Power Sequence Block
SLG47910

This abstract shows how to use structural modeling in the ForgeFPGA Software. The three instances of delay modules are used to create a Power Sequencing Function. This Application Note is designed with ForgeFPGA Workshop v6.43.

This application note comes complete with a design file which can be found in the Reference Section.

Contents

1. Terms and Definitions

FPGA	Field Programmable Gate Array
FPGA Editor	Main FPGA design and simulation window
Go Configure	Software Hub Main window for device selection
ForgeFPGA Window	Main FPGA project window for debug and IO programming

2. References

For related documents and software, please visit: https://www.renesas.com/eu/en/products/programmable-mixed-signal-asic-ip-products/forgefpga-low-density-fpgas.Download our free ForgeFPGA™ Designer software [1] to open the .ffpga design files [2] and view the proposed circuit design.

[1] ForgeFPGA Designer Software, Software Download and User Guide

[2] AN-FG-008 Power Sequence Example.ffpga, ForgeFPGA Design File

[3] ForgeFPGA SLG47910, Datasheet, Renesas Electronics

3. Introduction

The Delay Block is a module that postpones rising and/or falling edges for the duration of the register values. The Delay Block has 5 inputs as seen in Figure 1.

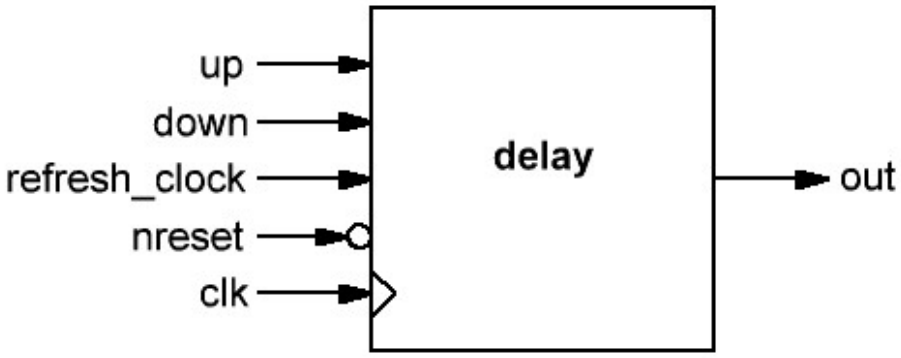

Figure 1: Delay Block Pin Diagram

The following signal names are the PINs that are used in the design:

- clk - input clock signal
- nreset - input negative reset signal
- refresh_clock - divide clock from the external counter
- up - input up signal
- down - input down signal (Highest Priority)
- out - output delay signal

The register value is defined by two parameters: RISING_COUNTER_DATA & FALLING_COUNTER_DATA. These two parameters define the length of the rising and falling delay period. By default, it is set to 255 in the delay Block Design. If the input signal is shorter than the delay time, it can work as filter. In Figure 2, assume that the RISING_COUNTER_DATA = FALLING_COUNTER_DATA = 2, hence we can observe that the Rising DLY1 Edge goes High after 2 clock cycles from when the UP signal goes High. Similarly, the Falling DLY1 edge goes Low after two clock cycles of the rising edge of the DOWN signal. As the DOWN signal has the highest priority, the Falling edge of the DOWN signal triggers a Rising DLY1 edge again on the OUT signal.

