**Question:** 01
**DATE:** October 6, 2017
Sabbir Ahmed

# 1 Background

Consider the following 3-input,2-output truth table:

| a | b | c | y | z |
|---|---|---|---|---|
| 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 0 | 1 | 1 |
| 1 | 0 | 0 | 0 | 0 |
| otherwise | | | 0 | 1 |

where - denotes don't care

Implement the table as a module using the following variations. Synthesize each version and submit a screen-capture of the RTL view.

a) using behavioral code with if statements

b) using concurrent continuous assignment statement(s)

c) using behavioral code with case statements

# 2 Implementation

## 2.1 Using Behavioral Code With 'If' Statements

The module implementation along with its testbench can be found in the 'scripts' directory. A sample of the waveform generated is provided:

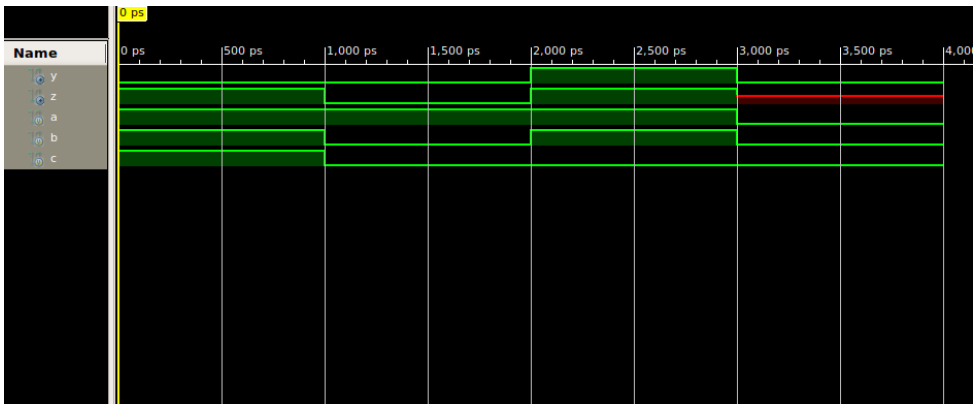


Figure 1: Waveform Generated from Part 1 Version A Test Bench

The module was synthesized and its RTL view was generated:

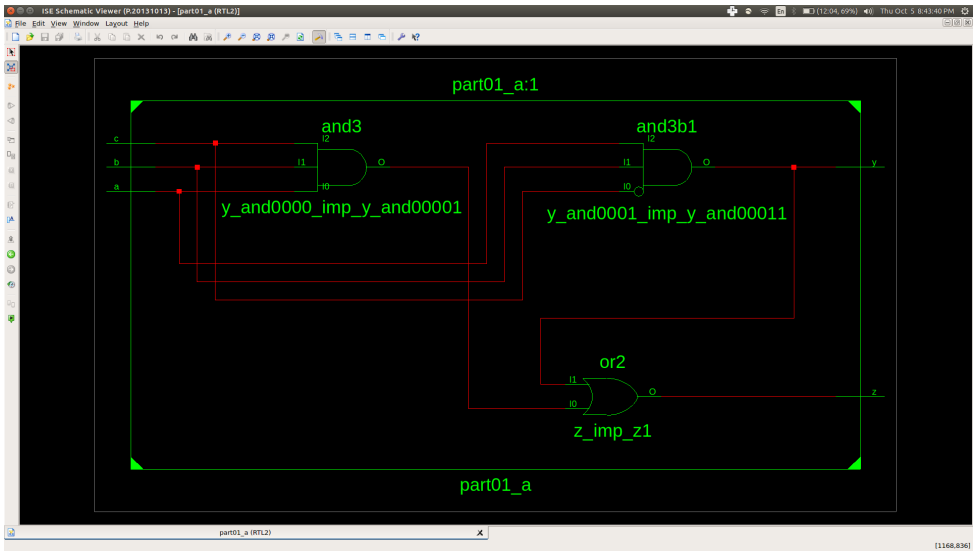


Figure 2: RTL View of Part 1 Version A

## 2.2 Using Concurrent Continuous Assignment Statement(s)

The module implementation along with its testbench can be found in the 'scripts' directory. A sample of the waveform generated is provided:

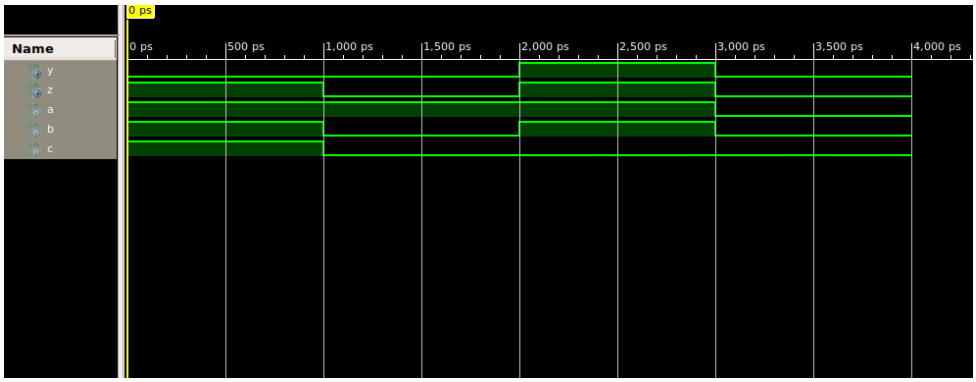


Figure 3: Waveform Generated from Part 1 Version B Test Bench

The module was synthesized and its RTL view was generated:

