Combinational Logic Loop Analysis in Digital Circuit Simulation

Chenyuan Chu Yitian Sun Boxin Zheng

November 13, 2024

1 Complete Flow for Problem 1 to 4

First of all, we present the complete solution flow for problem 1 to 4:

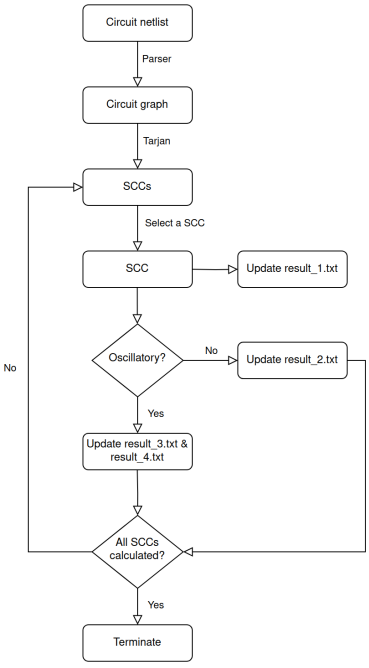

Figure 1: Complete Flow

In the following sections, we will illustrate our solution to each problem in detail.

2 Problem 1

In this section, we applied Tarjan's algorithm [1] to find out all the strongly connected components (SCCs) in the directed graph. The results were printed in SCC units, which can be accomplished in the time complexity of

$$O(|V| + |E|),$$

where V and E denote the amount of vertices and edges in the graph we are interested in, respectively.

3 Problem 2 & 3

In this section, we applied the concept of positive and negative feedback loops to determine the oscillation. For those logic loops that are positive, they cannot be oscillating for sure (see in figure 2 for example). However, the logic loops that are negative (see in figure 3 for example) are not always capable of sustained oscillation due to the interactions between the loops in the same SCC.

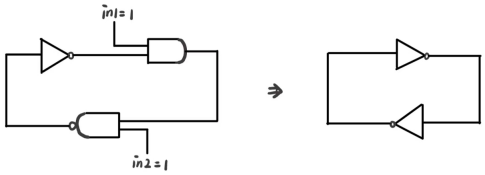

Figure 2: Positive Loop

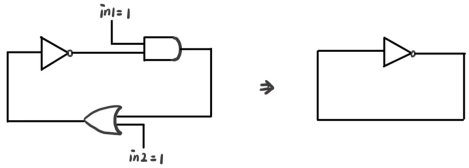

Figure 3: Negative Loop

First, we give the flowchart of problem 2 and 3:

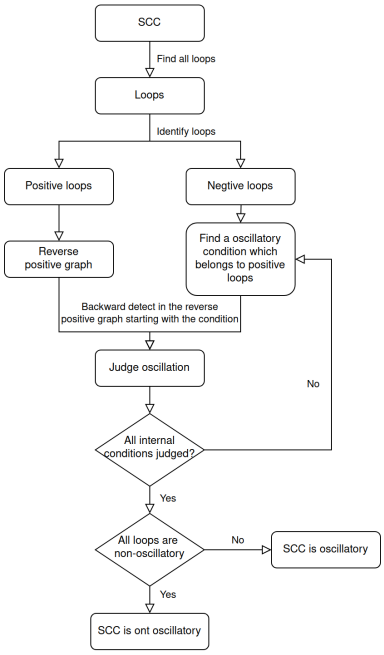

Figure 4: Flowchart of Problem 2 and 3

Realizing the overestimation of the amount of loops with sustained oscillation, we implemented the detection methodology based on the work mentioned before, where we obtained the necessary conditions for oscillation at first, and then we brought these conditions into the algorithm one by one for backwards detection.
For example, consider the negative loop we currently detected, given a condition that the signal of the corresponding wire should satisfy, there are three possibilities:

1. The signal comes from the input of the entire SCC, and under this possibility the condition can be satisfied for sure;

2. The signal comes from another negative loop in the same SCC, and under this possibility the condition cannot be satisfied because the fixed wire in a negative loop is impossible;

3. The signal comes from a positive loop in the same SCC, and under this possibility we did backwards detection recursively.

Moreover, for a oscillating loop, the condition for the SCC to be oscillating is proven to be unique. The prove is shown below. For the SCC that we obtained from a certain testcase shown in figure 5, we stipulate that capital letters represent logic gates and lowercase letters represent input signals that are not included in the given SCC. We call such signals external signals. The loop condition we are interested in is that under what inputs of these external signals, the given SCC is oscillating. In the combinational logic circuit below, the external signals are a, b, c, d, e, f.

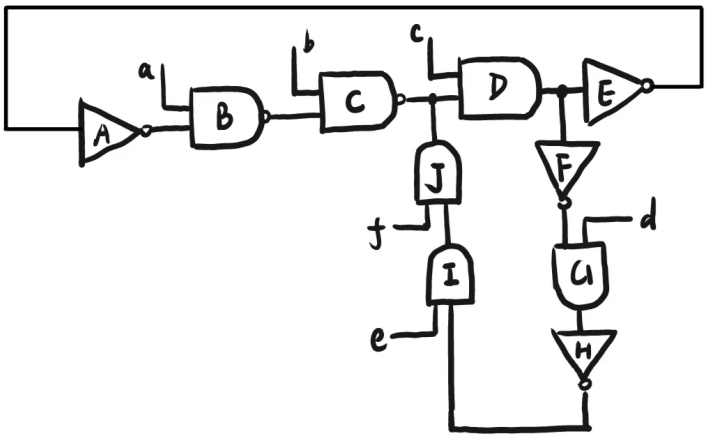

Figure 5: An SCC for Example

For the negative loop composed of gates C, F, G, H, I, J, the necessary condition for this loop to be oscillating is given by $b = 1, c = 1, d = 1, e = 1$ (the influence of other loops is ignored). However, consider the positive loop composed of gates A, B, C, D, E, the signal a should be logic 1, otherwise the output of gate B is fixed by 1, which violates the definition of an oscillating loop. Now focus on the signal c, it can only be logic 1, otherwise the output of gate D is fixed by 0, the output of gate E is fixed by 1, the output of gate A is fixed by 0, and the output of gate B is fixed, which also violates the definition of an oscillating loop in the end.
Now we promote this theory to any other SCC.

definition: We define that the 2-input gates whose off-path input is an external signal and output directly enters another negative loop in a positive

loop (if exists) as the **Backwards Gates** (BGs) and the other 2-input gates with external signal input but do not directly connect another negative loop as **Don't Care Gates** ($DCGs$).
It is obvious that if an SCC contains more than 1 loop, for example, 2 loops, then these 2 loops must share some gates. From the backwards detection we proposed, all the external signals of BGs can be determined. Now consider the external signals of $DCGs$. For a DCG, if the external signal leads to a fixed output, then under any given external signals, the state of the positive loop is fixed, and thus the logic values of shared edges between the positive loop and negative loop are fixed, which violates the definition of the oscillating loop. Now we can say that the external signals of DCGs must lead to uncertain output. For the possible types if gates in this problem, the external signals are given by:

Gate Type	External Signal (Logic Value)
AND2	1
OR2	0
NAND2	1

4 Problem 4

In this section, the concept of partial oscillation is proposed. We define the oscillating part of the given SCC as $OSC-SCC$. We can extract the $OSC-SCC$ from the given SCC, and the problem is transformed into deleting the minimum number of nodes so that no loop exists in $OSC-SCC$. Now we can discuss all possible scenarios:

1. Only one negative loop in an SCC, then randomly break an edge in this loop.

2. More than 1 negative loop in an SCC. The in-degree of all the nodes in $OSC-SCC$ is no more than 2 because all the gates are 2-input. Therefore it is impossible that more than 1 negative loop share a single edge.

   Now the problem is transformed into finding out the gates whose both inputs are in the given SCC.

Note that the positions of the registers we inserted into the circuit are stored in an *unordered_set* in our cpp files in case of possible duplication.

5 Problem 5

5.1 Requirements and Core Idea

This problem involves transforming the given original circuit to a new circuit (both are .v file), which meets the following three requirements:

1. It can avoid oscillation given every possible inputs;
2. Wires in the original circuit have a one-to-one mapping in the new circuit (two circuit share the same input), and when the input does not cause oscillation in the original circuit, each wire pair has the same value;
3. A wire named $OscFlag$ will be created in the new circuit for predicting oscillation, whose value will be 1 if the input cause oscillation in the original circuit, and 0 otherwise.

Given the requirements above, we come up with a simple and intuitive idea: we can cut off some wires to destruct positive loops so that oscillation will never occur; meanwhile, these wires we cut off serve as the inputs of other gates $g_1, g_2, ..., g_m$, so we must use something to replace them. Fortunately, it is obvious that when the input does not cause oscillation in the original circuit, each wire can be described as a logic function of input wires, like $F(I_1, I_2, ..., I_n)$, here n is the number of input wires and $I_i (i = 1, 2, ..., n)$ is their names. These logic functions can take the role of inputs for gates $g_1, g_2, ..., g_m$, and via this method we can guarantee the new circuit's function is the same as the original one. Similarly, the $OscFlag$ signal can be described as a logic function of input wires as well.
In the following subsection, we will explain how this idea can be implemented precisely and elegantly in detail.

5.2 Step 1: Analysis of the Case

The original circuit $Q5\_testcase.v$ has 8 inputs, 14 logic gates and 6 logic loops, whose graph is shown in figure 6.

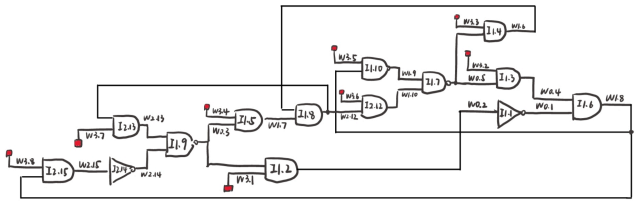

Figure 6: Circuit

Among the 6 loops, there are 4 negative loops that may cause oscillation, which are named $Loop1 - Loop4$ and drawn in deep blue, yellow, purple and red respectively. They are shown in figure 7, whose wires are listed below.

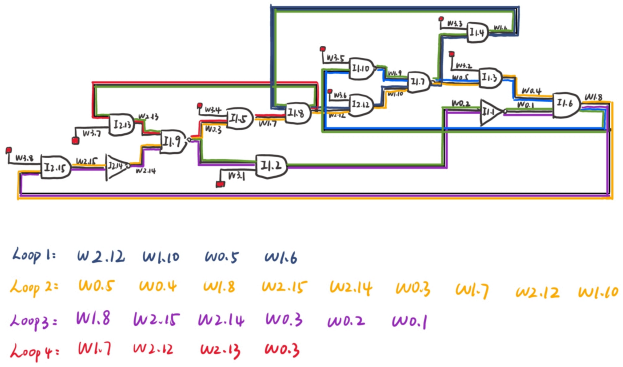

Figure 7: Loops

Given the graph above, we have come up with the following simple but intuitive idea: cut off some wires in the positive loop, and employ the logic function to take on the role of inputs for other gates.
To realize this idea, we must conduct simulation with Verisim first.

5.3 Step 2: Simulation with Verisim

To begin with, we conduct simulation on the original circuit. Apparently, there are some inputs causing oscillation, which will force the simulation to abort. But as long as we change the initial input to this oscillating state, Verisim will output oscillating results in consecutive states. In the example below, Verisim senses the oscillation in input [00110100], so it just stop at the [11010100] state:

```
Time=  0, Signals: [0 0 0 0 0 0 0 0] [1 0 1 0 1 0 0 0 1 0 0 0 1 0]
Time=  1, Signals: [1 0 0 0 0 0 0 0] [0 1 1 0 1 0 0 0 1 0 0 0 1 0]
Time=  2, Signals: [0 1 0 0 0 0 0 0] [1 0 1 1 1 0 0 1 1 0 0 0 1 0]
Time=  3, Signals: [1 1 0 0 0 0 0 0] [0 1 1 1 1 0 0 0 1 0 0 0 1 0]
Time=  4, Signals: [0 0 1 0 0 0 0 0] [1 0 1 0 1 1 0 0 1 0 0 0 1 0]
Time=  5, Signals: [1 0 1 0 0 0 0 0] [0 1 1 0 1 1 0 0 1 0 0 0 1 0]
Time=  6, Signals: [0 1 1 0 0 0 0 0] [1 0 1 1 1 1 0 1 1 0 0 0 1 0]
Time=  7, Signals: [1 1 1 0 0 0 0 0] [0 1 1 1 1 1 0 0 1 0 0 0 1 0]
Time=  8, Signals: [0 0 0 1 0 0 0 0] [1 0 1 0 1 0 1 0 1 0 0 0 1 0]
Time=  9, Signals: [1 0 0 1 0 0 0 0] [0 1 1 0 1 0 1 0 1 0 0 0 1 0]
Time= 10, Signals: [0 1 0 1 0 0 0 0] [1 0 1 1 1 0 1 1 1 0 0 0 1 0]
Time= 11, Signals: [1 1 0 1 0 0 0 0] [0 1 1 1 1 0 1 0 1 0 0 0 1 0]
Time= 12, Signals: [0 0 1 1 0 0 0 0] [1 0 1 0 1 1 1 0 1 0 1 0 1 0]
Time= 13, Signals: [1 0 1 1 0 0 0 0] [0 1 1 0 1 1 1 0 1 0 1 0 1 0]
Time= 14, Signals: [0 1 1 1 0 0 0 0] [1 0 1 1 1 1 1 1 1 0 1 0 1 0]
Time= 15, Signals: [1 1 1 1 0 0 0 0] [0 1 1 1 1 1 1 0 1 0 1 0 1 0]
Time= 16, Signals: [0 0 0 0 1 0 0 0] [1 0 1 0 1 0 0 0 1 0 0 0 1 0]
Time= 17, Signals: [1 0 0 0 1 0 0 0] [0 1 1 0 1 0 0 0 1 0 0 0 1 0]
Time= 18, Signals: [0 1 0 0 1 0 0 0] [1 0 1 1 1 0 0 1 0 0 0 0 1 0]
Time= 19, Signals: [1 1 0 0 1 0 0 0] [0 1 1 1 1 0 0 0 1 0 0 0 1 0]
Time= 20, Signals: [0 0 1 0 1 0 0 0] [1 0 1 0 1 1 0 0 1 0 0 0 1 0]
Time= 21, Signals: [1 0 1 0 1 0 0 0] [0 1 1 0 1 1 0 0 1 0 0 0 1 0]
Time= 22, Signals: [0 1 1 0 1 0 0 0] [1 0 1 1 1 1 0 1 0 0 0 0 1 0]
Time= 23, Signals: [1 1 1 0 1 0 0 0] [0 1 1 1 1 1 0 0 1 0 0 0 1 0]
Time= 24, Signals: [0 0 0 1 1 0 0 0] [1 0 1 0 1 0 1 0 1 0 0 0 1 0]
Time= 25, Signals: [1 0 0 1 1 0 0 0] [0 1 1 0 1 0 1 0 1 0 0 0 1 0]
Time= 26, Signals: [0 1 0 1 1 0 0 0] [1 0 1 1 1 0 1 1 0 0 0 0 1 0]
Time= 27, Signals: [1 1 0 1 1 0 0 0] [0 1 1 1 1 0 1 0 1 0 0 0 1 0]
Time= 28, Signals: [0 0 1 1 1 0 0 0] [1 0 1 0 1 1 1 0 1 0 1 0 1 0]
Time= 29, Signals: [1 0 1 1 1 0 0 0] [0 1 1 0 1 1 1 0 1 0 1 0 1 0]
Time= 30, Signals: [0 1 1 1 1 0 0 0] [1 0 1 1 1 1 1 1 0 0 1 0 1 0]
Time= 31, Signals: [1 1 1 1 1 0 0 0] [0 1 1 1 1 1 1 0 1 0 1 0 1 0]
Time= 32, Signals: [0 0 0 0 0 1 0 0] [1 0 1 0 1 0 0 0 1 0 0 0 1 0]
Time= 33, Signals: [1 0 0 0 0 1 0 0] [0 1 1 0 1 0 0 0 1 0 0 0 1 0]
Time= 34, Signals: [0 1 0 0 0 1 0 0] [1 0 1 1 1 0 0 1 1 0 0 0 1 0]
Time= 35, Signals: [1 1 0 0 0 1 0 0] [0 1 1 1 1 0 0 0 1 0 0 0 1 0]
Time= 36, Signals: [0 0 1 0 0 1 0 0] [1 0 1 0 1 1 0 0 1 0 0 0 1 0]
Time= 37, Signals: [1 0 1 0 0 1 0 0] [0 1 1 0 1 1 0 0 1 0 0 0 1 0]
Time= 38, Signals: [0 1 1 0 0 1 0 0] [1 0 1 1 1 1 0 1 1 0 0 0 1 0]
Time= 39, Signals: [1 1 1 0 0 1 0 0] [0 1 1 1 1 1 0 0 1 0 0 0 1 0]
Time= 40, Signals: [0 0 0 1 0 1 0 0] [1 0 1 0 1 0 1 0 1 0 0 0 1 0]
Time= 41, Signals: [1 0 0 1 0 1 0 0] [0 1 1 0 1 0 1 0 1 0 0 0 1 0]
Time= 42, Signals: [0 1 0 1 0 1 0 0] [1 0 1 1 1 0 1 1 1 0 0 0 1 0]
Time= 43, Signals: [1 1 0 1 0 1 0 0] [0 1 1 1 1 0 1 0 1 0 0 0 1 0]
```

