2024 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION EUROPE
MUNICH, GERMANY
OCTOBER 15-16, 2024

Auto Generation is the key to rapid integration to UPF-like UPVM libraries for Unified Power Verification

Gopi Srinivas Deepala, Engineer - VLSI Design, Silicon Interfaces, Mumbai, India (gopi@siliconinterfaces.com)

Lakshya Miglani, Engineer - VLSI Design, Silicon Interfaces, Mumbai, India (lakshya@siliconinterfaces.com)

Himanshu Vishwakarma, Engineer - VLSI Design, Silicon Interfaces, Mumbai, India (himvish@siliconinterfaces.com)

Priyanka Gharat, India (gharatpriyanka0@gmail.com)

***Abstract*- This paper illustrates the powerful automation resource, a python based tool, smoothly combines UPF power strategies with UPVM class libraries. UPVM, the evolving open source standard for Unified Power Verification Methodology™, serves as a bridge between UVM and UPF. This automation offers a fresh perspective on generating UPF constructs, facilitating a seamless journey from the RTL to GLS and GDSII stages for Design under Test (DUTs) with a focus on low-power features. This versatile utility simplifies intricate tasks such as managing power domains, organizing hierarchical sets, creating supply ports, connecting them, handling state transitions, and implementing comprehensive power strategies to bridging the gap between UPF and UVM methodologies, overcoming challenges related to power management during verification. Additionally, it optimizes the translation of power goals from RTL to GLS/GDSII, ensuring strict compliance with low-power requirements. Through its holistic and integrated approach, this tactic aims to revolutionize low-power semiconductor design practices, leading to improved power efficiency, reduced errors, and smoother transitions throughout the various stages of design development.**

Keywords—Unified Power Verification Methodology (UPVM), Unified Power Format (UPF), Universal Verification Methodology (UVM), Automation, Power Management, Power Verification.

I. INTRODUCTION

Recent advancements in semiconductor design have seen the active integration of power verification methodologies. Notably, the combination of Unified Power Format (UPF) and Universal Verification Methodology (UVM) techniques has been explored to address power management challenges during the verification phase as shown in Figure I and Ref 3 and Ref 4 as given in the Reference Section.

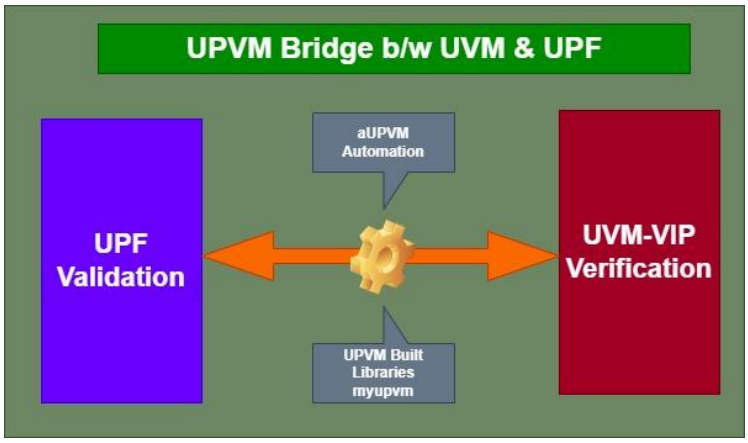

Figure I: Unified Power Verification Methodology (UPVM) – Bridge

A significant innovation in this area is the development of an automation named aUPVM.py. aUPVM is a Python-based tool, which makes it simple for the Designer to integrate Low Power methodologies and strategies to Design Under Test (DUT) at an early stage and integrate with SystemVerilog Verification using previously developed myUPVM templates and UPVM libraries, like upf_version, set_design_top, set_scope, set_directory, create_power_domain, create_ports, Etc. (UPVM, the evolving open source standard libraries for Unified Power Verification Methodology™, serves as a bridge between UPVM and UPF, previously developed).

The automation's capabilities are further enhanced by integrating the low-power strategies developed for myUPVM templates which are extensions of the UPVM libraries. By leveraging SystemVerilog classes and packages, the automation implements UPF-likeSystemVerilog files based on low-power strategies defined in lpdut files. This ensures the effective translation of low-power strategies into the design, aligning with industry standards and regulations.