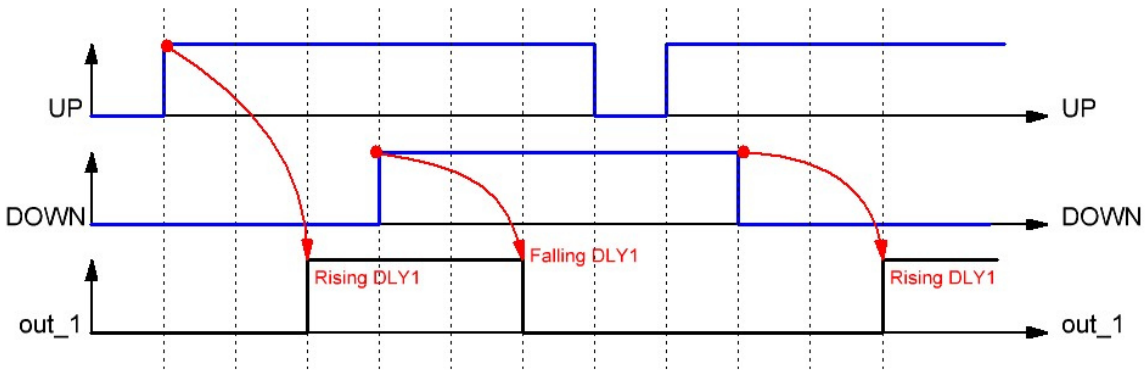

Figure 2: Delay Block Functionality Waveform

This application note is focused on designing a Power Sequencing Function which has 3 Delay Block instances (see Figure 3). We can see that the output of each Delay Block becomes a control signal for the other Delay Blocks. Figure 4 shows the output waveform functionality of the Power Sequencing where each Rising and Falling Edge of the Output Signal depends on the RISING_COUNTER_DATA & FALLING_COUNTER_DATA parameter value specified in each Delay Block Instance.

