Complex Low Power Verification Challenges in NextGen SoCs : Taming the Beast !

Abhinav Nawal (Freescale)

Gaurav Jain (Freescale)

Joachim Geishauser (Freescale)

accellera SYSTEMS INITIATIVE

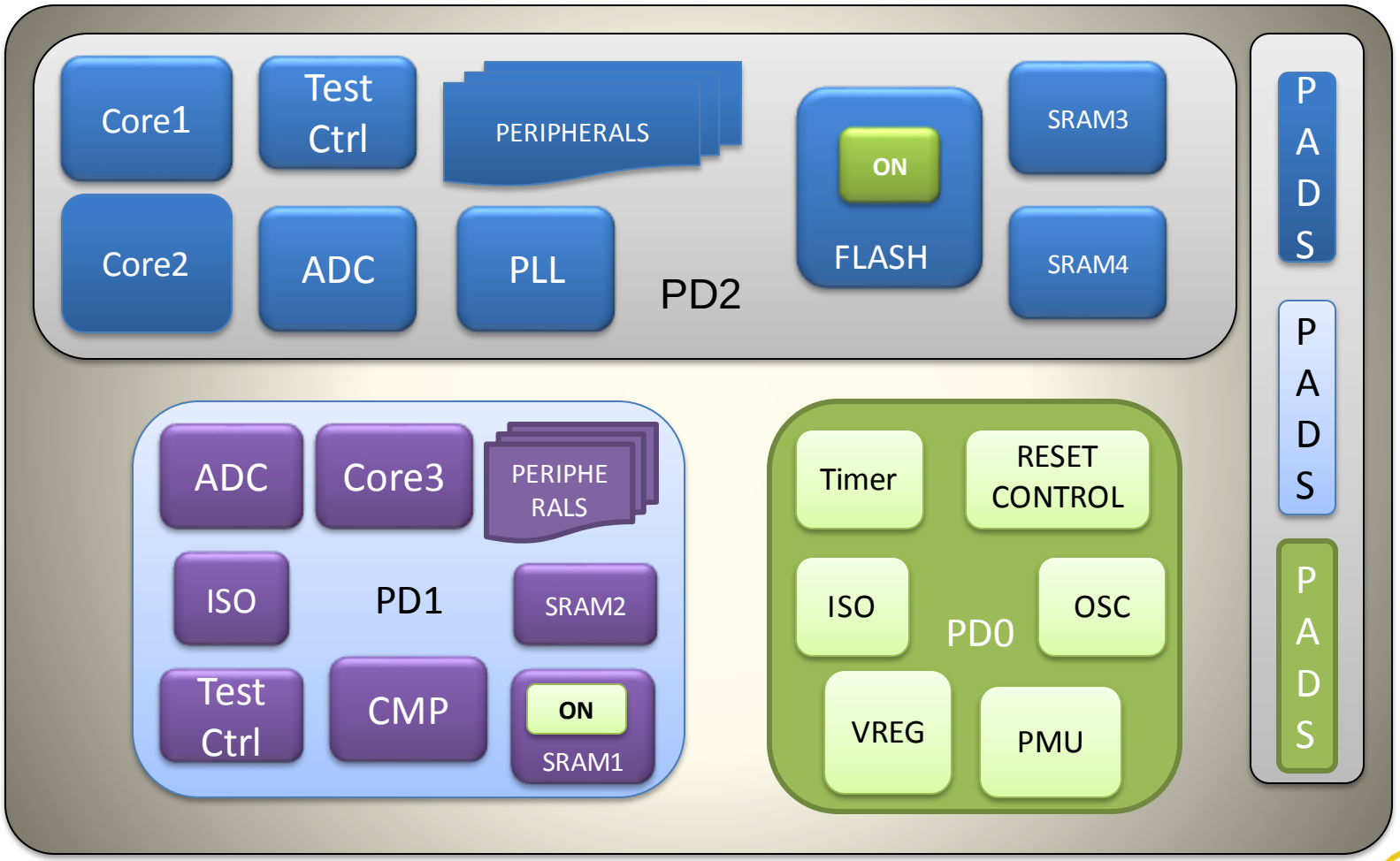
freescale semiconductor

2014 DESIGN AND VERIFICATION DVCON CONFERENCE AND EXHIBITION EUROPE

SCOPE

- Introduction
- Low Power Verification Challenges
- Low Power Verification Flow
  - CPF Enabled RTL Low Power Simulation
  - Low Power Assertions
  - Low Power Intent Coverage
  - Uncovering Power Up Issues Hidden by X Optimism
  - CPF Enabled Gate Level Simulations
- Low Power Design Issues Caught
- Conclusion
- References
- Questions & Answers

Low Power Mixed Signal SOC



Low Power Verification Challenges

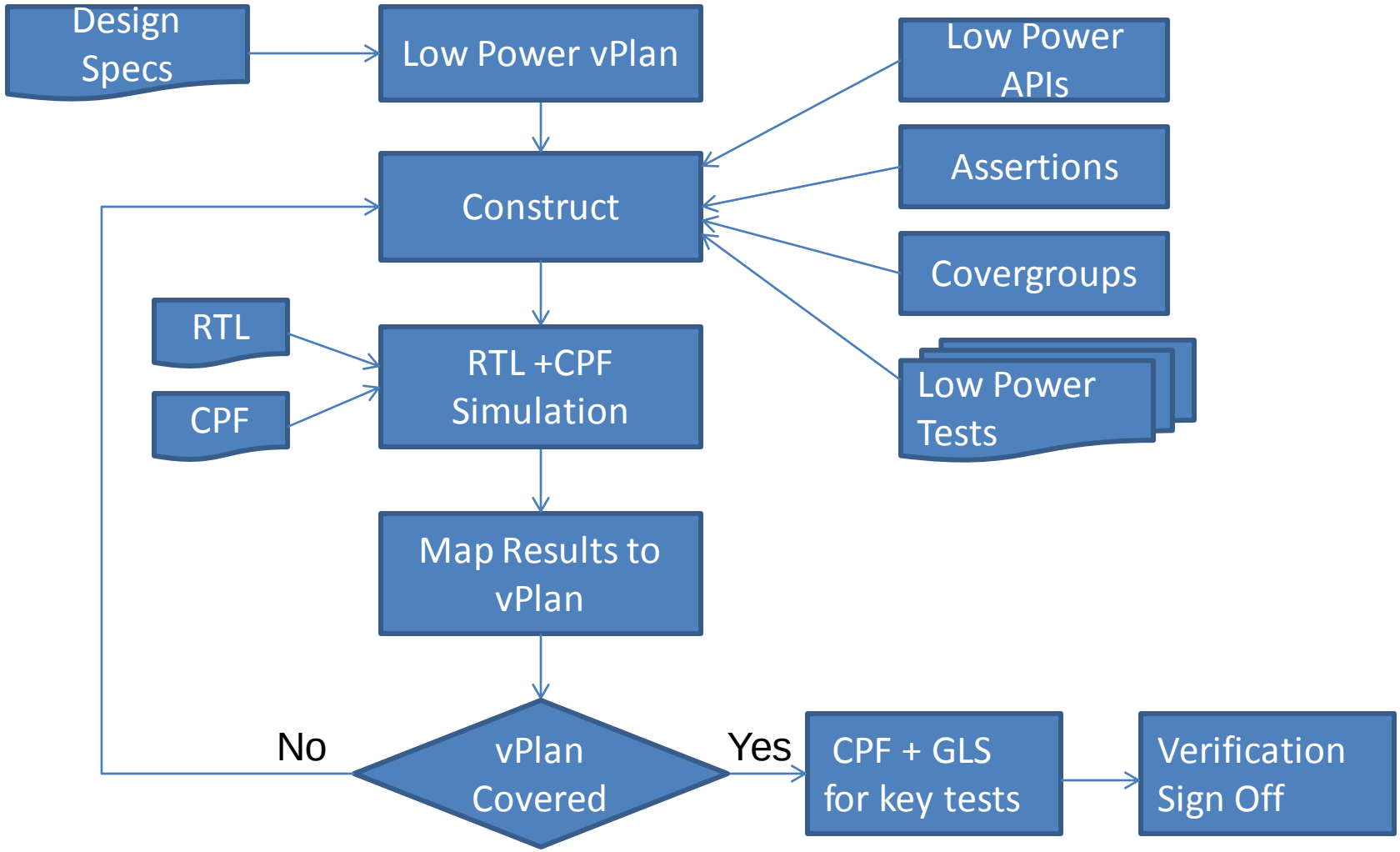
- Low Power Architecture , Power Control & Monitoring
- Power Gating and Power Up Operation of each Domain
- Multiple Low Power Modes and Mode Transitions
- Power Gating & Data Retention in Memories
- Low Power Integration of Hard Macros and Analog IP
  - ADC , Comparator , Power Switches ,Voltage Regulators , PADS
- Impact of Clock frequency Switching , Reset Domains on PMU & System Bring up
- Fine Grained Clock Gating & Dynamic Voltage Scaling
- X-Optimism Implications Post Power Up
- Test Mode Checks for Design & Analog IPs in Low Power Modes

Low Power Verification Challenges (cont.)

