Simple Adder and Subtractor Testing with UVM

Justin Bui

October 18, 2018

Contents

1 Introduction

This document discusses the adder/subtractor circuit built to help introduce the concepts of UVM. UVM, or the Universay Verification Methodology is a testing and validation platform which allows for the design and development of testbenches and more advanced simulations of our SystemVerilog designs. This project's aim is to briefly introduce the basics of UVM, as well as develop a simple testbench based on the UVM Hello World Example. This project has been completely written and tested using the **EDA Playground**.

The code for my UVM Adder/Subtractor can be found here.

2 Project Description

The Adder and Subtractor project is a simple implementation of an 8 bit circuit designed to introduce the concepts of UVM testing and testbench development. In order to implement the testbench prototype provided by the **Hello World** example, I had to modify the number of interface signals (we know them as ports when using Vivado). For this project, I use the following signals:

- **clock** - 1 bit clock
- **reset** - 1 bit reset
- **opp** - 1 bit operator
- **numA** - 8 bit number A
- **numB** - 8 bit number B
- **sum** - 8 bit output
- **cout** - 1 bit carry out

The logic for this circuit is pretty simple, and can be implemented using one **If-Else** statement. The operation values are defined in the project requirements. A value of **0** for the operator results in an **add** operation, and a value of **1** testults in a subtraction operation. The system performs the operation on the positive clock edge. In order to account for the carry bit, I make use of an internal sum value, **isum**, which is 9 bits long to allow for the storage of the carry out. After the caluclation is performed, values of sum and cout are updated. All of the signals are then printed using a system print.

3 Design Validation

The strenght of UVM is in the ability to develop test benches, allowing us to determine whether the functionality of our device matches our expectations. The breakdown of the UVM testbench is well documented, and a quite complex.

Test benches allow for multiple environments, sequences, and drivers, however, for our introductory project, we only make use of one driver, sequence and environment. I have written by sequence to generate 3 random signals, one for the operator bit, and two for the 8 bit numbers **numA** and **numB**. I have programmed the device to run through 25 loops, the results of which are printed in the log file when the program is run.

The Log at the bottom of the screen will print out an "overview" which essentially sumarizes the UVM outputs. Figure 1 below shows the UVM outputs for the simulation I ran prior to submission of this document. Figure 2 shows the results for the 25 simulated runs.

```
# KERNEL: UVM_INFO :   28
# KERNEL: UVM_WARNING :    1
# KERNEL: UVM_ERROR :    0
# KERNEL: UVM_FATAL :    0
# KERNEL: ** Report counts by id
# KERNEL: []     1
# KERNEL: [DUT]    25
# KERNEL: [RNTST]     1
# KERNEL: [TEST_DONE]     1
# KERNEL: [UVM/RELNOTES]     1
```

