International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. 15, No.1 (Mar-24)

http://dx.doi.org/10.12785/ijcds/150172

Multi-Voltage Design of RISC Processor for Low Power Application: A Survey

Dheeraj Kumar Sharma[1] **and Rahul Vikram**[2]

[1,2]*Department of Electronics and Communication Engineering, National Institute of Technology Kurukshetra, Haryana, India*

Received 10 Nov. 2023, Revised 5 Feb. 2024, Accepted 7 Feb. 2024, Published 1 Mar. 2024

Abstract: Power management is becoming important aspect as the size of transistor is shrinking. For processor design, Reduced Instruction Set Computer (RISC) architecture is preferable as compared to Complex Instruction Set Computer (CISC) architecture because of its simplicity and availability. To design the low power RISC processor, there are a few techniques that had been used earlier, such as a) pipelining and b) Common Power Format language to generate power intent of RISC processor design. In the present work, for designing a 16-bit RISC processor with low power consumption, a multi-voltage design technique has been used. In this technique, different supply voltages are provided to different blocks of the design. This technique is implemented with the help of Unified Power Format (UPF). Further, various operations such as ADD, SUB, INVERT, AND, OR, Right Shift, Left Shift, and Less Than are verified on modelsim for the designed 16-bit RISC processor.

Keywords: Unified Power Format (UPF), Low Power Design, Multi-Voltage Design, Power Gating, Clock Gating, Retention, Isolation, Architecture

1. Introduction

Earlier, processor performance and circuit speed or processing power were synonymous, e.g., MIPS (million instructions per second) or MFLOPS (million floating point operations per second). In designing ICs, power consumption was not a major concern. However, power consumption became the most important issue in nanometre technology [1]. Process parameter variability increased due to excess device size scaling, which causes problems like testing and reliability, that causes to change the trend of measuring the performance of processor. Nowadays, RISC processor is preferred over CISC processor because of its simple instruction set and its low power consumption feature. RISC takes a shorter time to execute instructions. It makes RISC faster than CISC. It is because that RISC uses pipelining concept to maximize the speed and low power consumption. In this paper, we have enlightened the multi-voltage design of RISC processor to make the processor chip consume less power.

There are two types of power consumption, dynamic and static. Dynamic power consumption is the power consumed by the transistor, when output value switches to another value. Therefore, there is dependency on clock frequency and switching activity. Static power consumption is the power consumed by transistor when output is not switching. Once power is applied to the transistor, there is leakage current. It causes static power consumption, and there is no dependency on clock frequency and switching activity.

A. Dynamic power

Dynamic power is the power dissipated during logic value change on nets, having two components internal power and switching power. Charging and discharging of output capacitance results in switching power. Internal power results from crowbar (short circuit current) that flows during logic value changes on nets, in PMOS-NMOS stack.

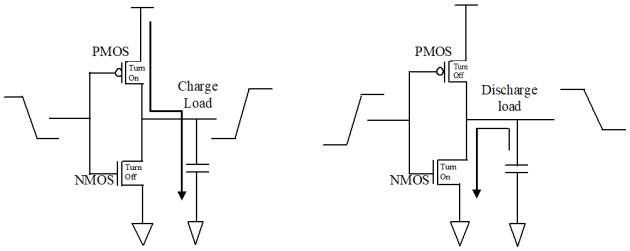

Figure 1. Switching Power

As illustrated in Figure 1, the transition of output net value from 0 to 1 charge the output capacitance through PMOS transistor. A transition of output net value from 1 to 0 discharge the output capacitance through NMOS transistor. Energy dissipated in every transition depends on capacitive load of net and supply voltage. Also, during the logic transitions on net, because of current flow, there is a

E-mail address: sharma986dheeraj@nitkkr.ac.in, rahulvikramcool@gmail.com

dependency of long-term dynamic power consumption on switching activity and clock frequency [2], [3], [4].

Internal power consumption happened only for short duration of time at an intermediate voltage level of input signal when both the NMOS and PMOS are conducting. This situation results in approximately short circuit path from VDD to GND. It has been shown in Figure 2 [5], [6].

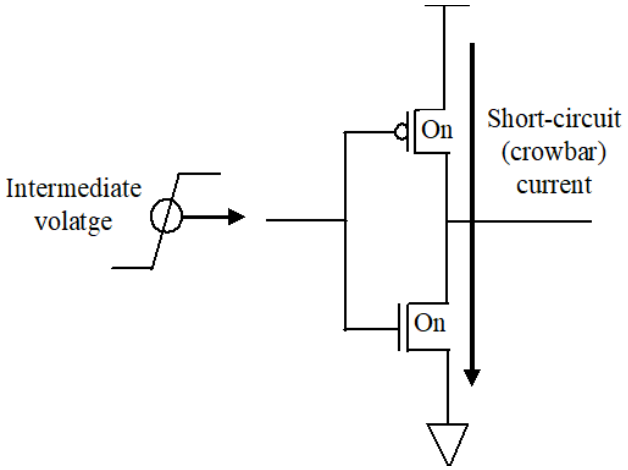

Figure 2. Internal Power

B. Static Power

In earlier CMOS technologies, leakage current was negligible. However, with the reduced feature size of transistor and reduced threshold voltages, leakage power increases significantly [7]. Subthreshold leakage, gate leakage and p-n junction reverse bias diode leakage are the main source of static power consumption. Leakage path are shown in CMOS inverter in Figure 3 [4], [6].

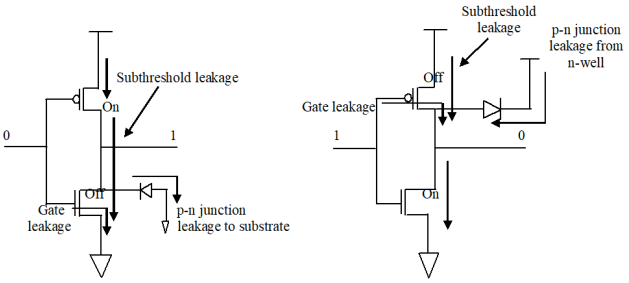

Figure 3. Leakage Current

There is permanent leakage at reverse biased p-n junction diode in all CMOS circuits as soon as power supply is applied. In NMOS transistor this leakage current flows from n-type drain and p-type substrate (p-well) while in PMOS it flows from n-type substrate (n-well) to p-type drain [8].

Once power supply is connected, there is small current flow through channel from drain to source in NMOS transistor and source to drain in PMOS transistor even when in the "OFF" state of transistor known as subthreshold leakage. Earlier this leakage current was negligible. However, with reduced power supply and reduced threshold voltage as soon as "OFF" gate voltage becoming close to "ON" threshold voltage, subthreshold leakage current increases exponentially [9].

Oxide layer used as an insulator between gate conductor and channel of MOS transistor is very thin layer resulting in gate leakage. Source and drain are separated by so thin layer i.e. Only dozen or fewer layer of isolating atoms. Under these situations, quantum effect tunnelling of electron can occur through gate oxide [10].

As power is supplied to transistor, leakage current gets generated irrespective of switching activity or clock frequency [11]. Leakage current can be reduced by lowering the power supply voltage or by disconnecting the transistor from power supply voltage.

As discussed earlier, static and dynamic power should be reduced. Therefore, low power design techniques are required to lower the power in circuits and chip. In the present work, we have discussed various low power design techniques with multiple voltages supply, such as reduction in supply voltage, power gating, multi-voltage design, clock gating, dynamic voltage and frequency.

If a low power design technique is used for circuit or chip. Then, there is requirement to verify it. Low power verification techniques are used for this purpose. We have discussed low power verification techniques such as power aware verification environment (PAVE) and x-propagation in this paper. The power specification of the low power design is required at the beginning. It can be found out by using unified power format (UPF). It tells power intent of design. UPF and its various commands have been described in this paper. For low power design of 16-bit RISC processor, UPF has been used. Further, the simulation of various operations is also performed on RISC processor.

The main contribution of paper includes:
a) A survey on various multi-voltage design techniques used for low power applications.
b) A study and discussion of low power verification techniques.
c) A discussion on unified power format and its command.
d) Simulation of different operations on 16-bit RISC processor using unified power format.

The remaining of the paper has been organized in 8 sections. Section 2 discusses about low power design techniques. The low power verification techniques are discussed in Section 3. Section 4 tells about unified power format. The tools used for low power verification are described in Section 5. Section 6 discusses about 16-bit RISC processor. Simulations results are shown in section 7 and section 8 includes summary. The conclusion is written in section 9.

2. Low Power Design Techniques

In this section, we will discuss various low power design techniques for circuits. Few techniques use multiple voltage supply also. There are some techniques to follow to mitigate the static power consumptions [6], [12]. As static power consumption has much dependency on power supply voltage, therefore keeping these things in consideration, we have mentioned some techniques. The dynamic power consumption depends on the switching power. Considering switching of voltages, few techniques for reduction of dynamic power consumption are described. Some of these techniques can also be used for reducing total (static and dynamic) power consumption. These are given below.

A. Reduction in Supply Voltage

Most simple technique to reduce the static power consumption is by reducing supply voltage. As power consumption is directly proportional to square of supply voltage, thus on reducing voltage 50 percent, current also reduced by 50 percent and power consumption is reduced by 75 percent. It has been illustrated in Figure 4. CMOS circuit achieves same degree of reduction in power for both static and dynamic power [2], [3], [4].

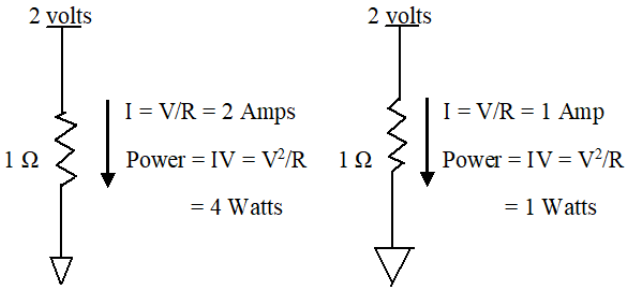

Figure 4. Supply Voltage vs Power example

As CMOS technologies have been lowering, so to reduce energy consumption it is necessary to reduce supply voltages. In 1980s, bipolar TTL (Transistor Transistor Logic) circuits were compatible with 5 volts supply voltage. Now for advanced technologies, it has fallen to about 1 volt. As a result of reducing power, power supply per gate reduces which results in lower energy consumption. But at the same time, it reduces the performance (switching speed) of each gate. In addition, threshold voltage of transistor should be reduced, because higher threshold voltage on reduced power supply have very less noise immunity and higher threshold voltage also cause subthreshold leakage and crowbar currents problems. With the reduced power supply voltage, output voltage swing would be reduced that can cause interfacing with the chip that require higher voltage swing [13].

