IMPACT OF VOLTAGE SCALING ON RISC-V CORE PERFORMANCE: A STUDY OF NEAR-THRESHOLD AND SUB-THRESHOLD OPERATION IN 45NM TECHNOLOGY

by

SAHITHI JEEDIMALLA

A thesis submitted to the faculty of
The University of North Carolina at Charlotte
in partial fulfillment of the requirements
for the degree of Master of Science in
Electrical Engineering

Charlotte

2025

Approved by:

_____
Dr. Jeremy Holleman

_____
Dr. Ronald Sass

_____
Dr. Fareena Saqib

ABSTRACT

SAHITHI JEEDIMALLA. Impact of Voltage Scaling on RISC-V Core Performance: A Study of Near-Threshold and Sub-Threshold Operation in 45nm Technology. (Under the direction of Dr. JEREMY HOLLEMAN)

This thesis presents the physical design and analysis of a RISC-V core operating at near-threshold and sub-threshold voltages (0.45V–1.0V) using 45nm CMOS technology. Standard cell libraries were characterized at multiple voltages and PVT corners using Cadence Liberate, revealing significant timing degradation and failures below 0.6V. The generated Liberty files enabled full synthesis-to-signoff flows using Cadence Genus and Innovus, including logic synthesis, floorplanning, placement, CTS, routing, STA, and IR drop analysis. Timing, area, leakage, and dynamic power were evaluated at 1.0V, 0.9V, and 0.8V, with 0.9V–1.0V emerging as the optimal energy-timing trade-off. Physical design flow failed at 0.7V and 0.6V due to invalid cell characterizations, but 0.7v successfully ran at low clock speed with many glitch violations . This work provided key insights into the real-world challenges of low-power design, including sub-threshold timing degradation, library limitations, and placement issues at ultra-low voltages. These observations sparked a strong technical curiosity, motivating future work toward re-characterizing standard cells at 0.6V and 0.7V, and exploring adaptive voltage techniques for better energy-efficient processor design.

DEDICATION

I dedicate this thesis to the pillars of strength and inspiration in my life. To my professor, whose mentorship has shaped my academic journey in ways words cannot fully capture. Your guidance, patience, and dedication to excellence have been a beacon throughout this research. I am deeply grateful for the knowledge you have shared and the confidence you've instilled in me. To my family, who have been my unwavering source of love and encouragement. Your sacrifices, prayers, and constant support have given me the strength to overcome every obstacle. This accomplishment is as much yours as it is mine. To my friends, thank you for being my cheerleader, my sounding boards, and my safe space during moments of doubt. Your companionship made this journey lighter, and your faith in me made it brighter. Each page of this work carries a part of your contribution. I am thankful forever.

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to my advisor, Dr. Jeremy Holleman, for his continuous guidance, encouragement, and support throughout the course of this thesis. His ability in low-power design and his insightful feedback at every stage have been instrumental in shaping the direction and success of this work. I would also like to thank Professor Saqib and Professor Sass for serving on my thesis committee. Their valuable comments and suggestions helped enhance the depth and clarity of this research. I am thankful to the Department of Electrical and Computer Engineering at the University of North Carolina at Charlotte for providing the resources and facilities necessary to conduct this work, including access to industry-standard EDA tools and research infrastructure. Lastly, I am eternally grateful to my family for their unconditional love, understanding, and encouragement. Their support has been the foundation of my academic pursuits.

TABLE OF CONTENTS:

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

NTC : Near-Threshold Computing

STC : Sub-Threshold Computing

ISA : Instruction Set Architecture

RTL : Register Transfer Level

GDSII : Graphic Design System II (a file format for IC layout)

PVT : Process, Voltage, Temperature

SS : Slow-Slow (process corner)

FF : Fast-Fast (process corner)

CTS : Clock Tree Synthesis

CHAPTER 1: INTRODUCTION

Power efficiency is currently a central concern in modern computing, particularly for IoT, edge computing, and ultra-low-power embedded systems. With these systems demanding more computational power under stringent energy constraints, energy-efficient processor design is increasingly needed.

One of the promising methods is voltage scaling, or near-threshold (NTC) and sub-threshold (STC) computing, where supply voltages are reduced near or even below the transistor threshold voltage. Yet, digital circuit operation at such low voltages causes severe problems, including timing degradation, delay increase, IR drop, and heightened sensitivity to process variation. This thesis explores voltage-aware physical design of a RISC-V RV32I core under various supply voltages, i.e., 1.0V (nominal) and between 0.95V to 1.0V. RISC-V instruction set architecture (ISA) was selected as it is open-source and gaining wider adoption in low-power computing. The core designed in this thesis is an extension of the `iiitb_rv32i` open-source RV32I core design that was chosen due to its simplicity, synthesizability, and conduciveness for use in embedded systems [1].

One of the biggest problems faced was the unavailability of pre-characterized low voltage standard cell libraries. To address this issue, re-characterization of standard cells was done across a voltage range from 0.55V to 0.95V using Cadence Liberate, and this created Liberty (.lib) files with precise timing and power data. These libraries were subsequently employed during logic synthesis and complete physical implementation with Cadence Innovus and Voltus. This thesis presents a complete comparison of design outcomes at each voltage point regarding timing closure, leakage

power, dynamic power, area overhead, and overall design feasibility. The study aims to present a valuable reference for voltage-aware digital design and physical implementation using real EDA tools in an academic setting.

1.1 Problem Statement

Energy efficiency is now a central issue in modern processor design, particularly in areas such as mobile, embedded, and IoT platforms where power budgets are constrained. [2] Voltage scaling is among the most effective ways to reduce power consumption since dynamic power dissipation decreases quadratically with supply voltage. Digital circuits in near-threshold and sub-threshold voltage regimes introduce catastrophic problems regarding performance, timing reliability, and variability tolerance. This thesis addresses these difficulties by performing a complete physical design realization of the open-source RISC-V RV32I processor core iiitb_rv32i from IIIT Bangalore. The 5-stage pipelined core implements a subset of base instructions including arithmetic, branch, and memory instructions. It is synthesized from Verilog RTL, functionally simulated using Icarus Verilog and GTKWave, and passed through the full RTL-to-GDSII digital backend flow with Cadence tools. All experiments are performed over scaled voltages (1.0V, 0.9V, 0.8V, 0.7V) to analyze power-performance trade-offs and physical design constraints[1]. The main goal of this research is to examine the effect of voltage scaling on timing closure, power dissipation, and layout feasibility applied to a realistic synthesizable core. By characterizing standard cell libraries at low voltages using Cadence Liberate

and simulating the resultant netlist at each voltage point in Cadence Innovus, this research provides quantitative insights regarding the feasibility of low-power design for RISC-V cores for the near and sub-threshold regimes.

1.2 Processor Architecture of RV32I

In this thesis, the processor used is a five-stage pipelined, single-cycle RV32I instruction set architecture-based core. As can be prominently seen in Figure 1.1, a very detailed graphical view of the `iiitb_rv32i` processor architecture is visible, showing the movement of instruction and data among the major blocks like Program Counter, Instruction Memory, Register File, ALU, Data Memory, and the related control logic.

As part of an open-source educational project at IIIT Bengaluru (Bangalore), the iiitb_rv32i core has all the stages of instruction execution: Instruction Fetch (IF), Instruction Decode (ID), Execute (EX), Memory Access (MEM), and Write Back (WB). All the stages are independent functional and coupled by pipeline registers that store state and data. The core component of the design is a 32-bit ALU for executing arithmetic, logic, shift, and comparison instructions. Instruction memory is connected to the IF stage, and a general-purpose 31-register register file performs ID, EX, and WB stage. The MEM stage employs the data memory interface to execute load/stores. During decoding, the control signals are generated and forwarded ahead for hazards detection and execution.

One of the more complex sections of the design is the register file, which needs to support concurrent reading and writing from/to pipeline stages. Communication

bottlenecks occur most directly on data buses of interfaces between the EX and MEM pipeline stages and interfaces between the MEM and WB pipeline stages, where they need to transfer full 32-bit results and memory addresses. Also, forwarding and hazard detection logic ensures control complexity and avoids incorrect sequence of execution of associated instructions. This processor's longest data paths are usually from the register file to the ALU and then memory, especially in instructions where branch operation and memory access are required. These kinds of paths are heavily involved in obtaining the shortest possible clock period as well as overall processor performance. Timing closure can only be obtained by proper planning of control logic and multiplexers, datapaths with balance, and proper pipelining. This is particularly so in voltage scaling, which was investigated in this thesis. Being an open-source implementation, the `iiitb_rv32i` core is highly flexible and can be tailored to implement changes in instruction decoding, pipeline depth, memory interfaces, and data path width. This made it an ideal candidate for voltage-aware experiments, where different configurations could be tried on different PVT conditions without being encumbered by licensing issues. The RV32I ISA used includes basic computational, memory, and control flow instructions that support arithmetic, logical, shift, load/store, and branch operations, keeping the core simple but educationally complete.

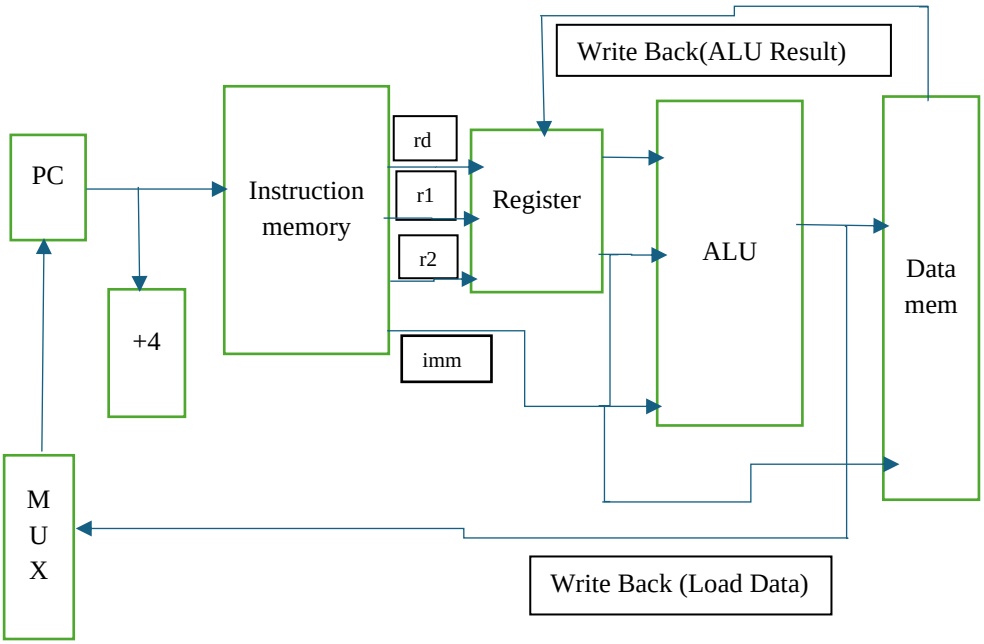


Figure 1.1: Block Diagram Of iiitb_rv32i

1.3 Interconnect Complexity and Long Bus Requirements

In the iiitb_rv32i core long buses are required to facilitate communication between temporally distant pipeline stages and shared resources like the register file and chunks of memory. The Register File is read by the Decode stage (to fetch source operands) and written by the Write Back stage (to store results), and therefore, bidirectional 32-bit buses spanning multiple stages are required. These buses need to be high fanout and low latency to prevent critical path delays, so they are physically long and timing-critical in design.

Instruction Type	RV32I Instructions	Functions
1.Computational (9)	add, addi, sub, sll, srl, and, or xor, slt	Perform addition and subtraction (immediate value is a 12-bit signed value). Perform logical left and right shifts. Perform bitwise operations. Set if less than (slt): sets the destination register to 1 if the first operand is less than the second.
2. Control Flow (2)	beq, bne (Branch if not equal)	Perform comparisons and, if satisfied, transfer control to the address offset provided at the 12-bit signed immediate value.
3. Memory Access (2)	lw, sw	Load a 32-bit word into the destination register (lw). Store a 32-bit word (sw).

Table 1.1: Instruction Set of iiitb_rv32i

In addition, buses between the ALU (during the Execute stage) and the memory and Write Back stages are 32-bit operands and results, often forming the critical data path in the pipeline. The Instruction Fetch and Memory Access stages also depend on particular buses to fetch and cache 32-bit instructions or data, often in parallel to ongoing computation. These buses stretch across the datapath and are likely to be sites of congestion, especially during clock tree synthesis and place-and-route phases of the physical design flow.

Besides, forward flows implemented to recover data hazards (e.g., bypassing ALU results back to the Decode or Execute stage) provide long and high-speed point-to-point data buses, which increase interconnect complexity. Therefore, buses between Register File, ALU, Memory, WB units are among the longest and timing-critical components, necessitating floorplanning attention, buffering, and routing techniques.

Major high-performance blocks in the iiitb_rv32i RISC-V processor are the Execution Unit and the Memory Access stage since these perform ALU operations, address computation, and data memory access. Such blocks involve conditional logic, data forwarding, and hazard resolution and thus constitute fundamental blocks within performance and correctness. Critical communication bottlenecks appear on the register file buses, which get accessed in parallel at the decode and write-back stages. Since each instruction can have two sources of operands and one destination register, the register file must do multiple reads and writes in each clock cycle, necessitating wide and long data buses. The interface to data memory also contributes latency and bandwidth concerns during load/store instructions. These communication paths, especially between the EX/MEM and MEM/WB phases that require long buses, and their saturation could impact pipeline efficiency, particularly at high frequencies or when tighter timing constraints are imposed. Having these buses efficiently routed and timed correctly is critical in achieving design closure during physical implementation.

