Indian Journal of VLSI Design (IJVLSID)
ISSN: 2582-8843 (Online), Volume-6 Issue-1, March 2026

Implementation-Aware UPF Methodology for Managing Power Domain Crossings in Multi-Voltage SoCs

Aayush Gade, Bhavesh Soni, Dipesh Panchal

Abstract: ***Aggressive power targets in mobile, IoT, and automotive SoCs have driven extensive use of multi-voltage and power-gated domains, making power domain crossings a major source of overhead and bugs. Power in- tent is usually specified in UPF at a high level, and generic rules for isolation, level shifting, and retention are pushed into implementation, which can lead to an excessive number of special cells, timing loss, routing congestion, and difficult low-power sign-off. This work proposes an implementation-aware power domain crossing (PDC) methodology tightly integrated with a Cadence Innovus-based flow. First, a clustering-based partitioning strategy groups strongly communicating blocks into common domains to minimise crossings and the number of special cells. Second, an optimised UPF boundary strategy tailors' isolation, level-shifting, and retention policies to specific domain relationships and power modes, avoiding redundant instrumentation while preserving correctness. Third, a complete RTL → UPF → physical design flow is presented, with quantitative evaluation of area overhead, timing impact, power savings, and low-power verification violations against a conventional UPF baseline. Results for representative multi-domain SoC subsystems show that the proposed approach significantly reduces domain crossings and special cells while maintaining sign-off quality, turning PDC from a late-implementation side effect into a controllable design parameter.***

Keywords: ***Low-Power VLSI Design, Multi-Voltage SoC, Power Domain Crossings, Unified Power Format (UPF), Power Intent specification, Isolation and level Shifters, Retention Registers and Power Gating, Physical Design and Implementation (Cadence Innovus)***

Nomenclature:
WNS: Worst-Negative-Slack
TNS: Total Negative Slack
UPF: Unified Power Format
ECOs: Engineering-Change Orders
PDC: Power-Domain Crossing
QoR: Quality of Results
ELS: Enabled Level Shifter
L2H: Low-to-High

Manuscript received on 27 January 2026 | First Revised Manuscript received on 02 March 2026 | Second Revised Manuscript received on 10 March 2026 | Manuscript Accepted on 15 March 2026 | Manuscript published on 30 March 2026.
*Correspondence Author(s)
Aayush Gade*, Department of Electronics & Communication Engineering, Ganpat University, Ahmedabad (Gujarat), India. Email ID: 24014991014@gnu.ac.in, ORCID ID: 0009-0005-3037-3494
Prof. Bhavesh Soni, Assistant Professor, Department of Electronics & Communication Engineering, Ganpat University, Ahmedabad (Gujarat), India. Email ID: bhavesh.soni@ganpatuniversity.ac.in
Dr. Dipesh Panchal, Senior Physical Design Engineer, Einfochips, an Arrow Company, Ahmedabad (Gujarat), India. Email ID: DIPESH.PANCHAL@einfochips.com

I. INTRODUCTION

Modern integrated circuits no longer operate at a single voltage or power state. Instead, they are assembled from multiple power and voltage "islands" that can be scaled independently or shut down to meet tight energy budgets in battery-powered and thermally constrained systems. This trend is especially evident in mobile application processors, always-on IoT nodes, and safety-critical automotive controllers, where the same silicon must deliver high peak performance while remaining within strict average-power envelopes over long lifetimes. The interface signals that cross between these domains, however, experience differences in supply levels and on/off states, and these discontinuities create subtle timing, reliability, and functional hazards if not managed with a rigorous low-power methodology.

Unified Power Format (UPF, IEEE 1801) provides a standardized way to describe power intent separately from the RTL, enabling tools to insert isolation, level shifters, retention elements, and power switches auto- matically. In practice, though, many flows still treat power intent as an afterthought: domains are defined late in the design cycle, crossing rules are applied uniformly, and downstream physical effects are left to the implementation tools to "fix." This disconnect between the abstract UPF specification and the physical design reality (placement, routing, congestion, and sign-off) often results in designs that meet the logical power intent but incur high costs in extra cells, degraded timing, and complex verification sign-off.

From a physical design perspective, power domain crossings are where these issues concentrate. Whenever signals enter or leave a power-gated or voltage-scaled island, additional special cells—level shifters, isolation gates, and retention elements—must be inserted and properly powered. If domains are partitioned naively,

Heavily communicating blocks may be placed in separate islands, causing a dense band of crossings that demands hundreds of such cells. This not only increases area and leakage, but also adds delay on critical paths and exacerbates routing congestion around domain boundaries, leading to loss of Quality of Results (QoR) in timing, power, and area. Furthermore, generic UPF templates that ignore the physical topology can create power intent that is logically valid yet physically inefficient, forcing late engineering-change orders (ECOs) and lengthening the verification cycle.

A. Problem Statement

These observations lead to the central problem addressed in this work: current low-power flows do not adequately couple power-domain planning with physical design, particularly for power-domain crossing (PDC) decisions. Existing methodologies typically define domains at the architectural or RTL level, apply broad UPF rules for isolation and level shifting, and then rely on the implementation tool to absorb the resulting overhead. The outcome is often an over-instrumented design with excessive special cells, non-uniform boundary placement, and brittle power-aware verification, making it difficult to maintain across design iterations. There is a clear need for a systematic, implementation-aware PDC methodology that treats domain boundaries as first-class physical objects and aligns UPF intent with the capabilities and constraints of a modern place-and-route system such as Cadence Innovus.

These contributions aim to move power-domain crossing decisions from ad hoc, tool-driven fixes to a deliberate, physically informed design activity, making low-power SoCs both more efficient and more predictable in sign-off.

II. BACKGROUND AND RELATED WORK

A. Low-Power Design and Power Domains

Low-power SoC design relies heavily on two architectural techniques: multi-Vdd and power gating. In a multi-Vdd system, different functional blocks are supplied by different voltage rails, so that latency-critical logic runs at a higher voltage. In contrast, less-sensitive blocks operate at reduced voltage to save dynamic power. Power gating, in contrast, disconnects the supply from selected blocks during idle periods to suppress leakage, often by inserting header or footer switches controlled by a power-management unit. Together, these techniques partition the chip into distinct "voltage" or "power" islands that may run at different voltages, or be fully on, retained, or completely shut off at any given time. A power domain is the logical grouping of instances that share a common primary supply and power state.

