Creating and Submitting a Tiny Tapeout Design with HDL

Anthony Rizzuto, James Doyle, Esteban Chacon, Dominic Cosentino, Adrian Skudrinja, Sean Whilden
Arizzut1@asu.edu

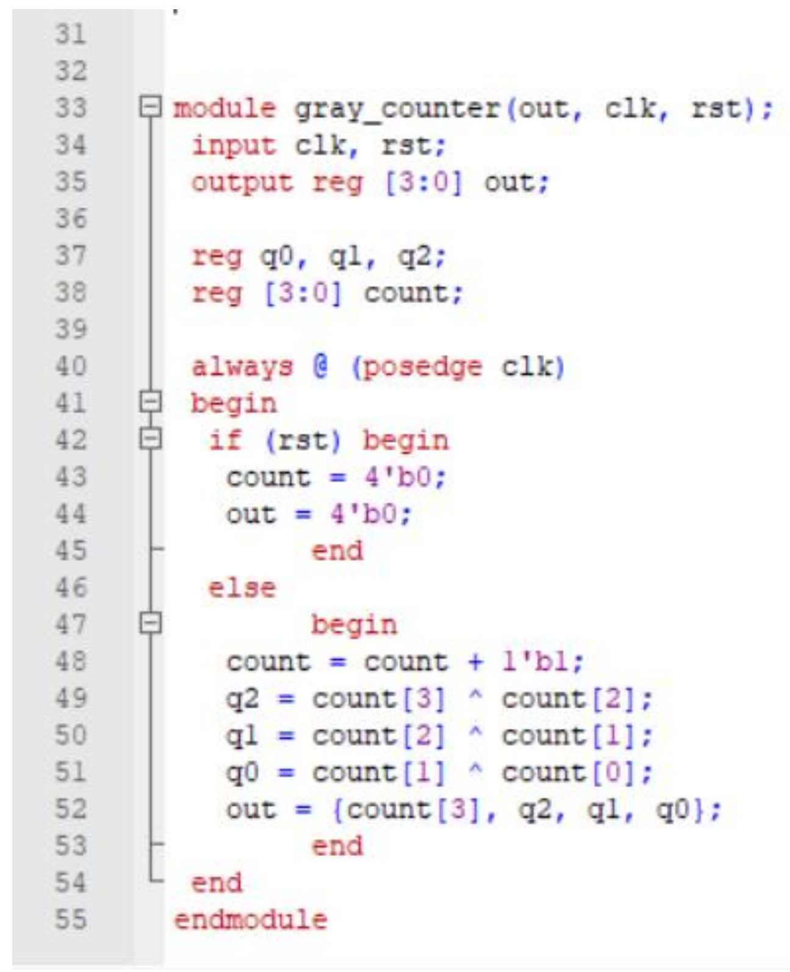
Introduction - Tiny Tapeout is an educational initiative aimed at democratizing the process of creating digital designs and manufacturing them on real chips. This project leverages integrated circuit design software developed by Tapeout, enabling users to either utilize Tapeout's software or upload Hardware Description Language (HDL) such as Verilog to a GitHub repository. Once uploaded, the HDL is processed to generate a circuit design that can be printed onto a real chip. In this poster presentation, we showcase a Tiny Tapeout design implemented using HDL, demonstrating the feasibility and accessibility of the Tiny Tapeout platform for educational and prototyping purposes. We discuss the design process of a 4-bit gray code counter, highlighting the potential of Tiny Tapeout to inspire and empower individuals in the field of digital design and chip manufacturing.

Starting a Design with HDL

Initiating a design with Hardware Design Language (HDL) can be done using various software choices. There are open-source options that provide a cost-effective solution to users seeking accessibility in their design process such as Icarus Verilog. There are paid software's, such as ModelSim, Which offers student editions and provides more advanced features for larger scale projects. Either software or any other can be used to complete what will be shown for this Tiny Tapeout. In this example, the student edition of ModelSim is used.

