

FPGA Simulator – User Manual

Introduction.....	2
Project Overview.....	2
Getting Started.....	2
Prerequisites.....	2
Starting the Interface.....	3
Using the Interface.....	3
Loading Examples or Custom Files.....	4
The "Code" View.....	5
The "Simulation" View.....	6
The "Mix" View.....	7
Component Representations.....	8
Troubleshooting and Support.....	10
Feedback and Contribution.....	11

Introduction

Welcome to the FPGA Simulator Web Interface manual. This guide explains how to use our intuitive, browser-based tool for visualizing FPGA signal propagation in real-time. Users can load their own FPGA simulations (.sdf and .v files generated from tools like Impulse and ModelSim) or explore preloaded educational examples.

This tool is ideal for educators and students looking to understand FPGA behavior visually.

Project Overview

This web-based interface allows you to:

- Visualize BELs and signal routing inside an FPGA through interactive 2D diagrams.
- Perform real-time simulations reflecting accurate signal timing.
- Access preloaded Verilog applications suitable for educational demonstrations.

Getting Started

Prerequisites

Ensure you have the following installed:

- Node.js and npm
- Vite (bundled with the provided files)

Starting the Interface

Follow these steps to launch the FPGA simulator interface:

1. **Open a terminal** in the directory containing the project.
2. **Install dependencies** (only the first time):

```
npm install
```

3. **Start the server:**

```
npm run dev
```

If successful, the terminal should display:

```
Vite server running at: http://127.0.0.1:5173
```

Open this URL in your browser.

Screenshot Placeholder: Terminal showing successful launch

Using the Interface

The FPGA Simulator Web Interface has three main views accessible through tabs:

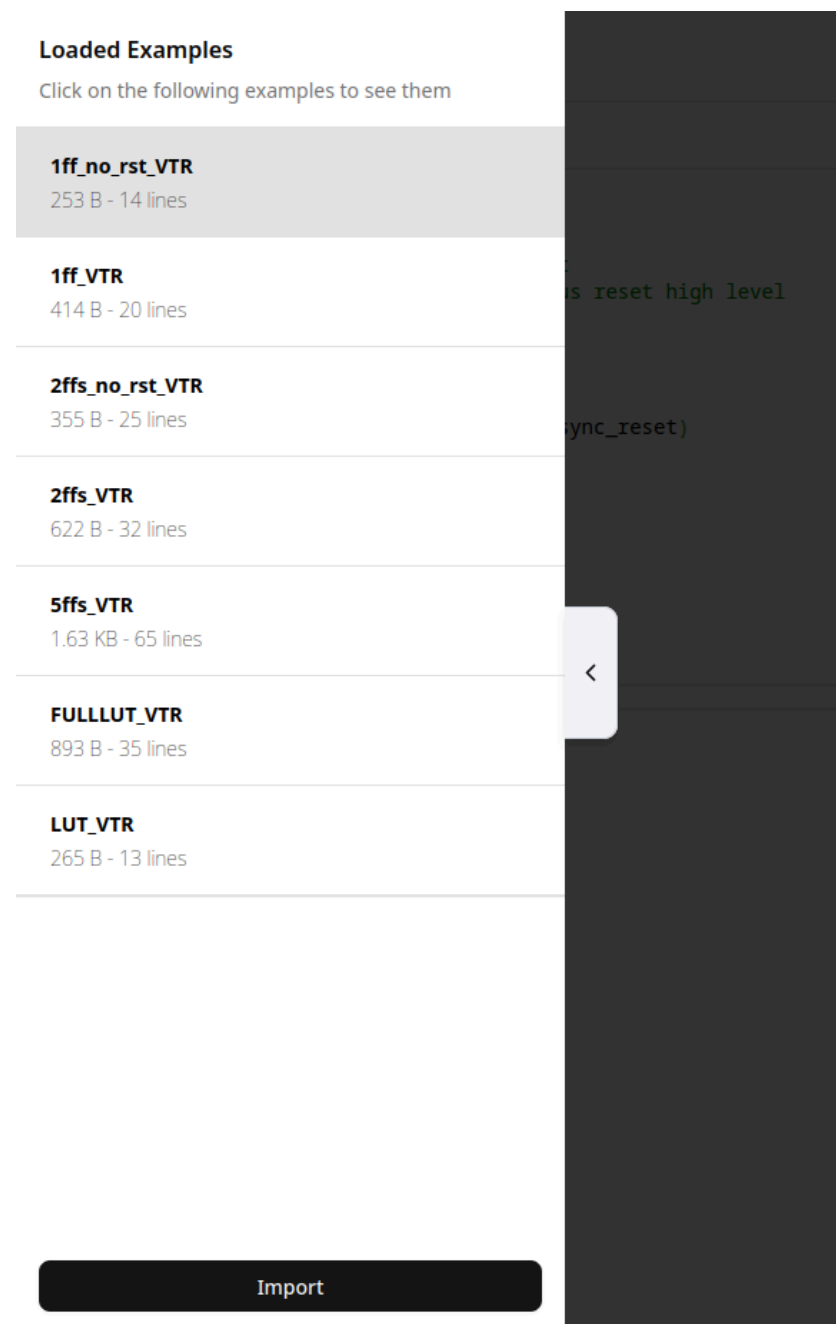
- **Code**
- **Simulation**
- **Mix**

Loading Examples or Custom Files

The collapsible sidebar on the left allows you to:

- Choose from preloaded examples.
- Load new simulation files (**.sdf** and **.v**) generated externally (Impulse, ModelSim).