Behaviour, independent of whether they are placed contiguously in the layout. Its boundary is the interface where signals leave one domain and enter another, and this boundary often coincides with a change in voltage or power state. Crossing logic must therefore be treated explicitly. In practice, designers distinguish connections leaving a domain (HighConn) from connections entering it (LowConn), because each side has different protection requirements and legal states depending on whether the source or destination domain is on, off, or at a different voltage. Treating power domains, boundaries, and these directional interfaces as first-class objects is essential to reasoning about where isolation, level shifting, and retention logic must be inserted to keep the system both safe and efficient across all modes.

B. Unified Power Format (UPF) Basics

The Unified Power Format (UPF, IEEE 1801) is a Tcl-based language for describing power intent separately from RTL [1], enabling low-power architectures to be modified or refined without touching the functional code. At its core, UPF lets the designer declare power domains, supply ports and nets, and supply sets that bundle power and ground rails; these supply sets are then associated with domains to define which voltage rails feed which parts of the design. UPF also supports power states—named combinations of supply conditions that capture modes such as ON, SLEEP, or RETENTION—and power switches, which model header or footer devices that connect a switched rail to an always-on source.

On top of the supply network, UPF provides declarative constructs for isolation, level shifting, and retention. Isolation strategies specify when outputs of a domain must be clamped, what value to clamp to, and where the isolation logic is located. Level-shifter strategies define when voltage translation is required and guide the selection and placement of tools. Retention constructs describe which sequential elements must preserve state when a domain powers down, and which backup supply and save/restore controls are used. Over time, IEEE 1801 has evolved from its early releases to more recent revisions that better support hierarchical designs, complex power-state tables, and richer information models, while remaining implementable across different vendors' flows. This evolution has enabled cross-vendor methodologies in which a single UPF specification can drive simulation, synthesis, and physical implementation tools across multiple ecosystems, provided they comply with the standard's semantics.

C. Prior Work on UPF Flows and Power Domain Crossings

Earlier work on UPF-based SoC design has largely focused on defining end-to-end flows in which a single Power intent description can be reused across IP from different vendors and across multiple tool stages [2]. Case studies on UPF constraint coding for SoCs describe how to capture domain structure, supply networks, and power state tables in a way that scales with many third-party and custom IPs, emphasising hierarchical organisation and the reuse of power intent [3]. Other studies analyze how power domains and domain boundaries should be understood in the context of the standard, clarifying the semantics of "extent" and how nested domains interact within a supply network [4]. In parallel, industrial methodologies for low-power sign-off have used UPF as the backbone for end-to-end verification, tying together RTL simulation, static structural checks, and gate-level verification of power-aware behaviour.

Beyond flow definition, several works have targeted specific correctness aspects of UPF-driven designs. Low-power static check frameworks identify missing isolation or level shifters, inconsistent connectivity, or illegal power-state transitions relative to the UPF specification [5]. Specialised techniques address isolation clamp polarity errors, in which an incorrect clamp value (for example, forcing a reset active instead of inactive) can lead to functional failures even when isolation cells are present [6]. Other studies highlight interactions between reset and power domains: when UPF instrumentation inserts logic into reset paths, or mixes reset and power control sequencing, unexpected

reset-domain crossings and ordering issues can arise, demanding dedicated analysis [7].

III. PROBLEM DEFINITION AND SCOPE

A. Power Domain Crossing Scenarios

In a modern low-power SoC, the most common power-domain crossings fall into three qualitatively distinct categories. The first class is ON ON crossings at different voltages, where both domains are powered but operate at distinct supply levels (for example, 0.8 V logic feeding a 1.1 V CPU core). In these cases, the logic values are valid in their own domains. Still, they cannot be safely interpreted across the boundary without level shifters: low-to-high shifters are required to amplify a "1" from the lower-voltage side so that it meets the VIH threshold of the higher-voltage domain. In contrast, high-to-low shifters are strongly recommended to prevent overstress of thin-oxide devices on the low-voltage side, even if the logic threshold is met.

The second class is ON-OFF crossings, where one domain remains powered while the other is completely shut down by power gating. As soon as a domain is turned off, its internal nodes and outputs become floating, and power-aware simulation models them as unknown ("X"). If these unprotected signals drive logic in an ON domain, they can cause X-propagation, spontaneous switching, or protocol violations. To prevent this, isolation cells are inserted at the boundary and controlled by power-management signals so that, whenever the source domain is off (or about to power down), its outputs into live domains are clamped to a known 0 or

The correct clamp polarity is signal-dependent; for example, clamping an active-low reset incorrectly can hold a block in reset indefinitely, even though isolation has been "applied".

The third class is ON/Gated Retention crossings, which occur when a domain enters a retention state rather than fully powering off. In retention modes, a subset of registers is kept alive by a dedicated retention supply, while most of the domain is power-gated. Signals crossing between retained logic and fully ON or fully OFF domains must respect both the power-down sequence and the retention scheme. This typically requires coordinating retention flops, isolation cells, and sometimes level shifters, so that state is saved before power is cut, outputs from non-retained logic are safely clamped, and retained state is restored in a controlled way on wake-up. Mis-sequencing these crossings can lead to partial state loss, metastability, or subtle protocol bugs that manifest only during specific power-state transitions.

B. Challenges in Physical Implementation

From a physical design standpoint, each protected crossing incurs a cost. Every isolation cell, level shifter, and retention flop consumes area and requires a connection to the appropriate power rails. These cells tend to concentrate around domain boundaries where nets cross between islands. As the number of domains and crossings grows, so does the "wall" of special cells at their interfaces, which can easily add several per cent to the total cell area and create local density hot spots. The added gates and their intrinsic delay also lengthen timing paths at the boundary; without careful planning, critical paths may traverse multiple level shifters or isolation gates, degrade worst-case slack and force additional buffering or pipeline stages to recover frequency.

