How to Create Customized Code (Full Adder)
SLG47910V

Abstract

This application shows how to combine multiple custom modules and implement it using the SLG47910 FPGA. This application is demonstrated with the help of a Full Adder example. The functionality of the full adder is verified by creating a Testbench and observing the output waveforms on the inbuilt GTKWave Software. This application note comes complete with design files which can be found in the References section.

Contents

1. Terms and Definitions

CLB	Configuration Logic Block
HDL	Editor Workspace where Verilog code is entered
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.dialog-semiconductor.com/products/greenpak/low-power-low-cost-forgefpga . 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-002 How to Create Customized Code (Full Adder).ffpga, ForgeFPGA Design File, Renesas Electronics

[3] ForgeFPGA SLG47910 Datasheet, Renesas Electronics

3. Introduction

A Full Adder module is created with the help of 2 Half Adder modules and an OR Gate module mapped together.

Structural modelling can be implemented in ForgeFPGA Workshop by defining the topmost module with the keywork (*top*). This ensures that the software knows the difference between the topmost module and the modules mapped to it using port mapping syntax. The functionality of the Full Adder can be verified by creating a Test Bench and providing simulation within it. The simulation can be visualized using GTKWave Software within the ForgeFPGA software. This Application is designed to understand the different aspects of the software without the need of programming the part to check functionality.

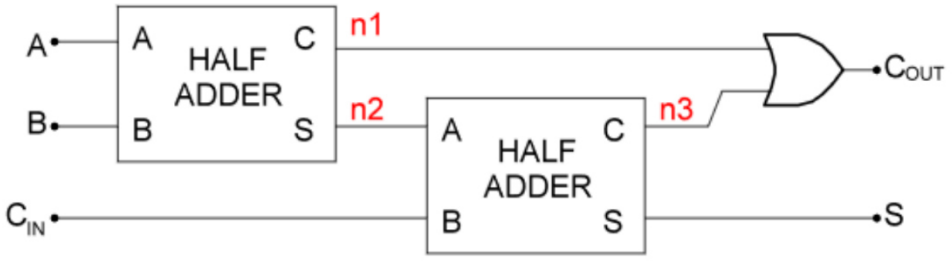

Figure 1: System Diagram

4. Ingredients

- Latest Revision of ForgeFPGA Workshop software

5. Verilog Code

Shown below is the (*top*) module called Full_Adder. The Verilog code displays how to connect different modules under one top module using synchronous style of Verilog coding.

```
//Full adder coding example

(*top*)module Full_Adder(
 (* iopad_external_pin *) input A,
 (* iopad_external_pin *) input B,
 (* iopad_external_pin *) input Cin,
 (* iopad_external_pin *) output Sum,
 (* iopad_external_pin *) output Sum_oe,
 (* iopad_external_pin *) output Cout_oe,
 (* iopad_external_pin *) output Cout
);

wire n1,n2,n3;
//OE
assign Sum_oe = 1;
assign Cout_oe = 1;

//port mapping
Half_Adder   Half_Adder_1 (.x(A), .y(B), .sum(n2) , .carry(n1));;
Half_Adder   Half_Adder_2 (.sum(Sum), .carry(n3), .x(n2) , .y(Cin));
OR_Gate      OR_Gate_1(.a(n1), .b(n3) ,.c(Cout));
```

```
endmodule

//Half Adder Module for Full Adder
module Half_Adder
(input x,y,
output sum,carry );

 assign sum = x ^ y;
 assign carry = x & y;

 endmodule
//OR gate module for Full Adder
module OR_Gate(
    input a,b,
    output c );

assign c = a | b; //OR gate operation

endmodule
```

6. Test Bench

The Test Bench is used to provide Stimulus to verify the correctness or working of our DUT, hence we must instantiate our design module. The module name must be given as to be recognized by the ForgeFPGA Studio as a testbench file (see Figure 2).

The functionality can be verified using the waveforms produced by GTKWave software inbuilt.

