Using Placement Constraints in ForgeFPGA Designs

SLG47921

Abstract

This application note shows how to use placement constraints in the ForgeFPGA™ design to control where specific logic cells are located on the chip. It is applicable to all devices from the ForgeFPGA product family (SLG47910/12, SLG47920/21). This document (using the SLG47921 as an example) comes complete with a design file that can be found in section 8 References.

Contents

Figures

1. Introduction

Placement constraints refer to the rules or limitations applied during the physical design phase of the FPGA development to control where specific components or blocks are placed on the chip. These constraints ensure that the design meets performance, timing, and manufacturing requirements.

2. Floorplan System Coordinates

The floorplan is composed of Input Output Blocks (IOBs), which are marked with green and red squares, and Configurable Logic Blocks (CLBs), marked with yellow squares. Each of these blocks is assigned specific X and Y coordinates denoting their location on the floorplan starting with [0,0] in the bottom left.

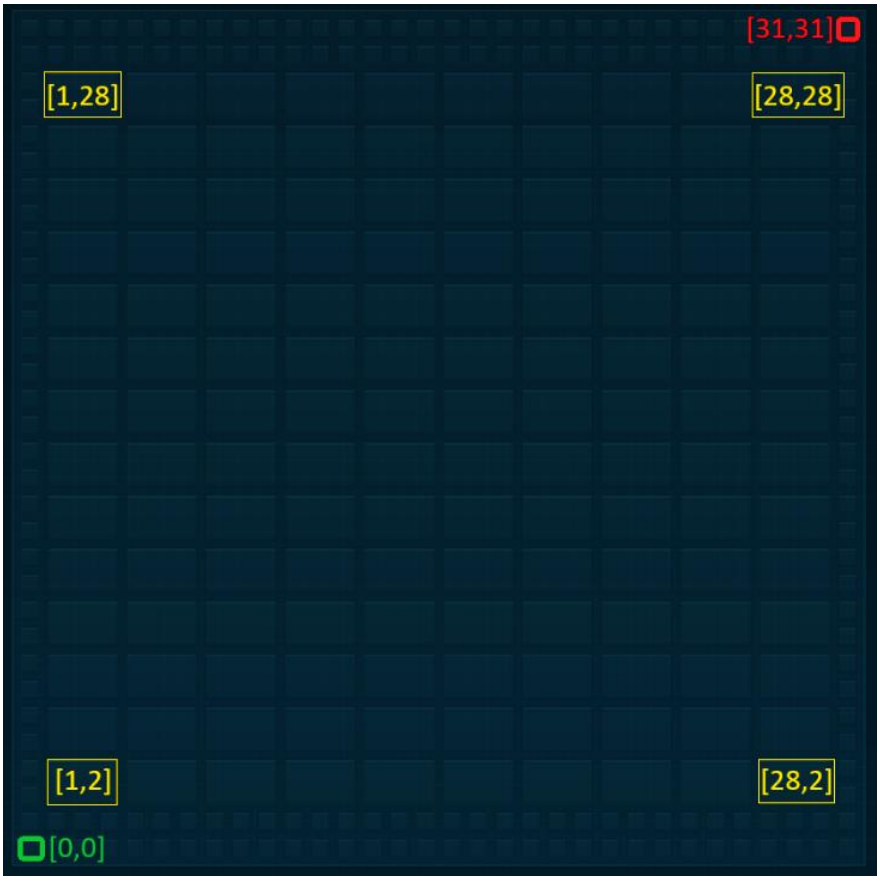

Figure 1. Floorplan

Typically, a CLB occupies three IOBs horizontally and two vertically. The initial CLB begins at coordinates [1,2]. To determine the coordinates for each CLB, use the following formulas:

$$X_n = 3n + 1$$

$$Y_n = 2n + 2$$

where:

x, y – CLB coordinates

n – number of CLBs (counting from 0)

The floorplan coordinates can be viewed in the ForgeFPGA Workshop (Figure 2).