Figure 8: Stop at Oscillation Point

Then we can change the initial input from [00000000] to [00110100]:

```
initial begin
    $deposit(I0.w_002_013, 1'b1);
    r1 = 1'b0;
    r2 = 1'b0;
    r3 = 1'b1;
    r4 = 1'b1;
    r5 = 1'b0;
    r6 = 1'b1;
    r7 = 1'b0;
    r8 = 1'b0;
    $monitor("Time=%-3t, Signals: [%b %b %b %b %b %b
        %b %b] [%b %b %b %b %b %b %b %b %b %b %b %b
       %b %b]", $time,
```

```
12                      w_003_001, w_003_002, w_003_003,
                            w_003_004, w_003_005, w_003_006,
                            w_003_007, w_003_008,
13                      I0.w_000_001, I0.w_000_002, I0.
                            w_000_003, I0.w_000_004, I0.
                            w_000_005,
14                      I0.w_001_006, I0.w_001_007, I0.
                            w_001_008, I0.w_001_009, I0.
                            w_001_010,
15                      I0.w_002_012, I0.w_002_013, I0.
                            w_002_014, I0.w_002_015);
16      #260;
17      $finish;
18  end
```

In this way, we can obtain the result of simulation when oscillation occurs:

```
Time=  0, Signals: [0 0 1 1 0 1 0 0]  [1 0 1 0 x x 1 0 1 x x 0 1 0]
Time=  1, Signals: [1 0 1 1 0 1 0 0]  [0 1 1 0 x x 1 0 1 x x 0 1 0]
Time=  2, Signals: [0 1 1 1 0 1 0 0]  [1 0 1 x x x 1 x 1 x x 0 1 0]
Time=  3, Signals: [1 1 1 1 0 1 0 0]  [0 1 1 x x x 1 0 1 x x 0 1 0]
```

Figure 9: Result

Here, "x" indicate the oscillating signal, and we can find Verisim stops when switch from the oscillatory input to non-oscillatory input. So we should change the initial input once again to proceed simulation:

```
1  initial begin
2      $deposit(I0.w_002_013, 1'b1);
3      r1 = 1'b0;
4      r2 = 1'b0;
5      r3 = 1'b0;
6      r4 = 1'b0;
7      r5 = 1'b1;
8      r6 = 1'b1;
9      r7 = 1'b0;
10     r8 = 1'b0;
11     $monitor("Time=%-3t, Signals: [%b %b %b %b %b %b
           %b %b] [%b %b %b %b %b %b %b %b %b %b %b %b
          %b %b]", $time,
```

```
12                      w_003_001, w_003_002, w_003_003,
                            w_003_004, w_003_005, w_003_006,
                            w_003_007, w_003_008,
13                      I0.w_000_001, I0.w_000_002, I0.
                            w_000_003, I0.w_000_004, I0.
                            w_000_005,
14                      I0.w_001_006, I0.w_001_007, I0.
                            w_001_008, I0.w_001_009, I0.
                            w_001_010,
15                      I0.w_002_012, I0.w_002_013, I0.
                            w_002_014, I0.w_002_015);
16      #260;
17      $finish;
18  end
```

We can obtain the result:

```
Time=  0, Signals: [0 0 0 0 1 1 0 0]  [1 0 1 0 1 0 0 0 1 0 0 0 1 0]
Time=  1, Signals: [1 0 0 0 1 1 0 0]  [0 1 1 0 1 0 0 0 1 0 0 0 1 0]
Time=  2, Signals: [0 1 0 0 1 1 0 0]  [1 0 1 1 1 0 0 1 0 0 0 0 1 0]
Time=  3, Signals: [1 1 0 0 1 1 0 0]  [0 1 1 1 1 0 0 0 1 0 0 0 1 0]
Time=  4, Signals: [0 0 1 0 1 1 0 0]  [1 0 1 0 1 1 0 0 1 0 0 0 1 0]
Time=  5, Signals: [1 0 1 0 1 1 0 0]  [0 1 1 0 1 1 0 0 1 0 0 0 1 0]
Time=  6, Signals: [0 1 1 0 1 1 0 0]  [1 0 1 1 1 1 0 1 0 0 0 0 1 0]
Time=  7, Signals: [1 1 1 0 1 1 0 0]  [0 1 1 1 1 1 0 0 1 0 0 0 1 0]
Time=  8, Signals: [0 0 0 1 1 1 0 0]  [1 0 1 0 1 0 1 0 1 0 0 0 1 0]
Time=  9, Signals: [1 0 0 1 1 1 0 0]  [0 1 1 0 1 0 1 0 1 0 0 0 1 0]
Time= 10, Signals: [0 1 0 1 1 1 0 0]  [1 0 1 1 1 0 1 1 0 0 0 0 1 0]
Time= 11, Signals: [1 1 0 1 1 1 0 0]  [0 1 1 1 1 0 1 0 1 0 0 0 1 0]
```

Figure 10: Result

If we keep changing the initial input to proceed simulation, we can ultimately enumerate all 2^8 inputs and their corresponding results. The complete result file is attached in the appendix.

5.4 Step 3: Find the Pattern

In this step, we will explain the detailed realization of our intuitive idea: find the pattern between input pin and specific wires, and convert it to a logic function.
First and foremost, we should list the wires of the potential oscillating positive loops:

1. **Loop1**: $w\_000\_005, w\_001\_006, w\_001\_010, w\_002\_012$

2. **Loop2**: $w\_000\_003, w\_000\_004, w\_000\_005, w\_001\_007, w\_001\_008, w\_001\_010, w\_002\_012, w\_002\_014, w\_002\_015$
3. **Loop3**: $w\_000\_001, w\_000\_002, w\_000\_003, w\_001\_008, w\_002\_014, w\_002\_015$
4. **Loop4**: $w\_000\_003, w\_001\_007, w\_002\_012, w\_002\_013$

We observe that Loop1 and Loop4 share the wire w_002_012, while Loop2 and Loop3 share the wire w_002_015. That means, if we cut off these two wires and replace them with two designed logic function, we can guarantee the new circuit share the same result with the original circuit while avoid oscillation. So we will find patterns to represent these logic functions next.

5.4.1 Logic Function for w_002_012

Let's focus on the third and fourth columns of the input, as well as the fourth column from the end of the output:

```
Time=  0, Signals: [0 0 0 0 0 0 0 0]  [1 0 1 0 1 0 0 0 1 0 0 0 1 0]
Time=  1, Signals: [1 0 0 0 0 0 0 0]  [0 1 1 0 1 0 0 0 1 0 0 0 1 0]
Time=  2, Signals: [0 1 0 0 0 0 0 0]  [1 0 1 1 1 0 0 1 1 0 0 0 1 0]
Time=  3, Signals: [1 1 0 0 0 0 0 0]  [0 1 1 1 1 0 0 0 1 0 0 0 1 0]
Time=  4, Signals: [0 0 1 0 0 0 0 0]  [1 0 1 0 1 1 0 0 1 0 0 0 1 0]
Time=  5, Signals: [1 0 1 0 0 0 0 0]  [0 1 1 0 1 1 0 0 1 0 0 0 1 0]
Time=  6, Signals: [0 1 1 0 0 0 0 0]  [1 0 1 1 1 1 1 0 1 1 0 0 1 0]
Time=  7, Signals: [1 1 1 0 0 0 0 0]  [0 1 1 1 1 1 1 0 0 1 0 0 1 0]
Time=  8, Signals: [0 0 0 1 0 0 0 0]  [1 0 1 0 1 0 1 0 1 0 0 0 1 0]
Time=  9, Signals: [1 0 0 1 0 0 0 0]  [0 1 1 0 1 0 1 0 1 0 0 0 1 0]
Time= 10, Signals: [0 1 0 1 0 0 0 0]  [1 0 1 1 1 0 1 1 1 0 0 0 1 0]
Time= 11, Signals: [1 1 0 1 0 0 0 0]  [0 1 1 1 1 0 1 0 1 0 0 0 1 0]
Time= 12, Signals: [0 0 1 1 0 0 0 0]  [1 0 1 0 1 1 1 0 1 0 1 0 1 0]
Time= 13, Signals: [1 0 1 1 0 0 0 0]  [0 1 1 0 1 1 1 0 1 0 1 0 1 0]
Time= 14, Signals: [0 1 1 1 0 0 0 0]  [1 0 1 1 1 1 1 1 1 0 1 0 1 0]
Time= 15, Signals: [1 1 1 1 0 0 0 0]  [0 1 1 1 1 1 1 1 0 1 0 1 0 1 0]
Time= 16, Signals: [0 0 0 0 1 0 0 0]  [1 0 1 0 1 0 0 0 1 0 0 0 1 0]
Time= 17, Signals: [1 0 0 0 1 0 0 0]  [0 1 1 0 1 0 0 0 1 0 0 0 1 0]
Time= 18, Signals: [0 1 0 0 1 0 0 0]  [1 0 1 1 1 0 0 1 0 0 0 0 1 0]
Time= 19, Signals: [1 1 0 0 1 0 0 0]  [0 1 1 1 1 0 0 0 1 0 0 0 1 0]
Time= 20, Signals: [0 0 1 0 1 0 0 0]  [1 0 1 0 1 1 0 0 1 0 0 0 1 0]
Time= 21, Signals: [1 0 1 0 1 0 0 0]  [0 1 1 0 1 1 0 0 1 0 0 0 1 0]
Time= 22, Signals: [0 1 1 0 1 0 0 0]  [1 0 1 1 1 1 1 0 1 0 0 0 1 0]
Time= 23, Signals: [1 1 1 0 1 0 0 0]  [0 1 1 1 1 1 1 0 0 1 0 0 1 0]
Time= 24, Signals: [0 0 0 1 1 0 0 0]  [1 0 1 0 1 0 1 0 1 0 0 0 1 0]
Time= 25, Signals: [1 0 0 1 1 0 0 0]  [0 1 1 0 1 0 1 0 1 0 0 0 1 0]
Time= 26, Signals: [0 1 0 1 1 0 0 0]  [1 0 1 1 1 0 1 1 0 0 0 0 1 0]
Time= 27, Signals: [1 1 0 1 1 0 0 0]  [0 1 1 1 1 0 1 0 1 0 0 0 1 0]
Time= 28, Signals: [0 0 1 1 1 0 0 0]  [1 0 1 0 1 1 1 0 1 0 1 0 1 0]
Time= 29, Signals: [1 0 1 1 1 0 0 0]  [0 1 1 0 1 1 1 0 1 0 1 0 1 0]
Time= 30, Signals: [0 1 1 1 1 0 0 0]  [1 0 1 1 1 1 1 1 0 0 1 0 1 0]
Time= 31, Signals: [1 1 1 1 1 0 0 0]  [0 1 1 1 1 1 1 1 0 1 0 1 0 1 0]
Time= 32, Signals: [0 0 0 0 0 1 0 0]  [1 0 1 0 1 0 0 0 1 0 0 0 1 0]
Time= 33, Signals: [1 0 0 0 0 1 0 0]  [0 1 1 0 1 0 0 0 1 0 0 0 1 0]
Time= 34, Signals: [0 1 0 0 0 1 0 0]  [1 0 1 1 1 0 0 1 1 0 0 0 1 0]
Time= 35, Signals: [1 1 0 0 0 1 0 0]  [0 1 1 1 1 0 0 0 1 0 0 0 1 0]
```