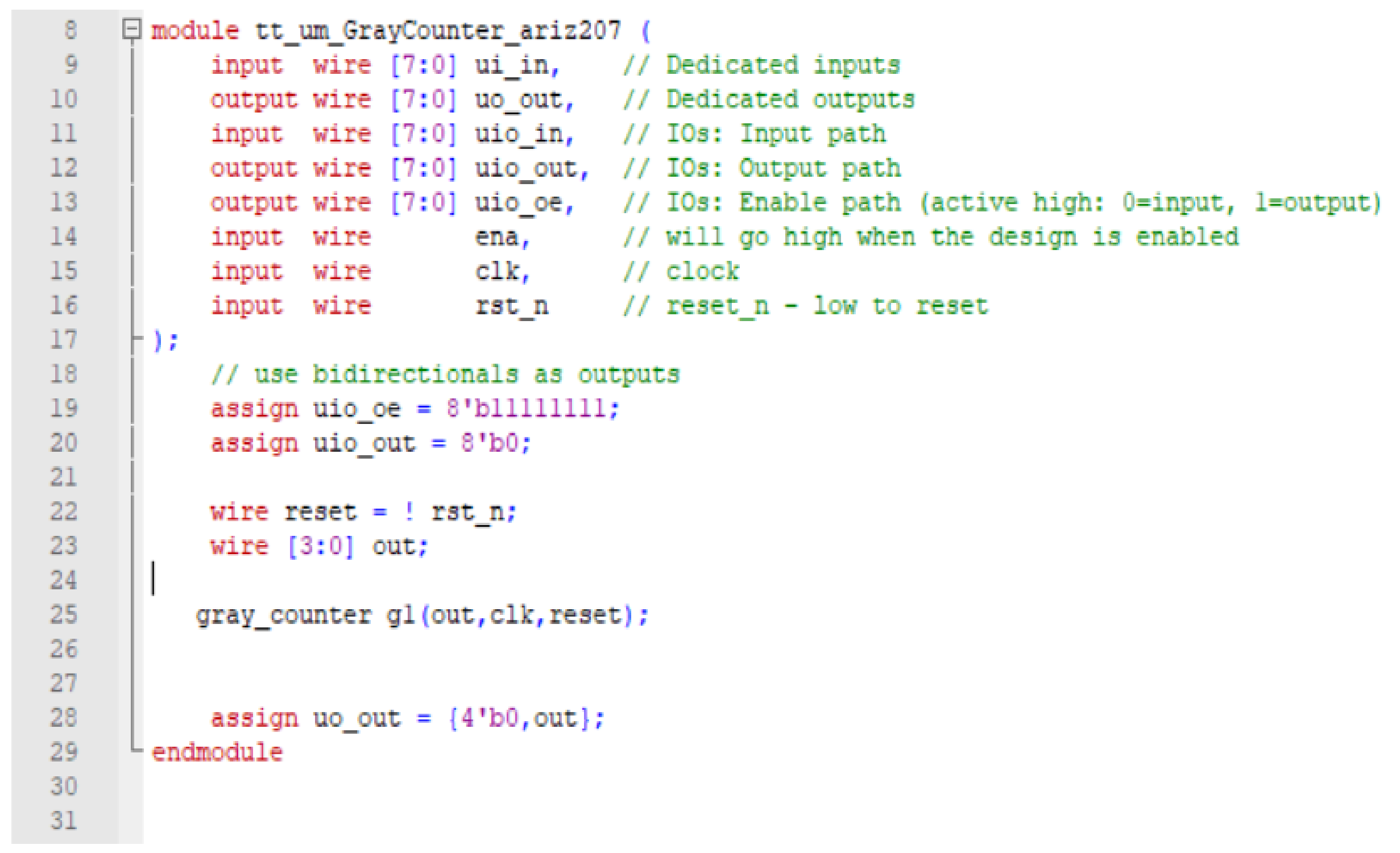
Compiling and Formatting HDL Code

To adhere to the requirements for compatibility with the Tiny Tapeout submission process, the code must absolutely have the specific module input and output ports. These ports work as the interface between your code and the chip, ensuring seamless integration and functionality with the submission framework. The unformatted Gray Code is shown in Fig.1. The code as is would not work with the Tiny Tapeout submission process, as the module input and output ports have not been properly formatted to match the framework.



