ECE 530 Digital Hardware Design
Project 01 - ALU Design

Richard Groves

September 30, 2023

1 Overview

In this project, the goal was to design an Arithmetic and Logic Unit (ALU) that executes 8 functions. The ALU operated on 4-bit unsigned numbers. The inputs and outputs of the ALU were stored using edge detection which worked at a 25kHz clock speed. The final output is exhibited on the 4-LEDs of the Zybo board.

2 Design Procedure

2.1 N-bit Ripple Carry Adder Design

The N-bit Ripple Carry Adder was designed to perform the addition of two N-bit binary numbers. It comprised a series of full adders connected so that the carry output from one adder fed into the next adder as the carry input. For the purpose of this project, the adder had a width of 4 bits.

2.1.1 Schematic

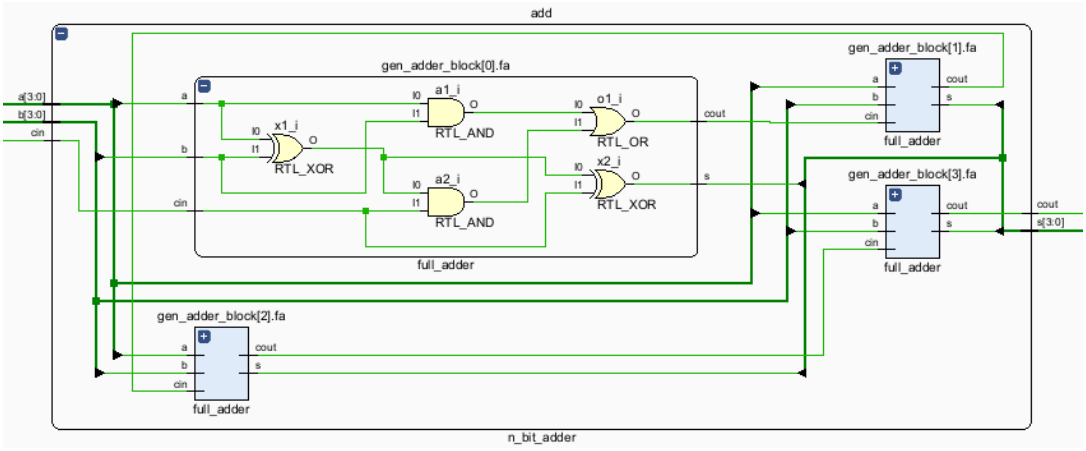

Figure 1: Schematic of the Adder

2.1.2 Simulation

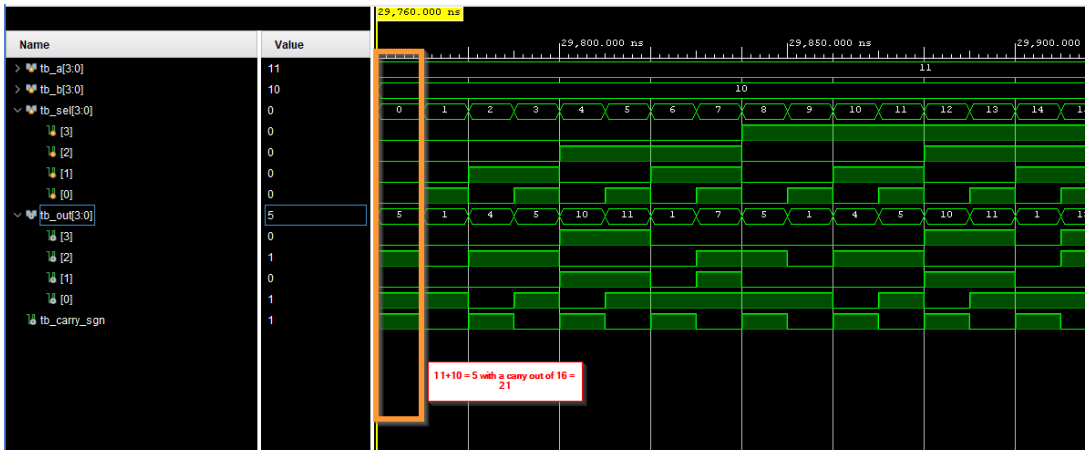

Figure 2: Simulation of the Adder

2.2 Two's Complement Design

We created a two's complement module. This module was required for the subtraction module. Taking the two's complement of the input "a" was also a function of the ALU.

