Successive Refinement – An approach to decouple Front-End and Back-end Power Intent

Kotha Kavya ,Sinha Rohit Kumar

Intel Technology Private Ltd, Bangalore, India -560103

2022 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES VIRTUAL FEBRUARY 28 - MARCH 3, 2022

Introduction

Use of IP in SoC's is essential in order to meet time-to-market requirements and leveraging existing technologies efficiently.
So, designing the power management mechanism should involve both IP requirements and SoC concerns.
IEEE 1801 UPF which we are using now follows Implementation approach enabling early verification of the power management architecture before any implementation decisions are made.
After investing a lot of verification effort to prove that the strategy and UPF file are correct, UPF file must be modified or re-created after selection of the target process technology.
Can not ensure efficient verification and usage of IP.
IPs are re-used by different customers in different configurations and different target technologies. So, if IP UPF should be re-generated based on the customer's usage.
Need decoupling of technology dependent and technology independent UPF.
Therefore, IEEE 1801-2018 UPF (UPF3.1) has come up with a methodology called Successive refinement that supports Incremental specification to improve productivity at IP and reduce IP integration TAT.

Successive Refinement flow

UPF-based specification of the power intent is developed incrementally in three stages –Constraint UPF, Configuration UPF and Implementation UPF
Power intent is specified in a technology independent manner and verified abstractly before implementation
Implementation specific abstraction [power switches, supply set to physical rails, power cells ….]
Captures the constraints inherent in an IP block without predicting a particular configuration.
Separates the logical functionality of power management for a system from the technology-specific implementation of the system.
Enables efficient verification and re-use of IP

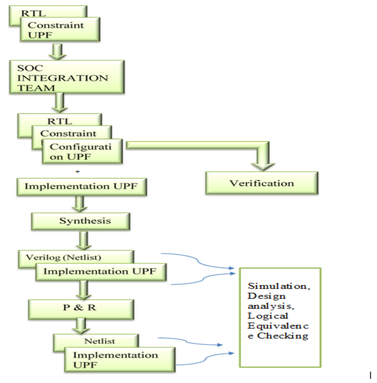

Figure 1 Successive Refinement Flow

Constraint UPF

Provided by the IP provider without any knowledge of how they are used
Power intent inherent to IP
Part of source IP attached with RTL
Atomic power domains , isolation and retention constraints , basic power states(doesn't include supply expressions)
Supply sets without any –function information

Basic information that is present in Constraint UPF:

1. Defining Atomic power domains

It defines the power domains that are used in the IP, and these are atomic power domains which means user is not allowed to further partition them.

```
#################################################################
# Power Domains
#################################################################
create_power_domain PD_CLUSTER_AON   -include_scope \
                                     -supply "primary          SS_${SOC_VDD_CPU}" \
                                     -supply "extra_supplies_0 SS_$SOC_VDD_1V1_TS_CPU" \
                                     -supply "extra_supplies_1 SS_$SOC_VDD_1V8_PLL_A55" \
                                              -supply "extra_supplies_2 SS_$SOC_VDD_1V8_PLL_A76" \
                                              -supply "extra_supplies_3 SS_$SOC_VDD_1V8_PLL_DSU" -atomic


create_power_domain PDSYS   -elements "$PD_SYS_ELE" \
                                 -supply "primary SS_${SOC_VDD_CPU}_CLUSTER_GATED" \
                                -supply "extra_supplies_0 SS_${SOC_VDD_CPU}" -atomic
```

Figure 2 Atomic Power Domain

2. Define clamp value for ISO strategy

Provides isolation and retention constraints which tells the user what they must retain or isolate if chosen to power down parts of the IP in soc application.

```
#################################################################
## specify the isolation clamp values that must be used
#################################################################
set_port_attributes -ports $DSU_CLAMP1 \ -clamp_value 1
set_port_attributes -ports $DSU_CLAMP0 \ -clamp_value 0
set_port_attributes -ports "big_little_cluster/CLUSTER/u_corinth_cluster/dsu_mbist_shell/big_little_cluster_rtl
_tessent_mbist_SRAM_c1_shared_bus_bisr_mux_inst/in0 big_little_cluster/CLUSTER/u_corinth_cluster/dsu_mbist_shel
l/big_little_cluster_rtl_tessent_mbist_SRAM_c2_shared_bus_bisr_mux_inst/in0" \ -clamp_value 0
```