B. Power Gating

It is a technique to save power by completely shutting down some portion of chip, which are currently not in use. It is shown in Figure 5. As an example, in smartphone chip when user is watching video, at that time voice processing block of chip can be shut down. As soon as user need to have call or need to receive call, then voice processing block should be "wake up" form shut down state [14].

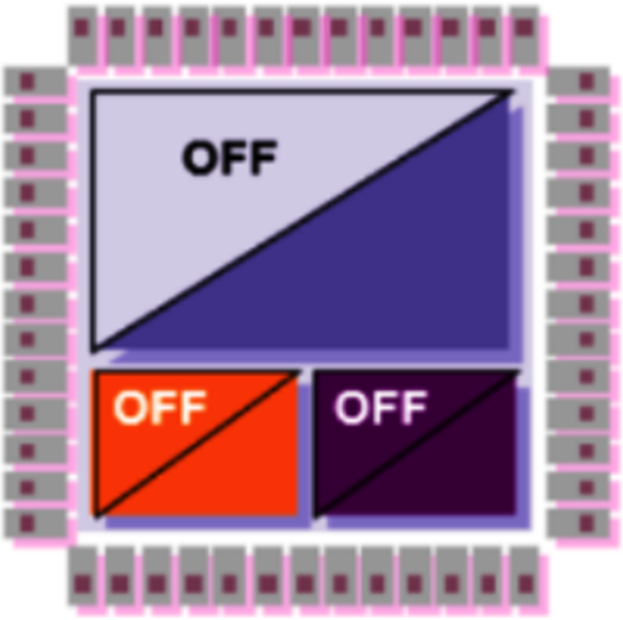

Figure 5. Power Gating

Power gating has good potential to decrease total power dissipation. Because it reduces both switching power (dynamic power) and static power (leakage power), by introducing power gating additional challenges appear. It includes power switching circuits, power controller circuits, retention register and isolation cells [15].

In Multiple-Threshold CMOS (MTCMOS) technology, high V_t (high threshold voltage) transistors are used as power switches, because these transistors have better switching speed and also reduces the leakage current than normal transistor [7], [16].

PMOS transistor is placed between supply voltage and power supply pin block as shown in Figure 6 and NMOS transistor is placed between VSS and power ground pin block [17]. The switching techniques as shown in Figure 6 and Figure 7 is also known as "coarse-grain technique" because this provides power gating for whole block. A common supply net for this block drives multiple transistor parallel [18], [19].

Another technique of power saving is "fine-grain technique". In this technique, each cell library contains its power switch allowing fine-grains to control library cells i.e., which cell needs to power down and which needs to power up. Power saving approach of this technique is better but requires more area to implement [20].

With the use of power gating, one block is power-down and another is always on (currently power-up). If both are

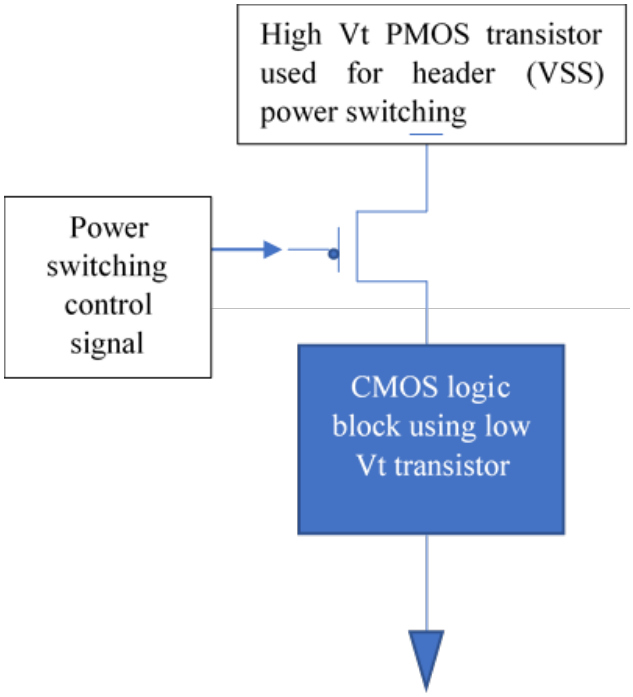

Figure 6. PMOS transistor as a power switch

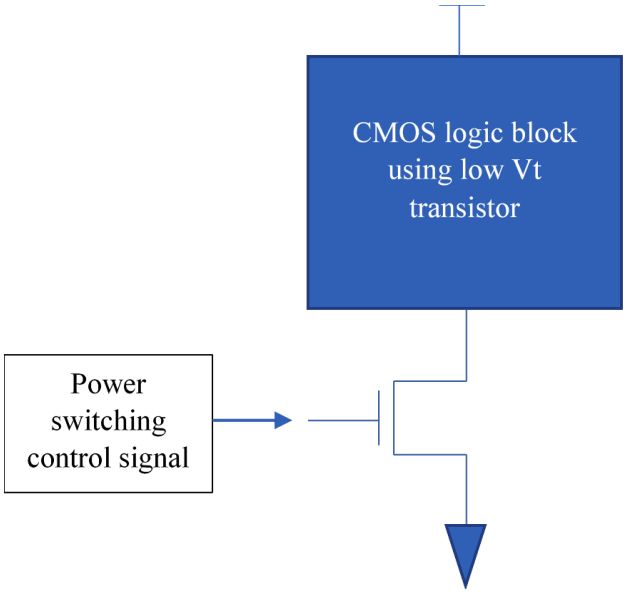

Figure 7. NMOS transistor as a Power Switch

connected, then it became necessary to make some isolation between these two blocks to prevent the Always-on block from taking intermediate or unknown values through the connected signal. It can cause to flow crowbar current. To provide isolation, we use isolation cell. The implementation of simple isolation cell is illustrated in Figure 8.

In Figure 8, when block on the left side of isolation cell is power-up, at that time P_up signal will be logic 1

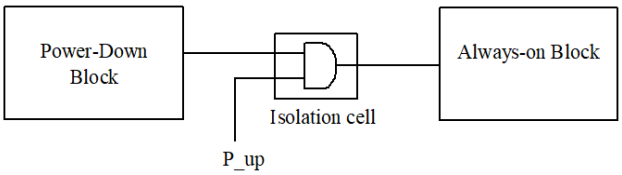

Figure 8. Isolation cell

so signal of left side block can pass to Always-on block. When left side block is power down, at that time P_up will be logic 0 which helps in blocking any intermediate or unknown value to the Always-on block. So, in this case output of isolation cell will be holding logic value 0. Isolation cell can hold logic 1 and it can also latch signal value during power-down event. Isolation cell must be connected with the power supply for the periods when left side block is power-down [21].

Multi-voltage and power gating techniques can be combined. Different block on the chip can have different supply voltages. Also power-down event occur for each block is different. Thus, in such cases interface cells between two different power domain blocks must provide both isolation and level shifting operations. That cell operation depends on power state of the block connected with the cell, so if one is power-down and another is Always-on block, then it will perform like an isolation cell. If both side blocks are connected with different voltage supply, then cell must operate like a level shifter. An enable level shifter cell that performs both the function, same as level shifter, this cell must have two separate supply voltages [22].

When any block of system goes through power down and back to power up, then it became important to restore the state of that block before it goes to power down. There are many strategies through which, we can store the previous state. One of them is by storing the states of block into the RAM present outside of the block and restore back, once wake up event occurs.

Another technique to store the state of block state is by using retention registers with in the power down block. Retention register stores the state of block in shadow register (bubble register) before power down. Once wake up event occur shadow register copy the stored data into main register. Shadow register is connected withA lways-on supply but this register use high V_t transistor to construct it so that we can minimize the leakage current from this register. The main register is constructed with low V_t leaky transistors for fast performance [5].

Implementation of retention register is illustrated in the Figure 9. In this figure, there are two signal one is SAVE and another is RESTORE. The saving of data into shadow register before power down is done by SAVE signal. For restoring data after power up is done by RESTORE signal. As compared to ordinary register, retention resister occupies

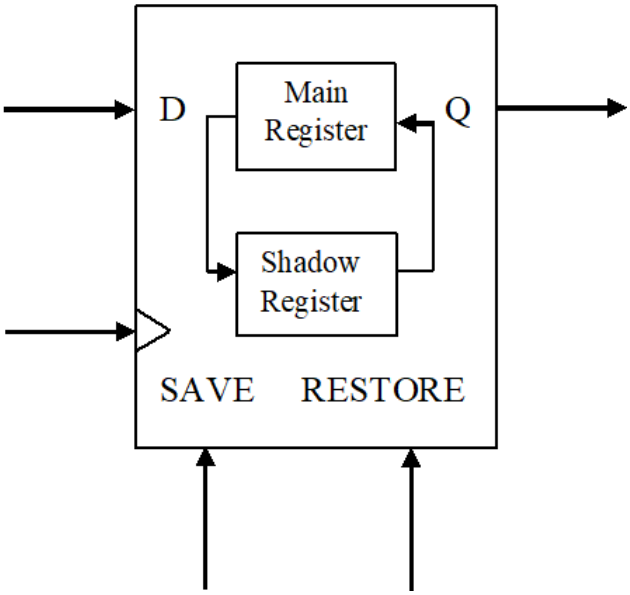

Figure 9. Retention Register

larger area and its shadow register requires Always-on power supply connection and rest of the device uses power down supply. This strategy of restoring data to the registers is faster and simple as compared to other [5].