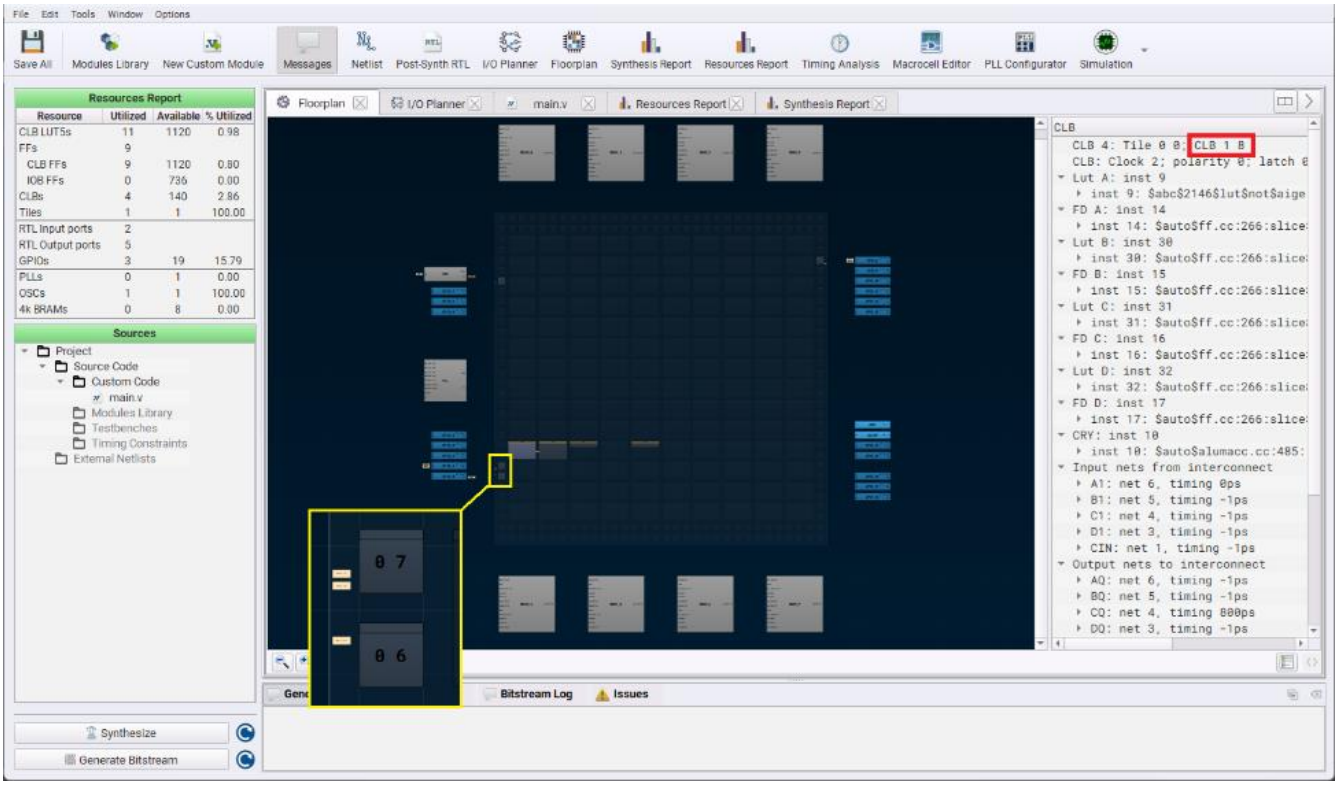

Figure 2. ForgeFPGA Workshop

3. Placement Constraints

Placement constraints can be defined to enforce a specific floorplan for the design. This approach can improve the Fmax when the constraints are properly defined according to the design architecture. There are two key concepts used in setting placement constraints:

- Placement group. A placement group is a collection of instances in the netlist.
- Placement region. A placement region is a rectangular area, specified by the left-bottom and top-right corners of the region, using matrix X/Y coordinates.