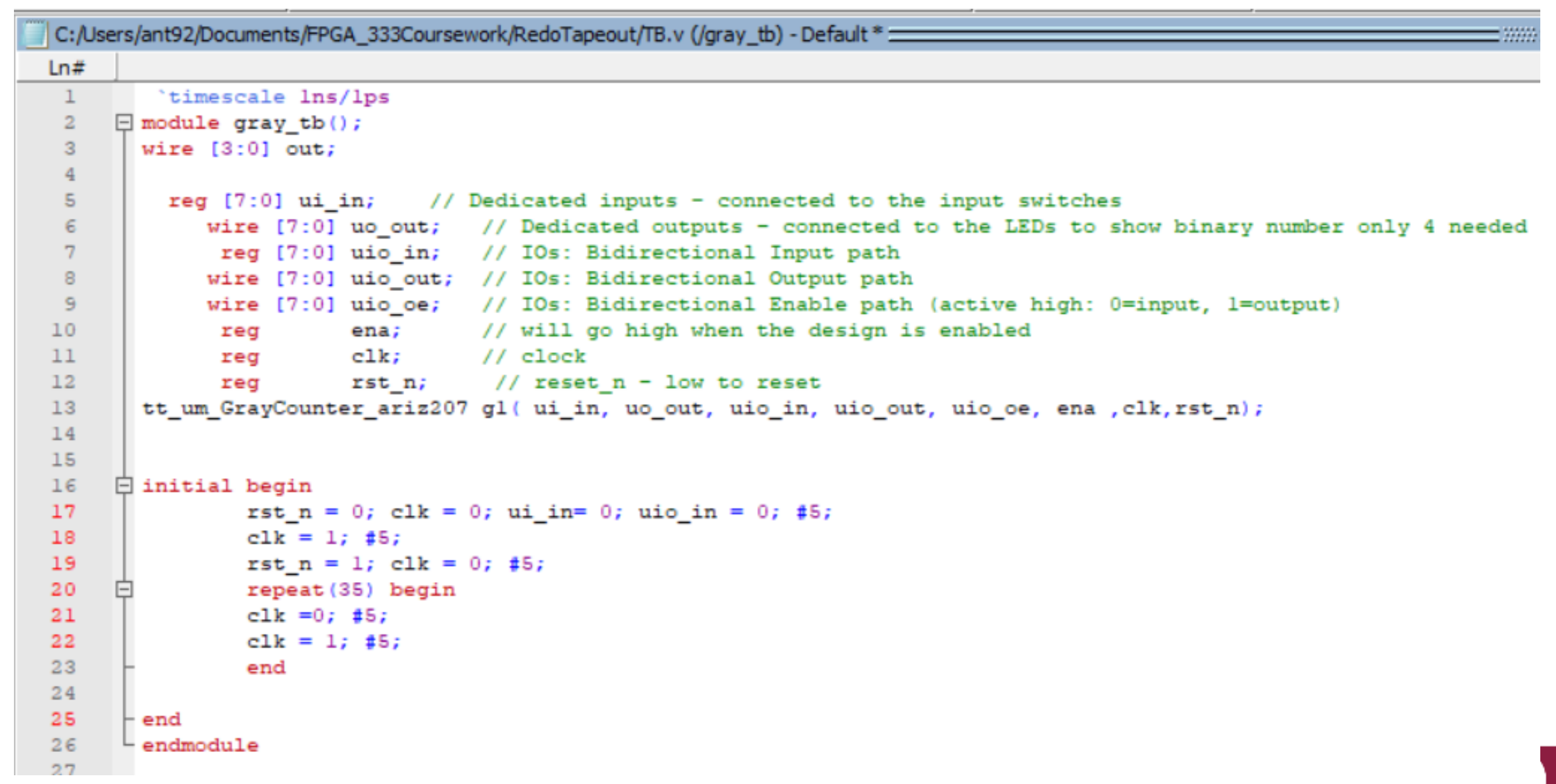
```
module gray_counter(out, clk, rst);
 input clk, rst;
 output reg [3:0] out;

 reg q0, q1, q2;
 reg [3:0] count;

 always @ (posedge clk)
 begin
  if (rst) begin
   count = 4'b0;
   out = 4'b0;
        end
  else
        begin
   count = count + 1'b1;
   q2 = count[3] ^ count[2];
   q1 = count[2] ^ count[1];
   q0 = count[1] ^ count[0];
   out = {count[3], q2, q1, q0};
        end
 end
endmodule
```

Fig. 1 – Gray Code Counter unformatted Code



```
module tt_um_GrayCounter_ariz207 (
    input  wire [7:0] ui_in,    // Dedicated inputs
    output wire [7:0] uo_out,   // Dedicated outputs
    input  wire [7:0] uio_in,   // IOs: Input path
    output wire [7:0] uio_out,  // IOs: Output path
    output wire [7:0] uio_oe,   // IOs: Enable path (active high: 0=input, 1=output)
    input  wire       ena,      // will go high when the design is enabled
    input  wire       clk,      // clock
    input  wire       rst_n     // reset_n - low to reset
);
    // use bidirectionals as outputs
    assign uio_oe = 8'b11111111;
    assign uio_out = 8'b0;

    wire reset = ! rst_n;
    wire [3:0] out;

   gray_counter g1(out,clk,reset);


    assign uo_out = {4'b0,out};
endmodule
```

Fig. 2 – Gray Code Counter Formatted Code



C:/Users/ant92/Documents/FPGA_333Coursework/RedoTapeout/TB.v (/gray_tb) - Default *

