Functional Design Bugs Detected With Low Power Verification

Alexander Kryvoshaev

Qualcomm
Markham, Canada

www.qualcomm.com

ABSTRACT

Power aware verification is a new field of design verification, which is currently used mostly for finding missing clamp cells. This paper classifies functional bugs caught within a power aware flow. An emphasis is placed on how to apply the latest VCS Native low power (NLP) simulation technologies to improve power verification while highlighting recent advances in NLP. Numerous creative examples, demonstrating the efficiency of the NLP flow, are introduced. Special attention is paid to verification gaps, which can be detected with NLP flow. Thus, NLP flow will be introduced as a tool for finding verification inaccuracies along with design functional bugs.

Table of contents

Table of figures

Table of tables

1. Introduction and background

In the last couple of years saving static power has become one of the biggest challenges of the semiconductor industry. Decreasing the transistor channel length increases the dynamic power consumption and decreases the static power.

VCS NLP, the power aware tool from Synopsys is a powerful one available on the market. From functional verification prospective, the tool forces 'X's on signals of collapsed blocks. At the first glance, injecting 'X's does not seem to make any difference. This is because the collapsed logic is not verified by functional tests anyway. After deeper consideration, forcing 'X's on internal signals of the collapsed logic is crucial for complete functional verification.

This paper considers different types of functional bugs to be caught with power aware (PA) flow, which would otherwise be missed with regular simulations.

For the rest of the paper the words collapse and power gated will be used interchangeably.

1.1 Benefits of saving power

High capacity batteries are expensive and are not widely used. This makes the semiconductor companies look for various ways to save power. Also, usage of too many transistors per area unit causes cooling problems, which are difficult to be resolved. Below is a table, summarizing these problems.

Battery limited capacity	Heat problem
• Batteries have a limited capacity, which is not sufficient for contemporary needs • Battery technology is unlikely to get improved in the near future. • The existing improvements are expensive.	• Number of transistors per area unit increases from year to year. This leads to an increase in temperature, which might burn the circuit. • Cooling systems are not sophisticated enough. • CPUs with irresponsible power consumption may burn the board.

Table 1 - Battery capacity issues along with cooling technology problems

1.2 Increasing Design Area

Increasing the design area would resolve some of the problems specified in the previous section. However, this would lead to additional noise and timing issues. The table below summarizes the advantages and disadvantages of increasing the area.

Area increase would resolve:	Side effects:
• Less transistors per area unit, which means less heat.	• Area increase doesn't resolve the limited battery capacity issue. • Increase of the number of routing paths would cause noise issues. • Noise is being resolved by adding repeaters. • Additional repeaters would lead to increased clock cycle, which means smaller circuit frequency. • Splitting of long paths into cycles by additional latches leads to increased latency.

Table 2 – Advantages and disadvantages of increasing design area

These and other reasons led the industry to move in a direction of power gating and developing power maps along with logic design.

1.3 Power Gating

Clock gating techniques have provided excellent results for older processes where dynamic power was the source of power consumption in the chip. Clock gating techniques have decreased considerably mainly due to improvements made in transistor technology such as shrinking the size of transistors. These days static power dissipation exceeds dynamic power dissipation. The best way to save static power is to power gate them. During power gating or design collapse, the power supplies Vdd and Vss are disconnected which essentially shuts down the logic. Please refer to the figure below. This measure prevents static power wastage.

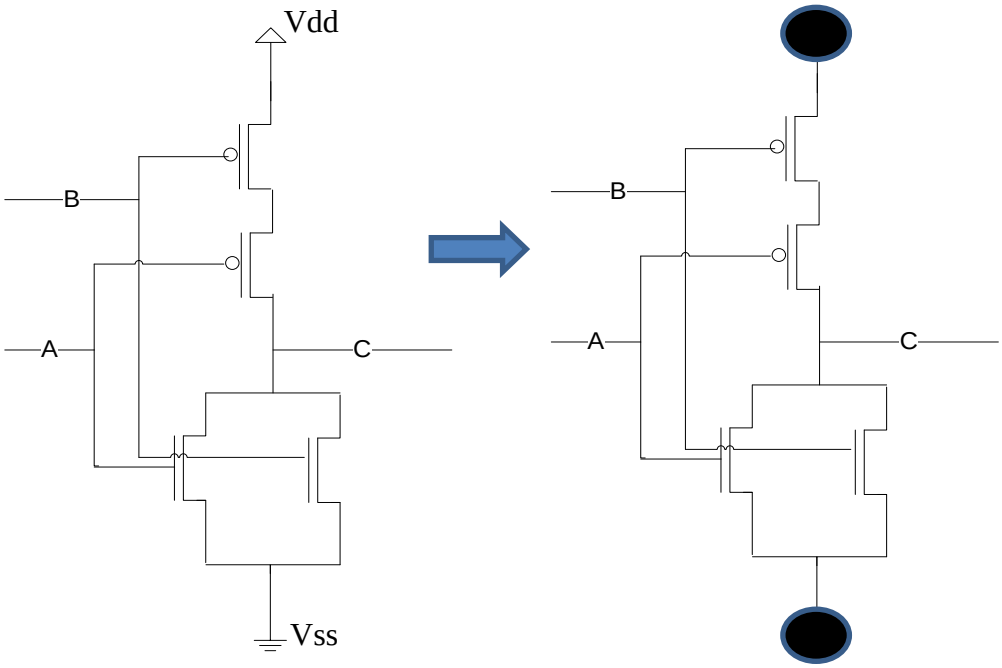


Figure 1 - Design Collpase/Power Gating

2. Power Aware Verification Flow

In a typical power aware verification flow we need to describe the power intent of the design and be able to simulate the power logic. The power intent of the design is described in a power intent or UPF (Unified Power Format) file. This file will typically describe the power domains, isolation cells, level shifters, power switches etc.

The other aspect of power aware verification is to be able to simulate this UPF file. VCS NLP, the simulator, understands UPF files and maps the power intent to virtual logic in the RTL, which is then used during simulation.

The next few sections will get into some more details of what is mentioned above.

2.1 Power Aware Simulator Behaviour

For the blocks that are collapsed/power gated, VCS NLP will corrupt or inject X on all internal signals for the duration that block is collapsed.

The simulator behaves as follows:

- When the design is collapsed, the simulator forces all the internal signals to X
- On power recovery the internal signals are stopped from being forced to X.

However, they retain the X until the propagation of valid signals from inputs or reset values.

In order to prevent unknown signals from being propagated, it is common practice to use clamp cells. When enabled, clamp cells provide 1, 0 or retention value on the output ports. When clamp cells are not enabled they behave like a regular buffer. Using clamp cells is the best practice to isolate collapsed blocks from the rest of the design. When you add clamp cells you are essentially clamping the outputs of a power domain. This prevents X's to propagate to the downstream logic.