A design may contain zero or more placement groups and zero or more placement regions.

Each instance can belong to at most one placement group. A placement group can be linked to a placement region, which constrains all instances in that group to stay within the defined region.

Alternatively, a placement group might not be linked to any particular region, in which case the placement group acts as a cluster serving as a placement guide. Placement regions can also overlap with one another.

The constraint file is composed of the following commands:

- create_placement_group [name_of_the_group]
  - chip_tile_x = tilex, chip_tile_y = tile_y, chip_x = coordinate_x, chip_y = coordinate_y
  - chip_tile_x = tilex, chip_tile_y = tile_y, chip_x = coordinate_y, chip_y = coordinate_y
- create_cluster_only_placement_group [name_of_the_group]
- add_to_placement_group [name_of_the_group] [get_cells cells]
- add_to_placement_group [name_of_the_group] [get_ports ports].

The placement region is specified by the eight numbers in the **create_placement_group** command, using the IO specifications of the defined coordinates. The first four numbers give the tile coordinates X/Y and X/Y within the tile, for the left-bottom corner of the placement region and the next four numbers give coordinates of the top-right corner of the region. Note that the corner coordinates are inclusive to the region. The placement group for ports must be “one-sided”, thereby constraining the ports to only one side of the array.

Exceptions:

Due to the packing algorithms used in the PnR tool, there are a few exceptions to the general rules described above.

- The compiler packs a flop into the IOB whenever possible. If the flop and the IOB have incompatible group assignments, the flop will be re-assigned to the IOB's group.
- Similarly, the compiler packs a flop into the RAM's IOs whenever possible (typically using IOBs between the tiles). If the flop and the RAM have incompatible group assignments, the flop will be re-assigned to the RAM's group.
- The compiler packs a load flop into the CARRY chain that drives it whenever possible. If the flop and the CARRY chain have incompatible group assignments, the flop will be re-assigned to the CARRY's group.

4. Verilog Code

Shown below is the top module that implements an 8-bit counter and 3-bit shift register. The 8-bit counter increments its value on every rising edge of the clock and sets HIGH on the o_out_1 when it achieves a maximum counter value, and the shift register shifts the input signal by three clock cycles.

```
(* top *) module top (
  (* clkbuf_inhibit *) input i_clk,
  input  i_in,
  output o_out_0,
  output o_out_1,
  output o_out_0_oe,
  output o_out_1_oe,
  output o_osc_en
);
  reg [8:0] r_cnt;
  reg [2:0] r_ff;

  //Turn OSC ON
  assign o_osc_en   = 1'b1;

  always @(posedge i_clk) begin
    r_cnt <= r_cnt + 1;
  end

  //Shift register (SRL)
  always @(posedge i_clk) begin
    r_ff <= {r_ff[1:0], i_in};
  end

  assign o_out_0 = (&r_cnt);
  assign o_out_1 = r_ff[2];
  //Output enable
  assign o_out_0_oe = 1'b1;
  assign o_out_1_oe = 1'b1;


endmodule
```

