QCADesigner Digital Simulator User Guide

Dominic Antonelli

Originally Written: July 22, 2003
Latest Revision: July 23, 2003

1 Introduction

The purpose of this document is to provide the basic information needed to use the digital simulator in QCADesigner. This document assumes a basic familiarity with both QCADesigner and QCA.

If you have any questions, comments, etc. about this document or the simulator please email me at *dantonel@nd.edu*.

The digital simulator is designed to be significantly more efficient than the other, more exact, simulators while still providing the accuracy necessary to design and diagnose QCA circuits. The digital simulator is mainly concered with configurations of cells that have well-defined behavior and applications to digital design.

2 Restrictions of the Simulator

If you want to be sure the digital simulator is accurate and will not crash (which you obviously do), the following design restrictions should be taken into account.

2.1 Indivual Component Restrictions

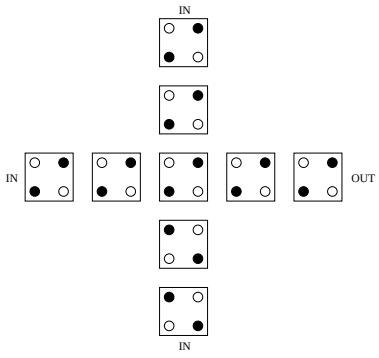
The digital simulator only works with specific well-defined cell configurations. Besides the basic 45- and 90-degree wires (straight or bent), it can handle majority gates (Figure 1(a)), inverters (Figure 1(b)), wire crossings (Figure 2(a)), rippers (Figure 2(b)), one-to-two fan-outs (Figures 3(a) and 3(b)), and one-to-three fan-outs (Figure 3(c)).

Cells forming a wire cannot have an angle between their centers of more than 45 degrees. Thus, it takes three cells to rip off a 45 degree wire and be ready to cross another 45 degree wire (provided both 45 degree wires start at the same y-coordinate) using 90 degree cells. Refer to Figure 2(c) to see a picture of how this should work. Another point to note on the 45 degree wires is that the 90 degree cells immediately next to them should be in the same clock zone. Failures are possible if they are not. It is suggested, but not required, that the output cell of a majority gate be in the next clock zone when compared to the rest of the majority gate.

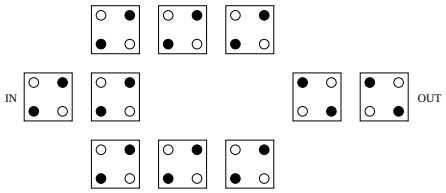
In addition, fixed cells do not send any information to cells that are diagonal from them. Fixed cells completely ignore them. This works perfectly for fixed cells adjacent to majority gates and for most normal uses of fixed cells. However, strange uses of fixed cells that have them specifically interacting with cells that are diagonal from them will not work. This may be changed in a later version, but only if there is a compelling reason.

2.2 Multiple Component Restrictions

In addition to these restrictions, there are a few that limit what can be placed together within the same clocking zone. Series of majority gates within a clocking zone are NOT guaranteed to work, although they

