An effective and efficient methodology for SoC power management through UPF

By

Renduchinthala Anusha

Under the Supervision of

Dr. Sujay Deb
Mr. Akhilesh Chandra Mishra
Mr. Rangarajan Ramanujam

Indraprastha Institute of Information Technology Delhi
June, 2016

An effective and efficient methodology for SoC power management through UPF

By

Renduchinthala Anusha

Submitted
in partial fulfilment of the requirements for the degree of
Master of Technology in Electronics & Communication
Engineering with specialization in VLSI & Embedded Systems

To

Indraprastha Institute of Information Technology Delhi
June, 2016

Abstract

With technology scaling and increase of chip complexity, power consumption of chip has been rising and its power architecture is getting complicated. Many power management techniques like power gating, multi-voltage, multi-threshold are applied to reduce power dissipation of devices. UPF is an IEEE 1801 standard format to describe the power architecture, also called as power intent, including power network connectivity and power reduction methods. It enables verification of power intent at early phases of the design cycle. The UPF developed should be consistent with the design at all stages of the design cycle and it should be updated according to the modifications made in the design.

In conventional UPF flow through design cycle, few practical challenges are faced. Many bugs are not detected at earlier phases which might lead to the wrong implementation of power intent. In addition, parallel development of power intent for complex designs, limitations of UPF standard to describe few power intent components effectively and time-consuming conventional UPF flow hinder efficient UPF development and management.

In this work, a UPF methodology is proposed which detects bugs at earlier stages and enables development of ideal UPF at RTL stage itself, which is to be just refined successively at later stages. This methodology also ensures proper restructuring and demotion of UPF along with automation of UPF development, demotion and verification. The proposed methodology is applied on the development of power-aware set-top box device, as a case study, with power gating and multi-supply voltage techniques applied. The results show that an overall error reduction of 81% is noticed with 63% at RTL stage. Verification time is reduced by 93% due to automation and early detection of issues. Automation of UPF development and demotion saved time by around 99% and 97% respectively.

Certificate

This is to certify that the thesis titled "**An effective and efficient methodology for SoC power management through UPF"** by **Renduchinthala Anusha** to the **Indraprastha Institute of Information Technology Delhi** for the award of the Master of Technology, is an original research work carried out by him under my supervision. In my opinion, the thesis has reached the standards fulfilling the requirements of the regulations relating to the degree.

The results contained in this thesis have not been submitted in part or full to any other university or institute for the award of any degree/diploma.

June, 2016

Dr. Sujay Deb
Department of ECE
Indraprastha Institute of Information Technology Delhi
New Delhi - 110020

June, 2016

Mr. Akhilesh Chandra Mishra
Consumer Product Division, ST Microelectronics
Greater Noida - 201308

June, 2016

Mr. Rangarajan Ramanujam
Consumer Product Division, ST Microelectronics
Greater Noida – 201308

Acknowledgement

This thesis would not have been possible without the guidance and the help of several individuals who in one way or another contributed and extended their valuable assistance in the preparation and completion of this study.

First and foremost, my sincere gratitude to **Dr. Sujay Deb**, Assistant Professor, IIIT Delhi, for his continuous support, motivation, immense knowledge. His guidance has helped me in all the time of research and writing of the thesis.

Mr. Rangarajan Ramanujam, Group Manager, CPD, ST Microelectronics, for accepting me as an intern in his team. I am grateful to him for his expertise, kind concern and consideration regarding my academic requirements.

Akhilesh C. Mishra, Group Manager, TRnD, ST Mircoelectronics, for his patience and steadfast encouragement, his constructive criticisms at different stages of my work were thought-provoking and they helped me focus on my ideas.

I am deeply grateful to **Surat Swarup Devulapalli**, Design Engineer, CPD, ST Microelectronics, for the long discussions that helped me in understanding the technical details of my work. I am also thankful to him for his unfailing support as my thesis adviser and for having faith in me.

Last but not the least, my family and dear friends. None of this would have been possible without their love. And the one above all of us, the omnipresent God for giving me the strength to plod on, thank you so much everyone.

Contents

List of Figures

List of Tables

List of Abbreviations

SoC	System of Chip
HDL	Hardware Description Language
VHDL	VHSIC Hardware Description Language
DVS	Dynamic Voltage Scaling
DVFS	Dynamic Voltage Frequency Scaling
PG	Power and Ground
RTL	Register Transfer Level
UPF	Unified Power Format
CPF	Common Power Format
ASIC	Application Specific Integrated Circuit
IP	Intellectual property
ALU	Arithmetic Logic Unit
PU	Partition Unit
EDA	Electronic Design Automation
SAIF	Switching Activity Interchange Format
IEEE	Institute of Electrical and Electronics Engineers

Chapter 1

Introduction

A decade ago, the major challenges of semiconductor device design were performance and area. Designers used to aim at achieving the required target speed in as minimum chip area as possible. Power was not of major concern those days. But at present, the entire System on Chip (SoC) industry is shifted to low power designing. Power has become one of the major challenges of semiconductor design along with performance and area. Many low power design techniques are being developed and implemented to minimize chip power consumption. In this chapter, the need for defining power intent through a standard format and challenges faced in taking it through design cycle are discussed along with a brief introduction to proposed solution.

1.1 Reasons for power increase:

Electronic devices are transformed from large computers to handheld devices. Consumers are demanding for more and more features in a single portable device with long battery life. The addition of newer features to SoC complexes the functionality of the device and in turn, leads to higher power dissipation heating up the device and may lead to its damage. Moreover, the life of a battery depends on the power drawn by the system. So, for the long lasting functioning of devices, the overall power consumption of chip should be within acceptable limits.

Technology has been shrinking very rapidly resulting in denser chips. According to Moore's law, the number of transistors in an integrated circuit doubles every two years [1]. With every technology node introduced, almost twice the number of transistors are being placed in almost same chip area. This greatly impacts overall power consumption of a device. As the density of chip is getting increased, the power dissipation of the overall chip is almost raising by 2.7X for every new process technology [2].

It is well known that power consumption of a device depends on the supply voltage [3]. With technology scaling, the supply voltage has also been decreasing but not at the same pace as the increase in density of chip as shown in Table 1.1 [4]. This results in higher power dissipation at newer technology nodes.

Another factor impacting chip power consumption is the frequency of operation. To increase the speeds of electronic devices as per the user demands and meet the target performance, more number of components are being placed in same chip area and are being operated at higher frequencies. This leads to a drastic increase in power consumption of chip as dynamic power is directly proportional to the frequency of operation.

Technology	Gate density (in mm^2)	Supply Voltage
90 nm	350 K	1.0 V
65 nm	700 K	1.0 V
40 nm	1750 K	0.9 V
28 nm	3400	0.85 V

Table 1.1 Change in gate density and supply voltage with technology scaling

With every new generation of technology, threshold voltage values of transistor devices are getting reduced. Leakage power of a transistor, which is the power dissipated by the device when it is not in active state, increases with the decrease of threshold voltage leading to higher power dissipation [3]. Moreover, to make the chips work at higher speeds, the threshold voltage of transistors is being varied as delay of a transistor depends on threshold voltage level. Lower threshold devices are preferred as they decrease the delays enhancing the speed of operation which in turn leads to increase of leakage power. Leakage power has become a major contributor along with dynamic power for overall chip power consumption at advanced technology nodes, especially below 65 nm [5].

1.2 Need to define and verify power intent at earlier stages of design cycle:

In past, power connections to the design were made at the physical implementation stage. As the number of cells per chip was lesser, it was easy to manage connecting power pins of cells to power supplies and handle power related issues even at the physical implementation stage. Moreover, all the blocks in the design were treated always ON as the power consumption of the chip was within acceptable limits due to lesser chip density. So, power architecture of chip was very simple, just with supply ports and supply nets, as there were no power reduction methods applied to complicate it.

In recent years, chip's power dissipation is dramatically raised with technology scaling and an increase of design complexity as discussed in section 1.1. To reduce power, many power reduction methods are being applied. These include power gating, multi-voltage technique, multi-threshold technique, Dynamic Voltage Frequency Scaling (DVFS) and much more [4]. Application of few of these methods requires special cells like isolation cells, level shifter cells, power switches, etc. to be inserted into the design. For example, few non-active blocks are switched off in power gating method. For proper operation of active blocks connected to

these switched off blocks, isolation cells are to be inserted between those blocks which ensure valid signal transmission. Level shifter cells are needed between the blocks supplied with different voltage levels. A detailed description of frequently used power reduction methods and special cells is given in section 2.2.
It is to be ensured that the power reduction methods applied do not disturb the functionality of chip as these may lead to switching off few blocks in few modes of operation, operating the design at different voltage and frequency levels and insertion of some special cells. Moreover, power intent has become complicated with an increase of chip density. This paved a way for the need to define power intent including power management methods so that these can be verified easily along with power pin connections of cells to the power supplies. But if power intent is defined and verified late in the design cycle, then bug fixing at that stage is very expensive in terms of time and effort. Hence power architecture of a SoC is defined as early as at the RTL stage in the design cycle. In addition, one will have more options to play with if power reduction methods are defined and verified at earlier stages so that necessary modifications in the design can be made at later stages without affecting its functionality.

1.3 Means of defining power intent

There are various Hardware Description Languages (HDL) like Verilog, VHDL and System Verilog to define the functionality of a design. Once the design is defined, it can be easily verified using relevant simulators. In the same way, a standard language or description is needed to define and verify power architecture of a chip.

Common Power Format (CPF) and Unified Power Format (UPF) are standard formats usually used to describe the power intent. .attx files are also used which are just a textual representation as per user requirement and convenience and are very specific to an organisation's development environment. So, the entire design can be described by using Functional Netlist, which defines its functionality and Power Intent, which gives information related to optimize powering of design components, as shown in Figure 1.1.

a) ***CPF:*** CPF is developed by Cadence to specify power contents of a design and verify power modes [6]. Cadence also proposed a low power design flow to describe power intent at earlier stages in design cycle making use of CPF [7]. But this methodology and CPF file are mostly specific to Cadence tool set which imposes a restriction on the usage of different vendor tools leading to compromise in efficiency.

