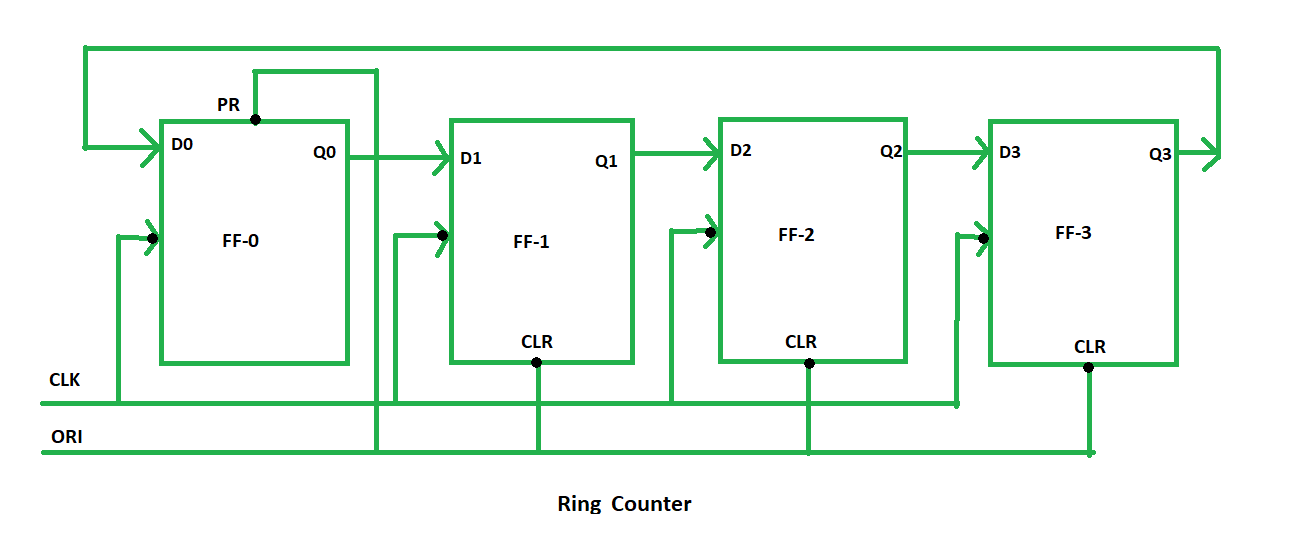
**4Bit\_ring\_counter\_iiitb**

**INTRODUCTION**

This project simulates the design of a 4-bit ring counter using verilog HDL. A ring counter works in a similar way as a shift register. The only difference is that the output of the last flip-flop is connected to the input of the first flip-flop. In this way, the counter forms a ring and hence is called ring counter. In this design, four D-Flip-flops are used with clock and ori(override input) signals.

