Symposium on High Performance Computer Architecture (HPCA), Burlingame, CA, 2015, pp. 366-377.doi:10.1109/HPCA.2015.7056047

[5] R. R. M. Pinto, "Power Management Verification – An Evolving Discipline," 2008 Ninth International Workshop on Microprocessor Test and Verification, Austin, TX, 2008, pp. 57-60.doi:10.1109/MTV.2008.13

[6] Freddy Bembaron, Sachin Kakkar,Rudra Mukherjee,Amit Srivastava "Low Power Verification Methodology Using UPF"

[7] Bhanu Kapoor, J. Marc Edwards, Shankar Hemmady, Shireesh Verma, and Kaushik Roy, " SoC Power Management Verification and Testing Issues", 2008 IEEE

DOI 10.1109/MTV.2008.21