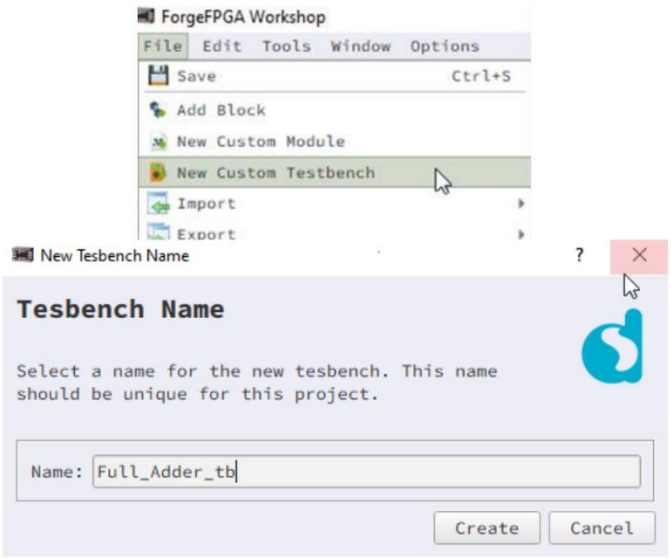

Figure 2: Steps to Launch Custom Testbench

7. Design Steps

1. Open the ForgeFPGA Workshop software in GreenPAK and select the SLG47910 device From the ForgeFPGA tool bar, select the FPGA Editor Tab (see Figure 5).

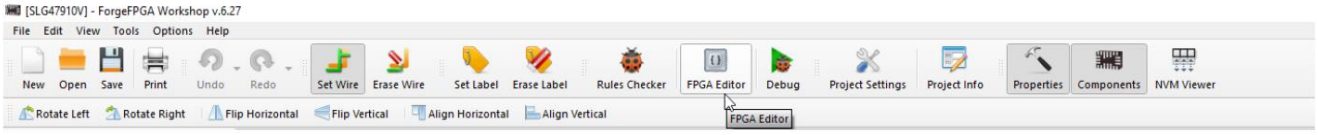

Figure 5: ForgeFPGA Tool Bar

2. Create 3 different Custom Modules called OR_Gate , Half_Adder & Full_Adder and enter the respective Verilog Codes in the HDL Editor for each module. (see Figure 6).

