Question: 11
DATE: October 7, 2017
Sabbir Ahmed

1 Background

Implement as a strictly combinatorial module ALU that will add or subtract two 5-bit two-signed values a and b based on a control signal sub. The outputs should be a 5-bit two's-complement result y and an over-/under-flow bit c. Implement two variations of this module using

a) structurally using full 1-bit adders that are in turn built from Verilog primitives

b) using behavioral code with arithmetic operators

2 Implementation

2.1 Structurally Using Full 1-bit Adders That Are In Turn Built From Verilog Primitives

The structural version of the 5-bit adder/subtractor was built using basic gates. Circuit diagrams were used as a reference to minimize the number of gates and ensure accurate outputs. A full 1-bit adder was implemented as a separate module and imported into the circuit design.

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

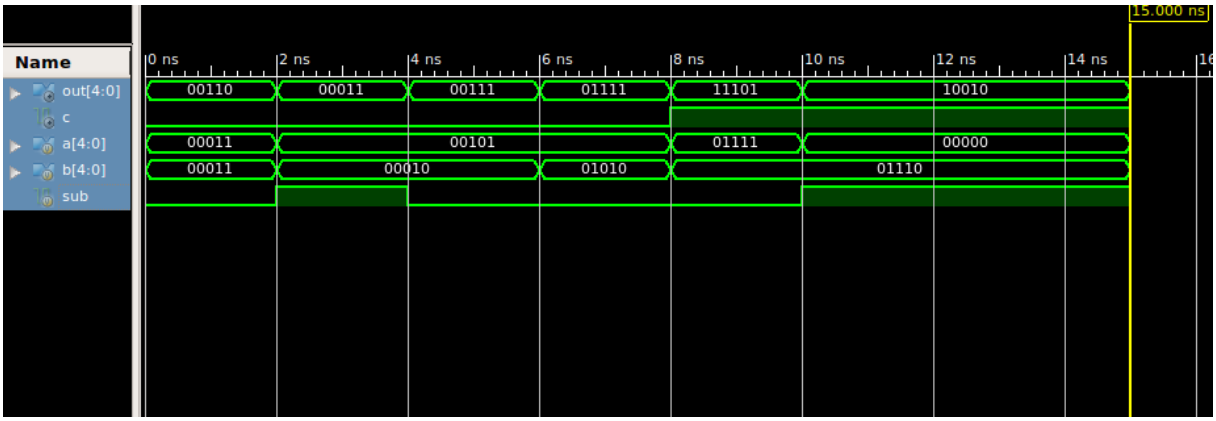

Figure 1: Waveform Generated from Part 11 Structural Version Test Bench

Table 1: Inputs and Outputs of The Part 11 Structural Version Test Bench

a	b	sub	c	out
00011 (3)	00011 (3)	0	0	00110 (6)
00101 (5)	00010 (2)	1	0	00011 (3)
00101 (5)	00010 (2)	0	0	00111 (7)
00101 (5)	01010 (10)	0	0	01111 (15)
01111 (15)	01110 (14)	0	1	11101 (29)
00000 (0)	01110 (14)	1	1	10010 (18)

2.2 Using Behavioral Code With Arithmetic Operators

The behavioral version of the 5-bit adder proved to be a much simpler implementation, requiring much fewer lines of code.

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

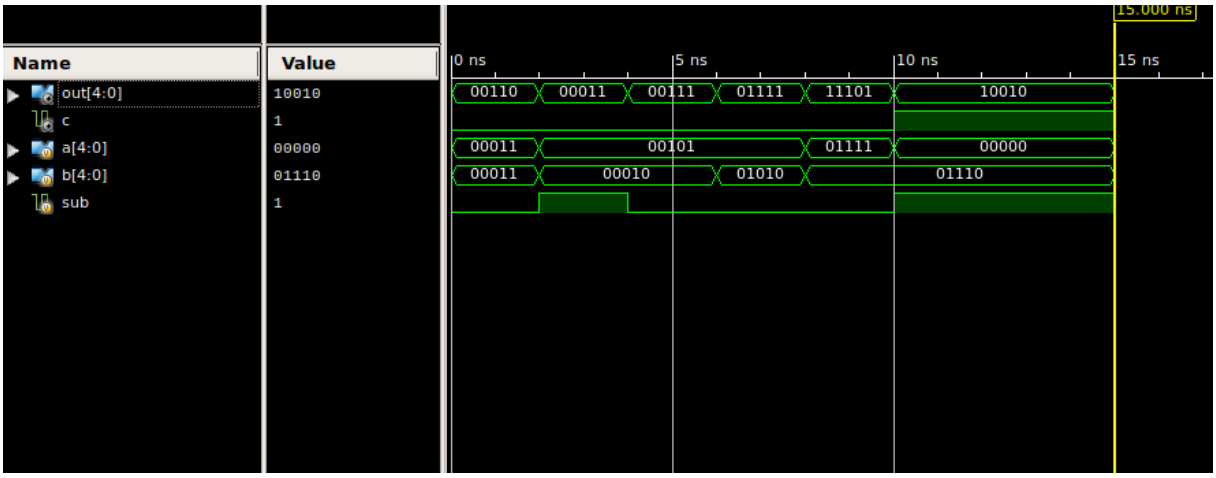

Figure 2: Waveform Generated from Part 11 Behavioral Version Test Bench

Table 2: Inputs and Outputs of The Part 11 Behavioral Version Test Bench

a	b	sub	c	out
00011 (3)	00011 (3)	0	0	00110 (6)
00101 (5)	00010 (2)	1	0	00011 (3)
00101 (5)	00010 (2)	0	0	00111 (7)
00101 (5)	01010 (10)	0	0	01111 (15)
01111 (15)	01110 (14)	0	1	11101 (29)
00000 (0)	01110 (14)	1	1	10010 (18)

3 Observations

Both the versions of the implementation provided identical outputs. The structural implementation proved to be the more difficult of the 2 versions because of its requirement of understanding the physical design of multi-bit adders and subtractors. A basic comparison in the

behavioral version of the implementation appeared to be equivalent to using several gates in the structural version.