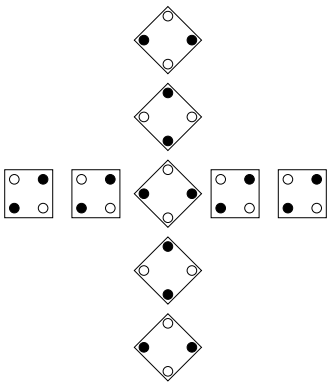


(a) Majority Gate

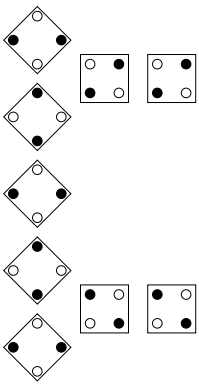


(b) Inverter

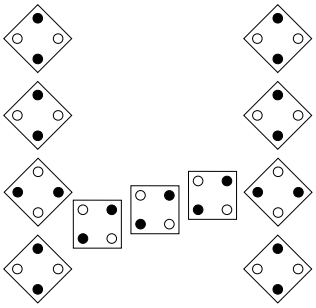
Figure 1: Majority gate and inverter



(a) Wire Crossing

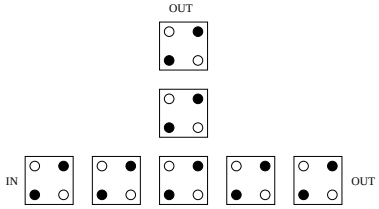


(b) Ripper

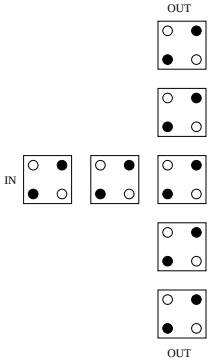


(c) Spacing needed for rip then cross

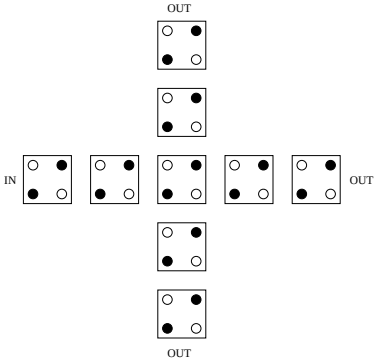
Figure 2: Wire crossing and ripper



(a) One-to-Two Fan-out



(b) One-to-Two Fan-out 2



(c) One-to-Three Fan-out

Figure 3: Fan-outs

do in some cases. Additionally, a series may be considered as the output of a majority gate being fed into the input of a majority gate in the same clocking zone. Dividing this into two clocking zones should fix this minor limitation. In addition, One-to-Three Fan-outs connected to exactly two inputs of a majority gate within the same clocking zone are NOT guaranteed to work, although they do in some cases.

2.3 Design Rule Checker

At some point, a design rule checker will be built that will warn if a design falls into one of the above categories or contains an invalid cell configuration, but until then, the user must ensure that the design contains only acceptable cell configurations.

3 How the Simulator Works

NOTE: This section is not required for an understanding of how to use the simulator.

The purpose of this section is to describe, at a fairly high level, how the simulator works. It is actually not a very complicated concept, though the details can be quite tedious.

Basically, the simulator maintains a list of all the cells with their nearest neighbors. At the beginning of a clock phase, all cells not in the hold phase are set to a state called UND (for undetermined) except for input cells in the switch phase. All input cells in the switch phase and all cells in the hold phase then send messages to their neighbors with their current state, distance, and direction info. These messages are put on a queue and are handled first come first serve. When a cell receives a message, it decides whether or not to change state based on the message received and the design rules. If the cell does change state, it sends a message to all of its neighbors with the same current state, distance, and direction info as well as previous state info. When all the messages (including the new ones that some of the cells place on the queue) are handled, the simulation increments all the clock zones and continues (in the stand-alone version, it asks for the inputs for that cycle).

There is one important wrinkle in this. Any cell that has four neighbors in a plus-sign arrangement is potentially a majority gate. These cells have to be handled slightly differently (why this is so is one of those tedious details). All cells that are potentially majority gates are placed on a second queue which is checked whenever the first queue is empty. When checking the second queue, the first cell that has had exaclty one or three inputs arrive is handled as a fan-out or majority gate respectively. It's messages are placed on the first queue, and the first queue is then handled as before until it is empty again, at which point the simulator checks the second queue again. Once both queues are empty, the simulator can move on to the next cycle.

After the first few cycles, all of the potential majority gates are marked as either fan-outs or majority gates, and thus can be handled more easily and without the overhead of the second queue.