C. Multi-voltage Design

Different speed is required in different block of chip. As an example, peripheral block need not to be much faster while RAM and CPU speed must be faster. As we have seen in earlier section, speed of transistor depends on supply voltage. Thus, as the supply voltage reduces the speed of transistor decreases. Similarly, by increasing supply voltage the speed of transistor increases. To obtain high speed with low power consumption, we need to supply higher supply voltage to RAM and CPU block and lower supply voltage to peripheral block as illustrated in Figure 10.

There are some costs and complexity arise by introducing more than one supply voltages on a single chip. It creates requirement of additional power pins to supply different voltages to the chip. It is necessary for power grid to distribute supply voltages to each block separately [23]. The chip with multi voltage block is shown in Figure 11.

When a voltage signal enters into other block with different supply voltage, then its logic level will be different than earlier block so we need a level shifter where ever two blocks are connected with different power supply voltage. Level shifter changes the voltage swing corresponding to the block in which signal is going to enter to maintain same logic in other power domain as well. For that, it requires two supply voltages. The level shifter is shown in Figure 12.

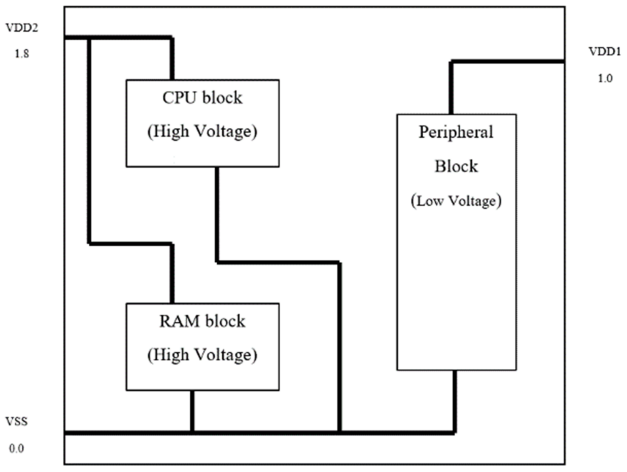

Figure 10. Multi-voltage design

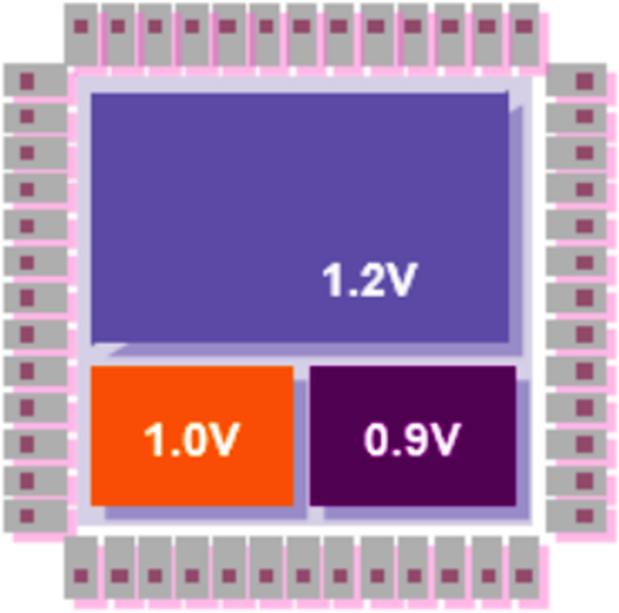

Figure 11. Chip with multi voltage block

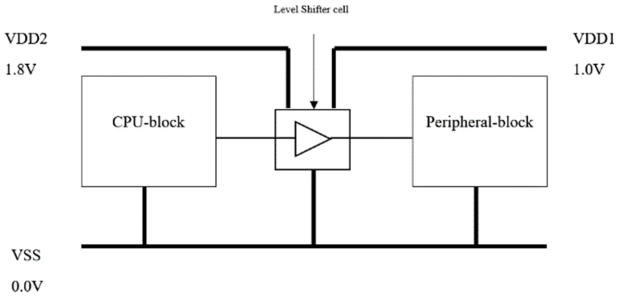

Figure 12. Level Shifter

D. Clock Gating

Dynamic power consumption can be significantly reduced by using clock gating technique. Using this technique, the time when register value are not changing, we can specifically supply clock signal to intended registers and can stop clock supply for other registers. A simple example of clock gating implementation is shown in the Figure 13.

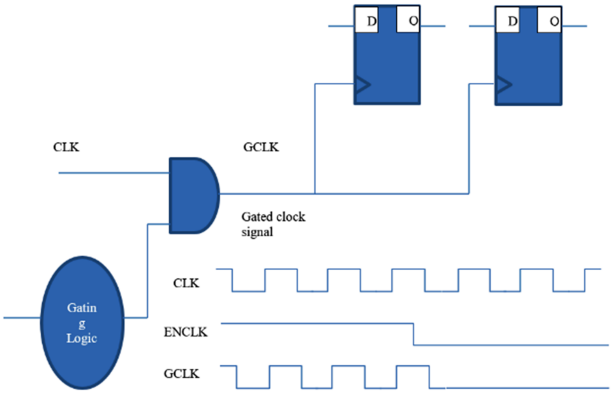

Figure 13. Clock Gating technique

Clock gating is beneficial particularly for those register banks that requires to retain their logic values over clock cycles. The clocks are disconnected from the registers that are not requiring clock cycles. This technique eliminates unnecessary dynamic power consumption due to switching activity by reloading register logic value at each clock cycles. The major challenge with this technique is to find the best location in the SoC to use this, and the logic to implement turning ON and turning OFF clock supply at proper time [24].

It is one of the best power saving techniques and well stablished as well. To implement this technique no additional supply voltage or any power related change is required and its implementation is relatively simple. It only requires some changes in netlist [25].

E. Dynamic Voltage and Frequency Scaling

Sometime workload changes during the operation, so when workload decreases correspondingly power should reduce because current workload would not require that much power as it requires when workload is higher. So, to facilitate voltage and frequency scaling during the operation we have one technique known as Dynamic Voltage and Frequency Scaling (DVFS).

As an example, during simple calculation in computer with a math processor, lower clock frequency and lower voltage might be applied to save power. During some complex or higher workload calculation, chip might have applied higher frequency and higher voltage supply, because this computation requires highest performance [26]. Clock frequency and supply voltage requires to change in such a way so that chip could meet required workload. It is known as dynamic voltage and frequency scaling. It is indicated in Figure 14.

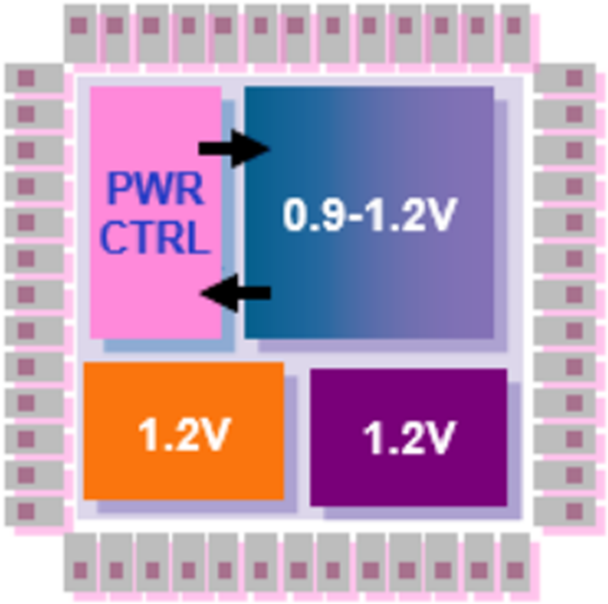

Figure 14. Dynamic Voltage and Frequency Scaling

There are two ways to design supply voltage to provide dynamic voltage scaling for a chip. One is by using different levels of voltages other is by defining some continuous range of voltage. A logic block to find proper voltage level for specific task, and a power supply with more than one voltage level, are required for dynamic voltage scaling. Analysis of different set of voltage levels and voltage ranges, and different operating frequencies are important that makes testing, verification, implementation, and design challenging. By merging both power gating and dynamic voltage scaling such that it can supply different voltage levels to the block according to workload for a specific block and can power down the block that are not in use.

3. Low Power Verification Techniques

As explained in various publications—verifying a design takes more time than writing the RTL code for it. “Everybody understands debugging is twice as difficult as programming,” Brian Kernighan, the developer of the C programming language, said in 1974. It’s almost as difficult as writing a program today in the first place. Specifying takes a lot of time and work. In this section, we will discuss various low power verification techniques that are used for verification of low power design of circuits and chips.

Power aware design provide some of essential functionalities like Isolation, power switches and retention register. These functionalities use low power techniques like multi-voltage, power shut down and some advanced strategies like standby, DVFS, Low power supply. Power format file includes strategies for level shifter, isolation and retention registers.

For dynamic low power verification, during RTL elaboration power format file read by simulator and evaluates the information of retention, isolation and power switching for simulation test preparation. Our first aim should be clear that all the sequences used for retention, shut down and isolation are correct. For dynamic simulation, another important aspect is usage of different strategies such as accurate isolation clamp value, assertions used in sequences for low power, handling of save/restore along with reset, handling of power on, handling of multi-rail macro and power state coverage.

Power aware simulations have another main feature is "X-propagation" which help to ensure that RTL behaviour of code suits with its corresponding GLS (Gate Lavel Simulation). With the help of this feature, we can deduct functional bugs present in current design and debug them in early stage of design cycle. It is necessary to have precise power aware simulator and well-organized power aware simulation tool so that we can be able to root cause and fix the test failures.

In the current scenario, speed of low power design is increasing rapidly, and cost of silicon failure after tape out is very high so to not escaping any bugs after product is taped out. We need to have efficient power aware simulator having powerful debug feature [27].

Requirement's verification of Power management Requirement of low power verification are mentioned below:

a) Power Control Management Verification.
b) Verify all the changes which software requested.
c) Verify the occurrence of power transition when expected.
d) Verify hardware situations which could cause power state changes.
e) Verification of behavior of each domain in state.
f) Verify checks related to low power behavior.
g) Verify by considering low power for normal checks.
h) Coverage.
i) Analyse all transitions and simstates.
j) Power aware debug.
k) Ensure using powerful debug tool.

The following are the low power verification techniques.