2.2 'X' Injection in Design

One may ask the question, why does it matter whether these internal signals are 'X' or any other value? The most obvious answer to this question would be to verify if 'X' values are not propagating out of the collapsed block. Missing clamps cells could potentially lead to 'X' propagation, which would not be caught without a power aware flow.

Checking for clamp cells in a design does not justify having a robust power aware verification flow. Verification of clamp cells could be performed with perl scripts.

The figure below shows an example of a power domain with clamp cells.

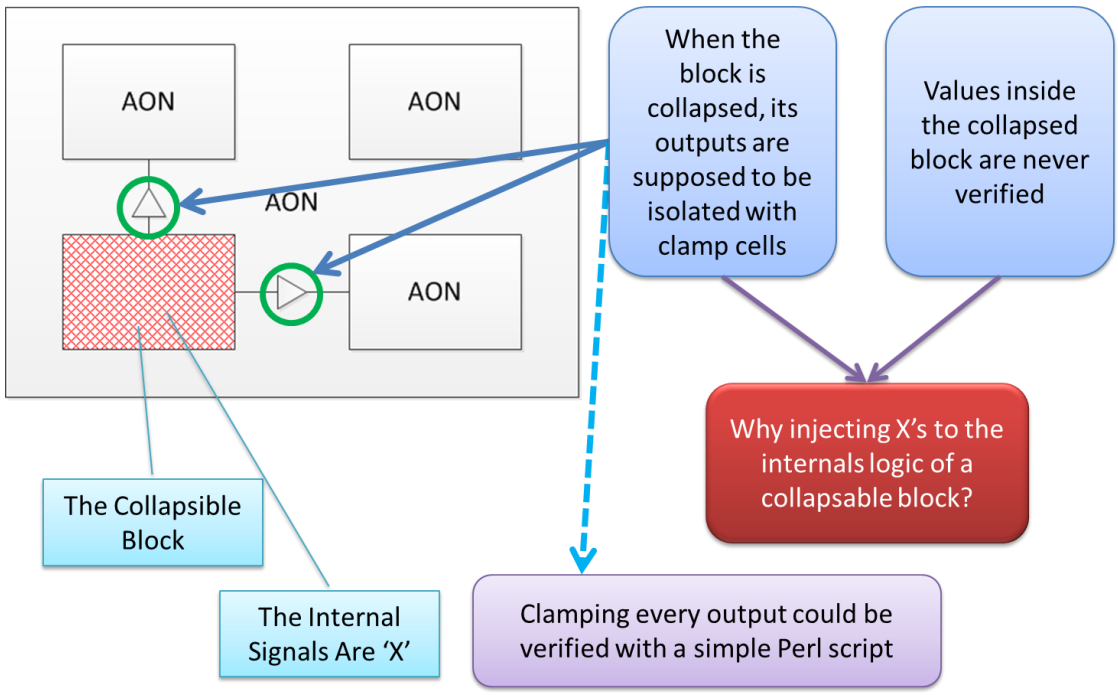


Figure 2 - Power Domain Example

2.3 Power recovery sequence

The recommended sequence when a power domain recovers is as follows:

- **Turn power ON**
  - Power ON wakes up all the flip-flops, however, their state is unknown and therefore X
- **Reset all flops**
  - Flip-flops get their default value. All the 'X' signals are cleared
- **Turn OFF clamp cells**
  - Turning off the clamp cells means switching them to a buffer mode

If reset and clamp values are good and if these 3 steps are done correctly, the values of internal signals during power collapse are not important. Therefore, insertion of X values by NLP is irrelevant.

This means that most of the bugs found with Power Aware are a result of a verification gap.

The next figure describes the power recovery sequence:

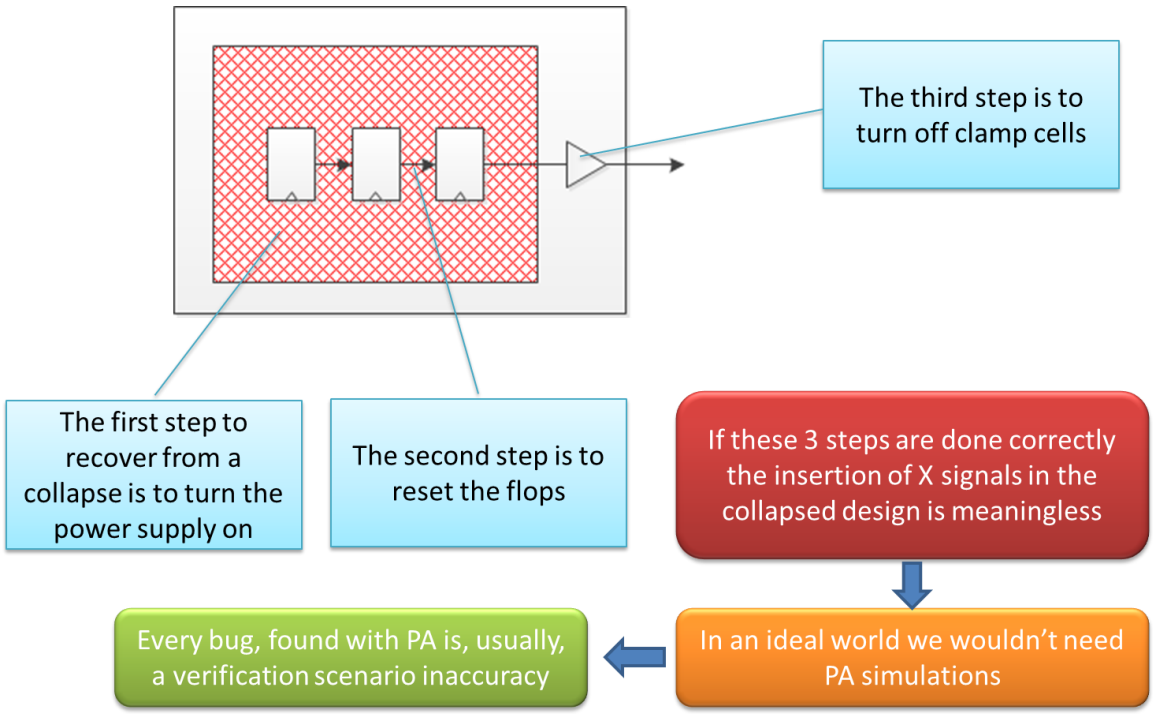


Figure 3 – Power Recovery Sequence

3. Bugs Identified With Power Aware Flow

This section describes the type of bugs caught with a power aware flow:

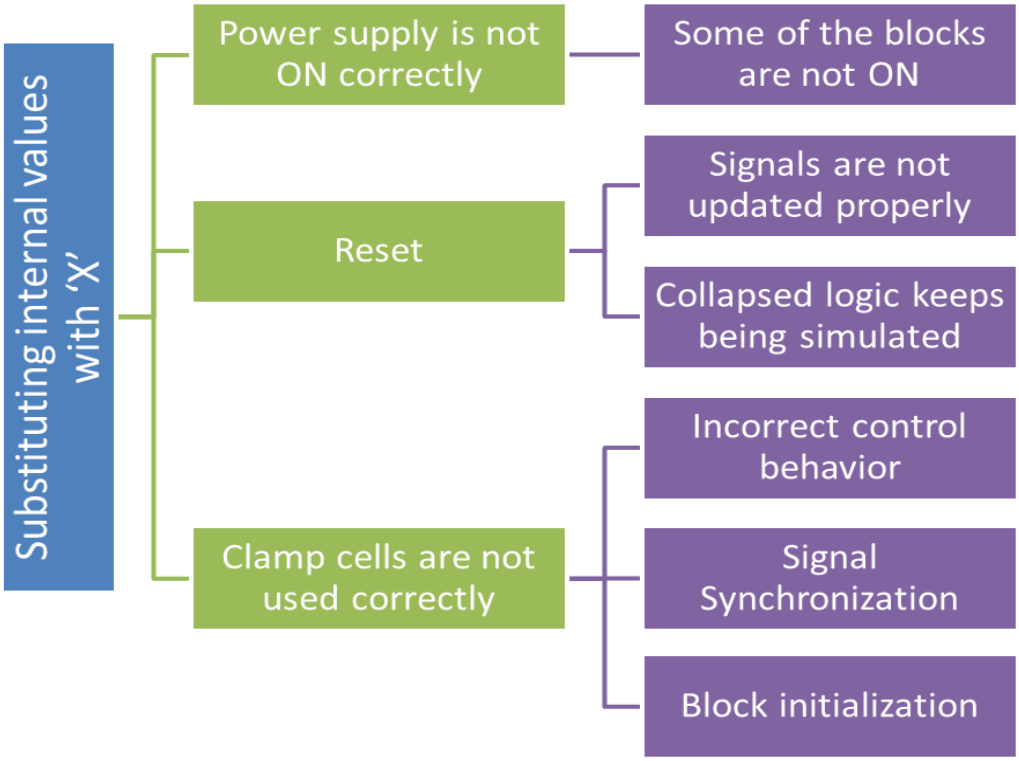


Table 3 - Classification of bugs to be caught with PA flow

1. Power supply not ON:
   a. Some of the blocks are not ON at all. This bug can't be caught by non-power aware simulation.

2. Resetting flops inaccurately:
   a. In a power aware simulation, all internal signals/flops during power collapse are 'X'. If the reset on the flop was missed during power recovery, then in a power aware flow, the test would fail due to the flop being X. However, in a non-power aware simulation, the tests would still pass.
   b. If the reset on the flops was missed during power recovery, the logic which was supposed to be reset will continue to be simulated in a non-power aware simulation. Therefore, non-power aware functional tests can miss bugs.

3. Clamp cells:
   In this section we will discuss RTL cells, which infer isolation functionality.
   A simple example is AND gate with one input forced to 1 for buffer functionality and forced to 0 for zero clamp functionality. The combinatorial logic, involved in implementation of

isolation functionality, can be a part of the RTL design. In this case clamp/isolation cells are not required.

a. If the clamps are disabled prematurely, X will propagate
b. Clamps are disabled prematurely. In non-PA simulations logic keeps being simulated during the collapse. If the clamp cells were disabled after the reset, then the consequences of simulation of the collapsed logic would be eliminated by the reset. Because the clamps are disabled before the reset, some data gets transmitted in a time window between the clamp disabling and the reset.
c. With PA, initialization takes longer than without PA flow. Non-PA flow doesn't find this bug because of premature clamp disabling. If clamps were disabled on time then non-valid data, caused by long initialization, wouldn't come out.

The following table provides an overview of the examples, presented in this paper:

Table 4 - examples overview

Example	Reset	Clamps	Bug overview	Comments
1	Ok	Ok	Power is not ON. In non-PA simulation the logic will be functioning correctly.	When the power is OFF, the logic is not being simulated with PA flow (‚X' forcing)
2	No reset	Ok	The reset is not set correctly. The internal values after the collapse don't get the init values. In non-PA simulation the missing reset is not detected.	
3	Late reset	Ok	Performance issue. In non-PA simulation the collapsed logic keeps being simulated. Reset is supposed to eliminate the results of this „illegal" simulation during the collapse.	Clamp cells are disabled on time. However this happens before the reset, which is prematurely.
4	N/A	Early disable	‚X' propagation. There are no ‚X' values in non-PA simulations.	
5	Ok	Early disable	Signal synchronization. In non-PA simulation the collapsed logic keeps being simulated. The result, generated between the clamp disabling and the reset, is wrongly used.	
6	N/A	Early disable	Block initialization. Block initialization occurs faster with non-PA flow than with PA flow.	Examples 4,5 and 6 are different bug types, detected for the same recovery sequence inaccuracy.

3.1 Power Supply not ON

Example 1

Let us consider an example where in the power collapsed block is not turned on correctly. In the figure below, there are 4 blocks: A, B, C and D. The functionality of the blocks is as follows:

- A and B convert between different interfaces. The conversion latency is 3 cycles.
- A is not collapsible, B is collapsible.
- C merges every 2 consecutive transactions
- C gives arbitration to the first transaction, which is ready. If both transactions are ready in the same cycle then A gets the arbitration.
- C sends acknowledgement if the read succeeded.
- C is not being collapsed, but it doesn't read during the collapse
- A gets its initial information from an external interface one cycle before B. Therefore, without a collapse, A provides output one cycle before B every time.