**BLOCK DIAGRAM**

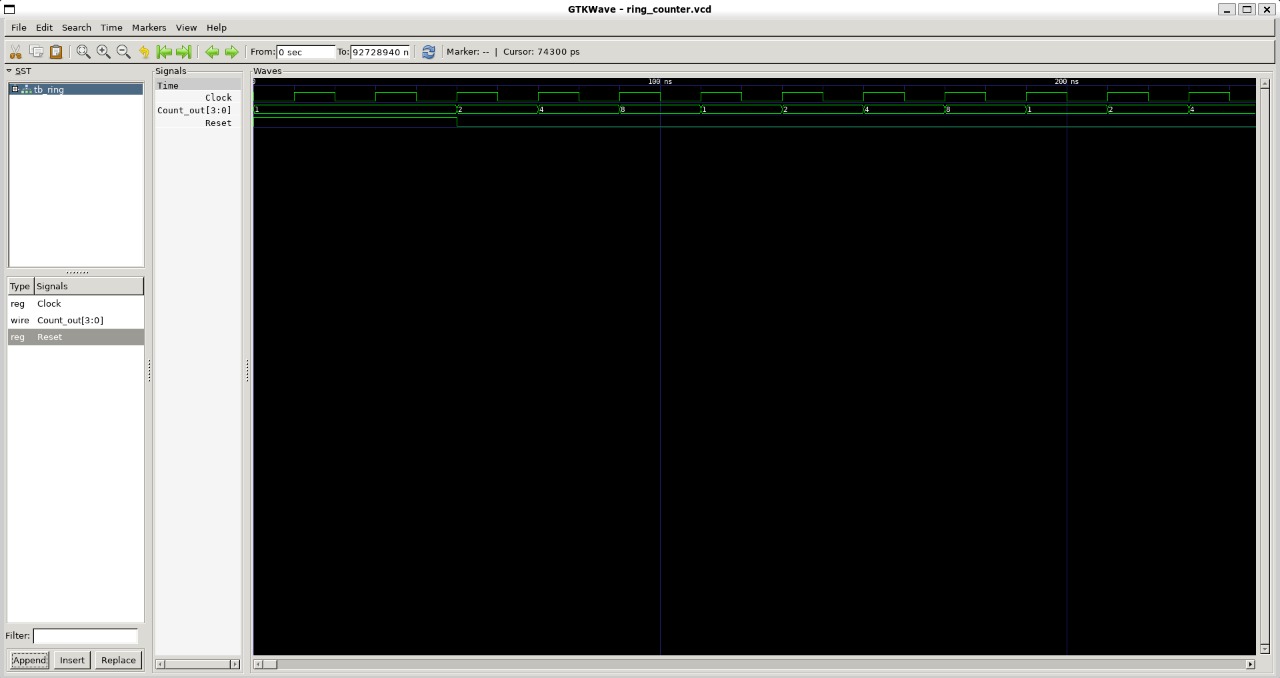


The above figure is the block diagram of a 4-bit ring counter. The figure shows four D flip flop connected with a clock and an ORI signal. The design uses an active high ORI signal which sets the first flip flop to '1' and the other three flip flops to '0' when ORI is high. The circuit uses a positive edge triggered clock.

**WORKING**

The counter is set to an initial state of *'1000'* by the ORI signal. In the next positive edge of the clock, the values of the flip flops are shifted right and the output of last flip flop is sent to the first one. So, the next state becomes *'0100'*. Similarly after next positive edge of clock, the state of the counter becomes *'0010'*. This continues until the ORI is again high which will set the counter back to *'1000'*.

**RTL SIMULATION**



In the above waveform, ORI signal sets the counter to '1000' and then the counter runs in a loop with three states until ORI is high again.

**TOOLS USED**

**IVERILOG**  
  
Icarus Verilog is a Verilog simulation and synthesis tool.  
To install iverilog, type the following command in the terminal:

$ sudo apt install iverilog

**GTKWAVE**  
  
GTKWave is a VCD waveform viewer based on the GTK library. This viewer support VCD and LXT formats for signal dumps.

$ sudo apt install gtkwave

**YOSYS**  
Yosys is a framework for Verilog RTL synthesis. It currently has extensive Verilog-2005 support and provides a basic set of synthesis algorithms for various application domains.

Synthesis transforms the simple RTL design into a gate-level netlist with all the constraints as specified by the designer. In simple language, Synthesis is a process that converts the abstract form of design to a properly implemented chip in terms of logic gates.

Synthesis takes place in multiple steps:

-Converting RTL into simple logic gates.  
-Mapping those gates to actual technology-dependent logic gates available in the technology libraries.  
-Optimizing the mapped netlist keeping the constraints set by the designer intact.  
  
Yosys can be adapted to perform any synthesis job by combining the existing passes (algorithms) using synthesis scripts and adding additional passes as needed by extending the yosys C++ code base.

Yosys is free software licensed under the ISC license (a GPL compatible license that is similar in terms to the MIT license or the 2-clause BSD license).  
To install Yosys in Ubuntu, follow the following steps:

$ sudo apt-get install build-essential clang bison flex \ libreadline-dev gawk tcl-dev libffi-dev git \ graphviz xdot pkg-config python3 libboost-system-dev \ libboost-python-dev libboost-filesystem-dev zlib1g-dev

To configure the build system to use a specific compiler, use one of the following command:

$ make config-clang

$ make config-gcc

To build Yosys simply type 'make' in this directory.