CHAPTER 2: RISC-V CORE DESIGN AND FUNCTIONAL VERIFICATION

Functional verification was a significant step in validating the correctness of the iiitb_rv32i open-source RISC-V rv32i processor before synthesis and physical design flow. The verification flow made use of Icarus Verilog for Verilog RTL compilation and GTKWave for waveform examination in the original work performed by the author.

A compliant testbench exercised the processor with a sequence of representative instructions such as arithmetic, logic, load/store, and control-flow instructions to stress all pipeline stages under actual utilization. Simulation outputs were examined cycle-by-cycle with GTKWave to validate proper instruction decoding, registering file updating, memory access, and data forwarding in the 5-stage pipeline. Observations of branch behavior, data hazards, and instruction ordering were also targeted in observations to ensure the control and data paths exhibit proper RISC-V semantics. Early functional verification here played a significant role to catch design-time bugs before their synthesis implemented them, ensuring thus RTL description logically as solid and synthesizable-ready. Functional verification of the open-source iiitb_rv32i RISC-V RV32I processor core was implemented using Cadence Xcelium, an industry-standard digital logic simulation tool. Verification was carried out in a way that RTL design was ready to execute before synthesis and physical design steps. Source code of the design was checked out from the official GitHub repository (https://github.com/vinayrayapati/iiitb_rv32i). The repository

holds thoroughly documented supported instructions, such as arithmetic instructions (add, sub, and, or xor, slt), immediate variants (addi), memory access instructions (lw, sw), and control flow instructions. A custom-made Verilog testbench emulated a set of these instructions passing through the 5-stage pipeline of the processor: Instruction Fetch (IF), Instruction Decode (ID), Execute (EX), Memory Access (MEM), and Write Back (WB). Register values, memory contents, and ALU outcomes were traced step by step in this work before using it for voltage scaling analysis. Cadence Xcelium's native waveform viewer, Sim Vision, was used to view internal signals. This allowed the ability to verify instruction execution, pipeline flow control, and correct register updating for each instruction. Simulation waveforms and two instruction sets were documented for verification.

Case-1: From Figure 2.1, program counter (PC) value 20 is interpreted as getting instruction ADD operation between registers r1 and r2. Instruction memory wise, the 32-bit word `0x02208300` represents the general RISC-V ADD instruction format with opcode, funct3, and funct7 fields set correctly. At the reset, the processor initializes register r1 with 1 and register r2 with 2. An addition between the two values (1 + 2) results in 3. As would be expected in a pipelined processor, the initial `WB_OUT` value is not the result immediately after the fetch because of the multiple pipeline processing stages. However, later on around four or five clock cycles, `WB_OUT` is rightly updated to 3, and this is indeed the correct ADD operation output. This checks the instruction decode, ALU execute, and write-backstage sequences for correct ADD operations.

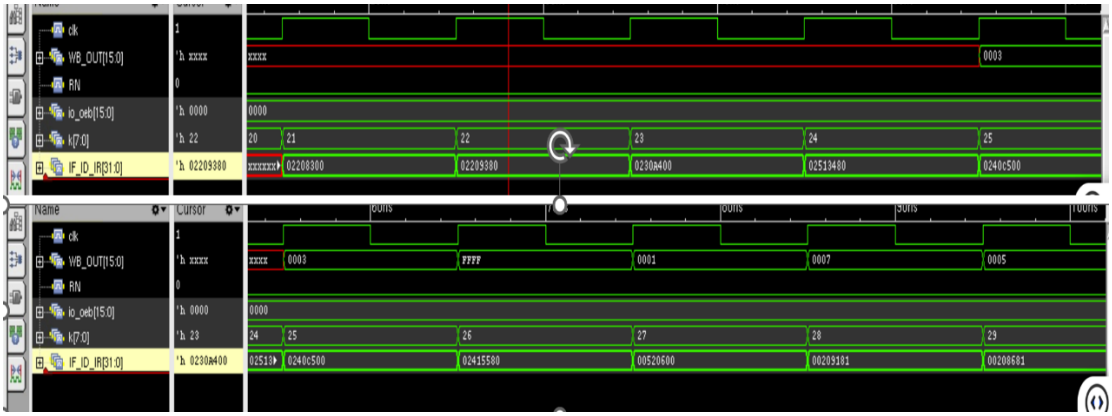


Figure 2.1 Instructions with two study cases

Case-2: The instruction loaded at program counter (PC) address 24 is an XOR operation between registers r1 and r4, as interpreted. From the instruction memory values, the 32-bit word loaded `0x0240C500` is consistent with the RISC-V instruction encoding for XOR with funct3, opcode, and funct7 fields being the standard encoding of the XOR operation. The initial register values of the processor also set r1 = 1 and r4 = 4 on reset. A bitwise XOR of these values (1 ^ 4) produces 5. Due to pipeline latency, the immediate value one observes on the `WB_OUT` signal after instruction fetch may not be the XOR result. Within three or four clock cycles, however, the correct value of 5 appears on `WB_OUT`, as expected from the XOR result. Thus, instruction decoding, execution, and result monitoring all verify that the processor executed the XOR operation correctly at PC = 24.

This functional verification step confirms that the core design is compliant with the RV32I instruction set and behaves as anticipated in typical test environments. After this library characterization implemented on different voltages.

CHAPTER 3: LIBRARY CHARACTERIZATION USING LIBERATE

3.1 Voltage-Aware Liberty Characterization Analysis

In the design of digital integrated circuits, accurate timing and power analysis are essential for proper implementation. Standard cell libraries are the foundational elements utilized for synthesis and physical design and must be characterized at the operating voltages targeted for effective analysis. [4] For this work, designed for near-threshold and sub-threshold voltage operation, one had to generate custom Liberty `(.lib)` files at different supply voltages based on Cadence Liberate.

The nominal `gpdk045` cell library was used as the basis. While pre-characterized Liberty files are supplied at nominal voltage (1.0V), they do not represent timing or power behavior for lower voltages like 0.45V-0.9V. Therefore, the full characterization flow was run at these voltages: 0.95, 0.85, 0.75, 0.65, 0.55V, and 0.45V. For each voltage corner, Cadence Liberate was used to run SPICE-level simulations for a matrix of output capacitive loads and input transition times. The tool then extracted fine-grained delay, power, and constraint information to populate.lib files.

The characterization flow involved the following steps:

1. Preparing SPICE netlists from the `.cdl` file of the GPDK045 standard cells.
2. Defining voltage, temperature, and PVT corners in Liberate using `char.tcl` and `settings.tcl` scripts.
3. Automating stimulus generation and running simulations for each cell.
4. Extracting setup/hold transition, cell rise/fall times, and leakage power.

5. Generating validated `.lib` files at each voltage corner using automation tcl scripts `char.tcl, settings.tcl, template.tcl, cells.tcl, and Makefile` to run at different voltage corners at 125ºC for slow-slow corner and negative 40ºC (n40ºC) for fast corner.
6. `char.tcl`: Main Characterization Script for library generation.
7. The `char.tcl` script is the primary control file used in the Cadence Liberate flow to initiate and configure the characterization of a standard cell library. It defines user-defined variables such as file paths, voltage, temperature, and references to other supporting scripts and templates like sourcing `templates.tcl, cells.tcl and settings.tcl`.

   Below is a breakdown of the main parameters:

   a. `SRC_DIR` and `RUN_DIR`: Define the source directory for input files and the directory to store output files (typically set to the current working directory).
   b. LIB: Name of the library being characterized.
   c. PROCESS: Specifies the process corner (e.g., tt for typical-typical, ff for fast-fast, ss for slow-slow).
   d. VDD: Supply voltage for the characterization (e.g., 1.0 V).
   e. TEMP: Operating temperature for simulation (e.g., 70°C).
   f. SETTINGS FILE: Path to an auxiliary configuration file `(settings.tcl)` for tool-specific options.

g. TEMPLATE FILE: Path to a template file used during simulation setup.

h. CELLS FILE: Tcl file listing cells that need to be characterized.

i. NETLIST DIR: Directory containing SPICE/CDL netlist files.

j. USERDATA: Optional Liberty file used as a reference or baseline.

k. This script also supports overriding variables from the command line when launching Liberate, allowing flexible control for batch processing or parameter sweeps.

8. The `settings.tcl` file sets Liberate-specific variables, especially related to the external SPICE simulator (Spectre) and internal tool behavior. This script is sourced within `char.tcl` to apply these configurations globally during characterization.

   The key sections include:

   a) Tool Version Parsing: Extracts and stores the tool version from Liberate to manage version-specific behaviors.

   b) Spectre Simulation Options:

      i. `extsim_cmd_option`: Sets runtime options like `+aps`, `-mt`, `+liberate`, and simulation time limits `(+maxns=20u)`.

      ii. `extsim_deck_header`: Adds headers to SPICE decks for simulation options and model languages.

iii. `extsim_option` and `extsim_leakage_option`: Define simulation control flags such as handling of parameter redefinition and hierarchical ambiguity.

c) SKI Settings:

i. `ski_enable`: Enables the Shared Knowledge Interface (SKI) used to manage simulation data sharing.

ii. `ski_clean_mode`: Ensures old semaphores are cleaned before new runs.

iii. `power_tend_match_tran`: Ensures consistency in waveform analysis by aligning power and transition end times.

9. The template file is an important configuration script used when characterizing a library using Cadence Liberate. It defines how delay and power values are measured for various input transition times and output load conditions. Specifically, it defines measurement windows for input slew (slew lower and slew upper values typically spanning 0.1 to 0.9 of VDD) and delay (delay in and delay out referenced at 50% transitions).
10. The script also constrains the legal span of transitions through `min_transition` and `max_transition` and sets minimum output load capacitance. Delay and power templates for standard delays and powers are created with 2x2 index tables, under which characterization will

be performed in a small representative grid of conditions. There is a complete inventory of standard cells (e.g., INVX1, NAND2X1, DFFX1), and for every one of those, the pins of input and output and corresponding templates are defined. Cells are only considered if they are tagged as active in that run. Overall, the template file modularizes the template definition logic such that there is a controlled and consistent method of generating Liberty timing and power models.

11. The `cells.tcl` file explicitly defines all the cells which are to be used for the characterization process. It uses conditional blocks to check whether each cell exists (`ALAPI_active_cell`) before it defines the characterization parameters of the cell. For each existing cell, the script sets its input and output pins, pin list, and assigns the cell a predefined delay and power template. This module approach guarantees that only valid, simulation-compatible cells are generated by Liberate. It also allows selective cell inclusion or exclusion for increased runtime efficiency or simulation failure avoidance.
12. Makefile is an automation layer that controls when and how Liberate initiates generation for different voltage corners. It generates targets such as `char_tt` for normal corners and `char_all` to support parallel characterization of multiple corners (e.g., SS at 0.55V, FF at 0.65V, etc.). Each target dynamically overrides process, voltage, and temperature variables while calling `char.tcl`, thus supporting flexible multi-corner execution without code redundancy. This batch-oriented design allows for

characterization of libraries at various PVT conditions and provides uniform log tracking and output organization. This file modularizes simulator options from the main characterization logic and makes the flow more manageable and reusable across different PVT corners.

13. Several issues arose at lower voltages, particularly below 0.45V-0.65V, where certain cells failed to show valid timing arcs, and almost half of the cells failed. The gpdk045 has 505 cells in its library and at 0.45 V, 0.55 V 184 cells failed. Even after increasing simulation time faced few errors and failed to recharacterize. So, all the libraries generated at this point are analyzed.

As a result, for example, inverter and buffer cells exhibited hugely oversized delays or nonexistent transitions completely and hence were not synthesizable by CTS for clock tree synthesis at physical design. This introduced a very wide bottleneck towards overall synthesis, place and route closure, as well as floorplanning, at 0.55V and 0.65V. However, other voltages `(.lib)` files were still convenient during the exploration of power scaling as well as delay scaling tendencies through voltages and in the netlist synthesis at gate level. This chapter provides the foundation for understanding the effect of voltage scaling on building blocks of digital logic and provides the framework necessary for multi-voltage-aware synthesis and implementation.

The characterization of standard cell libraries across multiple voltage levels and process corners offers valuable information on timing behavior, particularly for delay-critical digital blocks. In this work, libraries were established at voltages ranging

from 0.45 V to 0.95 V at both slow-slow (ss, 125°C) and fast-fast (ff, -40°C) corners. The results show how aggressive voltage scaling and environmental conditions affect standard cell performance, with implications for synthesis, clock tree synthesis (CTS), and overall timing closure.

At 0.45 V under the SS corner, cell delays are extremely costly. For instance, the INVX1 cell that is widely used in inverter chains and buffers has rise delay values over 10 ns for normal fanouts and loading. Similarly, the NAND2X1 gate has 12–15 ns delays, while some other cells have over 100 ns delay which is several times worse than its nominal performance. These tremendous delays result in extreme setup time violations and render the design not feasible for actual implementation. This voltage level is best left for power modeling or near-threshold research, but not for standard synthesis or place-and-route flows.

Increasing the voltage to 0.55 V (SS) provides a modest reduction in delay, but the advantage is insufficient for implementation. For the same INVX1 cell, the increase delay reduces only slightly still being well above 8 ns while multi-input logic like AOI21X1 continues to see large transition times. These cells would likely be flagged as timing constraint violators even with relaxed setup budgets. At 0.65 V (SS), the INVX1 delay improves, typically reducing to the 4–6 ns range. Although this is a significant improvement, it still poses problems for designs that contain critical timing paths. Such a library might be minimally usable for very low-speed digital logic but remains risky for general usage. But because of other cells which has 100s delay at this point caused many errors and violations during physical design flow.

