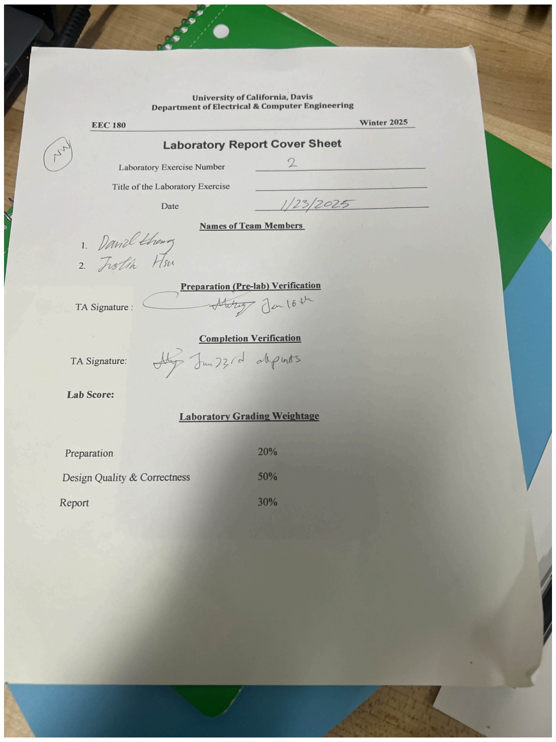
EEC 180 Lab 2
Justin Hsu and David Zhang
1/23/25

Check off



University of California, Davis
Department of Electrical & Computer Engineering

EEC 180 Winter 2025

Laboratory Report Cover Sheet

Laboratory Exercise Number 2
Title of the Laboratory Exercise
Date 1/23/2025

Names of Team Members

1. David Zhang
2. Justin Hsu

Preparation (Pre-lab) Verification

TA Signature : [illegible] Jan 16th

Completion Verification

TA Signature: [illegible] Jan 23rd all parts

Lab Score:

Laboratory Grading Weightage

Preparation	20%
Design Quality & Correctness	50%
Report	30%

How our design works:

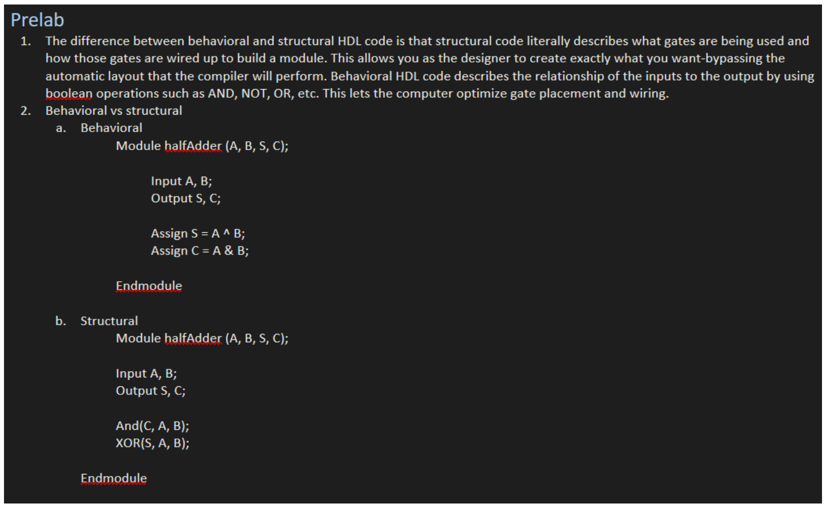
For part 1, we manually instanced each full adder for a total of 8 full adders in order to perform 8 bit binary addition. For part 3, rather than manually instance a k bit adder, we instead used a generate loop where we could instead specify how many to generate rather than manually instance it instead. Using the generate loop was much less tedious, infinitely more scalable, and generally made the code less cluttered and more cohesive.

For part 2, we simply wired all of the full adders and gates together using a lot of wires. Since it was supposed to be done structurally, we just followed the diagram provided on the lab handout.

For the testbenches we wrote self checking testbenches where an error would occur when the simulated result did not match with the expected result. This made debugging much easier as we did not have to go back and manually match the outputs one by one.

Statement of contribution:
We worked backwards starting from part 3. Work for part 3 was evenly distributed, both labmates worked to debug the synthesis and simulation. Justin wrote the structural code for part 2, David wrote the decoder for display. Part 1 was done by Justin. Write up work was split evenly between both lab mates.