Once loaded, simulations are displayed as tabs at the top, allowing seamless switching between different examples, much like browser tabs.



The "Code" View

This view (default) displays the Verilog code clearly with syntax highlighting for enhanced readability.

1ff_VTR ×	1ff_no_rst_VTR ×
1	
2	module FF1(D,clk,async_reset,Q);
3	input D; // Data input
4	input clk; // clock input
5	input async_reset; // asynchronous reset high level
6	output reg Q; // output Q
7	
8	
9	//simple flipflop example
10	always @(posedge clk or posedge async_reset)
11	begin
12	if(async_reset==1'b1)
13	Q1 <= 1'b0;
14	else
15	Q <= D;
16	end
17	
18	
19	endmodule
20	

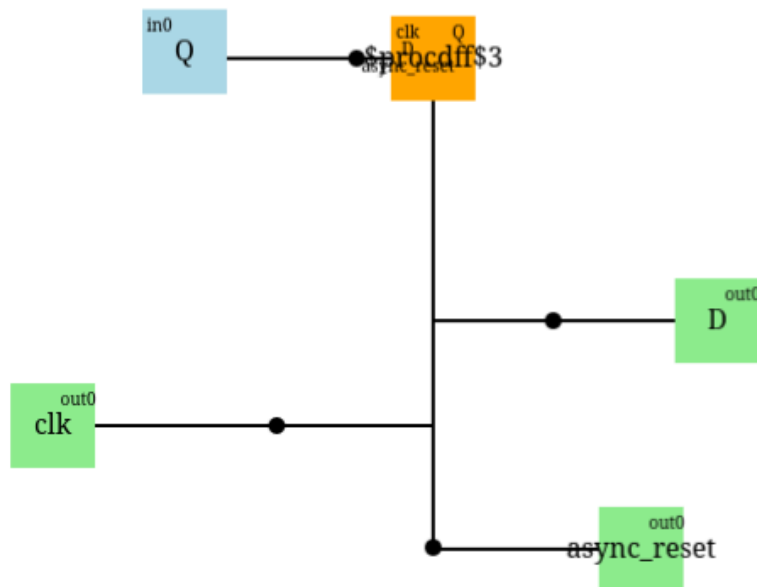
The "Simulation" View

This view visually represents FPGA components (BELs) and wires, allowing users to:

- Rearrange components by clicking and dragging.
- Pan the camera using **Shift + Scroll**.
- Zoom in/out using the mouse wheel.

Simulation Controls:

- **Play/Pause Button:** Starts or stops real-time simulation, visualizing signals propagating through wires.
- **Undo/Redo Buttons:** Revert or reapply changes to component arrangement.
- **Restore Button:** Resets the arrangement to default.