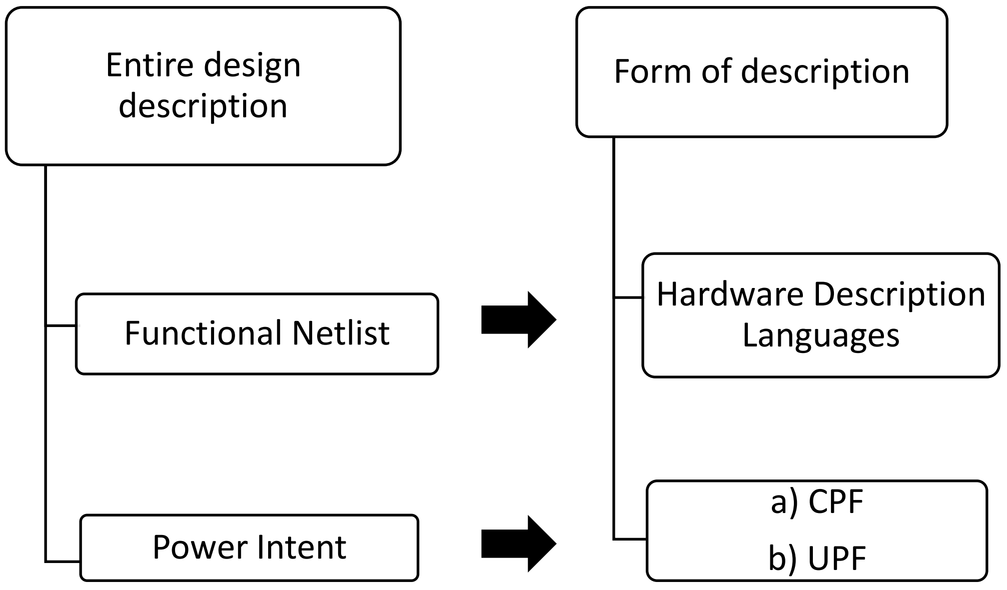


Figure 1.1: Division of design description and forms of defining

b) ***UPF:*** To allow interoperability among most of the EDA tools for taking power intent through the design cycle, a more generalised power format named UPF 1.0 was proposed by Accellera organisation in 2007 [8] making use of some of the features of CPF [9]. UPF can be used for simulation and static verification of power intent and it supports Tcl syntax and semantics. UPF is constantly being evolved to address the requirements resulting in an IEEE 1801 UPF standard in 2009, popularly called UPF 2.0 version followed by UPF 2.1 in 2013 and UPF 3.0 in 2015 [10]. The main components of UPF include power intent commands to describe power architecture, package UPF for creating test benches, query commands to customize reports and Switching Activity Interchange Format (SAIF) to store activity data for power analysis [11].

1.4 Conventional UPF flow in coordination with design cycle

Once power intent is developed at RTL stage through UPF, it is verified against the design for consistency. But the design undergoes many changes through design cycle and hence the power intent developed at RTL stage will no longer be consistent with the design. So, UPF should also flow through the design cycle in coordination with the design changes.

Let us now have a broad overview of the design flow, more popularly called as Application Specific Integrated Circuit (ASIC) flow. A design gets transformed mainly at three stages:

RTL stage, Synthesis stage and Physical Implementation stage. Based on specifications, a design is described in one of the HDL at RTL stage. It is functionally verified through simulation. Then it is converted into gate level netlist at Synthesis stage. The gate level netlist is now verified for functionality and timing. Finally at Physical Implementation stage, the design is physically implemented giving out all pins netlist.

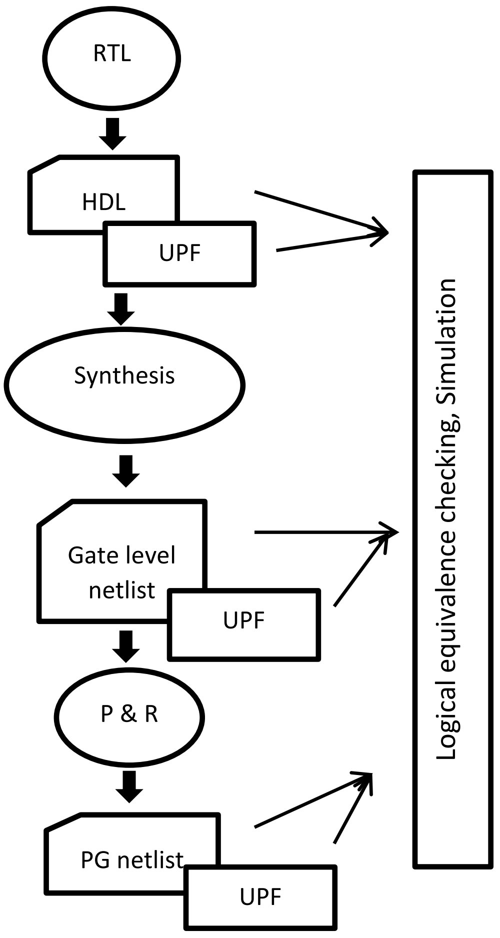


Figure 1.2: Conventional UPF flow through design cycle

Now let us discuss UPF flow merged with basic design flow which is shown in Figure 1.2 [12]. Based on power specifications, power intent is defined and described using UPF at RTL stage. Power intent is verified against RTL level design dynamically and statically (2.3). Note that at RTL stage, no special management cells like isolation cells, level shifters and retention cells are present in the design. But the management strategies are defined in UPF based on which the verification is done. At synthesis stage, special management cells are inserted in the design according to strategies described in UPF. As the design now has additional cells and might undergo few other changes, UPF is described accordingly and verified against gate level netlist. At Physical Implementation stage, few cells like clamp cells are added into the design. UPF is described as per the design and verified against all pins netlist.

1.5 Challenges in UPF development and conventional UPF flow

There are a lot of problems which are faced during the development of power intent and taking it through the design cycle.

a) ***Unreliable port declarations:*** Power architecture of a SoC includes pad ring from which main supplies enter into the core, padcell connections, analog block connections, digital block connections, power specifications of various IPs, central bump supplies, if flip chip package [13] is used, as shown in Figure 1.3, along with special cells according to power reduction techniques applies and power sequencing information.

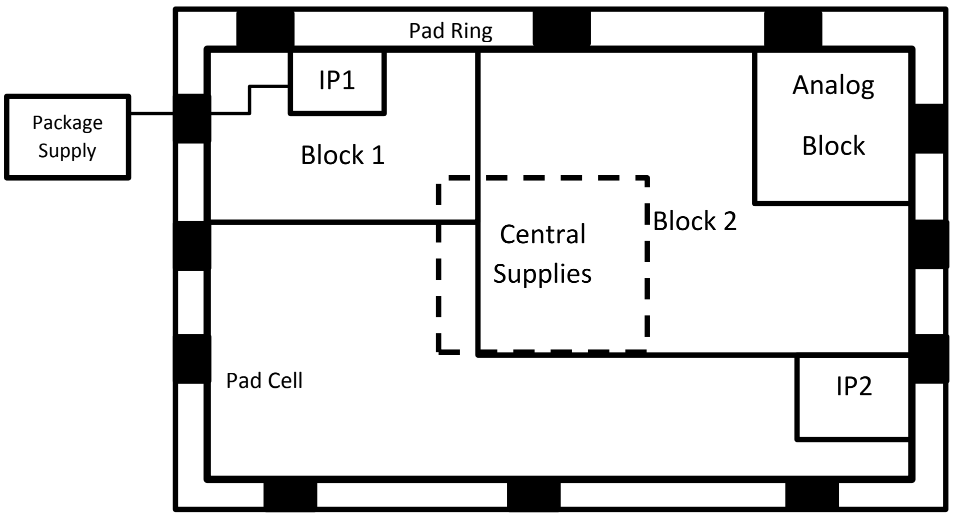


Figure 1.3: An example power architecture of SoC

Generally, all this data is defined and described by various specialized teams. But when integrated at SoC level for the development of entire power intent, this might lead to the wrong implementation due to the involvement of various sources. The inconsistencies like unreliable port declarations cannot be detected at earlier stages and sneak to the later stages, which as a result lead to shorts or opens at the final implementation stage. Fixing of these bugs identified late in the design cycle is very cost effective.

b) ***Bus management:*** A bus is a group of bit lines. A bus has its ports with specific width. For example, a bus with 16 bit width has an input and output port which are 16 bit wide. The buses in a design, especially the ones coming from hard macros are represented as a single in or out pin. But if they are treated in the same way in UPF without taking care of width, the specifications defined in UPF for that bus are applied only to a single bit line and remaining are left floating.

c) ***Mixing of power and signal connections:*** A cell has power pins to power up the cell and signal pins for in/out operations. A signal pin is related to a power supply to define its voltage range. But there arise situations where a signal pin needs to be grounded or connected to supply voltage. In such cases, if it is declared as another supply pin in UPF, the verification tool gets confused to infer it as signal pin or supply pin and flags errors.

d) ***Time-consuming UPF development:*** A SoC in recent years has millions of gates and is very complex. The power architecture of such a design has a lot of components to deal with (2.1 and 2.2). In addition, the power related data comes from various sources as discussed earlier and is not necessarily represented in UPF format but in some other nonstandard formats. It is very time consuming to integrate all the data converting into a standard format and to generate top level UPF manually, taking care of each and every component in the power intent.

e) ***Restructuring and demotion of UPF:*** A design is divided into various partition units, called restructuring, for parallel development at later stages to reduce time to market and as it is also easier to handle smaller chunks. When the design undergoes restructuring, top level UPF should also be restructured accordingly and demoted to partition level satisfying the power specifications of each design unit. This results in dedicated power intent for each unit so that partition level UPFs can be taken along with the design partitions to next stages and can be modified and verified accordingly.

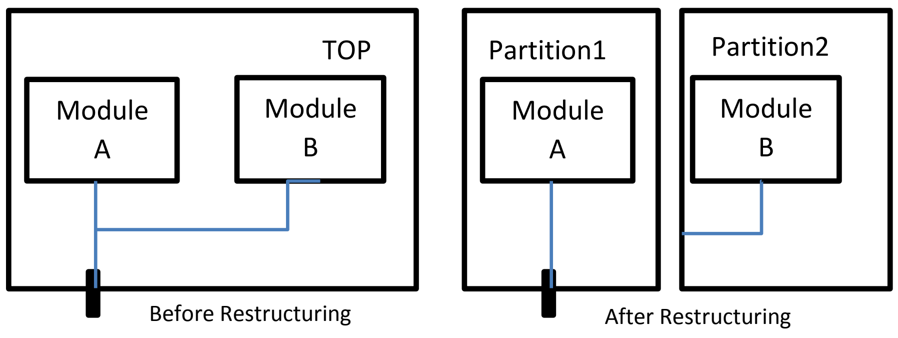


Figure 1.4: Supply net problem during restructuring

We know very well that restructuring of a design results in hierarchical changes. This creates a problem in demoting UPF as the elements described in it should also be modified accordingly taking care of their scope and power domains (2.1.1) created at the top level. In addition, if same supply net (2.1.2) is connected to elements in different partition units as per the restructured design, then the supply net in one partition unit is left hanging as the supply port to which it is connected is declared only once at top level and is in the other partition as shown in Figure 1.4. This leads to the wrong implementation of partitioned power intents. Analysing and handling all these issues manually consumes lot of time.

f) ***Time-consuming static verification of power intent:*** Static verification of power intent is usually done at three main stages of design cycle: RTL stage, synthesis stage and physical implementation stage, as the design undergoes major transformations at these stages. The list of static checks to be performed varies with design stage (2.3). And we know that design goes through many iterations during the design development process and so is the UPF. The manual approach of selecting the checklist based on design stage every time is very error-prone and time-consuming.