Prelab



Prelab

1. The difference between behavioral and structural HDL code is that structural code literally describes what gates are being used and how those gates are wired up to build a module. This allows you as the designer to create exactly what you want-bypassing the automatic layout that the compiler will perform. Behavioral HDL code describes the relationship of the inputs to the output by using boolean operations such as AND, NOT, OR, etc. This lets the computer optimize gate placement and wiring.
2. Behavioral vs structural
   a. Behavioral

```
Module halfAdder (A, B, S, C);

        Input A, B;
        Output S, C;

        Assign S = A ^ B;
        Assign C = A & B;

Endmodule
```

   b. Structural

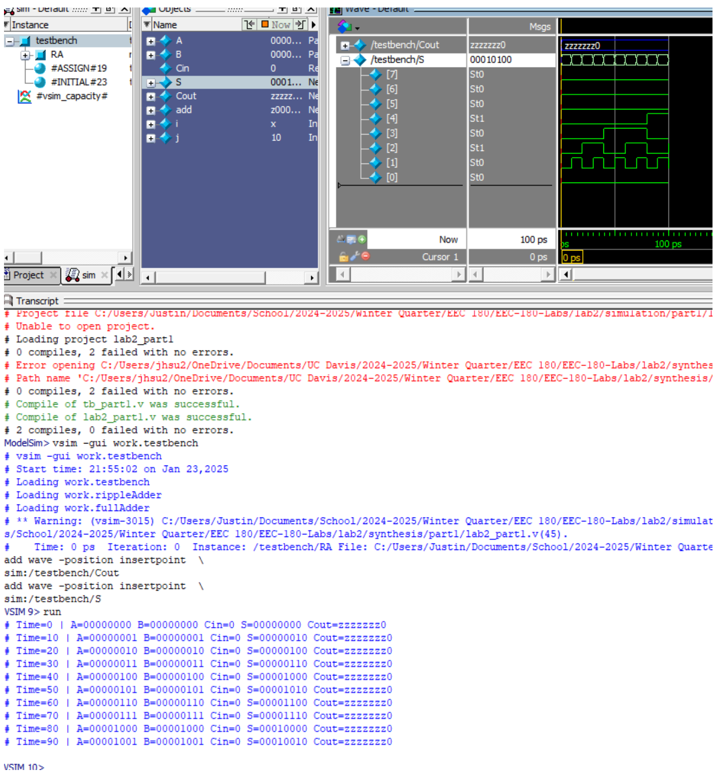
```
Module halfAdder (A, B, S, C);

Input A, B;
Output S, C;

And(C, A, B);
XOR(S, A, B);

Endmodule
```

