100Days RTL Challenge

📘 What is the 100 Days of RTL Challenge?

The 100 **Days of RTL Challenge** is a personal commitment to **learn, build, and practice RTL design (primarily Verilog)** every day for 100 days. Inspired by the #100DaysOfCode challenge, this version is focused on developing strong foundations in **digital logic design, hardware description languages, and test benching**.

🎯 Goals of the Challenge

- ✅ Build hands-on experience with Verilog / System Verilog / VHDL
- ✅ Practice writing and simulating hardware modules
- ✅ Understand and implement combinational/sequential circuits
- ✅ Improve test benching and debugging skills
- ✅ Explore synthesis and FPGA implementation (optional)
- ✅ Document learning for self-reference and to help others

🛠️ Recommended Tools

Purpose	Tool
Coding	VS Code, Sublime Text
Simulation	Icarus Verilog, Model Sim, Vivado, GHDL
Waveform Viewer	GTK Wave
FPGA Deployment (optional)	Xilinx Vivado, Intel Quartus
Version Control	Git & GitHub
Notes/Blog	GitHub README, Notion, Obsidian, Hack MD

🧱 Suggested Folder Structure for GitHub Repository :

```
100-days-rtl-challenge/
|
├── Day01_BasicGates/
|    ├── basic_gates.v
|    ├── basic_gates_tb.v
|    └── README.md
```

```
│
├── Day02_Booleanfunction/
├── ...
│
├── Day100_FinalProject/
│   └── mini_cpu.v
│
└── README.md
```

🗂️ Master README.md Structure (Main GitHub Repository Page) :

# 💻 100 Days of RTL Challenge - Verilog Edition 🧠

Welcome to my **100-day journey into RTL design using Verilog!**

This challenge is about building confidence in designing, simulating, and understanding digital logic systems from scratch.

## 📅 Challenge Plan

- 1 Verilog module per day (or 1 concept/project broken into daily chunks)

- Daily simulations + testbenches

- GitHub commit + README update

- Weekly learning summaries

## 📚 Topics Covered (Planned)

### 🔸 Beginner Level (Days 1–20)

- Basic Gates (AND, OR, NOT, etc.)
- MUX / DEMUX
- Encoders / Decoders
- Adders / Subtractors
- Comparators
- Seven Segment Display

### 🔸 Intermediate Level (Days 21–50)

- Flip-Flops & Latches
- Registers
- Counters (up/down/BCD)
- FSMs (Moore & Mealy)
- Priority Logic

### 🔸 Advanced Level (Days 51–80)

- ALU Design
- Memory (RAM/ROM)
- Pipelining Concepts
- Hierarchical Design
- UART / SPI

### 🔸 Final Projects (Days 81–100)

- Mini-CPU

- RISC-V Components
- Traffic Light Controller
- 4-bit Calculator
- FSM-based Game Logic

## 📦 Logs

Day	Topic	Folder
1	Basic Gates	[Day01](./Day01_BasicGates)
2	Boolean function	[Day02](./Day02_Boolean function)
...	...	...
100	Final Project	[Day100_FinalProject](./Day100_FinalProject)

## 🧠 Learnings & Reflections

Every README inside each day's folder will contain:

- Description of the module
- Verilog Code for design & test bench
- Waveform screenshots
- Key learning points
- Test cases & coverage

## 🧪 How to Run Simulations

Example for Icarus Verilog:

```bash
iverilog -o design_tb design.v design_tb.v
vvp design_tb
gtkwave dump.vcd
```

Tool used : Xilinx Vivado version 2023.2

Tool URL : https://www.xilinx.com/support/download.html

💬 **Sharing & Community**

- Connect on LinkedIn: www.linkedin.com/in/nandyala-harika-0a6a212b9
- Hashtags: #100DaysOfRTL #Verilog #DigitalDesign #FPGA