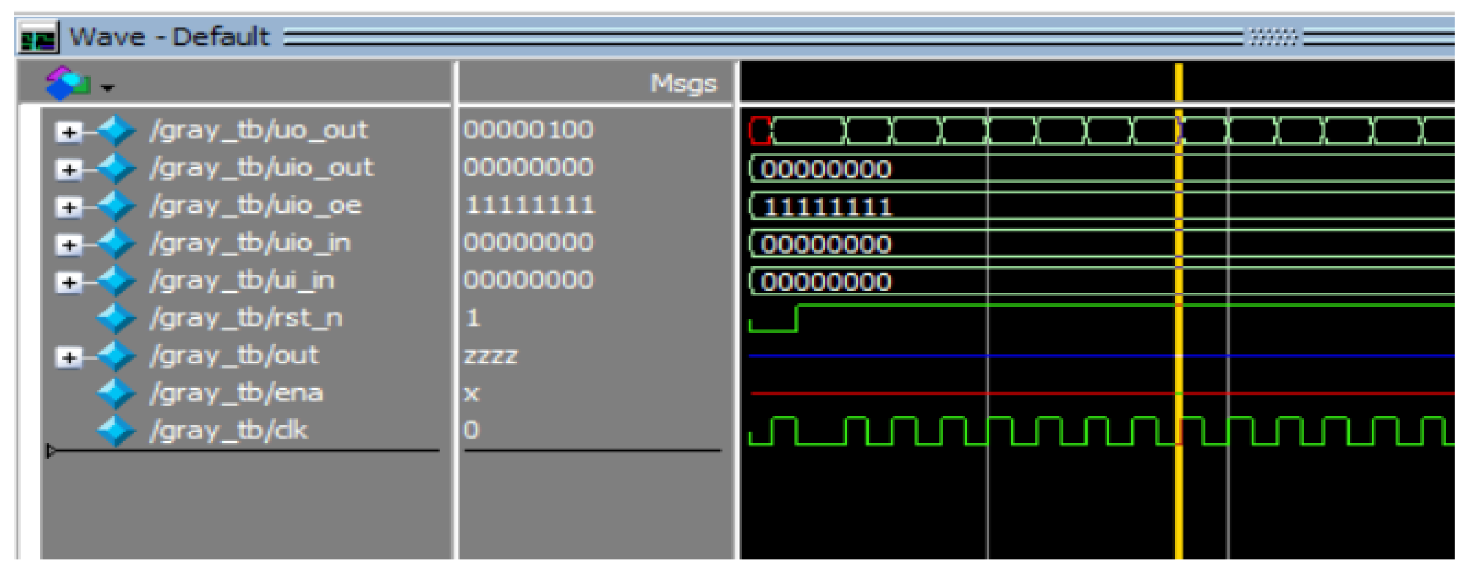
```
 `timescale 1ns/1ps
module gray_tb();
wire [3:0] out;

  reg [7:0] ui_in;    // Dedicated inputs - connected to the input switches
     wire [7:0] uo_out;   // Dedicated outputs - connected to the LEDs to show binary number only 4 needed
      reg [7:0] uio_in;   // IOs: Bidirectional Input path
     wire [7:0] uio_out;  // IOs: Bidirectional Output path
     wire [7:0] uio_oe;   // IOs: Bidirectional Enable path (active high: 0=input, 1=output)
      reg       ena;      // will go high when the design is enabled
      reg       clk;      // clock
      reg       rst_n;     // reset_n - low to reset
tt_um_GrayCounter_ariz207 g1( ui_in, uo_out, uio_in, uio_out, uio_oe, ena ,clk,rst_n);


initial begin
        rst_n = 0; clk = 0; ui_in= 0; uio_in = 0; #5;
        clk = 1; #5;
        rst_n = 1; clk = 0; #5;
        repeat(35) begin
        clk =0; #5;
        clk = 1; #5;
        end

end
endmodule
```

Fig. 3 – Testbench Code with Clock Signal

Shown in Fig.2 is the formatted code to include the right ports. In this is the parent module, which instantiates the previous gray counter in Fig. 1 but is now correctly adjusted for the submission.

Testing and Validation

A testbench is created to validate the code and ensure proper behavior of the module. The waveforms in Fig. 4 are the results from ModelSim as well as additional verification from another open-source software, Qspice. Both demonstrate that the counter is functional using the required formatting for submission and can now be brought to GitHub for the next steps.



```
/gray_tb/uo_out    00000100
/gray_tb/uio_out   00000000
/gray_tb/uio_oe    11111111
/gray_tb/uio_in    00000000
/gray_tb/ui_in     00000000
/gray_tb/rst_n     1
/gray_tb/out       zzzz
/gray_tb/ena       x
/gray_tb/clk       0
```