If voltage is increased to 0.75 V at the SS corner, cells begin to experience reasonable delay values. The INVX1 cell, for example, goes down to around 2–3 ns for the rising transition, and the NAND2X1 cell shows similar reductions. CTS is possible at this stage, and designs can be closed with modest effort. This voltage level is a compromise between power reduction and functionality at worst-case process conditions. At 0.85 V (SS), most cells show near-nominal timing behavior. The INVX1 gate typically has rise delays of around 1.5–2 ns, and the logic gates like MUX2X1 and AOI22X1 typically function under a wide range of loads and transition conditions. Consequently, this is the most suitable SS corner for full physical implementation.

At -40°C FF corner, the delay values are reduced significantly due to the enhanced speed of the transistor and reduced threshold voltage. At 0.55 V (FF), INVX1 delay is already shorter than its counterpart at 0.75 V under the SS corner, typically below 4 ns. This enables limited functionality even at lower voltages, particularly in low-frequency or non-timing-critical domains. At 0.65 V (FF), delays continue to improve. For example, INVX1 and NAND2X1 show stable delay numbers in the 2–3 ns range, for which this corner is amenable to energy-efficient block-level implementation.

At 0.75 V (FF), most cells are at their optimal operating point for low-power systems. INVX1 delay can be as low as 1.2 ns, and MUX2X1 and OAI21X1 gates are good across all test conditions. Delay values are excellent by the time voltage is at 0.85 V (FF), with many facilitating aggressive frequency targets and setup constraints. The INVX1 gate, for example, has around 0.9–1.2 ns for both rising and falling

transitions, showing very high suitability for physical design closure. Finally, at 0.95 V (FF), cells have their lowest delays, often under 1 ns for simple logic. This is the best corner for best-case timing analysis and reference implementation, especially for high-speed blocks.

In summary, the delay behavior of key standard cells like INVX1, NAND2X1, and MUX2X1 is extremely voltage- and process-dependent. At SS conditions, functional timing is observed only at or above 0.75 V, while at FF conditions, voltages as low as 0.65 V can support functional designs. This work emphasizes the necessity of selecting suitable corners based on the timing-criticality of a design and the power saving vs. achievable performance trade-off.

3.2 Failure of Standard Cells at 0.45 V- 0.65 V (SS, 125°C)

At ultra-low voltages such as 0.45 V-0.55 V, particularly in the slow-slow (SS) corner condition at high temperature (125°C), an overwhelming majority of standard cells become functionally unusable. The fundamental reason is the combined effect of low supply voltage, high threshold voltage, and slow transistor switching, all of which are worsened by the worst-case SS corner and high temperature.

A closer inspection of key standard cells reveals why this failure occurs.

1. INVX1 (Inverter): At 0.45 V, the delays of INVX1 cells are greater than 10 ns in some cells and greater than 100ns in other cells. This is largely because the overdrive voltage `(Vgs - Vth)` is considerably reduced, which results in negligible current through the pull-up or pull-down networks. The charges or discharges of the output node are slow, especially

when capacitive loaded, and produce unacceptably high transition delays. At 0.55 V, latency remains greater than 8 ns and more than 100 ns are short of being viable for digital implementations at even modest clock speeds.

2. NAND2X1 (2-input NAND): Stacked NMOS or PMOS logic gates (such as NANDs and NORs) worsen further at low supply voltages due to headroom in voltage. At 0.45 V, NAND2X1's delay is much worse than that of INVX1, whose rise delays are as high as 12-14 ns due to not just weak drive but also less switching in the stacked transistors. Even at 0.55 V, while fewer, delays remain in the 9-11 ns range. Furthermore, functional failure or sluggard’s transitions may also prevent full logic level swings, especially during driving downstream gates.
3. MUX2X1 and AOI/OAI Gates: Larger up/down stacks show functional issues at 0.45 V. In simulation, the output may not completely resolve to logic '0' or '1' within the required window, especially under realistic loads. These cells either show invalid logic levels or cause setup and hold time violations in flip-flops due to incomplete transitions.
4. DFFPOSX1 (Positive-edge Flip-Flop): At both 0.45 V and 0.55 V, flip-flops show not just delay degradation, but reliability and hold time violations. Their internal clock-to-Q paths are too slow, and setup windows are excessively wide, making it nearly impossible to meet timing in real-time operation. Register chains built with such flops become unstable or nonfunctional.

5. While the delays in nanoseconds or picoseconds can be managed and try to get the core synthesized and design for power and timing analysis, but because of the other cell delays above 100s made it very difficult to implement at 0.45 V- 0.65 V.

3.3 Low-Voltage Cell Failures

The failure of standard cells at 0.45 V and 0.55 V under SS 125°C conditions is primarily due to:

1) Reduced carrier mobility and drive current in MOSFETs
2) Increased threshold voltage limiting switching activity
3) Incomplete or delayed signal transitions under capacitive loading
4) Poor performance of stacked transistor configurations
5) Severe timing violations in sequential cells (e.g., flip-flops)
6) The higher delay values like 100 seconds might be a invalid values given by tool or some simulation error.

As a result, these libraries are not used for any timing-critical or synthesized logic.

3.4 Conclusion of Library Characterization

1. Energy Efficiency vs. Performance Trade-off: While voltage scaling reduces dynamic power significantly, the consequent increase in gate delay poses intimidating timing closure problems. For instance, 0.45 V standard cells on worst-case corners have delays greater than 10 ns-100s, making

them unsuitable for real-time processing applications. Designers must achieve an exact balance between voltage scaling and performance needs to avoid unsustainable operating points.

2. Technology Limitation of Unoptimized Libraries: The findings made using the gpdk045 standard cell libraries highlight that traditional libraries are not optimized for near-threshold or sub-threshold operation. Unlike commercial low-voltage libraries, gpdk045 lacks high-drive or low-`Vth` versions and adaptive biasing methods. This limitation explains the failure of most cells at lower voltages, despite the theoretical benefits of voltage scaling being asserted.
3. Impact on Clock Tree Synthesis (CTS): CTS is unsuccessful at voltages such as 0.45 V-0.65 V since buffer cells that can be used are not available. The significantly degraded delay characteristics pose difficulties in building balanced and functional clock trees. This is significant since even a functional data path cannot work correctly without a reliable clock network.
4. Leakage-Dominated Design Regions: As frequency reduces in low-voltage regimes, leakage power plays a significant role in total power consumption. In always-on domains or power-gated areas, special design methods such as multi-threshold libraries, clock gating, and body biasing may be needed to minimize energy wastage.
5. Relevance to Near-Threshold and IoT Design: Such findings are particularly relevant for near-threshold and ultra-low-power computing,

such as IoT nodes or biomedical electronics, where the speed is compromised for lower power consumption. However, making functioning possible at sub-nominal voltages dependable can require recreating libraries or using adaptive voltage scaling techniques.

CHAPTER 4: SYNTHESIS AT LOW VOLTAGES

4.1 Synthesis Constraints and Setup

The RV32I core synthesized using an automated RTL-to-gate-level synthesis flow across different supply voltages: 0.6 V, 0.7 V, 0.8 V, 0.9 V, and 1.0 V. The multi-voltage approach enabled the investigation of timing and power trade-offs under different operating modes, particularly near-threshold and nominal cases. To guide the synthesis process and timing analysis, several constraint sets were defined using Synopsys Design Constraints (SDC). These restrictions comprised clock definitions, uncertainty margins, I/O delays, transition limits, and output loads optimized for various frequencies of the clock. Baseline synthesis performed for a 10 ns clock period.

The constraints from Figure 4.1.1 were utilized to synthesize the RV32I core with a target clock period of 10 ns, in the hope of verifying the design under loose timing constraints.

1. Clock Definition: A primary clock named clock is created on the input port clock, with a period of 10 ns. This defines the timing reference for all sequential elements and enables the synthesis tool to analyze timing paths correctly.
2. Clock Uncertainty: Uncertainty values are set to account for variations due to clock jitter, skew, and modeling inaccuracies. The setup uncertainty of 0.4 ns ensures margin for timing closure, while 0.2 ns is allocated for hold analysis.

```
# Clock Definition
create_clock -name clock -period 10 [get_ports clk]

# Clock Uncertainty
set_clock_uncertainty -setup 0.4 [get_clocks clock]
set_clock_uncertainty -hold  0.2 [get_clocks clock]

# Clock Properties
set_clock_transition -max 0.5 [get_clocks clock]
set_clock_transition -min 0.1 [get_clocks clock]
set_clock_latency 0.5 [get_clocks clock]

# Input Constraints
set_input_delay      -clock [get_clocks clock] 5.0 [all_inputs]
set_input_transition -max 0.2 [all_inputs]

# Output Constraints
set_output_delay -clock [get_clocks clock] 5.0 [all_outputs]
set_load         0.05 [all_outputs]

# Design-Wide Constraints
set_max_transition 0.5 [current_design]
set_max_fanout     16  [current_design]
```

Figure 4.1.1: Constraints at 10ns used for Synthesis at 0.6V - 1.0V

3. Clock Transition and Latency: These constraints limit the acceptable slew (rise/fall transition) of the clock signal to ensure signal integrity and reduce skew. A clock latency of 0.5 ns is used to model the delay from the clock source to clock pins of sequential cells.
4. Input Constraints: An input delay of 5.0 ns assumes the external environment launches data midway through the clock cycle. The input transition constraint restricts how slowly input signals can rise/fall, helping to avoid poor gate performance due to sluggish signals.
5. Output Constraints: Output delay models the time budget allocated to downstream circuitry to capture signals, also set to half the clock period (5 ns). A reduced capacitive load of 0.05 pF is assumed for outputs to reflect light fan out or short routing paths in this case.
6. Design-Wide Constraints: These global constraints enforce signal integrity by limiting maximum transition time and fanout per net. Excessive fanout can

degrade performance and increase delay, so it's capped at 16 to guide the synthesis tool toward balanced buffering.

4.2 Synthesis Flow Automation

To ensure consistency across voltage corners and reduce manual intervention, an automated synthesis flow was developed using a Tcl script executed within the Cadence Genus environment. This script takes RTL files, voltage-specific Liberty libraries, and SDC timing constraints as inputs. Key environment variables such as the supply voltage (VDD), base directory, and paths to slow and fast libraries are preloaded to ensure flexibility and reuse across experiments. The flow performs elaboration, generic synthesis, technology mapping, and post-mapping optimization in three distinct stages: *syn_generic, syn_map,* and *syn_opt,* respectively. At each stage, snapshots, reports, and datapath summaries are generated and stored in structured folders named by voltage. The design is synthesized using medium generic effort and high mapping and optimization effort settings. Constraints are read from an SDC file, and common synthesis directives such as *set_dont_use* for scan DFFs and *set_max_fanout* are enforced. The final netlist (iiitb_rv32i_m. v) and timing constraints `(i) iitb_rv32i_m.sdc` are exported, and formal equivalence (LEC) scripts are automatically generated to verify consistency between RTL and synthesized netlists. This automated approach enabled fast iteration and ensured reproducibility across all voltage scenarios (0.6V, 0.7V, 0.8V, 0.9V, and 1.0V).

To analyze the behavior of the RISC-V core across a realistic operating range, voltage corners were strategically chosen around nominal and sub-threshold regions.

For each operating voltage level (e.g., 0.6V, 0.7V, etc.), corresponding slow and fast corners were selected to reflect worst-case and best-case silicon behavior. For example, at 0.6V, the slow corner was characterized at 0.55V (SS, 125°C), and the fast corner at 0.65V (FF, -40°C). This ±50 mV spread mimics process variation across supply extremes and thermal boundaries. Similarly, for 0.7V, the SS and FF corners used VDD values of 0.65V and 0.75V, respectively. This method provides a realistic margin for physical design and ensures that the synthesized and implemented design remains robust under voltage and process variations, particularly for energy-constrained near-threshold applications.

1. At 0.6V: the iiitb_rv32i RISC-V core was synthesized using the slow_vdd0v6 standard cell library at the ss_0.550_125 corner in Cadence Genus. The synthesis targeted a 10 ns clock period and resulted in 17,495 leaf cell instances, including 2,784 sequential and 14,711 combinational cells. The total cell area was 101,730.29 μm², and the overall design area including interconnects was 119,976.91 μm². Due to the reduced supply voltage, the design exhibited significant timing violations, with a worst negative slack of -19,441 ps and a total negative slack (TNS) of -61.2 million ps across 4,460 failing paths. Despite these violations, the synthesis process completed successfully with a peak memory usage of approximately 1.33 GB and a total runtime of about 34 minutes.
2. At 0.7 V: The iiitb_rv32i RISC-V core was synthesized using the slow_vdd0v7 standard cell library at the ss_0.650_125 corner in Cadence Genus. The synthesis targeted a 10 ns clock period and resulted in 20,107 leaf cell instances, comprising both sequential and combinational elements. The total cell area was 95,600.63

µm², and the overall design area including interconnects was 115,910.71 µm². Unlike the 0.6V case, the final synthesis met timing with 0 ps slack and zero failing paths. This shows improved performance and recoverability at 0.7V, where voltage is sufficient to close timing at the given frequency. The synthesis process completed in under 20 minutes with a peak memory usage of approximately 1.33 GB.

3. At 0.8V, the iiitb_rv32i RISC-V core was synthesized using the slow_vdd0v8 standard cell library at the ss_0.750_125 corner using Cadence Genus. The synthesis targeted a 10 ns clock period and achieved full timing closure, with a final slack of +4 ps and zero failing paths. The design contained 9,019 leaf cell instances, with a total cell area of 39,225.35 µm² and net area of 11,427.04 µm², resulting in a combined total area of 50,652.38 µm². Leakage power was measured at 860.75 nW, with 52.1% attributed to sequential elements and 34.1% to logic cells. This voltage corner demonstrated excellent trade-offs in area, power, and timing, completing synthesis in under 8 minutes with peak memory usage around 1.4 GB.
4. At 0.9V, the iiitb_rv32i RISC-V core was synthesized using the slow_vdd0v9 standard cell library at the ss_0.850_125 corner in Cadence Genus. With a clock period of 10 ns, the design successfully achieved timing closure, ending with a final slack of +4 ps and zero failing paths. The synthesized design included 8,772 leaf cell instances, resulting in a total cell area of 37,911.04 µm² and a combined area (including nets) of 48,826.46 µm². Leakage power was measured at 967.99 nW, with 50.2% from sequential elements and 40.9% from logic gates. Synthesis

completed efficiently in under 7 minutes, using approximately 1.41 GB of peak memory. This result reflects a favorable balance of power, performance, and area at 0.9V.