Figure 11: Pattern of w12

Surprisingly, we can find a simple pattern between input wire w_003_003 / w_003_004 and wire w_002_012, which can be describe as an "AND" operation. Although the result picture above displays a fraction of the whole result only, we can confirm this conclusion using the complete result in the appendix.
In Verilog, we can use a new wire named $new\_w\_002\_012$ to replace the original w_002_012 with an elegant *assign* operation:

```
1 assign new_w_002_012 = w_003_003 & w_003_004;
```

Now w_002_012 only serves as the output of gate $I001\_008$, while the $new\_w\_002\_012$ takes on the role of the input for gate $I002\_013$ and $I002\_014$.

```
1 and2     I001_008(w_002_012, w_001_006, w_001_007);
2 and2     I002_013(w_002_013, w_003_007, new_w_002_012
      );
3 not1     I002_014(w_002_014, new_w_002_015);
```

The following picture clearly clarify the modification (the blue part):

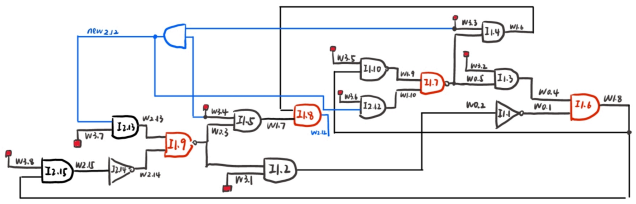

Figure 12: Modified Circuit

5.4.2 Logic Function for w_002_015

Following the analysis of w_002_012, we can get similar conclusion from the result below:

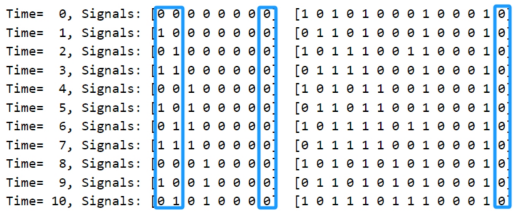

Figure 13: Pattern of w15 (1)

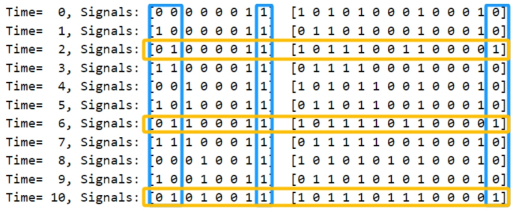

Figure 14: Pattern of w15 (2)

The conclusion can be expressed as a logic function in Verilog as follows:

```
assign new_w_002_015 = ~w_003_001 & w_003_002 &
    w_003_008;
```

And the final modified circuit is shown below:

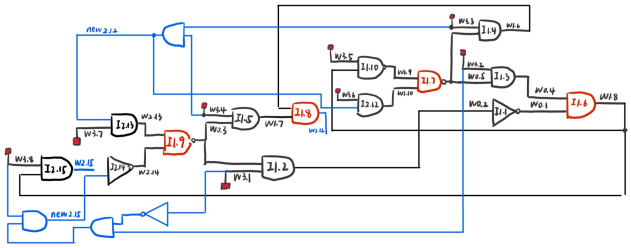

Figure 15: Final Circuit

5.4.3 Logic Function for *OscFlag*

In this subsubsection, we realize the function of predicting oscillation: when the input causes the original circuit to oscillate, a wire named *OscFlag* in the modified circuit will be 1, and otherwise 0.

Following the analysis in section 1, we can get similar conclusion from the result below:

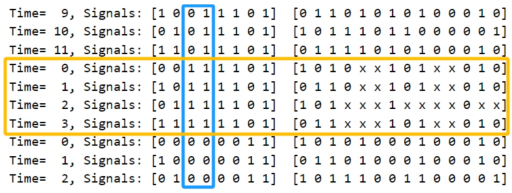

Figure 16: Pattern of OscFlag (1)

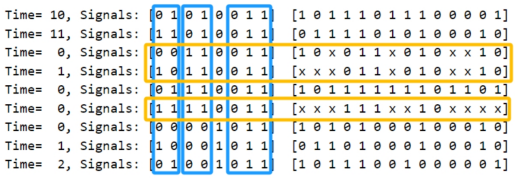

Figure 17: Pattern of OscFlag (2)

The conclusion can be expressed as a logic function in Verilog as follows:

```
assign OscFlag = w_003_003 & w_003_004 & (w_003_006
    | w_003_007) & ~(~w_003_001 & w_003_002 & ~
    w_003_006 & w_003_007 & w_003_008);
```

Next, we add *OscFlag* to the simulation list:

```
initial begin
    $deposit(I0.w_002_013, 1'b1);
    r1 = 1'b0;
    r2 = 1'b0;
    r3 = 1'b0;
    r4 = 1'b0;
    r5 = 1'b0;
    r6 = 1'b0;
    r7 = 1'b0;
    r8 = 1'b0;
    $monitor("  Time=%-3t, Signals: [%b %b %b %b %b
        %b %b %b] [%b %b %b %b %b %b %b %b %b %b %b
        %b %b %b] [%b]", $time,
```

```
12                      w_003_001, w_003_002, w_003_003,
                            w_003_004, w_003_005, w_003_006,
                            w_003_007, w_003_008,
13                      I0.w_000_001, I0.w_000_002, I0.
                            w_000_003, I0.w_000_004, I0.
                            w_000_005,
14                      I0.w_001_006, I0.w_001_007, I0.
                            w_001_008, I0.w_001_009, I0.
                            w_001_010,
15                      I0.w_002_012, I0.w_002_013, I0.
                            w_002_014, I0.w_002_015,
16                      I0.OscFlag    );
17      #260;
18      $finish;
19  end
```

And we get the result like the figure below, where the last [0] or [1] is the value of *OscFlag*:

```
Time=218, Signals: [0 1 0 1 1 0 1 1]  [1 0 1 1 1 0 1 1 0 0 0 0 0 1] [0]
Time=219, Signals: [1 1 0 1 1 0 1 1]  [0 1 1 1 1 0 1 0 1 0 0 0 1 0] [0]
Time=220, Signals: [0 0 1 1 1 0 1 1]  [1 0 0 0 1 1 0 0 1 0 0 1 1 0] [1]
Time=221, Signals: [1 0 1 1 1 0 1 1]  [1 0 0 0 1 1 0 0 1 0 0 1 1 0] [1]
Time=222, Signals: [0 1 1 1 1 0 1 1]  [1 0 1 1 1 1 1 1 0 0 1 1 0 1] [0]
Time=223, Signals: [1 1 1 1 1 0 1 1]  [1 0 0 1 1 1 0 1 0 0 0 1 1 1] [1]
Time=224, Signals: [0 0 0 0 0 1 1 1]  [1 0 1 0 1 0 0 0 1 0 0 0 1 0] [0]
Time=225, Signals: [1 0 0 0 0 1 1 1]  [0 1 1 0 1 0 0 0 1 0 0 0 1 0] [0]
Time=226, Signals: [0 1 0 0 0 1 1 1]  [1 0 1 1 1 0 0 1 1 0 0 0 0 1] [0]
Time=227, Signals: [1 1 0 0 0 1 1 1]  [0 1 1 1 1 0 0 0 1 0 0 0 1 0] [0]
```

Figure 18: Final Result

The final circuit with predicting function is shown below:

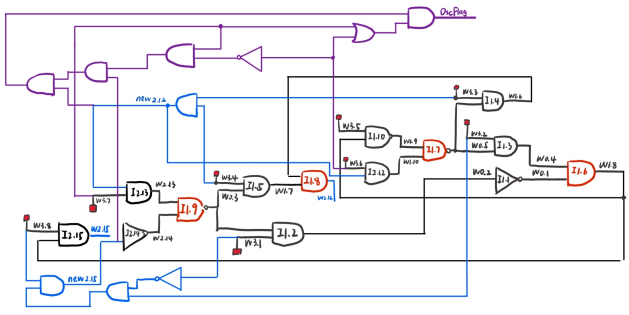

Figure 19: Final Circuit with Predicting Function

5.5 Step 4: Generalized Solution

In step 1 to 3, we obtain a powerful circuit without oscillation possibility, while maintaining the same function as the original circuit and equipped with the predicting function. Despite the success in outputting the right result, we try to give a more generalized solution, which can be applied to circuits with various structures and sizes.

5.5.1 Circuit Topology Analysis

Apparently, a logic function is entailed to precisely describe the relationship between input wires and our target wire. We should realize one important point: not all the input wires are needed when building the logic function. Consequently, before the logic function simplification step in the next subsection, we try to minimize the number of inputs involved because it will lead to an exponential growth in the possible scenarios of logical functions. To do this, we analyze the topology of circuit following the designed algorithm flow.
For cutting off wires (like requirement 1 and 2), we can adopt the following flow:

Algorithm 1: Algorithm for Cutting off Wires

Input: Input wires $I_1, I_2, ..., I_n$
Output: Logic Function F_w
Use Verisim to get simulation result;
Construct a Priority List for input wires based on their distance to the target wire (when the distance equals, arrange them in ASCII code character order);
$i = 0, F_0 = none$;
while True **do**
  Add next input wire I_i from the top of Priority List into current logic function $F_{i-1}(I_1, I_2, ..., I_{i-1})$;
  Construct and minimize F_i;
  if F_i can satisfy the simulation result **then**
    $F_w = F_i$;
    break;
  else
    $i++$;
  end if
end while

Now we explain the algorithm flow in the following example: as figure 20 shows, the purple wire w_002_012 is the target wire, so the first priority in the Priority List is w_003_003, and then w_003_003, which are the closest two input wires to the target wire (drawn in red). Searching backwards, the following priority will belong to w_003_005, w_003_006 and w_003_007, which are drawn in blue.

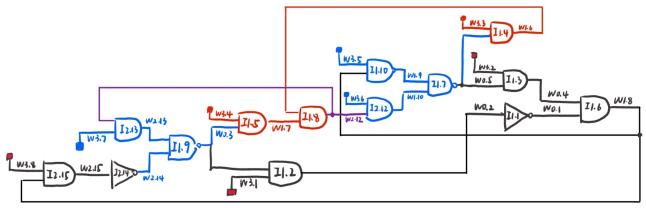

Figure 20: Example for Cutting off Wires

Next, we design a similar algorithm for finding the minimum inputs for the logic function of $OscFlag$:

Algorithm 2: Algorithm for *OscFlag*

Input: Input wires $I_1, I_2, ..., I_n$
Output: Logic Function F_o
Use Verisim to get simulation result;
Find gates with multiple fanouts or located in the intersection of positive loops (core gates);
Construct a Priority List for input wires based on their distance to the core gates (when the distance equals, arrange them in ASCII code character order);
$i = 0, F_0 = none$;
while True **do**
  Add next input wire I_i from the top of Priority List into current logic function $F_{i-1}(I_1, I_2, ..., I_{i-1})$;
  Construct and minimize F_i;
  if F_i can satisfy the simulation result **then**
    $F_o = F_i$;
    break;
  else
    $i++$;
  end if
end while

We can demonstrate the algorithm flow in the following example: as figure 21 shows, first we find core gates, which are the gates with multiple fanouts or located in the intersection of positive loops, and they are drawn in red. Next, similar to the algorithm for cutting off wires, we construct Priority List based on the distance to core gates. The first tier in the Priority List includes the input wires within dashed boxes: w_003_003, w_003_004, w_003_005, w_003_006 and w_003_007.

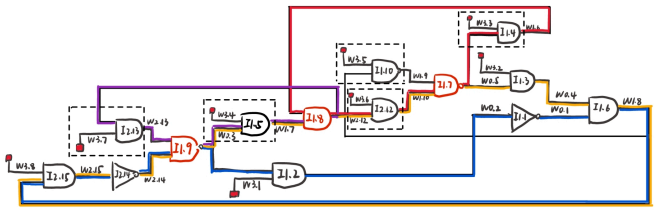

Figure 21: Example for *OscFlag*

Actually, we can refer to the logic functions we construct in *result5.v*:

```
assign new_w_002_012 = w_003_003 & w_003_004;
assign new_w_002_015 = ~w_003_001 & w_003_002 &
    w_003_008;
assign OscFlag = w_003_003 & w_003_004 & (w_003_006
    | w_003_007) & ~(~w_003_001 & w_003_002 & ~
    w_003_006 & w_003_007 & w_003_008);
```

Obviously, these logic functions involve a fraction of input wires only, rather than all of them, which confirms our theorem.

5.5.2 Logic Function Construction and Minimization

As the flow above shows, the step of logic function construction and minimization is the key to the algorithm, whose efficiency mainly accounts for the overall performance. So we should really emphasize its implementation. Because the number of inputs n is finite, $k(\leq 2^n)$ effective input-output pairs can be always completely listed as a LUT. According to this LUT, the output can be definitely represented as the sum of minterms. Next, with the assistance of some mature algorithm, such as Quine-McCluskey [2] and ESPRESSO [3], the logic function can be effectively minimized.
Finally, we will obtain desired logic functions with conciseness, which can be implemented efficiently by fewer gates.

6 Innovations

1. Efficient algorithm for Tarjan's algorithm to find SCCs.
2. The backwards detection mechanism is applied to avoid the interactions between logical loops.
3. Proposed *BGs* and *DCGs* to obtain the oscillating condition of an SCC, which is proven to be stable in performance.
4. An intuitive and universal method to transforming the circuit to one without oscillation, meanwhile holds the prediction function.

References

[1] Tarjan, R. (1972). Depth-First search and linear graph algorithms. SIAM Journal on Computing, 1(2), 146–160. https://doi.org/10.1137/0201010

[2] McCluskey, E. J. “Minimization of Boolean Functions*.” Bell System Technical Journal, vol. 35, no. 6, Nov. 1956, pp. 1417–44. https://doi.org/10.1002/j.1538-7305.1956.tb03835.x.

[3] McGeer, Patrick, et al. ”ESPRESSO-SIGNATURE: A new exact minimizer for logic functions.” Proceedings of the 30th international design automation conference. 1993.