Fig. 4 – Testbench Waveforms

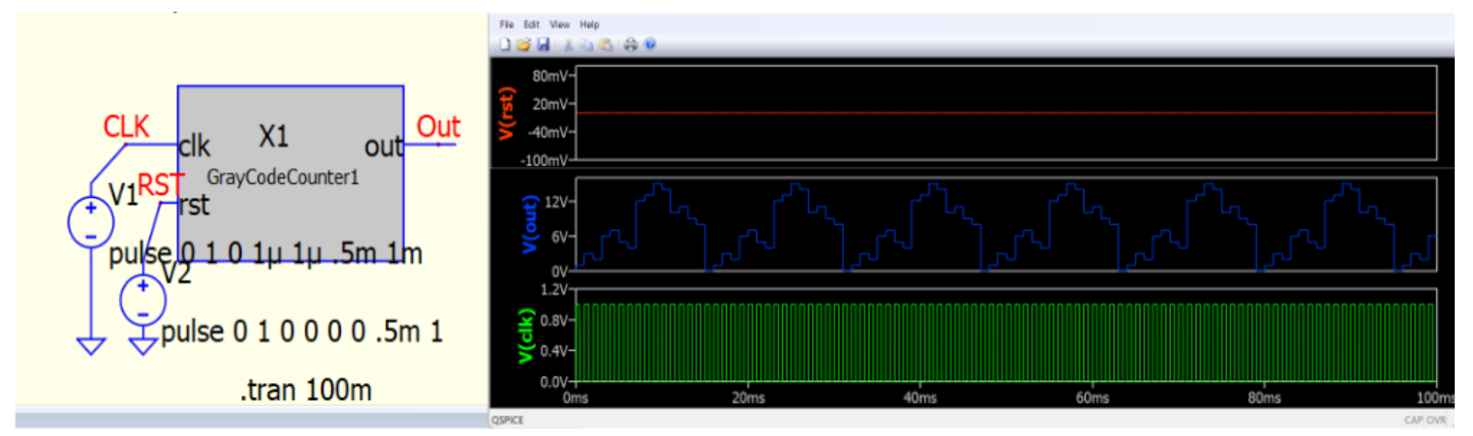


Fig. 5 – Outside Verification Using Qspice

Ira A. Fulton Schools of Engineering
Arizona State University

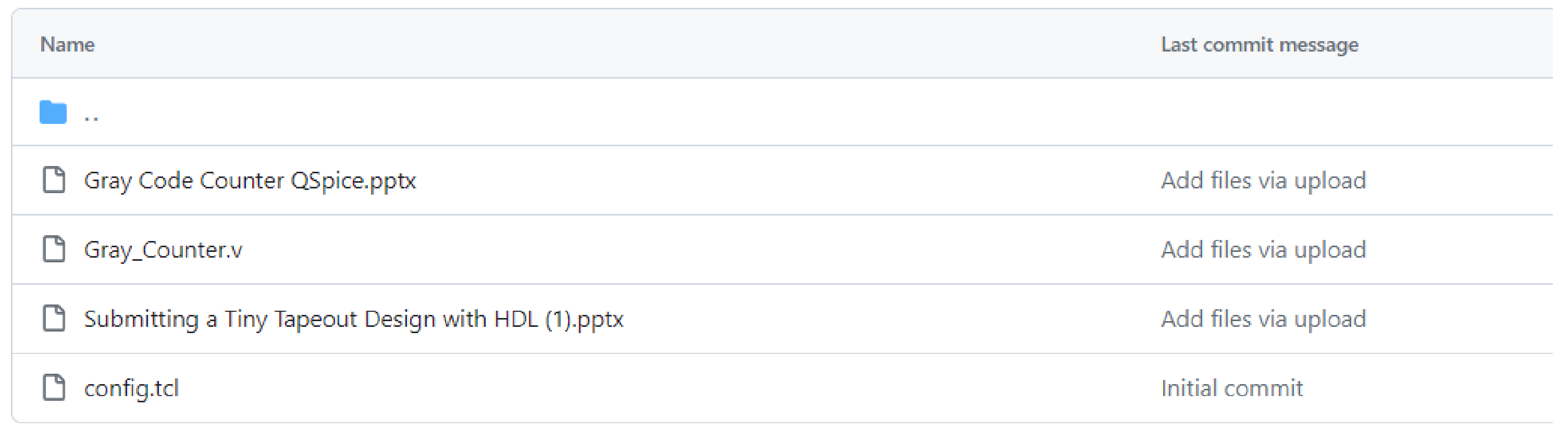
Creating and Submitting a Tiny Tapeout Design with HDL

Bringing the Code to GitHub

Find the GitHub repository for the submission template on Tiny Tapeout’s website. Find the “Use this template” button to create a new repository and name it relevant to the project. Tiny Tapeout requires you to use this submission template, as it contains certain code to adapt the template to your project. You begin by enabling GitHub Actions. To enable GitHub Actions, navigate to the repository's settings and locate the "Pages" tab. Underneath the "Source" section within the "Pages" tab, activate GitHub Actions

Uploading the Source Files

In the "src" directory located on the main page of the repository, proceed with the upload of Verilog files, including associated testbenches. This action facilitates the organization and accessibility of essential design and verification components within the designated project structure.