4 QCADesigner Integrated Version Use

4.1 Starting the Simulation

Before doing any simulation, you need to lay out your circuit in QCADesigner. This is described in the *QCADesigner User Guide*. Once the circuit is laid out, you select digital simulator from the "Simulation Engine Setup" option in the "Simulation" menu. Then, from the "Simulation Type Setup" dialog also accessed from the "Simulation" menu, you chose which method of input generation you would like. These are described in Section 4.2. Finally, you click "Start Simulation" in the "Simulation" menu, and wait for the simulation to finish. The simulator will then display a waveform with the inputs and outputs. See Section 4.3 for an explanation of the waveform.

4.2 Input

In the "Simulation Type" dialog box, you select "Exhaustive Verification" or "Vector Table". If you choose "Exhaustive Verificaiton", click "OK", and start the simulation, the simulator will generate all the possible inputs and run the simulation. If you choose "Vector Table", you need to click "Options" in the "Simulation

Type" dialog box. In the window that pops up, you need to select which inputs are active by clicking the input from the "Available" list on the left and clicking "Add". Once you have decided which inputs are active, you need to enter the vectors of data in the large text box at the bottom.

The syntax for the vector data is as follows. Each line that begings with a "#" is a comment and is ignored. All other lines must consist of "0"s and "1"s. The values on a given line correspond to each input for one full clock cycle. They are applied in order from top to bottom. The values need to be given in the order than the inputs appear in the "Activated" list. For example, if inputs A, D, B, and C are activated and in that order from top to bottom, then 0101 applies a "1" to both C and D and a "0" to both A and B, and 1110 applies "1"s to A, D, and B, and a "0" to C.

4.3 Output

The output of the simulation is a set of waveforms for all the inputs and outputs that appears in a new window. They look like this:

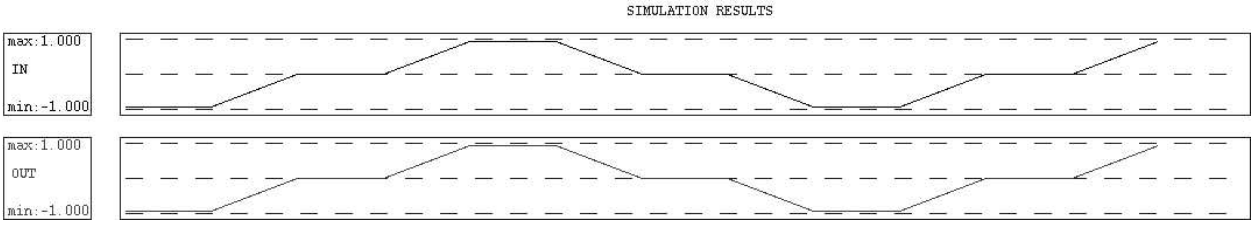


Figure 4: Digital Simulator Waveform

This is a waveform for a simple wire with the input named "IN" and the output named "OUT". The time that the value is horizontal and in the middle of the graph is the relax phase. The time that the value goes from the middle of the graph to either the top or bottom is the switch phase. The time that the value is horizontal at the top or bottom of the graph is the hold phase. The time that the value goes from the top or bottom of the graph back to the middle is the release phase.

Also, note how the digital simulator wave form differs from this waveform from a different simulator:

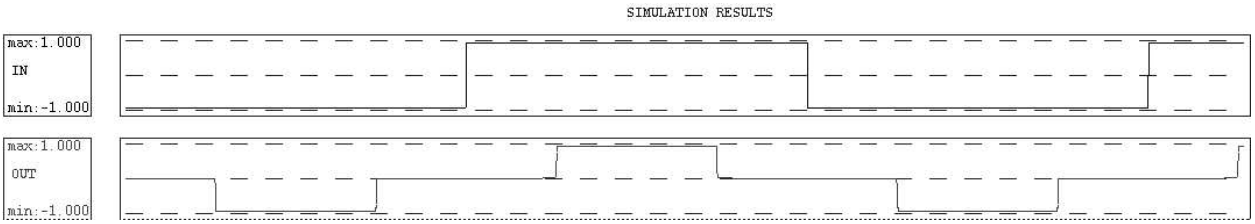


Figure 5: A Different Simulator's Waveform

The main way in which they differ is in the application of the inputs and the slope of the moves from middle to top or bottom. The slope of the other simulator's inputs and outputs is higher because it uses much smaller time steps (something like 1/30 of a clock phase instead of 1 clock phase). The reason the inputs look so different is that in the digital simulator, I tried to make the inputs looks like they would if they were coming from the output of another circuit.

