Circuit Verification

Bijoy Singh (120050087)
Sai Kiran Mudulkar(120050068)
Manik Dhar(120050006)

Problem

Digital circuit simulator:

Use prolog to implement AND, OR, NAND and NOT gates;

then simulate more complex circuits

Method 1 Logic Programming

This method uses prolog logic and the connections that are between various gates are implemented using the variables in Prolog.

The Facts

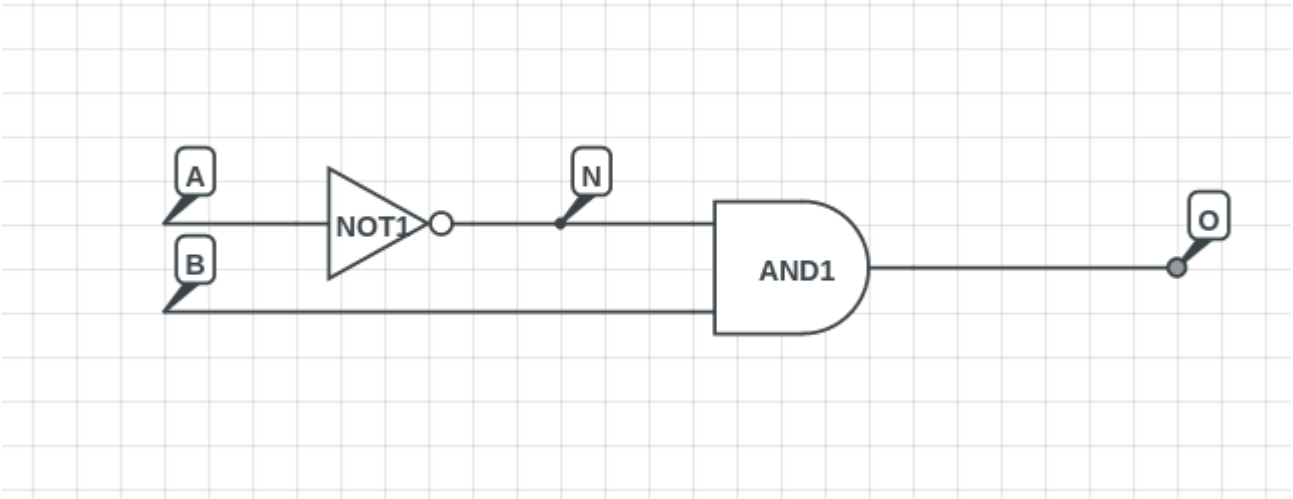
```
not([0],1).
not([1],0).

and([1,1],1).
and([0,1],0).
and([1,0],0).
and([0,0],0).

or([0,0],0).
or([0,1],1).
or([1,0],1).
or([1,1],1).
```

The Algorithm

The idea this method uses is that, for each connecting wire you assign variables.



Algorithm contd.

Now write the boolean condition for each of the gates and simply AND all these conditions

```
circuit(A,B,O):-
   not(A,N) , and(N,B,O).
```

This method allows for using circuit(A,B,O) in other modules

Benefits Of Method 1

1) Smaller and more intuitive code
2) The gates that you define previously are reusable and hence bigger logic circuits will be more modular
3) This method allows Prolog to backtrack.
4) So stronger queries like XOR([A,0],1). can be called giving the output of 1.

Additions to Method 1

We have added a system which stores definitions of user-defined circuits which can be used later to define new circuits.

Method 2:

Here we define circuits in a more traditional way by defining components, assigning them values, and attaching these components to each other.

In this method we used dynamic predicates, added using aassert and removed using retractall.

Definitions Used:

The basic facts state what are the inputs of the circuit and what are the outputs.

Other facts include which connectors are connected to each other and which connectors are connected to gates.

Definition of the circuit:

Finally we add a predicate which asserts signal values over the inputs, and asserts the type of the gates used. The definition of the primitive gates leads to addition of a statement which assigns a value to the output wire.

In the end we check, if certain constraints are satisfied for eg: No dangling wires, no short circuits,

Automated Code Generation

As this process of writing code on prolog with connections is pretty cumbersome and requires a lot of accuracy, we made a method by which the users can simply add the boolean function and the system will generate the prolog code for that boolean function for testing.

Thank You