Explanation of X pessimism effect in gate level simulation

Application note 2016-03-23

Doc. No GRLIB-AN-0010

Issue 1.0

CHANGE RECORD

Issue	Date	Section / Page	Description
1.0	2016-03-23		First issue.

TABLE OF CONTENTS

1 INTRODUCTION

1.1 Scope of the document

This document is a collection of information on the X pessimism phenomenon that occurs when simulating digital designs on gate level. The note is informational and the applicability is general and not restricted specifically to GRLIB.

The work has been performed by Cobham Gaisler AB, Göteborg, Sweden.

1.2 Reference documents

[RD1] GRLIB IP Core User's Manual, Cobham Gaisler AB, http://www.gaisler.com/products/grlib/grip.pdf

2 ABBREVIATIONS

TBC	To Be Confirmed
TBD	To Be Defined

3 INTRODUCTION

3.1 Overview

Gate level simulation is a method of verifying a digital design after synthesis to a technology library has been done. When simulating on gate level, the synthesized netlist and simulation models for the logical gates are loaded into the simulator. The state of all signals in and out of the logical gates are tracked by the simulator, and whenever an input signal for a gate changes the simulation model for that gate determines if the output signal values should change in response. Gate level simulation can be performed with ideal timing where the gates are infinitely fast (zero delay, or unit delay, simulation) or with delay data for each gate and signal read in from the design (back-annotated simulation).

Both VHDL and Verilog provide one or more values ('U','X','Z','-', etc) to represent an unknown logical value. X is used in this note to refer to any of these values. When simulating on gate level, a cell seeing such value on one of its input must assume the input could be any of either logical-0 or logical-1 at that time. If this means that the cell model can not know whether its output would become logical 1 or 0, most cell libraries will output an unknown logical value X.

Note that there are also values that represent weakly driven high or low, such as 'H' and 'L' in VHDL. There can be resolved to a logical value in the cell and therefore are not considered X values for the purposes of this note.

3.2 Sources of X

Most technologies can not guarantee a specific value of digital state elements when powering up the device. A flip flop or memory bit will power up to 1 or 0 depending on environmental factors, exact shape of the power supply ramp, chip-to-chip process variations, etc. For that reason, the simulation model for flip-flops will output X from time 0 until the first clock edge where it gets loaded with a real logical value. Most memory (SRAM) simulation models will for the same reason return X on its inputs until you read from a memory address that has been previously written to with well-defined data.

There are some other common sources of X in simulation:

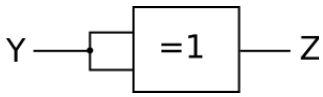
- Missing simulation models, or other error in simulation setup. This should be checked first
- For any input or bi-directional pin where the signal on the outside (in the test bench) is undriven or tristated, the simulation model for the IO buffer will propagate X into the design.
- Some PLL simulation models output X on its clock outputs until lock has been achieved
- Integrated clock gating (ICG) cells where the clock-enable input of the cell is X may output X on its output clock.

- Depending on simulation settings and cell simulation model design, the cell may outputs may go to X when there is a timing violation
- When simulating with back annotated timing, all internal signals in the design will be X for some time before the inputs will have time to propagate through the design. This even if the signal are driven by constants in the simulation test bench.
- Some cell libraries may be pessimistically modelled and return X as soon as one of inputs is X, even though it has enough information to know what value would come out.
- For modern ASIC technologies with power gating capabilities, any power domain that is turned off will have all state in that domain become X. This can make full gate-level simulation with such libraries difficult.

4 EXPLANATION OF X PESSIMISM

X pessimism is when outputs from combinatorial logic becomes X in cases where it would never become X in reality, considering all possible combinations of 1 and 0 at that time. The root cause for it is that when X values converge, the cell model can not know which combinations of X are possible. To illustrate the idea, a couple of examples with increasing complexity are provided below.