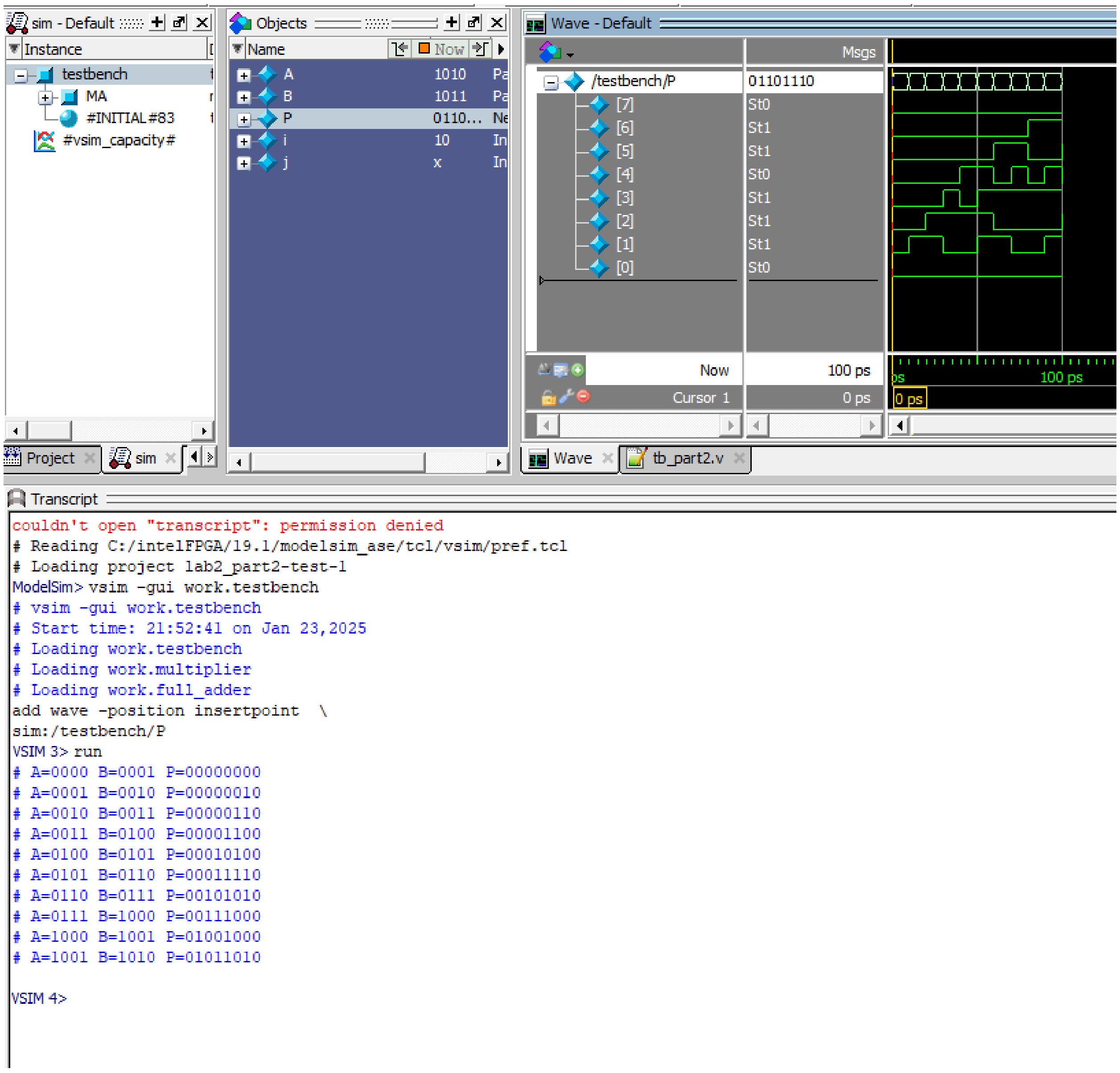
Part 1



```
# Project file C:/Users/Justin/Documents/School/2024-2025/Winter Quarter/EEC 180/EEC-180-Labs/lab2/simulation/part1/l
# Unable to open project.
# Loading project lab2_part1
# 0 compiles, 2 failed with no errors.
# Error opening C:/Users/jhsu2/OneDrive/Documents/UC Davis/2024-2025/Winter Quarter/EEC 180/EEC-180-Labs/lab2/synthes
# Path name 'C:/Users/jhsu2/OneDrive/Documents/UC Davis/2024-2025/Winter Quarter/EEC 180/EEC-180-Labs/lab2/synthesis/
# 0 compiles, 2 failed with no errors.
# Compile of tb_part1.v was successful.
# Compile of lab2_part1.v was successful.
# 2 compiles, 0 failed with no errors.
ModelSim> vsim -gui work.testbench
# vsim -gui work.testbench
# Start time: 21:55:02 on Jan 23,2025
# Loading work.testbench
# Loading work.rippleAdder
# Loading work.fullAdder
# ** Warning: (vsim-3015) C:/Users/Justin/Documents/School/2024-2025/Winter Quarter/EEC 180/EEC-180-Labs/lab2/simulat
s/School/2024-2025/Winter Quarter/EEC 180/EEC-180-Labs/lab2/synthesis/part1/lab2_part1.v(45).
#    Time: 0 ps  Iteration: 0  Instance: /testbench/RA File: C:/Users/Justin/Documents/School/2024-2025/Winter Quarte
add wave -position insertpoint  \
sim:/testbench/Cout
add wave -position insertpoint  \
sim:/testbench/S
VSIM 9> run
# Time=0 | A=00000000 B=00000000 Cin=0 S=00000000 Cout=zzzzzzz0
# Time=10 | A=00000001 B=00000001 Cin=0 S=00000010 Cout=zzzzzzz0
# Time=20 | A=00000010 B=00000010 Cin=0 S=00000100 Cout=zzzzzzz0
# Time=30 | A=00000011 B=00000011 Cin=0 S=00000110 Cout=zzzzzzz0
# Time=40 | A=00000100 B=00000100 Cin=0 S=00001000 Cout=zzzzzzz0
# Time=50 | A=00000101 B=00000101 Cin=0 S=00001010 Cout=zzzzzzz0
# Time=60 | A=00000110 B=00000110 Cin=0 S=00001100 Cout=zzzzzzz0
# Time=70 | A=00000111 B=00000111 Cin=0 S=00001110 Cout=zzzzzzz0
# Time=80 | A=00001000 B=00001000 Cin=0 S=00010000 Cout=zzzzzzz0
# Time=90 | A=00001001 B=00001001 Cin=0 S=00010010 Cout=zzzzzzz0

VSIM 10>
```