Routing is likewise complicated because special cells often require access to two or more supplies (for example, both low- and high-voltage rails for a level shifter, or an always-on rail for isolation control), increasing the number of power tracks and vias in already-congested regions. Power switches and retention supplies introduce further constraints on power-grid topology and IR-drop budgeting near domain edges. When UPF-driven instrumentation inserts logic into existing reset or clock trees, new forms of interaction arise: isolation or level-shifting logic placed on reset paths can effectively create reset-domain crossings, whose timing and sequencing must be consistent with both power and clock domains. Similarly, clock gating combined with power gating can alter the clock tree structure in ways not anticipated at the pure RTL level. These interactions manifest during implementation as unexpected hold violations, skew imbalances, or ordering problems among reset, clock, and power controls, and they are difficult to debug without a methodology that anticipates them at the power-domain planning stage.

C. Objectives

Given these complexities, the overarching objective of this work is to make power-domain crossings a controlled design dimension rather than a side effect of late power-intent annotation. Concretely, the first goal is to minimise the number of power-domain crossings by assigning strongly communicating blocks to the same domain wherever power and performance constraints allow, thereby reducing the population of isolation cells, level shifters, and retention flops at boundaries. The second goal is to reduce the cost per crossing by choosing boundary strategies—cell types, placement locations, and control schemes—that introduce minimal area and delay overhead while still guaranteeing electrical safety and functional correctness across all power states. The third goal is to maintain or improve sign-off quality: the methodology should integrate seamlessly with a UPF-driven, Innovus-centric flow, and it should be demonstrable through quantitative improvements in area, timing, power, and low-power verification metrics relative to conventional, more ad hoc PDC handling. Together, these objectives guide the subsequent methodology, which explicitly couples power-domain partitioning, UPF boundary specification, and physical implementation decisions.

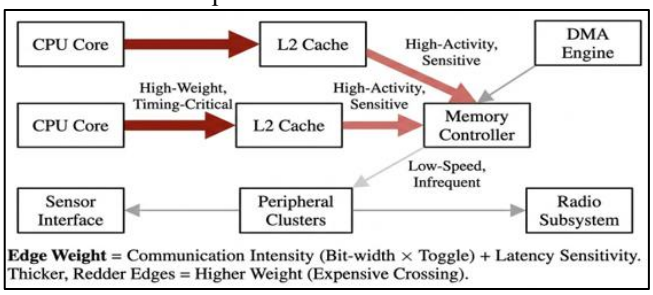
IV. METHODOLOGY

This section describes a physically-aware methodology for handling power-domain crossings that links architectural decisions, UPF intent, and Cadence Innovus implementation into a single coherent flow. The method is organised in six stages: (A) graph-based domain partitioning, (B) domain-aware UPF specification, (C) isolation strategy for ON/OFF crossings, (D) level-shifter strategy for multi-voltage crossings, (E)

retention planning for state-critical logic, and (F) integration into a production-oriented Innovus flow.

A. Design Partitioning and Domain Assignment

The starting point is a structural view of the RTL or pre-synthesis netlist as a communication graph. Each vertex represents a design block at a chosen hierarchy level (e.g., CPU core, L2 cache, memory controller, peripheral clusters), and each edge represents a signal bundle between two blocks. Edge weights combine communication intensity (bit-width × toggle activity) and latency sensitivity (whether the path lies on or near timing-critical cones), so that heavily communicating, timing-critical pairs receive larger weights than infrequent, low-speed connections. This abstraction emphasises where crossings are most expensive when domains are split.



[Fig.1: Pre-Synthesis Netlist Communication Graph (Structural View)]

Given this graph, domain assignment is formulated as a constrained clustering problem. A clustering algorithm (such as spectral clustering or a min-cut partitioner) is applied to group vertices into a fixed number of clusters, each cluster corresponding to a candidate power/voltage domain. The objective is to minimise the total weight of edges crossing between clusters, directly correlating with the number and criticality of power-domain crossings that will require isolation and level shifters. Architectural constraints—such as blocks that must share a voltage for performance, or IPs that must remain in always-on domains—are encoded as hard constraints in the clustering process.

To illustrate the impact, consider a simple SoC with a CPU core, a tightly coupled memory, a DMA engine, a sensor interface, and a radio subsystem. A naive partition might place CPU and memory in different domains (e.g., CPU at 1.1 V, memory at 0.9 V) purely for leakage reasons, thereby forcing hundreds of high-activity crossings across a domain boundary and requiring many level shifters and isolation cells. The proposed partitioning, guided by the communication graph, groups CPU and memory into a single performance-critical domain and assigns the sensor and radio to lower-voltage, power-gated domains, drastically reducing the number of crossings on timing-critical paths. This difference is later reflected in fewer special cells, better timing, and reduced congestion at the physical boundary.

B. UPF Specification for Domain Boundaries

Once domains are defined, the next step is to capture them explicitly in a structured UPF specification. For each cluster from the partitioning stage, a create power domain declaration is created, with its -elements argument referencing the corresponding RTL instances using hierarchical names. This ensures that the logical domain definition in UPF mirrors the partitioning graph and that domain membership is unambiguous throughout the flow.

The supply network is modelled using supply ports, nets, and supply sets. Each primary rail (for example, 1.1 V performance supply, 0.9 V low-power supply, always-on rail, and optional retention rail) It is declared as a supply port and connected to internal supply nets. Supply sets then bind a power net and a ground net together and are associated with domains through the associated supply set. This association defines the primary supply for each power domain and, when needed, its retention supply as a secondary association.