A Appendix A: Standard Result (simulation of Q5_testcase.v)

Time= 0, Signals: [0 0 0 0 0 0 0 0] [1 0 1 0 1 0 0 0 1 0 0 0 1 0]
Time= 1, Signals: [1 0 0 0 0 0 0 0] [0 1 1 0 1 0 0 0 1 0 0 0 1 0]
Time= 2, Signals: [0 1 0 0 0 0 0 0] [1 0 1 1 1 0 0 1 1 0 0 0 1 0]
Time= 3, Signals: [1 1 0 0 0 0 0 0] [0 1 1 1 1 0 0 0 1 0 0 0 1 0]
Time= 4, Signals: [0 0 1 0 0 0 0 0] [1 0 1 0 1 1 0 0 1 0 0 0 1 0]
Time= 5, Signals: [1 0 1 0 0 0 0 0] [0 1 1 0 1 1 0 0 1 0 0 0 1 0]
Time= 6, Signals: [0 1 1 0 0 0 0 0] [1 0 1 1 1 1 0 1 1 0 0 0 1 0]
Time= 7, Signals: [1 1 1 0 0 0 0 0] [0 1 1 1 1 1 0 0 1 0 0 0 1 0]
Time= 8, Signals: [0 0 0 1 0 0 0 0] [1 0 1 0 1 0 1 0 1 0 0 0 1 0]
Time= 9, Signals: [1 0 0 1 0 0 0 0] [0 1 1 0 1 0 1 0 1 0 0 0 1 0]
Time= 10, Signals: [0 1 0 1 0 0 0 0] [1 0 1 1 1 0 1 1 1 0 0 0 1 0]
Time= 11, Signals: [1 1 0 1 0 0 0 0] [0 1 1 1 1 0 1 0 1 0 0 0 1 0]
Time= 12, Signals: [0 0 1 1 0 0 0 0] [1 0 1 0 1 1 1 0 1 0 1 0 1 0]
Time= 13, Signals: [1 0 1 1 0 0 0 0] [0 1 1 0 1 1 1 0 1 0 1 0 1 0]
Time= 14, Signals: [0 1 1 1 0 0 0 0] [1 0 1 1 1 1 1 1 1 0 1 0 1 0]
Time= 15, Signals: [1 1 1 1 0 0 0 0] [0 1 1 1 1 1 1 0 1 0 1 0 1 0]
Time= 16, Signals: [0 0 0 0 1 0 0 0] [1 0 1 0 1 0 0 0 1 0 0 0 1 0]
Time= 17, Signals: [1 0 0 0 1 0 0 0] [0 1 1 0 1 0 0 0 1 0 0 0 1 0]
Time= 18, Signals: [0 1 0 0 1 0 0 0] [1 0 1 1 1 0 0 1 0 0 0 0 1 0]
Time= 19, Signals: [1 1 0 0 1 0 0 0] [0 1 1 1 1 0 0 0 1 0 0 0 1 0]
Time= 20, Signals: [0 0 1 0 1 0 0 0] [1 0 1 0 1 1 0 0 1 0 0 0 1 0]
Time= 21, Signals: [1 0 1 0 1 0 0 0] [0 1 1 0 1 1 0 0 1 0 0 0 1 0]
Time= 22, Signals: [0 1 1 0 1 0 0 0] [1 0 1 1 1 1 0 1 0 0 0 0 1 0]
Time= 23, Signals: [1 1 1 0 1 0 0 0] [0 1 1 1 1 1 0 0 1 0 0 0 1 0]
Time= 24, Signals: [0 0 0 1 1 0 0 0] [1 0 1 0 1 0 1 0 1 0 0 0 1 0]
Time= 25, Signals: [1 0 0 1 1 0 0 0] [0 1 1 0 1 0 1 0 1 0 0 0 1 0]
Time= 26, Signals: [0 1 0 1 1 0 0 0] [1 0 1 1 1 0 1 1 0 0 0 0 1 0]
Time= 27, Signals: [1 1 0 1 1 0 0 0] [0 1 1 1 1 0 1 0 1 0 0 0 1 0]
Time= 28, Signals: [0 0 1 1 1 0 0 0] [1 0 1 0 1 1 1 0 1 0 1 0 1 0]
Time= 29, Signals: [1 0 1 1 1 0 0 0] [0 1 1 0 1 1 1 0 1 0 1 0 1 0]
Time= 30, Signals: [0 1 1 1 1 0 0 0] [1 0 1 1 1 1 1 1 0 0 1 0 1 0]
Time= 31, Signals: [1 1 1 1 1 0 0 0] [0 1 1 1 1 1 1 0 1 0 1 0 1 0]
Time= 32, Signals: [0 0 0 0 0 1 0 0] [1 0 1 0 1 0 0 0 1 0 0 0 1 0]
Time= 33, Signals: [1 0 0 0 0 1 0 0] [0 1 1 0 1 0 0 0 1 0 0 0 1 0]

Time= 34, Signals: [0 1 0 0 0 1 0 0] [1 0 1 1 1 0 0 1 1 0 0 0 1 0]
Time= 35, Signals: [1 1 0 0 0 1 0 0] [0 1 1 1 1 0 0 0 1 0 0 0 1 0]
Time= 36, Signals: [0 0 1 0 0 1 0 0] [1 0 1 0 1 1 0 0 1 0 0 0 1 0]
Time= 37, Signals: [1 0 1 0 0 1 0 0] [0 1 1 0 1 1 0 0 1 0 0 0 1 0]
Time= 38, Signals: [0 1 1 0 0 1 0 0] [1 0 1 1 1 1 0 1 1 0 0 0 1 0]
Time= 39, Signals: [1 1 1 0 0 1 0 0] [0 1 1 1 1 1 0 0 1 0 0 0 1 0]
Time= 40, Signals: [0 0 0 1 0 1 0 0] [1 0 1 0 1 0 1 0 1 0 0 0 1 0]
Time= 41, Signals: [1 0 0 1 0 1 0 0] [0 1 1 0 1 0 1 0 1 0 0 0 1 0]
Time= 42, Signals: [0 1 0 1 0 1 0 0] [1 0 1 1 1 0 1 1 1 0 0 0 1 0]
Time= 43, Signals: [1 1 0 1 0 1 0 0] [0 1 1 1 1 0 1 0 1 0 0 0 1 0]
Time= 44, Signals: [0 0 1 1 0 1 0 0] [1 0 1 0 x x 1 0 1 x x 0 1 0]
Time= 45, Signals: [1 0 1 1 0 1 0 0] [0 1 1 0 x x 1 0 1 x x 0 1 0]
Time= 46, Signals: [0 1 1 1 0 1 0 0] [1 0 1 x x x 1 x 1 x x 0 1 0]
Time= 47, Signals: [1 1 1 1 0 1 0 0] [0 1 1 x x x 1 0 1 x x 0 1 0]
Time= 48, Signals: [0 0 0 0 1 1 0 0] [1 0 1 0 1 0 0 0 1 0 0 0 1 0]
Time= 49, Signals: [1 0 0 0 1 1 0 0] [0 1 1 0 1 0 0 0 1 0 0 0 1 0]
Time= 50, Signals: [0 1 0 0 1 1 0 0] [1 0 1 1 1 0 0 1 0 0 0 0 1 0]
Time= 51, Signals: [1 1 0 0 1 1 0 0] [0 1 1 1 1 0 0 0 1 0 0 0 1 0]
Time= 52, Signals: [0 0 1 0 1 1 0 0] [1 0 1 0 1 1 0 0 1 0 0 0 1 0]
Time= 53, Signals: [1 0 1 0 1 1 0 0] [0 1 1 0 1 1 0 0 1 0 0 0 1 0]
Time= 54, Signals: [0 1 1 0 1 1 0 0] [1 0 1 1 1 1 0 1 0 0 0 0 1 0]
Time= 55, Signals: [1 1 1 0 1 1 0 0] [0 1 1 1 1 1 0 0 1 0 0 0 1 0]
Time= 56, Signals: [0 0 0 1 1 1 0 0] [1 0 1 0 1 0 1 0 1 0 0 0 1 0]
Time= 57, Signals: [1 0 0 1 1 1 0 0] [0 1 1 0 1 0 1 0 1 0 0 0 1 0]
Time= 58, Signals: [0 1 0 1 1 1 0 0] [1 0 1 1 1 0 1 1 0 0 0 0 1 0]
Time= 59, Signals: [1 1 0 1 1 1 0 0] [0 1 1 1 1 0 1 0 1 0 0 0 1 0]
Time= 60, Signals: [0 0 1 1 1 1 0 0] [1 0 1 0 x x 1 0 1 x x 0 1 0]
Time= 61, Signals: [1 0 1 1 1 1 0 0] [0 1 1 0 x x 1 0 1 x x 0 1 0]
Time= 62, Signals: [0 1 1 1 1 1 0 0] [1 0 1 x x x 1 x x x x 0 1 0]
Time= 63, Signals: [1 1 1 1 1 1 0 0] [0 1 1 x x x 1 0 1 x x 0 1 0]
Time= 64, Signals: [0 0 0 0 0 0 1 0] [1 0 1 0 1 0 0 0 1 0 0 0 1 0]
Time= 65, Signals: [1 0 0 0 0 0 1 0] [0 1 1 0 1 0 0 0 1 0 0 0 1 0]
Time= 66, Signals: [0 1 0 0 0 0 1 0] [1 0 1 1 1 0 0 1 1 0 0 0 1 0]
Time= 67, Signals: [1 1 0 0 0 0 1 0] [0 1 1 1 1 0 0 0 1 0 0 0 1 0]
Time= 68, Signals: [0 0 1 0 0 0 1 0] [1 0 1 0 1 1 0 0 1 0 0 0 1 0]
Time= 69, Signals: [1 0 1 0 0 0 1 0] [0 1 1 0 1 1 0 0 1 0 0 0 1 0]
Time= 70, Signals: [0 1 1 0 0 0 1 0] [1 0 1 1 1 1 0 1 1 0 0 0 1 0]
Time= 71, Signals: [1 1 1 0 0 0 1 0] [0 1 1 1 1 1 0 0 1 0 0 0 1 0]

Time= 72, Signals: [0 0 0 1 0 0 1 0] [1 0 1 0 1 0 1 0 1 0 0 0 1 0]
Time= 73, Signals: [1 0 0 1 0 0 1 0] [0 1 1 0 1 0 1 0 1 0 0 0 1 0]
Time= 74, Signals: [0 1 0 1 0 0 1 0] [1 0 1 1 1 0 1 1 1 0 0 0 1 0]
Time= 75, Signals: [1 1 0 1 0 0 1 0] [0 1 1 1 1 0 1 0 1 0 0 0 1 0]
Time= 76, Signals: [0 0 1 1 0 0 1 0] [1 0 x 0 1 1 x 0 1 0 x x 1 0]
Time= 77, Signals: [1 0 1 1 0 0 1 0] [x x x 0 1 1 x 0 1 0 x x 1 0]
Time= 78, Signals: [0 1 1 1 0 0 1 0] [1 0 x 1 1 1 x 1 1 0 x x 1 0]
Time= 79, Signals: [1 1 1 1 0 0 1 0] [x x x 1 1 1 x x 1 0 x x 1 0]
Time= 80, Signals: [0 0 0 0 1 0 1 0] [1 0 1 0 1 0 0 0 1 0 0 0 1 0]
Time= 81, Signals: [1 0 0 0 1 0 1 0] [0 1 1 0 1 0 0 0 1 0 0 0 1 0]
Time= 82, Signals: [0 1 0 0 1 0 1 0] [1 0 1 1 1 0 0 1 0 0 0 0 1 0]
Time= 83, Signals: [1 1 0 0 1 0 1 0] [0 1 1 1 1 0 0 0 1 0 0 0 1 0]
Time= 84, Signals: [0 0 1 0 1 0 1 0] [1 0 1 0 1 1 0 0 1 0 0 0 1 0]
Time= 85, Signals: [1 0 1 0 1 0 1 0] [0 1 1 0 1 1 0 0 1 0 0 0 1 0]
Time= 86, Signals: [0 1 1 0 1 0 1 0] [1 0 1 1 1 1 0 1 0 0 0 0 1 0]
Time= 87, Signals: [1 1 1 0 1 0 1 0] [0 1 1 1 1 1 0 0 1 0 0 0 1 0]
Time= 88, Signals: [0 0 0 1 1 0 1 0] [1 0 1 0 1 0 1 0 1 0 0 0 1 0]
Time= 89, Signals: [1 0 0 1 1 0 1 0] [0 1 1 0 1 0 1 0 1 0 0 0 1 0]
Time= 90, Signals: [0 1 0 1 1 0 1 0] [1 0 1 1 1 0 1 1 0 0 0 0 1 0]
Time= 91, Signals: [1 1 0 1 1 0 1 0] [0 1 1 1 1 0 1 0 1 0 0 0 1 0]
Time= 92, Signals: [0 0 1 1 1 0 1 0] [1 0 x 0 1 1 x 0 1 0 x x 1 0]
Time= 93, Signals: [1 0 1 1 1 0 1 0] [x x x 0 1 1 x 0 1 0 x x 1 0]
Time= 94, Signals: [0 1 1 1 1 0 1 0] [1 0 x 1 1 1 x 1 0 0 x x 1 0]
Time= 95, Signals: [1 1 1 1 1 0 1 0] [x x x 1 1 1 x x x 0 x x 1 0]
Time= 96, Signals: [0 0 0 0 0 1 1 0] [1 0 1 0 1 0 0 0 1 0 0 0 1 0]
Time= 97, Signals: [1 0 0 0 0 1 1 0] [0 1 1 0 1 0 0 0 1 0 0 0 1 0]
Time= 98, Signals: [0 1 0 0 0 1 1 0] [1 0 1 1 1 0 0 1 1 0 0 0 1 0]
Time= 99, Signals: [1 1 0 0 0 1 1 0] [0 1 1 1 1 0 0 0 1 0 0 0 1 0]
Time=100, Signals: [0 0 1 0 0 1 1 0] [1 0 1 0 1 1 0 0 1 0 0 0 1 0]
Time=101, Signals: [1 0 1 0 0 1 1 0] [0 1 1 0 1 1 0 0 1 0 0 0 1 0]
Time=102, Signals: [0 1 1 0 0 1 1 0] [1 0 1 1 1 1 0 1 1 0 0 0 1 0]
Time=103, Signals: [1 1 1 0 0 1 1 0] [0 1 1 1 1 1 0 0 1 0 0 0 1 0]
Time=104, Signals: [0 0 0 1 0 1 1 0] [1 0 1 0 1 0 1 0 1 0 0 0 1 0]
Time=105, Signals: [1 0 0 1 0 1 1 0] [0 1 1 0 1 0 1 0 1 0 0 0 1 0]
Time=106, Signals: [0 1 0 1 0 1 1 0] [1 0 1 1 1 0 1 1 1 0 0 0 1 0]
Time=107, Signals: [1 1 0 1 0 1 1 0] [0 1 1 1 1 0 1 0 1 0 0 0 1 0]
Time=108, Signals: [0 0 1 1 0 1 1 0] [1 0 x 0 x x x 0 1 x x x 1 0]
Time=109, Signals: [1 0 1 1 0 1 1 0] [x x x 0 x x x 0 1 x x x 1 0]