5. At 1.0 V: The iiitb_rv32i RISC-V core was synthesized using the slow_vdd1v0 standard cell library at the PVT_0.9V_125C corner in Cadence Genus. The design targeted a 10 ns clock period and met timing with a final slack of +218 ps and no failing paths. The core consisted of 9,479 leaf cell instances, with a total cell area of 36,951.39 µm² and a total area (including interconnects) of 48,373.20 µm². Leakage power was 814.05 nW, with contributions of 48.7% from sequential elements and 42.3% from logic. The synthesis completed in approximately 6.5 minutes with peak memory usage around 1.39 GB, confirming robust performance and low leakage at the nominal voltage level.

Metric	0.6V	0.7V	0.8V	0.9V	1.0V
Library Corner	ss_0.550_125	ss_0.650_125	ss_0.750_125	ss_0.850_125	ss_0.9_125
Slack (ps)	-19,441	0	4	4	218
Total Negative Slack (ps)	-61,208,586	0	0	0	0
Failing Paths	4,460	0	0	0	0
Leaf Cell Count	17,495	20,107	9,019	8,772	9,479
Total Cell Area (μm²)	101,730.29	95,600.63	39,225.35	37,911.04	36,951.39
Total Area incl. Nets (μm²)	119,976.91	115,910.71	50,652.38	48,826.46	48,373.20
Leakage Power (nW)	2,372.74	2,972.06	860.75	967.99	814.05
Synthesis Runtime (min)	34	19	7.5	6.7	6.5

Table 4.2.1: Synthesis results at different voltages from 0.6 V -1 V

4.3 Synthesis Results and Voltage-Wise Observations

The synthesis of the iiitb_rv32i RISC-V core was performed across five different voltage levels (0.6V to 1.0V) using voltage-specific Liberty libraries. The results reveal distinct behaviors in timing, area, and leakage power as the supply voltage is varied. Below are the key observations at each voltage point, along with a brief discussion of the reasons behind them:

1. At 0.6V: The design did not meet timing, with a worst-case slack of -19,441 ps and 4,460 failing paths. It also had the highest area and leakage power (2,372.74 nW). This is because the standard cells operate in the sub-threshold region, where drive strength is severely limited. To compensate, the synthesis tool used more and larger cells, increasing both area and power.
2. At 0.7V: This was the first voltage level where the design successfully met timing (0 ps slack and 0 failing paths). However, both area (95,600 μm^2) and leakage (2,972.06 nW) remained relatively high. Although the cells begin to switch reliably, the synthesis tool retained conservative sizing and buffering to ensure closure, leading to inefficient power and area usage.
3. At 0.8V: Timing was met with ease (+4 ps slack), and the design saw a sharp drop in area (39,225 μm^2) and leakage (860.75 nW). This indicates that the cells operated efficiently enough for the synthesis tool to aggressively downsize and optimize logic, making this voltage point the most power-efficient while still keeping timing.
4. At 0.9V: The design continued to meet timing comfortably (+4 ps slack). Area decreased slightly to 37,911 μm^2, while leakage rose modestly to 967.99 nW. This behavior suggests that the tool focused on area optimization, while the minor increase in leakage could be attributed to more frequent switching or the use of slightly faster, leakier cells.
5. At 1.0V: The core achieved its best timing margin (+218 ps), the smallest area (36,951.39 μm^2), and the lowest leakage (814.05 nW). At nominal voltage, the standard cells operate at full performance, allowing the tool to use minimal-sized

cells and fewer buffers. This results in the best performance, area, and power, the best PPA trade-off across all voltage levels.

These results clearly prove the impact of voltage scaling on synthesis outcomes and highlight 0.8V(slow VDD=0.75V and fast VDD =0.85V) to 1.0V(slow VDD =0.9 V and fast VDD=1.1 V) as the best regions for implementation, depending on whether power efficiency or performance margin is prioritized. These results also reflect the synthesis tool's internal optimization strategies including cell selection, buffer insertion, and path restructuring which are voltage-dependent due to changes in cell timing and power models.

CHAPTER 5: PHYSICAL DESIGN AT LOW VOLTAGES

This chapter presents the complete physical design implementation of the iiitb_rv32i RISC-V core at various operating voltages ranging from 0.6V to 1.0V.[3] In accordance with the synthesis results, all designs were undergoing complete place-and-route (P&R) through Cadence Innovus, including floorplanning, power planning, placement, clock tree synthesis (CTS), routing, and final optimization. The primary objective is to investigate how the supply voltage influences physical characteristics such as timing closure, area utilization, routing congestion, and power integrity. For consistency, the same design constraints as a 10 ns clock period, voltage-specific Liberty libraries, and floorplan size were applied to all voltage corners.

This chapter also highlights the difficulties faced during implementation at lower voltages (especially 0.6V and 0.7V), where timing degradation and cell selection limitations imposed significant complication in achieving a clean physical layout.

5.1 Automated Physical Design Flow Using Tcl

To enable stable and efficient physical design implementation across all voltage corners, a generic Tcl script was developed to simplify the overall digital back-end flow in Cadence Innovus. The script supports different voltages automatically using environment variables such as `$VDD`, `$base_dir`, and `$SLOW_LIB`, making it reusable across different voltage points without manual reconfiguration. The flow begins from design initialization and LEF/DEF loading, then goes through floorplanning, power ring and stripe generation, and global net connections for VDD and VSS. It then proceeds with placement, clock tree synthesis (CTS) with custom

buffer and inverter cell lists, and detailed routing with SI-driven and DRC-aware options on. The post-route optimization steps include hold fixing, DRC cleanup using Eco Route, and static timing analysis at each stage.

Additionally, the script automates static power analysis by generating switching activity, power views, and output directories, with detailed reports on leakage, dynamic power, and rail usage. The voltage-awareness and modularity of `pd.tcl` enable the whole flow to be executed repetitively over voltage corners (0.6V–1.0V) so that the physical characteristics can be compared reasonably while maintaining the design as a constant.

5.2 Floorplanning, Power Planning, and Design Strategy

The physical design flow for the iiitb_rv32i core begins with a carefully constructed floorplan that remains consistent across all voltage corners to enable fair comparison. A fixed core area of 320 × 320 units with 15 µm core margins was used. This ensures sufficient space for standard cell placement, routing, and power distribution. The core was aligned with a standard site (CoreSite) to match the standard cell height and avoid placement legality issues.

Power planning is implemented through the generation of core rings and vertical stripes for both VDD and VSS nets. The tcl script configures power rings on Metal7 (horizontal) and Metal8 (vertical) with defined widths, spacings, and offsets. Vertical stripes are inserted in Metal8 with a uniform 100 µm spacing to distribute power across the core, ensuring IR drop is minimized. The power planning setup is tailored to support static rail analysis for later stages of verification.

The overall design strategy emphasizes modularity, reusability, and voltage scalability. Each stage from placement to clock tree synthesis (CTS), routing, optimization, and power analysis is configured to support multiple voltage domains via parameterized scripting. The flow uses synthesis-aware floorplans and timing-driven placement and routing to maximize performance while ensuring design rule compliance and power integrity.

5.3 Clock Tree Synthesis (CTS)

Clock tree synthesis is a critical step in the physical design flow to ensure uniform clock distribution with minimal skew and acceptable transition times across all sequential elements. The iiitb_rv32i core uses a single root clock, and CTS is configured using Cadence Innovus ccopt engine for optimal balancing.

The tcl script explicitly defines a set of voltage-compatible buffer and inverter cells (e.g., BUFX and CLKINVX series) for clock tree construction, ensuring the tool selects cells available in each Liberty library. The `target_max_trans` parameter is tuned to 1.2 ns to prevent excessive slew degradation along clock nets. CTS also accounts for skew control, maximum fanout, and transition constraints to ensure timing closure post-synthesis.

After clock tree build, the script performs post-CTS timing analysis, followed by hold and setup optimization `(optDesign -postCTS)` to ensure all paths meet timing before routing begins. This structured approach provides a stable and repeatable timing base for the remainder of the physical design flow.

Routing Strategy and DRC Cleanup, following clock tree synthesis, detailed routing is performed using the Cadence Innovus Nano Route engine with both timing- and signal-integrity-driven options enabled. The routing is constrained between Metal1 and Metal11, with routing preferences set to avoid shorts, via congestion, and antenna effects.

To improve routing quality and reduce post-route violations, settings are applied such as

1. `routeWithSiDriven`
2. `routeWithTimingDriven`
3. `and earlyGlobalHonorMsvRouteConstraint`

Once routing is completed, the design undergoes post-route optimization, including hold fixing and slack tuning using `optDesign -postRoute -hold`. Special attention is given to dense routing areas and corners where signal congestion or electromigration risks are higher. Any residual DRC violations are resolved using the `ecoRoute -fix_drc` command, which performs localized net rewiring and spacing corrections to clean up violations without disturbing timing.

The routing strategy ensures reliable connectivity while maintaining timing integrity and design rule compliance, providing a robust base for final power and rail analysis across all voltage corners.

The next section summarizes and compares key physical design metrics obtained at five voltage corners (0.6V to 1.0V). The analysis focuses on timing closure, area utilization, leakage power, and power integrity. The purpose is to

evaluate how voltage scaling affects the physical design characteristics and to identify the optimal operating point for the iiitb_rv32i core.

5.3.1 Physical Design at 1 V:

At 1.0V, the RV32I core underwent a complete physical design flow using the Innovus tool. The process included floorplanning, power planning (ring and stripe creation), placement, clock tree synthesis (CTS), routing, optimization, and analysis phases such as timing, power, and IR drop analysis.

The design achieved clean timing closure, both for setup and hold paths, indicating no violations in the post-route stage. The positive slack can be observed from Figure 5.3.1-1.

Design Rule Violations (DRVs)

a) No DRC violations were reported post-routing.
b) Design Density: 43.80%
c) Glitch Violations: 0
d) From Figure 5.3.1-2, Max capacitance, transition, and fan-out constraints were respected. Some of them are too close at fan-out16, but there is no negative worst slack so further not optimized.

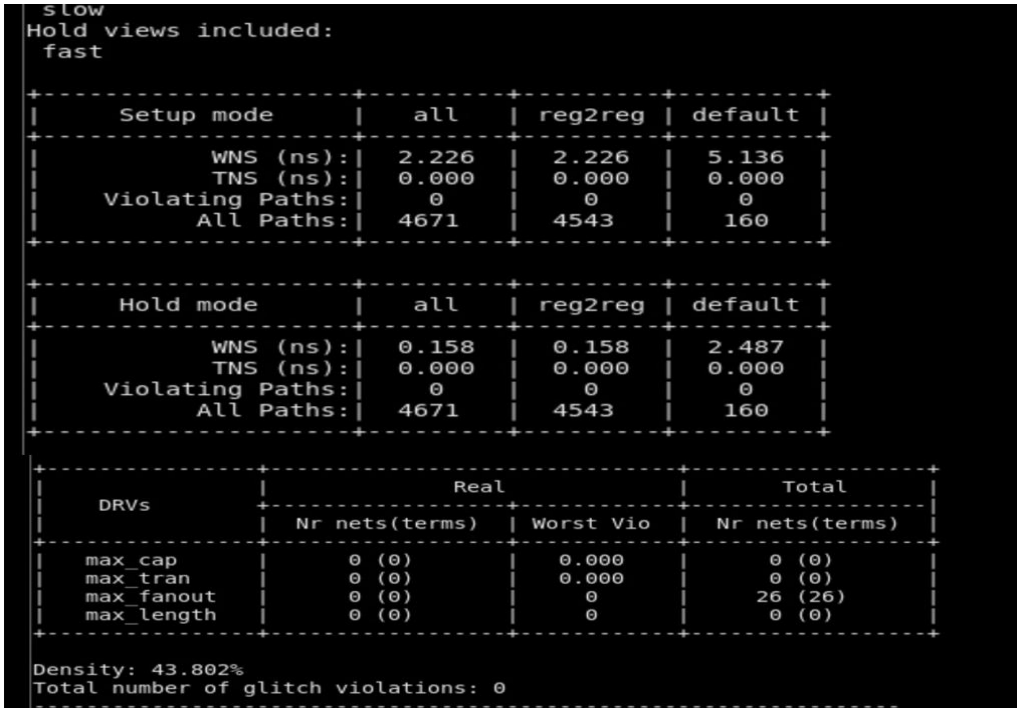


Figure 5.3.1-1: Timing Analysis at 1 V

1. Routing Statistics

   a) Total Wire Length: 212,904 µm

   b) Via Count: 72,135

   c) Multi-cut vias: 94.3%

   d) Single-cut vias: 5.7%

   e) Routing Layers Used: Metal1 through Metal8

   f) Density: 43.8%

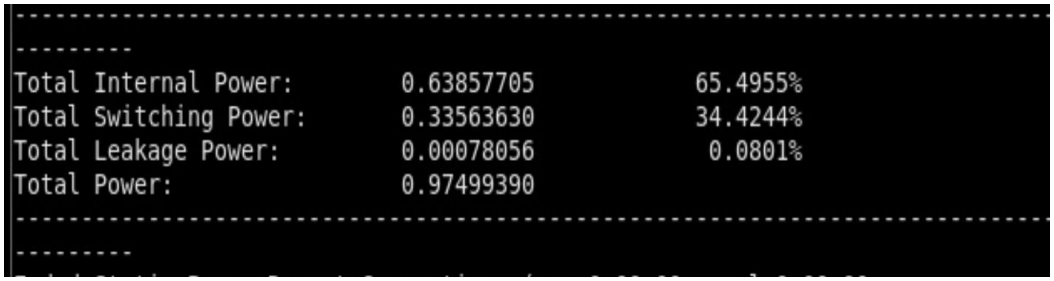


Figure 5.3.1-2: Power Analysis at 1V

g) Detail routing was completed without failures or violations, ensuring optimal signal integrity and manufacturability.