A. *Power Aware Verification Environment (PAVE)*

For writing power aware assertions, to make accessible UPF objects and for low power events monitoring, these tasks require an infrastructure that take these tasks at one platform. That infrastructure is known as Power Aware Verification Environment (PAVE) as depicted in Figure 15. With the help of efficient UPF command such as query and bind_checker, it set up proper PAVE, where query command used for query and bind_checker command used to bind UPF items such as retention technique, power domains, isolation technique and power switch with the checker modules. To write custom assertion, user can use this infrastructure for cases when reset is "low" and clock is "high" [28].

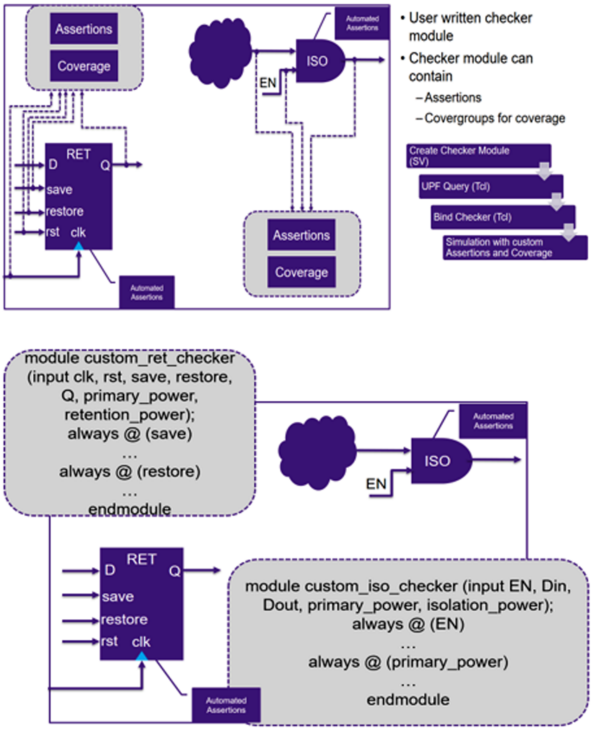

Figure 15. Power Aware Verification Environment Infrastructure

B. *X-Propagation*

With the help of X-propagation technique, we could know the exact effect of power, this is very powerful technique for verifying low power design. With the help of this technique, we could easily find the missing isolation cell related bugs such as inactivity of any isolation enable signal that would cause malfunctioning of isolation cell. It is shown in Figure 16.

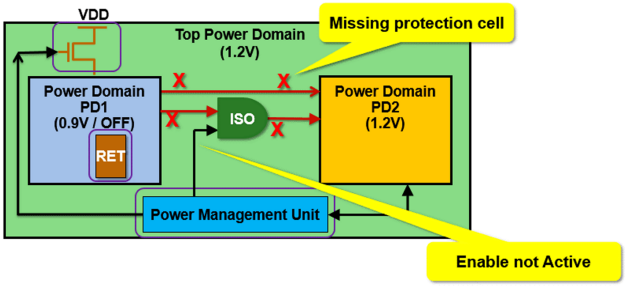

Figure 16. Root causing bugs with the help of X-propagation

Because of different style of RTL coding "X" might be escaped, that leads to a design bug, but later this can be caught when gate level simulation (GLS) starts running, although it is both performance incentive and time consuming. X-propagation technology provide behaviour like a gate level simulation at very early stage of RTL simulation to find the bugs [29].

4. Unified Power Format

As we discussed in previous section, Unified Power-Format (UPF) is used for low power verification. In the persent work, UPF and its commands are used during simulation. Unified Power format is an IEEE standard to specify power intent of design. It has an ability to define power specification at very beginning of the process. It is possible due to its ability to implement appropriate power information early in register transfer level (RTL). UPF come up with a format to describe power aware design which is not possible to describe using HDL. Consistent semantics of UPF also define across implementation and verification. i.e same can be verified whatever is implemented. UPF uses Tool Command Language (TCL) to specify power intent [30].

Simulation Semantics

There are multiple commands used in writing the UPF code to resemble the design and add some extra element to implement low power scenarios in the design. Some of the simulation semantics are discussed below.

A. Semantics of Supply Network Simulation

During low power simulation, there are two important information state of supply net and voltage value, contained by every supply net. Further, supply state is classified into two of its state, one is on/off state and another is fully/partially on/off state. Assertion of supply net and port in on/off state depends on the connection of supply ports and nets to the supply pad. The on state is asserted if supply ports and nets are connected to supply pad, and off state is asserted when supply nets and ports are disconnected to supply pad. Default state of pad is on. Term full/partial state of supply nets and ports is the conductance of power switches in the path of supply nets. Assertion happens when switch is partially on and de-assertion happens when power switch is fully on or fully off. Voltage value is same as the value of supply pad [31]. UPF Flow is depicted in Figure 17.

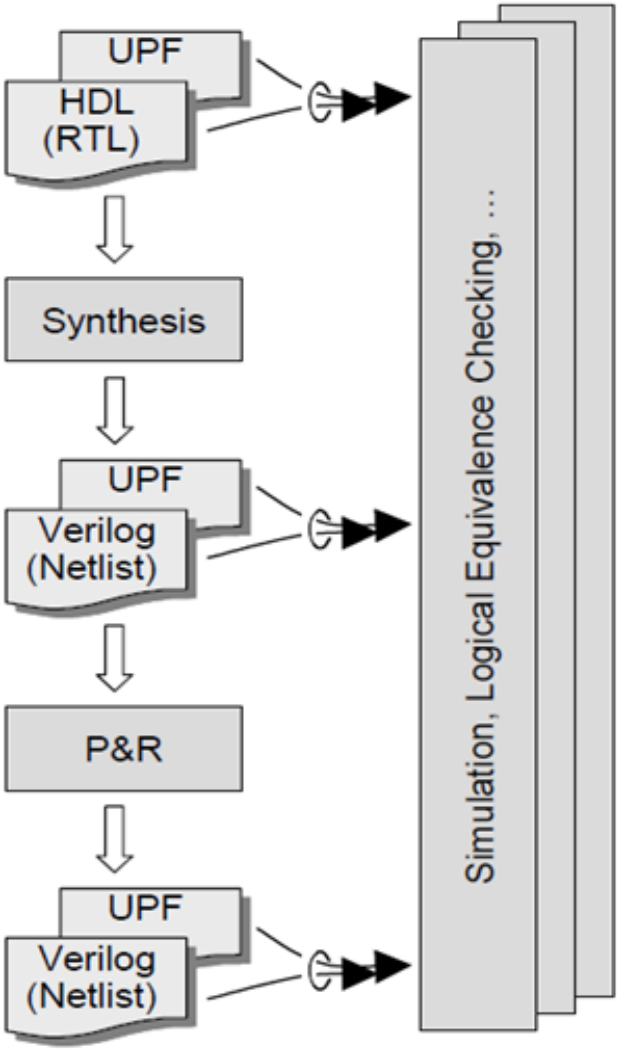

Figure 17. UPF Flow

B. Modelling of Power Switches

Power switches process the voltage change at its input port and its control port. At its control port, voltage changes occur. This voltage is compared with on-supply Boolean function. If it matches with on-supply Boolean function, then power switch is closed and supply input value to its output port. If it does not match, then power switch is open which turns off the output port. If x and z value appear at control signal, then power switch behaves as undefined. In this case, simulation gives some warning or error [32].

C. Hardware Description Language Supply Net Control

To turn on and turn off power supply with the help of test bench, hardware description language (HDL) should provide appropriate functions.

HDL Verilog, VHDL both use upf_supply package which provide functions such as supply_on (pad_name: string, value: real), supply_off (pad_name: string, value: real) and supply_partial_on (pad_name: string, value: real). If operation is successful, then it returns value 1 and if opretaion is not successful, then it returns value of 0. The functions are described below.

a) supply_on (): it turns on the specified supply pad and on/off assertion happens in associated supply net.

b) supply_off (): it turns off the specified supply pad and on/off de-assertion happens in associated supply net.

c) supply_partial_on (): it turns on the specified supply pad and assertion happens from on/off state to off state and partial/full state [33].

D. Data Type of Supply Net

A portable and simple data type is used to make confirmed voltage value. The supply net states can be modelled and propagate easily in different hardware description language. Integer type is used as portable and simple data

type across all languages. Since, supply net contains two information. One is voltage value and other is on/off state. For this purpose, 32-bit signed integer is used. The bit encoding is depicted in Figure 18. To store voltage value and net state, data type is used as given below.
a) To store voltage value, 32-bit signed integer is used with the uV precision.
b) To store net state, 32-bit bit vector is used.

Full/Partial Bit State [1]	On/Off bit state [0]	Semantics
0/1	0	Supply net is OFF (Full/Partial bit is immaterial).
0	1	Supply is fully ON.
1	1	Supply is partially ON.
1	0	This value is undefined.

Figure 18. Bit Encoding

E. Supply Net Resolution

Supply nets are usually connected with the output of single switch but in some of the applications, multiple switches are required to be connected with the same supply net. So, it is necessary to have some resolution technique to specify states and voltages for supply net, that are supplied by each and every switches. To specify which resolution method to be used, create_supply_net command is used. Resolution methods are mentioned below [34].
a) unresolved: single output (single power switch) may only be connected with supply net.
b) one hot: supply net might be connected with multiple output. At a time only one output will be ON.
c) parallel: supply net might be connected with many outputs. More than one output might be ON at the same time. But voltage value will be same for all output.

Connecting to the output of more than one switch with an unresolved resolution of a particular supply net is not allowed. Doing so causes an error in log.

F. Isolation Command

Isolation technique is used whenever any part of design is going through power down and adjacent power domains with which this design is connected, are in power up. Therefore, to isolate signal from power up domains, set_isolation command is used and to control which port to be isolated, set_isolation_control command is used [35]. To distinguish between different isolation cells, there is a command name_format, is used to derive different names. Throughout the time of simulation, each process corresponds to isolation signal. The isolation signal is sensitive to its non-isolated signals, input to isolation cell and the control signal corresponding to process. Whenever control signal changes its value, corresponding isolation compares the new value with the sense value of corresponding isolation. If the sense value is same as control signal, isolation signal is driven by the specified clamp value. Otherwise, it will propagate isolation value corresponding to non-isolated signal. If enable signal is X or power supply is turned off for corresponding isolation element, in these cases, isolation signal driven to X [36], [37], [38].

