Simplifying Hierarchical Low power designs using Power Models in Intel Design

Rohit Kumar Sinha, Intel India

accellera SYSTEMS INITIATIVE™

2019 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION INDIA

Motivation

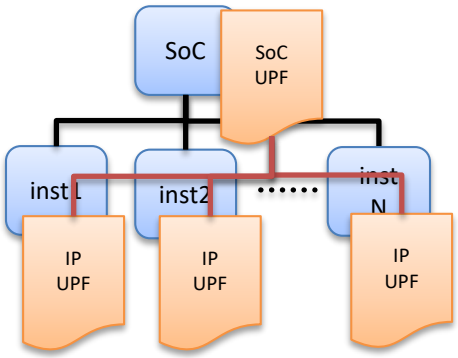
- With the increased complexity of Intel Client SoC design
  - IP designs are sourced from both internal and external channels
  - the bottom-up SoC UPF with 1000+ IPs
  - Limitations of hierarchical UPF methodology
    - File Based
    - Instance Hierarchy
  - Power aware functional verification in the hierarchical design
- Hard Marco Blocks
  - Pre-Implemented IPs
  - Huge number of Instantiations leading to huge UPF code
  - UPF is verbose and repeatable

Structure of Hierarchical UPF

- Hierarchical UPFs are scoped UPFs
- Power Intents specified in separate files
- UPF files are associated with instances in design hierarchy
- Typical hierarchical UPF structure is like below

```
Load_upf $::env($MODEL_ROOT)/source/*/pgpunt_wrap.upf –scope punit_wrap
connect_supply_net “VCCA” –ports “pgpunt*/vcca”
connect_supply_net “VCCB” –ports “pgpunit*/vccb”
connect_supply_net “VSS” –ports “pgpunt*/vss”

Load_upf $::env($MODEL_ROOT)/source/*/soc_pg_pwrup_cell_wrap.upf –scope soc_pg_punit_wrap
connect_supply_net “VCCA” –ports “soc_pg_pgpunit*/vcca”
connect_supply_net “VCCC” –ports “soc_pgpunit*/vccc”
connect_supply_net “VCCD” –ports “soc_pgpunit*/vccd”
connect_supply_net “VSS” –ports “soc_pgpunit*/vss”
```



- Huge number of load_upf for each instances

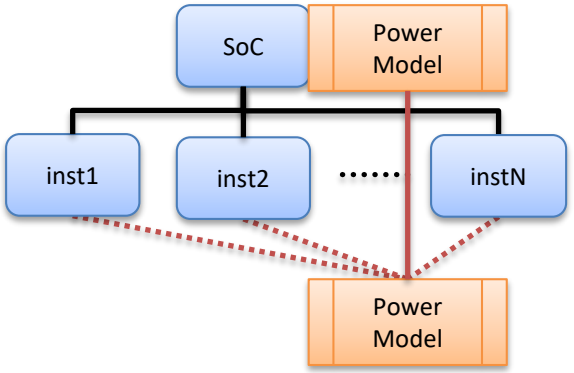
Power Models with Hard Macros

- For Hard IP, the behavioral model
  - Shipped with the IP is used which contains only the internal functionality
  - Power aware behavior Not Available.
- Power Intent of Hard IP in 2 parts,
  - one describes the internal behavior and
  - one gives the interface information or connectivity with the other IP’.
- The typical design hierarchies for integrating IP blocks into a SoC are:

  IP block -> Unit -> Partition - > SoC

UPF Power Models

- Power Models provide a modular way of writing UPF
  - Similar to module/endmodule in SystemVerilog
- Power Intent described for logical power model
  - Mapped to one or more models
- Uses
  - begin_power_model/end_power_model
  - define_power_model (in UPF 3.1)
- Applied to one or more instances of model
  - apply_power_model command




```
begin_power_model macro_cell -for { c74p13rflfc1r2w184x128h }
create_power_domain pd_ungated_rf
create_supply_port vcc -direction in -domain pd_ungated_rf
create_supply_net vcc -domain pd_ungated_rf
connect_supply_net vcc -ports vcc
set_domain_supply_net pd_ungated_rf -primary_power_net vcc -primary_ground_net
vss
add_port_state vcc    -state "vcc_on_hv $VCC_SUPPLY_HV" -state "vcc_on_lv
$VCC_SUPPLY_LV" -state "vcc_ret $VCC_SUPPLY_RET"
... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ...
... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ... ...
create_pst pst_pd_rf    -supplies            "vcc                                vss"
add_pst_state hv -pst pst_pd_rf -state { vcc_on_hv  ps_VSS_0p0 }
add_pst_state lv -pst pst_pd_rf -state { vcc_on_lv  ps_VSS_0p0 }
add_pst_state all_off   -pst pst_pd_rf -state { vcc_ret  ps_VSS_0p0  }
end_power_model
```

Power Model are created, similar to module/endmodule of System verilog

Power Aware Modelling of Hard IP

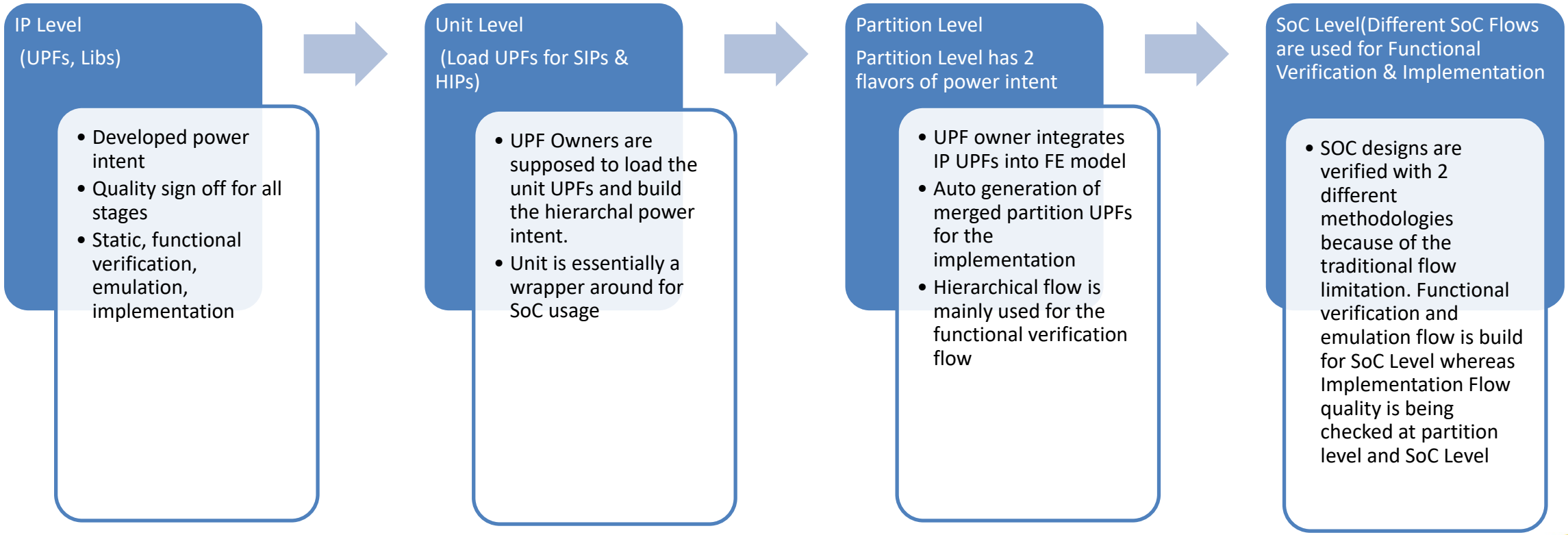
- Power Aware Model Components
  - Power Management Interface
    - can be expressed in UPF or liberty
    - includes the information of the top level power domain and connections
  - Supply Information
    - Pins at the boundary of IP need to be connected to the supplies correctly
  - Related Supplies
    - the related supply attributes defined for those pins
  - Power States
    - different power modes in which IP can operate
  - Interface Protection Cells
    - Isolation cells inside the HIPs must be conveyed while RTL integration

