Lab04 – FSMs & Cellular automatons

Dragomir MILOJEVIC, v3.3

1 Objectives

Learn how to model and simulate a more complex design using `generate` statement.

2 Exercise

Cellular Automatons (or CAs) is a regular one- (or two-) dimensional grid of *cells*. There are of course multi-dimensional CAs, out of scope here.

Each cell in the CA can be seen as a Finite State Machine (so sequential logic circuit) that calculates the value of the next state depending on the value of its current state and the value of the state of the neighbor cells.

Let's first consider one-dimensional (1D) CA and the cell state dependency depending on the state of its immediate left & right neighbors. For the cells at the boundary (left and right-most cells), you should respectively consider last cell as the left neighbor for the first cell, and first cell as a right cell for the last cell (CA appears as a tore).

As for the cell next state rule (there are many different rules), we suggest to start with Rule 110 specified as follows:

Current state	Next state
111	0
110	1
101	1
100	0
011	1
010	1
001	1
000	0

Table 1: Operation description

"Current state" in the table above is concatenated state vector composed of current cell states: left cell – current cell – right cell. "Next state" is the current cell future states.

You should implement Verilog model of a 1D-CA with an arbitrary number of cells. Of course you should also implement the test-benches enabling us to validate individual cell and complete automaton.

Finally you should be able to observe the "life" of CA, i.e. the patterns composed of each cell state. The patterns that you should observe are neither completely random nor completely repetitive. Localized structures appear and interact in various complicated-looking ways.

1. Write the Verilog model of a single cell and validate the operation using the rule above. Make sure

that the model could accommodate easy implementation of different rules. The initial state value of the cell will be provided through the test-bench. Perform full implementation and note down typical performance/area parameters.

2. Write the Verilog model of a complete 1D automaton, where the size of the CA could be parametrized. For this you will need to use `generate` statements to allow CA size configurability at design time. Validate the operation of the circuit as a whole, and observe the system behavior for different initial conditions.

3. Perform physical implementation (place and route) of three different automata sizes so that you have a gradual increase in the resource utilization (say 10, 35 and 85%) of all FPGA resources. Compare the tendency in area/performance of these three circuits.