G. Retention Command

To determine the register which is needed to be a retention register in a specific power domain and to set the restore and save signals for the corresponding retention register, commands used are set_retention and set_retention_control.

At the time of simulation, there are two additional processes have been created by every retention register. One is for the save signal to sense the save activity and another process is for the restore signal to sense the restore activity for the corresponding retention register. Also, create a retention memory to retain the state for each sequential element, which is also called as shadow register. It is same as the register.

When value of save signal changes and matches the corresponding save sense value, then register value is saved in shadow register. Similarly, when the value of restore signal changes and matches with corresponding restore sense value, then register restore the value from shadow register.

When value of save signal is X or Z, then default initial value will be saved in the shadow register. Similarly, when value of restore signal is Z or X, then default initial value will be restored in the register. Whenever retention register supply is shut off, then default initial value will be loaded in both, shadow register and register [39].

H. Set_design_top

This command is used to specify the top most level of design instances. It is the design in which all other required design instances are instantiated. It is shown in Figure 19. This is very useful information for verification and simulation tools [40].

Purpose	To specify top level design.
Syntax	**set_design_top** *name_of_instance*
Arguments	*name_of_instance* specify the name of top design module.
Return Value	If succeeded return 1 else return 0.

Figure 19. Description of set_design_top command

I. Create_power_domain

This is important and mostly used upf command, which is used to specify different power domain as required by power management strategies. It is depicted in Figure 20.

J. Connect_supply_net

It is used to connect supply net to supply ports or pins. Figure 21 shows this command.

K. Add_pst_state

This command specifies state of every supply net. It's purpose, syntax, arguments and return value is shown in Figure 22.

Purpose	To specify distribution of power supply for each and every set of design element.
Syntax	**create_power_domain** *name_of_domain* **-elements** *list_of_element* **-include_scope** **-scope** *name_of_instance*
Arguments	*name_of_domain* specify some appropriate name of domain. **-elements** *list_of_element* specify elements under that domain. **-include_scope** specify the scope in which current domain is. **-scope** *name_of_instance* specify some appropriate name of scope.
Return Value	If power domain is created then it will return the name of power domain else return null.

Figure 20. Description of create_power_domain command

Purpose	To connect supply net to supply ports or pins.
Syntax	**Connect_supply_net** *name_of_net* [**-ports** *list*] [**-pins** *list*] [<**-cell** *list*|**-domain** *name_of_domain*>] [<**-rail_connection** *type_of_rail*>|<**-pg_type** *type_of_pg*>]* [**-vct** *name_of_vct*]
Arguments	*name_of_net* **-ports** *list* **-pins** *list* **-cell** *list* **-domain** *name_of_domain* **-rail_connection** *type_of_rail* **-pg_type** *type_of_pg* (power/ground type) **-vct** *name_of_vct* (To indicate connection to which HDL ports are connected)
Return Value	If successful returns the name of supply net else return 0.

Figure 21. Description of connect_supply_net command

Purpose	To specify the state of each supply net
Syntax	**add_pst_state** *name_of_state* **-pst** *name_of_table* **-state** *supply_state*
Arguments	*Name_of_state* **-pst** *name_of_table* **-state** *supply_state*
Return Value	If successful returns 1 else return 0.

Figure 22. Description of add_pst_state command

5. Tool Used For Low Power Verification

In this section, tools used for static, dynamic low power verification and low power coverage have been discussed.

A. Static Low Power Verification

When RTL design and corresponding UPF design has been performed, then we need to check design correctness by verifying it statistically. In static verification, we need not to write test bench. Static verification is done with the help of static verification tools such as VC-LP by Synopsys, which generates the report file and mentions all the initial error in the report.

The main target of static verification is to uncover all the issues related to structure that affects the design in microarchitectural and architectural aspects. Most of the changes occur due to insertion of multi-voltage and power management element cells, such as level shifter, isolation, power switches, retention flops and enable level shifter. These power management cells are essential for power shutdown.

VC-LP

In today's scenario, electronic products require advanced power management elements to be supported in low power design of System-on-chip (SoC) of electronic products. So, after enablement of advanced power management element in SoC design, verification of such SoC becomes more complex and challenging as compared with the always on SoC design. To make easier verification of these complex design, VC-LP (Figure 23) tool helps to verify many initial checks. VC-LP tool offers full chip performance and capacity, and includes more than 400 for whole low power static signoff [40].

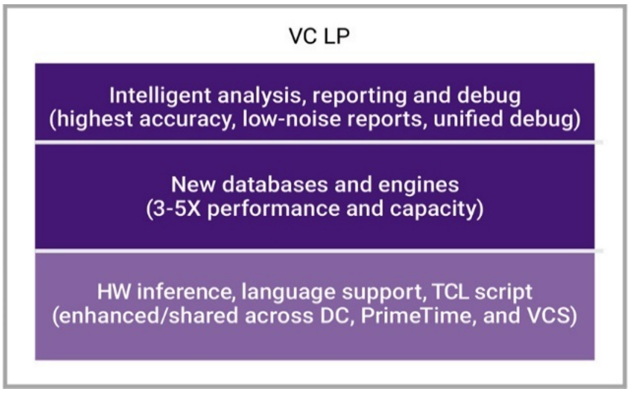

Figure 23. VC-LP tool flow

Features and Benefits

At different stages of SoC design flow like RTL, post-synthesis and post-P&R, this tool is used to catch low power bugs. It is also faster in comparison with traditional methods. VC-LP is needed at different levels of design because new design element and low power design techniques, are added. So, bugs could be present at any stage. The following are the checks performed by VC-LP.

a) *Power Intent Consistency Checks*: Before the implementation, it is necessary to perform syntax and semantic checks on UPF to ensure that power intent is correct unless incorrect power intent leads to incorrect low power design.

b) *Architectural Checks*: Global checks are performed on RTL signals if it is not following power architecture rules. VC-LP tool validates the design by performing multiple checks on some important signals in the design for different power modes, which helps in finding bugs, and debug them in early design cycle [41].

c) *Structural and Power Ground Checks*: With the help of this tool, it is possible to check proper connection and insertion of different power management element like level shifters, power switches, isolation cells, always-on cells and retention registers connected in the design.

B. Dynamic Low power Verification

Dynamic low power verification is done with methods such as simulation, emulation and GLS verification. This method requires to develop test benches. It is very time consuming because its simulation takes more time to complete as compared to normal simulation. For this verification, VCS-NLP tool is used to perform low power simulation.

VCS-NLP

As there are different low power strategies included in design so to understand those strategies, we require low power simulation tool, which could understand the different supply rails for specific block of design. VCS-NLP does this role very well. Whereas normal simulator could not be able to understand the power rails specified in the design and also don't have understanding of UPF (Unified Power Format) [42].

VCS-NLP is equipped with the feature of VCS with native low power simulation. The VCS is able to understand UPF that allows to verify correct design and advanced voltage control techniques in a comprehensive manner.

VHDL/Verilog, gate level netlist or RTL representation design are taken by VCS-NLP in similar manner as by VCS. The difference between VCS-NLP and VC-LP, is that VCS-NLP uses native low power flow, which requires power intent and that is given in the form of UPF file. Both RTL code and UPF file, are loaded in simulation tool along with corresponding test bench [43]. The low power simulation flow of VCS-NLP is illustrated in Figure 24.

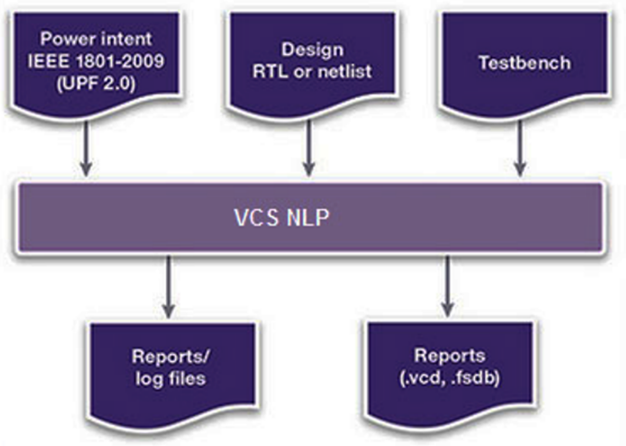

Figure 24. Low power simulation flow of VCS-NLP

C. Low Power Coverage

Low power coverage is used to get the coverage of all power management element data and their control signal value. The coverage database contains the data whether all retention register control signals like save and restore, are toggling or not, whether control signals of isolation cells are toggling or not, whether all simstate value covering all transition scenario like CORRUPT→NORMAL and NORMAL→CORRUPT etc., whether all the power state table is following i.e. all the power state are occurring or not.

To generate a .vdb file for low power coverage, it is required to pass some important options specified by VCS-NLP tool during the compilation of design code. One of the important option is after the compilation of design. There is a (.vdb) extension file generated in the log, which designed vdb.

To get the coverage of low power object in the data base file, we have to pass some option during the compilation of design code. There is a compile time option -power=coverage that helps in creating covergroups automatically. These cover groups are created for low power objects mentioned in the design such as states/transitions of supply set simstate, power switches, supply net/port, supply set power, enable signals of isolation cells, control and ACK port of power switches, and SAVE/RESTORE signal of retention registers [44].

There is another very important compile time option that is used along with -power=coverage is -power=dump_hvp which generate a (.hvp) extension file, which is required to extract the coverage data after the generation of coverage report.

For generation of low power coverage report, it is required to pass -lpcov in URG command line. It searches hvp (Hierarchical Verification Plan) file in design and test directory mentioned in the command line and extract the low power coverage data.

In Figure 25, some of the compile time options are mentioned for enabling coverage of low power objects specified in the design. These options can be used in VCS command. In VCS command, more than one options can be used by using + sign in different options.