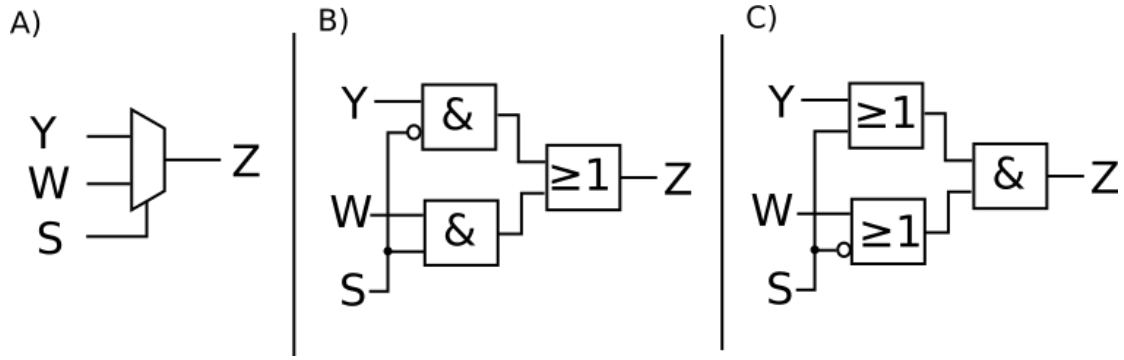
4.1 Example 1: Exclusive-or with common inputs



This trivial example is just one XOR gate with its input tied together. Such a gate will always output logical zero but when provided with X on its inputs, since the cell simulation model for the XOR gate does not know that the inputs are always equal, it will see (X xor X) and from this it can not determine the output value and therefore output X.

This example is just a toy example intended to introduce the idea, since a synthesis tool would recognize that Z output is always low and optimize out the logic gate.

4.2 Example 2: Two-to-one multiplexer made with multiple cells



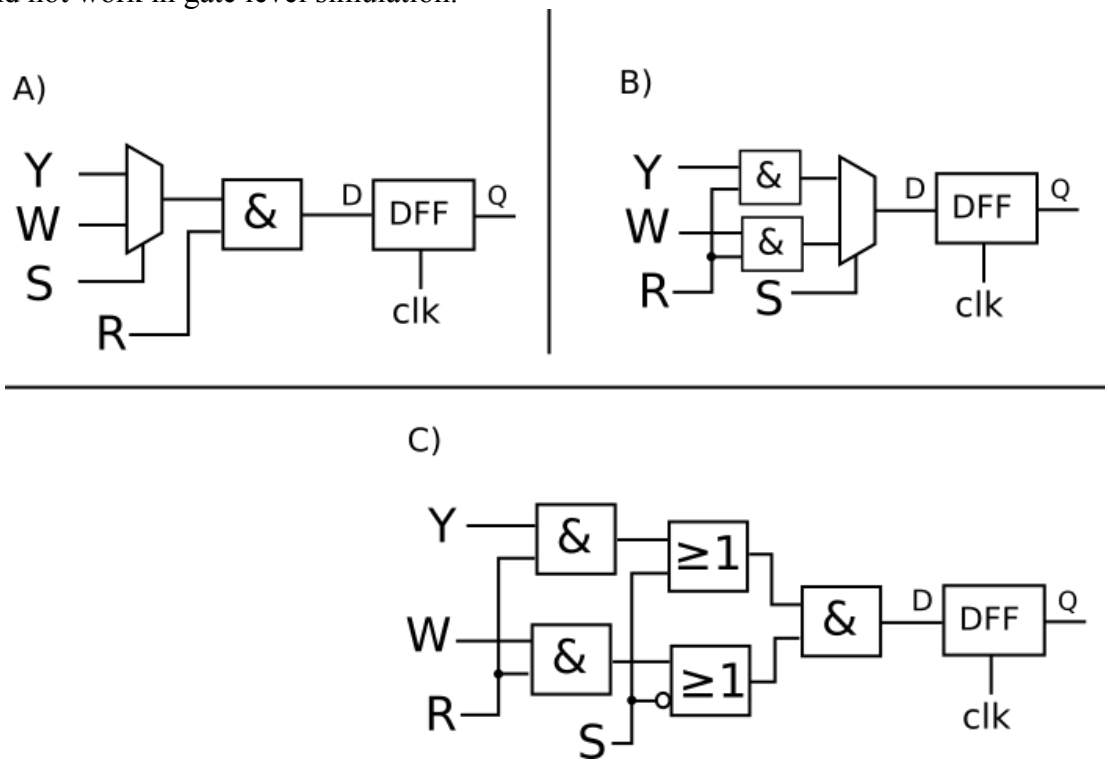
This example consists of a two-to-one multiplexer that forwards either its Y or W input depending on the value of S. Three variants are shown, A) using a monolithic multiplexer cell, B) implemented with and-or logic, C) implemented with or-and logic. All the three variants of the multiplexer shown have the exact same truth table, as shown below, and the synthesis tool is allowed to generate any of them.