5. Design Steps

1. Launch the latest version of the Go Configure™ Software Hub and select the ForgeFPGA family.

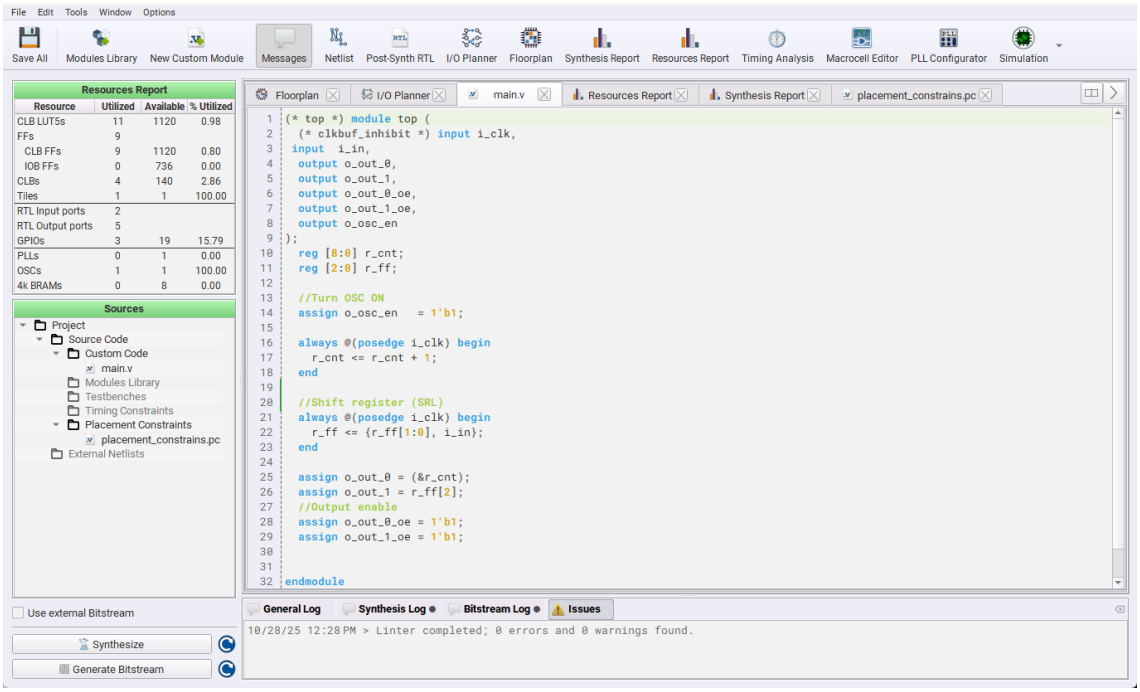

Figure 3. ForgeFPGA Workshop

2. Download and open the design example.
3. Open the **FPGA Editor** and review the Verilog code.
4. Open the **I/O Planner** tab in the FPGA editor and review the pin assignments.
5. Next select the **Synthesize** button on the lower left side of the FPGA editor.
6. Select the **Generate Bitstream** button on the lower left side of the FPGA editor.
7. Click on the **Floorplan** tab to see the CLB utilization. Hold Ctrl and scroll the mouse wheel to zoom.