Option	Description
-power=cov_psw	To enable coverage of low power objects ACK, control port and power switch state of power switches.
-power=cov_pst	To enable coverage of both transition and states of power state table (PST).
-power=cov_iso	To enable coverage of enable signal of isolation cells.
-power=cov_ret	To enable coverage of SAVE and RESTORE signals to retention registers mentioned in the design.
-power=cov_pst_transition	If it is requiring to enable only transition of power state table (PST) then this option is useful.
-power=cov_pst_state	If it is requiring to enable only state of power state table (PST) then this option is useful.
-power=cov_port_state	To enable all port state mentioned in the design.

Figure 25. Compile time options

For generation of coverage report from VCS-NLP generated coverage database file, URG command is used to merge coverage database of design and test coverage database. By default, it generates the coverage report in html format. URG command support different useful options. Some of the important options are mentioned below [45].

a) -dir: Used to specify the directory of coverage database of design and test.

b) -report: Used to specify the report generation directory and name of the report file.

c) -lpcov: Used to generate low power coverage report by fetching the coverage database and its .hvp file, which contains all low power objects coverage.

d) -dbname: Used to specify the name of merged coverage

database.
e) -format text: Used to generate the report in the text format instead of html format.
f) -elfile: Used to exclude coverable object specified in the file.

6. 16-bit RISC Processor

In this section, 16-bit RISC processsor, its architecture and components have been discussed. It's implementation and results are discussed in next section. One of the low power design technique (multi-voltage) is used for implementation with unified power format. The verification of design is performed by using VC-LP (for static power) and VCS-NLP (for dynamic power) tools. RISC stands for Reduced Instruction Set Computer. It is one of the categories of microprocessor architecture, that uses highly optimized and small instructions. About the history of RISC processor, in late 70's RISC processor is implemented by IBM, UC Berkely and Stanford. Berkely RISC-1 and RISC-2, IBM 801 and Stanford MIPS, are designed on the same architecture which is known as RISC. It works on single cycle execution time. It works on CPI (Clock Per Instruction) of one cycle [46]. The circuit of 16-bit RISC processor architecture is shown in Figure 26. It consists of 16-bit general purpose registers and 16-bit program counter. To indicate parity, zero and carry flag, 3-bit flag register is used in this processor [47].

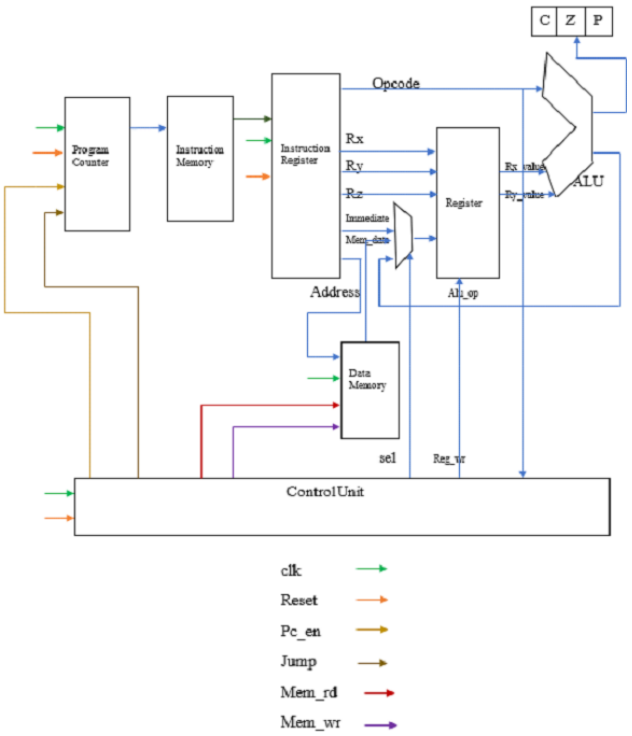

Figure 26. Architecture of 16-bit RISC processor.

There are four states in this processor idle, fetch, decode and execute. To perform functions as expected control unit, generate necessary signal in all states. To locate address of memory to fetch the instruction, 16-bit program counter generate different combinations of addresses. After execution of every instruction, program counter is incremented by 1 or it executes the instruction on "JUMP" value or decremented by offset value [48].

In instruction memory, instructions are written. Instruction memory takes program counter output as the input. Corresponding to address generated by program counter, instruction is fetched from instruction memory. Instruction registers store and decode the fetched instruction. Based on opcode immediate value, register address, memory location, source register and destination register, are assigned during decoding of instruction. On the basis of opcode selected, ALU performs operations for the same opcode. Corresponding control signal is generated from the control unit. There are eight 16-bit general purpose registers, named R0, R1,..., R7 [49]. The following are various components of RISC processor.

A. Instruction Set

Combination of commands in machine language for processor is called as Instruction set. It is also called as Instruction Set Architecture (ISA). Instruction set generates commands for processor to perform different operations. Some instructions and their format are mentioned below in Figure 27 with ALU operations in Figure 28 [50].

Op (4-bit)	Rs1 (3-bit)	Ws (3-bit)	Offset (6-bit)

Memory access: Store

Op (4-bit)	Rs1 (3-bit)	Rs2 (3-bit)	Offset (6-bit)

Data Processing:

Op (4-bit)	Rs1 (3-bit)	Rs2 (3-bit)	Ws (3-bit)	useless (3-bit)

Branch:

Op (4-bit)	Rs1 (3-bit)	Rs2 (3-bit)	Offset (6-bit)

Jump:

Op (4-bit)	Offset (12-bit)

Figure 27. Instruction Format

Op	
0000	Load word
0001	Store word
0010	Add
0011	Subtract
0100	Invert (1's complement)
0101	Logical Shift Left
0110	Logical Shift Right
0111	Bitwise AND
1000	Bitwise OR
1001	Set on less than
1010	Hamming Distance
1011	Branch on Equal
1100	Branch on Not Equal
1101	Jump

Figure 28. ALU Operations

B. Control Unit

This unit generates all the control signal on the top level of the processor design. It is containing an FSM to generate control signal for different cases. It provides the controlling signal for each of its sub module like ALU, instruction memory, instruction register, program counter, etc. Specification of all the control signals and their configurations with respect to instruction are mentioned below in Figure 29 [51].

Control Signal									
Instruction	Reg Dest	ALUsrc	Mem to Reg	Reg write	Mem read	Mem write	Branch	ALUop	Jump
Data Processing	1	0	0	1	0	0	0	00	0
LW	0	1	1	1	1	0	0	10	0
SW	0	1	0	0	0	1	0	10	0
BEQ, BNE	0	0	0	0	0	0	1	01	0
JUMP	0	0	0	0	0	0	0	00	1

Figure 29. Control Unit Configuration

C. ALU Control Signal

To perform arithmetic and logical operations, ALU is used. ALU stands for Arithmetic and Logical Unit. The opcode of instruction controls the ALU operation. After execution of instruction, result is written into output register. Arithmetic operations are ADD, SUB, etc. The logical operations are AND, OR, Invert, Left Shift, Right Shift, etc. The operands of instructions which are taken by ALU as input are 16-bit unsigned number [52].

ALU performs all the arithmetic and logical operations of the processor. Thus, to control this module of the processor, one should have some control signal generator circuit for generating the control signal. This function is performed by control unit [53]. Different control signals for ALU operations are mentioned in Figure 30 [53].

ALU Control				
ALUop	Opcode (Hex)	ALUcnt	ALU operation	Instruction
01	xxxx	001	Subtraction	BEQ, BNE
10	xxxx	000	Addition	LW, SW
00	0010	000	Addition	ADD
00	0011	001	Subtraction	SUB
00	0100	010	Invert	INV
00	0101	011	Left Shift	LSL
00	0110	100	Right Shift	LSR
00	0111	101	Bitwise AND	AND
00	1000	110	Bitwise OR	OR
00	1001	111	Set on less than	SLT

Figure 30. ALU control signals

D. Instruction Memory

Instruction Memory is also known as program memory, that stores the application and constant value. It does not store the variables. Here, code for specific application will be stored permanently, which got executed when specific application code is selected for operation. Basically, it is a ROM memory. Instruction memory is non-volatile. It keeps the data whatever programmed on it even after power is off. It's connection in RISC processor is shown in Figure 26 [54].

E. Data Memory

Data memory is the place where variables are located. Here, we can read and write values. It is volatile memory. All the data get vanished once power is off. It stores the data temporarily during the operations. While processing the data, it is required to store the data to take that data for further operation or to reflect the data on the output terminal [55].

F. Program Counter

Program counter is used to generate the address, which is incremented by 2 after one clock cycle. The instruction performs the respective operation by fetching its operand values mentioned in the instruction [56].

G. Power Intent

Power intent of the RISC processor design is nothing but how power is supplied to the different blocks of design. In the present work, we have proposed the multi-voltage design of RISC processor.

Power supply distribution is done on the basis of performance of different blocks in the design. With the help of block diagram, the power supply distribution has been explained in Figure 31. Here, different colours of blocks are specified as different power domain of the blocks [45].

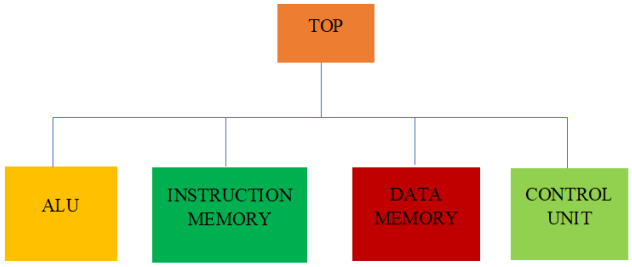

Figure 31. Power domain block diagram.

7. Simulation Results

Simulation results of different operations on RISC processor is mentioned below in points with the help of waveforms. In the present work, we have implemented 16-bit RISC processor on modelsim – INTEL FPGA STARTER EDITION 2021.1 on 90nm CMOS library. The code has been written in Verilog for every block of design according to its behaviour. To implement Multi-voltage design methodology for low power consumption, the Unified Power Format (UPF) has been used. First, 16-bit RISC processor elements or blocks are designed. Secondly, distribution of power supply to each elements is provided. Then, ports to all power domains are provided. Next, nets to power domains are supplied. Then, state of ports is defined. At last, level shifters are set. The various commands used are create_power_domain (to specify distribution of power suuply for design element), create_supply_port (to supply port to all power domains), create_supply_net (to supply net to all power domains), connect_supply_net (to connect

supply nets to all domains and elements), add_port_state (specify the state of supply net) and set_level_shifter (for setting the voltage swing to block).