1.6 Motivation and Proposed Solution

- Power has become one of the major design constraints due to the increase in complexity of chips and high demand for low power devices in the market. The concept of developing power intent for such complex chips separately through a standard format, apart from design functionality description, is introduced very recently, just a few years back and the organisations are still in the process of adapting and using it effectively.
- Parallel development of power architecture of a design leads to a lot of inconsistencies and leaves top level UPF integrator with basic knowledge of power intent. Any misrepresentation of power components in UPF may not be caught at earlier stages resulting in inefficient power-aware designs.
- Conventional UPF flow is not fully matured and should be able to address practical implementation challenges like restructuring and demotion.
- Development of UPF at the top level, restructuring and demoting it according to design partitioning and static verification of UPF are time-consuming and also inefficient if done manually. These indirectly delay the entire design development process increasing the time to market.

In this work, solutions are proposed to overcome the above challenges, which are described below very briefly and an effective and efficient methodology is introduced for development and management of power intent at SoC level through UPF (Chapter 5).

- Alignment of input power data: The root causes of inconsistencies in the power intent description are analysed and an alignment process for power data received from various sources is proposed, which should be applied much before the generation and RTL level verification of top level UPF.

- Restructuring and demotion process: The components of top level power intent which get affected in restructuring process are analysed and a generalized UPF restructuring and demotion process is proposed. Some part of the process is also automated to save the time.
- Automation of UPF generation: An algorithm is proposed for developing UPF at top level automatically, taking in data related to entire power intent. It also takes care of hierarchical path changes before and after restructuring process automatically.
- Automation of UPF static verification: A verification environment is introduced which saves time for static verification of UPF and is flexible enough to customize as per the user requirements.

1.7 Thesis organization

Next chapters are organised as follows. Chapter 2 gives an insight into the basic components of power architecture and their description using UPF along with an introduction to static verification checks. Chapter 3 discusses the work done so far in this domain and current methodologies. The proposed solutions and overall UPF methodology are explained in Chapter 4 followed with a discussion of results of implementing it in Chapter 5. Finally, the work is concluded in Chapter 6 along with its future scope.

Chapter 2

Conceptual Overview

This chapter builds a fundamental idea about power architecture of present-day chips and the checks to be done to verify it. The basic components of power intent are explained along with UPF commands to describe them (2.1). It is to be noted that UPF 1.0 version of commands are specified with the examples as it helps to understand the basic concepts in a better way and most of them are carried forward to latest versions with minor changes. A brief introduction to power reduction techniques is given and special cells to be added for proper application of those methods are discussed simultaneously (2.2). Finally, the main checks that are to be performed at various stages of design cycle for static verification of UPF are mentioned (2.3).

2. 1 Components of a basic Power Intent

The basic components of power architecture of a design include Power Domains and Supply network.

2.1.1 Power Domain

A power domain is a group of logic in a design which has a common power supply and has similar power characteristics. In other words, logic blocks which can be operated with the same power supply are identified and grouped together to form a power domain with the condition that their supply voltage change or switching to ON/OFF can be done all at the same time without affecting functionality of the chip. Hence a power domain can be completely shut down based on its state or kept always ON; can be operated at different voltages or at the same voltage [4].

A power domain is created using UPF command ‘create_power_domain’ [12][14]. For example, let us consider a design as shown in Figure 2.1. It has a module A with two sub-modules B and C which have different power characteristics. So, three power domains are created using ‘create_power_domain’ command as shown below. But before writing the commands, the scope is set which specifies at what hierarchical level of design the commands are to be applied.

```
set_scope A

create_power_domain PDA –include_scope

create_power_domain PDB –elements {B}

create_power_domain PDC –elements {C}
```

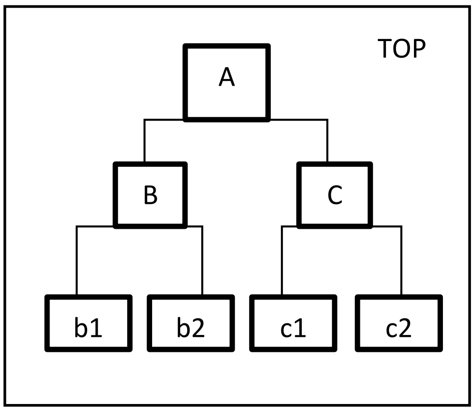


Figure 2.1: A design example

The power domains are created as shown in Figure 2.2. The –include_scope option includes all the elements in the current scope which consists of A, B and C. But as succeeding commands are given preference, PDB and PDC come under power domains PDB and PDC respectively.

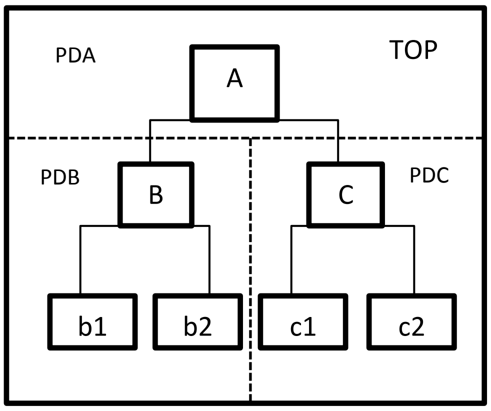


Figure 2.2: Design with power domains

2.1.2 Supply Network

The supply network in power architecture includes supply ports, supply nets and supply sets. Power to the cells in the design comes from supply ports through the supply nets. Depending upon power requirement, a design may have a large number of supply ports each operating at same or different voltage levels. Practically, the power to supply ports comes from main power sources, placed on or off the chip, through I/O padring [15]. The supply nets are used to connect supply ports to power pins of the cells.

A supply port is created using 'create_supply_port' command and 'create_supply_net' is used to define a supply net [12][14]. And the supply ports are connected to supply nets using 'connect_supply_net' command. Finally, the respective supply nets are specified to power domains using 'set_domain_supply_net' [12]. Once the supply net is assigned to a power domain, all the instances in power domain are connected to that supply net by default.

But instead of creating feeder nets explicitly for domains, supply sets can be used [16]. Supply set concept is introduced in UPF 2.0 version and a supply set is like a multi-conductor power cord with six pre-defined functions where each function represents a supply net. It can be created using 'create_supply_set' command [14].

Let us now consider that the power domains PDA and PDB, created in the example of section 2.1.1, operate at the same voltage, say V1 and power domain PDC operates at a voltage, say V2 where V2 < V1. After creation of two supply ports, one for V1 and the other for V2 along with common ground, GND and corresponding supply nets Pwr1, Pwr2 and gnd, the power intent of design will be as shown in Figure 2.3.

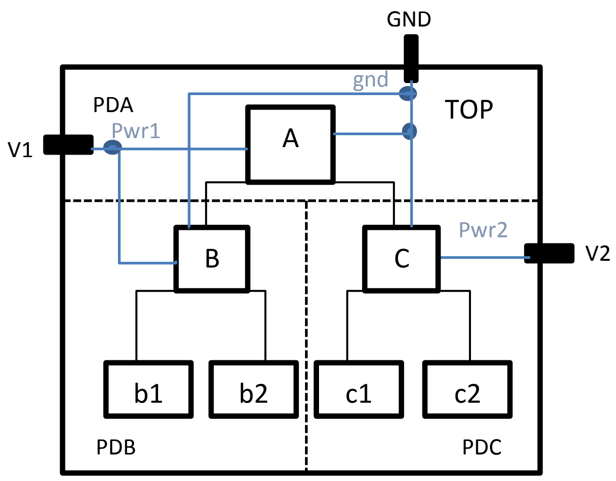


Figure 2.3: Design with supply network and power domains

2. 2 Power reduction techniques and special management cells

Power reduction techniques are used to reduce the power consumption of a design. The popular power management methods include Power Gating, Multi-voltage, Dynamic voltage scaling (DVS), Dynamic Voltage Frequency Scaling (DVFS), Multi-threshold method, Active Body Bias. Few of those aim at reducing dynamic power and few others decrease leakage power [17]. In this section, power gating and multi-voltage techniques, the basic and more general reduction methods, are discussed which help to understand the need of special cells like Isolation cells, level shifters and retention cells in the power intent when power reduction techniques are applied.

2.2.1 Power Gating

All blocks of a design are not active all the time. Depending upon the mode of operation, few blocks may be functioning and the remaining blocks may be in the inactive state. The inactive blocks result in leakage power as power supplies are connected to them though they are not using the power. Power Gating is a technique to reduce this leakage power by switching OFF power supply to the inactive blocks. For proper application of power gating technique to a design, we need to use some special cells like switches, isolation cells and retention cells.

a) Power Switch: To turn OFF power supply to a power domain for a particular time of operation, we need a power switch to cut the power supply to the domain. A switch, if placed between main power supply and block, is called header switch cell and a switch placed between ground and block is called footer switch cell [4]. Either of the switches solves the purpose. This switch is turned ON or OFF depending upon the mode of operation which determines the activeness of the block and this switching is controlled by a control signal.

A switch is described in UPF using ‘create_power_switch’ command [12]. The input of switch is the main power supply and the output is set as supply net to the domain [18].

b) Isolation cells: When Power Gating method is applied to a design, in a particular mode of operation, we might end up with few ON domains and few OFF domains. If the outputs of the shutdown domain are connected to the active part of the design, then it might lead to invalid signal transmission and crowbar current because the output pins of OFF domain might be in the metastable state. To avoid this, an isolation cell is placed on the output nets of switched OFF domains interacting with an active portion of the design.

An isolation cell clamps the signal at its input pin to a defined known state, either logic ‘0’ or logic ‘1’, thus ensuring a valid voltage level transmission to the ON domains. It is controlled by an enable signal which goes active when the domain connected to the cell’s input goes OFF. If the cell is enabled, it clamps the signal to a known state and if it is not enabled, it transmits the signals from input to output without any change.

An isolation cell strategy is defined in UPF 1.0 using ‘set_isolation’, ‘set_isolation_control’ and ‘map_isolation_cell’ commands [12]. But in UPF 2.1 version, last two commands are deprecated and only ‘set_isolation’ is used to describe the entire isolation strategy [19].

c) *Retention cells:* A retention cell is used to retain the data required in shut down power domain. A retention cell consists of a flipflop and a save latch and has two control signals, SAVE and RESTORE [4]. SAVE signal indicates when the data should be saved in the latch, which is just before switching OFF the power. RESTORE signal tells when the data stored in latch should be restored which is when the domain gets back to active state.

   A retention cell strategy is defined in UPF 1.0 using ‘set_retention’, ‘set_retention_control’ and ‘map_retention_cell’ commands [12]. But in UPF 2.1 version, ‘set_retention_control’ is deprecated and in addition, ‘set_retention_elements’ command is used to indicate the cells whose data should be retained [19].

Let us now consider our previous example with three power domains A, B and C with B switchable while the remaining two are always ON. Let us say that the design has three modes of operation; mode 1, mode 2 and mode 3. In mode 1, all the blocks are ON. In mode 2, say domain B is inactive, so it is shut down to save power and in mode 3, the entire chip is OFF. The modes of operation for this design are as shown in Table 2.1.