Y	W	S	Z
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

However, if we look at the case where both inputs Y and W are equal, and S is undefined, the implementations B and C will under some conditions forward X in simulation even though we know that Z can only get one value. For the B (and-or) variant, this will happen when Y=W=1 and S=X, for the C (or-and) variant, this will happen when Y=W=0 and S=X.

4.3 Example 3: Synchronous reset logic moved by optimization

This example is intended to show that synchronous reset may be implemented in an X pessimistic way and not work in gate level simulation.



Circuit A shows a D-flipflop that is synchronously reset to zero by a reset input signal R. When it is not reset, it is fed by a multiplexer with inputs Y,W,S that we assume are unknown and therefore X during reset. The implementation in circuit A is guaranteed to correctly reset the flip flop in simulation since the AND cell simulation model will see one of its inputs as zero and therefore know the output must become zero even if the output of the multiplexer is X.

However, if the synthesis tool finds the path from S to be timing critical, it might decide to move the reset gate to before the multiplexer, and put it only on the data inputs, which leads to circuit B. Now when reset is asserted, both the inputs to the multiplexer will be 0 and the selector will be undefined. Regardless of selector value, it will propagate logical 0 to the register so the optimization is logically correct.

Now consider the possible multiplexer implementations given in example 2. If the synthesis tool chooses to implement the multiplexer according to example 2C, the resulting gate network will become like example 3C. This will not correctly propagate the reset in simulation is Y,W,S=X and R=0 however the real world circuit still resets correctly.

5 MITIGATION TECHNIQUES AND WORK-AROUNDS

This section describes some different possible approaches that can be used to deal with X pessimism. Which ones that are appropriate to use varies from project to project.

5.1 Simulator support

Some simulators (either as built-in functionality or through third-party plugins) can provide support for resolving X pessimism at run time. The simulator detects when an X is generated, checks by looking at the full combinatorial logic network if it is a pessimistic X, and if so overrides it with the correct value. This provides a clean solution in the sense that it does not reduce simulation coverage since real X-values will still propagate correctly so design errors such as missing resets etc will still correctly cause X propagation.

5.2 Modifying design to reduce pessimism

It is in some cases possible to add gates to reduce the amount of X in the design, which reduces the probability of X pessimism problems. One approach is to add resets to registers that don't really need to be reset for functional correctness. To avoid the problems shown in example 4, there may need to be special “don't touch” attributes added on the reset gates.

Synchronous reset is prone to the problems shown in example 4, but asynchronous reset is normally not since it is never mixed with combinatorial logic. Therefore another method to reduce pessimism is to switch over from synchronous to asynchronous reset.

To solve the same problem for RAM blocks in general, a wrapper would need to be added to prevent X propagation after reset, for example with a state machine that clears the memory directly after reset. These things may also require special attributes on the wrapper to avoid being folded into the surrounding combinatorial logic on optimization, that could lead to effects similar to those shown in example 4.

5.3 Test bench reduction of X

The test bench can be modified to avoid driving X into the design, for example by adding pull-ups to data buses so they input a logical high when the buses are undriven. This reduces to some degree the coverage of the test if the value on the data bus affects behaviour of the design.

5.4 Forcing state on simulation start

Avoid having unknown state in flip flops and memory blocks on power-up by simply forcing a specific power-up value when the simulation starts. Forcing can be done either through the simulator or through modifying the simulation models for the cells.

This method is normally very effective, however the downside is it reduces test coverage. Especially it might mask missing resets in the design. One way to partially recover that is to do the simulation multiple times with different start up values (all set to 0, all set to 1, random values).

5.5 Modifying cell models to be X optimistic

X propagation can be reduced by modifying cells so they drive 0 or 1 instead of X even when the output value is unknown. This may be useful, for example for PLL simulation models that output X on the clocks before locking. This technique may also be useful in the pad simulation models as an alternative to implementing pull-ups in the test bench.