A. Addition Operation

Addition operation performs on data whenever combination of ALUop, Opcode and ALUcnt, are as per the value mentioned in the third row of Figure 32.

Basically, it is fetching the data and program file, which is already provided during the simulation and performing the operation with respect to generated control signals. As shown in Figure 32, the waveform result ← a+b, and while writing the result into register reg_write signal should be high.

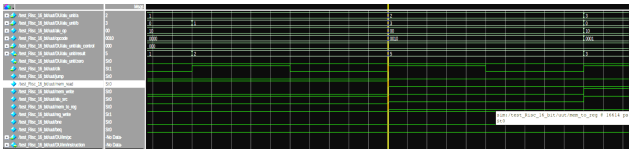

Figure 32. ADD operation

B. Subtraction Operation

Subtraction operation is performed whenever ALU control signals match their corresponding combination processor. Processor fetches the data from the register, which is present in the data file, as displayed in Figure 33. The subtraction operation performs result ← a-b and when the result is ready then, reg_write should be enabled to write the result into the register, that can be verified from Figure 33.

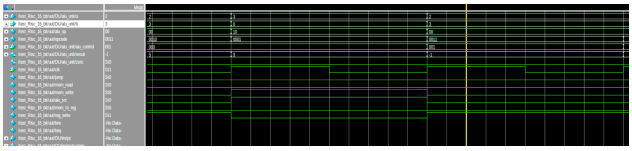

Figure 33. SUB operation

C. INVERT Operation

When control signals and their opcode match the combination with respect to INVERT operation, INVERT operation invert each bit in the operand of the instruction, which is correctly verified in Figure 34 with the help of waveforms.

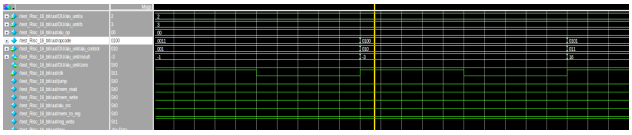

Figure 34. INVERT operation

D. Right Shift Operation

In this operation, first register data which is “a” in this case is shifted to right by decimal value of data in second register, which is “b” in this case. Thus, result ← a≫b operation will be performing, that has been correctly reflected in the simulation waveforms of Figure 35.

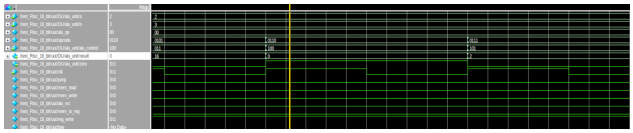

Figure 35. Right Shift Operation

E. AND Operation

With this operation, AND operation between two operands is executed by performing it on same position of bits present in both operand registers. It is clearly reflected in the waveforms shown in Figure 36.

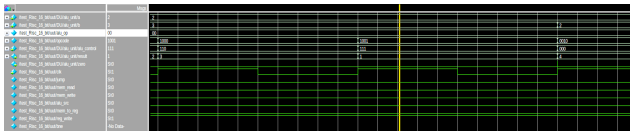

Figure 36. AND operation

F. OR Operation

This logical operation is performed on respective bit of the operands and the result of the operation is stored in the register and at that time, reg_write should be enabled. Functionality of this operation is properly figured in the form of waveforms in Figure 37.

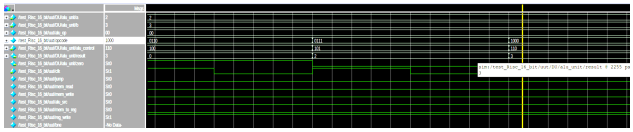

Figure 37. OR operation

G. Less Than Operation

In this operation, result will be logic ‘1’ whenever a is less than b, i.e.

$$a < b$$

In all other cases, when a is greater than or equal to b, i.e. $(a \geq b)$, result will be ‘0’. This operation is verified by the waveforms present in Figure 38.

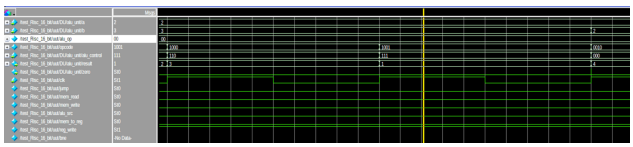

Figure 38. Less than operation

H. Application

RISC processor is very dominant these days. It is extensively used in the laptop, PC, cell phone, tablet and other electronic devices which requires to have a simple processing device [57].

RISC architecture is followed by many of the processor companies like ARM, INTEL, etc. because of its simple

architecture, design and small instruction set. This feature of processor also concludes its low power consumption feature with respect to CISC processor [58].

8. Summary

This paper discusses about comprehensive study on various low power design techniques such as Reduction in Supply Voltage, Power Gating, Multi-voltage design, Clock Gating, and Dynamic Voltage and Frequency Scaling. It also briefs about low power verification techniques like Power Aware Verification Environment (PAVE) and X-Propagation. Further, this paper deliberates Unified Power format and their commands such as Hardware description Language supply net control, Data type of supply net, Supply net resolution, Isolation command, Retention command, set_design_top, Create_power_domain, Connect_supply_net and Add_pst_state. The tools used for low power verification includes VC-LP for static power and VCS-NLP for dynamic power. Low power coverage is used to get the coverage of all power management element data and their control signal value. Low power coverage also uses VC-LP and VCS-NLP of Synopsys. Further, it discusses about 16-bit RISC processor and performing logical and arithmetic operations on Verilog. The present work as per our literature survey has not been reported till now. Therefore, no comparison can be made on results.

9. Conclusion

Low power design and verification is very crucial in these days to ensure the device consume minimum power. There are different verification techniques like static low power verification and dynamic low power verification, which we use to verify different low power objects like isolation cell, retention registers, power state table (PST), power switches, power supply nets, power supply ports, and functionality of the design along with power aware strategies. Testing different test case scenarios on a low power design with the help of writing test benches, assertions and sequences. At the end, generate low power coverage report and analyse the coverage report to ensure all the control signals of every low power object included in the design, are toggling and SoC is going through all the power states. Hence, these are the strategies needed to follow during the design and verification of low power SoC. In this paper, a survey on various multiple voltage supply designs has been done. Multi-voltage design of few instructions on 16-bit RISC processor has been performed. As per literature survey, this is fist time multi-voltage low power design implementation of 16-bit RISC processor.

16-bit RISC processor is implemented on modelsim – INTEL FPGA STARTER EDITION 2021.1. The code has been written in Verilog for every block of design according to its behaviour. To implement Multi-voltage design methodology for low power consumption, the Unified Power Format (UPF) has been used. The Unified Power Format contains all the low power design components with its proper configuration such as creation of power domains, connection of supply nets to its ports, addition of states to different ports, and creation of level-shifter for required blocks.

Additionally, ADD, SUB, INVERT, AND, OR, Right Shift, Left Shift, and Less Than operations of 16-bit RISC processor have been verified on modelsim and their simulated results are presented. In the future scope, the implementation of multi-voltage design with pipelining can be performed to improve the processing and lowering the power consumption at the same time.

References

[1] R. Sapna, M. M. Ulla *et al.*, "Low-power methodologies and strategies in vlsi circuits," in *Energy Systems Design for Low-Power Computing*. IGI Global, 2023, pp. 1–16.

[2] A. Pal, *Low-power VLSI circuits and systems*. Springer, 2014.

[3] G. K. Yeap, *Practical low power digital VLSI design*. Springer Science & Business Media, 2012.

[4] P. Khondkar, *Low-Power Design and Power-Aware Verification*. Springer, 2018.

[5] "Ieee standard for design and verification of low-power, energy-aware electronic systems," *IEEE Std 1801-2015 (Revision of IEEE Std 1801-2013)*, pp. 1–1068, 2016.

[6] A. Karthik, N. Domala, and G. S. Kumar, "An overview of low-power vlsi design methods for cmos and cntfet-based circuits," in *2023 International Conference on Computer Communication and Informatics (ICCCI)*. IEEE, 2023, pp. 1–4.

[7] V. K. Sharma, "Design and simulation of reliable low power cmos logic gates," *IETE Journal of Research*, vol. 69, no. 2, pp. 1022–1032, 2023.

[8] C. Piguet, *Low-power electronics design*. CRC press, 2018.

[9] S. Singar and P. Ghosh, "Fault-free d-latch configurations for low power applications," *Journal of Nanoelectronics and Optoelectronics*, vol. 13, no. 5, pp. 701–707, 2018.

[10] P. Kataria, A. Rai, A. Singh, A. Anand, R. Kapoor, and S. K. Singh, "Low-power vlsi design of arithmetic and logic circuits using the multiple-threshold cmos technique," in *Smart Computing*. CRC Press, 2021, pp. 624–632.

[11] H. Okuhara, "Power-efficient body bias control for ultra low-power vlsi systems," Ph.D. dissertation, Keio University, 2018.

[12] U. Kumari and R. Yadav, "A review about the design methodology and optimization techniques of cmos using low power vlsi," in *International Conference on IoT, Intelligent Computing and Security: Select Proceedings of IICS 2021*. Springer, 2023, pp. 181–197.

[13] A. Parveen and T. T. Selvi, "Power efficient design of adiabatic approach for low power vlsi circuits," in *2019 Fifth International Conference on Electrical Energy Systems (ICEES)*. IEEE, 2019, pp. 1–4.

[14] T. Suguna and M. J. Rani, "Survey on power optimization techniques for low power vlsi circuit in deep submicron technology,"

International Journal of VLSI design and Communication Systems, vol. 9, no. 1, pp. 1–15, 2018.

[15] K. S. Singh, S. Kumar, K. Nigam, and V. A. Tikkiwal, "Tunnel field effect transistor for ultra low power applications: a review," in *2019 International Conference on Signal Processing and Communication (ICSC)*. IEEE, 2019, pp. 286–291.

[16] K. Thilagavathi and S. Sivanantham, "Two-stage low power test data compression for digital vlsi circuits," *Computers & Electrical Engineering*, vol. 71, pp. 309–320, 2018.