Time=110, Signals: [0 1 1 1 0 1 1 0] [1 0 x x x x x x 1 x x x 1 0]
Time=111, Signals: [1 1 1 1 0 1 1 0] [x x x x x x x x 1 x x x 1 0]
Time=112, Signals: [0 0 0 0 1 1 1 0] [1 0 1 0 1 0 0 0 1 0 0 0 1 0]
Time=113, Signals: [1 0 0 0 1 1 1 0] [0 1 1 0 1 0 0 0 1 0 0 0 1 0]
Time=114, Signals: [0 1 0 0 1 1 1 0] [1 0 1 1 1 0 0 1 0 0 0 0 1 0]
Time=115, Signals: [1 1 0 0 1 1 1 0] [0 1 1 1 1 0 0 0 1 0 0 0 1 0]
Time=116, Signals: [0 0 1 0 1 1 1 0] [1 0 1 0 1 1 0 0 1 0 0 0 1 0]
Time=117, Signals: [1 0 1 0 1 1 1 0] [0 1 1 0 1 1 0 0 1 0 0 0 1 0]
Time=118, Signals: [0 1 1 0 1 1 1 0] [1 0 1 1 1 1 0 1 0 0 0 0 1 0]
Time=119, Signals: [1 1 1 0 1 1 1 0] [0 1 1 1 1 1 0 0 1 0 0 0 1 0]
Time=120, Signals: [0 0 0 1 1 1 1 0] [1 0 1 0 1 0 1 0 1 0 0 0 1 0]
Time=121, Signals: [1 0 0 1 1 1 1 0] [0 1 1 0 1 0 1 0 1 0 0 0 1 0]
Time=122, Signals: [0 1 0 1 1 1 1 0] [1 0 1 1 1 0 1 1 0 0 0 0 1 0]
Time=123, Signals: [1 1 0 1 1 1 1 0] [0 1 1 1 1 0 1 0 1 0 0 0 1 0]
Time=124, Signals: [0 0 1 1 1 1 1 0] [1 0 x 0 x x x 0 1 x x x 1 0]
Time=125, Signals: [1 0 1 1 1 1 1 0] [x x x 0 x x x 0 1 x x x 1 0]
Time=126, Signals: [0 1 1 1 1 1 1 0] [1 0 x x x x x x x x x x 1 0]
Time=127, Signals: [1 1 1 1 1 1 1 0] [x x x x x x x x x x x x 1 0]
Time=128, Signals: [0 0 0 0 0 0 0 1] [1 0 1 0 1 0 0 0 1 0 0 0 1 0]
Time=129, Signals: [1 0 0 0 0 0 0 1] [0 1 1 0 1 0 0 0 1 0 0 0 1 0]
Time=130, Signals: [0 1 0 0 0 0 0 1] [1 0 1 1 1 0 0 1 1 0 0 0 0 1]
Time=131, Signals: [1 1 0 0 0 0 0 1] [0 1 1 1 1 0 0 0 1 0 0 0 1 0]
Time=132, Signals: [0 0 1 0 0 0 0 1] [1 0 1 0 1 1 0 0 1 0 0 0 1 0]
Time=133, Signals: [1 0 1 0 0 0 0 1] [0 1 1 0 1 1 0 0 1 0 0 0 1 0]
Time=134, Signals: [0 1 1 0 0 0 0 1] [1 0 1 1 1 1 0 1 1 0 0 0 0 1]
Time=135, Signals: [1 1 1 0 0 0 0 1] [0 1 1 1 1 1 0 0 1 0 0 0 1 0]
Time=136, Signals: [0 0 0 1 0 0 0 1] [1 0 1 0 1 0 1 0 1 0 0 0 1 0]
Time=137, Signals: [1 0 0 1 0 0 0 1] [0 1 1 0 1 0 1 0 1 0 0 0 1 0]
Time=138, Signals: [0 1 0 1 0 0 0 1] [1 0 1 1 1 0 1 1 1 0 0 0 0 1]
Time=139, Signals: [1 1 0 1 0 0 0 1] [0 1 1 1 1 0 1 0 1 0 0 0 1 0]
Time=140, Signals: [0 0 1 1 0 0 0 1] [1 0 1 0 1 1 1 0 1 0 1 0 1 0]
Time=141, Signals: [1 0 1 1 0 0 0 1] [0 1 1 0 1 1 1 0 1 0 1 0 1 0]
Time=142, Signals: [0 1 1 1 0 0 0 1] [1 0 1 1 1 1 1 1 1 0 1 0 0 1]
Time=143, Signals: [1 1 1 1 0 0 0 1] [0 1 1 1 1 1 1 0 1 0 1 0 1 0]
Time=144, Signals: [0 0 0 0 1 0 0 1] [1 0 1 0 1 0 0 0 1 0 0 0 1 0]
Time=145, Signals: [1 0 0 0 1 0 0 1] [0 1 1 0 1 0 0 0 1 0 0 0 1 0]
Time=146, Signals: [0 1 0 0 1 0 0 1] [1 0 1 1 1 0 0 1 0 0 0 0 0 1]
Time=147, Signals: [1 1 0 0 1 0 0 1] [0 1 1 1 1 0 0 0 1 0 0 0 1 0]

Time=148, Signals: [0 0 1 0 1 0 0 1] [1 0 1 0 1 1 0 0 1 0 0 0 1 0]
Time=149, Signals: [1 0 1 0 1 0 0 1] [0 1 1 0 1 1 0 0 1 0 0 0 1 0]
Time=150, Signals: [0 1 1 0 1 0 0 1] [1 0 1 1 1 1 0 1 0 0 0 0 0 1]
Time=151, Signals: [1 1 1 0 1 0 0 1] [0 1 1 1 1 1 0 0 1 0 0 0 1 0]
Time=152, Signals: [0 0 0 1 1 0 0 1] [1 0 1 0 1 0 1 0 1 0 0 0 1 0]
Time=153, Signals: [1 0 0 1 1 0 0 1] [0 1 1 0 1 0 1 0 1 0 0 0 1 0]
Time=154, Signals: [0 1 0 1 1 0 0 1] [1 0 1 1 1 0 1 1 0 0 0 0 0 1]
Time=155, Signals: [1 1 0 1 1 0 0 1] [0 1 1 1 1 0 1 0 1 0 0 0 1 0]
Time=156, Signals: [0 0 1 1 1 0 0 1] [1 0 1 0 1 1 1 0 1 0 1 0 1 0]
Time=157, Signals: [1 0 1 1 1 0 0 1] [0 1 1 0 1 1 1 0 1 0 1 0 1 0]
Time=158, Signals: [0 1 1 1 1 0 0 1] [1 0 1 1 1 1 1 1 0 0 1 0 0 1]
Time=159, Signals: [1 1 1 1 1 0 0 1] [0 1 1 1 1 1 1 0 1 0 1 0 1 0]
Time=160, Signals: [0 0 0 0 0 1 0 1] [1 0 1 0 1 0 0 0 1 0 0 0 1 0]
Time=161, Signals: [1 0 0 0 0 1 0 1] [0 1 1 0 1 0 0 0 1 0 0 0 1 0]
Time=162, Signals: [0 1 0 0 0 1 0 1] [1 0 1 1 1 0 0 1 1 0 0 0 0 1]
Time=163, Signals: [1 1 0 0 0 1 0 1] [0 1 1 1 1 0 0 0 1 0 0 0 1 0]
Time=164, Signals: [0 0 1 0 0 1 0 1] [1 0 1 0 1 1 0 0 1 0 0 0 1 0]
Time=165, Signals: [1 0 1 0 0 1 0 1] [0 1 1 0 1 1 0 0 1 0 0 0 1 0]
Time=166, Signals: [0 1 1 0 0 1 0 1] [1 0 1 1 1 1 0 1 1 0 0 0 0 1]
Time=167, Signals: [1 1 1 0 0 1 0 1] [0 1 1 1 1 1 0 0 1 0 0 0 1 0]
Time=168, Signals: [0 0 0 1 0 1 0 1] [1 0 1 0 1 0 1 0 1 0 0 0 1 0]
Time=169, Signals: [1 0 0 1 0 1 0 1] [0 1 1 0 1 0 1 0 1 0 0 0 1 0]
Time=170, Signals: [0 1 0 1 0 1 0 1] [1 0 1 1 1 0 1 1 1 0 0 0 0 1]
Time=171, Signals: [1 1 0 1 0 1 0 1] [0 1 1 1 1 0 1 0 1 0 0 0 1 0]
Time=172, Signals: [0 0 1 1 0 1 0 1] [1 0 1 0 x x 1 0 1 x x 0 1 0]
Time=173, Signals: [1 0 1 1 0 1 0 1] [0 1 1 0 x x 1 0 1 x x 0 1 0]
Time=174, Signals: [0 1 1 1 0 1 0 1] [1 0 1 x x x 1 x 1 x x 0 x x]
Time=175, Signals: [1 1 1 1 0 1 0 1] [0 1 1 x x x 1 0 1 x x 0 1 0]
Time=176, Signals: [0 0 0 0 1 1 0 1] [1 0 1 0 1 0 0 0 1 0 0 0 1 0]
Time=177, Signals: [1 0 0 0 1 1 0 1] [0 1 1 0 1 0 0 0 1 0 0 0 1 0]
Time=178, Signals: [0 1 0 0 1 1 0 1] [1 0 1 1 1 0 0 1 0 0 0 0 0 1]
Time=179, Signals: [1 1 0 0 1 1 0 1] [0 1 1 1 1 0 0 0 1 0 0 0 1 0]
Time=180, Signals: [0 0 1 0 1 1 0 1] [1 0 1 0 1 1 0 0 1 0 0 0 1 0]
Time=181, Signals: [1 0 1 0 1 1 0 1] [0 1 1 0 1 1 0 0 1 0 0 0 1 0]
Time=182, Signals: [0 1 1 0 1 1 0 1] [1 0 1 1 1 1 0 1 0 0 0 0 0 1]
Time=183, Signals: [1 1 1 0 1 1 0 1] [0 1 1 1 1 1 0 0 1 0 0 0 1 0]
Time=184, Signals: [0 0 0 1 1 1 0 1] [1 0 1 0 1 0 1 0 1 0 0 0 1 0]
Time=185, Signals: [1 0 0 1 1 1 0 1] [0 1 1 0 1 0 1 0 1 0 0 0 1 0]

Time=186, Signals: [0 1 0 1 1 1 0 1] [1 0 1 1 1 0 1 1 0 0 0 0 0 1]
Time=187, Signals: [1 1 0 1 1 1 0 1] [0 1 1 1 1 0 1 0 1 0 0 0 1 0]
Time=188, Signals: [0 0 1 1 1 1 0 1] [1 0 1 0 x x 1 0 1 x x 0 1 0]
Time=189, Signals: [1 0 1 1 1 1 0 1] [0 1 1 0 x x 1 0 1 x x 0 1 0]
Time=190, Signals: [0 1 1 1 1 1 0 1] [1 0 1 x x x 1 x x x x 0 x x]
Time=191, Signals: [1 1 1 1 1 1 0 1] [0 1 1 x x x 1 0 1 x x 0 1 0]
Time=192, Signals: [0 0 0 0 0 0 1 1] [1 0 1 0 1 0 0 0 1 0 0 0 1 0]
Time=193, Signals: [1 0 0 0 0 0 1 1] [0 1 1 0 1 0 0 0 1 0 0 0 1 0]
Time=194, Signals: [0 1 0 0 0 0 1 1] [1 0 1 1 1 0 0 1 1 0 0 0 0 1]
Time=195, Signals: [1 1 0 0 0 0 1 1] [0 1 1 1 1 0 0 0 1 0 0 0 1 0]
Time=196, Signals: [0 0 1 0 0 0 1 1] [1 0 1 0 1 1 0 0 1 0 0 0 1 0]
Time=197, Signals: [1 0 1 0 0 0 1 1] [0 1 1 0 1 1 0 0 1 0 0 0 1 0]
Time=198, Signals: [0 1 1 0 0 0 1 1] [1 0 1 1 1 1 0 1 1 0 0 0 0 1]
Time=199, Signals: [1 1 1 0 0 0 1 1] [0 1 1 1 1 1 0 0 1 0 0 0 1 0]
Time=200, Signals: [0 0 0 1 0 0 1 1] [1 0 1 0 1 0 1 0 1 0 0 0 1 0]
Time=201, Signals: [1 0 0 1 0 0 1 1] [0 1 1 0 1 0 1 0 1 0 0 0 1 0]
Time=202, Signals: [0 1 0 1 0 0 1 1] [1 0 1 1 1 0 1 1 1 0 0 0 0 1]
Time=203, Signals: [1 1 0 1 0 0 1 1] [0 1 1 1 1 0 1 0 1 0 0 0 1 0]
Time=204, Signals: [0 0 1 1 0 0 1 1] [1 0 x 0 1 1 x 0 1 0 x x 1 0]
Time=205, Signals: [1 0 1 1 0 0 1 1] [x x x 0 1 1 x 0 1 0 x x 1 0]
Time=206, Signals: [0 1 1 1 0 0 1 1] [1 0 1 1 1 1 1 1 1 0 1 1 0 1]
Time=207, Signals: [1 1 1 1 0 0 1 1] [x x x 1 1 1 x x 1 0 x x x x]
Time=208, Signals: [0 0 0 0 1 0 1 1] [1 0 1 0 1 0 0 0 1 0 0 0 1 0]
Time=209, Signals: [1 0 0 0 1 0 1 1] [0 1 1 0 1 0 0 0 1 0 0 0 1 0]
Time=210 Signals: [0 1 0 0 1 0 1 1] [1 0 1 1 1 0 0 1 0 0 0 0 0 1]
Time=211, Signals: [1 1 0 0 1 0 1 1] [0 1 1 1 1 0 0 0 1 0 0 0 1 0]
Time=212, Signals: [0 0 1 0 1 0 1 1] [1 0 1 0 1 1 0 0 1 0 0 0 1 0]
Time=213, Signals: [1 0 1 0 1 0 1 1] [0 1 1 0 1 1 0 0 1 0 0 0 1 0]
Time=214, Signals: [0 1 1 0 1 0 1 1] [1 0 1 1 1 1 0 1 0 0 0 0 0 1]
Time=215, Signals: [1 1 1 0 1 0 1 1] [0 1 1 1 1 1 0 0 1 0 0 0 1 0]
Time=216, Signals: [0 0 0 1 1 0 1 1] [1 0 1 0 1 0 1 0 1 0 0 0 1 0]
Time=217, Signals: [1 0 0 1 1 0 1 1] [0 1 1 0 1 0 1 0 1 0 0 0 1 0]
Time=218, Signals: [0 1 0 1 1 0 1 1] [1 0 1 1 1 0 1 1 0 0 0 0 0 1]
Time=219, Signals: [1 1 0 1 1 0 1 1] [0 1 1 1 1 0 1 0 1 0 0 0 1 0]
Time=220, Signals: [0 0 1 1 1 0 1 1] [1 0 x 0 1 1 x 0 1 0 x x 1 0]
Time=221, Signals: [1 0 1 1 1 0 1 1] [x x x 0 1 1 x 0 1 0 x x 1 0]
Time=222, Signals: [0 1 1 1 1 0 1 1] [1 0 1 1 1 1 1 1 1 0 0 1 1 0 1]
Time=223, Signals: [1 1 1 1 1 0 1 1] [x x x 1 1 1 x x x 0 x x x x]