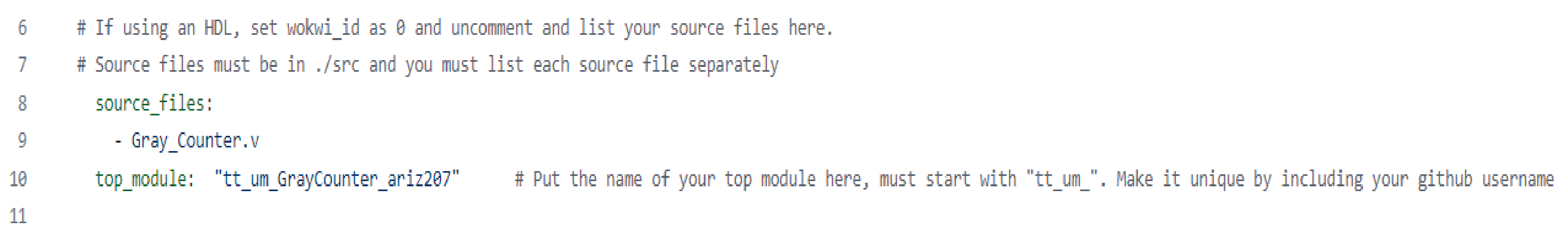
Name	Last commit message
..	
Gray Code Counter QSpice.pptx	Add files via upload
Gray_Counter.v	Add files via upload
Submitting a Tiny Tapeout Design with HDL (1).pptx	Add files via upload
config.tcl	Initial commit

Fig. 7 – Source Directory Updated

Fig.7 Shows what the updated source directory should look like after uploading all the relevant files to the project.