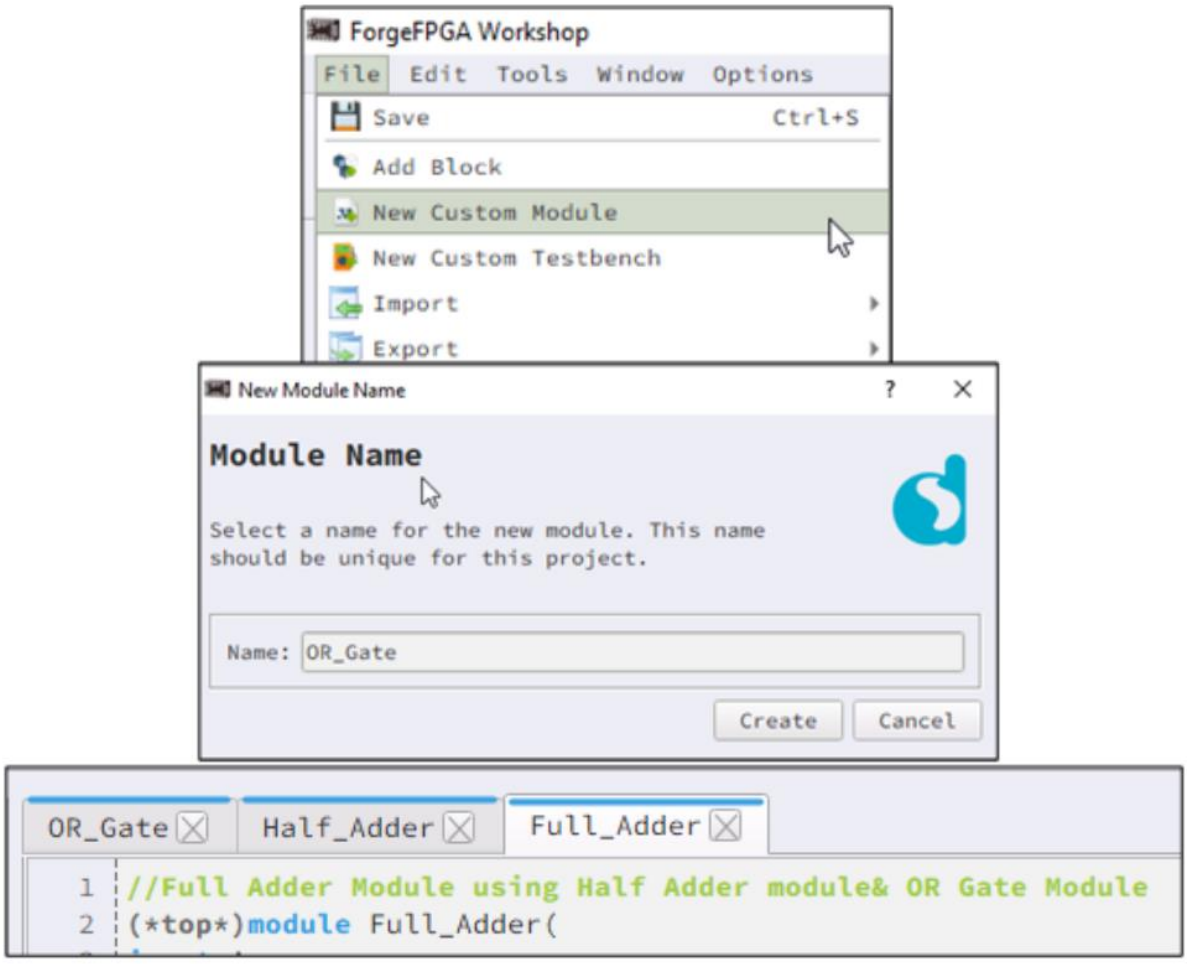

Figure 6: Steps to Launch Custom Module

3. Define the Full_Adder module as (*top*) so that it gets recognized as the topmost module and using port mapping, map the OR_Gate & Half_Adder module to it and save the code using the save button in the top left corner of the FPGA Editor

4. Check for any Syntax Errors using the Synthesize Button on the lower left side of the FPGA Editor & Check logger for any issues /warnings

5. Click on Generate Bitstream button. This generates bitstream file and other build files such as Floorplan Mapping, Routing information, Netlist, Timing Analysis, Resource Utilization report used in this design. All these information can be accessed by the different tabs found at the top of the ForgeFPGA Workshop software. Check the logger and issues tab on the bottom to make sure that the bit stream was generated correctly.

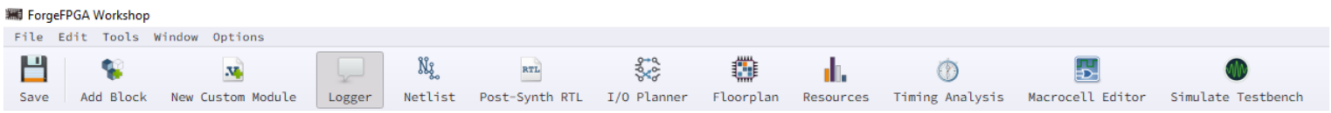

Figure 7: ForgeFPGA Tool Bar

6. Create Test Bench module and define the different inputs you want to run the code through to verify functionality at different time intervals. Simulate the testbench through the Simulate Testbench button on the toolbar (see Figure 7).

If there are no syntax errors on the Testbench written, then the GTKWave software should pop-up automatically provided that the naming of the module is correct.

7. Load the signals in the GTKWave software the needs to be observed. After this step, you should observe green/ red lines right side of the screen. Click ‘Zoom Fit’ to fir the waveforms in the selected size of the window and then click ‘Reload’ (see Figure 8).