Time=224, Signals: [0 0 0 0 0 1 1 1] [1 0 1 0 1 0 0 0 1 0 0 0 1 0]
Time=225, Signals: [1 0 0 0 0 1 1 1] [0 1 1 0 1 0 0 0 1 0 0 0 1 0]
Time=226, Signals: [0 1 0 0 0 1 1 1] [1 0 1 1 1 0 0 1 1 0 0 0 0 1]
Time=227, Signals: [1 1 0 0 0 1 1 1] [0 1 1 1 1 0 0 0 1 0 0 0 1 0]
Time=228, Signals: [0 0 1 0 0 1 1 1] [1 0 1 0 1 1 0 0 1 0 0 0 1 0]
Time=229, Signals: [1 0 1 0 0 1 1 1] [0 1 1 0 1 1 0 0 1 0 0 0 1 0]
Time=230, Signals: [0 1 1 0 0 1 1 1] [1 0 1 1 1 1 0 1 1 0 0 0 0 1]
Time=231, Signals: [1 1 1 0 0 1 1 1] [0 1 1 1 1 1 0 0 1 0 0 0 1 0]
Time=232, Signals: [0 0 0 1 0 1 1 1] [1 0 1 0 1 0 1 0 1 0 0 0 1 0]
Time=233, Signals: [1 0 0 1 0 1 1 1] [0 1 1 0 1 0 1 0 1 0 0 0 1 0]
Time=234, Signals: [0 1 0 1 0 1 1 1] [1 0 1 1 1 0 1 1 1 0 0 0 0 1]
Time=235, Signals: [1 1 0 1 0 1 1 1] [0 1 1 1 1 0 1 0 1 0 0 0 1 0]
Time=236, Signals: [0 0 1 1 0 1 1 1] [1 0 x 0 x x x 0 1 x x x 1 0]
Time=237, Signals: [1 0 1 1 0 1 1 1] [x x x 0 x x x 0 1 x x x 1 0]
Time=238, Signals: [0 1 1 1 0 1 1 1] [1 0 x x x x x x 1 x x x x x]
Time=239, Signals: [1 1 1 1 0 1 1 1] [x x x x x x x x 1 x x x x x]
Time=240, Signals: [0 0 0 0 1 1 1 1] [1 0 1 0 1 0 0 0 1 0 0 0 1 0]
Time=241, Signals: [1 0 0 0 1 1 1 1] [0 1 1 0 1 0 0 0 1 0 0 0 1 0]
Time=242, Signals: [0 1 0 0 1 1 1 1] [1 0 1 1 1 0 0 1 0 0 0 0 0 1]
Time=243, Signals: [1 1 0 0 1 1 1 1] [0 1 1 1 1 0 0 0 1 0 0 0 1 0]
Time=244, Signals: [0 0 1 0 1 1 1 1] [1 0 1 0 1 1 0 0 1 0 0 0 1 0]
Time=245, Signals: [1 0 1 0 1 1 1 1] [0 1 1 0 1 1 0 0 1 0 0 0 1 0]
Time=246, Signals: [0 1 1 0 1 1 1 1] [1 0 1 1 1 1 0 1 0 0 0 0 0 1]
Time=247, Signals: [1 1 1 0 1 1 1 1] [0 1 1 1 1 1 0 0 1 0 0 0 1 0]
Time=248, Signals: [0 0 0 1 1 1 1 1] [1 0 1 0 1 0 1 0 1 0 0 0 1 0]
Time=249, Signals: [1 0 0 1 1 1 1 1] [0 1 1 0 1 0 1 0 1 0 0 0 1 0]
Time=250, Signals: [0 1 0 1 1 1 1 1] [1 0 1 1 1 0 1 1 0 0 0 0 0 1]
Time=251, Signals: [1 1 0 1 1 1 1 1] [0 1 1 1 1 0 1 0 1 0 0 0 1 0]
Time=252, Signals: [0 0 1 1 1 1 1 1] [1 0 x 0 x x x 0 1 x x x 1 0]
Time=253, Signals: [1 0 1 1 1 1 1 1] [x x x 0 x x x 0 1 x x x 1 0]
Time=254, Signals: [0 1 1 1 1 1 1 1] [1 0 x x x x x x x x x x x x]
Time=255, Signals: [1 1 1 1 1 1 1 1] [x x x x x x x x x x x x x x]

B Appendix B: Our Result (simulation of result5.v)

Time= 0, Signals: [0 0 0 0 0 0 0 0] [1 0 1 0 1 0 0 0 1 0 0 0 1 0] [0]
Time= 1, Signals: [1 0 0 0 0 0 0 0] [0 1 1 0 1 0 0 0 1 0 0 0 1 0] [0]
Time= 2, Signals: [0 1 0 0 0 0 0 0] [1 0 1 1 1 0 0 1 1 0 0 0 1 0] [0]
Time= 3, Signals: [1 1 0 0 0 0 0 0] [0 1 1 1 1 0 0 0 1 0 0 0 1 0] [0]
Time= 4, Signals: [0 0 1 0 0 0 0 0] [1 0 1 0 1 1 0 0 1 0 0 0 1 0] [0]
Time= 5, Signals: [1 0 1 0 0 0 0 0] [0 1 1 0 1 1 0 0 1 0 0 0 1 0] [0]
Time= 6, Signals: [0 1 1 0 0 0 0 0] [1 0 1 1 1 1 0 1 1 0 0 0 1 0] [0]
Time= 7, Signals: [1 1 1 0 0 0 0 0] [0 1 1 1 1 1 0 0 1 0 0 0 1 0] [0]
Time= 8, Signals: [0 0 0 1 0 0 0 0] [1 0 1 0 1 0 1 0 1 0 0 0 1 0] [0]
Time= 9, Signals: [1 0 0 1 0 0 0 0] [0 1 1 0 1 0 1 0 1 0 0 0 1 0] [0]
Time= 10, Signals: [0 1 0 1 0 0 0 0] [1 0 1 1 1 0 1 1 1 0 0 0 1 0] [0]
Time= 11, Signals: [1 1 0 1 0 0 0 0] [0 1 1 1 1 0 1 0 1 0 0 0 1 0] [0]
Time= 12, Signals: [0 0 1 1 0 0 0 0] [1 0 1 0 1 1 1 0 1 0 1 0 1 0] [0]
Time= 13, Signals: [1 0 1 1 0 0 0 0] [0 1 1 0 1 1 1 0 1 0 1 0 1 0] [0]
Time= 14, Signals: [0 1 1 1 0 0 0 0] [1 0 1 1 1 1 1 1 1 0 1 0 1 0] [0]
Time= 15, Signals: [1 1 1 1 0 0 0 0] [0 1 1 1 1 1 1 0 1 0 1 0 1 0] [0]
Time= 16, Signals: [0 0 0 0 1 0 0 0] [1 0 1 0 1 0 0 0 1 0 0 0 1 0] [0]
Time= 17, Signals: [1 0 0 0 1 0 0 0] [0 1 1 0 1 0 0 0 1 0 0 0 1 0] [0]
Time= 18, Signals: [0 1 0 0 1 0 0 0] [1 0 1 1 1 0 0 1 0 0 0 0 1 0] [0]
Time= 19, Signals: [1 1 0 0 1 0 0 0] [0 1 1 1 1 0 0 0 1 0 0 0 1 0] [0]
Time= 20, Signals: [0 0 1 0 1 0 0 0] [1 0 1 0 1 1 0 0 1 0 0 0 1 0] [0]
Time= 21, Signals: [1 0 1 0 1 0 0 0] [0 1 1 0 1 1 0 0 1 0 0 0 1 0] [0]
Time= 22, Signals: [0 1 1 0 1 0 0 0] [1 0 1 1 1 1 0 1 0 0 0 0 1 0] [0]
Time= 23, Signals: [1 1 1 0 1 0 0 0] [0 1 1 1 1 1 0 0 1 0 0 0 1 0] [0]
Time= 24, Signals: [0 0 0 1 1 0 0 0] [1 0 1 0 1 0 1 0 1 0 0 0 1 0] [0]
Time= 25, Signals: [1 0 0 1 1 0 0 0] [0 1 1 0 1 0 1 0 1 0 0 0 1 0] [0]
Time= 26, Signals: [0 1 0 1 1 0 0 0] [1 0 1 1 1 0 1 1 0 0 0 0 1 0] [0]
Time= 27, Signals: [1 1 0 1 1 0 0 0] [0 1 1 1 1 0 1 0 1 0 0 0 1 0] [0]
Time= 28, Signals: [0 0 1 1 1 0 0 0] [1 0 1 0 1 1 1 0 1 0 1 0 1 0] [0]
Time= 29, Signals: [1 0 1 1 1 0 0 0] [0 1 1 0 1 1 1 0 1 0 1 0 1 0] [0]
Time= 30, Signals: [0 1 1 1 1 0 0 0] [1 0 1 1 1 1 1 1 0 0 1 0 1 0] [0]
Time= 31, Signals: [1 1 1 1 1 0 0 0] [0 1 1 1 1 1 1 0 1 0 1 0 1 0] [0]
Time= 32, Signals: [0 0 0 0 0 1 0 0] [1 0 1 0 1 0 0 0 1 0 0 0 1 0] [0]
Time= 33, Signals: [1 0 0 0 0 1 0 0] [0 1 1 0 1 0 0 0 1 0 0 0 1 0] [0]

Time= 34, Signals: [0 1 0 0 0 1 0 0] [1 0 1 1 1 0 0 1 1 0 0 0 1 0] [0]
Time= 35, Signals: [1 1 0 0 0 1 0 0] [0 1 1 1 1 0 0 0 1 0 0 0 1 0] [0]
Time= 36, Signals: [0 0 1 0 0 1 0 0] [1 0 1 0 1 1 0 0 1 0 0 0 1 0] [0]
Time= 37, Signals: [1 0 1 0 0 1 0 0] [0 1 1 0 1 1 0 0 1 0 0 0 1 0] [0]
Time= 38, Signals: [0 1 1 0 0 1 0 0] [1 0 1 1 1 1 0 1 1 0 0 0 1 0] [0]
Time= 39, Signals: [1 1 1 0 0 1 0 0] [0 1 1 1 1 1 0 0 1 0 0 0 1 0] [0]
Time= 40, Signals: [0 0 0 1 0 1 0 0] [1 0 1 0 1 0 1 0 1 0 0 0 1 0] [0]
Time= 41, Signals: [1 0 0 1 0 1 0 0] [0 1 1 0 1 0 1 0 1 0 0 0 1 0] [0]
Time= 42, Signals: [0 1 0 1 0 1 0 0] [1 0 1 1 1 0 1 1 1 0 0 0 1 0] [0]
Time= 43, Signals: [1 1 0 1 0 1 0 0] [0 1 1 1 1 0 1 0 1 0 0 0 1 0] [0]
Time= 44, Signals: [0 0 1 1 0 1 0 0] [1 0 1 0 0 0 1 0 1 1 0 0 1 0] [1]
Time= 45, Signals: [1 0 1 1 0 1 0 0] [0 1 1 0 0 0 1 0 1 1 0 0 1 0] [1]
Time= 46, Signals: [0 1 1 1 0 1 0 0] [1 0 1 0 0 0 1 0 1 1 0 0 1 0] [1]
Time= 47, Signals: [1 1 1 1 0 1 0 0] [0 1 1 0 0 0 1 0 1 1 0 0 1 0] [1]
Time= 48, Signals: [0 0 0 0 1 1 0 0] [1 0 1 0 1 0 0 0 1 0 0 0 1 0] [0]
Time= 49, Signals: [1 0 0 0 1 1 0 0] [0 1 1 0 1 0 0 0 1 0 0 0 1 0] [0]
Time= 50, Signals: [0 1 0 0 1 1 0 0] [1 0 1 1 1 0 0 1 0 0 0 0 1 0] [0]
Time= 51, Signals: [1 1 0 0 1 1 0 0] [0 1 1 1 1 0 0 0 1 0 0 0 1 0] [0]
Time= 52, Signals: [0 0 1 0 1 1 0 0] [1 0 1 0 1 1 0 0 1 0 0 0 1 0] [0]
Time= 53, Signals: [1 0 1 0 1 1 0 0] [0 1 1 0 1 1 0 0 1 0 0 0 1 0] [0]
Time= 54, Signals: [0 1 1 0 1 1 0 0] [1 0 1 1 1 1 0 1 0 0 0 0 1 0] [0]
Time= 55, Signals: [1 1 1 0 1 1 0 0] [0 1 1 1 1 1 0 0 1 0 0 0 1 0] [0]
Time= 56, Signals: [0 0 0 1 1 1 0 0] [1 0 1 0 1 0 1 0 1 0 0 0 1 0] [0]
Time= 57, Signals: [1 0 0 1 1 1 0 0] [0 1 1 0 1 0 1 0 1 0 0 0 1 0] [0]
Time= 58, Signals: [0 1 0 1 1 1 0 0] [1 0 1 1 1 0 1 1 0 0 0 0 1 0] [0]
Time= 59, Signals: [1 1 0 1 1 1 0 0] [0 1 1 1 1 0 1 0 1 0 0 0 1 0] [0]
Time= 60, Signals: [0 0 1 1 1 1 0 0] [1 0 1 0 0 0 1 0 1 1 0 0 1 0] [1]
Time= 61, Signals: [1 0 1 1 1 1 0 0] [0 1 1 0 0 0 1 0 1 1 0 0 1 0] [1]
Time= 62, Signals: [0 1 1 1 1 1 0 0] [1 0 1 0 0 0 1 0 1 1 0 0 1 0] [1]
Time= 63, Signals: [1 1 1 1 1 1 0 0] [0 1 1 0 0 0 1 0 1 1 0 0 1 0] [1]
Time= 64, Signals: [0 0 0 0 0 0 1 0] [1 0 1 0 1 0 0 0 1 0 0 0 1 0] [0]
Time= 65, Signals: [1 0 0 0 0 0 1 0] [0 1 1 0 1 0 0 0 1 0 0 0 1 0] [0]
Time= 66, Signals: [0 1 0 0 0 0 1 0] [1 0 1 1 1 0 0 1 1 0 0 0 1 0] [0]
Time= 67, Signals: [1 1 0 0 0 0 1 0] [0 1 1 1 1 0 0 0 1 0 0 0 1 0] [0]
Time= 68, Signals: [0 0 1 0 0 0 1 0] [1 0 1 0 1 1 0 0 1 0 0 0 1 0] [0]
Time= 69, Signals: [1 0 1 0 0 0 1 0] [0 1 1 0 1 1 0 0 1 0 0 0 1 0] [0]
Time= 70, Signals: [0 1 1 0 0 0 1 0] [1 0 1 1 1 1 0 1 1 0 0 0 1 0] [0]
Time= 71, Signals: [1 1 1 0 0 0 1 0] [0 1 1 1 1 1 0 0 1 0 0 0 1 0] [0]