Challenges with Power Models with Hard Macros

- Ensure the correctness and consistent results with Hard IP power models across all the Front-end & Back-end Flows
- Handling of domain dependent supply set and domain independent supply
- Handling of instrumental assertion code in UPF2.0 syntax in functional verification and emulation environment
- Handling of SRSNs which are now set using the set_port_attribute command.
- Equivalence checking between the traditional hierarchical UPF flow and the UPF flow with power models

Power Intent Body

Low Power Specs are defined by Micro Power Architect and it includes top interfaces, partitions level interfaces, crossing table & bump details.



Traditional Hierarchical UPF for Hard Marcos

```
create_power_domain soc_pd  -include_scope
create_supply_net VCC
create_supply_net VSS
……..
Load_upf ip.upf -scope {I1}
Load_upf ip.upf -scope {I2}

Load_upf ip.upf -scope {I3}
……..
Connect_supply_net VCC -ports { \
I1/Vcc \
I2/Vcc \
I3/Vcc }
Connect_supply_net VSS -ports { \
I1/Vss \
I2/Vss \
I3/Vss }
```

Soc.upf

```
create_power_domain ip_pd
create_supply_port vcc -direction in -domain  pd
create_supply_net vcc -domain pd
connect_supply_net vcc -ports vcc
……..
create_supply_port vss -direction in -domain  pd
create_supply_net vss -domain pd
connect_supply_net vss -ports vss
……..
set_domain_supply_net pd_ungated_rf - primary_power vcc -
primary_ground vss
```

hard_ip.upf

Hierarchical UPF with Power Models for Hard Marcos

```
create_power_domain soc_pd -include_scope
create_supply_net VCC
create_supply_net VSS
……..
apply_power_model macro_cell -elements { \
I1 \
I2 \
I3 \
} -port_map { {I1/vcc VCC} {I2/vcc VCC} {I3/vcc VCC} {I1/vss VSS} {I2/vss VSS} {I3/vss VSS} }

add_port_state "I1 /vcc" \
    -state "SIPPCIE2_CORE_PGD_ON_MIN  $SIPPCIE2_VCCSA_SUPPLY_MIN" \
    -state "SIPPCIE2_CORE_PGD_ON_MAX  $SIPPCIE2_VCCSA_SUPPLY_MAX" \
    -state "SIPPCIE2_CORE_PGD_OFF  off"
add_port_state "I2 /vcc" \
    -state "SIPPCIE2_CORE_PGD_ON_MIN $SIPPCIE2_VCCSA_SUPPLY_MIN" \
    -state "SIPPCIE2_CORE_PGD_ON_MAX $SIPPCIE2_VCCSA_SUPPLY_MAX" \
    -state "SIPPCIE2_CORE_PGD_OFF  off"

create_pst pt -supplies  " I1/vcc    I2/vcc "

add_pst_state s1 -pst pt -state { SIPPCIE2_CORE_PGD_ON_MIN SIPPCIE2_CORE_PGD_OFF }
```

Soc.upf

```
begin_power_model macro_cell -for { c74p13rflfc1r2w184x128h }
create_power_domain pd_ungated_rf
create_supply_port vcc -direction in -domain pd_ungated_rf
create_supply_net vcc -domain pd_ungated_rf
connect_supply_net vcc -ports vcc
set_domain_supply_net pd_ungated_rf -primary_power_net vcc -primary_ground_net
vss
add_port_state vcc   -state "vcc_on_hv $VCC_SUPPLY_HV" -state "vcc_on_lv
$VCC_SUPPLY_LV" -state "vcc_ret $VCC_SUPPLY_RET"
……..
……..
create_pst pst_pd_rf   -supplies          "vcc                              vss"
add_pst_state hv -pst pst_pd_rf -state { vcc_on_hv ps_VSS_0p0 }
add_pst_state lv -pst pst_pd_rf -state { vcc_on_lv ps_VSS_0p0 }
add_pst_state all_off  -pst pst_pd_rf -state { vcc_ret ps_VSS_0p0 }
end_power_model
```