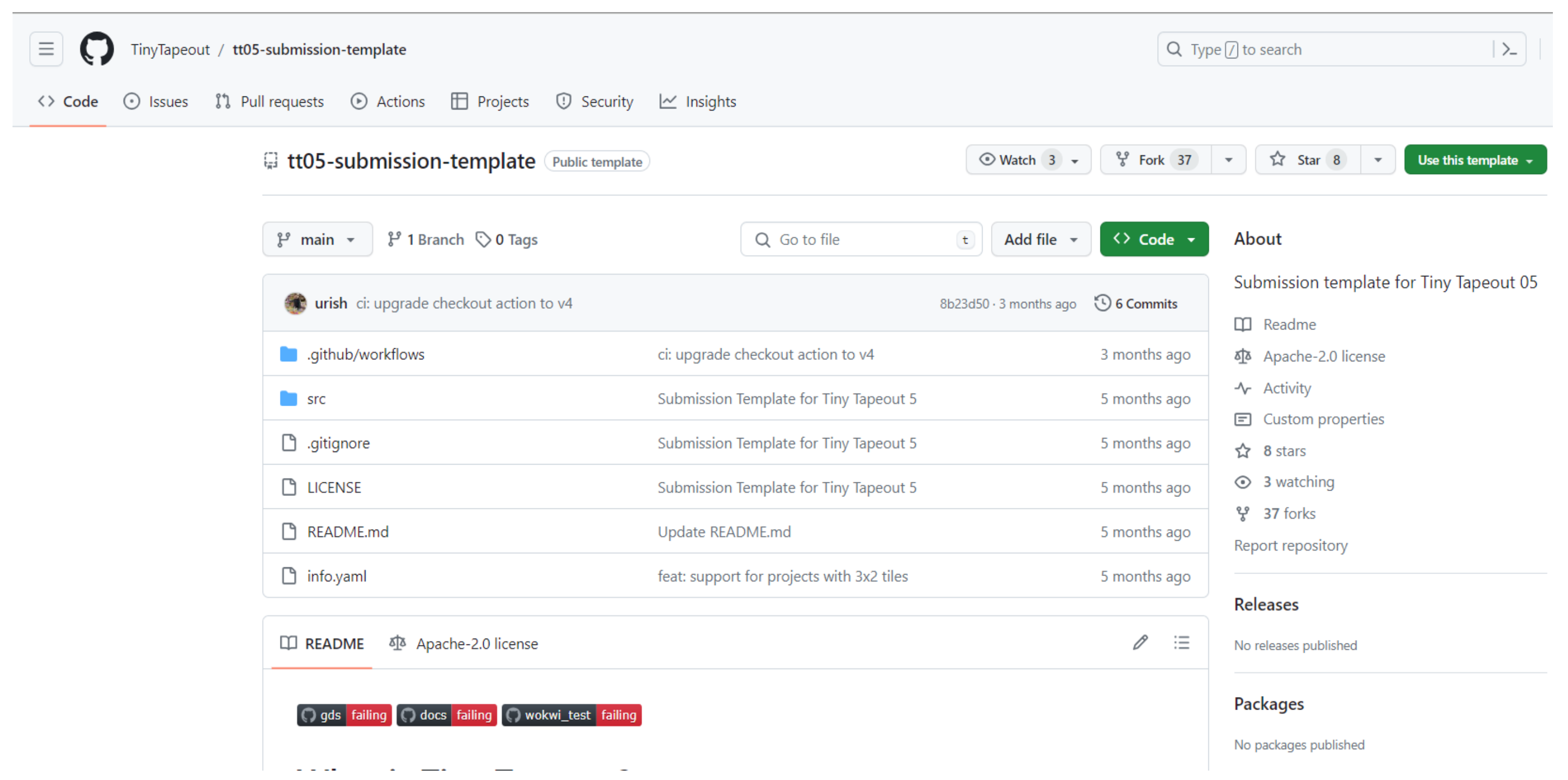
Updating info.yaml

Access the "info.yaml" file and proceed with updating it accordingly. Enumerate the source files stored in the "src" directory, including the identification of the top module used in the Verilog file. Additionally, revise the documentation segments to incorporate pertinent information such as authorship, project title, programming language utilized, design description, testing methodologies employed, and specification of input and output interfaces. Commit the changes as seen in Fig. 8 below.



```
# If using an HDL, set wokwi_id as 0 and uncomment and list your source files here.
# Source files must be in ./src and you must list each source file separately
  source_files:
    - Gray_Counter.v
  top_module:  "tt_um_GrayCounter_ariz207"      # Put the name of your top module here, must start with "tt_um_". Make it unique by including your github username

```

Fig. 8 Updated info.yaml



TinyTapeout / tt05-submission-template

tt05-submission-template

urish ci: upgrade checkout action to v4 — 8b23d50 · 3 months ago — 6 Commits

.github/workflows	ci: upgrade checkout action to v4	3 months ago
src	Submission Template for Tiny Tapeout 5	5 months ago
.gitignore	Submission Template for Tiny Tapeout 5	5 months ago
LICENSE	Submission Template for Tiny Tapeout 5	5 months ago
README.md	Update README.md	5 months ago
info.yaml	feat: support for projects with 3x2 tiles	5 months ago

README · Apache-2.0 license

gds failing · docs failing · wokwi_test failing

About

Submission template for Tiny Tapeout 05

Readme · Apache-2.0 license · Activity · Custom properties · 8 stars · 3 watching · 37 forks · Report repository

Releases: No releases published

Packages: No packages published

Fig. 6 – GitHub Repository
Source: https://github.com/TinyTapeout/tt05-submission-template

Actions Results and Passes

Ensure that the "src" directory's "info.yaml" files have been correctly integrated into the GitHub repository, verifying this under the "Actions" tab. The status indicators for the "gds," "docs," and "test" categories will provide confirmation of successful compilation and passage of all actions. Any encountered errors will manifest as a red "failing" box corresponding to the affected category.