[17] M. Hasan, M. J. Hossein, M. Hossain, H. U. Zaman, and S. Islam, "Design of a scalable low-power 1-bit hybrid full adder for fast computation," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 67, no. 8, pp. 1464–1468, 2019.

[18] S. Vidhyadharan, S. S. Dan, R. Yadav, and S. Hariprasad, "A novel ultra-low-power gate overlap tunnel fet (gotfet) dynamic adder," *International Journal of Electronics*, vol. 107, no. 10, pp. 1663–1681, 2020.

[19] S. Ahmad, N. Alam, and M. Hasan, "Robust tfet sram cell for ultra-low power iot applications," *AEU-International Journal of Electronics and Communications*, vol. 89, pp. 70–76, 2018.

[20] B. Tarabata Tupiza, "Design of a low-power vlsi temperature sensor," Master's thesis, Universitat Politècnica de Catalunya, 2020.

[21] M. Janveja, M. Tantuway, K. Chaudhari, and G. Trivedi, "Design of low power vlsi architecture for classification of arrhythmic beats using dnn for wearable device applications," in *2021 IEEE Nordic Circuits and Systems Conference (NorCAS)*. IEEE, 2021, pp. 1–6.

[22] H. Naseri and S. Timarchi, "Low-power and fast full adder by exploring new xor and xnor gates," *IEEE transactions on very large scale integration (VLSI) systems*, vol. 26, no. 8, pp. 1481–1493, 2018.

[23] V. Melikyan, T. Hakhverdyan, S. Manukyan, A. Gevorgyan, and D. Babayan, "Low power openrisc processor with power gating, multi-vth and multi-voltage techniques," in *2016 IEEE East-West Design & Test Symposium (EWDTS)*. IEEE, 2016, pp. 1–4.

[24] N. K. Challa and U. R. Nelakuditi, "The new era on low power design and verification methodology," in *11th IRF International Conference*, 2016.

[25] S. Kayala and K. Vasanth, "Low power vlsi architecture for data comparators for non-linear imaging applications," in *2021 International Conference on Innovative Computing, Intelligent Communication and Smart Electrical Systems (ICSES)*. IEEE, 2021, pp. 1–4.

[26] H. Ranjan and K. Siddesha, "Implementation of dvfs technique for processor power and temperature reduction."

[27] A. Srivastava, R. Mukherjee, E. Marschner, C. Seeley, and S. Dobre, "Low power soc verification: Ip reuse and hierarchical composition using upf," *DVCon proceedings*, 2012.

[28] R. Patel, A. Naik, A. Singh, A. Arya, and P. Bhatnagar, "Advanced upf based voltage-aware verification for ios," in *2015 19th International Symposium on VLSI Design and Test*. IEEE, 2015, pp. 1–2.

[29] P. Girard, N. Nicolici, and X. Wen, *Power-aware testing and test strategies for low power devices*. Springer Science & Business Media, 2010.

[30] P. Khondkar—Mentor and A. S. Business, "Effective elements lists and the transitive nature of upf commands."

[31] C. Ranjitha and B. Sujatha, "Power aware design and convergence of router using unified power format standards," in *2019 4th International Conference on Recent Trends on Electronics, Information, Communication & Technology (RTEICT)*. IEEE, 2019, pp. 307–312.

[32] P. Khondkar, P. Yeung, G. Chidolue, J. Hupcey, R. Koster, and M. Bhargava, "Free yourself from the tyranny of power state tables with incrementally refinable upf," *February-March, DVCon*, 2017.

[33] D. A. S. Committee *et al.*, "Ieee standard for systemverilog unified hardware design, specification, and verification language standard ieee 1800," *http://www. edastds. org/sv/*, 2005.

[34] M. R. Lebaka, A. Guizer, P. Khondkar, S. P. Engineer, and M. A. S. Business, "How upf 3.1 reduces the complexities of reusing power aware macros," in *Proceedings of the Design and Verification Conference and Exhibition (DVCON), San Jose, CA, USA*, 2020, pp. 2–5.

[35] J. S. Nagendra and N. Ramavenkateswaran, "An overview of low power design implementation of a subsystem using upf," in *2020 International Conference on Inventive Computation Technologies (ICICT)*. IEEE, 2020, pp. 1042–1047.

[36] N. Ravi Kumar, "A review of low-power vlsi technology developments," *Innovations in Electronics and Communication Engineering: Proceedings of the Fifth ICIECE 2016*, pp. 17–27, 2018.

[37] H. Kumar and V. Tomar, "A review on performance evaluation of different low power sram cells in nano-scale era," *Wireless Personal Communications*, vol. 117, no. 3, pp. 1959–1984, 2021.

[38] T. R. Patnala, S. Majji, and G. K. Pasumarthi, "Optimization of csa for low power and high speed using mtcmos and gdi techniques," *International Journal of Engineering and Advanced Technology (IJEAT)*, vol. 8, no. 5S3, 2019.

[39] D. Peterson and O. Bringmann, "Fully-automated synthesis of power management controllers from upf," in *Proceedings of the 24th Asia and South Pacific design automation conference*, 2019, pp. 76–81.

[40] [Online]. Available: https://www.synopsys.com/verification/static-and-formal-verification/vc-lp.html

[41] A. B. Mehta, "Asic/soc functional design verification," *Publ. Springer*, 2018.

[42] [Online]. Available: https://www.synopsys.com/verification/solutions/low-power/low-power-verification.html

[43] A. Kamath, B. Kumar, S. Aggarwal, S. Parameswaran, M. Goyal, P. Lonkar, and S. Sreenath, "A comprehensive multi-voltage design platform for system-level validation of standard cell library," in *2021 22nd International Symposium on Quality Electronic Design (ISQED)*. IEEE, 2021, pp. 285–291.

[44] P. Khondkar, G. Chidolue, and P. Yeung, "Low power coverage: The missing piece in dynamic simulation," *February March, DVCon*, 2018.

[45] M. Groma and D. Macko, "Simplified introduction of power intent into a register-transfer level model," *Design Automation for Embedded Systems*, vol. 25, no. 4, pp. 297–324, 2021.

[46] V. Melikyan, M. Martirosyan, D. Babayan, V. Janpoladov, and D. Musayelyan, "Multi-voltage and multi-threshold low power design techniques for orca processor based on 14 nm technology," in *2018 IEEE 38th International Conference on Electronics and Nanotechnology (ELNANO)*. IEEE, 2018, pp. 116–120.

[47] D. Babayan, E. Babayan, P. Petrosyan, A. Tumanyan, E. Kagramanyan, and T. Hakhverdyan, "1.9 ghz 1.05 v 16-bit risc core for high density and low power operation in 28nm technology," in *2016 IEEE East-West Design & Test Symposium (EWDTS)*. IEEE, 2016, pp. 1–4.

[48] K. Sharma and P. K. Dahiya, "Design and implementation of 16-bit risc processor on zed board," *IUP Journal of Information Technology*, vol. 17, no. 2, pp. 50–64, 2021.

[49] Z. Li, W. Hu, and S. Chen, "Design and implementation of cnn custom processor based on risc-v architecture," in *2019 IEEE 21st International Conference on High Performance Computing and Communications; IEEE 17th International Conference on Smart City; IEEE 5th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*. IEEE, 2019, pp. 1945–1950.

[50] P. Trivedi and R. P. Tripathi, "Design & analysis of 16 bit risc processor using low power pipelining," in *International Conference on Computing, Communication & Automation*. IEEE, 2015, pp. 1294–1297.

[51] D. Babayan, "Power optimization approach of orca processor for 32/28nm technology node," in *2015 Computer Science and Information Technologies (CSIT)*. IEEE, 2015, pp. 11–14.

[52] C. Venkatesan, M. T. Sulthana, M. Sumithra, and M. Suriya, "Design of a 16-bit harvard structure risc processor in cadence 45nm technology," in *2019 5th International Conference on Advanced Computing & Communication Systems (ICACCS)*. IEEE, 2019, pp. 173–178.

[53] S. Bhagat and S. Bhandari, "Design and verification of 16-bit risc processor using systemverilog." *I-Manager's Journal on Circuits & Systems*, vol. 6, no. 4, 2018.

[54] G. K. Dewangan, G. Prasad, and B. C. Mandi, "Design and implementation of 32 bit mips based risc processor," in *2021 8th International Conference on Signal Processing and Integrated Networks (SPIN)*. IEEE, 2021, pp. 998–1002.

[55] A. Yadav and V. Bendre, "Design and verification of 16 bit risc processor using vedic mathematics," in *2021 International Conference on Emerging Smart Computing and Informatics (ESCI)*. IEEE, 2021, pp. 759–764.

[56] D. Soundari, M. S. Ganesh, I. Raman, and R. Karthick, "Enhancing network-on-chip performance by 32-bit risc processor based on power and area efficiency," *Materials Today: Proceedings*, vol. 45, pp. 2713–2720, 2021.

[57] M. Mythili, S. D. Badiger, J. Singh, and V. R. Totakura, "A review report on multi-voltage rule check and formal verification of asic design," *Journal of VLSI Design and its Advancement*, vol. 3, no. 2, 2020.

[58] S. S. Omran and A. K. Abdul-abbas, "Design of 32-bits risc processor for hardware efficient qr decomposition," in *2018 International Conference on Advance of Sustainable Engineering and its Application (ICASEA)*. IEEE, 2018, pp. 69–73.

Dheeraj Kumar Sharma received his B.Tech (Bachelor of Technology) degree in Electronics Engineering from Aligarh Muslim University, Aligarh, Uttar Pradesh, India in 2007. He received M.Tech (Master of Technology) in Communication Systems from Indian Institute of Technology Roorkee in 2009. He received Ph.D (Doctor of Philosophy) in Electronics and Communication Engineering from National Institute of Technology Kurukshetra, Haryana, India in 2019. Presently, he is working as Assistant Professor in Department of Electronics and Communication Engineering from year 2014 onwards. He has 15 research publications in various journals and conferences.

Rahul Vikram received his M.Tech (Master of Technology) in VLSI Design from National Institute of Technology Kurukshetra in 2021. Presently, he is working as Engineer in Qulacomm Pvt. Ltd.