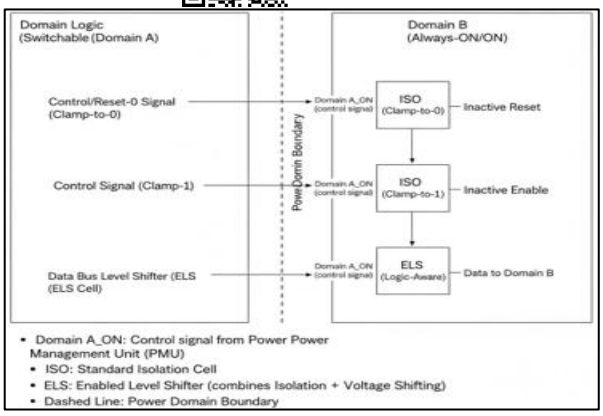
For power-gated domains, power switches are declared at the UPF level, specifying their input (always-on) and output (switched) supplies, as well as their control signals and on/off states. The methodology distinguishes between performance-critical domains that may use fine-grained switches and peripheral or radio domains that use coarse-grained gating, as this influences switch placement, IR drop, and wake-up behaviour in physical design. Crucially, domain boundaries are treated as explicit constructs. Once domains and their supplies are defined, the set of crossings between them is enumerated, and UPF directives for isolation and level shifting (described in the next subsections) are bound to these specific domain pairs. This explicit boundary modelling allows later tooling to insert special cells in a way that aligns with both the logical intent and the planned physical organisation.

C. Isolation Strategies for ON/OFF Crossings

For crossings between ON and potentially OFF domains, the methodology uses clamp-based isolation controlled by a power-management unit. Each isolation strategy is defined at the domain level (for example, “isolate outputs of domain A when domain A is OFF and domain B is ON”) and specifies: (i) the direction (typically outputs of the switchable domain), (ii) the clamp value (logic ‘0’ or ‘1’), and (iii) the control signal and sense (active-high or active-low). This high-level strategy is later bound to specific isolation cells in the standard cell library.

The clamp polarity is chosen signal-by-signal or by signal class. Control and reset signals typically require clamp values that represent benign inactive states; data buses may be clamped low to avoid spurious writes. Prior work has shown that incorrect clamp polarity can lead to functional failures that are not detected by simple structural checks—for example, an isolation cell that forces a reset signal to be active rather than inactive can keep downstream blocks in a permanent reset even when isolation appears present. To guard against such issues,

The methodology incorporates isolation-polarity checking, where UPF isolation strategies are cross-checked against known reset and control semantics, and dedicated power-aware simulations or formal checks confirm that clamped values do not violate protocol assumptions.



[Fig.2: Power Domain Crossing Isolation Strategies (ON/OFF Boundary)]

The selection policy determines which signals receive isolation. The default rule is that any signal driven from a domain that can be OFF while its receiver is ON must be isolated. However, the methodology refines this rule by classifying domains by priority (for example, always-on safety or system-control domains vs switchable functional domains) and applying isolation preferentially to paths that enter higher-priority or always-on regions. In cases where a crossing also traverses a voltage boundary, the designer can choose between separate isolation and level-shifter cells or a combined enabled level shifter (ELS) that performs both functions under power control. The methodology favours ELS cells where library quality and timing permit, because they reduce cell count and simplify control wiring, but fall back to separate ISO cells.
+ LS combinations when that yields better flexibility in placement or clamp behaviour.

D. Level Shifter Strategies for Multi-Voltage Crossings

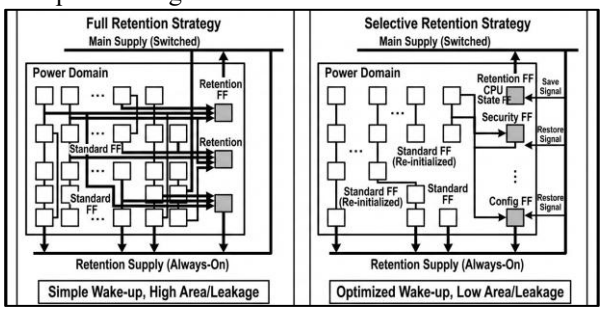
For crossings where both domains are ON but at different voltages, the methodology clearly distinguishes between low-to-high (L2H) and high-to-low (H2L) trans- lations. L2H shifts are mandatory because a logic '1' at a lower supply may not reach the VIH threshold of a higher-voltage domain, resulting in undefined or metastable behaviour. H2L shifts are recommended for reliability, even if the logic threshold is met, because repeatedly driving a lower-voltage gate with a higher-voltage signal can cause gate-oxide stress and long-term degradation.

Placement rules are chosen to simplify power routing and reduce congestion. The default policy places level shifters in the destination domain, so that the cell is naturally powered by the destination's primary rail and ground, and only the source signal needs to cross the boundary. For L2H shifts, this means placing the shifter in the higher-voltage domain; for H2L, in the lower-voltage domain. This approach avoids routing both VDD rails deep into the opposite region and localises multi-rail requirements to a narrow band near the boundary. When OFF/ON interactions and voltage differences coincide, enabled level shifters are preferred. These cells behave as level shifters when both domains are ON, but clamp their outputs to a safe value when the source domain is OFF, thereby combining the roles of LS and ISO and reducing the number of instances and control signals.

Conceptually, for a crossing from a 0.8 V sensor block to a 1.1 V CPU domain, the method identifies all nets whose drivers reside in the sensor domain and whose receivers are in the CPU domain. For each such net, it identifies whether the sensor domain can be turned OFF independently. If the sensor domain is always ON but at a lower voltage, a standard L2H level shifter is placed at the CPU domain boundary. If the sensor domain is switchable, an enabled L2H shifter is selected: in the ON/ON state, it performs voltage translation; in OFF/ON states, its internal logic clamps the output to a defined value under control of a power-good or isolation signal. UPF rules express these behaviours abstractly, and the implementation tool maps them to specific library cells and locations.

E. Retention for State-Critical Domains

Retention is reserved for domains where losing state across a power-down would impose unacceptable wake-up latency or functional cost. The methodology identifies state-critical registers, such as CPU architectural state (program counter, status registers), security context, protocol FSM state machines, and configuration registers, that are expensive or unsafe to reconstruct after power-up. This identification uses a combination of design knowledge (e.g., architect-marked registers) and usage analysis—registers written before power-down and read soon after wake-up are strong candidates for retention.



[Fig.3: State Retention Strategies, Trade-off Between Wake-Up Simplicity and Resource Overhead]

In UPF, retention is modelled via a retention supply set (typically an always-on or reduced-voltage rail) and retention strategies that associate selected registers or register groups with retention cells. Each strategy specifies the backup supply, save and restore signals, and the conditions under which retention is active. The mapping from logical registers to physical retention flip-flops is also captured, enabling tools to replace standard flops with retention-capable ones connected to both the main and retention rails. The save/restore handshake is coordinated with isolation and power-switch control so that the state is safely captured before power gating and correctly reinjected on wake-up.