Number of Power Domains	**PDn**
Number of Power Modes	PMn
Number of Clock Sources (IRC/OSC/PLL)/Clock Ratios	CSn
Number of Resets Domains & Reset Types	RSn
Number of Wakeup Sources (GPIO, NMI, Resets, Interrupts)	WSn
Number of Retention Memories	RMn
No. of Analog-Digital Interfaces and Macro Models	ADn
Number of Low Power Peripherals	LPn
Number of Abort Scenarios(early wakeup, resets)	ASn

Total Verification Scenarios = PDn X PMn X PMTn X CSn X RSn X WSn X RMn X ADn X LPn X ASn..

Low Power Verification Flow



Low Power Assertions Features

- Reset mapping checks to ensure correct type and power domain specific reset connection to each IP
- Low Power Mode Entry/Exit Protocol assertions
- Macro model Low Power Protocol assertions to verify sequencing and protocol for low power control signals
- Isolation - Reset Sequencing Checks
- Low Power Peripherals operation across mode transitions
- Fine Grained Clock gating scheme assertions for low power modes involving IP specific clock configurations
- Post Power Up Uninitialized signals detection checks
- Dynamic Voltage Scaling enablement checks

Low Power Assertions Examples

Reset Mapping Check

```
property prop_ip_reset_sig_chk (ip_reset_sig , domain_reset_sig ,power_domain);
@(posedge simulation_clk)  !ip_reset_sig  |-> !domain_reset_sig  ;  endproperty
```

VREG and PMU Low Power Protocol Checker

```
Standby_to_Run_when_sys_wakeup  : assert property (@(posedge simulation_clk)
$rose(sys_wakeup) & curr_mode==`STBY  |-> ##[0:N] (sw1_done & sw2_done )
##[0:N] !iso1 & !iso2 ##[1:N] curr_mode==`RUN);
```

SRAM Low Power Protocol for Sleep

```
sleep_asserted_before_rd_sw_open  : assert property (@(posedge simulation_clk)
$fell(`sw_rd.sw_open)  |-> $past(`sram.sleep,2)  &&  $past(`sram.sleep,1)  &&
`sram.sleep);
```

Isolation is asserted before Domain is Reset

```
iso_asserted_before_reset  : assert property (@(posedge simulation_clk)
`pmu.target_mode==`STBY && !`rst_ctrl.pd2_dest_reset   |-> $past(`pmc.pd2_iso
,2) &&  $past(`pmc.pd2_iso,1) && `pmc.pd2_iso );
```

Low Power Assertions Examples (cont.)

Low Power Peripheral Clock Gating Check

```
Lp_mode_periph_clk_gated_when_module_disabled : assert property
(@(posedge simulation_clk) `pmu.target_mode==`LP_STOP &&
`pmu.periph_stop_reg[0] |-> ##[40:100] `LP_Periph_num0.clk == 1'b0 ##1
$stable(`LP_Periph_num0.clk) );
```

Post Power Up Uninitialized outputs detection checks

```
property sig_not_x_post_power_up(isolation_condition,signal);
   @(posedge sim_clk) $fell(isolation_condition) |=> !$isunknown(signal);
endproperty
```

Low Power SoC Scenario Coverage

- SV Covergroups with Cross cover points to check if SoC Low Power Scenario is covered

```
// Standby Mode Exit via External Reset toggle with no System clock present

bit ext_reset_b , irc_clk_en;
bit[3:0] current_mode ;

covergroup  STBY_MODE_EXT_RST_IRC_OFF @(posedge simulation_clk);

EXT_RST : coverpoint ext_reset_b {
bins ext_reset_b_low = (1=>0);
bins ext_reset_b_high= (0=>1) ;
}

IRC_EN : coverpoint irc_clk_en {
bins irc_clk_en_high = {1};
bins irc_clk_en_low = {0};
}

CURR_MODE : coverpoint current_mode {
bins STBY = {4'hD};
bins RUN = {4'h3} ;
bins STOP = {4'hA};
}

stby_rst_irc_en : cross EXT_RST , IRC_EN , CURR_MODE {
bins stby_ext_rst_exit_irc_off = binsof(CURR_MODE.STBY) && binsof(IRC_EN.irc_clk_en_low) &&
binsof(EXT_RST.ext_reset_b_low);
}

endgroup

STBY_MODE_EXT_RST_IRC_OFF stby_mode_ext_rst_irc_off_inst = new;
```

Cross Cover Point checks all events occurred simultaneously

X-Optimism & Impact in Low Power Simulation

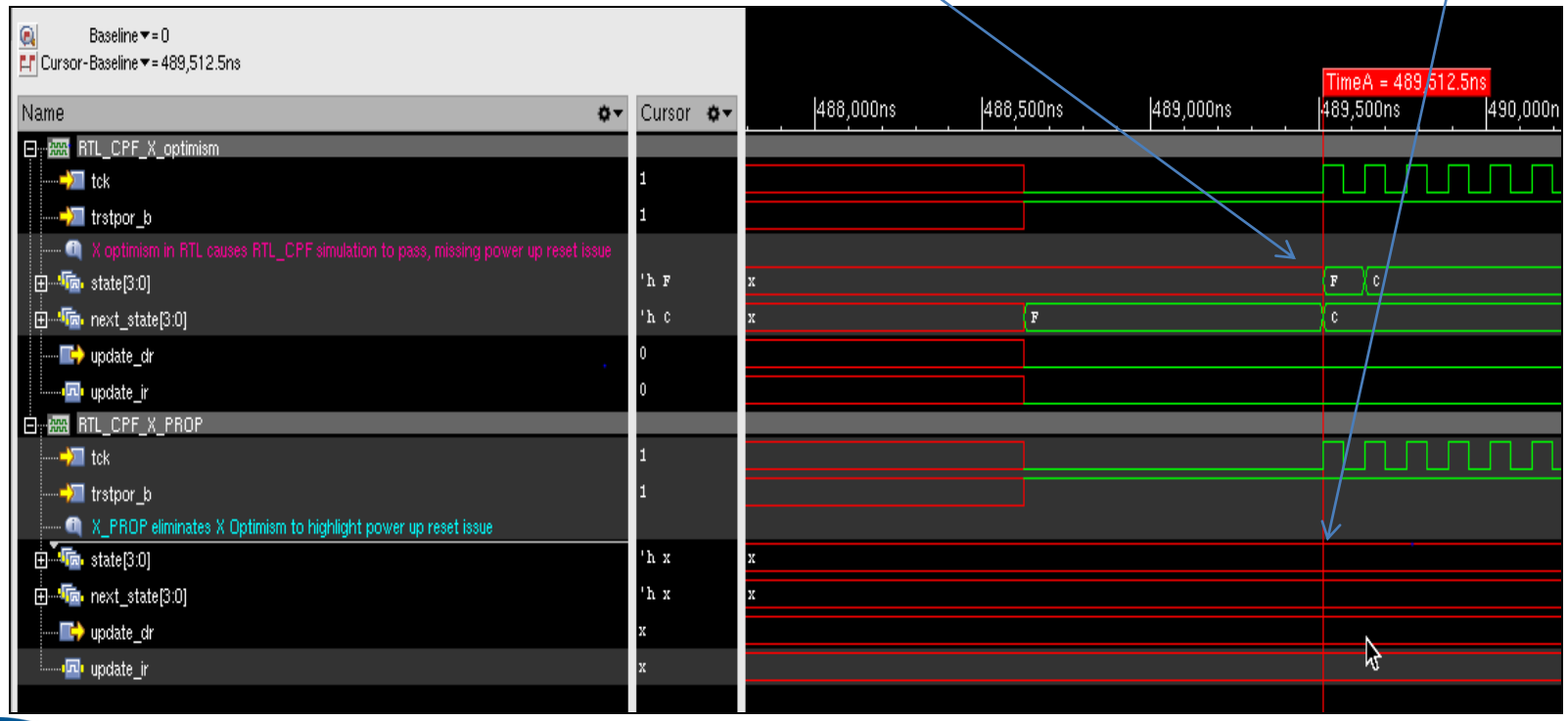
- X-Optimism In RTL Simulation
  - A state where simulators convert a “X” value input to a condition or gate to 0 or false value at the output
- X-Optimism Impact in Low Power Simulation
  - Simulators drive “X” to simulate power shutoff
  - RTL X-Optimism can prevent X propagation masking Low Power design issues in Power Down and Power Up domains
  - CPF+ RTL v/s CPF + GLS simulation mismatch
- Pessimistic X-Propagation Solution in Simulator for Low Power Simulations with CPF/UPF
- Assertion Based Solution to detect Residual X Post Power Up