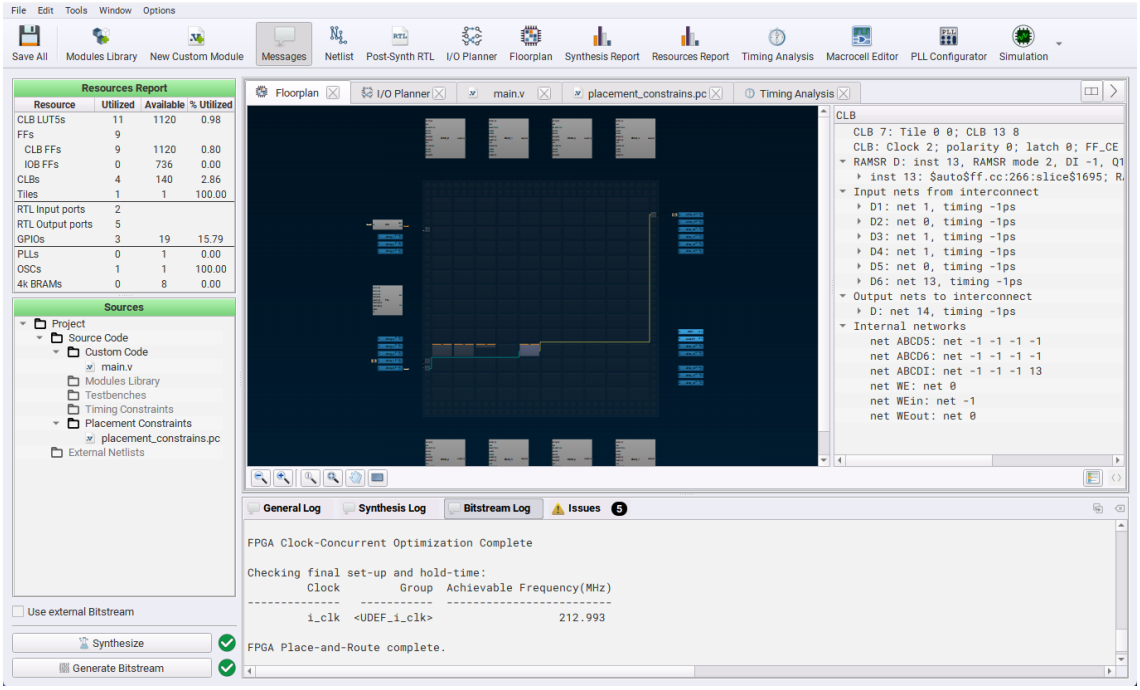

Figure 4. Floorplan Results Before Applying the Placement Constraints Commands

8. Uncomment the first set of placement constraint commands (Figure 5) to see how it affects the CLB utilization and frequency operation of the design (Figure 6).