2. The power profile is dominated by internal dynamic power, which is typical at higher voltages due to faster transitions and higher capacitance activity. The design shows balanced internal and switching power components in Figure 5.3.1-3, with minimal leakage. Power efficiency is maintained across the entire core.
3. Rail Analysis at 1.0V- A detailed rail analysis was performed for the 1.0V implementation using static IR drop verification. The objective was to ensure that the power delivery network can maintain voltage levels across the chip without degradation under peak conditions.
4. IR Drop Summary (Static Analysis)
    a) Nominal Voltage (VDD): 0.9V
    b) IR Drop Threshold: 0.81V
    c) Voltage Observed Range: 0.8995 V to 0.9
    d) Maximum IR Drop: ~0.00048 V
    e) Instance Voltage Coverage: 100% instances received full 0.9V

f) Maximum IR Drop Layer: Metal1

g) This confirms that the IR drop across the layout is negligible, and all instances are powered adequately. Metal1 exhibited the highest drop due to its proximity to logic cells.

5. Tap Current and Instance Load Distribution:
    a) Average Tap Current: ~78.5 nA
    b) Maximum Tap Current: 8.4 µA
    c) Total Tap Current: ~0.836 mA
    d) Tap cells are operating well below current limits, indicating good distribution and placement across the design.
6. Resistor and Via Analysis:
    a) Resistor Current:
        i. Average: 2.53 µA Maximum: 0.298 mA
    b) Resistor Voltage Drop:
        i. Average: 1.15 µV Maximum: 0.134 mV
7. Multi-Cut Via Utilization: 94.3%
8. The power grid shows excellent interconnect quality with minimal resistive losses and high via efficiency, critical for handling core currents. The 1.0V (VDD = 0.9V) implementation of the RV32I core demonstrates robust voltage rail integrity, zero IR drop violations, and optimized power grid planning. The power and current distribution across metal layers, taps, and vias validates the sufficiency of the routing, ring, and stripe configurations for reliable operation at nominal voltage.

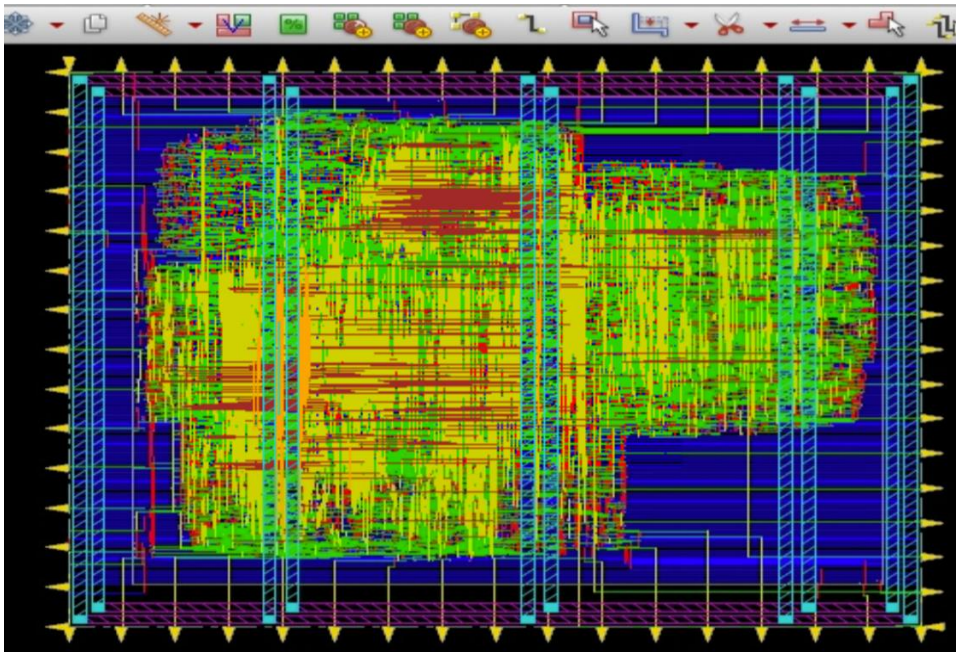
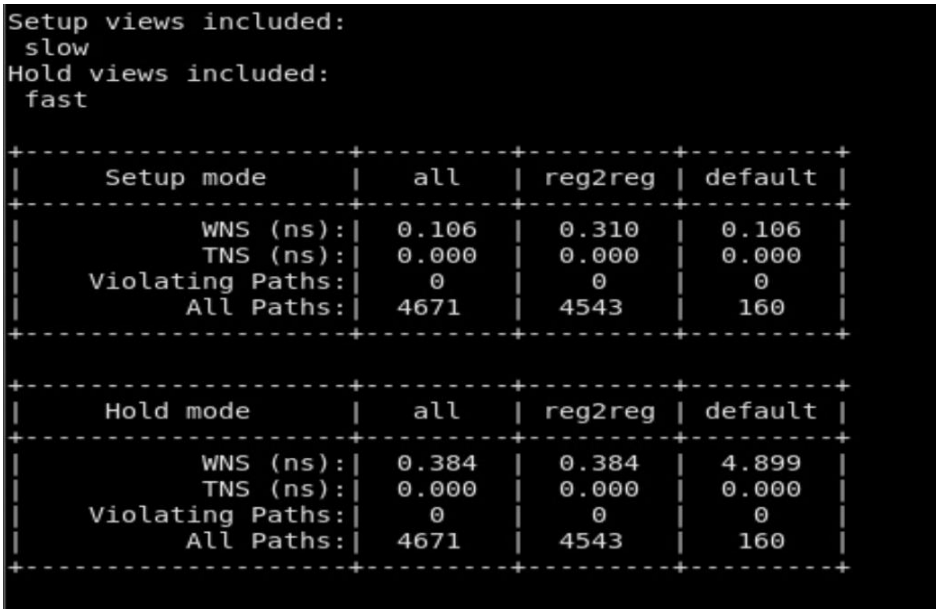


Figure 5.3.1-3: Final Physical Design at 1V

a. Final Observation: The design performed optimally at 1.0V, achieving full timing closure with zero setup/hold violations, clean routing, and minimal IR drop. Power consumption is balanced and well-distributed across switching and internal sources, with negligible leakage. This voltage serves as the nominal operating point with the best overall QoR (Quality of Results) among all analyzed corners.

5.3.2 Physical Design at 0.9 V:

The design achieved clean timing closure, both for setup and hold paths, indicating no violations in the post-route stage.

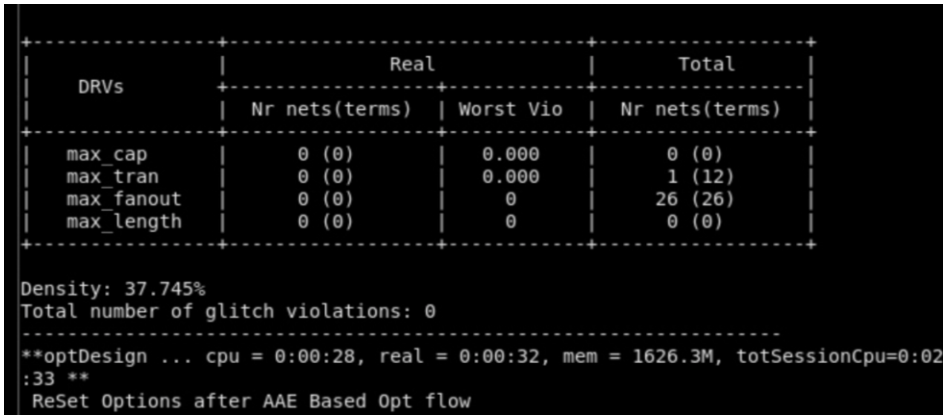


```
Setup views included:
 slow
Hold views included:
 fast

+--------------------+---------+---------+---------+
|     Setup mode     |   all   | reg2reg | default |
+--------------------+---------+---------+---------+
|           WNS (ns):|  0.106  |  0.310  |  0.106  |
|           TNS (ns):|  0.000  |  0.000  |  0.000  |
|    Violating Paths:|    0    |    0    |    0    |
|          All Paths:|  4671   |  4543   |   160   |
+--------------------+---------+---------+---------+

+--------------------+---------+---------+---------+
|     Hold mode      |   all   | reg2reg | default |
+--------------------+---------+---------+---------+
|           WNS (ns):|  0.384  |  0.384  |  4.899  |
|           TNS (ns):|  0.000  |  0.000  |  0.000  |
|    Violating Paths:|    0    |    0    |    0    |
|          All Paths:|  4671   |  4543   |   160   |
+--------------------+---------+---------+---------+
```

Figure 5.3.2-1: Timing Analysis at 0.9 V

At 0.9 V, the RV32I core achieved complete timing closure, with setup WNS of 0.106 ns and hold WNS of 0.384 ns. The lower voltage caused a modest reduction in slack due to increased gate delay, yet no violations were reported. The timing remained robust due to effective buffer insertion and path balancing. This demonstrates the design's stability across moderate voltage scaling, confirming that 0.9 V is a viable operating point without compromising timing integrity.

1. Design Density: 37.75%
2. Glitch Violations: 0
3. Design Rule Violations (DRVs): From Figure 5.3.2-1,
    a) No DRC violations were reported post-routing.



DRVs	Real		Total	
	Nr nets(terms)	Worst Vio	Nr nets(terms)	
max_cap	0 (0)	0.000	0 (0)	
max_tran	0 (0)	0.000	1 (12)	
max_fanout	0 (0)	0	26 (26)	
max_length	0 (0)	0	0 (0)	

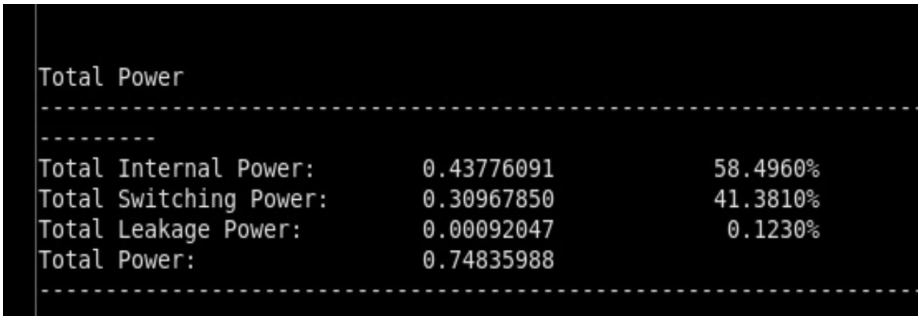
Density: 37.745%
Total number of glitch violations: 0

**optDesign ... cpu = 0:00:28, real = 0:00:32, mem = 1626.3M, totSessionCpu=0:02:33 **
ReSet Options after AAE Based Opt flow

Figure 5.3.2-2: DRVs at 0.9 V

b) Max capacitance, transition, and fan-out constraints were respected. Some of them are too close at fan-out =16 and maximum transition limits but there is no worst slack so further not improved. Max transition/capacitance/fanout Violations: 0 real, 1 maximum transition, 26 fanout (no worst violations).

4. From Figure 5.3.2-3 and 5.3.3-3, Power consumption dropped significantly compared to the 1.0V implementation, especially in dynamic components due to the lower supply voltage. Power savings confirms the theoretical explanation of low voltage low power savings.
5. Figure 5.3.2-3 and 5.3.3-3 show that power consumption decreased significantly compared to the 1.0 V implementation. This reduction is most prominent in the dynamic components, namely internal and switching power, which scale quadratically with supply voltage. The total power dropped due to the f, C relationship, validating the theoretical principles of dynamic power savings through voltage scaling.



```
Total Power
-------------------------------------------------------------------
---------
Total Internal Power:        0.43776091           58.4960%
Total Switching Power:       0.30967850           41.3810%
Total Leakage Power:         0.00092047            0.1230%
Total Power:                 0.74835988
-------------------------------------------------------------------
```

Figure 5.3.2-3: Power Analysis at 0.9 V

6. Moreover, the observed power reduction aligns with expectations from voltage-scaling literature: as voltage decreases, both the signal swing and short-circuit currents reduce, thereby lowering dynamic activity-related power dissipation.
7. The leakage power remains negligible and contributes minimally to total consumption at this node, further emphasizing that dynamic power dominates in this technology. This confirms that operating at 0.9 V offers an effective trade-off between power efficiency and performance. While some timing slack is reduced compared to 1.0 V, it remains within safe margins without requiring major architectural or buffer-level changes. This makes 0.9 V a highly viable corner for energy-efficient embedded designs using the RV32I core.
8. IR Drop Summary (Static Analysis):
    a) Voltage (VDD): 0.85V
    b) IR Drop Threshold: 0.765V
    c) Voltage Observed Range: 0.8496 V to 0.85 V
    d) Maximum IR Drop: ~0.00039 V

e) Violations Detected: None

f) Instance Voltage Coverage: 100% of instances received full VDD (0.85 V)

9. Maximum Drop Layer: Metal1
10. IR drop remains minimum across all metal layers, with Metal1 experiencing the highest drop due to high cell switching activity. All values are within safe margins.
11. Tap Current and Instance Load Distribution:

a) Average Tap Current: ~85 nA

b) Maximum Tap Current: 8.09 µA

c) Total Tap Current: ~0.867 mA

d) Tap cells are well distributed and operate below maximum allowed limits, ensuring reliable grid connectivity.