RTL + CPF Simulation v/s RTL+CPF+ X-Prop Simulation

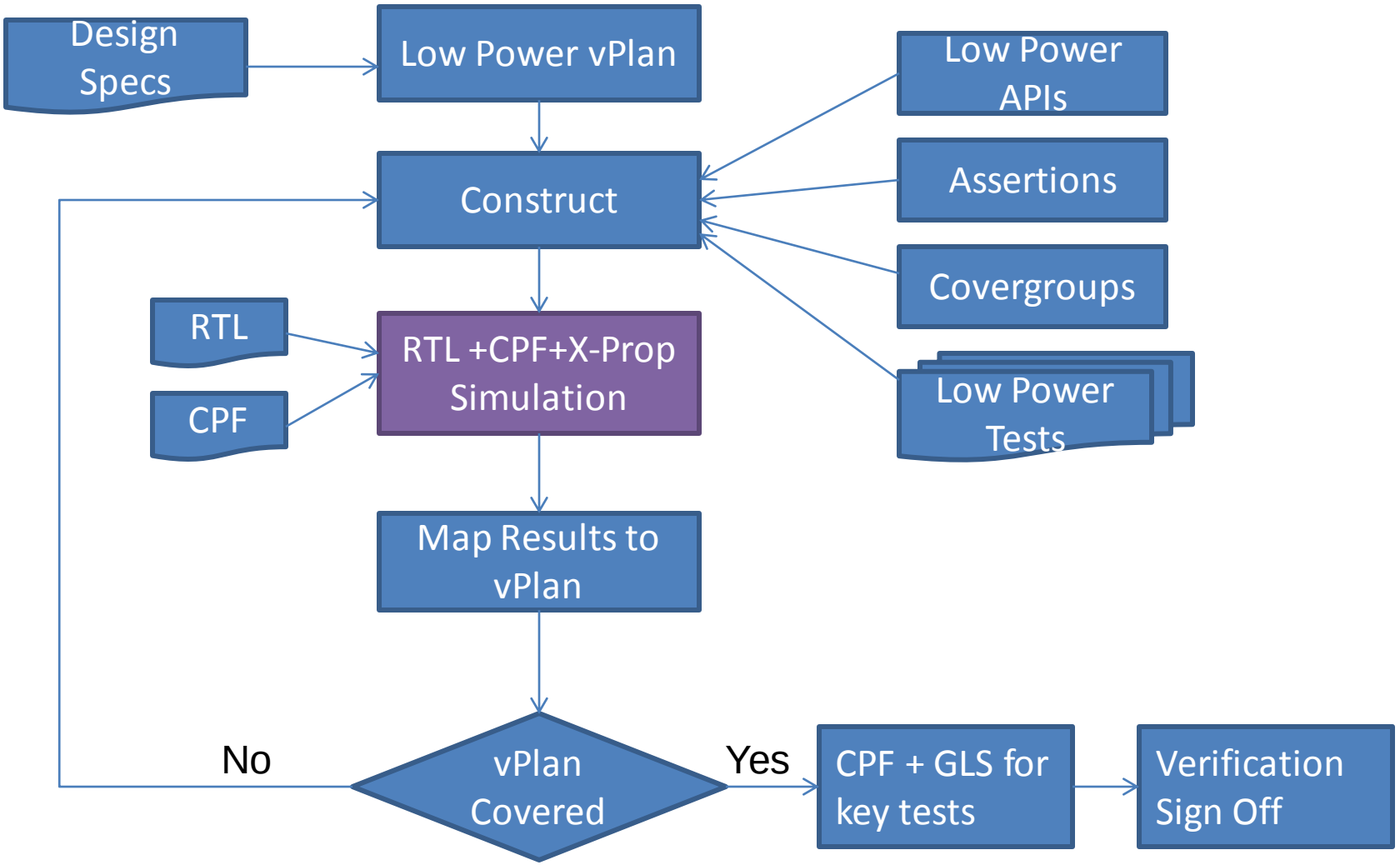
- X Optimism is … ent with …
- Power Up res… ked du… nt enabling FSM to proceed in RTL simulation

RTL + CPF Waves
X-Optimism due to default assignment as clock starts

RTL + CPF + X-Prop Waves
X being propagated in design



Low Power Verification Flow with X-Prop

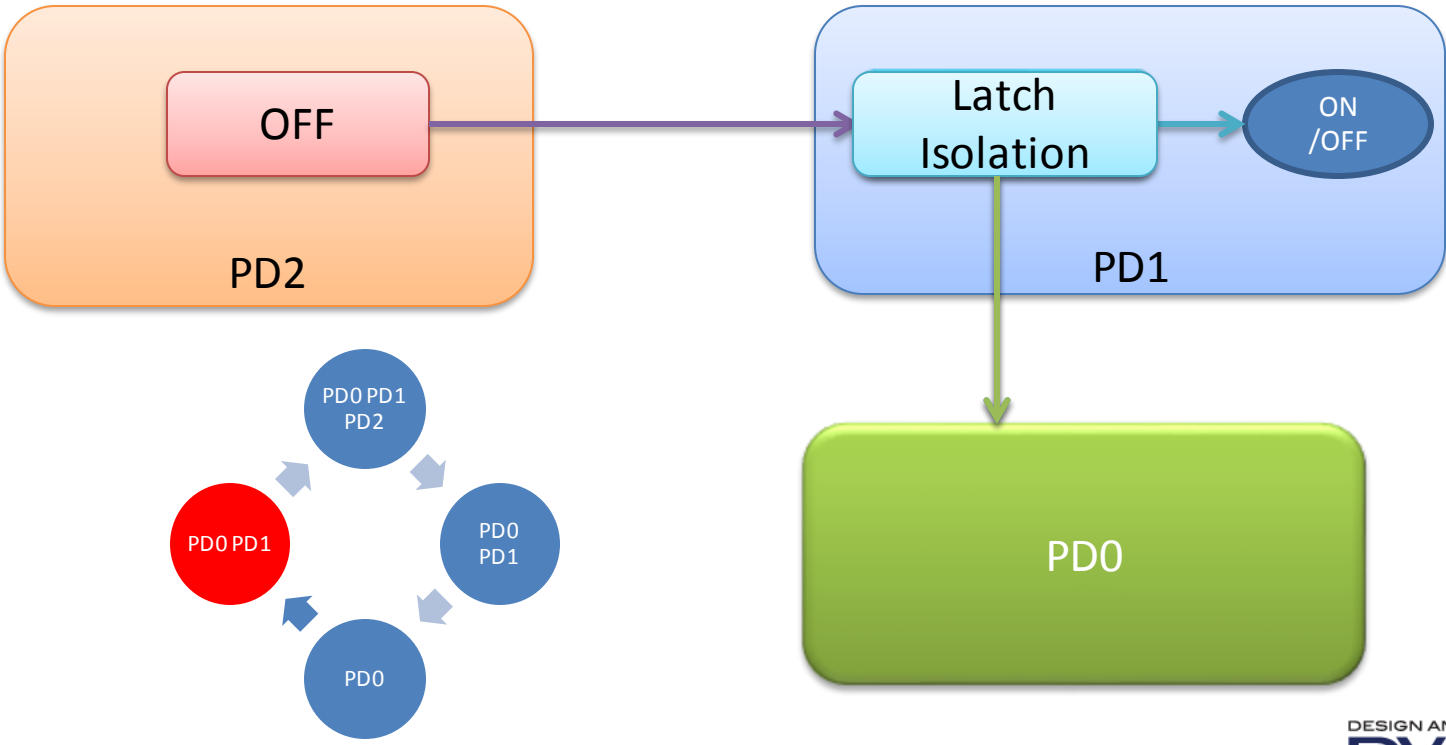


Low Power Design Issues

- Isolation Issues
- Low Power Mode Entry/Exit Protocol Issue
- Macro Model Design & Integration Issues
- Memory Retention Issue
- Power On Reset / System Bring UP issues
- Analog IP Power connectivity Issues
- X-Optimism Issues

Isolation Issues

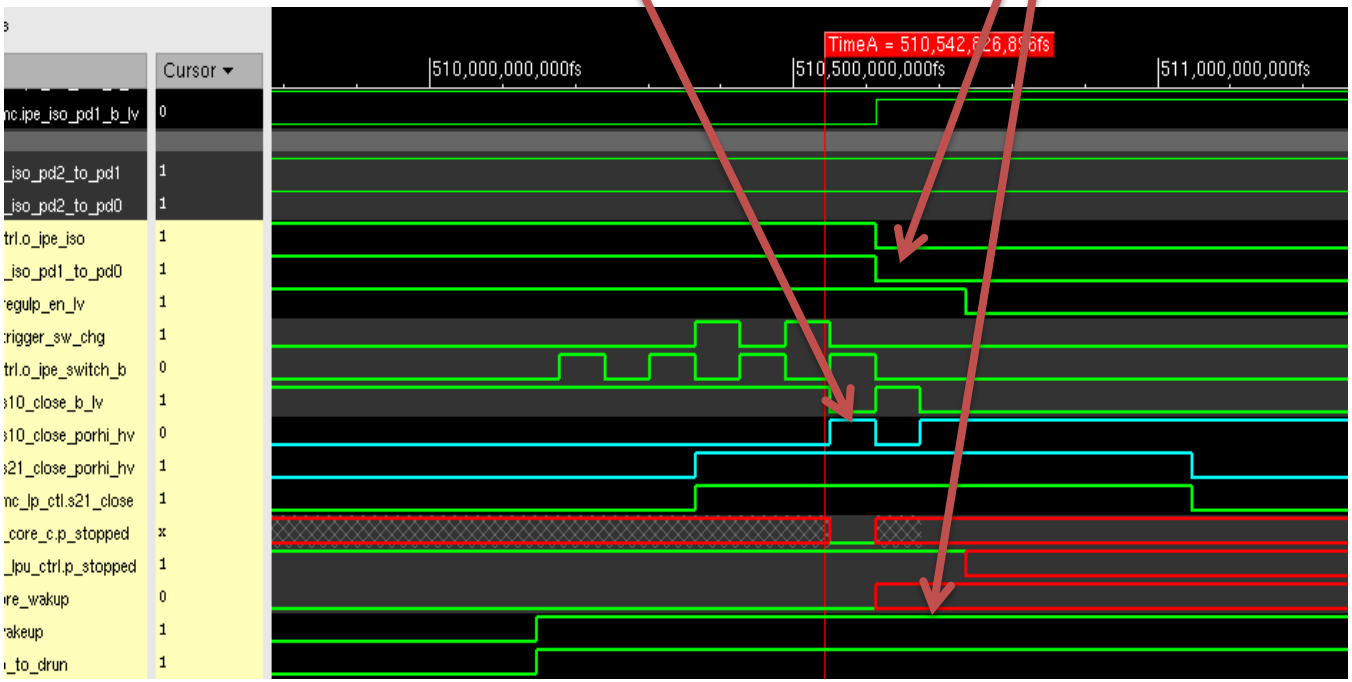
- Missing Isolation on nets crossing from OFF to ON domain
- Wrong Isolation type used