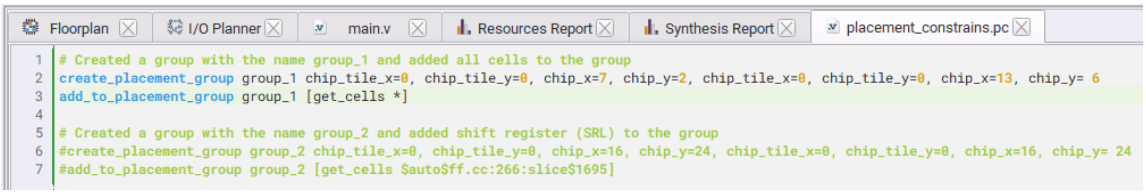

Figure 5. Uncommenting the First Set of Placement Constraints Commands

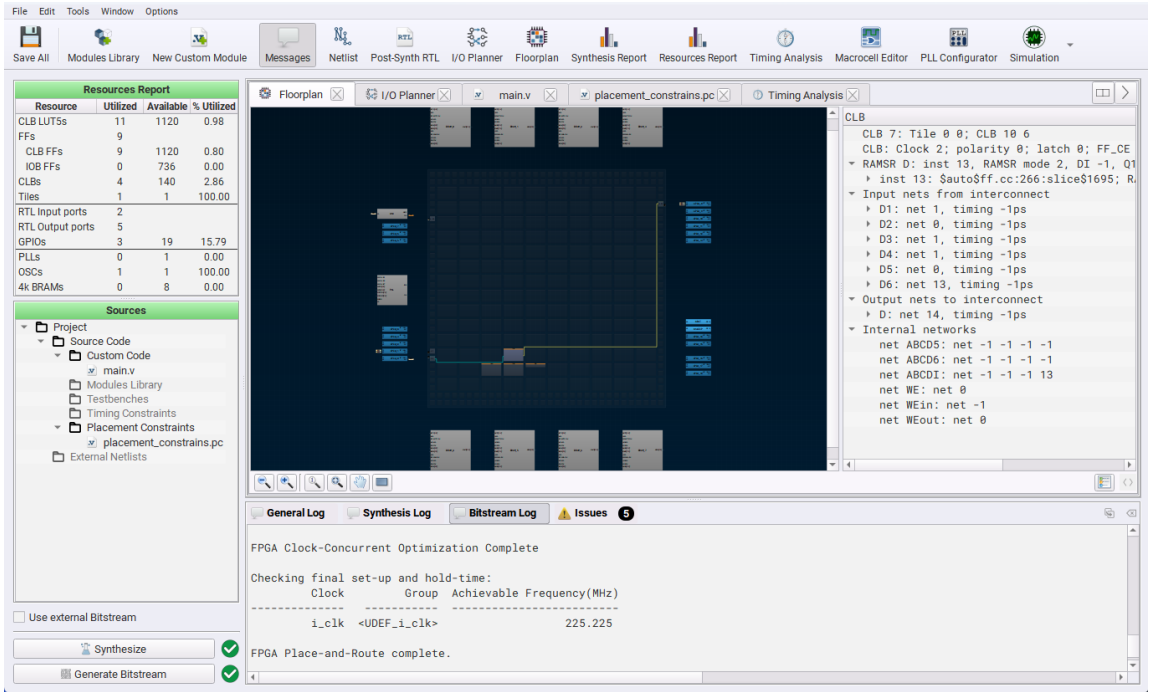

Figure 6. Floorplan Results After Applying the First Set of Commands

9. Uncomment the second set of placement constraint commands (Figure 7) to see how it affects the CLB utilization and frequency operation of the design (Figure 8).