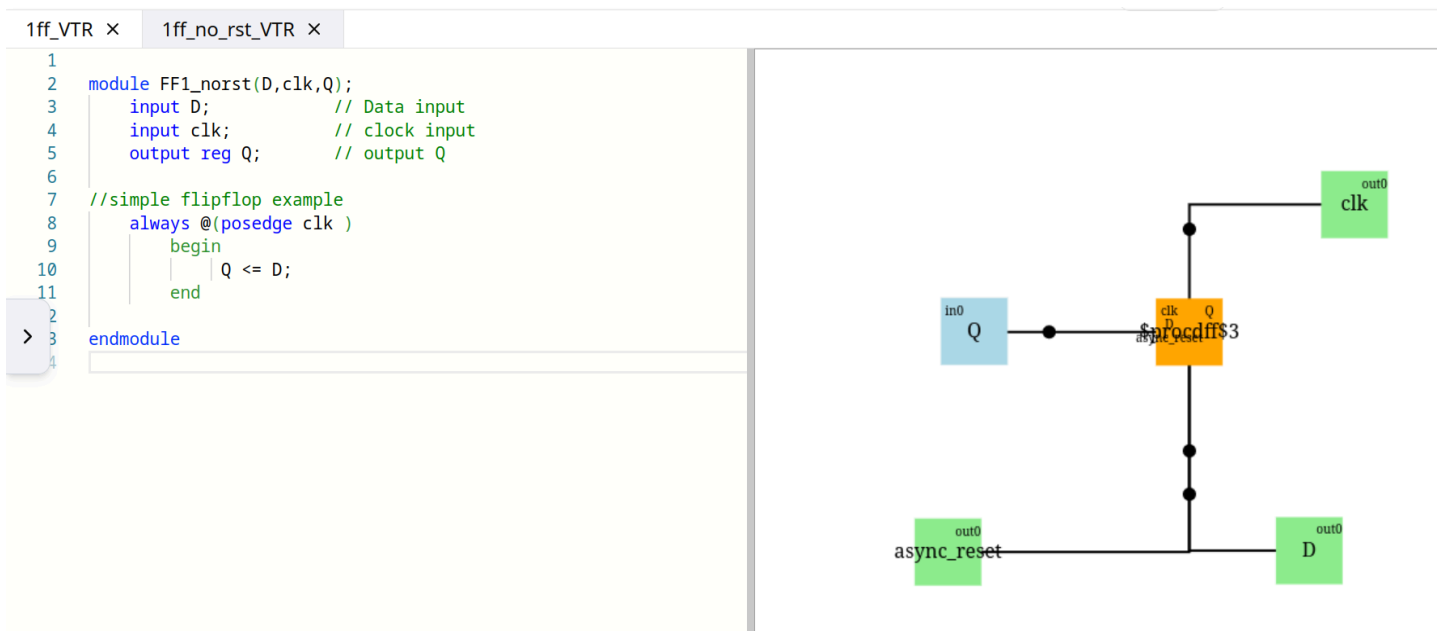


The "Mix" View

The Mix view divides the screen into two adjustable sections:

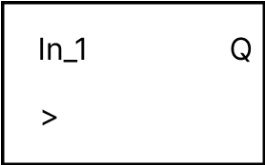
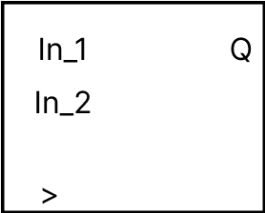
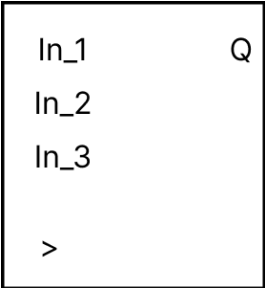
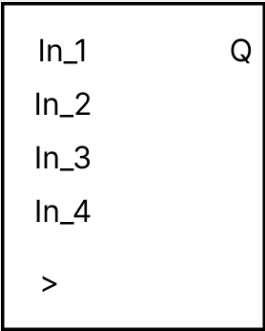

- **Left side:** Code view.
- **Right side:** Simulation view.

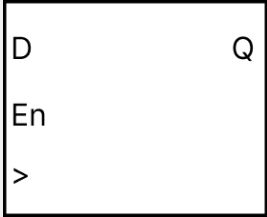






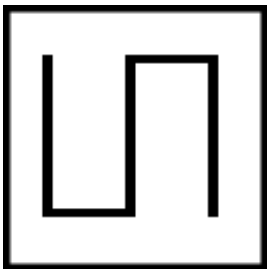
You can drag the divider to adjust the ratio between the two views according to your preference.



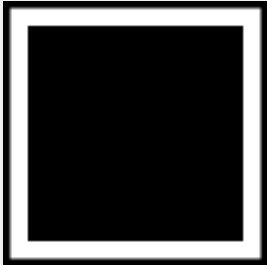
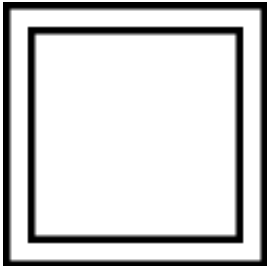


Component Representations

The FPGA components (BELs) can be represented in various forms to help illustrate different aspects of FPGA operation clearly:

Component	Representation
LUT1	
LUT2	
LUT3	
LUT4	
D-Flip Flop	

D-Flip Flop with enable	 <p>A rectangular symbol for a D-Flip Flop with enable. It has two inputs on the left: 'D' at the top and 'En' at the bottom. It has one output on the right labeled 'Q'. A small triangle symbol is located below the 'En' input.</p>
AND	 <p>A standard AND gate symbol with two inputs on the left and one output on the right.</p>
OR	 <p>A standard OR gate symbol with two inputs on the left and one output on the right.</p>
NAND	 <p>A standard NAND gate symbol, which is an AND gate with a small circle (bubble) at the output.</p>
NOR	 <p>A standard NOR gate symbol, which is an OR gate with a small circle (bubble) at the output.</p>
XOR	 <p>A standard XOR gate symbol, which is an OR gate with an additional curved line on the left side of the inputs.</p>
XNOR	 <p>A standard XNOR gate symbol, which is an XOR gate with a small circle (bubble) at the output.</p>
Clock	 <p>A square symbol containing a stylized clock waveform, consisting of a vertical line, a horizontal line, and another vertical line.</p>

LED on	
LED off	
Button on	
Button off	

Troubleshooting and Support

For any difficulties encountered:

- **Ensure file compatibility:** Verify your `.v` and `.sdf` files are correctly exported from Impulse or ModelSim.
- **Restart the web server:** Often resolves minor glitches.

If issues persist or you encounter errors, please contact your instructor or system administrator for further assistance.

Feedback and Contribution

Your feedback is invaluable for improving this tool. Please report bugs or suggest improvements directly on our [github issue](#)