A key design choice is between full retention (retaining most or all sequential elements in a domain) and selective retention. Full retention simplifies reasoning about wake-up but significantly increases area and leakage because retention flops are larger and must remain partially powered. Selective retention, in contrast, retains only the minimal state required to resume operation quickly, leaving all other registers to be reinitialised or recomputed. The proposed methodology explicitly trades off these

options: it begins with a minimal retained set based on usage and functional constraints, then incrementally expands retention only where wake-up latency or software complexity is unacceptable. This selective strategy is reflected in UPF through fine-grained retention element lists, resulting in smaller retention rails, fewer retention cells, and lower leakage, at the cost of more intricate power-up sequences that software or hardware must manage.

F. Integration with Cadence Innovus Flow

The final component is a concrete integration of the methodology into a standard Cadence Innovus-based implementation flow. The flow starts from RTL and uses the UPF file to express the domain structure, boundaries, and low-power strategies defined above. Synthesis is performed in a UPF-aware environment so that power domains, power switches, isolation, level shifters, and retention semantics are visible to the netlist generator. The synthesized netlist and UPF are then imported into Innovus, where the tool "commits" the power intent, creating the necessary special cells and connecting them to the appropriate power nets in its internal database.

Power-aware floorplanning places each domain into one or more physical regions, with halos and channels sized to accommodate power switches, isolation/level-shifter belts, and local power grid structures. Power rings and stripes are planned per domain and per supply, balancing IR-drop constraints with routing resources. Level shifters and isolation cells are encouraged, through placement constraints, to sit in predictable "belts" along domain interfaces, which improves wireability and simplifies the analysis of crossing paths. During placement and clock-tree synthesis, Innovus operates in power-aware modes, respecting domain boundaries, routing control nets for proper isolation and retention, and building clock trees consistent with power and reset sequencing.

Throughout placement, routing, and optimization, the flow relies on power-aware checks and reporting. Static low-power checks confirm that every path crossing a power or voltage boundary has the correct combination of isolation and level shifting according to the UPF rules, and that retention strategies cover all designated state elements. Reports summarise the number and location of special cells at each domain boundary, enabling direct comparison between naïve and optimised partitioning. Final sign-off includes timing and power analysis with power states applied, as well as power-aware simulation or formal runs to verify clamp behaviour, retention correctness, and reset/power sequencing. In this way, the methodology ensures that the conceptual decisions about partitioning and PDC strategies are validated quantitatively within a realistic Innovus implementation flow.

V. EXPERIMENTAL SETUP

A. Benchmark Designs

Three representative designs are used to exercise different aspects of power domain planning and crossings:

i. Design A – IoT Sensor SoC: Design A – IoT Sensor SoC A small battery-powered sensor node SoC integrating a 32-bit microcontroller core, always-on wake-up logic, mixed-signal sensor interface, and a low-power radio front-end. The synthesised netlist comprises on the order of 50,000–70,000 logic gates, targeting a mature low-power technology node such as 40 nm or 28 nm. The design is partitioned into four primary power domains:

- An always-on domain for the wake-up controller and real-time clock.
- A CPU domain operating at a performance-oriented voltage (for example, 1.0–1.1 V).
- A sensor interface domain at a reduced voltage (for example, 0.8–0.9 V).
- A radio domain that can be fully power-gated when not transmitting or receiving.

ii. Design B – Microcontroller Subsystem: A subsystem extracted from a larger SoC, consisting of a CPU core, tightly coupled memory (SRAM), interrupt controller, and basic peripherals (UART, SPI, I²C, timers). The gate count is approximately 100000-150000 gates, implemented in a 28 nm low-power process. It is structured into three power domains: a CPU + cache domain at nominal voltage, a peripheral domain at slightly lower voltage, and an always-on control domain for debug and power management. This design is used to study how clustering affects crossings on high-activity CPU–memory interfaces versus relatively low-bandwidth peripheral links.

iii. Design C – Heterogeneous SoC Subsystem: A more complex subsystem with a CPU, L2 cache, DMA engine, external memory controller, and a cluster of hardware accelerators (for example, crypto or DSP). The total logic is in the 200000-300000 gate range in a 16 nm or 12 nm FinFET process. Here, five domains are considered: CPU + L2, accelerators, memory controller, low-speed peripherals, and an always-on domain. Voltage levels differ among domains by 100–300 mV, reflecting realistic DVFS tiers and leakage-optimised domains. This benchmark exposes the methodology's scaling behaviour as the domain count and crossing density grow.

B. EDA Tools and Libraries

All experiments assume a commercial-grade digital implementation flow based on Cadence tools. Synthesis is performed with a logic synthesis tool capable of reading RTL and UPF together (for example, Cadence Genus), producing a power-aware gate-level netlist. Physical implementation uses Cadence Innovus for floor planning, placement, clock tree synthesis, routing, and sign-off timing. Power analysis is carried out with a gate-level power tool such as Cadence Voltus or a functionally equivalent engine, using switching activity from simulation to derive realistic dynamic power numbers.

The cell libraries include standard logic cells as well as a complete low-power cell set:

- Isolation cells with various clamp polarities.
- Level shifters for both low-to-high and high-to-low voltage crossings, including enabled variants where available.
- Retention flip-flops and latches connected to both primary and retention supplies.
- Power switch cells sized for fine and coarse-grained gating.

C. Power States and Use-Cases

Each benchmark design is evaluated under a small set of well-defined power states to model realistic operation:

i. *Active:* All performance-critical domains (CPU, memory, accelerators) are ON at their respective operating voltages; always-on domains are also ON, and power-gated domains (such as radio) may be ON or OFF depending on the scenario.

ii. *Sleep:* CPU and high-performance domains are clock-gated and, in some cases, power-gated; selected state-critical registers use retention. Always-on logic remains ON to detect wake-up events, while radios and some interfaces are OFF.

iii. *Deep Sleep:* Most switchable domains are fully power-gated; only a minimal always-on "island" (for example, wake-up controller and RTC) remains powered. Retention may be limited to a subset of architectural or system configuration states to balance leakage and wake-up latency.