Low Power Mode Entry Protocol Issue

Unexpected Toggle on shutoff condition

Corruption on AON Domain signals as ISO is not asserted



INTENDED BEHAVIOR

1. RUN > STBY > RUN
2. De-assert Power Shutoff
3. De-assert Isolation

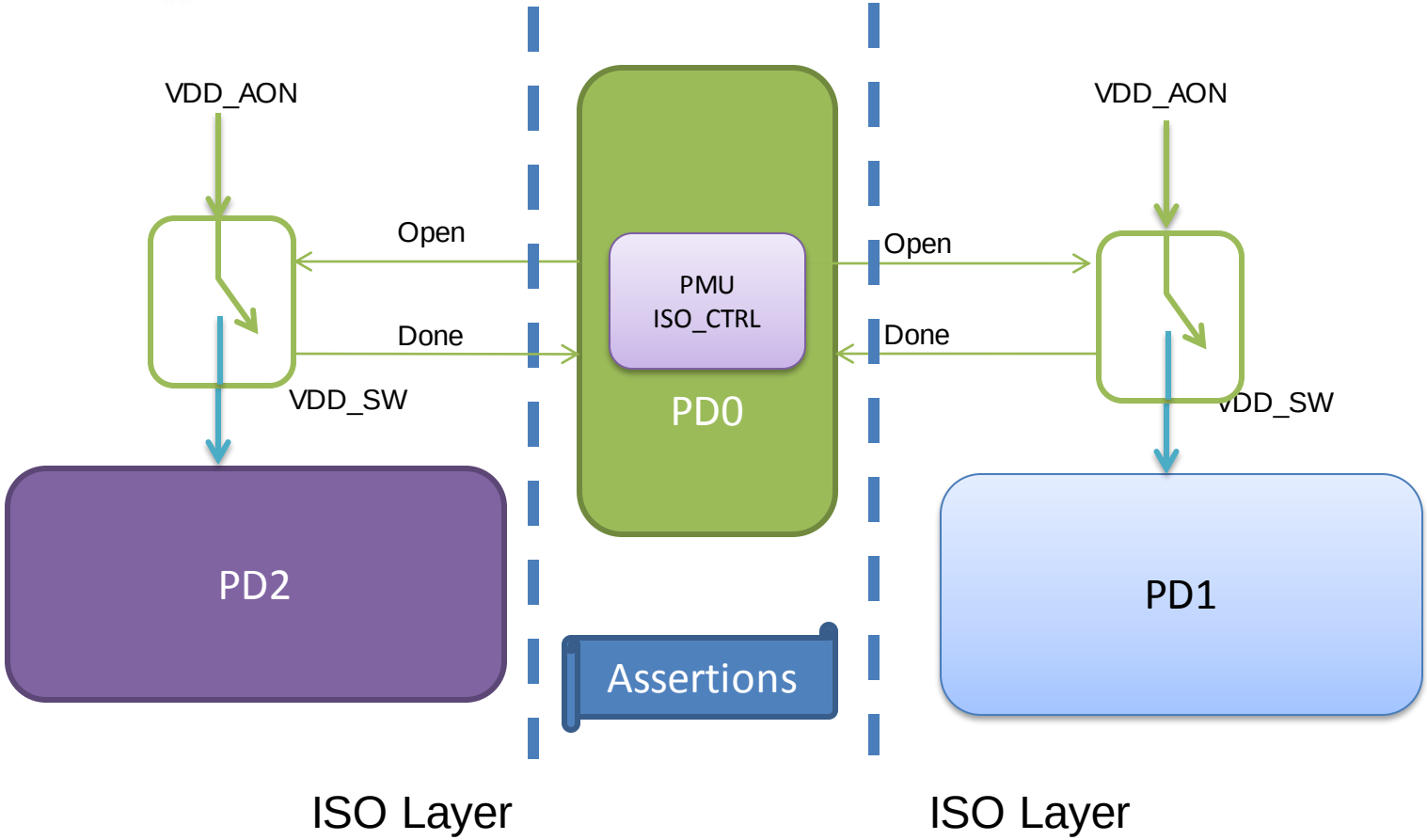
ACTUAL BEHAVIOR

1. De-assert Shutoff
2. Remove Isolation
3. Shutoff Asserted
4. No Isolation

Bug in PMU resulted in unwanted transition on shutoff signal causing corruption inside AON domain.