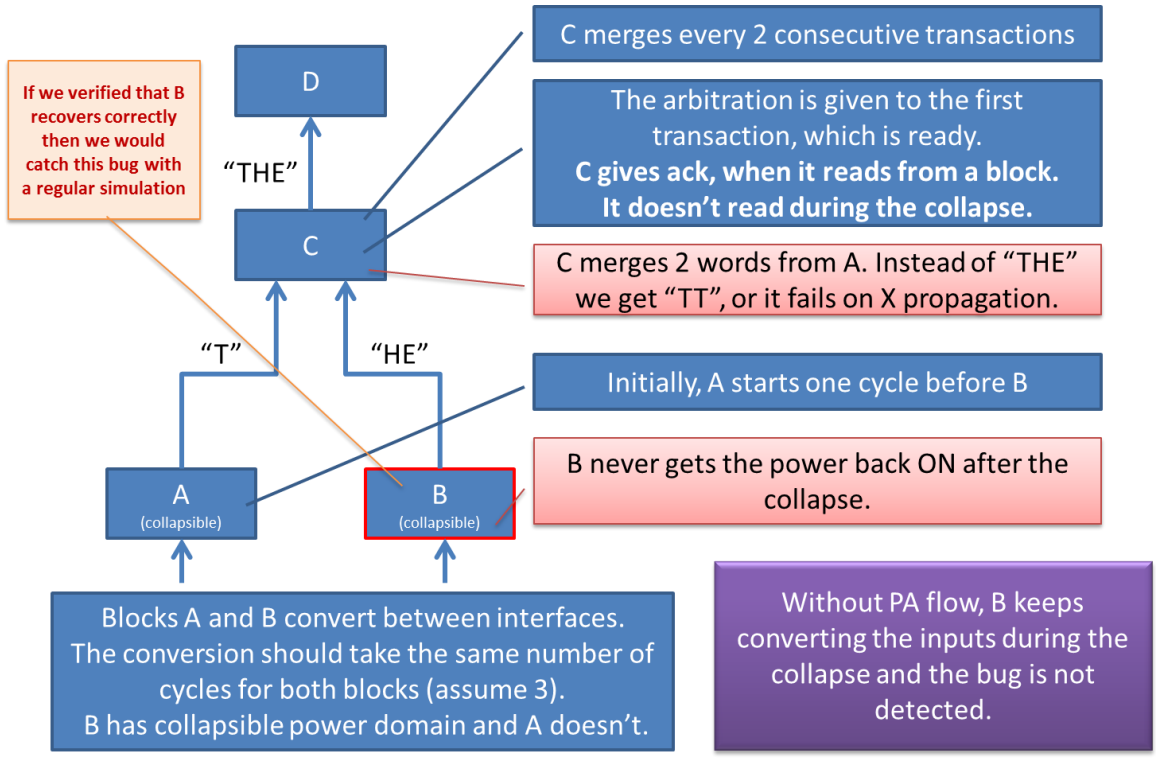


Figure 4 - Power Supply not ON

The bug:
Block B never powers ON after the collapse.

PA simulation:
The reason why this bug is caught by power aware is because during power collapse all the internal signals are ‘X’ and B will not produce any output.

Non - PA simulation:
In a non-power aware simulation, power collapse does not have any behavioral impact and the bug would be missed.

It is important to note that the power up/down sequence generator simply generates the signals while the low power simulation is the one that infers the power switch which helps simulate the actual behavior, hence making it possible to catch the bug.

3.2 Resetting Flops Inaccurately

Example 2

Consider a FIFO design with the following rules to be followed:

- A request has to be sent before a reset is applied. Acknowledgement is granted when FIFO is empty (read and write pointers are equal)
- FIFO has 2 pointers: read and write. The pointers are being reset by setting to 0.
- There are 2 flops in non-collapsible domain. POR of FF0 is 0, POR of FF2 is 1.
- After the first cycle the output of FF2 becomes to be 0.
- Output of FF2 is an input of the AND gate. This means that starting from the second cycle one of the inputs of AND gate is 0.
- When the second input of the AND gate is 0 and the reset input doesn't have any impact on the output of the gate.

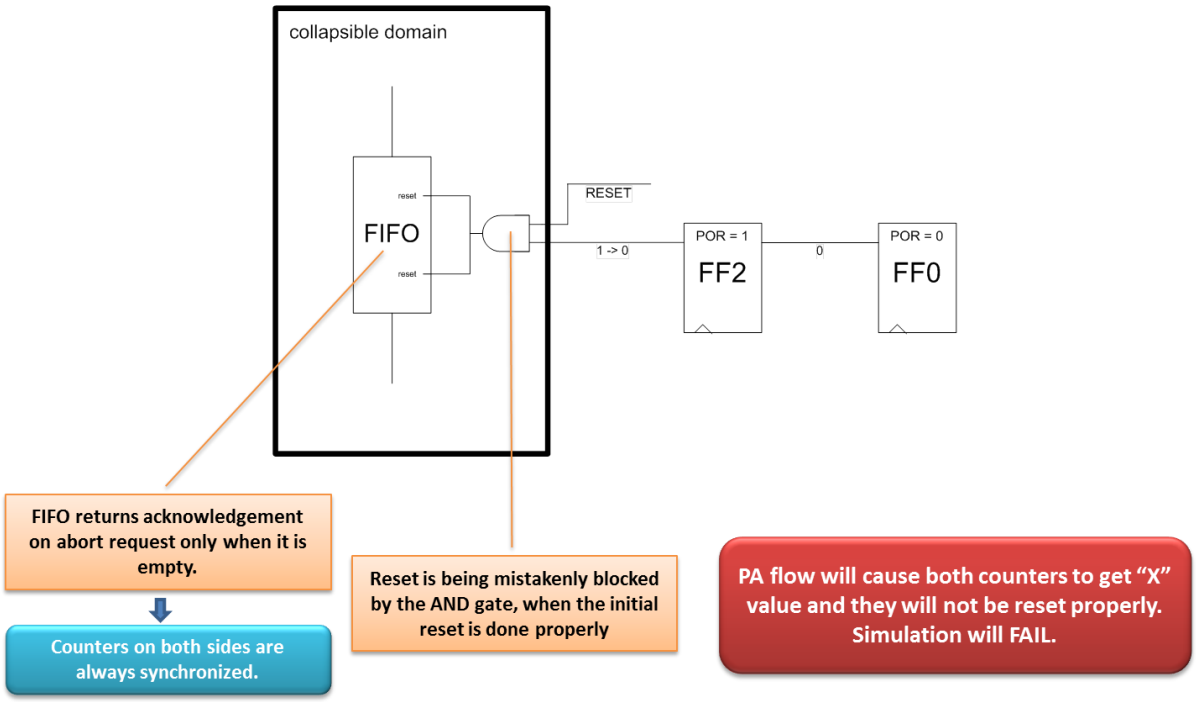


Figure 5 - Reset Bug Example

The bug:
FIFO block is reset during the initial reset phase. When the collapsible domain recovers, the reset is blocked by the AND gate logic. FF2 output is 0 and the AND gate logic propagates 0, thereby never being able to reset the FIFO pointers.

PA simulation:
During power collapse, the pointers are set to 'X'. When this domain recovers, not being able to reset the pointers makes them hold the 'X' value. This leads to a test failure.

Non-PA simulation:
In a non-power aware simulation, there is no power collapse. Hence, both pointers are always equal during reset. Therefore, setting of pointers to 0 is not required for correct functionality. Therefore, the test passes.