Time= 72, Signals: [0 0 0 1 0 0 1 0] [1 0 1 0 1 0 1 0 1 0 0 0 1 0] [0]
Time= 73, Signals: [1 0 0 1 0 0 1 0] [0 1 1 0 1 0 1 0 1 0 0 0 1 0] [0]
Time= 74, Signals: [0 1 0 1 0 0 1 0] [1 0 1 1 1 0 1 1 1 0 0 0 1 0] [0]
Time= 75, Signals: [1 1 0 1 0 0 1 0] [0 1 1 1 1 0 1 0 1 0 0 0 1 0] [0]
Time= 76, Signals: [0 0 1 1 0 0 1 0] [1 0 0 0 1 1 0 0 1 0 0 1 1 0] [1]
Time= 77, Signals: [1 0 1 1 0 0 1 0] [1 0 0 0 1 1 0 0 1 0 0 1 1 0] [1]
Time= 78, Signals: [0 1 1 1 0 0 1 0] [1 0 0 1 1 1 0 1 1 0 0 1 1 0] [1]
Time= 79, Signals: [1 1 1 1 0 0 1 0] [1 0 0 1 1 1 0 1 1 0 0 1 1 0] [1]
Time= 80, Signals: [0 0 0 0 1 0 1 0] [1 0 1 0 1 0 0 0 1 0 0 0 1 0] [0]
Time= 81, Signals: [1 0 0 0 1 0 1 0] [0 1 1 0 1 0 0 0 1 0 0 0 1 0] [0]
Time= 82, Signals: [0 1 0 0 1 0 1 0] [1 0 1 1 1 0 0 1 0 0 0 0 1 0] [0]
Time= 83, Signals: [1 1 0 0 1 0 1 0] [0 1 1 1 1 0 0 0 1 0 0 0 1 0] [0]
Time= 84, Signals: [0 0 1 0 1 0 1 0] [1 0 1 0 1 1 0 0 1 0 0 0 1 0] [0]
Time= 85, Signals: [1 0 1 0 1 0 1 0] [0 1 1 0 1 1 0 0 1 0 0 0 1 0] [0]
Time= 86, Signals: [0 1 1 0 1 0 1 0] [1 0 1 1 1 1 0 1 0 0 0 0 1 0] [0]
Time= 87, Signals: [1 1 1 0 1 0 1 0] [0 1 1 1 1 1 0 0 1 0 0 0 1 0] [0]
Time= 88, Signals: [0 0 0 1 1 0 1 0] [1 0 1 0 1 0 1 0 1 0 0 0 1 0] [0]
Time= 89, Signals: [1 0 0 1 1 0 1 0] [0 1 1 0 1 0 1 0 1 0 0 0 1 0] [0]
Time= 90, Signals: [0 1 0 1 1 0 1 0] [1 0 1 1 1 0 1 1 0 0 0 0 1 0] [0]
Time= 91, Signals: [1 1 0 1 1 0 1 0] [0 1 1 1 1 0 1 0 1 0 0 0 1 0] [0]
Time= 92, Signals: [0 0 1 1 1 0 1 0] [1 0 0 0 1 1 0 0 1 0 0 1 1 0] [1]
Time= 93, Signals: [1 0 1 1 1 0 1 0] [1 0 0 0 1 1 0 0 1 0 0 1 1 0] [1]
Time= 94, Signals: [0 1 1 1 1 0 1 0] [1 0 0 1 1 1 0 1 0 0 0 1 1 0] [1]
Time= 95, Signals: [1 1 1 1 1 0 1 0] [1 0 0 1 1 1 0 1 0 0 0 1 1 0] [1]
Time= 96, Signals: [0 0 0 0 0 1 1 0] [1 0 1 0 1 0 0 0 1 0 0 0 1 0] [0]
Time= 97, Signals: [1 0 0 0 0 1 1 0] [0 1 1 0 1 0 0 0 1 0 0 0 1 0] [0]
Time= 98, Signals: [0 1 0 0 0 1 1 0] [1 0 1 1 1 0 0 1 1 0 0 0 1 0] [0]
Time= 99, Signals: [1 1 0 0 0 1 1 0] [0 1 1 1 1 0 0 0 1 0 0 0 1 0] [0]
Time=100, Signals: [0 0 1 0 0 1 1 0] [1 0 1 0 1 1 0 0 1 0 0 0 1 0] [0]
Time=101, Signals: [1 0 1 0 0 1 1 0] [0 1 1 0 1 1 0 0 1 0 0 0 1 0] [0]
Time=102, Signals: [0 1 1 0 0 1 1 0] [1 0 1 1 1 1 0 1 1 0 0 0 1 0] [0]
Time=103, Signals: [1 1 1 0 0 1 1 0] [0 1 1 1 1 1 0 0 1 0 0 0 1 0] [0]
Time=104, Signals: [0 0 0 1 0 1 1 0] [1 0 1 0 1 0 1 0 1 0 0 0 1 0] [0]
Time=105, Signals: [1 0 0 1 0 1 1 0] [0 1 1 0 1 0 1 0 1 0 0 0 1 0] [0]
Time=106, Signals: [0 1 0 1 0 1 1 0] [1 0 1 1 1 0 1 1 1 0 0 0 1 0] [0]
Time=107, Signals: [1 1 0 1 0 1 1 0] [0 1 1 1 1 0 1 0 1 0 0 0 1 0] [0]
Time=108, Signals: [0 0 1 1 0 1 1 0] [1 0 0 0 0 0 0 0 0 1 1 0 1 1 0] [1]
Time=109, Signals: [1 0 1 1 0 1 1 0] [1 0 0 0 0 0 0 0 0 1 1 0 1 1 0] [1]

Time=110, Signals: [0 1 1 1 0 1 1 0] [1 0 0 0 0 0 0 0 1 1 0 1 1 0] [1]
Time=111, Signals: [1 1 1 1 0 1 1 0] [1 0 0 0 0 0 0 0 1 1 0 1 1 0] [1]
Time=112, Signals: [0 0 0 0 1 1 1 0] [1 0 1 0 1 0 0 0 1 0 0 0 1 0] [0]
Time=113, Signals: [1 0 0 0 1 1 1 0] [0 1 1 0 1 0 0 0 1 0 0 0 1 0] [0]
Time=114, Signals: [0 1 0 0 1 1 1 0] [1 0 1 1 1 0 0 1 0 0 0 0 1 0] [0]
Time=115, Signals: [1 1 0 0 1 1 1 0] [0 1 1 1 1 0 0 0 1 0 0 0 1 0] [0]
Time=116, Signals: [0 0 1 0 1 1 1 0] [1 0 1 0 1 1 0 0 1 0 0 0 1 0] [0]
Time=117, Signals: [1 0 1 0 1 1 1 0] [0 1 1 0 1 1 0 0 1 0 0 0 1 0] [0]
Time=118, Signals: [0 1 1 0 1 1 1 0] [1 0 1 1 1 1 0 1 0 0 0 0 1 0] [0]
Time=119, Signals: [1 1 1 0 1 1 1 0] [0 1 1 1 1 1 0 0 1 0 0 0 1 0] [0]
Time=120, Signals: [0 0 0 1 1 1 1 0] [1 0 1 0 1 0 1 0 1 0 0 0 1 0] [0]
Time=121, Signals: [1 0 0 1 1 1 1 0] [0 1 1 0 1 0 1 0 1 0 0 0 1 0] [0]
Time=122, Signals: [0 1 0 1 1 1 1 0] [1 0 1 1 1 0 1 1 0 0 0 0 1 0] [0]
Time=123, Signals: [1 1 0 1 1 1 1 0] [0 1 1 1 1 0 1 0 1 0 0 0 1 0] [0]
Time=124, Signals: [0 0 1 1 1 1 1 0] [1 0 0 0 0 0 0 0 1 1 0 1 1 0] [1]
Time=125, Signals: [1 0 1 1 1 1 1 0] [1 0 0 0 0 0 0 0 1 1 0 1 1 0] [1]
Time=126, Signals: [0 1 1 1 1 1 1 0] [1 0 0 0 0 0 0 0 1 1 0 1 1 0] [1]
Time=127, Signals: [1 1 1 1 1 1 1 0] [1 0 0 0 0 0 0 0 1 1 0 1 1 0] [1]
Time=128, Signals: [0 0 0 0 0 0 0 1] [1 0 1 0 1 0 0 0 1 0 0 0 1 0] [0]
Time=129, Signals: [1 0 0 0 0 0 0 1] [0 1 1 0 1 0 0 0 1 0 0 0 1 0] [0]
Time=130, Signals: [0 1 0 0 0 0 0 1] [1 0 1 1 1 0 0 1 1 0 0 0 0 1] [0]
Time=131, Signals: [1 1 0 0 0 0 0 1] [0 1 1 1 1 0 0 0 1 0 0 0 1 0] [0]
Time=132, Signals: [0 0 1 0 0 0 0 1] [1 0 1 0 1 1 0 0 1 0 0 0 1 0] [0]
Time=133, Signals: [1 0 1 0 0 0 0 1] [0 1 1 0 1 1 0 0 1 0 0 0 1 0] [0]
Time=134, Signals: [0 1 1 0 0 0 0 1] [1 0 1 1 1 1 0 1 1 0 0 0 0 1] [0]
Time=135, Signals: [1 1 1 0 0 0 0 1] [0 1 1 1 1 1 0 0 1 0 0 0 1 0] [0]
Time=136, Signals: [0 0 0 1 0 0 0 1] [1 0 1 0 1 0 1 0 1 0 0 0 1 0] [0]
Time=137, Signals: [1 0 0 1 0 0 0 1] [0 1 1 0 1 0 1 0 1 0 0 0 1 0] [0]
Time=138, Signals: [0 1 0 1 0 0 0 1] [1 0 1 1 1 0 1 1 1 0 0 0 0 1] [0]
Time=139, Signals: [1 1 0 1 0 0 0 1] [0 1 1 1 1 0 1 0 1 0 0 0 1 0] [0]
Time=140, Signals: [0 0 1 1 0 0 0 1] [1 0 1 0 1 1 1 0 1 0 1 0 1 0] [0]
Time=141, Signals: [1 0 1 1 0 0 0 1] [0 1 1 0 1 1 1 0 1 0 1 0 1 0] [0]
Time=142, Signals: [0 1 1 1 0 0 0 1] [1 0 1 1 1 1 1 1 1 0 1 0 0 1] [0]
Time=143, Signals: [1 1 1 1 0 0 0 1] [0 1 1 1 1 1 1 0 1 0 1 0 1 0] [0]
Time=144, Signals: [0 0 0 0 1 0 0 1] [1 0 1 0 1 0 0 0 1 0 0 0 1 0] [0]
Time=145, Signals: [1 0 0 0 1 0 0 1] [0 1 1 0 1 0 0 0 1 0 0 0 1 0] [0]
Time=146, Signals: [0 1 0 0 1 0 0 1] [1 0 1 1 1 0 0 1 0 0 0 0 0 1] [0]
Time=147, Signals: [1 1 0 0 1 0 0 1] [0 1 1 1 1 0 0 0 1 0 0 0 1 0] [0]