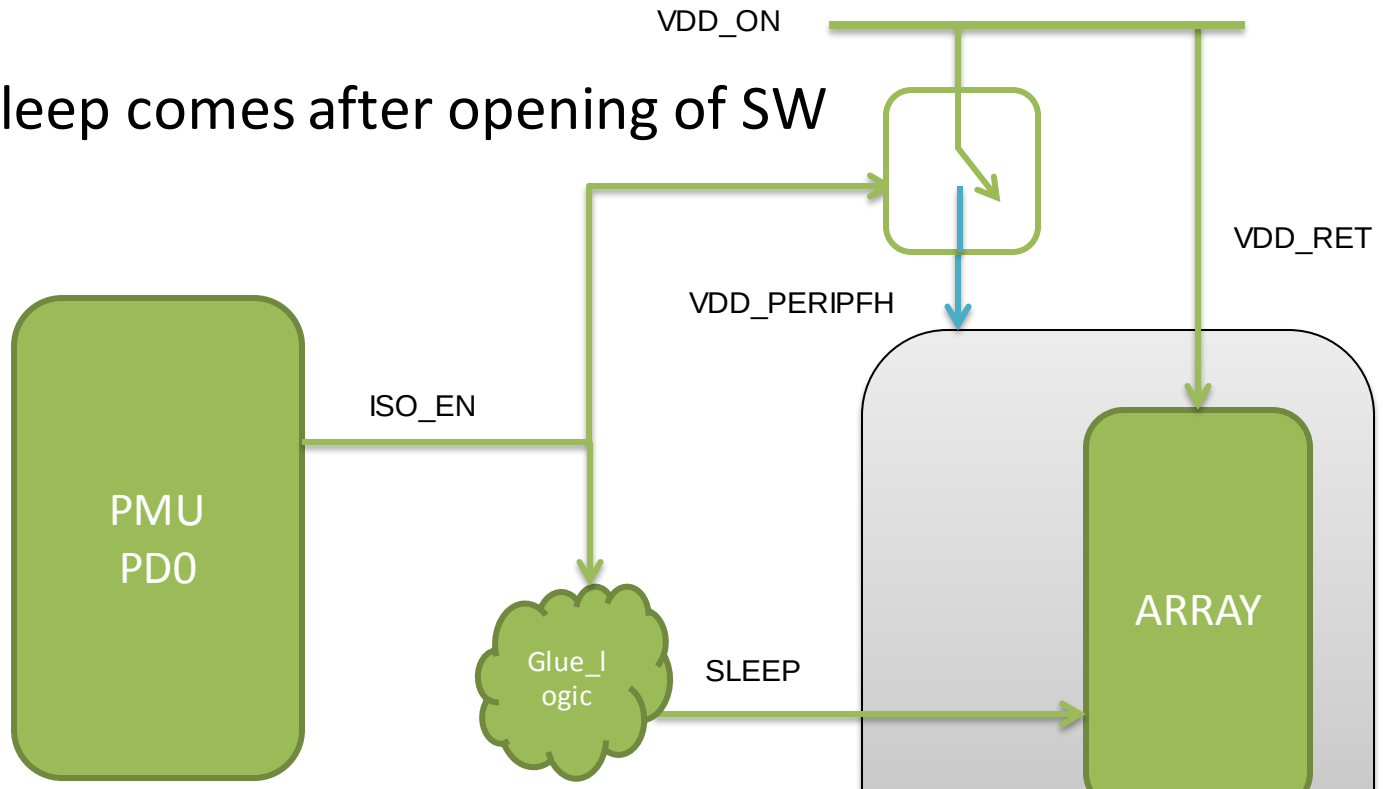
Low Power Mode Exit Protocol Issue

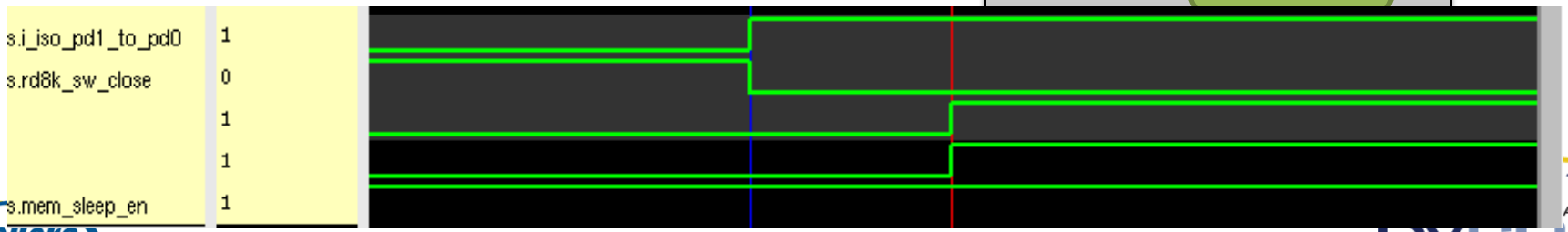
- PMU waiting for SW Done from PD2_PERF SW only & removes Isolation before PD_LP SW done



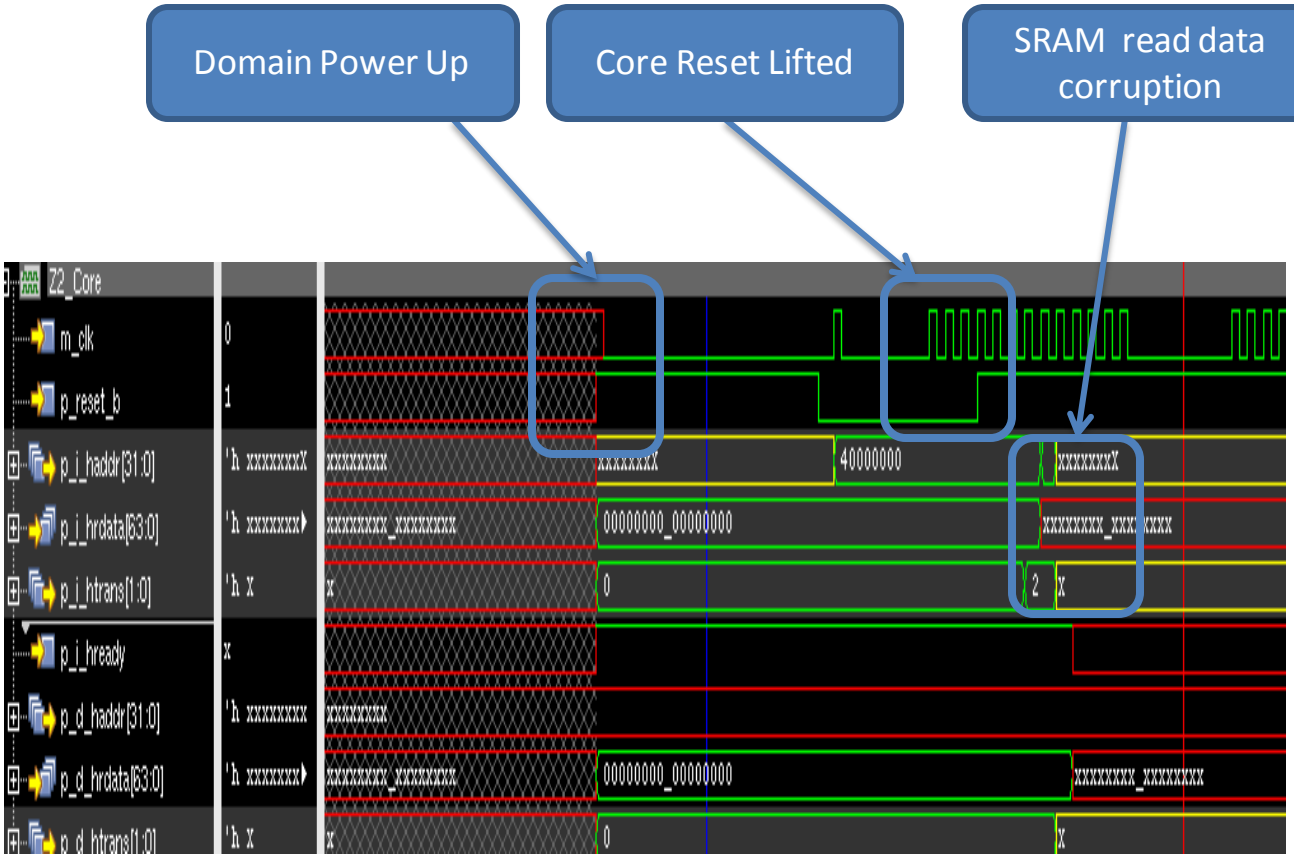
Macro Model Low Power Entry Protocol Issue

- SRAM Sleep comes after opening of SW





Memory Retention Issue



Intended Behavior

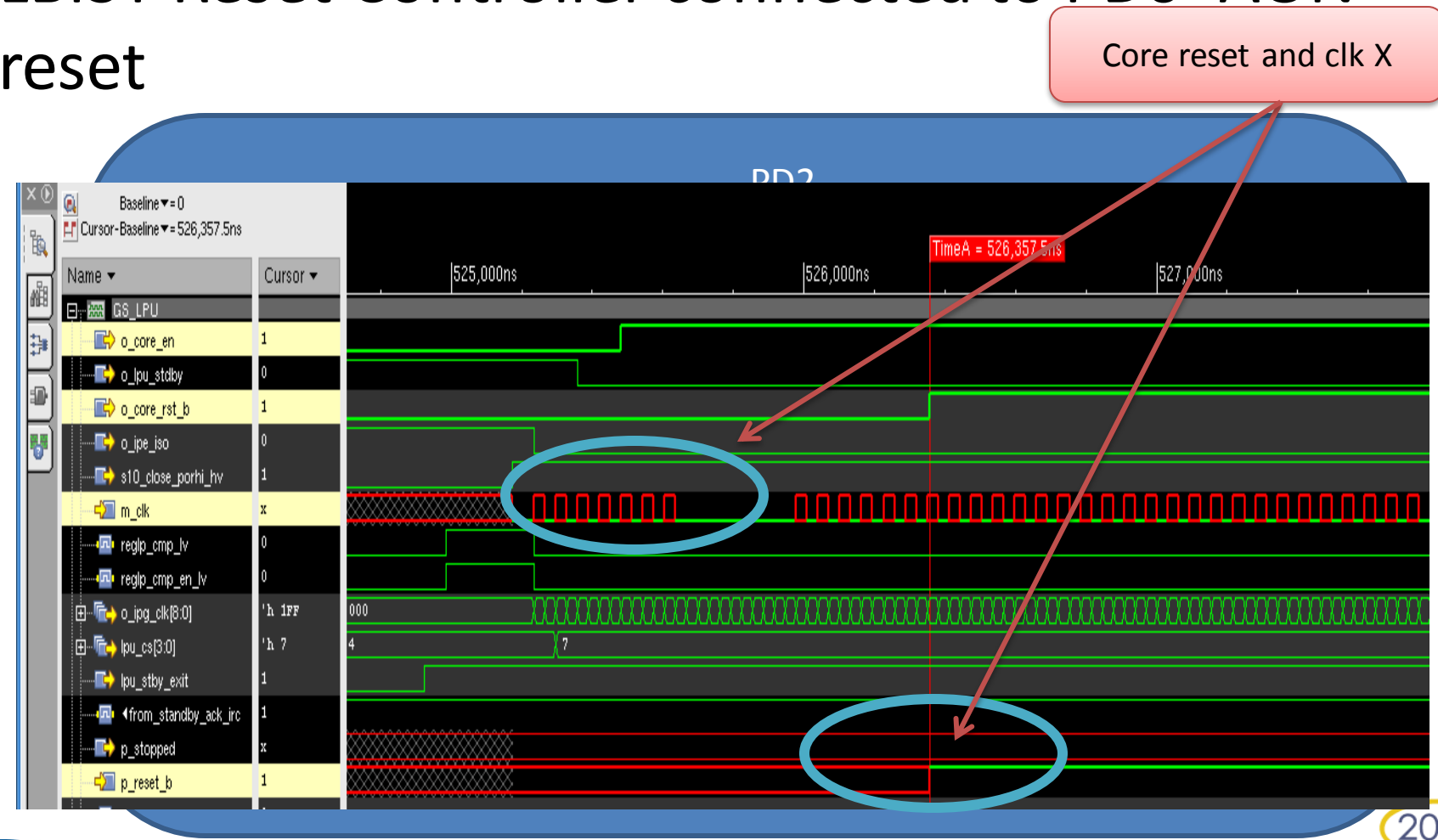
1. Domain power up
2. Core bring up via reset
3. Core to boot from SRAM after power up