Example 3:

Consider the same example as in section 3.1

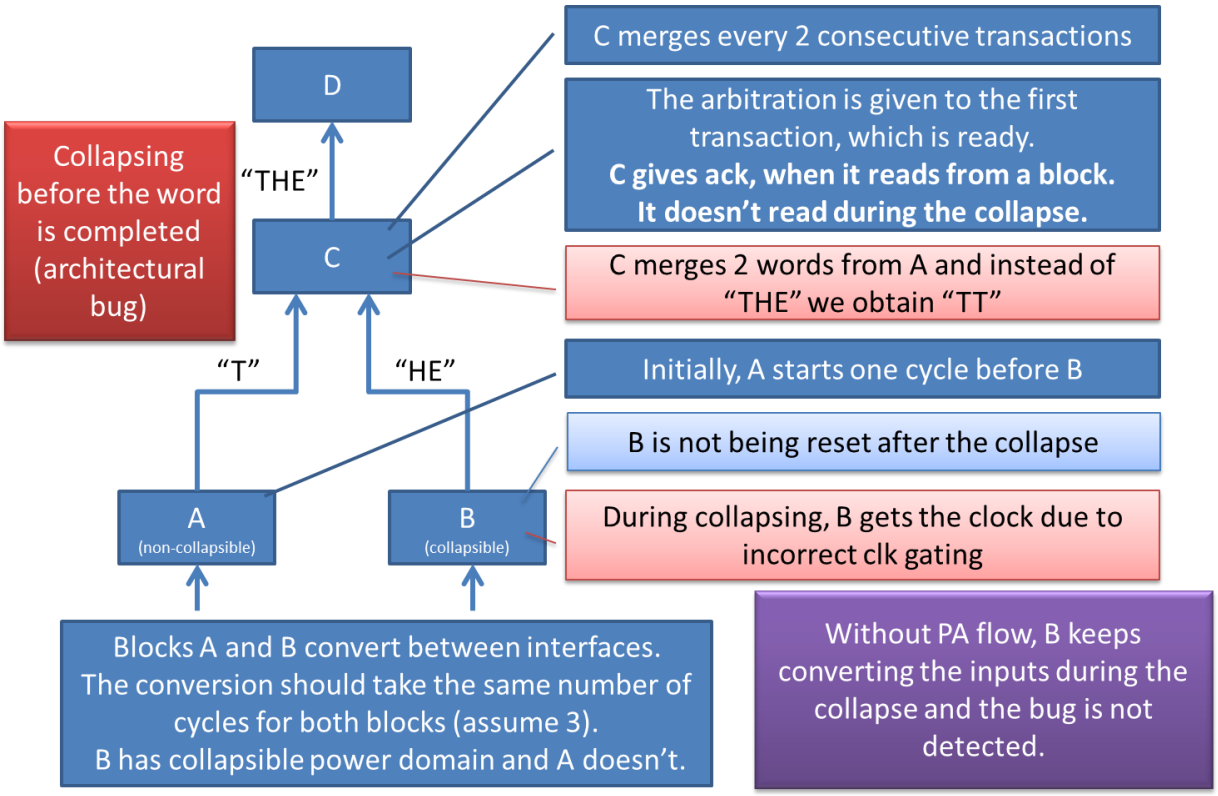


Figure 6 - Collapsed Logic Being Simulated

The bug:

The power collapse is wrongly allowed after A finishes the first half of the word, but before B produces the second half. This shouldn't be allowed (architectural bug).

PA simulation:

During the power collapse the internal signals of B are forced to get 'X' values. This prevents the block from reading and elaborating data.

Cycle	A	B	C	PWR status
1	Works on "T"		Waiting	
2	Works on "T"	Works on "HE"	Waiting	
3	Transmits "T"	Works on "HE"	Receives "T", ACK to A	
4	Works on "T"	Transmits "HE"	Receives "HE", ACK to B	
5	Works on "T"	Works on "HE"	Waiting	
6	Transmits "T"	Works on "HE"	Receives "T"	
7	Works on "T"	Transmits "HE"	Receives "HE"	
8	Works on "T"	Works on "HE"	Waiting	
9	Transmits "T"	Works on "HE"	Receives "T"	
10	Works on "T"	Collapsed	Waiting	Collapse
11	Works on "T"	Collapsed	Waiting	Collapse
12	Transmits "T"	Collapsed	NACK to A	Collapse
13	Transmits "T"	Collapsed	NACK to A	Collapse
14	Transmits "T"	Collapsed	NACK to A	Collapse
15	Transmits "T"	Collapsed	NACK to A	Collapse
16	Transmits "T"	Before reset	ACK to A	
17	Works on "T"	"Late reset" values	Waiting	
18	Works on "T"	Works on "HE"	Waiting	

Table 5 - Non-PA flow simulation

Non-PA simulation:

During the collapse block B keeps reading and elaborating data.

Cycle	A	B	C	PWR status
1	Works on "T"		Waiting	
2	Works on "T"	Works on "HE"	Waiting	
3	Transmits "T"	Works on "HE"	Receives "T", ACK to A	
4	Works on "T"	Transmits "HE"	Receives "HE", ACK to B	
5	Works on "T"	Works on "HE"	Waiting	
6	Transmits "T"	Works on "HE"	Receives "T"	
7	Works on "T"	Transmits "HE"	Receives "HE"	
8	Works on "T"	Works on "HE"	Waiting	
9	Transmits "T"	Works on "HE"	Receives "T"	
10	Works on "T"	Transmits "HE"	NACK to B	Collapse
11	Works on "T"	Transmits "HE"	NACK to B	Collapse
12	Transmits "T"	Transmits "HE"	NACK to A and B	Collapse
13	Transmits "T"	Transmits "HE"	NACK to A and B	Collapse
14	Transmits "T"	Transmits "HE"	NACK to A and B	Collapse
15	Transmits "T"	Transmits "HE"	NACK to A and B	Collapse
16	Transmits "T"	Transmits "HE"	ACK to B, NACK to A	
17	Transmits "T"	“Late Reset” values	ACK to A	
18	Works on "T"	Works on "HE"	Waiting	

Table 6 – Power Aware flow simulation

Logic in a non-PA flow is still being simulated. However, if resets are set correctly then the result of the simulation during the collapse will be eliminated after the reset stage.

Therefore, in non-PA simulation the word is being concatenated correctly.

If we verified that the reset occurs on time then the bug would be caught without PA flow (verification gap).

Technical summary describing the differences between PA and non-PA case:

- B keeps processing data during the power collapse because the clock is ON and the internal signals are not substituted by 'X'. When the internal signals are substituted by 'X' the logic doesn't proceed with the simulation.
- Once the power is up, the results are submitted to C. Whereas in the PA case, the results wouldn't be ready at this point.
- As a result, the second part of the word is ready faster with non-PA simulation than with PA simulation.

3.3 Clamp Cells

Example 4:

Consider the design below with the mux on the output, which is identical in its functionality to a zero clamp cell. By the spec, the flip-flops are not supposed to be reset after the collapse.