12. Resistor and Via Analysis:

    1. Resistor Current:

        i. Average: 2.11 µA Maximum: 0.444 mA

    2. Resistor Voltage Drop:

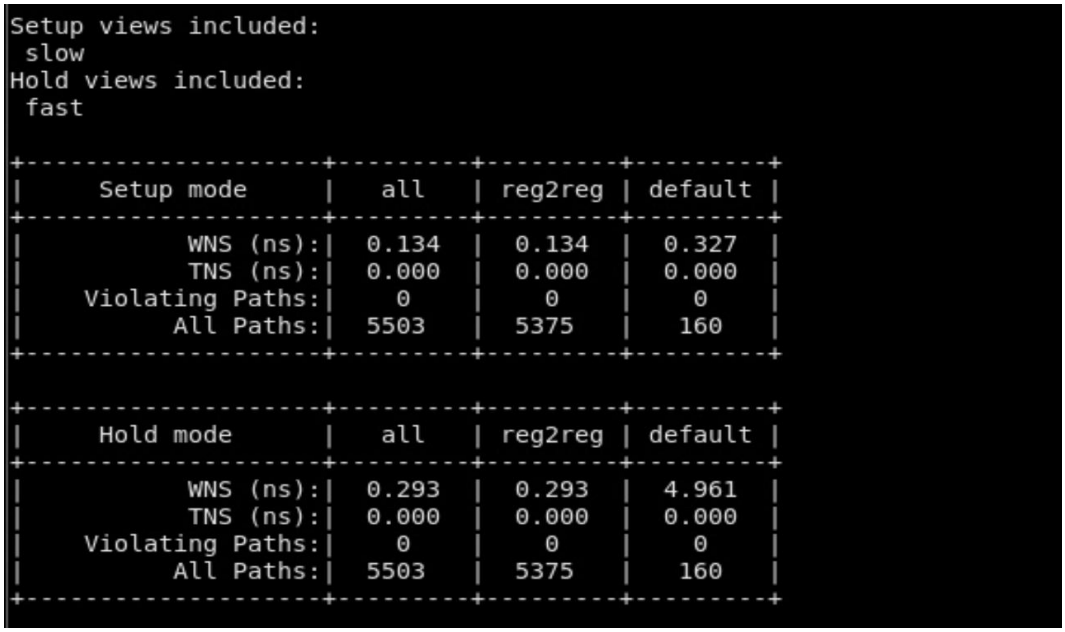
        i. Average: 1.50 µV Maximum: 0.139 mV

13. Multi-Cut Via Utilization: High (consistent with routing strategy)
14. The routing and metallization strategy ensures minimal resistive losses, good current spread, and efficiency via usage throughout the power grid.
15. Final Observation: The 0.9V (VDD = 0.85V) implementation of the RV32I core demonstrates strong power integrity and minimal IR drop. With zero

timing or DRC violations, low power consumption, and excellent voltage coverage, this implementation is optimized for energy-efficient operation under reduced voltage. Metal routing and tap current distributions confirm robust grid design even at low voltage levels.

5.3.3 Physical Design at 0.8 V:

At a 10 ns clock period, the RV32I design achieves excellent timing closure. Setup analysis across all paths reports a WNS of 0.134 ns and TNS of 0 ns, while hold analysis shows a WNS of 0.293 ns. No violating paths were observed in either case. This confirms that the design is robust and timing-clean under relaxed clock conditions, with generous margins supporting stable operation. The timing analysis revealed large positive slack margins, indicating minimal buffering and sizing were required to meet setup constraints. The wide hold slack values suggest that hold buffer insertion was likely used to balance path delays. Overall, the relaxed timing window allowed for simplified CTS and reduced gate effort, leading to a power-efficient and timing-clean implementation.



```
Setup views included:
 slow
Hold views included:
 fast

+--------------------+---------+---------+---------+
|     Setup mode     |   all   | reg2reg | default |
+--------------------+---------+---------+---------+
|           WNS (ns):|  0.134  |  0.134  |  0.327  |
|           TNS (ns):|  0.000  |  0.000  |  0.000  |
|    Violating Paths:|    0    |    0    |    0    |
|          All Paths:|  5503   |  5375   |   160   |
+--------------------+---------+---------+---------+

+--------------------+---------+---------+---------+
|     Hold mode      |   all   | reg2reg | default |
+--------------------+---------+---------+---------+
|           WNS (ns):|  0.293  |  0.293  |  4.961  |
|           TNS (ns):|  0.000  |  0.000  |  0.000  |
|    Violating Paths:|    0    |    0    |    0    |
|          All Paths:|  5503   |  5375   |   160   |
+--------------------+---------+---------+---------+
```

Figure 5.3.3-1: Timing Analysis at 0.8 V

At a clock cycle of 10 ns, the total power consumption of RV32I design was 0.6509 mW. The internal power contributed to 56.50%, 43.34% of switching power, and a very small 0.16% leakage power. Low leakage and same internal-to-switching power ratio reflect an efficient design at low clock frequency, suitable for low-power applications. This result validates the theoretical relationship $P \propto V^2$ governing dynamic power, as both the internal and switching components dropped significantly in response to the reduced supply voltage. The balance between internal and switching power also indicates efficient clock and data path activity without excessive toggling or glitching.

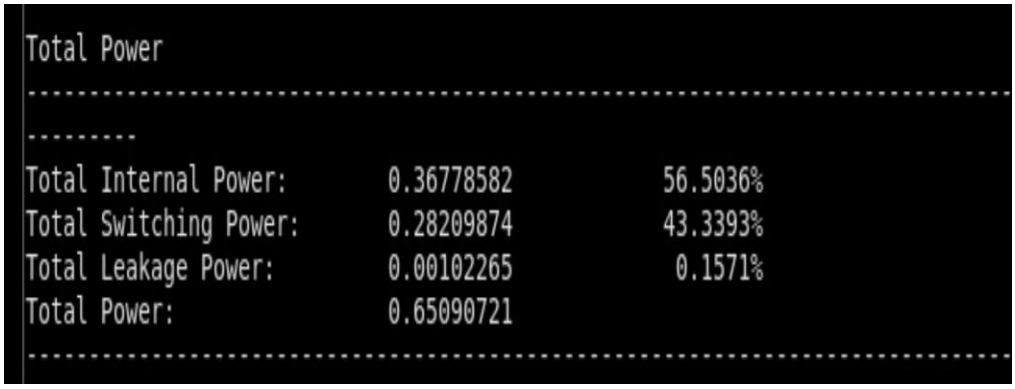


Figure 5.3.3-2: Power Analysis at 0.8 V

1. IR Drop Analysis Summary (Static Analysis):
    a. Voltage (VDD): 0.75 V
    b. IR Drop Threshold: 0.675 V
    c. Voltage Observed Range: 0.74967 V to 0.75 V
    d. Maximum IR Drop: ~0.00033 V
    e. Violations Detected: None
    f. Instance Voltage Coverage: 100% of instances received full VDD (0.75 V)
    g. Layer with Maximum IR Drop: Metal1
    h. The IR drop remains minimal across all metal layers, with Metal1 experiencing the highest drop due to high cell switching activity. All values are within safe margins.
2. Tap Current and Instance Load Distribution
        a) Average Tap Current: ~62.65 nA
        b) Maximum Tap Current: 8.07 µA

c) Total Tap Current: ~0.852 mA

3. Tap Cell Distribution: Well-distributed and operating below maximum allowed limits, ensuring reliable grid connectivity.
4. Resistor and Via Analysis
   i. Resistor Current:

      Average: 1.79 µA Maximum: 0.444 mA

   ii. Resistor Voltage Drop:

      Average: 1.106 µV Maximum: 0.108 mV

   iii. Multi-Cut Via Utilization: High (consistent with routing strategy). The routing and metallization strategy ensures minimal resistive losses, good current spread, and efficiency via usage throughout the power grid.

At 0.75 V, the IR drop analysis showed excellent power rail integrity, with a maximum drop of only ~0.00033 V and no violations. All instances received full VDD, and Metal1 experienced the highest drop due to switching activity, still within safe limits. Tap currents and resistor drops were minimal, and the high multi-cut via utilization ensured efficient power distribution across the grid. Resistor voltage drops remained extremely low, averaging just 1.106 µV with a peak of 0.108 mV, confirming minimal resistive losses. The clean rail profile and full voltage coverage highlight the effectiveness of the physical design and routing strategy at this voltage corner.

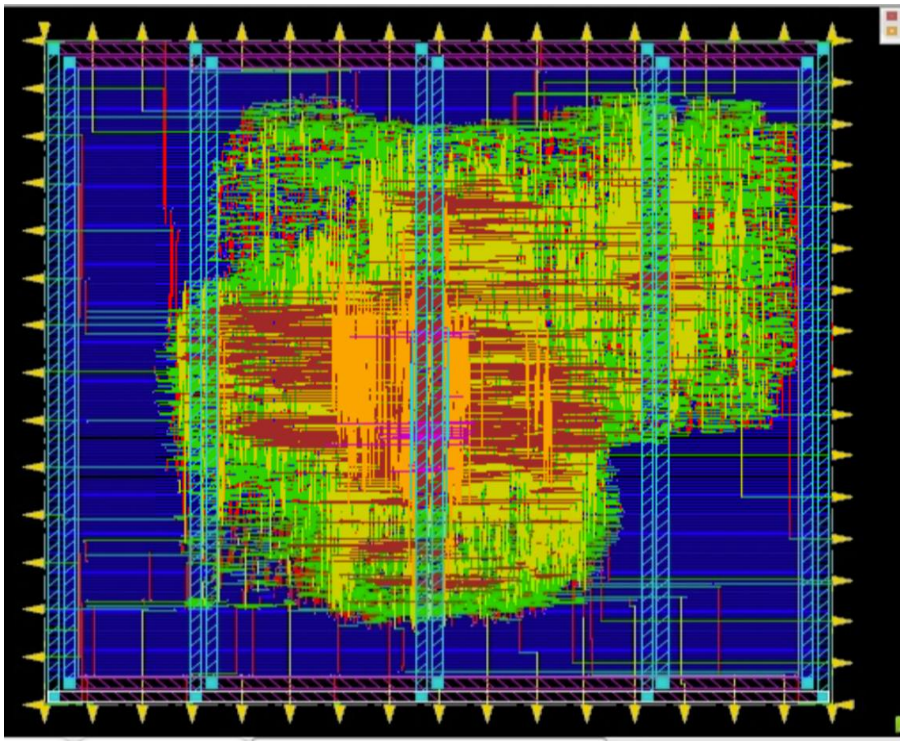


Figure 5.3.3-3: Final Physical Design at 0.8v

Final Observation: The 0.8V (VDD = 0.75V) implementation of the RV32I core demonstrates strong power integrity and minimal IR drop. With zero timing or DRC violations, low power consumption, and excellent voltage coverage, this implementation is optimized for energy-efficient operation under reduced voltage. Metal routing and tap current distributions confirm robust grid design even at low voltage levels.

After the voltage level of 0.8 V, the standard cell libraries were characterized with a slow corner at VDD = 0.75 V and a fast corner at VDD = 0.85 V. Although synthesis was successfully completed at lower voltages such as 0.7 V and 0.6 V, the physical design flow encountered critical issues. Specifically, the 0.7 V and 0.6 V implementations failed during the clock

optimization (`coptDesign`) stage due to missing legal CTS cells and excessive cell delays, which resulted in timing constraint violations, tool and characterization errors. As a result, physical design and implementation could not proceed for these voltage levels.

Therefore, this chapter focuses on the complete physical design flow and analysis for 0.8 V, 0.9 V, and 1.0 V, where all stages including placement, clock tree synthesis, routing, timing closure, and power/IR drop analysis were successfully completed.

5.3.4 Challenges at 0.7 V (Slow VDD 0.65V and Fast VDD=0.75V):

At 0.7V, place-and-route failed due to an unplaced design triggering the error IMPCCOPT-4254 during clock tree synthesis *(ccopt_design)*, halting the flow before CTS. At 0.7V (VDD = 0.65V), the physical design failed during the flow. The tool returned the fatal error that design must be placed before running *'ccopt_check_prerequisites_internal'*, showing that the design was not properly placed or legally populated across standard cell sites before *ccopt_design* was invoked.

This failure is attributed to:

1. Missing or invalid site rows for standard cells in the floorplan at low voltage. Even after adding `CoresiteDouble` in floorplan, they are not snapped to rowsites and faced unlegalized cells.
2. Cell usage issues (due to incomplete Liberty timing arcs at 0.65V).

3. Tool constraints are not satisfied due to degraded timing and invalid cell placements under voltage stress.
4. As a result, clock tree synthesis could not proceed, and the entire physical design flow was stopped before CTS.

This is probably because max trans at this `ccopt` is set to 1.2ns when the design was implemented but after removing that command the physical design flow ran successfully but with lots of errors and warnings about slow .lib file and power was very huge which does not make any valid point with some instances not snapped to Rowsite warnings. At VDD=0.65, Liberate failed to generate and characterize the cells properly at slow corner and set delay values like 100s rise transition and above for some cells.

This is the main reason that the physical design flow was not completed at 0.7 V (Slow VDD =0.65 V and fast VDD =0.75v). The same reason for 0.6 V failed physical design because the delays are 100s which are values or incorrect values given when characterized by liberate. During the implementation at 0.7V, significant issues were encountered that prevented the physical design flow from progressing beyond the initial stages.

Upon loading the characterized Liberty files for 0.7V, it was observed that many cells exhibited invalid rise transition values exceeding 100 seconds. Such extreme values are unrealistic in digital designs, where typical transitions are expected to occur within a few picoseconds to nanoseconds.