Figure 3 Isolation Clamp Values

3. Define power states without voltage value

Fundamental IP block legal power states without technology information

```
#################################################################
# Power States
#################################################################
add_power_state SS_$SOC_VDD_CPU \
                -state "SS_${SOC_VDD_CPU}_VM"   "-simstate NORMAL" \
                -state "SS_${SOC_VDD_CPU}_GND"  "-simstate NORMAL" \
                -state "SS_${SOC_VDD_CPU}_OFF"  "-simstate CORRUPT"
```

Figure 4 Fundamental power states

4. Supply sets without any –function information

IP supply set information without any function information

```
#################################################################
# Create Supply Set
#################################################################
create_supply_set SS_$SOC_VDD_CPU
create_supply_set SS_${SOC_VDD_CPU}_CLUSTER_GATED
create_supply_set SS_${SOC_VDD_CPU}_CORE0_GATED
```

Figure 5 Supply sets

Configuration UPF

Application specific configuration
Required for simulation – created by end user
Composite power domains, logic expressions , power states , Isolation and retention strategies

Information that is present in Configuration UPF:

1. Define design ports

IP Design Ports that the system want to control are added using create logic port and create logic net and connected using connect_logic_net

```
#################################################################
# Create logic Port
#################################################################
create_logic_port $SOC_VDD_CPU                    -direction in
create_logic_port $SOC_GROUND                  -direction in
create_logic_port $SOC_VDD_1V1_TS_CPU          -direction in

#################################################################
# Create logic Net
#################################################################
create_logic_net $SOC_VDD_CPU
create_logic_net $SOC_VDD_1V1_TS_CPU
```

Figure 6 Logic ports, nets definition and connection

2. Define ISO/RET strategies and how they are controlled

Isolation strategy specifies clamp values consistent with the specifications in the constraint UPF
Specifies the retention strategies to be used for each power domain.

```
#################################################
# ISO and retention strategies implementation
#################################################
set_isolation iso_cluster_0 \
        -domain PDSYS \
        -isolation_supply_set SS_${SOC_VDD_CPU} \
        -source SS_${SOC_VDD_CPU}_CLUSTER_GATED \
        -sink SS_${SOC_VDD_CPU} \
    -exclude_elements $DSU_CLAMP0 \
        -isolation_signal $nISOLATECPU_PDSYS \
        -isolation_sense low \
        -location parent
```

Figure 7 Isolation strategy

Implementation UPF

Technology specific implementation of UPF
Required for Implementation
Supply nets/ports, switches etc.
Define Supply Voltages in power states

Information that is present in Implementation UPF:

1. Define supply and network elements for the design

```
#################################################################
# Create Supply Port
#################################################################
create_supply_port $SOC_VDD_CPU                    -direction in
create_supply_port $SOC_GROUND                  -direction in
create_supply_port $SOC_VDD_1V1_TS_CPU          -direction in

#################################################################
# Create Supply Net
#################################################################
create_supply_net $SOC_VDD_CPU
create_supply_net $SOC_VDD_1V1_TS_CPU
create_supply_net $SOC_VDD_1V8_PLL_A55
```