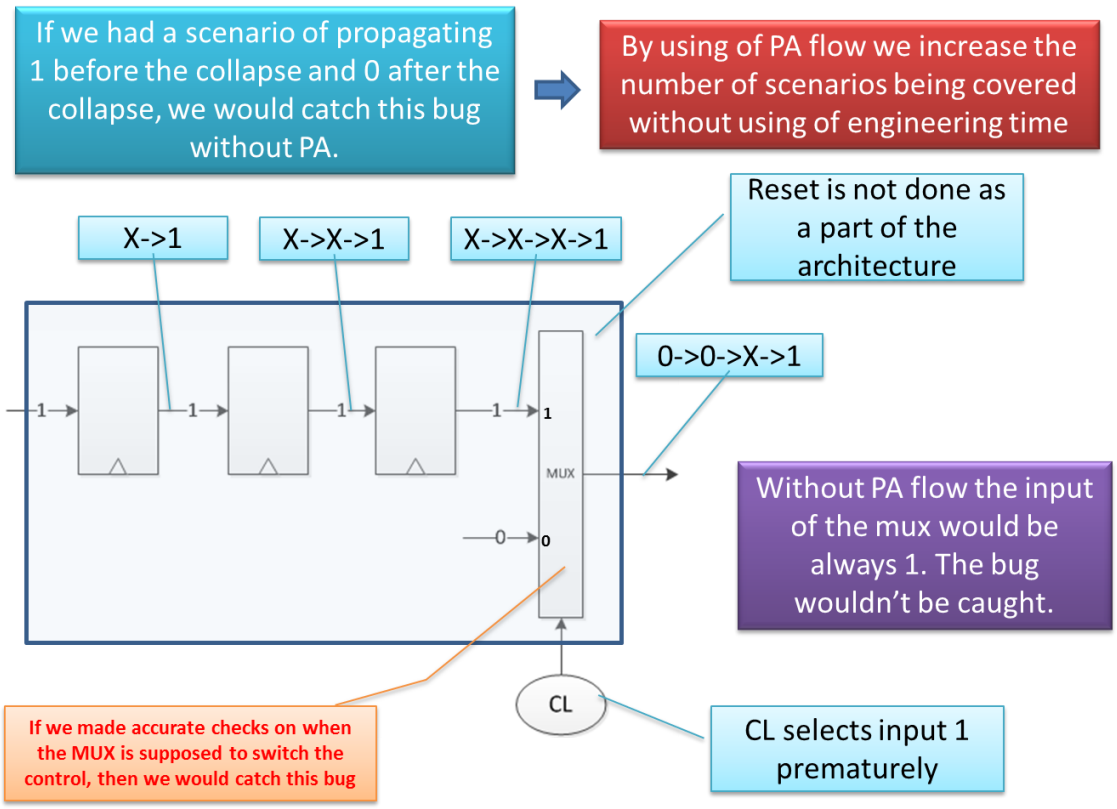


Figure 7 - Incorrect Control Behavior

The bug:
The mux is switched prematurely to the output coming from flip-flops. This leads to X signals propagated outside.

PA simulation:
In PA simulations the outputs of the flops would be collapsed and the 'X' output would lead to a failure.

Non-PA simulation:
In non-PA simulations the output of the flip-flop chain would be always '1', which makes the clamp functionality impossible to be verified.

Example 5:

Consider the follow example. The block as shown in the figure below gets data and makes a quick 1 cycle conversion of data.

- When clamp cells are enabled, the data_valid signal is 0 and the data is not being read.
- The combinatorial logic (CL block) is enabled by the Enable signal coming from a flip-flop. When the domain is collapsed, the flip-flop enable value is 'X' and is reset to 1 after the reset signal.
- The CL logic also depends on the input signal Power Switch. The Power Switch signal controls the power state of this domain.

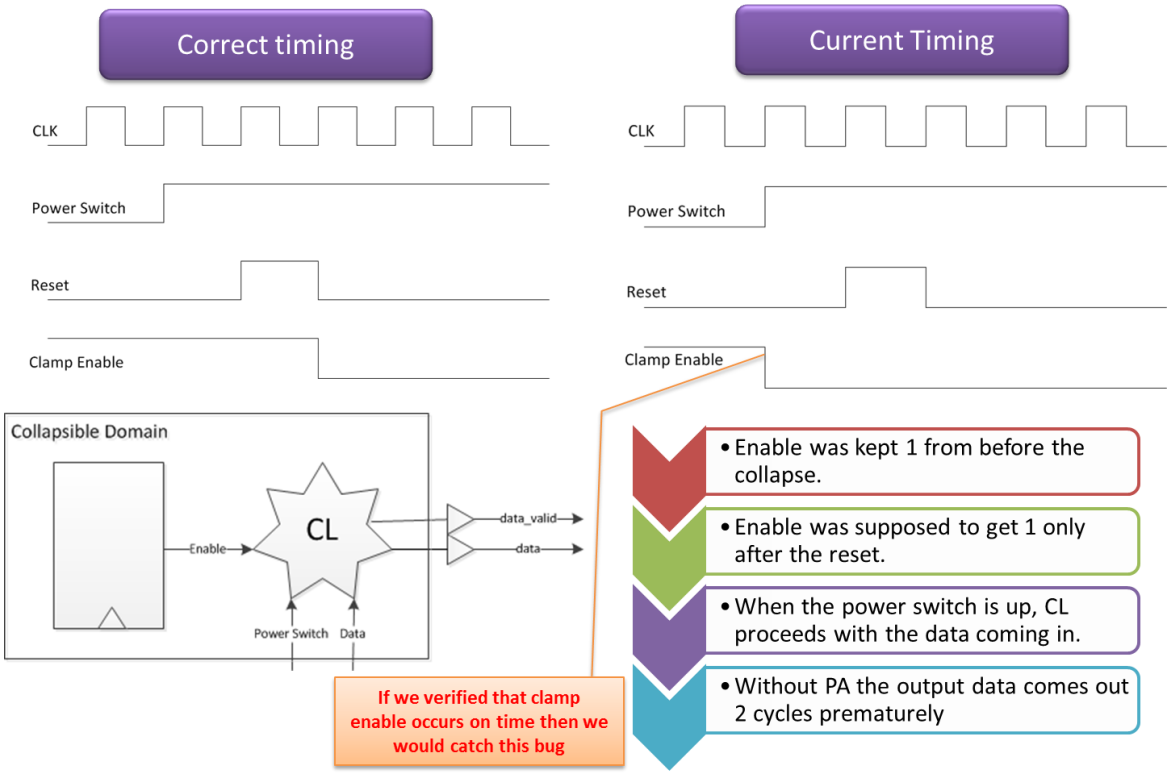


Figure 8 - Signal Synchronization Timing

The bug:

The output of the collapsible block is generated 2 cycles later than expected.

PA simulation:

In a PA-simulation, the output of the CL block is not generated because the Enable signal is 'X', which prevents the logic from producing output until reset. The Enable signal is needed for the CL logic to output good data. Therefore, the CL logic starts converting the data only after the reset has been applied.