Operation Mode	PDA	PDB	PDC
Mode 1	ON	ON	ON
Mode 2	ON	OFF	ON
Mode 3	OFF	OFF	OFF

Table 2.1: Modes of operation for considered design

As domain B is switchable, we need to place a switch in between main power supply port V1 and domain B. The output of switch, ‘Vsw’ is set as supply net to power domain B. Since ‘Vsw’ has two states, ON and OFF, these should be defined in UPF file for exactly replicating the actual behaviour of the port during simulations. This is done using ‘add_port_state’ command [12]. Port states should also be defined for each and every other supply port in the design for a clear description of port behaviour.

Once the port states are defined, the operation modes of a design shown in Table 2.1 are to be defined in UPF file. This table is called power state table and is created using ‘create_pst’ command [12]. The states are defined with respect to port states using ‘add_pst_state’ command.

In mode 2 where domain B is shut down and as outputs of domain B are connected to remaining two blocks, isolation cells are to be placed at the output pins of domain B to avoid corruption of data and crowbar current. If data in some registers of domain B are wished to be retained when it is OFF, retention cells are placed.

2.2.2 Multi-Voltage technique

In multi-voltage method, different blocks are operated at different supply voltages. Depending upon the characteristics, few blocks can work well even at lower supply voltages compared to other blocks in the design. So, by decreasing the supply voltage level to that type of blocks, a considerable amount of dynamic power can be reduced. Moreover, a block operating at a specific supply voltage in one operation mode may work well in another mode even if the supply voltage level is decreased. In such case, the supply voltage value for the same block is changed dynamically based on the state of operation which is called DVS.

When two blocks powered with different voltage levels interact with each other, invalid signal transmissions and crow current generation may take place. To avoid this, level shifter cells are placed between the blocks with distinct voltage values. A level shifter is a cell which shifts the voltage level of a signal. Level shifter cell can convert a high voltage level to low voltage level or a low voltage level to a high voltage level depending on its functionality ensuring valid voltage level signal transmission between the domains.

A level shifter strategy is described in UPF 1.0 version using ‘set_level_shifter’ and ‘map_level_shifter_cell’ commands [12]. But the mapping of level shifter is defined in ‘set_level_shifter’ itself in UPF 2.1 version [19].

Let us go to our design mentioned in previous sections. In that design, we know that the power domain C operates at a lower voltage, V2 compared to remaining two domains. So, we need to place level shifter cells around the power domain C for valid signal transmission.

Finally, the modified power architecture of the design, after insertion of special management cells looks like the one shown in Figure 2.4. This is the simplest power intent of a design with power gating and multi-voltage techniques applied.

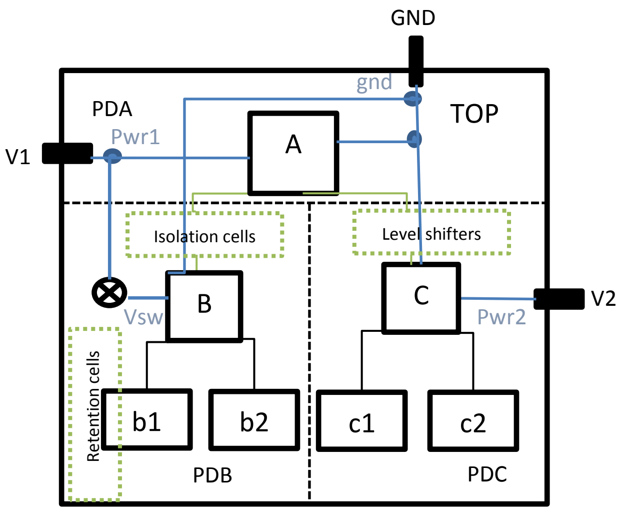


Figure 2.4: A simple power intent for considered design.

2. 3 Static Verification of power intent

Power intent described through UPF should be verified dynamically and statically before it is implemented.

Static verification mainly verifies

i. connectivity of power pins of cells to supplies
ii. connectivity and placement of special management cells like isolation cells, level shifter cells, retention cells and power switches
iii. UPF consistency with the design

Dynamic verification or power aware simulation verifies

i. if power domains are transitioning at the right time according to change in the operation mode, called power sequencing verification
ii. if special cells are enabled and disabled at the proper time
iii. if the design is functioning properly after the application of power management methods [20].

In this section, we focus only on static verification of UPF and the checks to be done at different stages of the design cycle. Static checks are broadly classified into five groups [21]:

a) *Power intent consistency checks*: verify syntax and semantics of UPF along with its consistency
b) *Structural checks*: verify the placement and connections of special management cells, once they are implemented in the design
c) *Signal corruption checks*: verify signal validity through enabling control signals of special management cells

d) *Functional checks*: verify the functionality of special management cells
e) *Power and ground (PG) checks*: verify power and ground connectivity of cells against UPF specifications at post route stage

All checks are not done at all the stages of the design cycle. Structural and functional checks are performed only after the implementation of special management cells in the design, which is usually done during or after synthesis stage. Similarly, PG checks are done only if the design is routed and have its PG pins connected to the relevant nets. So, these checks are performed on routed design against UPF at that stage. The type of checks performed at different stages of design is as shown in Figure 2.5 [21].

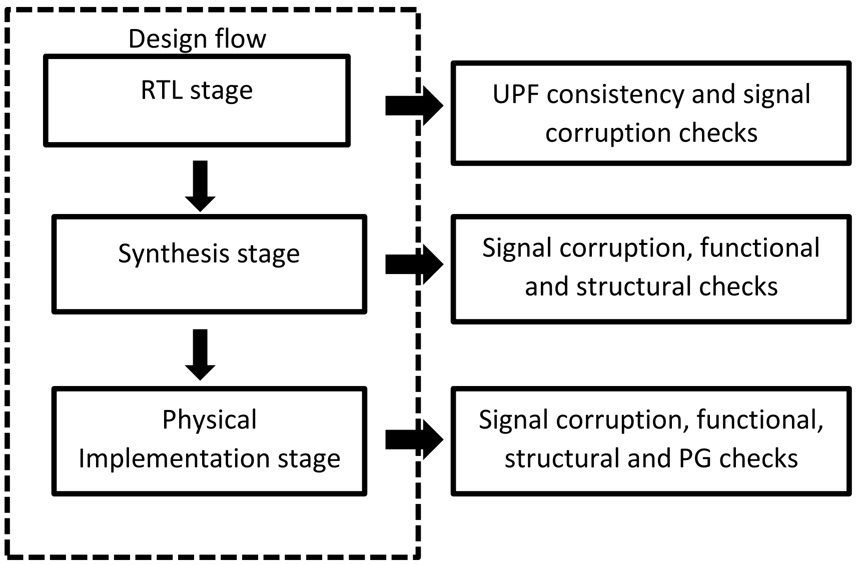

Design flow
RTL stage
Synthesis stage
Physical Implementation stage
UPF consistency and signal corruption checks
Signal corruption, functional and structural checks
Signal corruption, functional, structural and PG checks

Figure 2.5: Low power checks flow

Depending upon power intent characteristics and design stage, relevant checks are picked from the pool and power intent is verified. Let us now look into the more common and important error messages that we observe during static verification of UPF.

a) *UPF Supply unconnected:* This error occurs if supply net is created in the power intent using 'create_supply_net' but is not connected to any driver using 'connect_supply_net', i.e., the net is left hanging.

b) *UPF Supply No state:* The possible states of a supply net should be defined to identify different crossovers using 'add_power_state' command if it is a supply set. This error occurs if any supply net is defined in the power intent but its state is not mentioned.

c) *Unconnected Macro or pad cell power pins:* This error is flagged if supply net connections to power pins of macro blocks or pad cells are not mentioned explicitly. If it is the case, then those pins might be connected to the primary power supply of power domain in which these macroblocks are placed by implementation tool. This may lead to improper functioning of the design.

d) *Unconstrained port:* The voltage levels of signal pins of a block are usually associated with block's primary power and ground supplies. If these input and output ports should be associated with some other supplies, then it should be explicitly mentioned in UPF file using 'set_related_supply_net' or 'set_port_attribute' command. Otherwise, they might be wrongly associated by implementation tool and might result in removal or placement of level shifter cells. This error or warning is flagged if a top level port's associated supply is not mentioned.

e) *Missing Isolation or level shifter strategy:* The isolation and level shifter strategies are defined at RTL stage itself in UPF file. Based on the power states of nets connected to source and sink domains, if an isolation or level shifter strategy is to be defined between that source and sink but it is not, then no isolation or level shifter cells will be placed resulting in malfunctioning of the design. The missing of isolation and level shifter strategies are detected by the tool [22] and an error is flagged.

f) *Unconnected Level shifter power pins:* If level shifter power pins are not connected to any supply net using 'connect_supply_net' command, this error or warning occurs. In that case, these power pins are either left hanging or connected to default power supply of power domain in which these cells are present. This might result in malfunctioning of the cells.

These are few of the more common errors detected in UPF verification. There are many other errors related to supply ports, supply nets, isolation cells, level shifters, power state tables, retention cells [23].

Chapter 3

Related Work

This chapter briefly introduces the research done so far on power intent described using UPF and few methodologies which were proposed to address issues related to verification and maintenance of UPF through the design cycle.

A successive refinement methodology was introduced during IEEE 2009 UPF development. If a power intent is implementation dependent and later on if implementation specifications are changed, then entire power intent should be modified. If power intent file is defined at later stages in the design cycle only after finalization of implementation details, then the whole concept of verifying the effectiveness of applied power management strategies earlier goes useless. And moreover, if power intent is implementation specific, then power intent IPs which are used in different designs should be modified each time the technology gets changed. Hence, successive refinement methodology gives an idea of defining power intent in more abstract way with the help of new version of UPF at earlier stages and adding implementation details later to this abstract power intent. Here, power intent is incrementally specified at each stage.

The authors in [24] explained in detail the advantage of successive refinement methodology and how to use it effectively especially in an IP-based design. The categorized the power intent files at different stages as Constraint UPF which describes power intent specific to IP, Configuration UPF which has power details specific to the application and Implementation UPF which is technology specific Configuration UPF. Successive refinement methodology using these UPF files is explained using a processor based design example. The work gives an idea of refining power intent successively keeping the first defined power intent same but is more inclined towards power intent management of an IP in a design.

The practical challenges faced in the application of this Successive refinement methodology were discussed by authors in [25]. Few of the problems include handling explicit supply ports of hard macros, creating a hard boundary for soft macros for implementation which may require restructuring of UPF, representing the internal state of hard macros, isolating UPF created power control signals and redundancy in power state definitions. A refined successive refinement methodology is proposed addressing these issues and also to enable multi-vendor flow for successful handling of IP blocks.