Actual Behavior

1. Domain power up
2. Core bring up via reset
3. Boot up fails due to SRAM not being retained

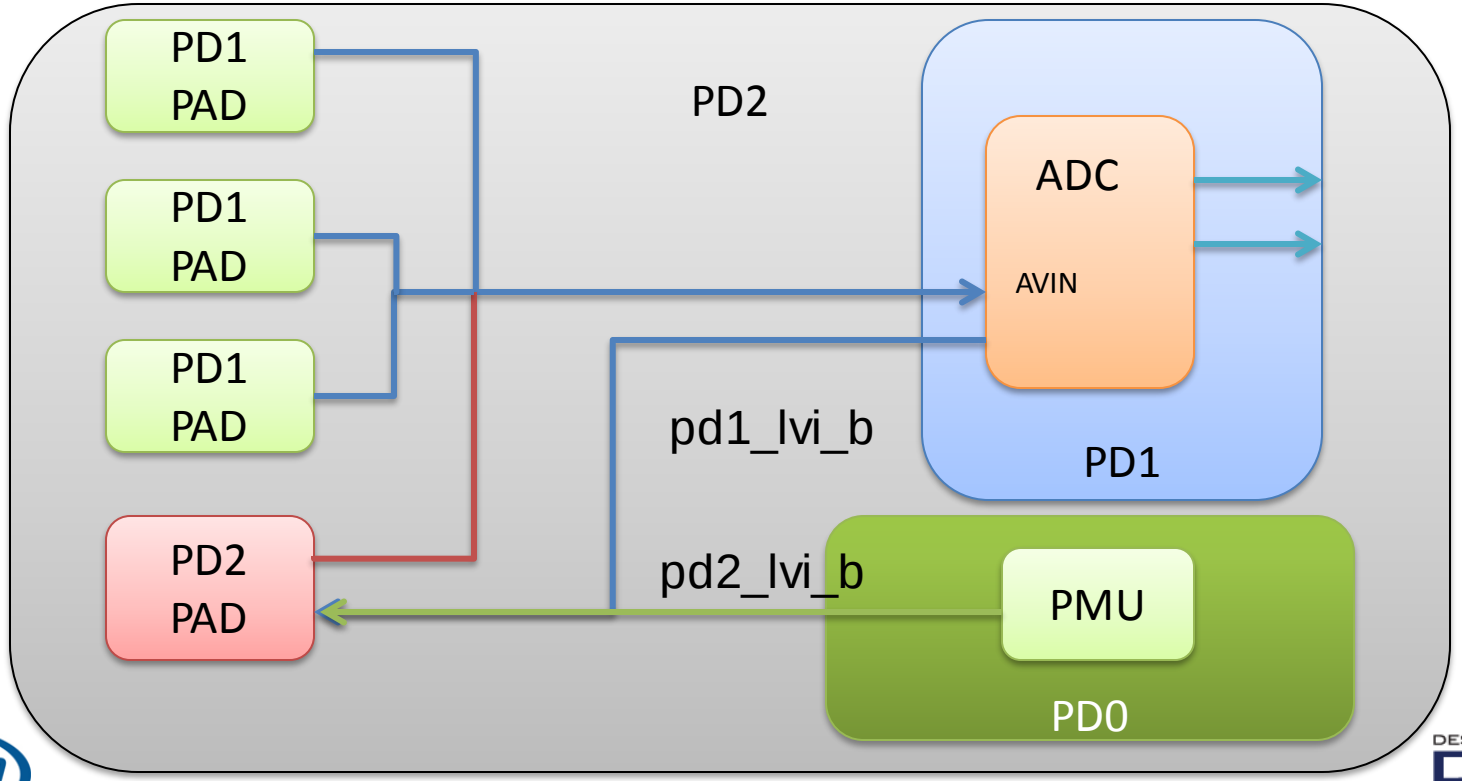
Power Up Issue – Incorrect Reset Mapping

- LBIST Reset Controller connected to PD0 AON reset



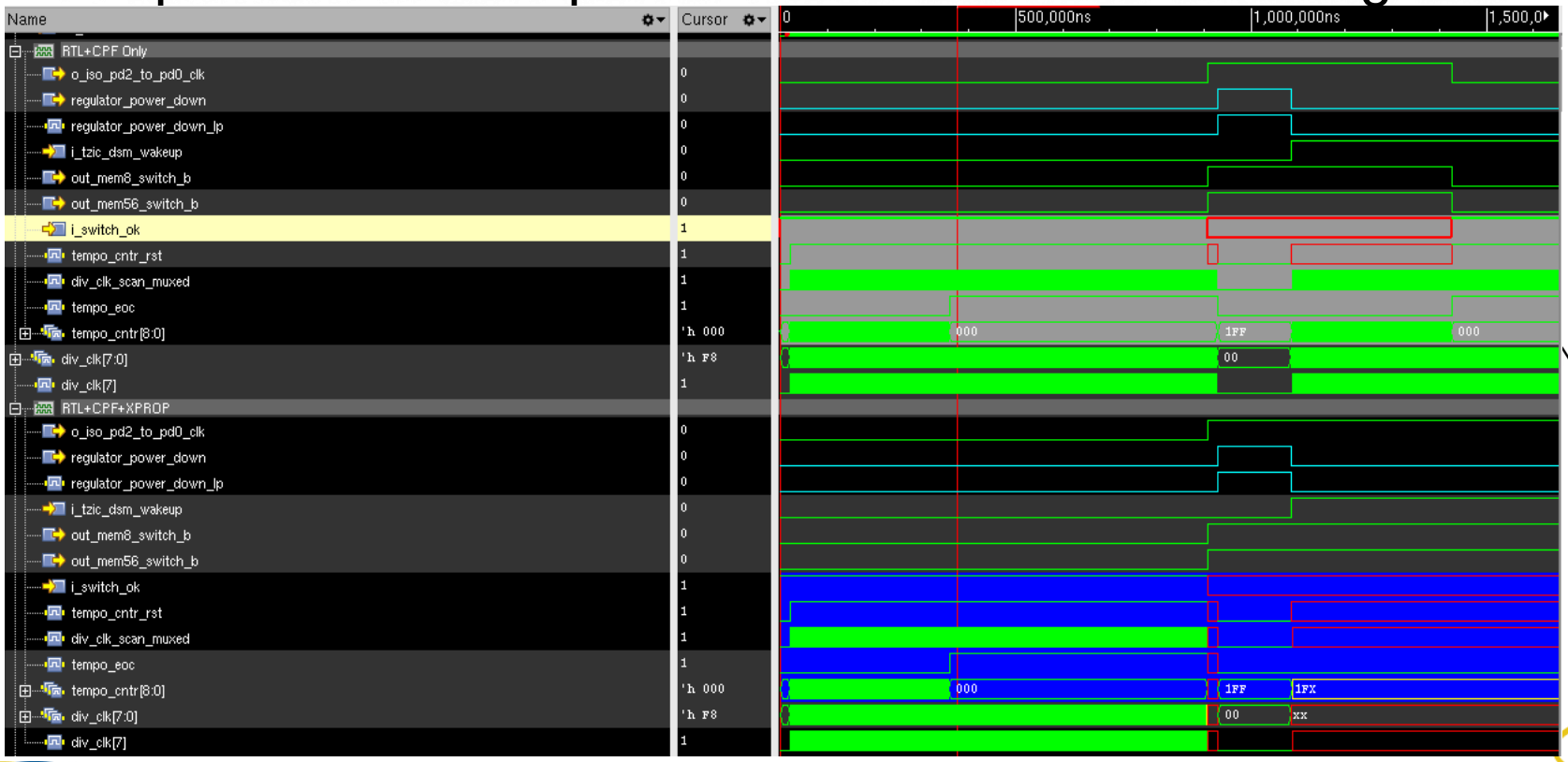
Analog IP Low Power Mode issue - PMU & ADC

- Transmission Gates of PD2 and PD1 pads shorted at SOC Level for connection to AVIN
- ADC alive channels unusable due to PD2 Pads' analog reset is connected to PD1 ADC voltage ok reset.