Time=148, Signals: [0 0 1 0 1 0 0 1] [1 0 1 0 1 1 0 0 1 0 0 0 1 0] [0]
Time=149, Signals: [1 0 1 0 1 0 0 1] [0 1 1 0 1 1 0 0 1 0 0 0 1 0] [0]
Time=150, Signals: [0 1 1 0 1 0 0 1] [1 0 1 1 1 1 0 1 0 0 0 0 0 1] [0]
Time=151, Signals: [1 1 1 0 1 0 0 1] [0 1 1 1 1 1 0 0 1 0 0 0 1 0] [0]
Time=152, Signals: [0 0 0 1 1 0 0 1] [1 0 1 0 1 0 1 0 1 0 0 0 1 0] [0]
Time=153, Signals: [1 0 0 1 1 0 0 1] [0 1 1 0 1 0 1 0 1 0 0 0 1 0] [0]
Time=154, Signals: [0 1 0 1 1 0 0 1] [1 0 1 1 1 0 1 1 0 0 0 0 0 1] [0]
Time=155, Signals: [1 1 0 1 1 0 0 1] [0 1 1 1 1 0 1 0 1 0 0 0 1 0] [0]
Time=156, Signals: [0 0 1 1 1 0 0 1] [1 0 1 0 1 1 1 0 1 0 1 0 1 0] [0]
Time=157, Signals: [1 0 1 1 1 0 0 1] [0 1 1 0 1 1 1 0 1 0 1 0 1 0] [0]
Time=158, Signals: [0 1 1 1 1 0 0 1] [1 0 1 1 1 1 1 1 0 0 1 0 0 1] [0]
Time=159, Signals: [1 1 1 1 1 0 0 1] [0 1 1 1 1 1 1 0 1 0 1 0 1 0] [0]
Time=160, Signals: [0 0 0 0 0 1 0 1] [1 0 1 0 1 0 0 0 1 0 0 0 1 0] [0]
Time=161, Signals: [1 0 0 0 0 1 0 1] [0 1 1 0 1 0 0 0 1 0 0 0 1 0] [0]
Time=162, Signals: [0 1 0 0 0 1 0 1] [1 0 1 1 1 0 0 1 1 0 0 0 0 1] [0]
Time=163, Signals: [1 1 0 0 0 1 0 1] [0 1 1 1 1 0 0 0 1 0 0 0 1 0] [0]
Time=164, Signals: [0 0 1 0 0 1 0 1] [1 0 1 0 1 1 0 0 1 0 0 0 1 0] [0]
Time=165, Signals: [1 0 1 0 0 1 0 1] [0 1 1 0 1 1 0 0 1 0 0 0 1 0] [0]
Time=166, Signals: [0 1 1 0 0 1 0 1] [1 0 1 1 1 1 0 1 1 0 0 0 0 1] [0]
Time=167, Signals: [1 1 1 0 0 1 0 1] [0 1 1 1 1 1 0 0 1 0 0 0 1 0] [0]
Time=168, Signals: [0 0 0 1 0 1 0 1] [1 0 1 0 1 0 1 0 1 0 0 0 1 0] [0]
Time=169, Signals: [1 0 0 1 0 1 0 1] [0 1 1 0 1 0 1 0 1 0 0 0 1 0] [0]
Time=170, Signals: [0 1 0 1 0 1 0 1] [1 0 1 1 1 0 1 1 1 0 0 0 0 1] [0]
Time=171, Signals: [1 1 0 1 0 1 0 1] [0 1 1 1 1 0 1 0 1 0 0 0 1 0] [0]
Time=172, Signals: [0 0 1 1 0 1 0 1] [1 0 1 0 0 0 1 0 1 1 0 0 1 0] [1]
Time=173, Signals: [1 0 1 1 0 1 0 1] [0 1 1 0 0 0 1 0 1 1 0 0 1 0] [1]
Time=174, Signals: [0 1 1 1 0 1 0 1] [1 0 1 0 0 0 1 0 1 1 0 0 0 0] [1]
Time=175, Signals: [1 1 1 1 0 1 0 1] [0 1 1 0 0 0 1 0 1 1 0 0 1 0] [1]
Time=176, Signals: [0 0 0 0 1 1 0 1] [1 0 1 0 1 0 0 0 1 0 0 0 1 0] [0]
Time=177, Signals: [1 0 0 0 1 1 0 1] [0 1 1 0 1 0 0 0 1 0 0 0 1 0] [0]
Time=178, Signals: [0 1 0 0 1 1 0 1] [1 0 1 1 1 0 0 1 0 0 0 0 0 1] [0]
Time=179, Signals: [1 1 0 0 1 1 0 1] [0 1 1 1 1 0 0 0 1 0 0 0 1 0] [0]
Time=180, Signals: [0 0 1 0 1 1 0 1] [1 0 1 0 1 1 0 0 1 0 0 0 1 0] [0]
Time=181, Signals: [1 0 1 0 1 1 0 1] [0 1 1 0 1 1 0 0 1 0 0 0 1 0] [0]
Time=182, Signals: [0 1 1 0 1 1 0 1] [1 0 1 1 1 1 0 1 0 0 0 0 0 1] [0]
Time=183, Signals: [1 1 1 0 1 1 0 1] [0 1 1 1 1 1 0 0 1 0 0 0 1 0] [0]
Time=184, Signals: [0 0 0 1 1 1 0 1] [1 0 1 0 1 0 1 0 1 0 0 0 1 0] [0]
Time=185, Signals: [1 0 0 1 1 1 0 1] [0 1 1 0 1 0 1 0 1 0 0 0 1 0] [0]

Time=186, Signals: [0 1 0 1 1 1 0 1] [1 0 1 1 1 0 1 1 0 0 0 0 0 1] [0]
Time=187, Signals: [1 1 0 1 1 1 0 1] [0 1 1 1 1 0 1 0 1 0 0 0 1 0] [0]
Time=188, Signals: [0 0 1 1 1 1 0 1] [1 0 1 0 0 0 1 0 1 1 0 0 1 0] [1]
Time=189, Signals: [1 0 1 1 1 1 0 1] [0 1 1 0 0 0 1 0 1 1 0 0 1 0] [1]
Time=190, Signals: [0 1 1 1 1 1 0 1] [1 0 1 0 0 0 1 0 1 1 0 0 0 0] [1]
Time=191, Signals: [1 1 1 1 1 1 0 1] [0 1 1 0 0 0 1 0 1 1 0 0 1 0] [1]
Time=192, Signals: [0 0 0 0 0 0 1 1] [1 0 1 0 1 0 0 0 1 0 0 0 1 0] [0]
Time=193, Signals: [1 0 0 0 0 0 1 1] [0 1 1 0 1 0 0 0 1 0 0 0 1 0] [0]
Time=194, Signals: [0 1 0 0 0 0 1 1] [1 0 1 1 1 0 0 1 1 0 0 0 0 1] [0]
Time=195, Signals: [1 1 0 0 0 0 1 1] [0 1 1 1 1 0 0 0 1 0 0 0 1 0] [0]
Time=196, Signals: [0 0 1 0 0 0 1 1] [1 0 1 0 1 1 0 0 1 0 0 0 1 0] [0]
Time=197, Signals: [1 0 1 0 0 0 1 1] [0 1 1 0 1 1 0 0 1 0 0 0 1 0] [0]
Time=198, Signals: [0 1 1 0 0 0 1 1] [1 0 1 1 1 1 0 1 1 0 0 0 0 1] [0]
Time=199, Signals: [1 1 1 0 0 0 1 1] [0 1 1 1 1 1 0 0 1 0 0 0 1 0] [0]
Time=200, Signals: [0 0 0 1 0 0 1 1] [1 0 1 0 1 0 1 0 1 0 0 0 1 0] [0]
Time=201, Signals: [1 0 0 1 0 0 1 1] [0 1 1 0 1 0 1 0 1 0 0 0 1 0] [0]
Time=202, Signals: [0 1 0 1 0 0 1 1] [1 0 1 1 1 0 1 1 1 0 0 0 0 1] [0]
Time=203, Signals: [1 1 0 1 0 0 1 1] [0 1 1 1 1 0 1 0 1 0 0 0 1 0] [0]
Time=204, Signals: [0 0 1 1 0 0 1 1] [1 0 0 0 1 1 0 0 1 0 0 1 1 0] [1]
Time=205, Signals: [1 0 1 1 0 0 1 1] [1 0 0 0 1 1 0 0 1 0 0 1 1 0] [1]
Time=206, Signals: [0 1 1 1 0 0 1 1] [1 0 1 1 1 1 1 1 1 0 1 1 0 1] [0]
Time=207, Signals: [1 1 1 1 0 0 1 1] [1 0 0 1 1 1 0 1 1 0 0 1 1 1] [1]
Time=208, Signals: [0 0 0 0 1 0 1 1] [1 0 1 0 1 0 0 0 1 0 0 0 1 0] [0]
Time=209, Signals: [1 0 0 0 1 0 1 1] [0 1 1 0 1 0 0 0 1 0 0 0 1 0] [0]
Time=210, Signals: [0 1 0 0 1 0 1 1] [1 0 1 1 1 0 0 1 0 0 0 0 0 1] [0]
Time=211, Signals: [1 1 0 0 1 0 1 1] [0 1 1 1 1 0 0 0 1 0 0 0 1 0] [0]
Time=212, Signals: [0 0 1 0 1 0 1 1] [1 0 1 0 1 1 0 0 1 0 0 0 1 0] [0]
Time=213, Signals: [1 0 1 0 1 0 1 1] [0 1 1 0 1 1 0 0 1 0 0 0 1 0] [0]
Time=214, Signals: [0 1 1 0 1 0 1 1] [1 0 1 1 1 1 0 1 0 0 0 0 0 1] [0]
Time=215, Signals: [1 1 1 0 1 0 1 1] [0 1 1 1 1 1 0 0 1 0 0 0 1 0] [0]
Time=216, Signals: [0 0 0 1 1 0 1 1] [1 0 1 0 1 0 1 0 1 0 0 0 1 0] [0]
Time=217, Signals: [1 0 0 1 1 0 1 1] [0 1 1 0 1 0 1 0 1 0 0 0 1 0] [0]
Time=218, Signals: [0 1 0 1 1 0 1 1] [1 0 1 1 1 0 1 1 0 0 0 0 0 1] [0]
Time=219, Signals: [1 1 0 1 1 0 1 1] [0 1 1 1 1 0 1 0 1 0 0 0 1 0] [0]
Time=220, Signals: [0 0 1 1 1 0 1 1] [1 0 0 0 1 1 0 0 1 0 0 1 1 0] [1]
Time=221, Signals: [1 0 1 1 1 0 1 1] [1 0 0 0 1 1 0 0 1 0 0 1 1 0] [1]
Time=222, Signals: [0 1 1 1 1 0 1 1] [1 0 1 1 1 1 1 1 0 0 1 1 0 1] [0]
Time=223, Signals: [1 1 1 1 1 0 1 1] [1 0 0 1 1 1 0 1 0 0 0 1 1 1] [1]

Time=224, Signals: [0 0 0 0 0 1 1 1] [1 0 1 0 1 0 0 0 1 0 0 0 1 0] [0]
Time=225, Signals: [1 0 0 0 0 1 1 1] [0 1 1 0 1 0 0 0 1 0 0 0 1 0] [0]
Time=226, Signals: [0 1 0 0 0 1 1 1] [1 0 1 1 1 0 0 1 1 0 0 0 0 1] [0]
Time=227, Signals: [1 1 0 0 0 1 1 1] [0 1 1 1 1 0 0 0 1 0 0 0 1 0] [0]
Time=228, Signals: [0 0 1 0 0 1 1 1] [1 0 1 0 1 1 0 0 1 0 0 0 1 0] [0]
Time=229, Signals: [1 0 1 0 0 1 1 1] [0 1 1 0 1 1 0 0 1 0 0 0 1 0] [0]
Time=230, Signals: [0 1 1 0 0 1 1 1] [1 0 1 1 1 1 0 1 1 0 0 0 0 1] [0]
Time=231, Signals: [1 1 1 0 0 1 1 1] [0 1 1 1 1 1 0 0 1 0 0 0 1 0] [0]
Time=232, Signals: [0 0 0 1 0 1 1 1] [1 0 1 0 1 0 1 0 1 0 0 0 1 0] [0]
Time=233, Signals: [1 0 0 1 0 1 1 1] [0 1 1 0 1 0 1 0 1 0 0 0 1 0] [0]
Time=234, Signals: [0 1 0 1 0 1 1 1] [1 0 1 1 1 0 1 1 1 0 0 0 0 1] [0]
Time=235, Signals: [1 1 0 1 0 1 1 1] [0 1 1 1 1 0 1 0 1 0 0 0 1 0] [0]
Time=236, Signals: [0 0 1 1 0 1 1 1] [1 0 0 0 0 0 0 0 1 1 0 1 1 0] [1]
Time=237, Signals: [1 0 1 1 0 1 1 1] [1 0 0 0 0 0 0 0 1 1 0 1 1 0] [1]
Time=238, Signals: [0 1 1 1 0 1 1 1] [1 0 1 0 0 0 1 0 1 1 0 1 0 0] [1]
Time=239, Signals: [1 1 1 1 0 1 1 1] [1 0 0 0 0 0 0 0 1 1 0 1 1 0] [1]
Time=240, Signals: [0 0 0 0 1 1 1 1] [1 0 1 0 1 0 0 0 1 0 0 0 1 0] [0]
Time=241, Signals: [1 0 0 0 1 1 1 1] [0 1 1 0 1 0 0 0 1 0 0 0 1 0] [0]
Time=242, Signals: [0 1 0 0 1 1 1 1] [1 0 1 1 1 0 0 1 0 0 0 0 0 1] [0]
Time=243, Signals: [1 1 0 0 1 1 1 1] [0 1 1 1 1 0 0 0 1 0 0 0 1 0] [0]
Time=244, Signals: [0 0 1 0 1 1 1 1] [1 0 1 0 1 1 0 0 1 0 0 0 1 0] [0]
Time=245, Signals: [1 0 1 0 1 1 1 1] [0 1 1 0 1 1 0 0 1 0 0 0 1 0] [0]
Time=246, Signals: [0 1 1 0 1 1 1 1] [1 0 1 1 1 1 0 1 0 0 0 0 0 1] [0]
Time=247, Signals: [1 1 1 0 1 1 1 1] [0 1 1 1 1 1 0 0 1 0 0 0 1 0] [0]
Time=248, Signals: [0 0 0 1 1 1 1 1] [1 0 1 0 1 0 1 0 1 0 0 0 1 0] [0]
Time=249, Signals: [1 0 0 1 1 1 1 1] [0 1 1 0 1 0 1 0 1 0 0 0 1 0] [0]
Time=250, Signals: [0 1 0 1 1 1 1 1] [1 0 1 1 1 0 1 1 0 0 0 0 0 1] [0]
Time=251, Signals: [1 1 0 1 1 1 1 1] [0 1 1 1 1 0 1 0 1 0 0 0 1 0] [0]
Time=252, Signals: [0 0 1 1 1 1 1 1] [1 0 0 0 0 0 0 0 1 1 0 1 1 0] [1]
Time=253, Signals: [1 0 1 1 1 1 1 1] [1 0 0 0 0 0 0 0 1 1 0 1 1 0] [1]
Time=254, Signals: [0 1 1 1 1 1 1 1] [1 0 1 0 0 0 1 0 1 1 0 1 0 0] [1]
Time=255, Signals: [1 1 1 1 1 1 1 1] [1 0 0 0 0 0 0 0 1 1 0 1 1 0] [1]