Non-PA simulation:
In a non-PA simulation, the Power Switch is used by the CL block. This signal controls the transmission of data to downstream logic. The Enable signal is always '1'.
When clamps are disabled prematurely, the output is successfully transmitted, this producing the output 2 cycles earlier.

Next figure describes the block, mentioned above, as a part of a subsystem.
The example is taken from the section 3.1, Figure 8.

'A' starts 2 cycles prematurely after the collapse, which is ok with non-PA simulation, however is a real bug and is detected with PA simulation.

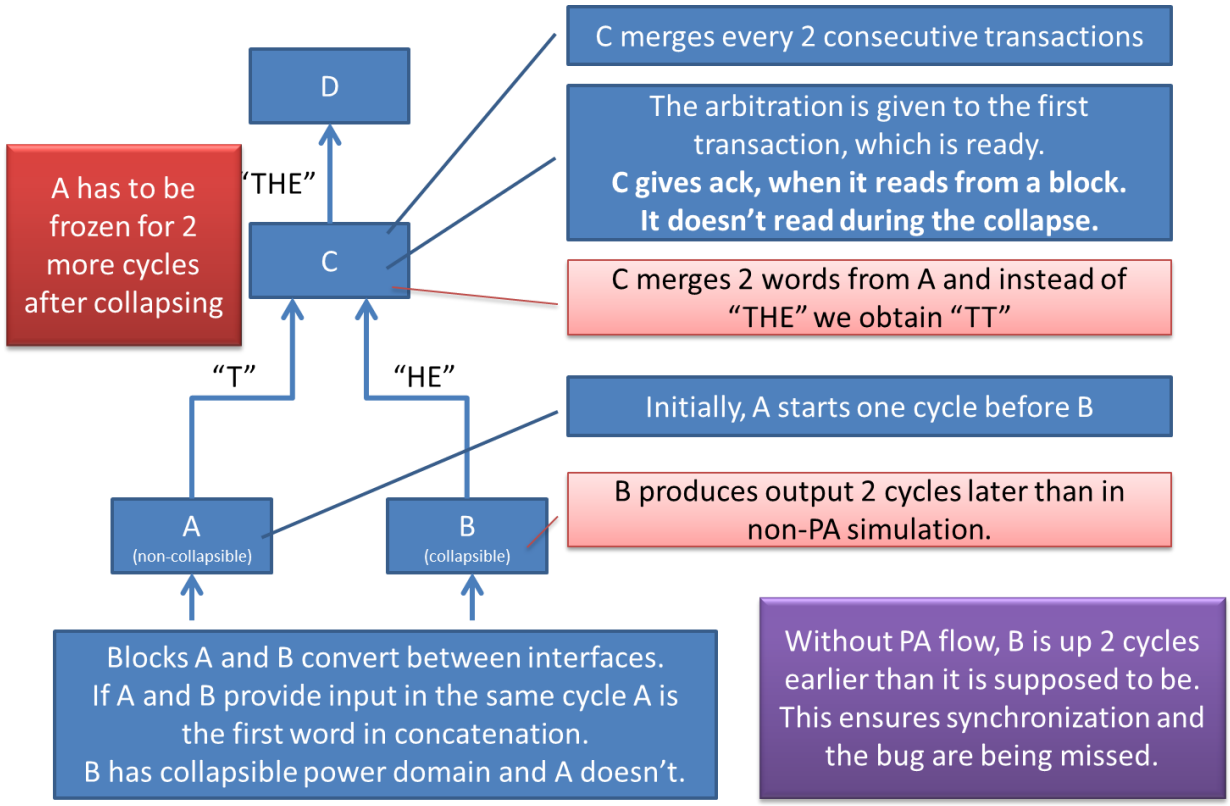


Figure 9 - Signal Synchronization Bug

Example 6:

Consider the following example, described in the figure below:

- The block has a collapsible subcomponent. When the subcomponent is collapsed, the constant '1' becomes 'X'.
- The flip-flops are not a part of the collapsible subcomponent and that is why they are not reset after the collapse recovery.
- CGC (clock gating control) is supposed to enable the clock of SLV, which receives AHB transaction.

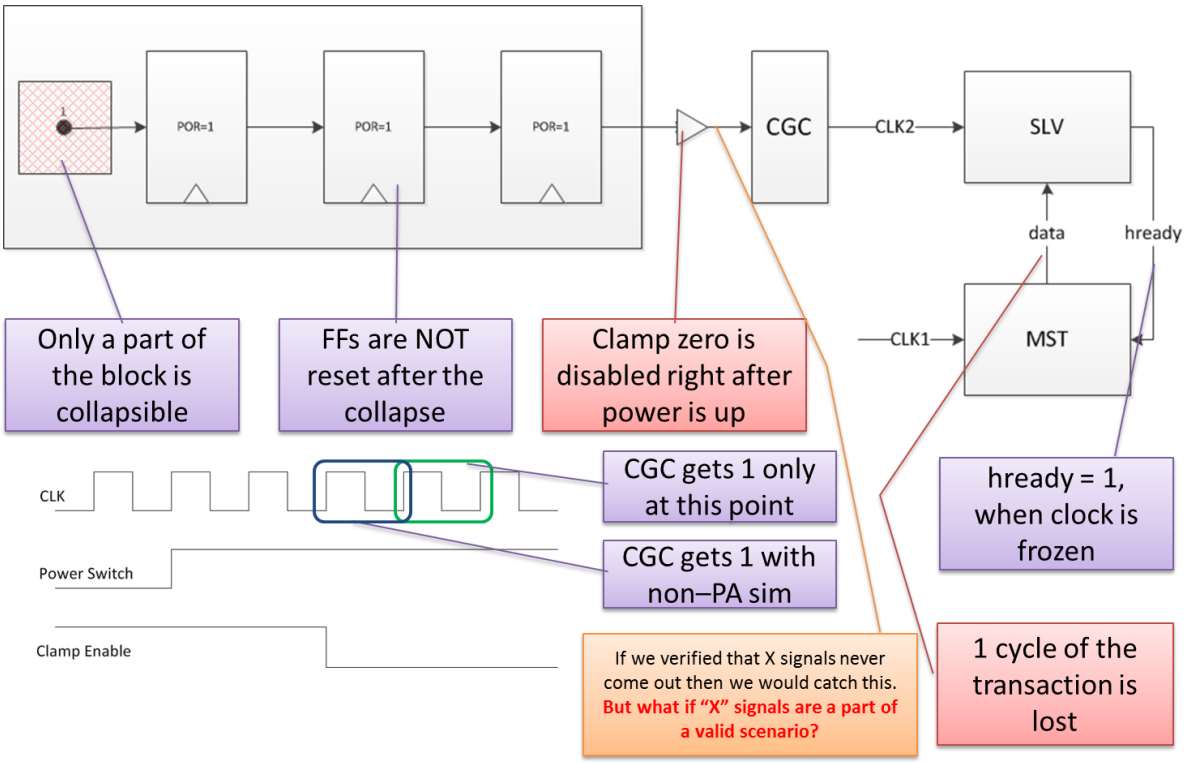


Figure 10 - Block Initialization Example

The bug:

In a PA simulation, the clamp cell is disabled on power recovery. After 3 clock cycles the input to CGC is valid. This means CGC control gets the enable signal 1 cycle later than planned. Late enabling of the slave clock leads to a loss of data (this could be found by non-PA simulations by eliminating the verification gap).