These errors suggested failure during the characterization process, likely caused by convergence problems or insufficient cell drive strength at low voltage,

rather than true slow cell behavior. Although the design successfully completed floor planning and initial power planning, the presence of these issues made the 0.7V implementation unsuitable for placement, clock tree synthesis, and timing closure. As a result, no further attempts were made to proceed with 0.7V, and the case was documented to highlight the challenges and limitations of near-threshold voltage physical design.

5.4 Comparison of Physical Design at 0.8 V, 0.9 V, 1.0 V

5.4.1 Timing Performance across three voltages

The timing performance of the RV32I RISC-V core across three voltage levels 1.0 V, 0.9 V, and 0.8 V is as expected from the observation at Table 5.4.1, a decrease in supply voltage leads to a degradation in timing margin, most notably observed in the Setup Worst Negative Slack (WNS), which drops from positive 2.226 ns at 1.0 V to just positive 0.134 ns at 0.8 V. Despite this, Total Negative Slack (TNS) stays zero across all voltages, indicating that the design meets setup and hold constraints across all paths. However, the number of timing paths slightly increases at 0.8 V, which may be due to changes in clock tree synthesis or cell selection during implementation. Hold WNS stays positive at all voltages, confirming no hold violations even at lower voltages. At 0.8 V timing was aggressively optimized and the timing degraded as the voltage goes down to 0.8 V.

Parameter	1.0V	0.9 V	0.8 V
Setup WNS	2.226 ns	1.106 ns	0.134 ns
Setup TNS	0.000 ns	0.000 ns	0.000 ns
Hold WNS	0.158 ns	0.384 ns	0.293 ns
Hold TNS	0.000 ns	0.000 ns	0.000 ns
Setup Paths	4671	4671	5503
Hold Paths	4671	4671	5503

Table 5.4.1: Timing Performance across three voltages

Overall, the design keeps functional correctness across all three voltages, though with diminished timing margins at reduced VDD, highlighting the challenges of voltage scaling in timing closure.

5.4.2 Power Comparison of Different Voltages:

1. Total power decreases significantly from 1.0V to 0.8V (from 0.6386 mW to 0.4377 mW).
2. A key observation from Table 5.4.2 is that at 0.8V, the reduction in supply voltage decreases internal power.

Voltage	1.0V (VDD = 0.9V)	0.9V (VDD = 0.85V)	0.8V (VDD = 0.75V)
Internal Power	0.6386 mW	0.4377 mW	0.3677 mW
Switching Power	0.3356 mW	0.3096 mW	0.2820 mW
Leakage Power	0.0007 mW	0.0009 mW	0.0010 mW
Total Power	0.9750 mW	0.7483 mW	0.6509 mW

Table 5.4.2: Power Analysis at Different Voltages with Slow corner

3. Switching Power: Drops steadily across all voltages (from 0.3356 mW at 1.0V to 0.2820 mW at 0.8V), reflecting the expected V^2 scaling.
4. Leakage Power: Remains minimal but slightly increases at lower voltages due to longer delays of cells.

5.4.3 Rail Analysis Summary and Observations

1. As the supply voltage is scaled from 1.0V down to 0.8V, the IR drop analysis reveals stable and reliable power delivery across all voltage levels, with no violations and full coverage of all standard cell instances.
2. At 1.0V (actual VDD = 0.9V), the maximum IR drop observed is approximately 0.00045V, with all instance voltages staying above the defined threshold of 0.81V.

3. At 0.9V (actual VDD = 0.85V), the drop reduces to about 0.00041V, with a threshold of 0.765V.
4. At 0.8V (actual VDD = 0.75V), IR drop is the lowest, ~0.00033V, and the threshold is set at 0.675V.
    a. These thresholds are conservatively chosen as 90% of the actual VDD used in each case. This margin ensures:
5. Sufficient headroom for IR drops due to dynamic switching activity.
6. Safe operation of standard cells that may not meet timing or functionality at degraded voltages.
    a. Despite voltage scaling: The power grid remains robust with no signs of rail integrity issues.
7. Metal1 continues to experience the highest drop due to high cell switching density.Power pads and tap insertion strategies provide consistent delivery, even at near-threshold voltages.
8. Overall, the rail analysis confirms that energy-efficient operation is possible at reduced supply voltages (down to 0.75V) without compromising the integrity of the power distribution network.

5.5 Performance analysis using Clock Speed 100ns(10MHz)

5.5.1 Synthesis Results

To explore ultra-low power operation, the RISC-V core was re-synthesized with a relaxed clock period of 100 ns, enabling operation at 10 MHz suitable for sub-threshold or energy-constrained

Metric	0.6V(VDD=0.55V)	0.7 V(VDD=0.65V)	0.8 V(VDD=0.75V)	0.9 V(VDD=0.85V)	1 V(VDD=0.9V)
Slack (ps)	+119.2	+10,418	+44,159	+45,363	+45,591
Total Negative Slack (ps)	0	0	0	0	0
Failing Paths	0	0	0	0	0
Leaf Cell Count	11440	13,416	8,640	8,646	9,377
Total Cell Area (μm²)	76,708	69,211	37,739	37,495	36,512
Total Area incl. Nets (μm²)	89643.05	84,171.76	48,823.79	48,246.43	47,800.42
Leakage Power (nW)	1,758.42	1,816.91	805.45	949.79	792.49
Synthesis Runtime (min)	35	16.8	6.7	6.4	6.5

Table 5.5.1 Synthesis results of all voltages at 100ns

applications. Comparing 10 ns and 100 ns clock period synthesis runs reveals the impact of relaxed timing constraints on the RISC-V core at various voltage levels. For accurate and correct static timing analysis at synthesis and physical design phases, a large set of design constraints was created in Synopsys Design Constraints (SDC) format. A clock cycle of 100 ns along with corresponding

setup and hold slack variances of 2.0 ns and 1.0 ns, respectively, was set to account for variation of clock jitter and skew. Clock transition parameters were specified as 0.1 ns to 0.8 ns to guide buffer insertion and transition optimization. 50 ns input and output delays were set to model realistic interface timing against local logic.

Additional global constraints e.g., a 0.8 ns maximum transition limit and 16 maximum fanout were given for the overall design to enable signal integrity and sustainable loading conditions to enable better QoR during synthesis and place-and-route processes.

The design failed timing at 0.6 V at 10 ns with large negative slack of –19,441 ps and 4,460 failing paths but passed at 100 ns with a positive slack of +10,418 ps, confirming timing closure. In all voltages, the switch to 100 ns caused drastic reductions in cell total area and power. For instance, at 0.6 V, area decreased from 101,730 μm² to 69,211 μm², and leakage power decreased from 2,372.74 nW to 1,816.91 nW. The same pattern repeats at 0.8 V and 0.9 V where total area fell by nearly 2,000 μm² and leakage power by over 100 nW. This readily demonstrates that longer clock period allows the synthesis tool to utilize slower, smaller, and less leaky cells with significant gains in power and area efficiency without sacrificing functional correctness.

The outcomes emphasize how performance requirements directly affect design choices, cell utilization, and power behavior, especially at near-threshold voltage.

5.5.2 Timing and Power Analysis

The RV32I processor core demonstrated consistent timing closure across all three target voltages 0.8 V, 0.9 V, and 1.0 V under a 100 ns clock period, with zero setup or hold violations observed in any case. Among these, the 0.9 V implementation exhibited the best timing performance with a Setup Worst Negative Slack (WNS) of 42.269 ns and a Hold WNS of 0.362 ns. These figures indicate a healthy timing margin and signal robustness, affirming 0.9 V as a reliable operating point for both functional correctness and design closure.

At 0.8 V, the processor still met timing with a Setup WNS of 35.004 ns, although the design faced increased physical complexity. Notably, the density was significantly higher (79.53%) compared to 57.80% at 0.9 V, indicating a more congested layout and here the die area was 350 x 350 (µm). Moreover, twenty-six nets exceeded the maximum allowed transition threshold, necessitating extra optimization effort in the form of buffering or re-placement. This implies that while 0.8 V is viable for ultra-low-power operation, it imposes stricter demands on placement, routing, and transition control, possibly impacting design turnaround time.

In contrast, the 1.0 V implementation, although slightly better in setup WNS (42.511 ns), showed minimal design rule violations, with only one net violating transition constraints and a total density of 59.65%. However, the power consumption at this voltage is inherently higher, making it less suitable for energy-constrained applications, despite offering clean timing and routability.

All three voltage corners showed zero total negative slack (TNS) and no timing violations across both setup and hold domains. Additionally, the maximum fanout constraint was violated on a similar scale across all voltages (26–27 nets) but these are not with worst violations they are just too close to the limit given in constraints, and there were no violations related to maximum capacitance or wire length. The total timing paths analyzed dropped slightly from 5503 at 0.8 V to 4671 at higher voltages due to differences in cell delays and optimization strategies.

In conclusion, while the 1.0 V implementation offers a robust and DRC-clean design environment, it does so at the cost of higher dynamic and internal power. The 0.8 V case, although power-efficient, is physically constrained. The 0.9 V implementation thus represents the optimal trade-off point, delivering strong timing performance, fewer transition violations, moderate density, and improved power efficiency making it the most balanced choice for low-power yet high-performance operation.

Parameter	0.8 V	0.9 V	1.0 V
Setup WNS (ns)	35.004	42.269	42.511
Hold WNS (ns)	0.047	0.362	0.340
Setup TNS (ns)	0.000	0.000	0.000
Hold TNS (ns)	0.000	0.000	0.000
Setup Violating Paths	0	0	0
Hold Violating Paths	0	0	0
Max Transition Violations	26 nets	11 nets	1 net
Max Fanout Violations	27 nets	26 nets	26 nets
Max Cap / Max Length	0	0	0
Total Paths Analyzed	5503	4671	4671
Density	79.53%	57.80%	59.65%

Table 5.5.2-1 Timing at all voltages (100ns)

Metric	1.0 V (slow_vdd1v0)	0.9 V (low_vdd0v9)	0.8V (slow_vdd0v8)
Internal Power (mW)	0.0792	0.0550	0.0479
Switching Power (mW)	0.0544	0.0461	0.0502
Leakage Power (mW)	0.0012	0.0013	0.0016
Total Power (mW)	0.1349	0.1025	0.0997

Table 5.5.2-2 Power Analysis at 100ns Clock Period

At 100 ns clock period, the total power consumption is significantly lower at all voltages than at 10 ns, as expected due to reduced switching activity and alleviated timing specifications. When the clock period is doubled from 10 ns to 100 ns, a significant reduction in total power consumption is observed in all the operating voltages, demonstrating the inversely proportional relationship between dynamic power and frequency. At 1.0 V, total power reduces from 0.9750 mW to 0.1349 mW, while at 0.9 V and 0.8 V it reduces from 0.7483 mW to 0.1025 mW and from 0.6509 mW to 0.0997 mW respectively. At 100 ns and 10ns where internal power at 0.8 V (0.0479 mW) is the lowest, as should be expected from voltage scaling. Additionally, leakage power, while relatively small, also has a gradual rising trend at lower voltages, induced by increased cell count or longer transitions. At 0.7 V timing closed and received inaccurate power due to 100sec delay cells. So, this voltage level

is not considered for further analysis. Overall, these findings reaffirm that clock speed reduction is an effective method of power reduction and lower voltages like for 0.65 V they run successfully even though it has huge delays with then automated flow. The power analysis at 0.7 V is completed even though it has some glitch violations and huge max transition slack like 12000ns above with few glitch violations. Many warnings like instances not snapped to row site appeared. This can be cleared with addition of CoresiteDouble to floorplan but even after adding that this appeared again. Due to many isues this might not be exact power it uses but even with those errors cleared there might be a slight increase like 0.2mW but it might not be greater than that but still these are assumptions without proper cells and maximum transition with that huge slack it might not be a better power analysis.

CHAPTER 6: RESULTS AND CONCLUSIONS

6.1 Results

During low-voltage synthesis, the RV32I RISC-V core was synthesized for five voltage levels: 0.6 V, 0.7 V, 0.8 V, 0.9 V, and 1.0 V, using voltage-specific Liberty libraries characterized under near-threshold and nominal operating conditions. Synthesis was successfully implemented at 0.6 V and 0.7 V using libraries characterized at those voltages. However, the resulting netlists exhibited large cell delays and reduced drive strengths, which negatively impacted physical design feasibility. At 0.8 V and higher, synthesis produced netlists with better timing behavior and fewer constraint violations, enabling full place-and-route and post-route analysis.

6.2 Key Observations:

1. Cell delays increased significantly at lower voltages, especially at 0.6 V and 0.7 V, resulting in reduced timing slack and limited optimization headroom.
2. Area and gate count remained mostly stable across all voltages, as the logic structure was preserved, but timing QoR (quality of results) varied sharply.
3. Although synthesis at 0.6 V and 0.7 V completed without functional errors, the netlists were not physically realizable due to incorrect or excessive delay values in the characterized Liberty libraries.
4. During physical design initialization, several standard cells were rejected by the tool at 0.6 V and 0.7 V because of invalid timing arcs and large transition values, which were not flagged during synthesis.

At 0.8 V, the design achieved clean timing closure and completed full place-and-route, as well as rail analysis, without issues. At 0.9 V, the implementation offered the best energy efficiency, balancing dynamic power and timing performance with clean signoff. The 1.0 V implementation exhibited the maximum timing margin and routing stability but consumed the highest total power. Power significantly drops as voltages go down.