Following the generation of the low-power UPF-like file and integrating the same to myUPVM, power validation is conducted during the UPVM verification phases. In the validation phase (part of UPVM libraries), the generated power strategies shall be validated with mapped library components for violations and conflicts. This enables comprehensive power verification, ensuring the accuracy and reliability of semiconductor designs.

II. AUTOMATION FOR STREAMLINING THE UPF STRATEGIES

This aUPVM represents a major advancement in automating the creation of UPF-like strategies for semiconductor design. Its primary goal is to streamline the generation of intermediate file formats which can be integrated into myUPVM template files by providing developers with an intuitive interface.

In the beginning, the automation begins by prompting developers to initialize the file with the correct UPF version being used (for this paper version 2.0) as shown in Figure II.

Figure II: aUPVM automation

The tool next facilitates directory selection, by navigating through the directories and selecting the specific directory which contains the design files. Use the provided commands to move up (goto), and do (goback) across directories as shown in Figure III.

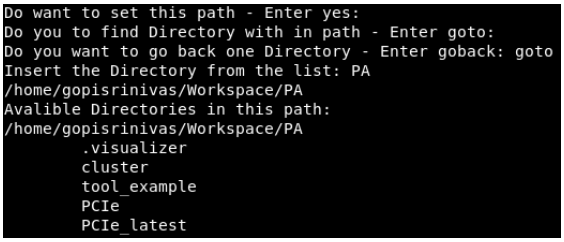

Figure III: Directory Selection Procedure

It integrates with the design and then it displays the design hierarchy, which will reduce the complexity to identify the Design hierarchy which makes it easy to select the top module to developers as shown in Figure IV.

A standout feature of automation is its ability to create power domains based on the design hierarchy. When creating power domains, the aUPVM automation prompts designers to specify which elements within the hierarchy require power domains. It then presents the module hierarchy for each module to assist in the selection process as shown in Figure V.

```
/home/gopisrinivas/Workspace/PA/PCIe_latest
Is This Directory Contains the Your Design Files (yes/no): yes
System Verilog Files Found in this Directory
pcie_tb
    pcie_top
        pcie_receiver
            pcie_tlp
            pcie_phy
                pcie_byte_unstrip
                pcie_phy_cdr
                    pcie_block_align
                    pcie_pll
                    pcie_elastic_buffer
                    pcie_s2p
                pcie_decoder_10x8
                pcie_packet_filtering
            pcie_dll
        pcie_transmitter
            pcie_phy_tx
                pcie_phy_packet_framing
                pcie_phy_byte_striping
                pcie_phy_p2s
            pcie_tlp_tx
            pcie_dll_tx
    pcie_s2p
```

Figure IV: Design Hierarchy

```
Available modules:
pcie_rx
/pcie_rx/tsl
/pcie_rx/phy_rx
/pcie_rx/phy_rx/us
/pcie_rx/phy_rx/cdr
/pcie_rx/phy_rx/cdr/ba
/pcie_rx/phy_rx/cdr/pll
/pcie_rx/phy_rx/cdr/eb
/pcie_rx/phy_rx/cdr/s2p
/pcie_rx/phy_rx/d0
/pcie_rx/phy_rx/pf
/pcie_rx/dll
pcie_tx
/pcie_tx/phy_tx
/pcie_tx/phy_tx/pf_tx
/pcie_tx/phy_tx/strip
/pcie_tx/phy_tx/p2s
/pcie_tx/tlp
/pcie_tx/dlp
```

Figure V: Module Hierarchy for selecting the Power Domain elements

By integrating with existing UPF files, the automation aligns power domain elements with the current design structure, ensuring compatibility before adding the legacy UPF file. This enhances the reuse, accuracy, and efficiency of power domain creation, crucial for effective power management in semiconductor designs.

One of the automation's key features is its interactive approach, which collects designer requirements for specific UPF strategies. Through a series of prompts and options, the automation skillfully guides designers through the necessary steps to develop UPF-like strategies that align with their design goals.

During the selection of the existing UPF file once you have reached the desired directory, confirm your selection to begin the existing UPF File from the listed as shown in Figure VI.