$ make

$ sudo make install

$ make test

**Diagram

Description automatically generated**

**Text

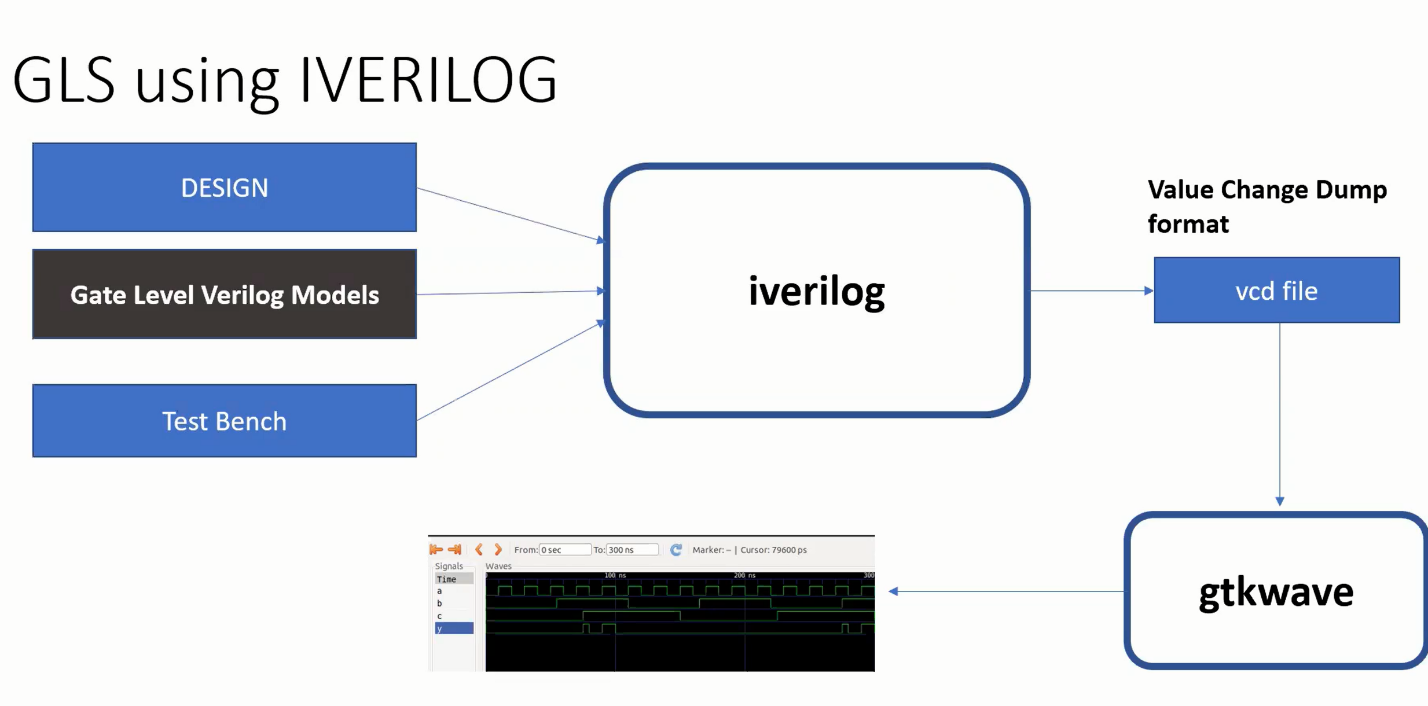
Description automatically generated**

**GTL- Gate Level Simulation**

GLS is generating the simulation output by running test bench with netlist file generated from synthesis as design under test. Netlist is logically same as RTL code, therefore, same test bench can be used for it.  
Below picture gives an insight of the procedure. Here while using iverilog, you also include gate level verilog models to generate GLS simulation.

**Graphical user interface

Description automatically generated**

[](https://user-images.githubusercontent.com/64605104/183838608-b56e1d75-929d-492a-b112-8203a5e40cff.png)

**ACKNOWLEDGMENTS**

* Kunal Ghosh, Director, VSD Corp. Pvt. Ltd.

**CONTRIBUTORS**

* Sirigiri Sai Keerthan, IMtech IIIT Bangalore
* Kunal Ghosh Director, VSD Corp.Pvt.Ltd.

**CONTACT**

* Kunal Ghosh, Director, VSD Corp. Pvt. Ltd. kunalghosh@gmail.com
* Sirigiri Sai Keerthan, iMtech IIIT Bangalore. [SaiKeerthan.Sirigiri@iiitb.ac.in](mailto:SaiKeerthan.Sirigiri@iiitb.ac.in)

**REFERENCES**

1] <https://en.wikipedia.org/wiki/Ring_counter>

[2] <https://www.geeksforgeeks.org/ring-counter-in-digital-logic/>

[3] <https://www.javatpoint.com/verilog-ring-counter>

[4] <https://www.allaboutcircuits.com/textbook/digital/chpt-12/ring-counters/>

[5] <https://github.com/ArshKedia/iiitb_3bit_rc/blob/main/README.md>