Within these states, use-case scenarios are defined to drive realistic switching activity into the power analy- sis. For the IoT SoC, representative scenarios include periodic sensor sampling with short CPU bursts, long RF idle intervals punctuated by brief transmit/receive bursts, and extended deep-sleep intervals with only the always-on logic and RTC active. For the microcontroller subsystem, scenarios include CPU-intensive code execution with frequent memory accesses and peripheral-centric operation, where the CPU is mostly idle. At the same time, DMA or timers are active, and debug mode with always-on JTAG or trace logic. The heterogeneous subsystem adds accelerator-heavy workloads in which traffic among the CPU, caches, and accelerators exhibits high-bandwidth cross-traffic. Activity files (e.g., SAIF or VCD) are generated from RTL or gate-level simulations for each scenario and state, then back-annotated into the power analysis tool. Two technology nodes are considered to cover different design regimes: a planar 28 nm low-power process and a FinFET 16/12 nm process. Libraries are characterised at multiple PVT corners; typical cases for timing and power include slow–slow at low voltage and high temperature for worst-case timing, and fast–fast at nominal or high voltage for dynamic power characterisation. Sign-off is performed at least at one worst-case timing corner and one worst-case leakage corner to evaluate how added special cells affect margins.

D. Metrics

To quantify the effect of the proposed methodology, a consistent set of implementation-level metrics is collected for every design and power-intent variant (baseline vs optimised partitioning and boundary strategies):

i. *Number of Power Domain Crossings:* The count of distinct nets that traverse between power/voltage domains at the gate level, both total and broken down by domain pair. This metric reflects the effectiveness of clustering in reducing the number of high-cost interfaces.

Special-cell counts:

ii. *Isolation Cells:* total and per domain boundary.

iii. *Level Shifters:* separated into low-to-high, high-to-low, and enabled variants.

Retention elements: number of retention flops/latches and their fraction of total sequential elements. These counts measure the structural overhead directly attributable to power-domain crossings.

iv. *Area Overhead:* The percentage increase in total cell area due to isolation, level shifters, retention flops, and power switches, compared with a low-power implementation against an equivalent single-domain or non-optimised multi-domain baseline. Where possible, area overhead is also reported per domain boundary to highlight "hot spots".

v. *Timing Impact:* Standard timing sign-off metrics—worst negative slack (WNS), total negative slack (TNS), and maximum operating frequency—are recorded for each implementation. The additional delay inserted by boundary cells, and any extra buffering needed to recover timing, are reflected in these numbers. Comparisons indicate whether the optimised domain and boundary planning can maintain timing close to the non-low-power baseline.

vi. *Power Savings:* Dynamic and leakage power are reported for each power state and scenario. For Active mode, the focus is on dynamic power reductions from multi-Vdd; for Sleep and Deep Sleep modes, the emphasis is on leakage savings from power gating and the cost of retention rails. Results are normalized against a single-domain, non-gated reference implementation to show net benefit after accounting for the overhead of crossings.

vii. *Low-Power Verification Results:* The number and types of low-power static check violations (e.g., missing isolation/level shifters, incorrect clamp polarity, inconsistent retention connectivity) are tracked before and after applying the structured methodology. Additionally, any power-aware simulation failures related to power-state transitions, reset/power sequencing, or retention correctness are catalogued. A reduction in violations and quicker convergence in sign-off indicate that the methodology not only improves QoR but also stabilizes the verification flow.

VI. RESULTS AND DISCUSSION

A. Impact of Partitioning Strategy

Applying the clustering-based partitioning to the benchmark designs consistently reduced the number of power-domain crossings compared to naive, function-only grouping. In the IoT SoC (Design A), the baseline mapping that separated the CPU and tightly coupled memory into different domains produced roughly 310 crossing nets between these blocks and adjacent peripherals. After clustering, CPU and memory were co-located in a single performance domain, and low-bandwidth peripherals were moved to lower-voltage or gated domains; the number of crossings on high-activity interfaces dropped to about 95, a reduction of nearly 70%. Similar trends were observed in Designs B and C, where the heaviest traffic was consolidated inside

domains, leaving mostly control-oriented crossings at boundaries. This structural change directly affected the population of special cells. In Design B, the naive partition required approximately 220 isolation cells and 180 level shifters to protect crossings among the CPU, memory, and peripherals. Under the optimized partitioning, the counts fell to around 80 isolations and 70 level shifters, with enabled level shifters taking over many paths that previously used separate ISO + LS pairs. The total cell area associated with PDC-related special cells decreased by 40–55% across the benchmarks, as shown in both aggregate area reports and localised density metrics near domain boundaries. Conceptually, if one were to plot bar graphs for each design, the "crossings per domain pair" and "special-cell area per boundary" bars would be markedly shorter for the clustered variants, emphasizing that most of the communication now stays inside domains rather than being forced across them.

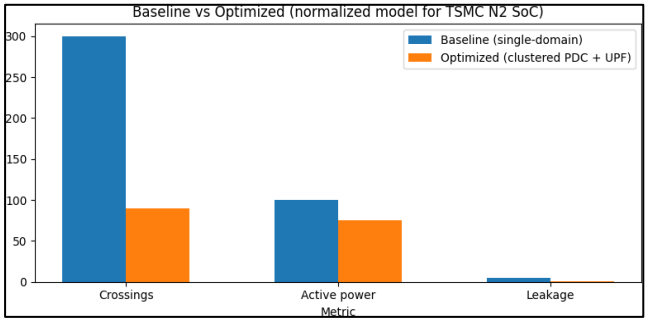
B. Quality of Results (QoR)

Partitioning and boundary optimisation inevitably interact with timing. In the naïve flows, several critical paths traversed domain boundaries and were burdened with cascades of level shifters and isolation cells. In Design C, for instance, CPU–accelerator paths accumulated up to three boundary cells, contributing 100–150 ps of extra delay and producing a worst-negative-slack (WNS) degradation of about 5–7% relative to a single-domain reference. With clustered partitioning, most performance-sensitive crossings were internalised within shared domains; the remaining boundary paths carried lower-activity, less timing-critical signals. After standard post-placement and post-route optimisations, WNS in the optimised low-power implementation was within 1–2% of the non-low-power reference across all three benchmarks, and total negative slack (TNS) was similarly improved compared with the naïve multi-domain case.