5 Stand-Alone Version Use

NOTE: This version is not "bundled" with QCADesigner. Please email *dantonel@nd.edu* **or** *tdysart@cse.nd.edu* **to request it.**

5.1 Starting the Simulation

In order to use the stand-alone version, you need to make your design in QCADesigner and save it to a file. To start the simulator, you simply type "*digital_simulation*" followed by the file. If the file is in the same directory, you just type the filename, otherwise you have to type the path too (relative path names should work).

5.2 Input

The simulator will give you two options on how you want to enter the inputs. The first is exhaustive verification which is exactly the same as the QCADesigner integrated version (Section 4.2). The second is manual input. If you choose this, the simulator will run in an interactive mode which asks you to press enter at the beginning of each clock phase (or q to quit) and asks you to input a value for all the inputs in the switch phase at that time. You simply enter 1 or 0 for each input.

At some point, a mode of input similar to the vector table mode for the QCADesigner integrated version may be added.

5.3 Output

The output samples in this section are for an XOR gate where all the cells are in the same clocking zone, the inputs are named A and B, and the output is named OUT.

The output from the exhaustive verification is currently printed out like this:

```
A: 0,0,X,X,1,1,X,X,0,0,X,X,1,1,X,X
B: 0,0,X,X,0,0,X,X,1,1,X,X,1,1,X,X
OUT: 0,0,X,X,1,1,X,X,1,1,X,X,0,0,X,X
```

This is obviously not the most user-friendly print out, and a better method will be used once someone decides which one we really want.

The output from the interactive mode is printed at each clock tick. Each input and output is listed with its state and clock zone like this: (Note that the first two lines are the requests for input and the sixth, tenth, fourteenth, and eighteenth lines are the prompts for the user to continue or end the simulation)

```
Enter state (1 or 0) for input A: 0
Enter state (1 or 0) for input B: 1
  A :    LOW   SWITCH
  B :   HIGH   SWITCH
OUT :   HIGH   SWTICH
Press enter to continue or 'q' to quit:
  A :    LOW   HOLD
  B :   HIGH   HOLD
OUT :   HIGH   HOLD
Press enter to continue or 'q' to quit:
  A :      X   RELEASE
  B :      X   RELEASE
OUT :      X   RELEASE
Press enter to continue or 'q' to quit:
  A :      X   RELAX
  B :      X   RELAX
OUT :      X   RELAX
Press enter to continue or 'q' to quit: q
Simulation complete
```

This form lets you easily see everything that is happening to the inputs and outputs at each clock tick.

6 Future Additions

The following is a list of future additions to the simulator. This is by no means a complete list, and items may be added or removed at any time.

- Add design rule checker (High Priority)
- Add something similar to (or exactly the same as) the vector table inputs for the stand-alone version
- Optimize for more speed
- Remove the "Multiple Component Restrictions" (hopefully . . .)

7 Errors

If the simulation should hang or if any of the outputs do not have a changing waveform, then two design changes should be considered. The first is to ensure that the path from input to output is not missing any clock zones (common occurance). The second is to slice the clocking zones into smaller regions.

8 Document Revision History

8.1 May 26, 2004

Changes made by Tim Dysart (*tdysart@cse.nd.edu*).

- Minor editing changes
- Added a clearer description of component limitations (particularly majority gates)
- Added description and picture of rip-cross interaction
- Added Errors section
- Added Document Revision History