When incorporating a UPF-like intermediate file, to facilitate hierarchical incorporation of legacy designs, the automation process will first read the power domains defined in the UPF file. It will then compare these domains with the current design hierarchy. The UPF file will only be added if there is a match between the power domains in the UPF file and the elements in the design hierarchy as shown in Figure VII. If a mismatch is detected, an error will be generated, indicating that the UPF file addition has failed. Therefore, it is crucial to ensure that the power domains specified in the UPF file align with the elements in the current design hierarchy before proceeding as shown in Figure VIII.

```
Do want to set this path - Enter yes:
Do you to find Directory with in path - Enter goto:
Do you want to go back one Directory - Enter goback: yes
/home/gopisrinivas/Workspace/PA/PCIe

        pcie_upf1.upf

        pcie.upf

        pcie_latest2.upf

        pcie_latest1.upf

        pcie_latest.upf

        pcie_final.upf

        pcie_latest3.upf

No more UPF files are Available
```

Figure VI: Listing the UPF Files within the Path

```
Enter the file Name from the list: pcie_latest1.upf
Available Power Domain in this file
System Verilog Files Found in this Directory

Match Found: The Power Domain Elements Present in this Design
Element in UPF File: pcie_tx
Design Hierarchy: pcie_tx


System Verilog Files Found in this Directory

Match Found: The Power Domain Elements Present in this Design
Element in UPF File: pcie_rx
Design Hierarchy: pcie_rx


System Verilog Files Found in this Directory
System Verilog Files Found in this Directory
System Verilog Files Found in this Directory
The Load UPF File Successful
```

Figure VII: Match Pass Elements Comparison

```
Enter the file Name from the list: rtl_top.upf
Available Power Domain in this file

WARNING: The Power Domain Elements are in not Matched with the Current Design:

Loading UPF File is Failed
```

Figure VIII: Match Fail Elements Comparison

Additionally, the automation facilitates the creation of ports, nets, and supply sets, along with their associated supply sets and logic ports. It establishes connections between logic nets and generates power switches and states. These features contribute to the comprehensive implementation of low-power strategies in UPF files for energy efficiency and reducing power consumption in semiconductor designs.

The automation provides designers with a streamlined approach to specifying their required strategies for UPF-like intermediate file creation. Designers are presented with options, including tasks such as creating nets, logic nets, and ports, as well as establishing connections between these elements. Additionally, designers can define supply sets and associated supply sets, configure power switches and states, and incorporate state and logic expressions into their UPF-like intermediate files. The automation also supports the integration of level shifters, isolation cells, and retention cells into the power management scheme. This allows designers to maintain control over these strategies, tailoring the UPF-like intermediate file generation process to meet their specific design requirements. Once designers have made their selections, the automation consolidates all the output into a lpdut file. This lpdut file serves as the foundation for integrating with myUPVM templates (which are leveraging UVPM libraries as shown in Figure XI)and then generating the UPF file to ensure the effective implementation of the specified power management strategies as shown in Figure IX.

```
[upf_version]:2.0
[top]:pcie_top
[scope]:pcie_top
[pd]:PD_PD_PCIE_TOP;
[pd]:PD_PD_PCIE_PHY_CDR;[elements]:pcie_rx/phy_rx/cdr;
[pd]:PD_PD_PCIE_RX;[elements]:pcie_rx;[supply]:primary;[supply]:backup;
[pd]:PD_PD_PCIE_TX;[elements]:pcie_tx;[supply]:primary;[supply]:backup;
[set_dir]:UPF_DIR /home/gopisrinivas/Workspace/PA/PCIe_latest

[upf_file]:/home/gopisrinivas/Workspace/PA/PCIe/pcie_latest1.upf
[upf_file]:/home/gopisrinivas/Workspace/PA/PCIe/pcie.upf
[upf_file]:/home/gopisrinivas/Workspace/PA/PCIe/pcie_upf1.upf
[upf_file]:/home/gopisrinivas/Workspace/PA/PCIe/pcie_final.upf
[ports]:VDD1;VSS1,VDD2;VSS2
[nets]:RX_Pwr;[domain]:PD_PD_PCIE_RX
[nets]:RX_Gnd;[domain]:PD_PD_PCIE_RX
[nets]:CDR_Pwr;[domain]:PD_PD_PCIE_PHY_CDR
[nets]:CDR_Gnd;[domain]:PD_PD_PCIE_PHY_CDR
[nets]:TX_Pwr;[domain]:PD_PD_PCIE_TX
[connect_net]:RX_Pwr;VDD1,RX_Gnd;VSS1,CDR_Pwr;VDD2,CDR_Gnd;VSS2
[create_supply_sets]:CDR_SS;[function]:power,CDR_Pwr;[function]:ground,CDR_Gnd
[associate_supply_set]:CDR_SS;[handle]:PD_PD_PCIE_PHY_CDR.primary
```