Figure 1: UVM Outputs

In order to verify the operating is working correctly, I checked the calculated values by hand. Although this is a slow method, I was able to verify the operation of the adder/subtractor. It is likely that, should time permit in the future, the testing capabilities will be expanded to include more in-depth test coverage, as well as test validation.

```
# KERNEL: UVM_WARNING /home/runner/my_testbench_pkg.svh(76) @ 10: uvm_test_top [] Performing Calulations!
# KERNEL: UVM_INFO /home/runner/design.sv(36) @ 15: reporter [DUT] Info: Opp=1, numA=0xbc, NumB=0x43, Sum=0x79, Cout=0
# KERNEL: UVM_INFO /home/runner/design.sv(36) @ 25: reporter [DUT] Info: Opp=0, numA=0xf3, NumB=0x78, Sum=0x6b, Cout=1
# KERNEL: UVM_INFO /home/runner/design.sv(36) @ 35: reporter [DUT] Info: Opp=0, numA=0x60, NumB=0xe1, Sum=0x41, Cout=1
# KERNEL: UVM_INFO /home/runner/design.sv(36) @ 45: reporter [DUT] Info: Opp=1, numA=0x03, NumB=0x7e, Sum=0x85, Cout=1
# KERNEL: UVM_INFO /home/runner/design.sv(36) @ 55: reporter [DUT] Info: Opp=1, numA=0xdd, NumB=0x50, Sum=0x8d, Cout=0
# KERNEL: UVM_INFO /home/runner/design.sv(36) @ 65: reporter [DUT] Info: Opp=0, numA=0xee, NumB=0x57, Sum=0x45, Cout=1
# KERNEL: UVM_INFO /home/runner/design.sv(36) @ 75: reporter [DUT] Info: Opp=0, numA=0x35, NumB=0x92, Sum=0xc7, Cout=0
# KERNEL: UVM_INFO /home/runner/design.sv(36) @ 85: reporter [DUT] Info: Opp=1, numA=0xb2, NumB=0x01, Sum=0xb1, Cout=0
# KERNEL: UVM_INFO /home/runner/design.sv(36) @ 95: reporter [DUT] Info: Opp=1, numA=0x66, NumB=0xa5, Sum=0xc1, Cout=1
# KERNEL: UVM_INFO /home/runner/design.sv(36) @ 105: reporter [DUT] Info: Opp=0, numA=0x51, NumB=0x7e, Sum=0xcf, Cout=0
# KERNEL: UVM_INFO /home/runner/design.sv(36) @ 115: reporter [DUT] Info: Opp=0, numA=0x72, NumB=0x8b, Sum=0xfd, Cout=0
# KERNEL: UVM_INFO /home/runner/design.sv(36) @ 125: reporter [DUT] Info: Opp=1, numA=0xc9, NumB=0xcc, Sum=0xfd, Cout=1
# KERNEL: UVM_INFO /home/runner/design.sv(36) @ 135: reporter [DUT] Info: Opp=1, numA=0x57, NumB=0x42, Sum=0x15, Cout=0
# KERNEL: UVM_INFO /home/runner/design.sv(36) @ 145: reporter [DUT] Info: Opp=0, numA=0x1c, NumB=0xed, Sum=0x09, Cout=1
# KERNEL: UVM_INFO /home/runner/design.sv(36) @ 155: reporter [DUT] Info: Opp=0, numA=0x17, NumB=0xcc, Sum=0xe3, Cout=0
# KERNEL: UVM_INFO /home/runner/design.sv(36) @ 165: reporter [DUT] Info: Opp=1, numA=0x48, NumB=0xdf, Sum=0x69, Cout=1
# KERNEL: UVM_INFO /home/runner/design.sv(36) @ 175: reporter [DUT] Info: Opp=1, numA=0xb0, NumB=0x27, Sum=0x89, Cout=0
# KERNEL: UVM_INFO /home/runner/design.sv(36) @ 185: reporter [DUT] Info: Opp=0, numA=0x4f, NumB=0xa4, Sum=0xf3, Cout=0
# KERNEL: UVM_INFO /home/runner/design.sv(36) @ 195: reporter [DUT] Info: Opp=0, numA=0x24, NumB=0x55, Sum=0x79, Cout=0
# KERNEL: UVM_INFO /home/runner/design.sv(36) @ 205: reporter [DUT] Info: Opp=1, numA=0x2f, NumB=0x3a, Sum=0xf5, Cout=1
# KERNEL: UVM_INFO /home/runner/design.sv(36) @ 215: reporter [DUT] Info: Opp=1, numA=0x71, NumB=0x54, Sum=0x1d, Cout=0
# KERNEL: UVM_INFO /home/runner/design.sv(36) @ 225: reporter [DUT] Info: Opp=0, numA=0xea, NumB=0xa3, Sum=0x8d, Cout=1
# KERNEL: UVM_INFO /home/runner/design.sv(36) @ 235: reporter [DUT] Info: Opp=0, numA=0x99, NumB=0x26, Sum=0xbf, Cout=0
# KERNEL: UVM_INFO /home/runner/design.sv(36) @ 245: reporter [DUT] Info: Opp=1, numA=0x7e, NumB=0xdd, Sum=0xa1, Cout=1
# KERNEL: UVM_INFO /home/runner/design.sv(36) @ 255: reporter [DUT] Info: Opp=1, numA=0x9a, NumB=0xc9, Sum=0xd1, Cout=1
```

Figure 2: Results