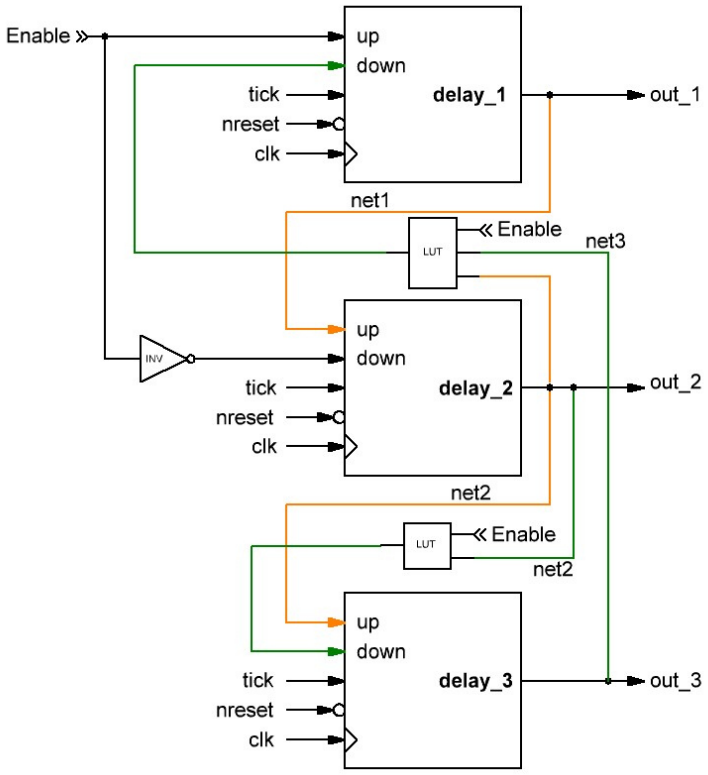

Figure 3: Power Sequencing Block Diagram

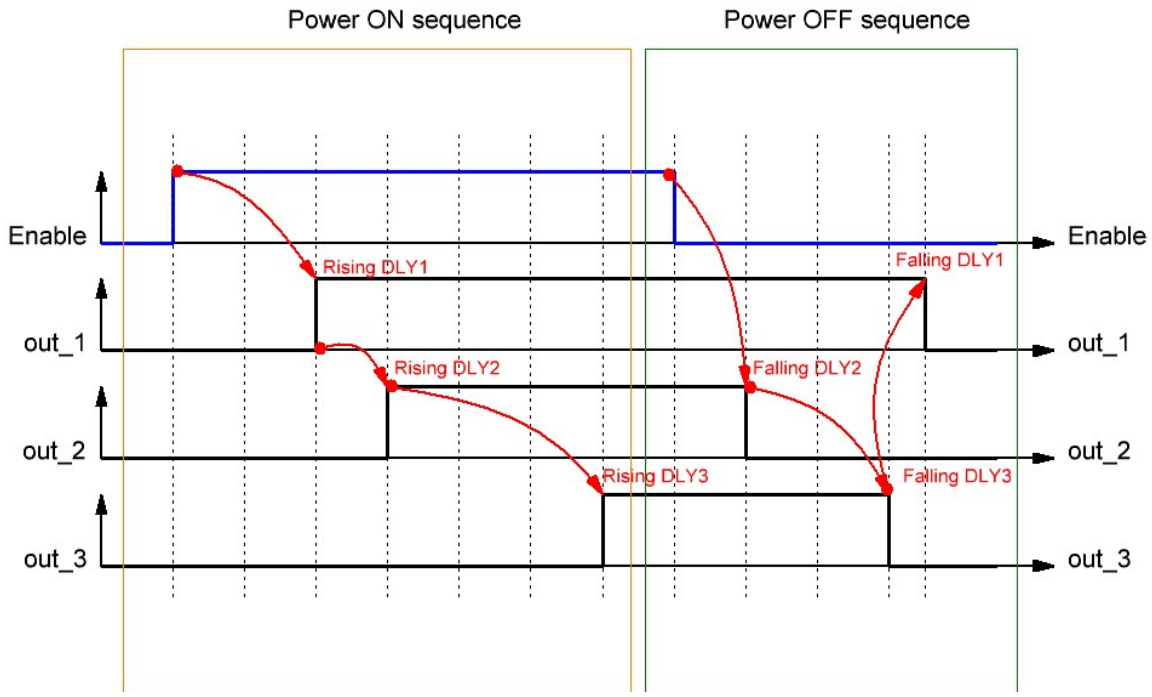

Figure 4: Power Sequencing Output Waveform

Using the ForgeFPGA Workshop Software, the Verilog code was synthesized and the Bitstream was loaded on the SLG47910 device.

4. Ingredients

- ForgeFPGA Device SLG47910V
- Latest Revision of ForgeFPGA Workshop software
- GTKWave software (inbuilt in ForgeFPGA Workshop software)

5. Delay Verilog Code

Shown below is the (*top*) module called Delay_top. It is available for download (AN-FG-008 Power Sequence Example.ffpga). The design below uses the Delay Module instantiation.

```
(* top *) module delay_top #(
  parameter WIDTH = 8
)(
// Main inputs
   (* iopad_external_pin, clkbuf_inhibit *) input clk,
   (* iopad_external_pin *) input nreset,
   (* iopad_external_pin *) input enable,
   (* iopad_external_pin *) output out_1,
   (* iopad_external_pin *) output out_2,
   (* iopad_external_pin *) output out_3,
   (* iopad_external_pin *) output out_1_oe,
   (* iopad_external_pin *) output out_2_oe,
   (* iopad_external_pin *) output out_3_oe,
   (* iopad_external_pin *) output OSC_CTRL_EN
);
 //osc
  assign OSC_CTRL_EN = 1'b1;
 //oe
  assign out_1_oe = 1'b1;
  assign out_2_oe = 1'b1;
  assign out_3_oe = 1'b1;

  reg [7:0] counter;

  wire tick;
  wire net1;
  wire net2;
  wire net3;

// refresh clock
  always @(posedge clk) begin
    if (!nreset)
      counter <= 'b0;
    else if ( counter == 10)
      counter <= 'b0;
    else
      counter <= counter + 1;
  end

  assign tick = (counter == 10)? 1'b1 : 1'b0;
```

```
   delay_1 #(
  .RISING_COUNTER_DATA (4),
  .FALLING_COUNTER_DATA (3),
  .WIDTH (WIDTH)
 ) delay_1_wrapper(
  .clk (clk),
  .up (enable),
  .down (~enable),
  .nreset (nreset),
  .refresh_clock (tick),
  .out (net1)
  );

   delay_1 #(
  .RISING_COUNTER_DATA (5),
  .FALLING_COUNTER_DATA (5),
  .WIDTH (WIDTH)
 ) delay_2_wrapper(
  .clk (clk),
  .up (net1),
  .down ( ~net1 & ~enable),
  .nreset (nreset),
  .refresh_clock (tick),
  .out (net2)
  );

   delay_1 #(
  .RISING_COUNTER_DATA (10),
  .FALLING_COUNTER_DATA (5),
  .WIDTH (WIDTH)
 ) delay_3_wrapper(
  .clk (clk),
  .up (net2 & enable),
  .down (~net2 & ~enable),
  .nreset (nreset),
  .refresh_clock (tick),
  .out (net3)
  );

  assign out_1 = net1;
  assign out_2 = net2;
  assign out_3 = net3;

endmodule
```