hard_ip.upf

Hard IP with Multiple instances UPF Comparison

Hierarchical UPF	Modular UPF with Power Models
Loading 10 instances of c74p13rflfc1r2w216x128h.upf→ 10 load_upf commands→ load_upf $ROOT_DIR/c74p13rflfc1r2w216x128h.upf -scope pxtss_rf_1 load_upf $ROOT_DIR/c74p13rflfc1r2w216x128h.upf -scope pxtss_rf_2 load_upf $ROOT_DIR/c74p13rflfc1r2w216x128h.upf -scope pxtss_rf_3 load_upf $ROOT_DIR/c74p13rflfc1r2w216x128h.upf -scope pxtss_rf_4 load_upf $ROOT_DIR/c74p13rflfc1r2w216x128h.upf -scope pxtss_rf_5 load_upf $ROOT_DIR/c74p13rflfc1r2w216x128h.upf -scope pxtss_rf_6 load_upf $ROOT_DIR/c74p13rflfc1r2w216x128h.upf -scope pxtss_rf_7 load_upf $ROOT_DIR/c74p13rflfc1r2w216x128h.upf -scope pxtss_rf_8 load_upf $ROOT_DIR/c74p13rflfc1r2w216x128h.upf -scope pxtss_rf_9 load_upf $ROOT_DIR/c74p13rflfc1r2w216x128h.upf -scope pxtss_rf_10 2 connect_supply_net commands→ connect_supply_net VCC -ports { pxtss_rf_1/vcc pxtss_rf_2/vcc pxtss_rf_3/vcc pxtss_rf_4/vcc pxtss_rf_5/vcc pxtss_rf_6/vcc pxtss_rf_7/vcc pxtss_rf_8/vcc pxtss_rf_9/vcc pxtss_rf_10/vcc } connect_supply_net VSS -ports { pxtss_rf_1/vss pxtss_rf_2/vss pxtss_rf_3/vss pxtss_rf_4/vss pxtss_rf_5/vss pxtss_rf_6/vss pxtss_rf_7/vss pxtss_rf_8/vss pxtss_rf_9/vss pxtss_rf_10/vss }	Creating power model for c74p13rflfc1r2w216x128h.upf → 1 apply_power_command→ apply_power_model c74p13rflfc1r2w216x128h -elements { pxtss_rf_1 pxtss_rf_2 pxtss_rf_3 pxtss_rf_4 pxtss_rf_5 pxtss_rf_6 pxtss_rf_7 pxtss_rf_8 pxtss_rf_9 pxtss_rf_10} I -port_map { {pxtss_rf_1/vcc VCC} {pxtss_rf_2/vcc VCC} {pxtss_rf_3/vcc VCC} {pxtss_rf_4/vcc VCC} ..... {pxtss_rf_1/vss VSS} {pxtss_rf_2/vss VSS} {pxtss_rf_3/vss VSS} {pxtss_rf_4/vss VSS} ... }

- ✓ Automatic instantiation with single command
- ✓ Supply & other connections in single command
- ✓ Inbuilt checks to ensure correctness of the UPF constructs

Advantages of Power Modelling

- Single apply_power_model command will provide the power interface information to the top upf.
- Number of hand-written UPF lines is greatly reduced thereby eliminating the manual error
- Front Tools flags to the point violation which eases debug ability while reducing noise in cleaning up the issues.
- Simulation time for the power aware simulation in vendor tool was reduced by 30%
- Power Aware models of HIPs are more accurate and eases debug ability

Conclusion

- The hierarchical low power intent with power models comprising of power management interface and power management behavior enable more accurate design flow verification and implementation
- This helps in easy readability and debug ability and it helps to catch complicated functional bugs early in the design flow thereby reducing the need for costly re-spins.