X-Optimism Issues : Low Power Mode Exit Deadlock

- Dead lock between SRAM switch Open signal and Isolation disable
- X-optimism in Low power mode exit code enabling exit

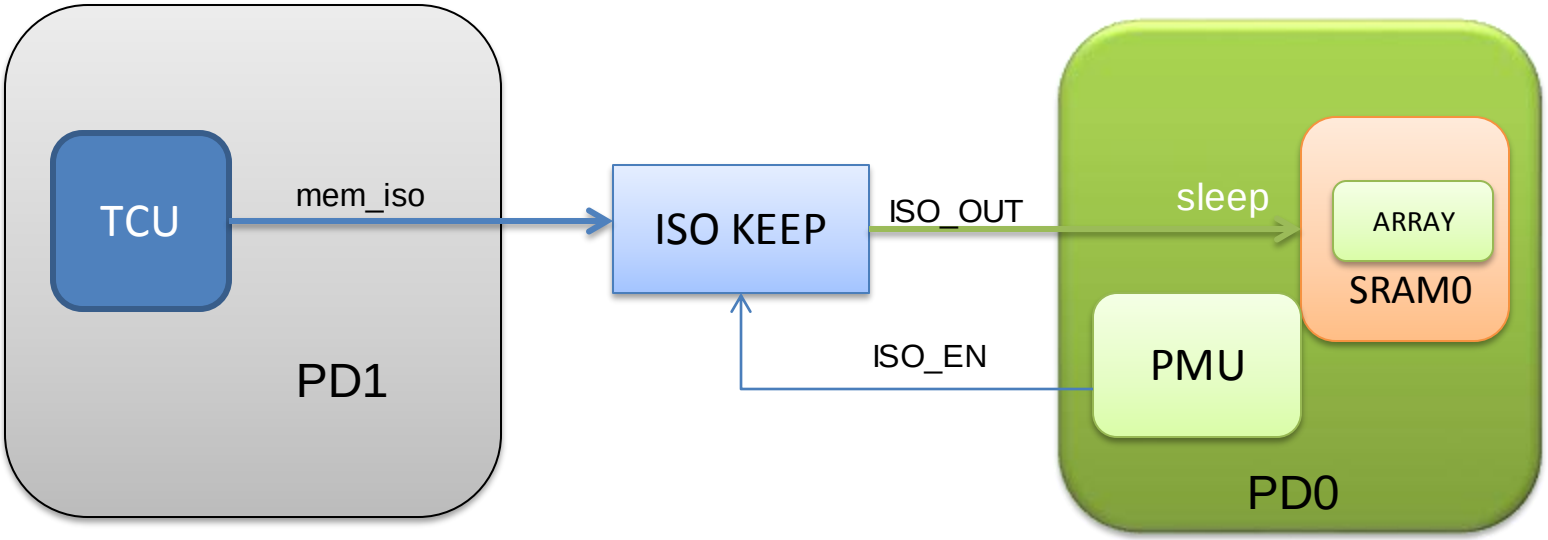


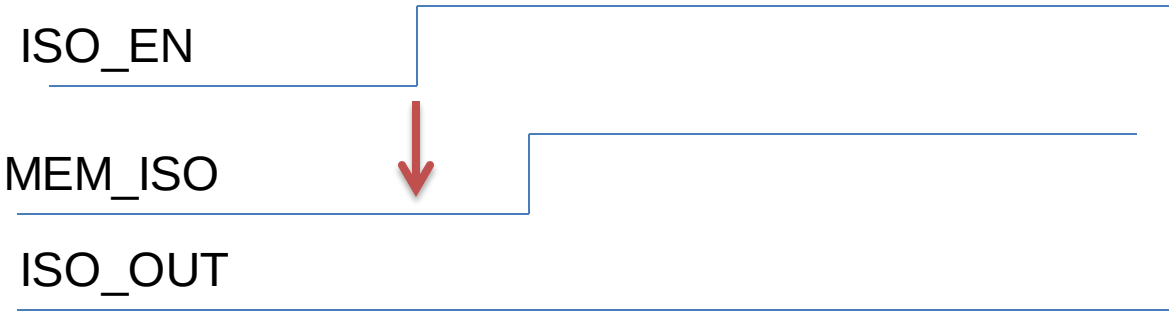
CPF Enabled Gate Level Simulations

- Gate Level netlist Simulation with Worst Case timing SDF and CPF to verify low power mode operation and power mode transitions
- Low Power Cells like level shifters, isolation and state retention cells defined in CPF file
- Enables final checking of timing and functionality related to Low Power Control Sequence and control signals for key Low Power tests
- Low Power Aware X-Prop simulation mitigates risk of missing issues due to X-optimism due to less number of tests run in CPF enabled Gate level Simulations

Worst Case Timing induced Issue

- ISO EN asserted prior to ISO Cell Input – Wrong Value latched





Future Scope

- IFV based verification of Low Power Control Logic
- Using Emulation Platform for System-level low-power verification and Dynamic Power Analysis to calculate average and peak power

Conclusion

- Presented a comprehensive Low Power Verification Methodology and Illustrated real low power design issues which we were able to catch using this flow
- Illustrated hazards of X-optimism in traditional Low Power RTL simulations and introduced X-Prop flow along with user assertions to mitigate these issues

References

- Michael Keating,David Flynn,Robert Aitken,Alan Gibbons,Kaijian Shi, Low Power Methodology Manual For System-On-Chip Design. Mill Valley, Springer.
- Neyaz Khan, Yaron Kashai , From Spec to Verification Closure: a case study of applying UVM-MS for first pass success to a complex Mixed-Signal SoC design . DVCON 2012
- Stuart Sutherland, I'm still in love with my X! . DVCON 2013
- Luke Lang , Christina Chu, New Challenges in Verification of Mixed-Signal IP and SoC Design. DVCON 2012

Thanks !

Questions & Answers

Abhinav.Nawal@freescale.com

Gaurav.Jain@freescale.com

Joachim.Geishauser@freescale.com

accellera SYSTEMS INITIATIVE

2014 DESIGN AND VERIFICATION DVCON CONFERENCE AND EXHIBITION EUROPE

Formal Specification of Power Intent in CPF

TESTBENCH.CPF

```
Set_design  testbench –testbench
Set_instance top
include DESIGN.cpf
Set_sim_control –action  power_up_replay   -instances {} ..
Create_assertion_control  -type reset –domain PD1 –assertions {
} ..
```

DESIGN.CPF

```
Set_design chip_top

Create_power_domain –name  PD2  –default           -
shutoff_condition { !sw_pd2_perf}
Create_power_domain –name  PD1  –instances { ..}        -
shutoff_condition { !sw_pd_lps}
Create_power_domain –name  PD0  –instances {...}

create_isolation_rule –name pd2_pd1 –from pd2_perf  -to
pd_aon   -isolation_condition {pmu/iso_pd2_to_pd1}

Create_nominal condition –name ON_v12 –voltage 1.2v
```

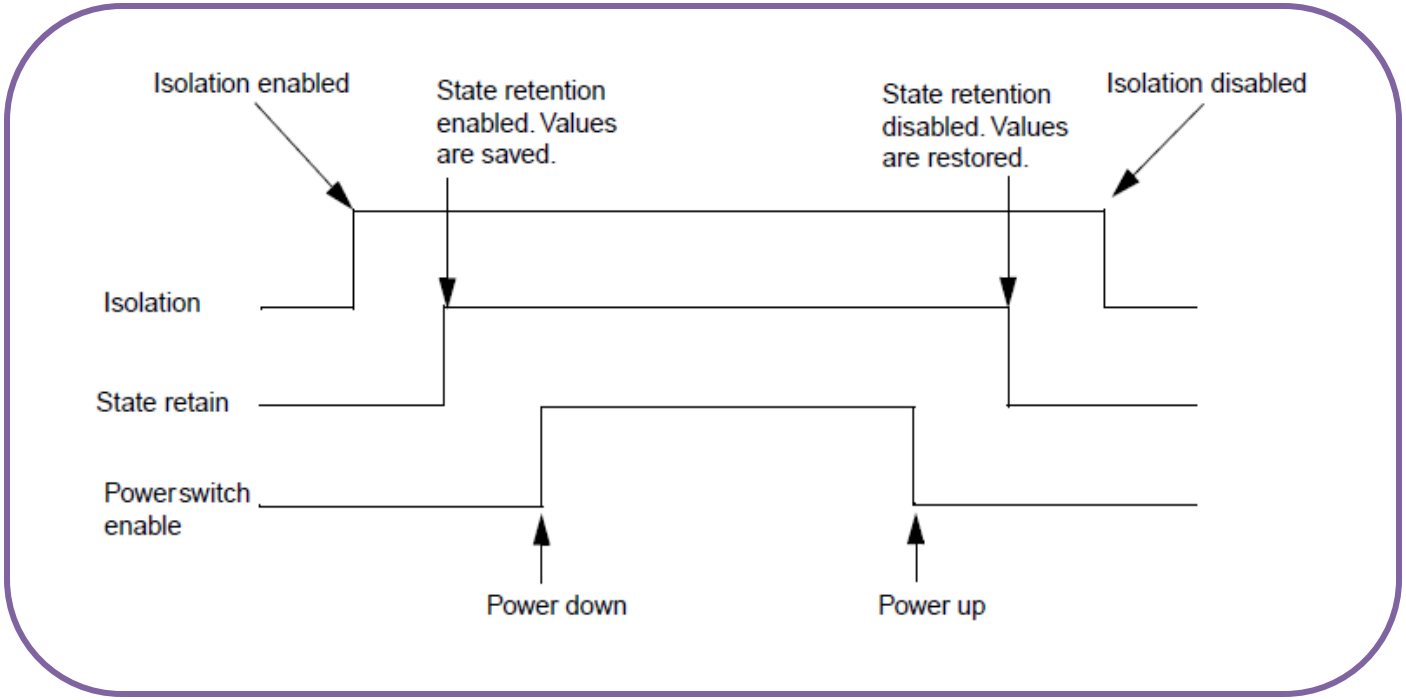
```
Include flash_macro.cpf
Include sram_macro.cpf
Include power_switch_macro.cpf
Include ADC macro.cpf
Include  macro_instance_mapping.cpf

Creat_power_modes   -name RUN –domain_conditions
{PD2_PERF@ON_v12   PD_LPS@ON_v12 .....}

Create_mode_transition – name  -from RUN –to STBY
 –start_condition {}

Set_instance  pd2_perf/flash   -domain_mapping {VDD_3.3
VDDF_33} {VDD_1.2SW VDDF_SW}
```

