WRIGHT STATE UNIVERSITY

EE4620/6620, CEG4324/6324
Digital Integrated Circuit Design
with PLDs and FPGAs

Lab 3 (Spring 2024):

Xilinx Vivado Booth Multiplier Design, Simulation, and Test in FPGA

A. Introduction

The main objective of this lab is for you to explore the design space for Booth multiplier circuit designs and to become familiar with the Xilinx Vivado simulation environment. Read all directions carefully. You will design a VHDL project in Xilinx Vivado and functionally verify and validate your design. Read through all of the steps before you begin the lab. Source files for lab 3 can be found in Pilot.

B. Instructions

Traditional hardware multiplication is performed in the same way multiplication is done by hand: partial products are computed, shifted appropriately, and summed. This algorithm can be slow if there are many partial products (i.e., many bits) because the output must wait until each sum is performed. Booth's algorithm reduces the number of required partial products in half. This increases the speed by reducing the total number of partial product sums that must take place. The two-stage Booth multiplier architecture is shown in Fig. 1.

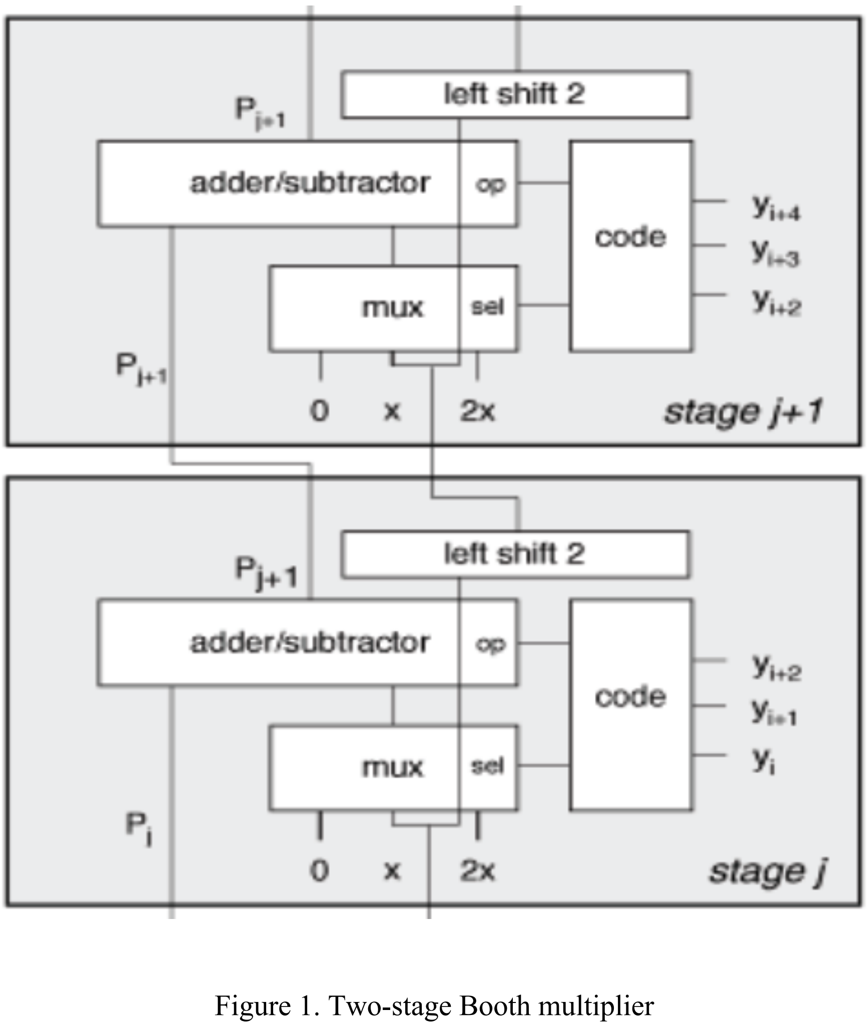

Figure 1. Two-stage Booth multiplier

1. 6x6 Booth multiplier

In this lab, you will first design a VHDL project for a 6x6 Booth multiplier in Xilinx Vivado. You optimize the multiplier for speed because Booth's multiplication algorithm is used to reduce the number of partial products, and thus the number of adders, providing a speed advantage. By optimizing the fundamental building blocks, your design also reduces the propagation delay and total area required.

C. Report [100 pts] (Submitted to your Pilot LAB Section Dropbox by 11:30 pm, Sunday, March 31, 2024)

Turn in a written report that captures your approach to the *6x6 Booth Multiplier*. It should include:

1. [40 pts] VHDL code and testbench of your *6x6 Booth Multiplier*.
2. [40 pts] Your functional verification.

   The following test cases for your *6x6 Booth Multiplier* are required:
   X_{10} = (19, 23)
   Y_{10} = (1, 13, -13, -11)
   You must test for all X*Y cases, which means you should see a total of 8 test cases.
   Note: Submit waveform snapshot of all 8 test cases. All answers in the waveforms must be labeled to show the correct results. Annotate the inputs and outputs in ***hexadecimal values*** next to their binary equivalents in the simulation waveform.

3. [20 pts] Area, power, and delay analysis, and the post-synthesis logic design schematic.