Figure IX: lpdut File outcome of Automation

III. Utilizing The UPVM Libraries to Integrate and Generate UPF Files

After generating the lpdut file using the Python automation, utilizing the UPVM libraries to extract the lpdut file into the UPF File, and integrating with Low Power Strategies for the Power Validation at the earlier stage of the UVM verification, these libraries developed using SystemVerilog packages as shown in Figure X, are specifically designed to provide the utilities and functionalities necessary to integrate with the power strategies for the Low Power Validation. The myupvm.sv file integrates these built-in UPVM libraries, facilitating the seamless construction of UPF files as shown in Figure XI.

Designers can be permitted to execute the myupvm.sv file uses the UPVM libraries to interpret the lpdut file and generate the corresponding UPF file. This process ensures the efficient translation of the low-power strategies outlined in the lpdut file into UPF files, aligning with industry standards and regulations.

Moreover, the UPVM libraries are crucial in accessing UPVM strategies within the UVM-like testbench and Scoreboard. These libraries are integrated into the UVM test-bench codebase, enabling access to UPVM strategies during UPF file creation and Verification. They are also incorporated into the Scoreboard to facilitate validation of methodologies and strategies. This comprehensive validation ensures the ability of the power management strategies to validate Low Power design within the UVM environment.

```
package upvm_pkg;

class upf_version;
  int fd;
  string line;
  string st;
  string substring;
  int fd_w,fd_a;
  string version[$];

  function string version_fun(string filename,string array[$]);
       //Passing the value for upf version if not passed in lpdut file
       return array.pop_back();
  endfunction
endclass

class set_design_top;
  string design_top;
  int fd;
  string line;
  string tp;
  string top_module;
  int fd_a;
```

Figure X: UPVM Libraries Package

```
`include "upvm.sv"
module myUPVM;
import upvm_pkg::*;

  class my_upf_version extends upf_version;
        string array[$];
        function new(string filename,string array[$]);
        //Reading line for set scope
        fd = $fopen(filename,"r");
        fd_w=$fopen("./pcie.upf","w");
        fd_a=$fopen("./pcie.upf","a");

        do begin
                $fgets(line, fd);
                if(line.substr(0,13)=="[upf_version]:")
                begin
                   for(int i = 14; i <=(line.len());i++)
                   begin
                        if(line.getc(i) == " ")
                        begin
                                substring=st;
                        end
                        else
                        begin
                                st={st,line.getc(i)};
                        end
```

Figure XI: myUPVM Template (myupvm.sv)

IV. APPLICATIONS

The primary function of the aUPVM tool is to facilitate UPVM generation, offering several valuable applications to the semiconductor industry:

1. **Standardized Power Domain Management:** UPVM ensures consistency and standardization in power domain management practices.
2. **Low-Power Design Implementation:** The tool enables the implementation of low-power design strategies, promoting energy efficiency in semiconductor devices.
3. **Streamlined Design Workflows:**aUPVM contributes to smoother design workflows, reducing complexity and optimizing resource allocation.
4. **Enhanced Power Verification:** UPVM-based verification methods enhance the accuracy and effectiveness of power verification processes.
5. **Efficient UPF File Generation:**aUPVM streamlines the process of generating UPF files, enhancing efficiency in downstream semiconductor design post-synthesis and physical layouts.

UPVM-based low-power verification provides significant advantages, such as cost reduction and faster time-to-market, by integrating low-power validation at an early stage in the design process alongside functional verification. These UPVM-based designs are set to play critical roles across various industries.

1. **Mobile Devices:** UPVM-based devices can expedite the release of smartphones and tablets, potentially advancing them ahead of current technology by half a generation.
2. **Automotive Electronics:** UPVM-based processors can minimize the launch time gap for advanced vehicle technologies reliant on semiconductor chips.
3. **Medical Devices:** Time-efficient semiconductor components enabled by UPVM can expedite the development of medical devices, potentially improving their timely deployment and impact on saving lives.
4. **Data Centers:** UPVM-based devices used in data centers can be delivered ahead of schedule, enhancing data processing capabilities and efficiency in managing vast amounts of data.