MACRO.CPF

```
Set_macro_model   FLASH

Create_power_domain –name VDD_3.3 –boundary_ports
{lp_en lp_ack }

Create_power_domain –name  VDD_1.2SW –boundary_ports
{reset clk  wdata  ...}

End_macro_model
```

Automatic Low Power Assertions from CPF

- Checks for Shutoff, Isolation, Retention Controls are not X or Z
- Checks for correct Power Down and Power UP Sequences



Automatic Low Power Intent Coverage

```
Instance name: ALPU_MODEL_VCOMP.coverage.domains.DOMAIN_VDD_PD1_SW.shutoff
Type name: alpu_cntl_sig_cov
File name: /project/calypso6m_verif_nbk1/users/abhinav/Cal6M_Cut2.0_Rel01.06.00/cut2_R06.00TB06.00/work/top_rt
A_libs/worklib/.alpu/cpf_cov.svp
Number of covered cover bins: 4 of 4
Number of uncovered cover bins: 0 of 4

Name                                  Average, Covered Grade          Line   Source Code
-------------------------------------------------------------------------------------------------
cg_control_sig                        100%, 100% (4/4)                35     covergroup lps_cg_control_sig @(trigger);
|--cntl                               100% (4/4)                      38     cntl: coverpoint control_signal
|  |--low                             100% (358/1)                    65     bins low= {0};
|  |--high                            100% (228/1)                    66     bins high= {1};
|  |--rising                          100% (228/1)                    67     bins rising = (0=>1);
|  |--falling                         100% (228/1)                    68     bins falling = (1=>0);

Instance name: ALPU_MODEL_VCOMP.coverage.domains.DOMAIN_VDD_PD1_SW.pd_state
Type name: ALPU_COV_TOP1_chip_top914_PD_VDD_PD1_SW_state
File name: /project/calypso6m_verif_nbk1/users/abhinav/Cal6M_Cut2.0_Rel01.06.00/cut2_R06.00TB06.00/work/top_rt
A_libs/worklib/.alpu/cpf_cov.svp
Number of covered cover bins: 2 of 2
Number of uncovered cover bins: 0 of 2

Name                                  Average, Covered Grade          Line   Source Code
-------------------------------------------------------------------------------------------------
power_state                           100%, 100% (2/2)                173177 covergroup lps_power_state @(trigger);
|--pwr_state                          100% (2/2)                      173180 pwr_state: coverpoint pd_state
|  |--auto[ON]                        100% (130/1)                    173180 pwr_state: coverpoint pd_state
|  |--auto[OFF]                       100% (228/1)                    173180 pwr_state: coverpoint pd_state


Instance name: ALPU_MODEL_VCOMP.coverage.power_mode_control_group.PMCG_chip_top
Type name: ALPU_COV_TOP1_chip_top914_PMCG_chip_top
File name: /project/calypso6m_verif_nbk1/users/abhinav/Cal6M_Cut2.0_Rel01.06.00/cut2_R06.00TB06.00/work/top_rtl_regression.b41776.13_10_08_16_07_45_
A_libs/worklib/.alpu/cpf_cov.svp
Number of covered cover bins: 2 of 23
Number of uncovered cover bins: 21 of 23

Name                                Average, Covered Grade          Line   Source Code
-------------------------------------------------------------------------------------------------
pmcg                                100%, 100% (23/23)                173982 covergroup lps_pmcg @(trigger);
|--pwr_mode                         100% (7/7)                      173985 pwr_mode: coverpoint pmode
|  |--pwr_mode[RUN]                 100% (86/1)                     174012 bins pwr_mode[] = {[$:$]};
|  |--pwr_mode[LPU_RUN]             100% (1/1)                      174012 bins pwr_mode[] = {[$:$]};
|  |--pwr_mode[LPU_STDBY]           100% (1/1)                      174012 bins pwr_mode[] = {[$:$]};
|  |--pwr_mode[LPU_STDBY_64K_RET]   100% (1/1)                      174012 bins pwr_mode[] = {[$:$]};
|  |--pwr_mode[LPU_STDBY_128K_RET]  100% (1/1)                      174012 bins pwr_mode[] = {[$:$]};
|  |--pwr_mode[LPU_STDBY_256K_RET]  100% (1/1)                      174012 bins pwr_mode[] = {[$:$]};
|  |--pwr_mode[STANDBY_256K_RET]    100% (30/1)                     174012 bins pwr_mode[] = {[$:$]};
|--pmode_trans                       100% (16/16)                    174015 pmode_trans: coverpoint pmode
|  |--RUN_2_LPURUN                   100% (1/1)                      174042 bins RUN_2_LPURUN = (RUN => LPU_RUN);
|  |--LPURUN_2_RUN                   100% (1/1)                      174043 bins LPURUN_2_RUN = (LPU_RUN => RUN);
|  |--RUN_2_STANDBY_256K_RET         100% (1/1)                      174044 bins RUN_2_STANDBY_256K_RET = (RUN => STANDBY_256K_RET);
|  |--STANDBY_256K_RET_2_RUN         100% (1/1)                      174045 bins STANDBY_256K_RET_2_RUN = (STANDBY_256K_RET => RUN);
|  |--LPURUN_2_LPU_STANDBY           100% (1/1)                      174046 bins LPURUN_2_LPU_STANDBY = (LPU_RUN => LPU_STDBY);
|  |--LPURUN_2_LPU_STANDBY_64K_RET   100% (1/1)                         174047 bins LPURUN_2_LPU_STANDBY_64K_RET = (LPU_RUN => LPU_STDBY_64K_RET);
|  |--LPURUN_2_LPU_STANDBY_128K_RET  100% (1/1)                         174048 bins LPURUN_2_LPU_STANDBY_128K_RET = (LPU_RUN => LPU_STDBY_128K_RET);
|  |--LPURUN_2_LPU_STANDBY_256K_RET  100% (1/1)                         174049 bins LPURUN_2_LPU_STANDBY_256K_RET = (LPU_RUN => LPU_STDBY_256K_RET);
|  |--LPU_STANDBY_256K_RET_2_LPU_RUN  100% (1/1)                        174050 bins LPU_STANDBY_256K_RET_2_LPU_RUN = (LPU_STDBY_256K_RET => LPU_RUN);
|  |--LPU_STANDBY_128K_RET_2_LPU_RUN  100% (1/1)                        174051 bins LPU_STANDBY_128K_RET_2_LPU_RUN = (LPU_STDBY_128K_RET => LPU_RUN);
|  |--LPU_STANDBY_64K_RET_2_LPU_RUN   100% (1/1)                        174052 bins LPU_STANDBY_64K_RET_2_LPU_RUN = (LPU_STDBY_64K_RET => LPU_RUN);
|  |--LPU_STANDBY_2_LPU_RUN           100% (1/1)                      174053 bins LPU_STANDBY_2_LPU_RUN = (LPU_STDBY => LPU_RUN);
|  |--LPU_STANDBY_256K_RET_2_RUN      100% (1/1)                      174054 bins LPU_STANDBY_256K_RET_2_RUN = (LPU_STDBY_256K_RET => RUN);
|  |--LPU_STANDBY_128K_RET_2_RUN      100% (1/1)                      174055 bins LPU_STANDBY_128K_RET_2_RUN = (LPU_STDBY_128K_RET => RUN);
|  |--LPU_STANDBY_64K_RET_2_RUN       100% (1/1)                      174056 bins LPU_STANDBY_64K_RET_2_RUN = (LPU_STDBY_64K_RET => RUN);
|  |--LPU_STANDBY_2_RUN               100% (1/1)                      174057 bins LPU_STANDBY_2_RUN = (LPU_STDBY => RUN);
```

X Optimism Code examples & LP X Propagation Solution

- X-Optimism Code Examples

If –else statement

```
reg sel;
If (sel)
 out = a;
else
 out = b;
```

Case with default

```
reg[1:0] sel;
case(sel)
2'b00: out = a;
2'b01: out = b;
default: out = c;
endcase
```

Asynchronous Reset

```
always @(posedge clk or
posedge rst)
begin
if (rst)
q = 0;
else
q = a;
end
```

- LP X Propagation Solution feature
  - X Propagation analysis as per Power Domains identified in CPF/UPF
  - X-Propagation is automatically suspended during power domain shut down based on CPF/UPF shutoff condition
  - X-Propagation from a switchable domain is automatically enabled just after domain is turned ON

CPF Enabled Metrics Driven Flow for Low Power Verification