The other famous and mostly used methodology is the one known as Synopsys low power flow using Synopsys toolchain [26]. In this methodology, UPF file describing power intent of design and RTL description of design logic are developed at RTL stage. At synthesis stage, Design Compiler, a synthesis tool, is fed with verified UPF and design description which gives out gate level netlist with special management cells added to the design logic and an updated UPF file, say UPF' with connections of the cells added during synthesis stage. Gate-level netlist and UPF' are given to IC compiler which performs physical implementation and gives out modified gate level netlist, all pins netlist and updated UPF', say UPF'' which automatically reflects any changes made in power intent of design along with connections of physical only cells added. Unlike in successive refinement methodology, here entire power intent is defined at an earlier stage and is modified according to changes made in design. This flow is very effective if only Synopsys tools are used in the design as few of the UPF concepts are not interoperable among different tools and UPF file should be restricted to or modified with tool support commands if the tool is changed [27]. The Synopsys low power flow was explained using a design example of processor with five power domains and by applying various power management techniques in [28].

A 16-bit Arithmetic Logic Unit (ALU) was implemented using UPF in [29]. The power reduction methods were applied during the implementation of ALU design. Power-aware simulation and verification were performed to check the effectiveness of management techniques applied and ensure that UPF could replicate the behaviour of power intent of a design.

Another concept named 'Golden UPF' is proposed by the authors in [30]. It addresses few practical problems in the implementation of incremental modifications of UPF methodology introduced by Synopsys. When UPF at RTL stage is modified at later stages, in updated UPF, user comments are lost, tool specific condition statements are not executed by tools used in further stages and object names are changed. To solve these problems, the authors suggest the usage of two UPF files at further stages, one is the UPF developed at RTL stage and a supplement UPF reflecting the changes made in power intent at a later stage. And the object path change problem is addressed with the use of mapping file. This partially resolves hierarchical path change problem due to restructuring (1.5).

A problem related to static power intent verification of power state switching expressions is addressed in [31]. The wrong refinement of logical power intent to physical power intent might lead to bugs which cannot be detected at earlier stages of the design cycle. The errors are caught by comparing power intents at logical and physical stages exclusively without involving design verification, thus removing inconsistencies. This work mainly focuses on finding inconsistencies related to power state switching expressions between power intents at various design stages.

A consistent power modelling from system level to implementation level was introduced in [32]. A design environment is developed which make use of metamodeling techniques to improve power closure by consistently observing power related values at various design levels. UPF is used for the description of power intent which contains power reduction methods applied to the design. Power consumption values of components obtained from simulation or estimation are stored in power metamodels. Making use of power intent model and power metamodels, the power consumption of the system is noted at each design level which helps in maintaining consistency in power related values from system level to silicon. But in this work, nothing related to UPF development challenges are mentioned as it is mainly concerned about chip power consumption.

A low power verification methodology for dynamic simulation was proposed in [33] to enable the use of customized assertions. The power intent of designs are getting complex day by day but the tool vendors provide fixed set of assertions for low power verification. There is an alarming need for user enabled assertions to create customized power aware models for dynamic simulation. This was achieved with the use of checker modules, find_objects and query commands. The effectiveness of methodology was also explained considering few case studies. This work is concerned about improving the effectiveness of dynamic verification of power intents.

Another low power dynamic verification methodology was introduced in [34] to effectively enable low power assertions and improve coverage. It also discusses the verification issues faced at various design stages of design cycle during low power verification. Few more methodologies and ideas to effectively carry out power aware simulations were introduced in [35], [36] and [37].

A methodology is proposed in Chapter 4, for UPF development and management through the design cycle with the help of insight into the work done so far in low power domain.

Chapter 4

Proposed Solution

In this Chapter, the solutions proposed to the challenges mentioned in 1.5 are discussed in detail. There is a total of four parts in proposed solution, which include alignment of input power data (4.1), restructuring and demotion of top level UPF (4.2), automation of UPF generation (4.3) and automation of static verification of UPF (4.4). Finally, a methodology is introduced (4.5), which include all the proposed features, for effective UPF management at SoC level.

4. 1 Alignment of power data

Three challenges, unreliable port declarations, mixing of power and signal pins and bus management are addressed here.

Before looking into the reasons leading to unreliable port declarations, let us have a brief overview of power related data. The main supplies to a chip come from package nets and are branched into various supplies due to current limitations of padring. These branched supplies, which can also be called as ring supplies, go into the chip's core side through pad cells. Along with ring supplies, central bump supplies also exist, if flip chip package is used. These are used to power up the central portions of chip effectively. The power pins of standard cells and hard blocks are connected to these supplies based on their requirements. But it is to be noted that analog routing is usually done separately taking care of its noise effects. To apply power reduction techniques, all these supplies are grouped based on their switching and voltage characteristics by power architect to ease the power sequencing.

One of the main reasons for unreliable port declarations is defining same supply port or supply net with different names resulting in shorts or opens after implementation, if not detected earlier. The possible cases of this situation along with other inconsistencies which might lead to the wrong implementation of power intent are discussed below.

Case i: The supplies branched from the main package supplies and the ones connected to the pad cells are not identical or few of them are missing. And if a ring supply and central supply are declared identical, then it leads to shorts. So, the following check shown in Figure 4.1 is done to clean main supplies and their corresponding supply nets along with pad cell connections.

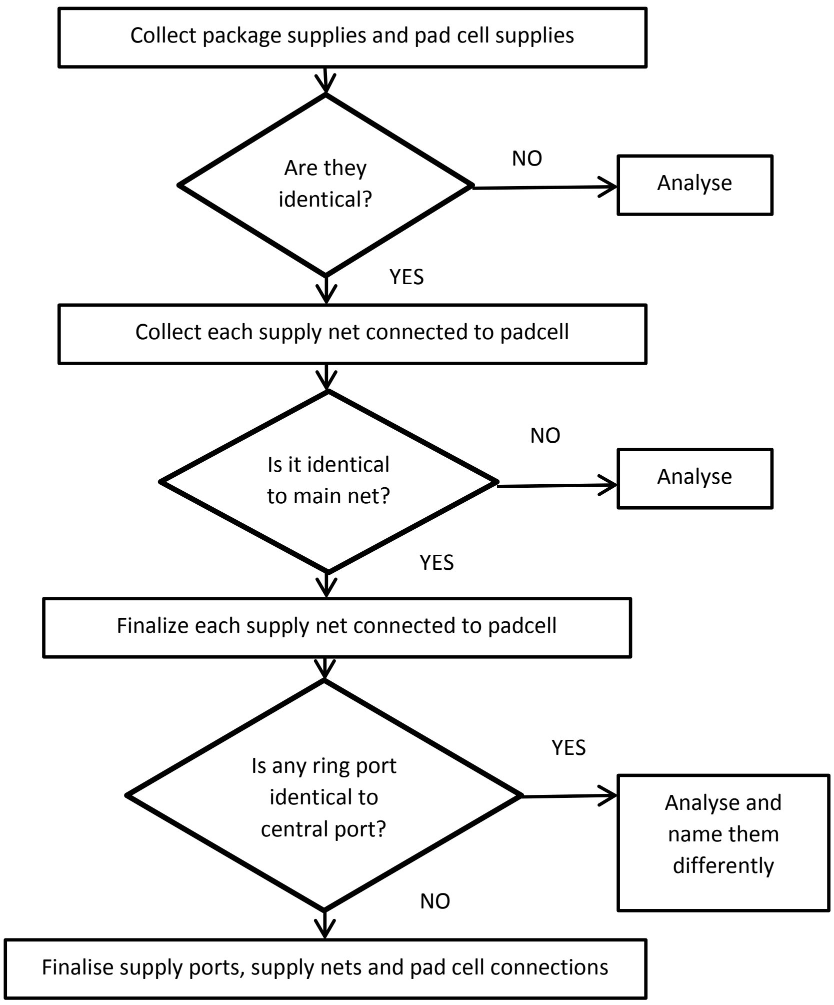
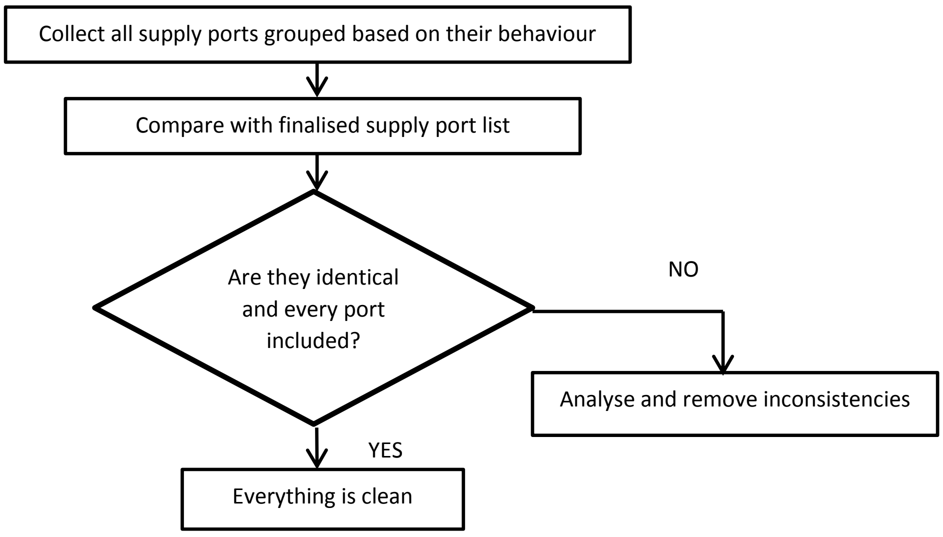

Collect package supplies and pad cell supplies
Are they identical?
NO
Analyse
YES
Collect each supply net connected to padcell
Is it identical to main net?
NO
Analyse
YES
Finalize each supply net connected to padcell
Is any ring port identical to central port?
YES
Analyse and name them differently
NO
Finalise supply ports, supply nets and pad cell connections

Figure 4.1: Algorithm to detect supply port, net and pad cell inconsistencies

Case ii: A supply port is missed while grouping the supplies, which is done to ease the application of power reduction methods and power sequencing. This results in an incomplete description of supply port's behaviour in UPF. The following check shown in Figure 4.2 is made to ensure that each and every port is taken care during power sequencing.

Case iii: As analog routing is done separately, there might arise a case where a supply net connected to an analog cell is not identical to the one connected to any supply port. This leads to hanging of nets connected to analog cells or creation of unnecessary supply ports with an assumption of the supply port connected to that supply net is missing. To avoid this, check if each supply net connected to analog cell is identical to a main supply net. If no, analyse the route cause and modify the connections accordingly.



Collect all supply ports grouped based on their behaviour
Compare with finalised supply port list
Are they identical and every port included?
NO
Analyse and remove inconsistencies
YES
Everything is clean

Figure 4.2: Flow chart to detect inconsistencies in supply groups

Case iv: The ports which are to be shorted are connected to different supply nets. This happens mostly at pad ring cuts. If a supply traversing through a padring is cut at both ends, it is represented differently at both the ends for ease of identification and declared as two different ports. But if the supplies at two ends need to be shorted, they should be connected to the same net and declared as a single supply. Otherwise, this results in the wrong implementation of pad ring behaviour. All such padring cuts are noted at the earlier stage and the corresponding ports are defined based on their characteristics.