V. RESULT

The aUPVM.py tool was tested on a low-power PCIe design, using automation to integrate Low Power methodologies and strategies in the Design & Verification phase, using UVM-like methodologies, and validating the same in the Scoreboards as shown in Figure XII and then generating UPF constructs as shown in Figure XIII. The results demonstrated a significant reduction in the complexity of writing UPF code, with noticeable simplification. Moreover, the seamless integration with UPVM libraries facilitated early-stage power validation. Additionally, the tool's ability to generate UPF files allowed for seamless integration with standard tools for GLS and GDSII validation.

```
class scoreboard;

 my_upf_version upf_ver;
 my_hirearchy_c my_hy;
 my_set_design_top msdp11;
 my_set_scope ssp;
 my_create_power_domain my_crt_pd;
 my_create_supply_port my_crt_sp;
 my_create_supply_net my_crt_sn;
 my_connect_supply_net my_cnt_spyn;
 my_set_domain_supply_net my_sdsn;
 my_create_logic_net_c my_crt_lg_nt;
 my_add_port_state my_aps;
 my_create_pst my_crt_pst;
 my_add_pst_st my_apst;
 my_set_retention_c my_srnt;
 my_set_isolation_c my_sit;
 my_set_level_shifter_c my_sls;
 my_power_switch_c my_pwr_stch;
 my_supply_sets my_ss;
 my_load_upf_c my_ldupf;
 my_associate_supply_set_c my_asociat_ss;
 my_add_power_state_c my_apwrs;
 my_set_dir_c my_sd;
```

Figure XII: UPVM Scoreboard for access Validate the low-power strategies

```
create_power_domain PD_PCIE_TOP
create_power_domain PD_TX -elements { pcie_tx } \
        -supply  {primary} \
        -supply  {backup} \
create_power_domain PD_RX -elements { pcie_rx } \
        -supply  {primary} \
        -supply  {backup} \
create_power_domain PD_TXPLL -elements { pcie_tx/phy_tx/pll }
create_power_domain PD_PHY -elements { pcie_rx/phy }
create_power_domain PD_CDR -elements { pcie_rx/phy/cdr }
create_supply_net Pwr -domain PD_TX
create_supply_net Gnd -domain PD_TX
create_supply_net PwrTxpll -domain PD_TXPLL
create_supply_net GndTxpll -domain PD_TXPLL
create_supply_net Pwr -domain PD_PHY
create_supply_net Gnd -domain PD_PHY
create_supply_net Pwr -domain PD_CDR
create_supply_net Gnd -domain PD_CDR
create_supply_net sw_out -domain PD_PHY
connect_supply_net Pwr -ports {VDD1}
connect_supply_net Gnd -ports {VSS1}
connect_supply_net PwrTxpll -ports {VDD2}
connect_supply_net GndTxpll -ports {VSS2}
add_port_state PST1 \
        -state {NORMAL 1.0} \
        -state {SHUTDOWN 0.8} \
        -state {OFF 0.0}
```

Figure XIII: UPF File Generated by myUPVM template

VI. CONCLUSION

This automation and integration to UPVM libraries has performance improvement and ease of use and is crucial for creating and verifying DUT with Low Power methodologies and strategies files early-stage in semiconductor designs' power management. By incorporating UPF strategies into the SystemVerilog UPVM test bench and Scoreboard, comprehensive verification (the lower power strategies are in SystemVerilog so the cause and effects of power management can be verified early stage in the design phase) and validation of power management strategies are achieved, facilitating early-stage Unified Power Verification. The advancement of the aUPVM & myUPVM template is there is no longer required to learn the UPF constraints to build the low-power design and the verification engineer also can generate the low-power design for DUT. Additionally, it simplifies the process of UPF file creation and enables thorough examination and debugging of low-power features, thereby contributing significantly to the overall success of semiconductor design projects.

REFERENCES

[1] UVM Community (accellera.org) https://accellera.org/community/uvm.

[2] Guide to changes in IEEE 1801-2013 (UPF 2.1) (techdesignforums.com)

[3] Low Power Classes as extension to UVM Package Library 59th Design Automation Conference 2022

[4] Low Power Extension in UVM Power Management DVCon India 2022

[5] Verification Methodology Manual for Low Power https://www.synopsys.com/company/resources/synopsys-press/vmm-low-power.html

[6] Arm Cortex-A53 MPCore Processor Technical Reference Manual r0p4