In addition, a 100 ns clock period simulation at 0.8 V confirmed the design's robustness under relaxed timing constraints, with no setup or hold violations observed. Importantly, power consumption dropped significantly at 100 ns compared to 10 ns, validating the theoretical relationship $P \alpha f$, were power scales linearly with frequency. This highlights the suitability of the RV32I core for ultra-low-power applications operating at reduced performance levels, such as IoT or always-on systems, where energy savings are prioritized over execution speed. At 0.7 V , physical design closed timing but because of 100 second delays the generated physical design flow might have some inaccuracies. So, this voltage level is not discussed for analyzing power.

This detailed analysis reinforces the importance of accurate Liberty characterization, multi-voltage-aware design, and frequency scaling, as synthesis alone may not reveal critical timing and physical feasibility issues that surface during full implementation.

6.3 Conclusion:

Voltage scaling from 1.0 V to sub-nominal levels of 0.8 V and 0.7 V provides better dynamic power saving opportunities. In this effort, it is shown that voltage scaling, however, poses real physical design issues at or below 0.7 V. At 0.8 V (slow corner VDD = 0.75 V, fast corner VDD = 0.85 V), full physical design was achieved with clean timing and rail analysis success. Through physical design analysis at different voltage levels, it was observed that for 1.0 V, the design had optimal timing margins and routing stability but with the highest total power consumption. Timing degradation and physical design feasibility issues grew with voltage scaling as the voltage went down. At 0.7 V, the flow experienced critical failures in floorplanning and place-and-route. Although synthesis completed flawlessly, physical design sanity checks unearthed underlying causes such as incorrect rise transition values greater than 100 ns in the characterized Liberty files that prevented the design from progressing beyond early power planning.

This analysis in the real world illustrates that at lower voltage corners such as 0.65 V and below, correct characterization of standard cells is essentially not practicable. Much work has to be done to switch each cell correctly, maintain the right drive strength, and meet timing under near-threshold. Also, this paper reconfirms that although synthesis may not catch characterization errors, they often occur during physical design via constraint imposition and sanity checks. Hence, careful characterization, verification, and validation of Liberty models are essential enablers of low-voltage physical design success.

These observations point to a fundamental trade-off in low-voltage digital design: while dynamic power grows as the square of voltage, internal effort and timing power grow non-linearly due to synthesis and place-and-route constraints. Thus, a properly selected voltage level, e.g., 1.0 V achieves the best compromise between energy efficiency and design practicability with standard libraries.

The paper shows a complete synthesis-to-signoff process of a RISC-V RV32I processor core at sub-threshold and near-threshold voltages, exposing the challenge and opportunity of ultra-low-power design in the 45 nm process. The study reaffirms that voltages like 0.8 V and above offer completely functional and efficient physical designs. However, voltages below 0.7 V (e.g., VDD = 0.65 V at the slow corner) result in catastrophic timing degradation and physical design failure due to insufficient or incomplete library characterization. Interestingly, the synthesis tool attempts to counter cell delay degradation by significantly increasing cell area and count and thereby canceling out the expected power savings. The 0.7 V physical design flow breakdown also determines the main limitation of existing cell libraries.

In addition, a low-frequency test case with a 100 ns clock period at 0.8 V attained full timing closure with substantial slack and much lower power consumption than 10 ns. This supports the theoretical power–frequency relationship with dynamic power acting linearly with clock rate, thereby making such implementations best suited for ultra-low-power embedded systems. This finding is a pointer that frequency slowdown operation can effectively compensate for voltage-driven performance degradation at no cost in energy efficiency.

This observation opens new directions of future study, namely, examining if recharacterized nominal cells at 0.7 V can provide an accessible operating point that maximizes energy savings with design functionality. This line of inquiry highlights the significance of voltage-scalable libraries to accommodate near-threshold operation for current low-power digital devices.

REFERENCES

[1] J. Ramirez-Angulo, R. G. Carvajal and A. Lopez-Martin, "Techniques for the Design of Low Voltage Power Efficient Analog and Mixed Signal Circuits," 2009 22nd International Conference on VLSI Design, New Delhi, India, 2009, pp. 26-27, doi: 10.1109/VLSI.Design.2009.112.

[2] Yu Shiang, Lin & Hanson, Scott & Albano, Fabio & Tokunaga, Carlos & Haque, Razi-ul & Wise, Kensall & Sastry, Ann & Blaauw, David & Sylvester, Dennis. (2008). Low-voltage circuit design for widespread sensing applications. Proceedings - IEEE International Symposium on Circuits and Systems. 2558-2561. 10.1109/ISCAS.2008.4541978.

[3] Fauzan, Muhammad & Yanuarsyah, Raihan & Hibatullah, Muhammad & Arisaputra, Radithya & Syafalni, Infall & Sutisna, Nana & Adiono, Trio & Ikeda, Makoto. (2024). Physical Design of RISC-V based System-on-Chip using OpenLane. 529-533. 10.1109/APCCAS62602.2024.10808656.

[4] F. Klemme, Y. Chauhan, J. Henkel and H. Amrouch, "Cell Library Characterization using Machine Learning for Design Technology Co-Optimization," 2020 IEEE/ACM International Conference On Computer Aided Design (ICCAD), San Diego, CA, USA, 2020, pp. 1-9.

[5] J. Abella et al., "An Academic RISC-V Silicon Implementation Based on Open-Source Components," 2020 XXXV Conference on Design of Circuits and Integrated Systems (DCIS), Segovia, Spain, 2020, pp. 1-6, doi: 10.1109/DCIS51330.2020.9268664.

APPENDICES A: SCRIPTS

A.1 Characterization Script

```
set SRC_DIR              [pwd]
set RUN_DIR              [pwd]
set LIB                  example
set PROCESS              tt
set VDD                  1.0
set TEMP                 70
set SETTINGS_FILE        ${SRC_DIR}/tcl/settings.tcl
set TEMPLATE_FILE        ${SRC_DIR}/template/template.tcl
set CELLS_FILE           ${RUN_DIR}/cells.tcl
set NETLIST_DIR          ${SRC_DIR}/netlist
set USERDATA             ${SRC_DIR}/userdata/userdata.lib
puts "INFO: Process command line input:"
foreach arg $argv {
    if { [string match *=* $arg] } {
        lassign [split $arg =] a b
        set $a $b
        puts "INFO:   Setting $a = $b"
    }
}
set PVT                  ${PROCESS}_${VDD}_${TEMP}
set LIBNAME              ${LIB}_${PVT}
set                                      MODEL_INCLUDE_FILE
${SRC_DIR}/models/spectre/include_${PROCESS}

set THREAD    0
set CLIENTS   0
puts "INFO:"
puts "    SRC_DIR              = ${SRC_DIR}"
puts "    RUN_DIR              = ${RUN_DIR}"
puts "    LIBNAME              = ${LIBNAME}"
puts "    PVT                  = ${PVT}"
puts "    SETTINGS_FILE        = ${SETTINGS_FILE}"
puts "    TEMPLATE_FILE        = ${TEMPLATE_FILE}"
puts "    MODEL_INCLUDE_FILE   = ${MODEL_INCLUDE_FILE}"
puts "    NETLIST_DIR          = ${NETLIST_DIR}"
puts "    USERDATA             = ${USERDATA}"
puts "    CELLS_FILE           = ${CELLS_FILE}"
puts "    THREAD               = ${THREAD}"
puts "    CLIENTS              = ${CLIENTS}"
puts ""
puts "INFO: Set Operating Condition"
set_operating_condition  -name  ${PVT}  -voltage  ${VDD}  -temp
${TEMP}
```

```
puts "INFO: Read settings file ${SETTINGS_FILE}"
source ${SETTINGS_FILE}

puts "INFO: Read template file ${TEMPLATE_FILE}"
source ${TEMPLATE_FILE}
if {[file exists ${CELLS_FILE}]} {
    puts "INFO: Read cell list file"
    source ${CELLS_FILE}
} else {
    puts "WARNING: Specified CELLS_FILE (${CELLS_FILE}) does
not exist."
}
puts "INFO: Define device models"
set_var extsim_model_include ${MODEL_INCLUDE_FILE}
define_leafcell -type nmos -pin_position {0 1 2 3} { g45n1lvt
g45n1svt g45n1hvt g45n2svt g45n1nvt g45n2nvt }
define_leafcell -type pmos -pin_position {0 1 2 3} { g45p1lvt
g45p1svt g45p1hvt g45p2svt }
define_leafcell -type diode -pin_position {0 1} { g45nd1svt
g45pd1svt }
puts "INFO: Read cell netlist"
set packet_cells [packet_slave_cells]
if {[llength $packet_cells] > 0} {
    set cells $packet_cells
}
set                                                    spicefiles
{/nfs/users/sjeedima/linux/Downloads/run_char/netlist/gsclib04
5.cdl}
read_spice -format spectre "$MODEL_INCLUDE_FILE $spicefiles"

if { $CLIENTS > 0 } {
    set_var packet_mode      arc
    set_var rsh_cmd          $RSH_CMD
    set_var packet_clients   $CLIENTS
    if { ($RSH_CMD eq "local") } {
        set_var packet_clients 1
    }
    if { $THREAD == 0 } {
        puts "WARNING: THREAD=0 in DRM mode may overload
machines. Consider THREAD=1."
    }
}
puts "INFO: Run Characterization"
char_library -extsim spectre -cells $cells -thread $THREAD
puts "INFO: Write ldb"
file mkdir ${RUN_DIR}/ldb
write_ldb -overwrite ${RUN_DIR}/ldb/${LIBNAME}.ldb

puts "INFO: Write Liberty"
file mkdir ${RUN_DIR}/lib
```

```
write_library -driver_waveform -unique_pin_data -bus_syntax
{[]}    -user_data    ${USERDATA}    -overwrite    -filename
${RUN_DIR}/lib/${LIBNAME}.lib ${LIBNAME}
```

A.2 Settings File Script

```
### Tool version parsing
lassign [split [ALAPI_version] .] x1 x2 version(minor)
version(sub)
set version(major) "${x1}.${x2}"

### Spectre simulator options
set_var extsim_cmd_option      "+aps +spice -mt +liberate
+rcopt=2 +maxns=20u"
set_var   extsim_deck_header                  "simulator
lang=spectre\nOpt1   options   reltol=1e-4   \nsimulator
lang=spice"
set_var extsim_option            "redefinedparams=ignore
hier_ambiguity=lower limit=delta "
set_var  extsim_leakage_option    "redefinedparams=ignore
hier_ambiguity=lower limit=delta "

### SKI options
set_var ski_enable               1
set_var ski_clean_mode           1
set_var ski_compatibility_mode   1
set_var power_tend_match_tran    1

### General Liberate options
set_var parse_auto_define_leafcell   0
set_var tmpdir /dev/shm
set_var extsim_deck_dir [file normalize "decks"]
set_var set_var_failure_action   error
set_var predriver_waveform       2
set_var min_capacitance_for_outputs 1
set_var force_condition             4
set_var constraint_info            2
set_var nochange_mode               1

### Version-specific behavior
if {($version(major)>=17.1) && ($version(minor)>=2)} {
    set_var constraint_vector_mode 4
}

### Leakage and power control
set_var conditional_mpw                   0
```

```
set_var max_leakage_vector                  [expr 2**10]
set_var leakage_float_internal_supply      0
set_var reset_negative_leakage_power       1
set_var voltage_map                         1
set_var pin_based_power                     0
set_var power_combinational_include_output  0
set_var force_default_group                 1
set_default_group -criteria                 {power avg}
set_var power_subtract_leakage              4
set_var subtract_hidden_power               2
set_var subtract_hidden_power_use_default   3
set_var power_multi_output_binning_mode     1
set_var power_minimize_switching            1
set_var max_hidden_vector                   [expr 2**10]

### CCSN / CCSP options
set_var ccsn_include_passgate_attr 1
set_var ccsn_model_related_node_attr 1
set_var ccsp_min_pts              15
set_var ccsp_rel_tol              0.01
set_var ccsp_table_reduction      0
set_var ccsp_tail_tol             0.02
set_var ccsp_related_pin_mode     2

### EM file setup (optional)
if    {    [info    exists    CHAR_EM_TECH_FILE]    &&
($CHAR_EM_TECH_FILE ne "") } {
    set_var       em_tech_file       [file       normalize
$CHAR_EM_TECH_FILE]
}

### Output customization
set_var write_library_is_unbuffered           1
set_var cell_use_both_ff_latch_groups         2
set_var user_data_override { power_down_function pg_pin
input_signal_level output_signal_level }
set_var sdf_cond_style                        1
set_var parenthesize_not                      0
set_var driver_type_model_pad_check           1
set_var ccsn_print_is_needed_if_false_attr_value 1
set_default_group -criteria {constraint off}
set_var write_library_allow_switching_and_hidden_power 1

### LIBERATE_LV mode
if { $::LIBERATE_program == "LIBERATE_LV" } {
```

```
    set validate_cells_per_bundle 10000
}

### VARIETY mode
if { $::LIBERATE_program == "VARIETY" } {
    set_var variation_mean_nominal_mode       4
    set_var lvf_constraint_early_late_mode    1
    set_var extsim_monte_option  "sampling=lds"
}
```

A3. Constraints at 100 ns

```
# Clock Definition
create_clock -name clock -period 100 [get_ports clk]

# Clock Uncertainty
set_clock_uncertainty -setup 2.0 [get_clocks clock]
set_clock_uncertainty -hold  1.0 [get_clocks clock]

# Clock Properties
set_clock_transition -max 0.8 [get_clocks clock]
set_clock_transition -min 0.1 [get_clocks clock]
set_clock_latency 1.0 [get_clocks clock]

# Input Constraints
set_input_delay          -clock  [get_clocks  clock]  50.0
[all_inputs]
set_input_transition -max 0.2 [all_inputs]

# Output Constraints
set_output_delay   -clock   [get_clocks   clock]   50.0
[all_outputs]
set_load         0.05 [all_outputs]

# Design-Wide Constraints
set_max_transition 0.8 [current_design]
set_max_fanout     16  [current_design]
```