## 2.3 Using Behavioral Code With Case Statements

The module implementation along with its testbench can be found in the 'scripts' directory. A sample of the waveform generated is provided:

The module was synthesized and its RTL view was generated:

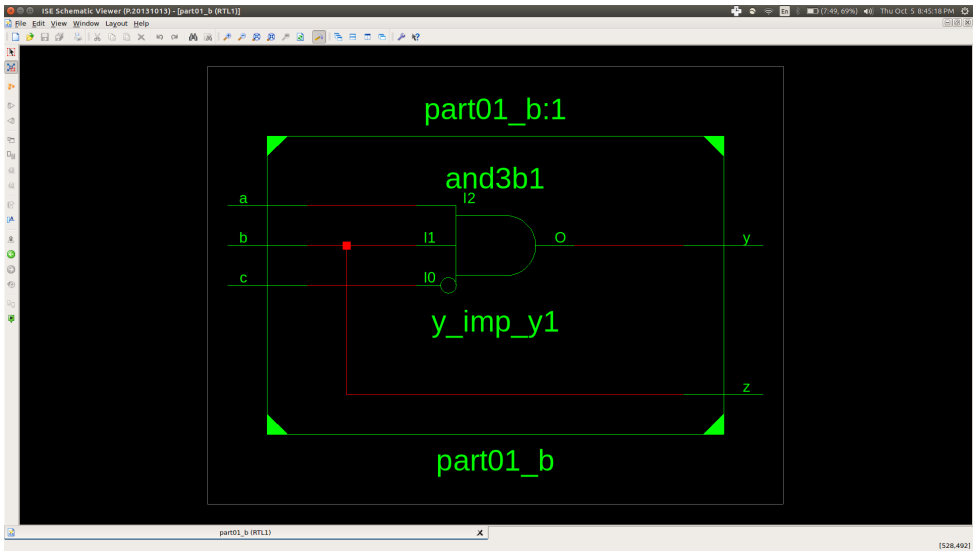


Figure 4: RTL View of Part 1 Version B

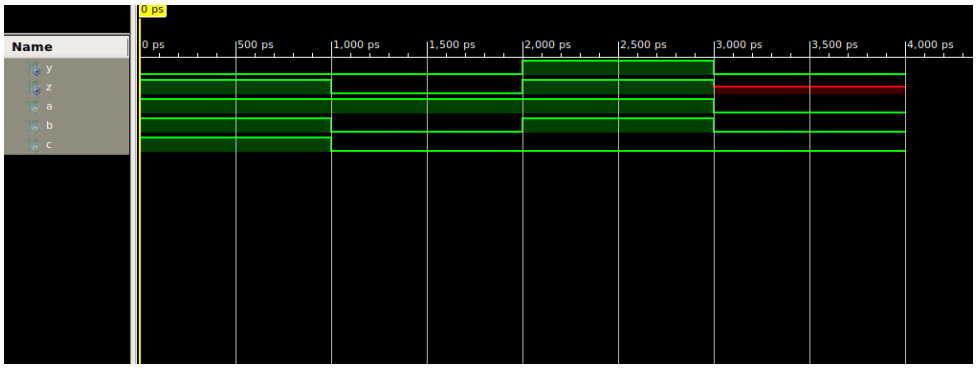


Figure 5: Waveform Generated from Part 1 Version C Test Bench

# 3 Observations

The Register Transfer Level of Version B appears to have the simplest harware implementation due to its minimized equations. However, using concurrent continuous assignemnt statements do not allow any explicit `don't care' assignments.

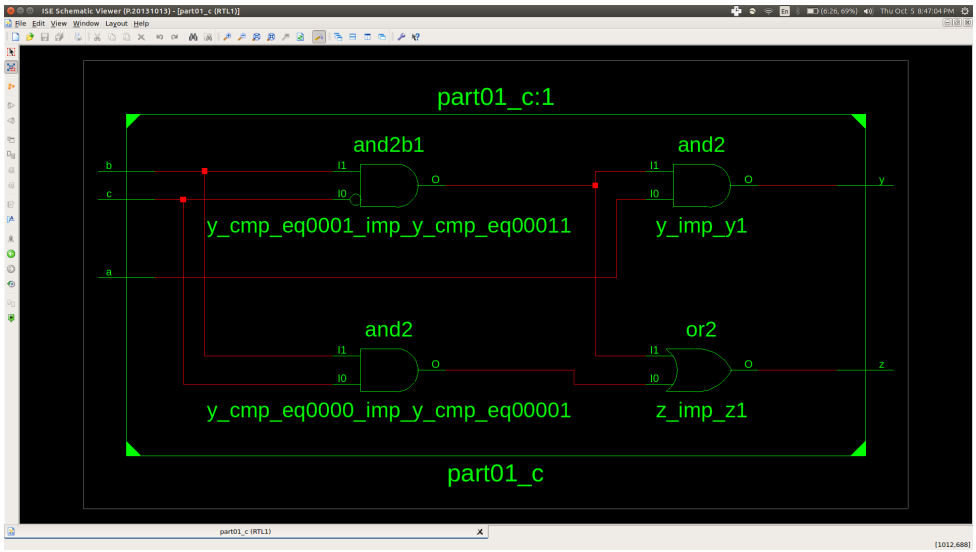


Figure 6: RTL View of Part 1 Version C