6. Floorplan: CLB Utilization

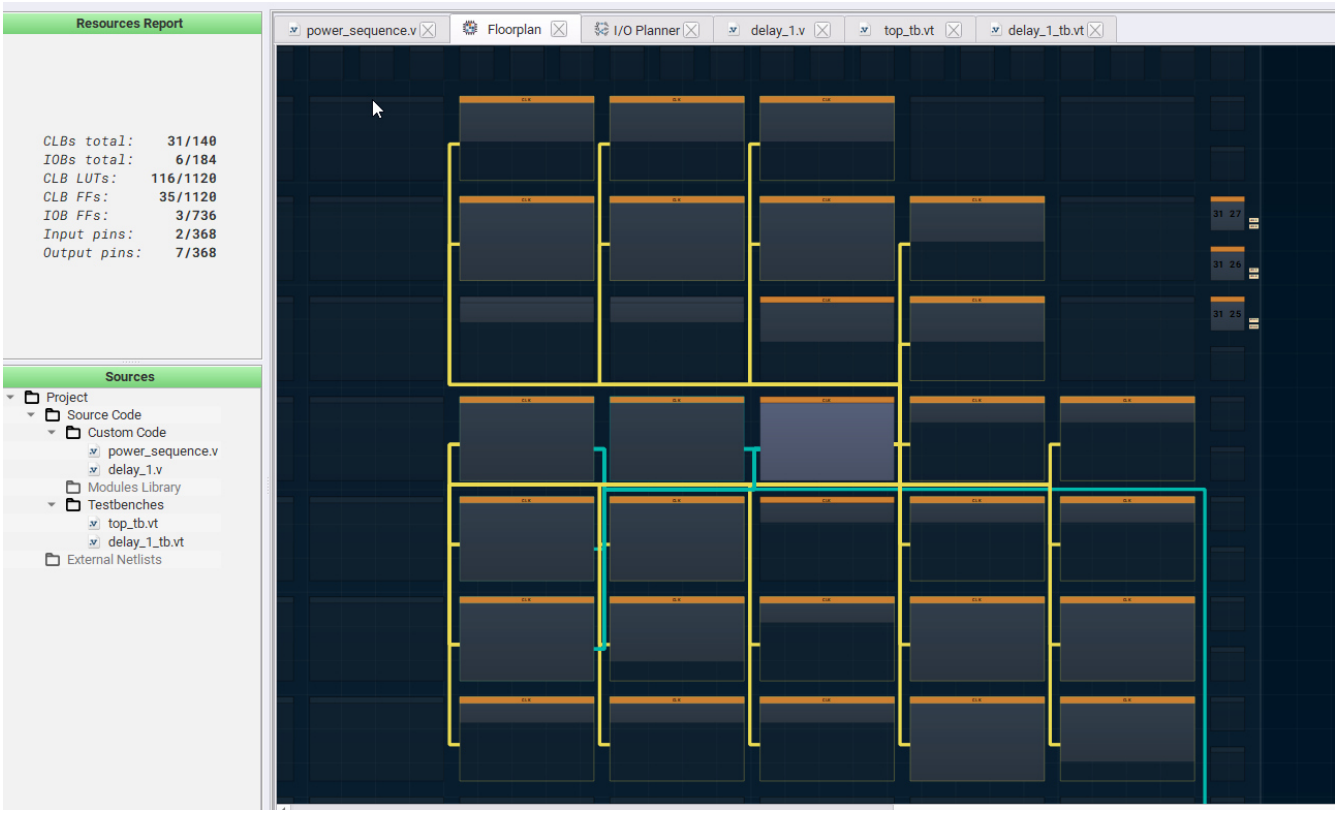

Figure 5: CLB Utilization

The Floor planner tab in the FPGA Editor shows the placement of CLBs and FFs (Figure 3). The resource utilization is shown in the top left corner.