PA simulation:

- When the power domain collapses, the flip-flop chain is not reset as they are not part of the collapsible domain

- After 3 cycles in the collapse mode the output of all the flip-flops is ‘X’
- On power recovery it takes 3 cycles for ‘1’ to get propagated to the output
- CGC gets the enable signal 3 cycles after the power recovery

Non-PA simulation:

- The output of the flip-flop chain is always ‘1’
- As a result the clock is enabled when the zero-clamp cell is disabled
- CGC gets the enable signal 2 cycles after the power recovery

Technical summary:

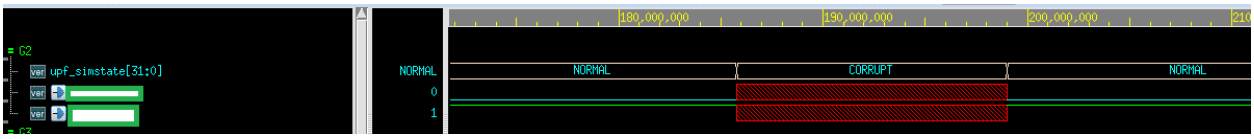
- In a non-PA simulation, the CGC block gets the enable signal one cycle early.
- In a PA simulation, the CGC is enabled late
- hready is kept high, when the slave doesn’t have a clock (in order to avoid hanging).
- All the above lead to a loss of the address phase: master assumes that because of hready high the slave received the address phase and continues to the data phase.

4. Debugging with VCS NLP flow

VCS NLP provides a set of virtual signals, helping users to perform debug.
A few examples are shown below.

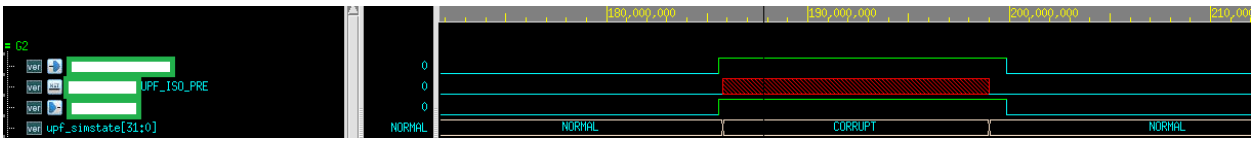
Design collapse

upf_simstate virtual signal keeps the state of the power domain.
The second and the third signals are collapsed, when the block is collapsed.



Isolation cells

The first signal is clamp control. Once it is enabled the isolation signal (the third one), gets value ‘1’. The pre-clamp value of the signal is kept in “_UPF_ISO_PRE” virtual signal.



UPF interpretation

DVE provides a convenient UPF interpretation:

We can observe power domain, isolation cells and element, included in this power domain.

LPA VCS built in assertions

Built in LPA assertions are a useful tool for catching low power issues.LPA provide warning messages when inputs are toggled during the collapse. This helps to detect bugs in examples 3 and 5.

Disbabling of LPA built in assertions

LPA VCS built in assertions are very useful, but sometimes the assertions trigger an alert on custom design. This is an example on how to disable LPA assertions:
set_lp_msg_onoff off -msg {LP_PSW_CTRL_INIT_INVALID}

Messages

Corrupt state message:

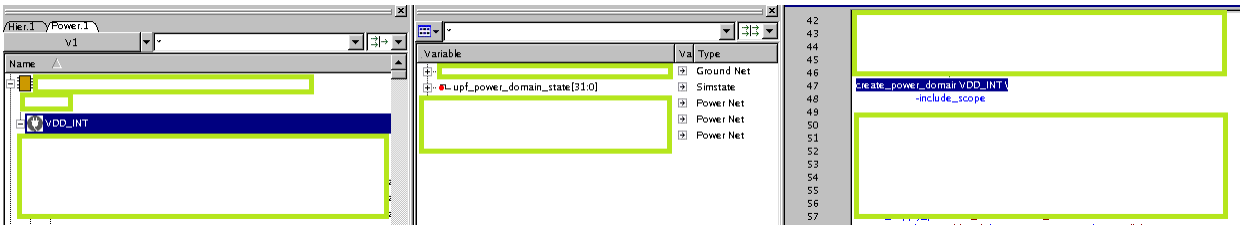
[1285825103 ps] [INFO] [LP_PPN_STATE_CHANGE] State of the primary power net 'vdd_int' of power domain 'SOME_PATH/VDD_INT' changed from FULL_ON to OFF.
[1285825103 ps] [INFO] [LP_PPN_VALUE_CHANGE] Voltage of the primary power net 'vdd_int' of power domain 'SOME_PATH/VDD_INT' changed from 1 V to 0 V.

Power recovery message:

[2656232 ps] [INFO] [LP_PPN_STATE_CHANGE] State of the primary power net 'vdd_int' of power domain 'SOME_PATH/VDD_INT' changed from OFF to FULL_ON.

[2656232 ps] [INFO] [LP_PPN_VALUE_CHANGE] Voltage of the primary power net 'vdd_int' of power domain 'SOME_PATH/VDD_INT' changed from 0 V to 1 V.

DVE has an option to show UPF source code



5. Conclusion

In this paper, numerous examples demonstrating the efficiency of the low power flow were introduced. Special attention was paid to verification gaps, which can be alerted with power aware flow as well. Therefore, VCS NLP flow was introduced as a tool for finding verification inaccuracies along with design functional bugs.

It has been concluded that PA flow is especially helpful with finding functional bugs when some of the standard collapse recovery steps are not followed.

The classification, provided in this paper, is especially helpful for better understanding of the major areas of use of PA flow.

6. References

[1] Bembaron F., Kakkar S., Mukherjee R. and Srivastava A., "Low Power Verification Methodology using UPF", DVCon 2009

[2] Freddy Bembaron, Rudra Mukherjee, Sachin Kakkar, Amit Srivastava: "Low Power Verification Methodology Using UPF", DVCon 2008

[3] Rudra Mukhejee, Amit Srivastava, Stephen Bailey: "Static and Formal Verification of Low Power Designs at RTL using UPF", DVCon 2008.

[4] Stephen Bailey, Amit Srivastava, Mark Gorrie, Rudra Mukherjee: "To Retain or Not to Retain: How do I verify the states of my low power design", DVCon 2008

[5] Stuart Sutherland, "I'm Still In Love With My X!", DVCon 2013

[6] VCS MVSIM Native User Guide.