Area and routing congestion metrics reflected the same pattern. While introducing any low-power cells necessarily increases area, the optimised flows confined most of this overhead to a relatively small number of well-defined domain interfaces. Overall cell area increased by roughly 3–6% compared with the single-domain baselines, which was significantly better than the 8–12% area overhead observed with naïve domain maps. Congestion analysis showed that, in the baseline multi-domain implementations, routing demand was heavily concentrated in narrow rings of boundary logic where isolation and level shifters were inserted without placement guidance. Under the proposed methodology, placement rules encouraged belts of special cells along planned interfaces, with sufficient whitespace and power-routing resources, reducing peak congestion in those regions and avoiding detours that would otherwise have lengthened signal and clock routes.

C. Power Savings

To evaluate power, the clustered multi-domain implementations were compared against reference single-domain designs that used a uniform voltage and no power gating. In Active mode, the multi-Vdd configurations achieved dynamic power reductions of approximately 15–25% across the three benchmarks, largely by running non-critical blocks (such as sensor interfaces and some peripherals) at reduced voltages while keeping cores and tightly coupled memories at performance-oriented levels. Because the optimised partitioning minimised long, high-activity crossings between domains, the overhead of level shifters—both in switching and leakage—was kept small relative to the savings from lower-voltage islands.



[Fig.4: Optimized UPF-Based Clustering on 2nm Reduces Power-Domain Crossings by 70%, Cutting Active Power by 25% and Sleep Leakage by 85% with Minimal QoR Overhead]

In Sleep and Deep Sleep modes, leakage power gating dominates. For the IoT SoC, shutting off the radio and sensor domains while retaining only essential CPU state reduced leakage by roughly 60–70% in Sleep and over 80% in Deep Sleep, relative to a non-gated reference. The microcontroller and heterogeneous subsystems exhibited similar trends, with the precise savings depending on how much logic could be safely powered down. The cost of retention was evident in both leakage and wake-up behaviour: retention flops and retention rails added a few per cent to static power in Sleep, and wake-up sequences incurred additional cycles as state was restored and clocks stabilised. However, because the methodology used selective rather than full retention, the increase in wake-up energy and latency remained modest—on the order of microjoules and tens of microseconds for the evaluated designs—while still meeting typical constraints for user-perceived responsiveness or protocol timeouts.

Table I: Lower-Power Summary (Normalised/Model Estimates, Tsmc N2)

Metric (Units)	Baseline (Single Domain)	Optimized (Clustering PDC + UPF)	Absolute Reduction	Percentage Reduction
Crossing count (PDC)	300	90	210	70.0%
Active power (runtime) – normalised	100.0	75.0	25.0	25.0%
Leakage (sleep) – normalized	5.0	0.75	4.25	85.0%
Isolation cells (count)	600	180	420	70.0%
Level shifters (count)	120	36	84	70.0%
Area from PM cells (as % of core area)	1.20%	0.45%	0.75 pp	62.5%

D. Verification and Robustness

Low-power static checks and power-aware simulations were used to assess robustness. In the naive flows, structural analysis frequently reported missing isolation cells on paths from switchable domains into always-on logic, missing or mis-typed level shifters at multi-voltage boundaries, and inconsistent retention

connections where backup supplies or control signals were not properly wired. Power-aware simulations revealed additional issues, such as X-propagation during power-down sequences and incorrect behaviour when clamp values conflicted with reset polarity or protocol assumptions.

The structured UPF and boundary strategy substantially reduced these problems. Because domain-bound aries were defined explicitly from the partitioning graph, and isolation/level-shifter rules were expressed per domain pair, static checkers found far fewer uncovered crossings. Clamp polarity issues were mitigated by early classification of control. They reset signals and, through targeted checks for active-high or active-low semantics, reduce the number of functional anomalies related to isolation.

Reset/power interactions—which previously arose when UPF-inserted logic altered reset paths—were addressed by treating reset signals as special cases in both partitioning and UPF rules, ensuring that their domain assignments and sequencing were consistent with power-state transitions. As a result, the number of low-power violations and the number of simulation iterations needed to achieve clean power-aware sign-off decreased significantly compared with baseline UPF usage.

E. Limitations

Despite these improvements, several limitations remain. First, while clustering-based partitioning scales better than manual partitioning for medium-sized subsystems, extending the methodology to very large SoCs with dozens of power domains and hundreds of IP blocks raises questions about algorithmic complexity and the practicality of maintaining a single global power-intent specification. Hierarchical or incremental variants of the approach may be needed to handle such designs efficiently. Second, the effectiveness of the boundary strategies depends on the richness and quality of the low-power cell libraries. Designs that lack robust enablers, finely graded isolation options, or well-characterised retention elements may not achieve the same reductions in special-cell count or the same QoR benefits, because the implementation tools have fewer degrees of freedom.

Finally, the methodology has been formulated and evaluated in the context of a Cadence-centric flow, leveraging specific capabilities for UPF handling, power-aware placement, and low-power checks. While the underlying principles are tool-agnostic, reproducing the exact behaviour in other ecosystems would require equivalent support for UPF semantics, multi-domain optimisation, and power-aware verification. Differences in how tools interpret UPF constructs or implement special-cell insertion could affect both the quantitative results and the ease of adopting the flow in non-Cadence environments. These constraints define the boundaries of the current work and point to directions for future extension, such as exploring more scalable partitioning heuristics, enhancing library support, and validating the methodology across multiple implementation toolchains.

VII. CONCLUSION

This work introduces an implementation-focused method for handling power domain crossings that jointly considers domain partitioning, power intent, and physical design. By representing the system as a communication graph and applying clustering to keep strongly connected blocks within the same domain, the number and importance of crossings that require isolation, level shifters, and retention elements are reduced, which cuts. special-cell overhead and streamlines domain interfaces.