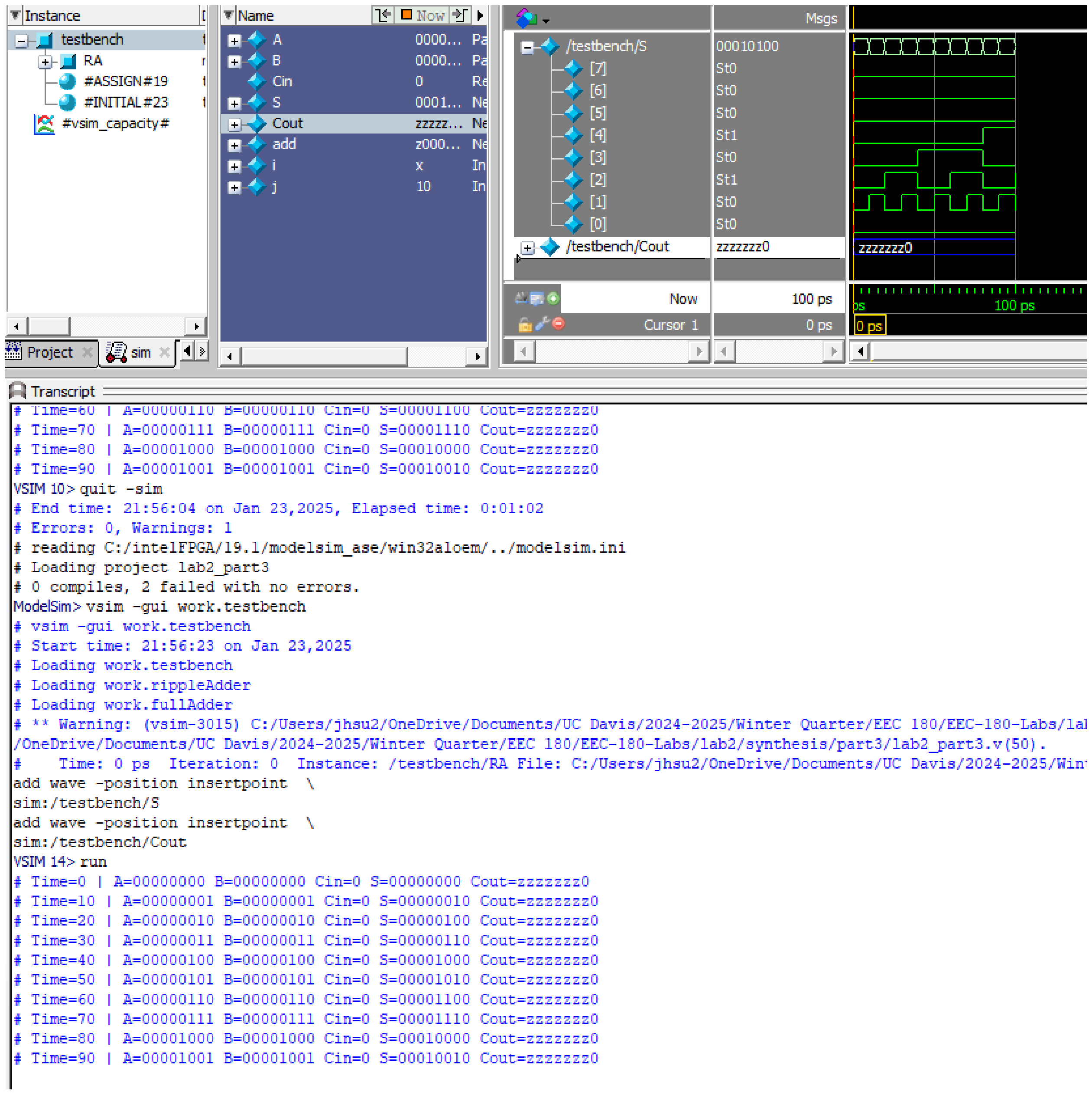
Part 2



```
couldn't open "transcript": permission denied
# Reading C:/intelFPGA/19.1/modelsim_ase/tcl/vsim/pref.tcl
# Loading project lab2_part2-test-1
ModelSim> vsim -gui work.testbench
# vsim -gui work.testbench
# Start time: 21:52:41 on Jan 23,2025
# Loading work.testbench
# Loading work.multiplier
# Loading work.full_adder
add wave -position insertpoint  \
sim:/testbench/P
VSIM 3> run
# A=0000 B=0001 P=00000000
# A=0001 B=0010 P=00000010
# A=0010 B=0011 P=00000110
# A=0011 B=0100 P=00001100
# A=0100 B=0101 P=00010100
# A=0101 B=0110 P=00011110
# A=0110 B=0111 P=00101010
# A=0111 B=1000 P=00111000
# A=1000 B=1001 P=01001000
# A=1001 B=1010 P=01011010

VSIM 4>
```