Now coming to the bus management challenge (1.5), one may not have much idea about the inside functionality of hard macro blocks or IPs. So, there is a possibility of missing such buses as most of the times, an entire bus is mentioned without specifying its width explicitly giving an impression of a single signal pin. Hence any buses of such hard macro blocks are identified much before the generation of UPF based on input and output characteristics of the macro block. As the bit lines of buses are nothing but signal pins, it is taken care that related supply nets are defined for each bit line of each bus found.

The other challenge is a connection of signal pins to power supplies. Let us treat such signal pin as a supply pin and connect it to the supply net. But based on the characteristic of cell, the verification tool views it as signal pin and flags port unconstrained error (2.3). Hence, we collect all such signal pins in advance and declare them as signal pins in UPF defining related supply nets to those pins. These signal pins are grouped based on whether they should be connected to ground or supply voltage pin and then all the pins in a group are shorted in netlist itself and connected to relevant supply net.

It is to be noted that all the checks mentioned above along with bus management and signal pin analysis are performed on the data related to the development of entire power intent of SoC using UPF. So, UPF generated from the cleaned and aligned data will be mostly flawless and reduces verification time to a great extent.

4. 2 Restructuring and demotion of top level UPF

4.2.1 Restructuring of top level UPF

Restructuring of design results in hierarchical changes and the same should be replicated in UPF of that design as discussed in 1.5. To restructure UPF, we make use of mapping file [30] which gives the information about hierarchical path changes in the design and is obtained as the output of design restructured process. Based on this file, the pre-restructured elements in UPF are mapped to post restructured paths. But after restructuring the entire UPF, if pre-restructured UPF needs to be modified for some reason, then those changes should also be resembled in post-restructured UPF. So, instead of maintaining two UPF files, we propose to maintain a single file but use two different files for paths of elements, one before restructuring and the other after restructuring. This is done by declaring paths of elements used in UPF as variables whose values are assigned according to the pre and post-restructuring stages based on mapping file. This also simplifies repetitive verification of pre and post restructured UPFs without changing the hierarchies every time.

4.2.2 Demotion of top level UPF

Demotion of top level power intent involves defining UPF to each and every partition unit according to the restructuring of the design, scope of power domains and location of supply ports. The following steps guide in demoting top level UPF based on portioning of design.

1) The first step is to collect the nets which are connected to supply ports in one partition unit but supplying the cells in other units. Few possibilities of partitioning a design with four power domains, A, B, C and D, into two units, PU1 and PU2, are shown in Figure 4.3.

   In Figure 4.3(a), supply net Vnet is connected to the elements in both power domains A and B and is connected to supply port Va in power domain B. But as power domains A and B come into one partition unit PU1, it does not result in hanging of net and it is enough to declare supply port Va only once. Coming to Figure 4.3(b), now power domains A and B are partitioned a part resulting in net

Vnet unconnected in power domain A. So, supply port Va should be created in UPFs of both the partition units. Similar is the case with the partitions made as shown in Figure 4.3(c).

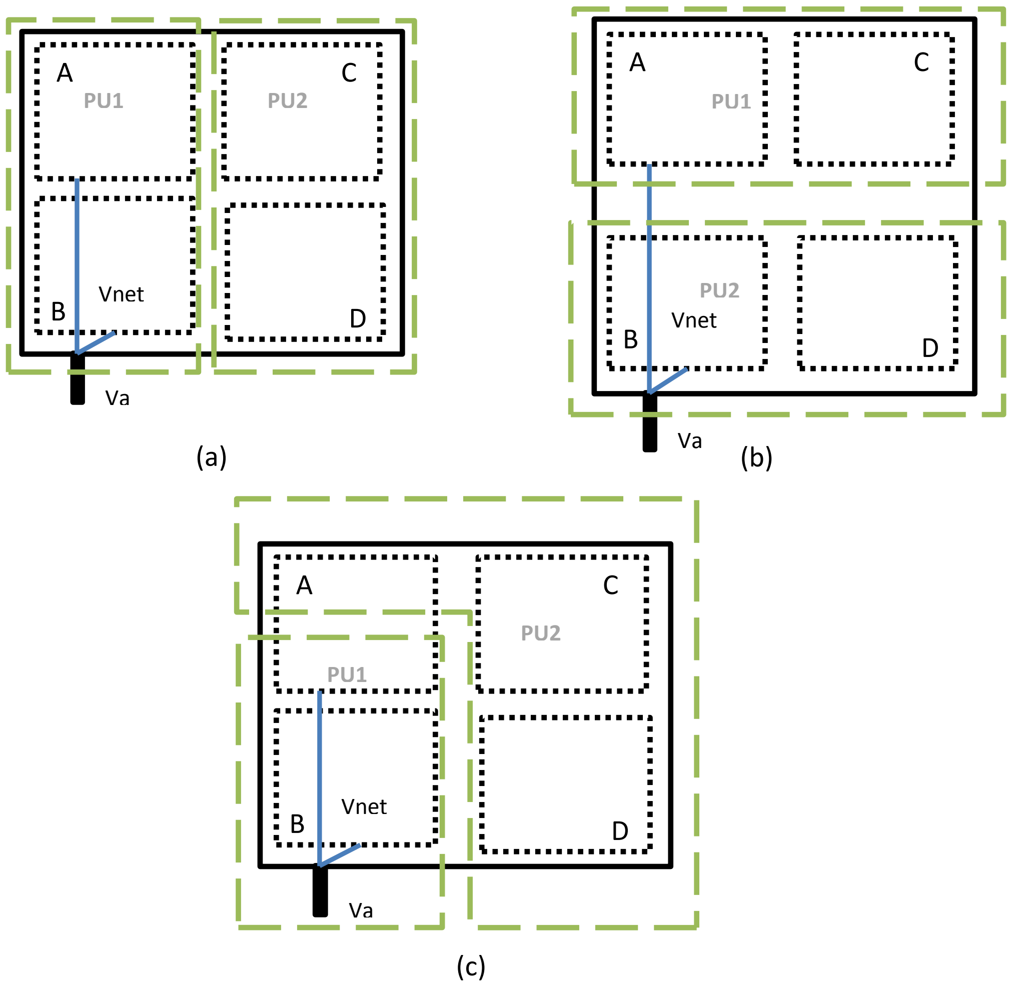


Figure 4.3: Possible partitioning of a design with four power domains

This type of nets can be collected based on hierarchical paths of elements to which those are connected, as different partition units have different path names. The following algorithm shown in Figure 4.4 is used to identify such nets and create ports along with defining port states accordingly.

2) The locations of elements are identified based on mapping file and their supply net connections are defined in respective partitioned UPFs.
3) The signal pins of cells are by default related to the primary supply of that cell. The signal pins which are related to some other supply are specially declared. Find such pins and define them in corresponding partitioned UPF to which they belong.

4) Power domains should also be declared in demoted power intents to represent the power characteristics of partition units. According to partition made and scope of power domains, these are created. For example, as per the partition shown in Figure 4.3(a), power intent of PU1 will have two power domains A and B. But power domain A is present in PU1 and PU2 according to partition in Figure 4.3(c). Hence elements under the scope of top level power domain A are divided as per the partitions and power domain A is created in both the units with respective element set assigned.

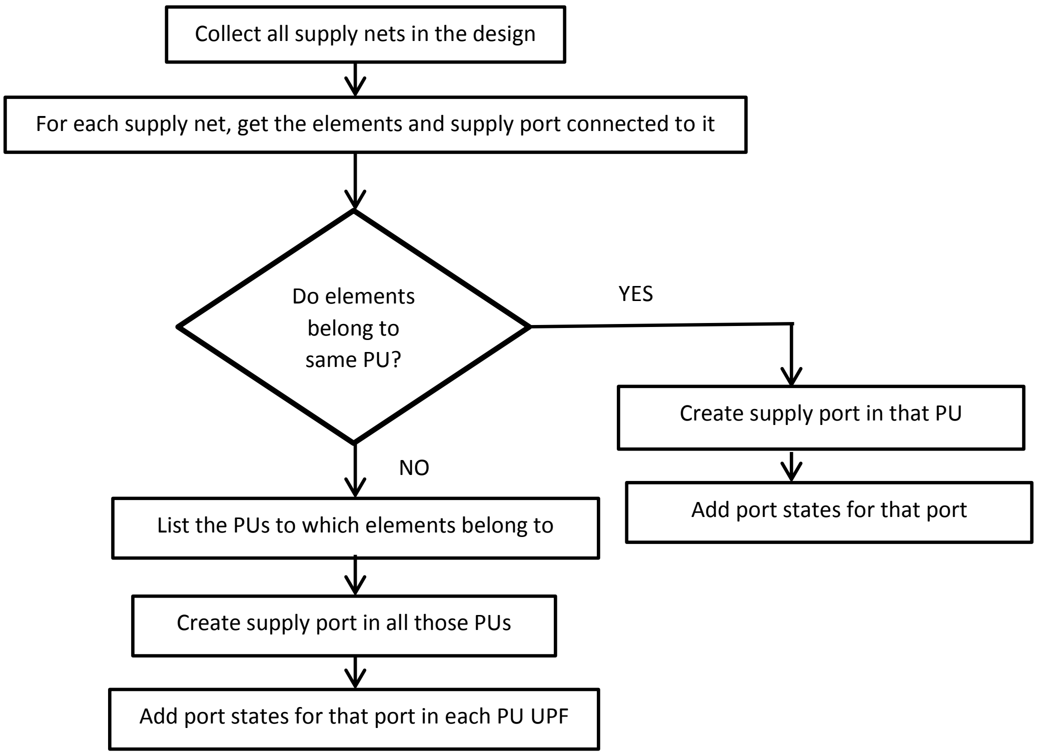

Collect all supply nets in the design
For each supply net, get the elements and supply port connected to it
Do elements belong to same PU?
YES
Create supply port in that PU
Add port states for that port
NO
List the PUs to which elements belong to
Create supply port in all those PUs
Add port states for that port in each PU UPF

Figure 4.4: Flowchart to create ports during demotion process

5) Management strategies like isolation, level shifter and retention are defined in respective demoted UPFs as per location of sources and sinks to which they are applied.
6) Finally, power state tables are created for each partitioned UPF based on the behaviour of supply ports in each partitioned unit.

The first three steps are automated which simplifies and quickens the demotion of top level power intent. But last three steps are currently performed manually.

4. 3 Automation of UPF generation

After the alignment of input power data (4.1), it is used for automatic generation of top level UPF using an algorithm. The generation setup takes in analog block routing data, power specifications of IPs, padring supplies and connections, central supplies, if flip chip package is used, and power sequencing and special strategies information from power architect as inputs. It converts all this data into UPF standard format, integrates and generates a top level UPF. It also takes in mapping file if restructuring is done and gives out lists of pre and post restructured path variable values. The algorithm follows the following process for generation of UPF.

1) All the ring and central supplies are collected and created using 'create_supply_port' command automatically.
2) The corresponding supply nets are collected, created and connected to these supply ports.
3) The pad cell power pin connections are extracted and connected to supply nets using 'connect_supply_net' command. The special digital block pin connections are also defined along with connections of analog blocks obtained from analog routing data.
4) Power supplies are associated to top signal ports using 'set_related_supply_net' command.
5) All ring and central supplies are grouped according to main package supplies and power sequencing data received from power architect and power states are defined using 'add_port_states' command.
6) The power specifications of IP blocks are extracted from their libraries, declared in UPF and promoted to top level.