Figure 8 Supply ports and nets creation

2. Defining Power Switches

Does not have to make any decisions prior to implementation UPF
Helps to keep the constraint UPF and configuration UPF files in an abstract form

```
#################################################################
# Power switches
#################################################################
set nPWRUPCPU_HAMMER_PDSYS
big_little_cluster/CLUSTER/nPWRUPCPU_HAMMER_PDSYS
set nPWRUPCPU_TRICKLE_PDSYS
big_little_cluster/CLUSTER/nPWRUPCPU_TRICKLE_PDSYS
set nPWRUPCPU_HAMMER_ACK_PDSYS
big_little_cluster/CLUSTER/nPWRUPCPU_HAMMER_ACK_PDSYS_DSU
set nPWRUPCPU_TRICKLE_ACK_PDSYS
big_little_cluster/CLUSTER/nPWRUPCPU_TRICKLE_ACK_PDSYS_DSU
set nISOLATECPU_PDSYS big_little_cluster/CLUSTER/nISOLATECPU_PDSYS
create_power_switch ps_PDSYS_main -domain PDSYS \
 -output_supply_port          "VDD SS_${SOC_VDD_CPU}_CLUSTER_GATED.power" \
   -input_supply_port           "TVDD SS_${SOC_VDD_CPU).power" \
   -control_port                [list NSLEEPIN1 $nPWRUPCPU_HAMMER_PDSYS] \
   -control_port                 [list NSLEEPIN2 $nPWRUPCPU_HAMMER_PDSYS]\
   -on_state                     { on TVDD { NSLEEPIN1 & NSLEEPIN2 }) \
   -off_state                    { off     { !NSLEEPIN1 | !NSLEEPIN2 })
```

Figure 10 Power Switches

3. Connecting supply nets with supply sets

Supply sets defined for each power domain are connected to supply nets provided by the implementation.
Done using the –function option of the 'create_supply_set' command .

```
#################################################################
# Create Supply Set
#################################################################
create_supply_set SS_$SOC_VDD_CPU       -update          -function "power $SOC_VDD_CPU"
  -function "ground $SOC_GROUND"  -function "pwell $SOC_GROUND"    -function "nwell $SOC_VDD_CPU"
create_supply_set SS_${SOC_VDD_CPU}_CLUSTER_GATED -update   -function "power ${SOC_VDD_CPU}_CLUSTER_GATE
D"   -function "ground $SOC_GROUND"  -function "pwell $SOC_GROUND"    -function "nwell $SOC_VDD_CPU"
create_supply_set SS_${SOC_VDD_CPU}_CORE0_GATED   -update -function "power ${SOC_VDD_CPU}_CORE0_GATED"
  -function "ground $SOC_GROUND"  -function "pwell $SOC_GROUND"    -function "nwell $SOC_VDD_CPU"

#################################################################
# Connect Supply Net
#################################################################
connect_supply_net $SOC_VDD_CPU                -ports $SOC_VDD_CPU
connect_supply_net $SOC_GROUND                 -ports $SOC_GROUND
connect_supply_net $SOC_VDD_1V1_TS_CPU         -ports $SOC_VDD_1V1_TS_CPU
```

Figure 11 Supply set and their connection with supply net

4. Define Supply Voltages in power states

voltage values for each supply sets are defined by using the '-supply_expr' option on add_power_state .

```
#################################################################
# Power States
#################################################################
add_power_state SS_$SOC_VDD_CPU \   -update
        -state       "SS_${SOC_VDD_CPU}_VM"       "-supply_expr       \(power       ==
`\{FULL_ON,$SOC_VDD_CPU_VM_MIN,$SOC_VDD_CPU_VM_NOM,$SOC_VDD_CPU_VM_MAX\}\}   -simstate
NORMAL" \          -update
        -state       "SS_${SOC_VDD_CPU}_GND"      "-supply_expr       \(ground      ==
`\{FULL_ON,$SOC_GROUND_VM_MIN,$SOC_GROUND_VM_NOM,$SOC_GROUND_VM_MAX\}\}      -simstate
NORMAL" \          -update
        -state       "SS_${SOC_VDD_CPU}_OFF"      "-supply_expr \(power ==  `\{OFF\}\} -
simstate CORRUPT"
```

Figure 12 Power states with voltage Information

Results/ Summary

Successive Refinement enables
Abstraction of power intent from Soft IP UPF with the goal to improve productivity at IP and reduce IP integration TAT
Decreases Risk and more successful usage of IP
Separation of Technology dependent and Technology Independent Verification
Earlier verification before technology information known
Easier retargeting to different technologies
No need of logical Reverification for different technologies