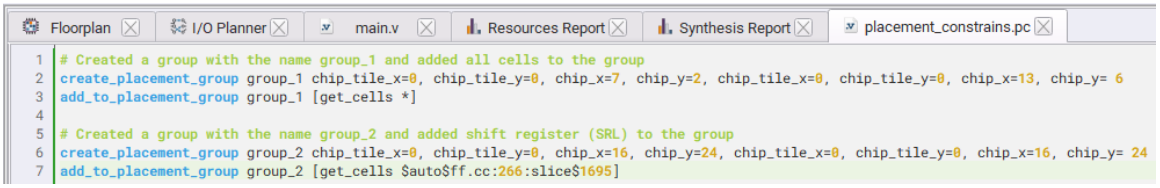

Figure 7. Uncommenting the Second Set of Placement Constraint Commands

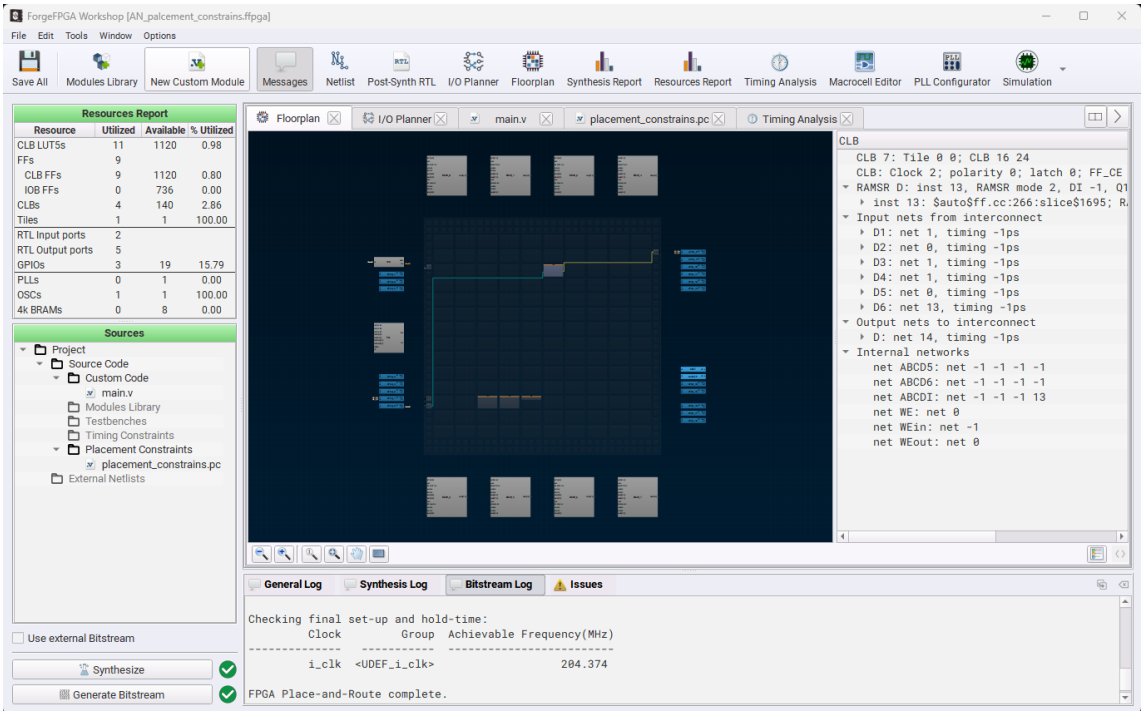

Figure 8. Floorplan Results After Applying the Second Set of Commands

10. See how the placement constraints commands affect the CLBs' placements in the **Floorplan** (Figure 9).

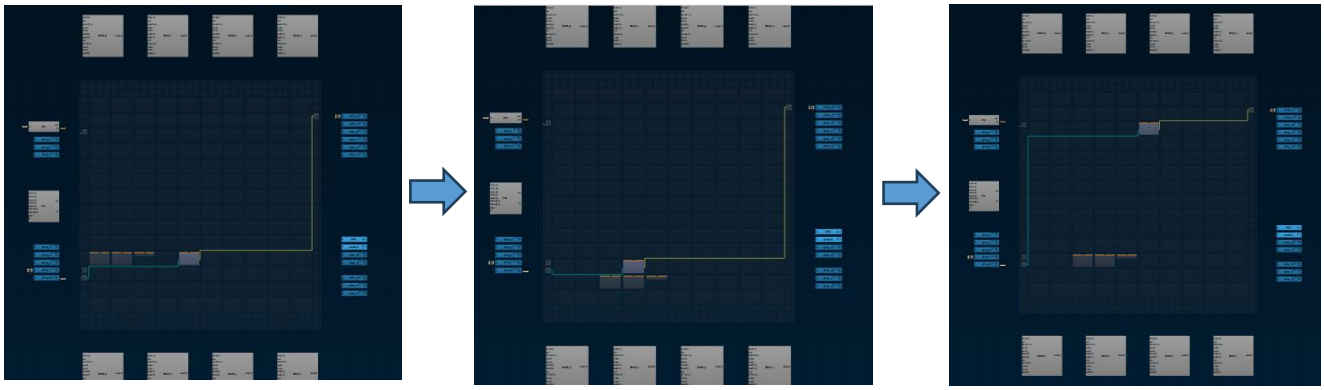

Figure 9. Showing the Results of Applying the Set of Constraints on the Floorplan (Before -> After)

6. Conclusion

This application note explains how to apply placement constraints on the ForgeFPGA design and examines the influence of the constraints on the design's operating frequency. It describes practical methods for defining and using placement groups and placement regions to guide the placement of logic elements within the device.

For more information, please contact the ForgeFPGA™ Business Support Team.

7. Terms and Definitions

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 software window for device selection
ForgeFPGA	Main FPGA project window for debug and IO programming

8. References

For related documents and software, visit: ForgeFPGA Low-density FPGAs | Renesas.

Download our free Go Configure Software Hub [1] to open the .ffpga designs files [2] and view the proposed circuit design.

[1] Go Configure Software Hub, Software Download and User Guide, Renesas Electronics

[2] AN-FG-027 Using Placement Constraints in ForgeFPGA™ Designs.ffpga, Design file, Renesas Electronics

[3] Product Selector: ForgeFPGA Low-density FPGAs, Renesas Electronics

[4] ForgeFPGA Software User Guide, Renesas Electronics

9. Revision History

Revision	Date	Description
1.00	March 5, 2026	Initial release