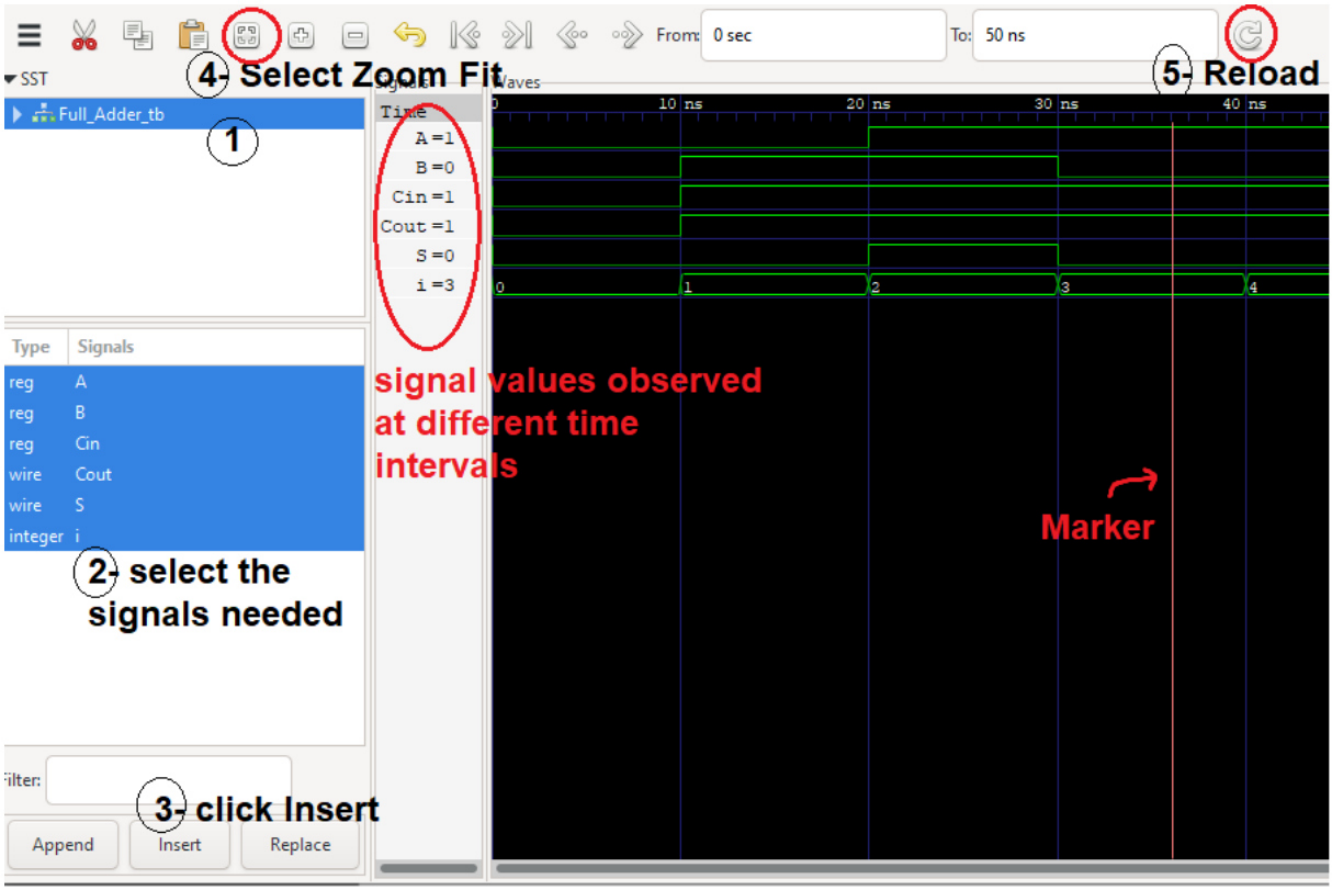

Figure 8: GTKWave Simulation Waveform

8. The corresponding waveforms can be observed. Each signal value matches the value assigned in the Test Bench code. With the help of the Marker, the values of the signals can be observed at different times.

8. Conclusion

The procedure outlines in this application example can be applied to any circuit by creating the custom modules on the similar lines and then port mapping them to the top module to make it a whole circuit. The usage of defining a module as "top" module has been demonstrate and the functionality has been verified using the inbuild GTKWave software and testbench. This testcase is available for download. If interested, please contact the ForgeFPFA Business Support Team

9. Revision History

Revision	Date	Description
1.0	Dec 10, 2021	Initial release.
2.0	Feb 20, 2024	Revised according to BB revision
3.0	Jul 17,2024	Updated as per ForgeFPGA Workshop v6.43