7. Design Steps

1. Launch the latest version of the Go Configure Software Hub. Select the SLG47910V device and the ForgeFPGA Workshop software will load.
2. Download the design example Power Sequence Example.ffpga. If you are not familiar with the ForgeFPGA Workshop software, review the Four-Bit Counter application notes that cover the basic design steps.
3. Open the Power Sequence Example.ffpga file after downloading.
4. Open the FPGA editor and review the Verilog code and the testbench code. There is a main code with the module name Delay_top which is the top module defining the whole design. This code has 3 instances for Delay Block mapping.
5. Open the IO planner tab on the FPGA editor and review the pin assignment.
6. Next select the Synthesize button on the lower left side of the FPGA editor. Select the Generate Bitstream button on the lower left side of the FPGA editor. Check the Logger and Issues tabs to make sure that the bit steam was generated correctly.
7. Now click on the Floorplan tab and see the CLB utilization (Figure 5). Press the Ctrl and the mouse wheel to zoom-in. Confirm that the IOs selected in the IO Planner are shown in the floorplan.
8. Now click on the Simulate Testbench button on the top and select **top_tb** to simulate the testbench for the complete design. The GTKWave will automatically open if there are no Syntax errors in the testbench. Check logger for errors.
9. In the GTKWave software, select the signals you want to view and Click Insert on the left corner to insert the signals in the wave window. Once the desired signals are selected, click on Reload (Figure 4).

10. You can observe the countdown in the waveform displayed in the GTKWave software.

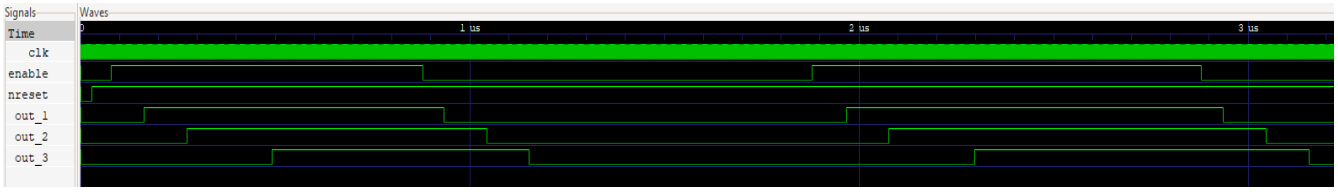

Figure 6: GTWave Simulation Results

8. Conclusions

The application note shows how the use structural modeling style by instantiating the Delay module in the design and how it can be used to delay the Rising and Falling edge of the input with the help of different Register Values thus forming a the whole power sequence design. This testcase is available for download (AN-FG-008 Power Sequence Example.ffpga). If interested, please contact the ForgeFPGA Business Support Team.

9. Revision History

Revision	Date	Description
1.0	26-Apr-2022	Initial Version
2.0	23-Feb-2024	Updated as per BB revision
3.0	17-Jul-24	Updated as per software v6.43

IMPORTANT NOTICE AND DISCLAIMER

RENESAS ELECTRONICS CORPORATION AND ITS SUBSIDIARIES (“RENESAS”) PROVIDES TECHNICAL SPECIFICATIONS AND RELIABILITY DATA (INCLUDING DATASHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT OF THIRD-PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for developers who are designing with Renesas products. You are solely responsible for (1) selecting the appropriate products for your application, (2) designing, validating, and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, or other requirements. These resources are subject to change without notice. Renesas grants you permission to use these resources only to develop an application that uses Renesas products. Other reproduction or use of these resources is strictly prohibited. No license is granted to any other Renesas intellectual property or to any third-party intellectual property. Renesas disclaims responsibility for, and you will fully indemnify Renesas and its representatives against, any claims, damages, costs, losses, or liabilities arising from your use of these resources. Renesas' products are provided only subject to Renesas' Terms and Conditions of Sale or other applicable terms agreed to in writing. No use of any Renesas resources expands or otherwise alters any applicable warranties or warranty disclaimers for these products.

(Disclaimer Rev.1.01)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

Contact Information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit www.renesas.com/contact-us/.

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.