DESIGN & VERIFICATION DVCON 2014 CONFERENCE & EXHIBITION

Is your Power Aware design really x-aware ?

Durgesh Prasad, Design and Verification Technology
Jitesh Bansal, Design and Verification Technology

Mentor Graphics®

Introduction

- Power Aware simulation
  - Inject x-values to mimic design shut-off, back biasing, voltage scaling, PA cells shut-off.
  - Simulation can catch many design issues arising due to these power techniques.
    - Automated power aware checks and assertions(bind checker) further help in catching these issues.
  - ssues arising due to x-optimism/pessimism of RTL can not be caught by a raw power aware simulation.

X Optimism: Condition stmts

- Appears commonly in simulation
  - If-statements :

```
if (cond)
  reg_a <= f0;
else
  reg_a <= f1;
```

Simulation skips this

Simulation picks this

  - Case-statements :

```
case (cond)
  2'b00:   reg_b <= f0;
  2'b11:   reg_b <= f1;
  default: reg_b <= f2;
endcase
```

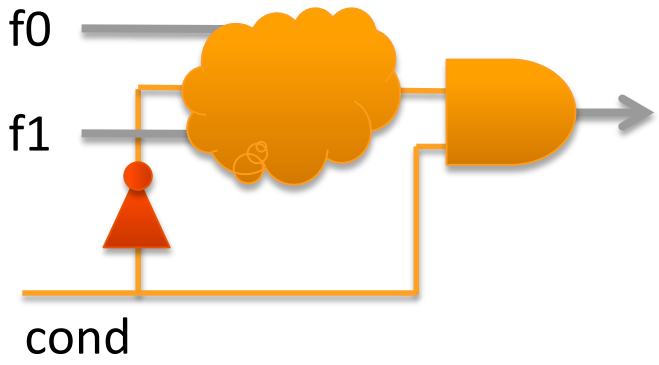
Simulation picks this

  - h/w will consider all scenarios
- "Simulation" is not simulating what happens in actual h/w
  - Some bugs cannot be detected using simulation

X-optimism solution: X-Prop

- Propagating X values forward

```
if (cond)
  reg_a <= f0;
else
  reg_a <= f1;
```



cond	f0	f1	RTLsim	X-Prop
X	0	0	0	X
X	0	1	1	X
X	1	0	0	X
X	1	1	1	X

Results from synthesis are unpredictable.
Simulation may not exercise the worst scenario.

-> Propagating Xs

X-Optimism Solution: x-Trap

- Conditional statement

```
always_comb begin
  if (cond)
    reg_a <= f0;
  else
    reg_a <= f1;
end
```

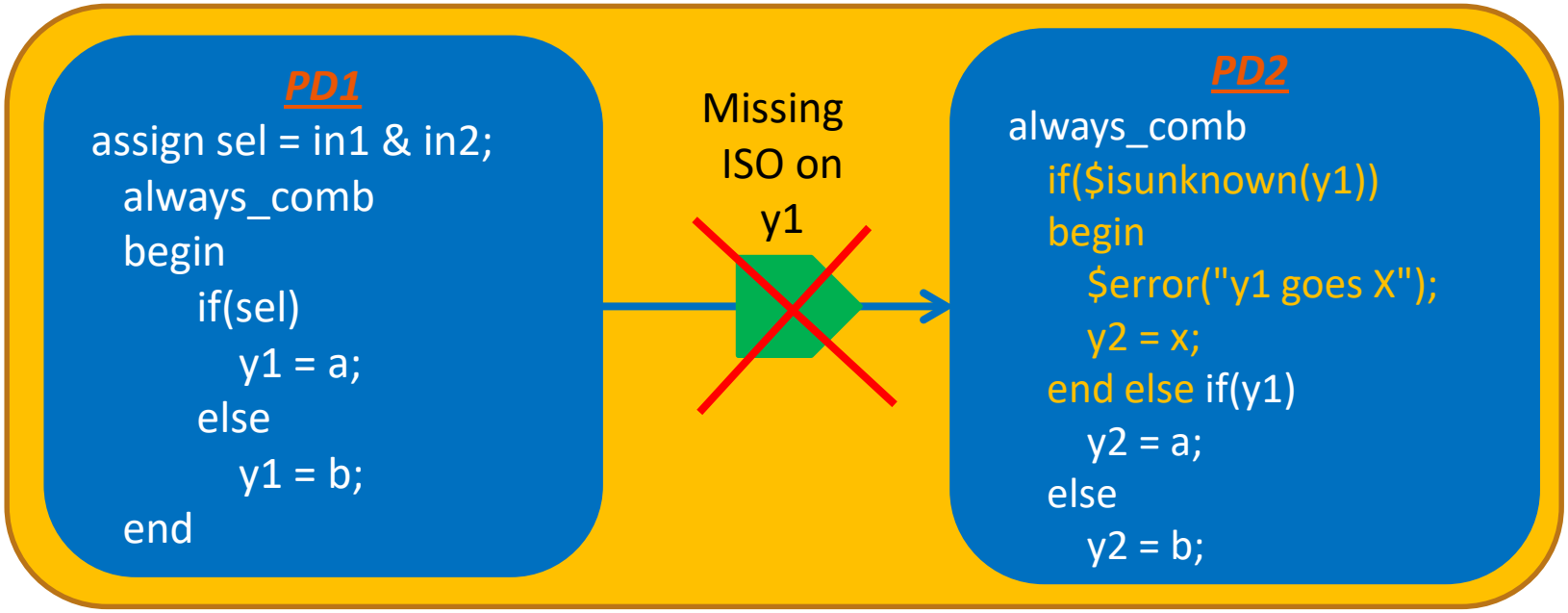
- Conditional statement with implicit assertion

```
always_comb begin
  assert !$isunknown(cond);
  if (cond)
    reg_a <= f0;
  else
    reg_a <= f1;
end
```

