Every Cloud – Post-Silicon Bug Spurs Formal Verification Adoption

Blaine Hsieh : Faraday Technology, Taiwan
Stewart Li : Mentor Graphics, Taiwan
Mark Eslinger : Mentor Graphics, Fremont
Every Cloud …

• It’s an exciting time getting first silicon back!
• Unfortunately there was a bug
  – A DDR3 write to pre-charge timing bug
  – Why?
  – Millions of cycles of simulation hadn’t found it

• All hands on deck to reproduce it
• What could we have done differently?
• Let’s see what formal can do!
The ABC’s of Formal

- **Assurance**: Proofs and bounded proofs
- **Bug Hunting**: Includes post-silicon debug
- **Coverage Closure**: Reachability analysis
• Run Formal and get CEX
• Use CEX as initialization for next formal run
• Example: 32 deep FIFO overflow bug
  – gp_1: cover property (@(posedge clk) buff_level == 5’d10);
  – gp_2: cover property (@(posedge clk) buff_level == 5’d20);
  – a_no_overflow: assert property (@(posedge clk) !(push && full) );
DDR3 SDRAM Features

• **Double Data Rate** dynamic random access memory
• Supports up to 2G bytes for each rank
  – Supports 2 ranks (cs0, cs1)
  – Burst lengths of 4 to 8 depending on mode/config
  – Supports 8-bit, 16-bit, 32-bit, and 64-bit widths
• Compliant with AMBA AHB/AXI protocols
  – Up to 8 slave interfaces
  – Synchronous and asynchronous modes
DDR3 Signals and Timing

• DDR3 Truth Table

<table>
<thead>
<tr>
<th>Function</th>
<th>CS#</th>
<th>RAS#</th>
<th>CAS#</th>
<th>WE#</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bank Precharge (PRE)</td>
<td>L</td>
<td>L</td>
<td>H</td>
<td>L</td>
</tr>
<tr>
<td>Bank Write (WR)</td>
<td>L</td>
<td>H</td>
<td>L</td>
<td>L</td>
</tr>
<tr>
<td>No Operation (NOP)</td>
<td>L</td>
<td>H</td>
<td>H</td>
<td>H</td>
</tr>
</tbody>
</table>

• Example: Write(BC4) to PreCharge

WR to PRE 11 cycles min

• Bug Scenario

WR to PRE  in 7 cycles
Design Block Diagram

- APB (disabled)
- AXI0 (ID = 2)
- AHB1
- AHB2
- AXI3 (ID = 2)
- AXI4 (disabled)
- AXI5 (disabled)
- AXI6 (disabled)
- AXI7 (disabled)
- DDR0 (cs[1] = 1)
- DDR1 (ignored)
Modeling Layer Code

```verilog
parameter PRECHARGE = 7'b11_0010_0;
parameter READ = 6'b11_0101;
parameter WRITE = 6'b11_0100;
parameter ACTIVE = 6'b11_0011;

reg pre_cke;
always @(posedge ddrclk) pre_cke <= ddr_cke;

wire [6:0] ddr_cmd = {pre_cke, ddr_cke, ddr_cs, ddr_ras,
                      ddr_cas, ddr_we, ddr_addr[10]};
wire precharge = (ddr_cmd == PRECHARGE);
wire active = (ddr_cmd[6:1] == ACTIVE);
wire write = (ddr_cmd[6:1] == WRITE);
wire read = (ddr_cmd[6:1] == READ);
```

From DDR3 Truth Table

Simplify Commands
Bug Hunt Assertions

Formal picks “start” of write

wire same_pre = precharge && (ddr_ba == my_ba);

First goal-post target

cov_gp: cover property (@(posedge ddr_clk) !my_wr && active );

Final bug target (wr to pre)

a_wr_to_pre_bug: assert property (@(posedge ddr_clk)
  $rose(my_wr) |-> (!same_pre)[*11] );
**Formal Counter Example**

**Assertion Signals**

**DDR3 Signals**

**Modeling Signals**

write (ba=110, cs=10)

precharge (ba=110)

7 cycles later
Counter Example (Close-up)

**Assertion Signals**

**DDR3 Signals**

- write (ba=110, cs=10)
- precharge (ba=110) 7 cycles later

**Modeling Signals**
• Assumptions of buggy module input pins
  – Address (row, bank, column) & data bus → random
  – If \( \text{cmd\_queue\_full} == 1 \) then \( \text{cmd\_go} = 0 \) (should be disabled)
  – Other input signals shall be tied to 0 or 1 (configuration values)
Failure Symptoms of the Bug

- Summary of formal results on the buggy block
  - Bug CEX of 71 cycles in 1.5 h
  - Fixed code had bounded proof at 35 cycles (48 hr run)
- Bug is triggered when following conditions happen
  1. tWL+4+tWR >=15 (DDR3 mode), or tWL+2+tWR >=8 (DDR2 mode)
     - For DDR3 1333 tWL+4+tWR = 7 + 4 +9 = 20
  2. At least 4 consecutive write commands with different bank address happened.
     - For example bank 0, 1, 2, 3 write commands happened at T9, T11, T13, T15
  3. After 4 write commands, another bank B write command with different bank from the previous 4 write commands is pushed into DDR command queue when command queue is empty.(T16)
  4. Another commands with bank C is pushed into command queue, and bank C is bank 0 or 1 or 2 or 3, but different row as previous 4 consecutive bank 0,1,2,3 write commands

Formal can find some Gnarly(with a G) bugs!
Formal w Subsequent Projects

• With our success at applying formal on the DDR3 project we successfully rolled formal out to other projects/designs
  – CPU Bus Interface
  – Interrupt Controller
  – MAC Controller
Property Best Practices

• Keeping assertions simple and sequentially short
  – Make it easier for others to apply and understand
  – More efficient for formal
• Simplify signal names for use in properties
• Using modeling code to simplify assertions
• Describe both desired and undesired behavior
• Using predefined formal verification IP whenever possible
Formal Best Practices

• Leveraging simulation data for DUT initialization when designs have complex initialization sequences

• Using various techniques to resolve inconclusives
  – Formal techniques such as goal posting and other abstraction techniques
  – Picking the level of hierarchy to run at
    • For simplification of the state space
    • For simplification of constraining the DUT
Conclusion

• Post-silicon bugs are never fun
• Application of formal covering the ABCs of formal for both assurance and bug hunting can mitigate their occurrence
• With the confidence of our success applying formal to the post-silicon bug hunt, we have been successful applying formal on multiple other projects
• We found our silver lining with formal!