The power domains, special cell strategies and power state tables are manually defined completing the representation of power intent in UPF. By automatically performing above mentioned steps, a lot of time is saved. It is to be noted that data extraction process needs to be modified if formats of input power data are changed, though the algorithm is generalized to any top level power intent generation.

4. 4 Automation of static verification of UPF

A verification setup is developed which simplifies the static verification of UPF and also saves time. The static checks are automatically selected based on the design stage. The inputs required for low power verification include UPF, functional netlist, library list and design stage information which is RTL stage or post-synthesis stage or post implementation stage. The setup reads in the design netlist and UPF, prepares checklist based on stage specification, verifies UPF and netlist based on checks and finally generates reports as output as shown in Figure 4.5. Reports include a summary of each check done and detailed error descriptions if any. Errors generated are resolved and verification process is repeated until UPF is error free.

As the complexity of chip increases, additional checks need to be performed which might not be supported by verification tool. The proposed setup is flexible enough to add additional custom checks at any desired stage and enable or disable them as per requirements.

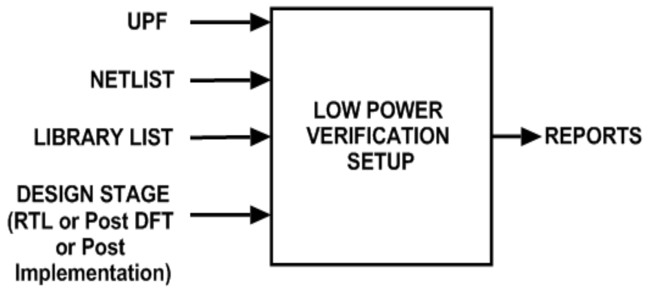

UPF
NETLIST
LIBRARY LIST
DESIGN STAGE (RTL or Post DFT or Post Implementation)
LOW POWER VERIFICATION SETUP
REPORTS

Figure 4.5: Low power verification setup

The setup also allows us to change the error severity of checks very easily and checks can be enabled or disabled based on the design under consideration. The reports to be generated can be customized as desired.

4. 5 Proposed UPF methodology

By incorporating above proposed solutions for power intent development and verification, an ideal methodology, starting from RTL stage to post-implementation stage, is introduced as shown in Figure 4.6 for effectively and efficiently handling power intent through the design cycle.

The power related data from various sources is aligned and inconsistencies, if any, are removed. As this is done at RTL stage itself, the possibilities of detecting bugs in power intent late in design cycle are avoided leading to the generation of ideal UPF at an earlier stage. Automatic development of UPF from this cleaned data and its verification saves time. Restructuring and demotion algorithm ensures that top level power intent characteristics are replicated at block level based on the partitioning of design. Automation of few portions of demotion process quickens the generation of partition level UPFs.

The basic concept of Synopsys low power flow [26] is used here for incremental modifications of UPF at synthesis and implementation stages, so that connections related to new cells introduced at these stages are automatically updated in UPF file. And static verification at each stage is performed using the proposed verification setup which greatly reduces time. But the proposed methodology is most effective for top-down approach of the design cycle.

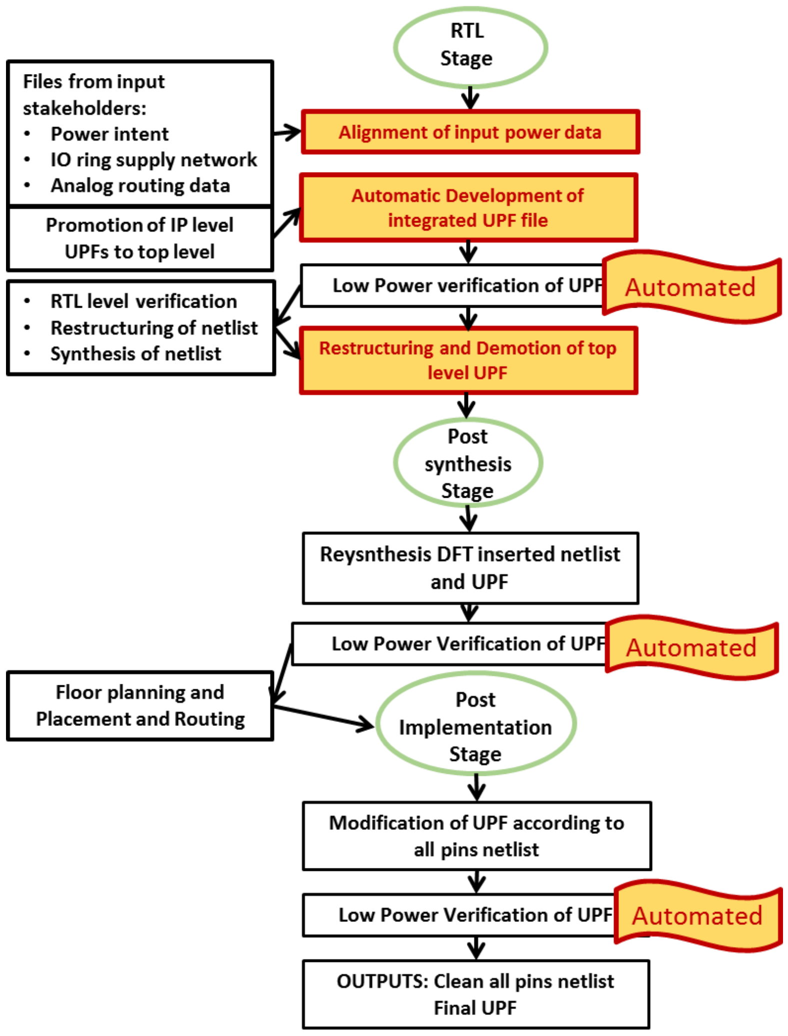

RTL Stage
Files from input stakeholders:
• Power intent
• IO ring supply network
• Analog routing data
Alignment of input power data
Promotion of IP level UPFs to top level
Automatic Development of integrated UPF file
Low Power verification of UPF
Automated
• RTL level verification
• Restructuring of netlist
• Synthesis of netlist
Restructuring and Demotion of top level UPF
Post synthesis Stage
Reysnthesis DFT inserted netlist and UPF
Low Power Verification of UPF
Automated
Floor planning and Placement and Routing
Post Implementation Stage
Modification of UPF according to all pins netlist
Low Power Verification of UPF
Automated
OUTPUTS: Clean all pins netlist Final UPF

Figure 4.6: Proposed UPF methodology

Chapter 5

Results

The proposed UPF methodology is applied on the development of power-aware set top box chip with 10.36 million cell count. It is partitioned into four units, out of which three are purely switchable and the other unit has both switchable and always ON blocks. The power management strategies applied are power gating and multi-supply voltage technique. So, both level shifter and isolation strategies are defined. The power intent of this chip has seven different modes of operation and contains two power domains.

The analog routing data, power intent specifications, package main supplies and pad ring information were obtained from various sources. All the data was cleaned using the algorithms proposed (4.1) and top level UPF was generated automatically. Static verification of this UPF was performed using Synopsys' VC LP tool. The verification results at RTL stage are shown in Table 5.1 and were compared with those of conventional UPF flow as RTL stage is where the main UPF is developed and at later stages, it is only incremented. It is noted that only those errors were considered which are mainly and commonly generated due to the inconsistent representation of UPF.

	Conventional methodology	**Proposed methodology**
Unconnected Macro pins	2673	1008
UPF supply nostate	14	2
Unconnected pad power pins	13	0
Unconstrained ports	54	0
UPF supply unconnected	7	0
Total	2761	1010

Table 5.1 Analysis of RTL stage static errors in conventional and proposed flows

The errors like unconnected pad cell power pins, unconnected macro power pins, UPF supply nostate and issues related to supply net and ports are greatly reduced in proposed methodology due to the removal of inconsistencies by aligning input power data. Port unconstrained and supply unconnected errors are eliminated due to effective handling of buses and supply connected signal pins in the system.

The graph in Figure 5.1 compares the errors generated by applying conventional and proposed methodologies, at three main stages of the design cycle, RTL, post synthesis and post-implementation stages. The errors at synthesis stage are decreased in proposed flow due to the use of incremental UPF concept due to which connections of special management cells are automatically updated in UPF file during synthesis stage. Same is the case with post implementation verification. And as an ideal UPF is developed at RTL stage itself, UPF updated at later stages is less error prone. Error reductions of 64%, 99% and 98% are noticed at RTL, post synthesis and post-implementation stages giving an overall error decrease of 81%.

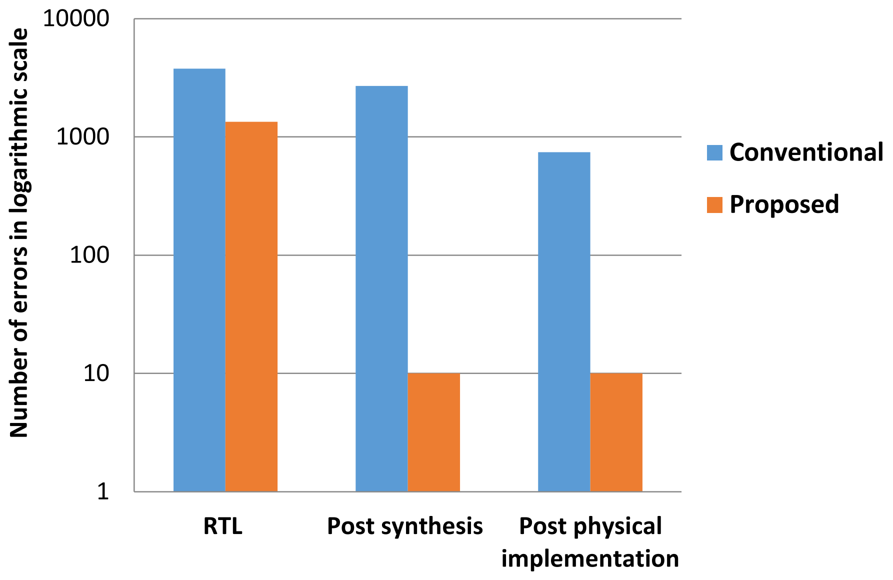


Figure 5.1 Verification errors at various stages in conventional and proposed methodologies

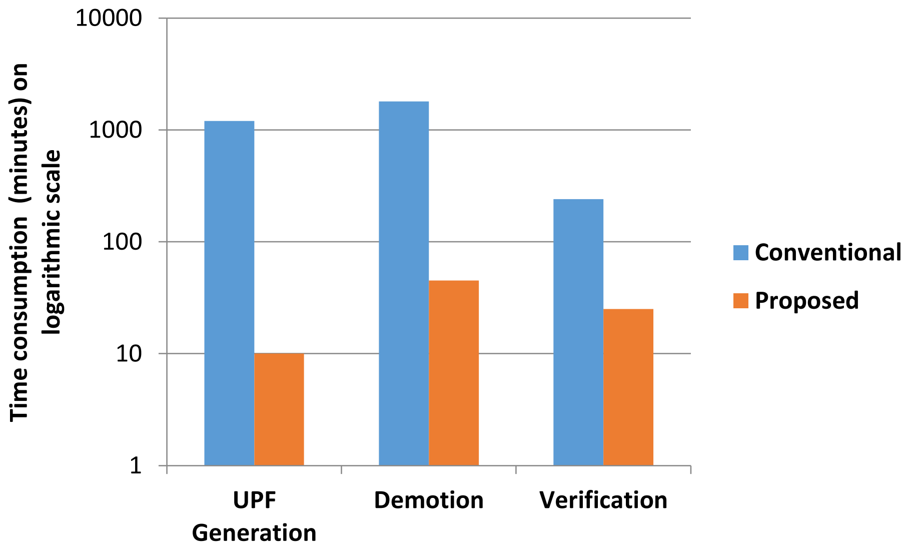


Figure 5.2 Time taken for generation, demotion and verification of UPF in conventional and proposed methodologies

The time gain due to automation of few steps in UPF methodology is also analysed. The time taken for development, verification and demotion of UPF are calculated and compared with a manual approach. The results shown in Figure 5.2 prove that time reductions of 99%, 97% and 89% are achieved with automation of development, demotion and verification processes respectively. It is to be noted that the verification time gain mentioned is for one stage and for a single iteration. And the time taken for writing UPF manually is excluded from both the flows for calculating UPF generation time.