Part 3



```
# Time=60 | A=00000110 B=00000110 Cin=0 S=00001100 Cout=zzzzzzz0
# Time=70 | A=00000111 B=00000111 Cin=0 S=00001110 Cout=zzzzzzz0
# Time=80 | A=00001000 B=00001000 Cin=0 S=00010000 Cout=zzzzzzz0
# Time=90 | A=00001001 B=00001001 Cin=0 S=00010010 Cout=zzzzzzz0
VSIM 10> quit -sim
# End time: 21:56:04 on Jan 23,2025, Elapsed time: 0:01:02
# Errors: 0, Warnings: 1
# reading C:/intelFPGA/19.1/modelsim_ase/win32aloem/../modelsim.ini
# Loading project lab2_part3
# 0 compiles, 2 failed with no errors.
ModelSim> vsim -gui work.testbench
# vsim -gui work.testbench
# Start time: 21:56:23 on Jan 23,2025
# Loading work.testbench
# Loading work.rippleAdder
# Loading work.fullAdder
# ** Warning: (vsim-3015) C:/Users/jhsu2/OneDrive/Documents/UC Davis/2024-2025/Winter Quarter/EEC 180/EEC-180-Labs/lal
/OneDrive/Documents/UC Davis/2024-2025/Winter Quarter/EEC 180/EEC-180-Labs/lab2/synthesis/part3/lab2_part3.v(50).
#    Time: 0 ps  Iteration: 0  Instance: /testbench/RA File: C:/Users/jhsu2/OneDrive/Documents/UC Davis/2024-2025/Win
add wave -position insertpoint  \
sim:/testbench/S
add wave -position insertpoint  \
sim:/testbench/Cout
VSIM 14> run
# Time=0  | A=00000000 B=00000000 Cin=0 S=00000000 Cout=zzzzzzz0
# Time=10 | A=00000001 B=00000001 Cin=0 S=00000010 Cout=zzzzzzz0
# Time=20 | A=00000010 B=00000010 Cin=0 S=00000100 Cout=zzzzzzz0
# Time=30 | A=00000011 B=00000011 Cin=0 S=00000110 Cout=zzzzzzz0
# Time=40 | A=00000100 B=00000100 Cin=0 S=00001000 Cout=zzzzzzz0
# Time=50 | A=00000101 B=00000101 Cin=0 S=00001010 Cout=zzzzzzz0
# Time=60 | A=00000110 B=00000110 Cin=0 S=00001100 Cout=zzzzzzz0
# Time=70 | A=00000111 B=00000111 Cin=0 S=00001110 Cout=zzzzzzz0
# Time=80 | A=00001000 B=00001000 Cin=0 S=00010000 Cout=zzzzzzz0
# Time=90 | A=00001001 B=00001001 Cin=0 S=00010010 Cout=zzzzzzz0
```