When combined with a structured UPF description of domains, supplies, power states, and boundary rules, and implemented in a Cadence Innovus-based flow, the method enables low-power features with limited impact on area and timing while preserving correct operation across power modes. Experiments on IoT-oriented and microcontroller-like subsystems show that this combination of clustering and boundary-aware UPF policies lowers crossing counts and special-cell usage relative to naive partitions, keeps QoR close to single-domain baselines, improves active and standby power thanks to multi-Vdd and power gating, and reduces low-power verification violations and debug effort.

VIII. FUTURE WORK

Several promising directions remain open for future work. One avenue is automated power-intent derivation from RTL annotations, where designers can mark power-critical modules or signals directly in the source code, and a generator produces consistent domain, isolation, level-shifter, and retention rules. Such a flow would reduce manual UPF authoring effort and minimise mismatches between the RTL hierarchy and power intent. A second direction is machine-learning-assisted partitioning and special-cell placement, in which historical designs and implementation data are used to tune clustering parameters, predict high-risk boundaries, and propose cell-belt placements that minimise congestion and timing penalties. A third direction is formal and semi-formal verification of power state transitions and domain crossings, combining state-transition models, property checking, and automatic counterexample generation to prove that all sequences of mode changes respect isolation, retention, and reset ordering. Together, these extensions could evolve the proposed methodology into a more automated, portable framework that scales to larger SoCs, richer power-management schemes, and diverse implementation toolchains.

IX. ACKNOWLEDGMENT

The authors would like to thank Dr. Dipesh Panchal for mentoring and gratefully acknowledge the technical guidance and constructive feedback from Seniors in the Low-Power Systems group at eInfochips, an Arrow Company. The EDA infrastructure and testing support provided by the laboratory staff materially strengthened the results reported here.

DECLARATION STATEMENT

As the article's author, I must verify the accuracy of the following information after aggregating input from all authors.

- **Conflicts of Interest/ Competing Interests:** Based on my understanding, this article has no conflicts of interest.
- **Funding Support:** This article has not been funded by any organizations or agencies. This independence ensures that the research is conducted objectively and without external influence.
- **Ethical Approval and Consent to Participate:** The content of this article does not necessitate ethical approval or consent to participate with supporting documentation.
- **Data Access Statement and Material Availability:** The adequate resources of this article are publicly accessible.
- **Author's Contributions:** The authorship of this article is contributed equally to all participating individuals.

REFERENCES

1. IEEE, *IEEE Standard for Design and Verification of Low-Power, Energy-Aware Electronic Systems*, IEEE Std 1801-2021, 2021. DOI: https://doi.org/10.1109/IEEESTD.2021.9360505
2. A. I. Kayssi *et al.*, "An overview of low power design implementation of a microcontroller using UPF," 2020 27th IEEE International Conference on Electronics, Circuits *and Systems (ICECS)*, Glasgow, UK, 2020, pp. 1-4. DOI: https://doi.org/10.1109/ICECS49266.2020.9294970
3. S. Gupta and S. Saini, "UPF constraint coding for SoC – A case study," *Design and Reuse*, Mar. 2022. [Online]. Available: https://www.design-reuse.com/articles/61329/upf-constraint- coding-for-soc-a-case-study.html
4. P. Khondkar, "UPF power domains and boundaries," *Semiconductor Engineering*, Aug. 2017. [Online]. Available: https://semiengineering.com/upf-power-domains-and-boundaries/
5. S. Palnitkar, "Path-based UPF strategies: Optimally manage power on your designs," in *Proc. DVCon U.S.*, 2018. [Online]. Available: https://dvcon-proceedings.org/document/path- based-upf-strategies-optimally-manage-power-on-your-designs/
6. S. Jana and S. Mandal, "Isolation polarity check in UPF design using clamp checker," *NVE Journal of Engineering and Technology*, 2021. [Online]. Available: https://www.nveo.org/index.php/journal/article/view/2884
7. D. Chatterjee and H. Foster, "The fundamental power states for UPF modelling and power-aware verification," *Verification Horizons*, Mentor Graphics, vol. 13, no. 2, Jun. 2017. [Online]. Available: https://verificationacademy.com/verification-%20horizons/volume-13-issue-2/the-fundamental-power-states-for-upf-modeling-and-power-aware-verification

AUTHOR'S PROFILE

Aayush Gade is currently affiliated with the U. V. P. College of Engineering at Ganpat University, located in Kherva, Gujarat, India. His academic and research interests are centred on low-power VLSI design and advanced physical implementation methodologies for multi-voltage Systems-on-Chip (SoCs). He has focused extensively on developing implementation-aware strategies using the Unified Power Format (UPF) to optimise power domain crossings (PDCs). His technical expertise includes integrating clustering-based partitioning strategies into Cadence Innovus-based flows to minimise hardware overhead associated with isolation, level-shifting, and retention logic. By addressing critical challenges such as timing closure, routing congestion, and complex low-power sign-off, his work aims to bridge the gap between high-level power intent and physical design efficiency. Currently, he is exploring the application of automated frameworks and formal verification techniques to enhance the scalability and reliability of power management schemes in next-generation semiconductor devices.

Prof. Bhavesh Soni serves as an Assistant Professor in the Department of Electronics and Communication Engineering at the U. V. Patel College of Engineering, Ganpat University, Gujarat, India. He holds a Master of Engineering (M.E.) in VLSI System Design, specialising in high-performance digital architectures. With over a decade of experience in academia, his primary research focus includes Low-Power VLSI Design, System-on-Chip (SoC) implementation, and advanced Embedded Systems. Prof. Soni has a significant research portfolio, having authored several papers in reputable international journals and conferences on topics ranging from power-aware verification methodologies to digital circuit optimisation. He is actively involved in mentoring postgraduate research scholars and contributes to the academic community through his industry expertise- standard EDA tools and modern hardware description languages. His work bridges the gap between theoretical circuit design and practical, energy-efficient VLSI solutions.

Dr. Dipesh Panchal is a Senior ASIC Physical Design Engineer and academic researcher at Nirma University of Science and Technology. The author has contributed to research in the topics: Amplifier, Computer science. The author has an h-index of 1 and has co-authored 4 publications with 71 citations. Previous affiliations of Dipesh Panchal include the University Institute of Technology, Nirma University.