Time is reduced due to automation of verification process and also due to the reduction of verification iterations to be done as most of the errors are cleaned prior to the development of UPF. Table 5.2 presents a number of verification iterations taken at each stage for conventional and proposed methodology and overall verification time is compared. It is clear that proposed flow results in lesser iterations and an overall time reduction of 93% is observed.

	Type of stage	**Conventional**	**Proposed**
Number of iterations	RTL	5	2
	Post Synthesis	2	1
	Post Implementation	2	1
Time taken for overall verification (in minutes)	RTL	510	85
	Post-Synthesis	360	20
	Post Implementation	360	20
Overall verification time at all stages (in minutes)		1830	125

Table 5.2 Analysis of overall verification time at various stages

In summary, an overall error reduction of 81% is observed due to early detection of inconsistencies in power data and removal of bugs. Time gains of 99% and 97% are achieved by automating UPF generation and demotion processes respectively. An overall verification time reduction of 93% is noticed due to automation and early bug detections leading to decrease in a number of iterations.

Chapter 6

Conclusion and Future Scope

6.1 Conclusion

The work proposed a methodology for developing and managing power intent effectively and efficiently at SoC level by evaluating various drawbacks of conventional UPF methodology. The inconsistencies in power related data which might arise due to the parallel development of power intent were addressed by introducing an alignment process before developing top level UPF. The possible cases leading to faulty power intent were analysed and an algorithm was proposed which helped in detecting the issues at earlier stages. The problems related to restructuring and demotion of top level UPF to partition levels were solved by introducing an algorithm which effectively represents top level power constraints at the block level and a part of the process in automated. Time taking UPF generation and verification steps were replaced with automated versions. The results obtained by applying the proposed flow to a complex set top box chip prove that the proposed methodology enables the development of golden UPF at earlier stages and quickens and simplifies UPF flow through design cycle.

6.2 Future Scope

IEEE 1801 UPF standard is being refined very frequently to effectively represent complex power architectures of designs. These new versions are to be adapted to make use of the advanced features. UPF commands are to be categorised based on tool support to enable effective interoperability of power intent among various tools. The impact of more complex power intents on UPF flow should be analysed. This can be performed by increasing the number of power domains and reduction techniques.

As power domains, power state tables are manually written in UPF right now, their automation helps in reduction of time to a great extent. The UPF methodology for bottom-up approach is to be analysed and optimized as not all design flows are top down.

References

[1] Moore, Gordon "Moore's Law", Intel company, 1965, 11 July 2016. Available: https://en.wikipedia.org/wiki/Moore%27s_law

[2] R. K. Krishnamurthy, Atila Alvandpour, Vivek De, Shekhar Borkar, "High-performance and low-power challenges for sub-70 nm microprocessor circuits" *CICC, 2002*.IEEE Proceedings, pp. 125–128, 2002

[3] Neil Weste and David Harris, "CMOS VLSI Design: A Circuits and Systems Perspective", Addison-Wesley Publishing Company, 4th edition, 2005.

[4] Chadha, Rakesh, J. Bhasker, "An ASIC Low power prime analyser", Springer Company, 1st edition, 2013

[5] Nam Sung Kim, Todd Austin, David Blaauw, Trevor Mudge, Krisztian Flautner, Jie S.Hu, Mary Jane Irwin, Mahmut Kandemir, Vijaykrishnan Narayanan, "Leakage current: Moore's Law Meets Static Power", *IEEE computer society*, 2003.

[6] Chi-Ping Hsu, "Pushing Power Forward with a Common Power Format - The Process of Getting it Right", EETimes, 5 Nov 2006.

[7] "A Practical Guide to Low-Power Design: User Experience with CPF", Cadence, March 2008.

[8] Richard Goering (September 18, 2006), "Accellera rolls power plan", EE Times, Retrieved July 7, 2011.

[9] Qi Wang, "The Evolution of Power Format Standards: A Cadence Viewpoint", Cadence Design, Systems, Inc, 2012.

[10] Erich Marschner, "Evolution of UPF: Getting Better All the Time", October, 2012. Available: https://verificationacademy.com/verification-horizons/october-2012-volume-8-issue-3/evolution-of-upf-getting-better-all-time

[11] Erich Marschner, "Overview of UPF", Mentor Graphics. Available: https://verificationacademy.com/sessions/overview-of-upf

[12] "IEEE P1801 - Unified Power Format Standard", *Accelera Organization Inc*, February 22, 2007.

[13] "Flip chip c4b", Mantra VLSI, 2015. Available: http://www.slideshare.net/DeepakFloria/flip-chip-technology

[14] "IEEE Standard for Design and Verification of Low Power Integrated Circuits", *IEEE Computer Society*, 19th March 2009.

[15] E.Sicard, S.Delmas-Bendhia, "Deep-submicron CMOS circuit design Simulator in hands", Brooks/Cole Publishing Company, December, 2003.

[16] Chuck Seeley, "Using Supply Sets", Mentor Graphics. Available: https://verificationacademy.com/sessions/using-supply-sets

[17] "Tutorial: SoC Power Management Verification and Testing Issues", *Ninth International Workshop on Microprocessor Test and Verification*, 2008, IEEE, pp. 67-72, 8-10 Dec. 2008.

[18] Erich Marschner, "Getting Started with UPF", *Mentor Graphics*. Available: https://verificationacademy.com/sessions/getting-started-with-upf

[19] "IEEE Standard for Design and Verification of Low-Power Integrated Circuits", *IEEE Computer Society*, 29th May 2013.

[20] Freddy Bembaron, Sachin Kakkar, Rudra Mukherjee, Amit Srivastava, "Low Power Verification Methodology Using UPF", *DVCon 2011*, Feb 28-Mar 3, 2011.

[21] "VC Low Power User Guide", Synopsys, June 2016.

[22] Rudra Mukherjee, Amit Srivastava, Stephen Bailey, "Static and Formal Verification of Power Aware Designs at the RTL Using UPF", Mentor Graphics Corporation, 2009.

[23] "VC LP Message Reference Guide", Synopsys, June 2016.

[24] Adnan Khan, John Biggs, Eamonn Quigley, Erich Marschner, "Successive Refinement: A Methodology for Incremental Specification of Power Intent", *DVCON 2015*, March 2015.

[25] Desinghu PS, Adnan Khan, Erich Marschner, Gabriel Chidolue, "Refining Successive Refinement: Improving a Methodology for Incremental Specification of Power Intent", *DVCON* 2015, March 2015.

[26] "Synopsys® Low-Power Flow User Guide",Synopsys, December 2011.

[27] Emilie Garat, David Coriat, Edith Beigne, Leandro Stefanazzi, "Unified Power Format (UPF) methodology in a vendor independent flow", *PATMOS,2015, 25TH International Workshop, IEEE*, pp.82-88, 1-4 September, 2015.

[28] Venkatesh Gourisetty, Hamid Mahmoodi, Vazgen Melikyan, Eduard Babayan, Rich Goldman, Katie, "Low Power Design Flow Based on Unified Power Format and Synopsys Tool Chain", *3rd Interdisciplinary Engineering Design Education Conference, IEEE*, pp.28-31,4-5 March, 2013.

[29] Roopa R. Kulkarni, S. Y. Kulkarni, "Energy efficient implementation, power aware simulation and verification of 16-bit ALU using unified power format standards", Advanced Electroncis, *ICAECC, IEEE*, pp. 1-6, 10-11 Oct. 2014.

[30] Himanshu Bhatt, Harsh Chilwal, "Golden UPF: Preserving Power Intent from RTL to Implementation", *DVCon*, March 2015.

[31] Srobona Mitra, Bhaskar Pal, Soumen Ghosh, Rajarshi Mukherjee, Kaushik De, “Static Power Intent Verification of Power State Switching Expressions”, *DVCON*, March 2015.

[32] Juergen Karmann, Wolfgang Ecke, “The Semantic of the Power Intent Format UPF: Consistent Power Modeling from System Level to Implementation”, *PATMOS*, pp. 45-50, 9-11 Sept. 2013.

[33] Madhur Bhargava, Durgesh Prasad, “Low-Power Verification Methodology using UPF Query functions and Bind checkers”, *DVCON* 2014, March 3-5 2014.

[34] Abhinav Nawal, Gaurav Jain, Joachim Geishauser, “Complex Low Power Verification Challenges in NextGen SoCs : Taming the Beast!”, *DVCON*, March 3-5, 2014

[35] Reza Sharafinejad, Bijan Alizadeh, Masahiro Fujita,“UPF-based Formal Verification of Low Power Techniques in Modern Processors”, *33rd VLSI Test Symposium, IEEE,* pp.1-6, 27-29 April, 2015.

[36] Mohit Jain, Amit Singh, J.S.S.S.Bharath, Amit Srivastava, Bharti Jain, “Power Aware Models: Overcoming barriers in Power Aware Simulation”, *DVCon*, October, 2014.

[37] Deepmala Sachan, Thameem Syed S, Raghavendra Prakash, Venugopal Jennarapu, “Low power Verification challenges and coverage recipe to sign-off Power aware Verification”, *DVCon,*25-26 September, 2014.

Publications

The work presented in this thesis is appreciated at following conferences:

[1] Renduchinthala Anusha et.al, “**An approach for early detection of power issues and automation of UPF flow**”, Design/IP Track, DAC 2016, Austin, Texas

[2] Renduchinthala Anusha et.al, “**An efficient approach to smoothen UPF management at SoC level**”, DVCon India, 2016