To catch Xs in condition expressions ASAP

X-Prop PA solution

- Catch x-optimism issues



PD1	PD2	sel	y1	RTL sim (y2)	X-prop PA (y2)
Off	On	x	x	b	x

X-Prop PA(Handling Noise)

- Automated Noise Reduction

PD1

```
always_comb
  begin
      if(sel)
         y1 = a;
      else
         y1 = b;
  end
```

PD2

```
assign ctrl = xprop_pa_logic(PD2 simstate);
always_comb
    if($isunknown(sel) && ctrl)
    begin
        $error("sel goes X");
        y1 = x;
    end else if(sel)
        y1 = a;
    else
        y1 = b;
```

PD1	PD2	sel	y1	y2	Error for 'sel'	Error for 'y1'	Xprop-PA Error for 'sel'	Xprop-PA Error for 'y1'
On	On	x	x	x	Yes	Yes	Yes	Yes
Off	On	x	x	x	Yes	Yes	No	Yes
Off	Off	x	x	x	Yes	Yes	No	No
On	Off	-	-	x	-	Yes	-	No

X-Prop PA(Handling Simstates)

- ***xprop_pa_logic*** can be defined by following table:

Simstate	Xprop for Combinatorial logic	Xprop for State elements
CORRUPT	OFF	OFF
CORRUPT_ON_ACTIVITY	OFF	OFF
CORRUPT_STATE_ON_ACTIVITY	ON	OFF
CORRUPT_STATE_ON_CHANGE	ON	ON

X-Prop PA(Debugging)

- The proposed solution uses SV Assertions, which is known for it's controllability and ease of debug.
- Design element categorization
  - XPROP-FF

    *# ** Error: XPROP-FF: 'reset' goes X.*

    # Time: 2 ns Scope: tb.dut File: ./src/vl_file/lib/rtl.v Line: 6
  - XPROP-CLK
  - XPROP-LATCH
  - XPROP-FSM
  - XPROP-COMB
- User controlled x-propagation
  - Provide enabling and disabling of x-prop logic based on timing to handle POR.

X-Prop PA(Automated checks)

- Control signal corruption check
  - save/restore signal(ret) of retention cell is ‘X’ at power up(a potential bug). The state logic controlled by these save/restore signals would trigger xprop assertion.

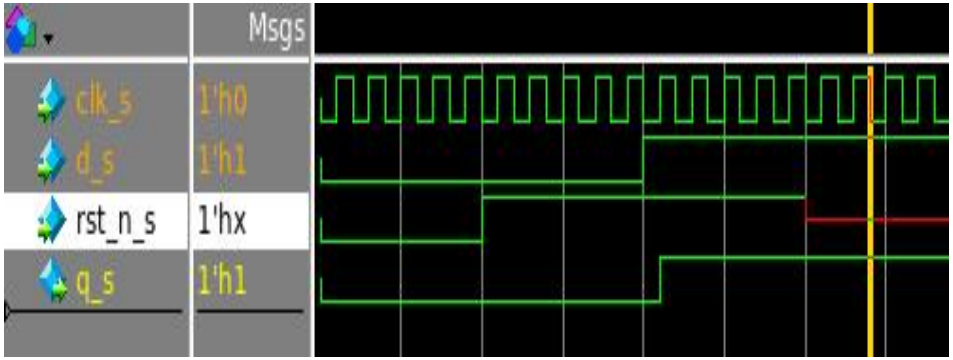
*# ** Error: XPROP-FF: 'ret' goes X.*

 # Time: 2 ns Scope: tb File: ./src/vl_file/lib/dut.v Line: 37

X-Prop PA(Automated checks)

- Reset failures
  - From RTL simulation, DFF with async-reset is not sensitive to ‘x’ at the reset pin which is a potential bug(would be caught by x-prop assertion)

```
always @(posedge clk_s or negedge rst_n_s)
begin: ff_verilog_model
    if(rst_n_s == 1;b0)
        q_s <= 1’b0;
    else
        q_s <= d_s;
end
```



Conclusion and References

- Conclusion
  - A controlled x-prop PA solution can catch x-optimism related issues specific to power aware designs.
  - The various simstates of the power domain can be simulated to catch potential issue without generating noise.
  - Techniques like “controlled assertion failure” and “design element categorization” can make debugging user friendly.
  - This technique also provide automated checks like “control signal corruption” and “reset failure”.
- References

  - Don Mills, “Being Assertive With Your X’, User2user 2013
  - Stuart Sutherland, “I’m Still In Love With My X!”, DVCon 2013
  - Mike Turpin, “The Dangers of Living with an X” ARM 2003