

Project: Part #2 for ECE 152

High Performance Adder

Must be submitted electronically by 10:00AM on Monday, Feb 18

In this part of the project, you will implement a 16-bit 2's complement adder. This adder will be one of the components used in your implementation of the Duke 152/16. The adder uses a combination of carry-lookahead (CLA) and carry-select techniques. The lowest 4-bit addition (adding bits 0-3 of the two addends) is performed by a single 4-bit CLA. The next most significant 4-bit addition (for bits 4-7) is performed using two 4-bit CLAs that are used in a carry-select fashion. Similarly, the addition of bits 8-11 and the addition of bits 12-15 are each performed using two CLAs in a carry-select fashion. Thus, you will be using 7 identical 4-bit CLA adders for the entire 16-bit adder.

The adder takes a single-bit input (`ctrl_subtract`) which, if true, denotes that subtraction should be performed instead of addition ($\text{sum} = \text{addendA} - \text{addendB}$). Figure 1 illustrates the adder, including the exact signal names that you must use in your design (to facilitate testing). Note that the adder also produces the result of logical AND and logical XOR; these results (`AandB`, `AxorB`) are produced as part of addition, when computing the propagate and generate signals (i.e., you don't need to add extra AND and XOR logic to your adder). Hint: propagate signals can be implemented with XOR instead of OR (you should be able to tell me why this is true, for example, on the midterm). If the `ctrl_subtract` signal is high, then the values of `AandB` and `AxorB` are "don't cares" (i.e., you don't have to worry about them).

After designing your adder, you must test it thoroughly to demonstrate that it works correctly in all cases. When you turn in this assignment, you must show the results of the test that I have provided for you at <http://www.ee.duke.edu/~sorin/ece152/project/addertest.vwf>. This test should show me how fast your adder can correctly perform the test. In addition, during the grading of this assignment, I will run additional tests (not provided), so don't assume you can ignore bugs that don't manifest themselves during the test that I have provided. Nor can you ignore slow circuit paths through your adder just because they aren't manifested by the test I have provided. I will use my tests to see how fast your adder is.

To submit this assignment, create a packaged and compressed file called `project2.tar.gz`. The high level `.bdf` file must be named `adder.bdf`. Names of lower-level components are unrestricted. Submit `project2.tar.gz` in the same way that you submitted Project Part #1.

You may re-submit as often as you like, but a re-submission will overwrite whatever you've previously submitted for this assignment. I will grade whatever has been submitted before 10:00AM on Monday, February 18th.

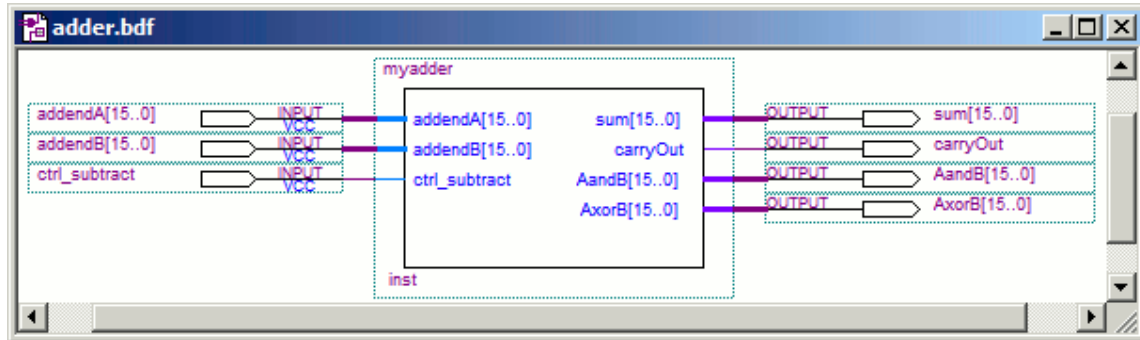


FIGURE 1. Illustration of 16-bit adder