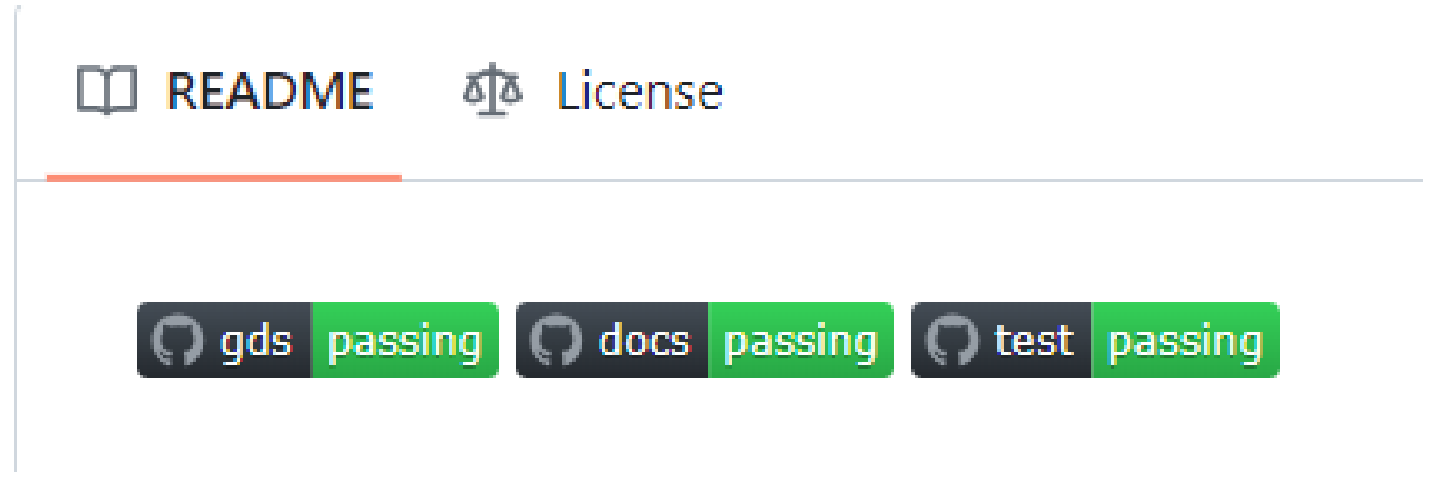
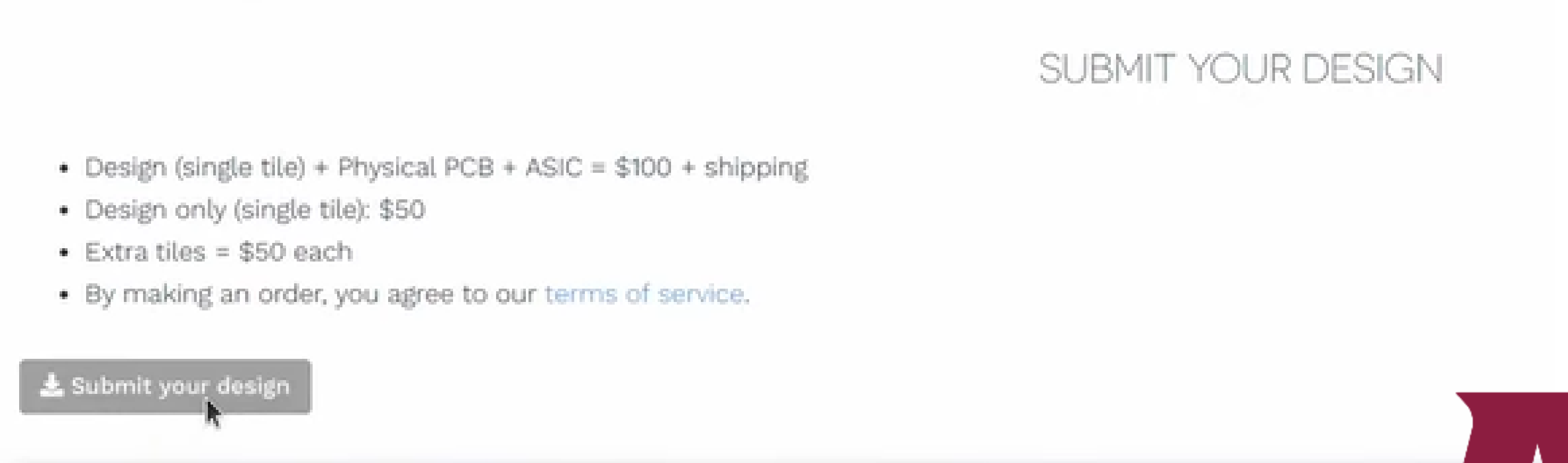


Fig. 9 Passing indicators on GitHub mainpage



SUBMIT YOUR DESIGN

- Design (single tile) + Physical PCB + ASIC = $100 + shipping
- Design only (single tile): $50
- Extra tiles = $50 each
- By making an order, you agree to our terms of service.

Fig. 10 Submission link Under Tapeout Website

Submitting GitHub Repository to Tapeout

After confirming the proper functioning of the GitHub repository actions, the subsequent step accesses the Tiny Tapeout website at the following URL: https://tinytapeout.com/. Navigate to the submission section by selecting the submission button located at the bottom of the initial page. Proceed to link your GitHub account and submit the URL of the repository as instructed.

Physical Demo and Testing

We expect to get our first submission of the Tiny Tapeout chip mid April. When this arrives, we will test and validate it at ASU Campus using a using a voltage-controlled oscillator built with open-source PCB design software

Fig. 11 Testing area at ASU Campus

References and Important links

Tiny Tapeout Working with HDL

•https://tinytapeout.com/hdl/

Matt Venn Tapeout submission video

•https://www.youtube.com/watch?v=m62HLt4BjeA

Tapeout 5 Github Submission Template

•https://github.com/ariz207/tt05_GrayCounter

•Tapeout 5 submission example

•https://github.com/TinyTapeout/tt05-verilog-demo

Icarus Verilog Download Tutorial

•https://www.youtube.com/watch?v=3Xm6fgKAO94

Actions Failing Submission Help

•https://www.youtube.com/watch?v=m62HLt4BjeA

•Timestamp for help 04:07

Ira A. Fulton Schools of Engineering
Arizona State University