2.2.1 Schematic

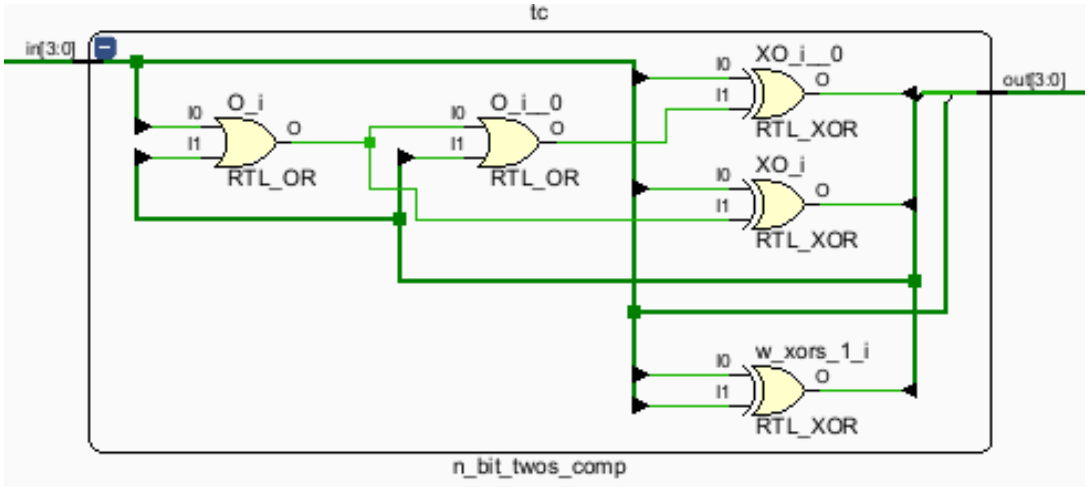

Figure 3: Schematic of the Two's Complement

2.2.2 Simulation

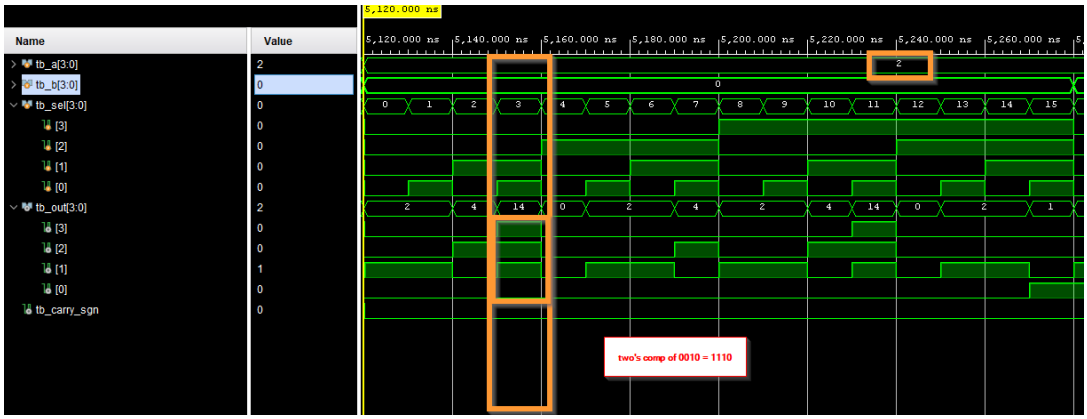

Figure 4: Simulation of the Two's Complement

2.3 N-bit Subtractor

The N-bit Subtractor was designed to perform the subtraction of two N-bit binary numbers. It comprised adding the inversion of input "b" with "a". The result of the addition determined if taking the two's complement was necessary based on the sign of the output. For the purpose of this project, the adder had a width of 4 bits.

2.3.1 Schematic

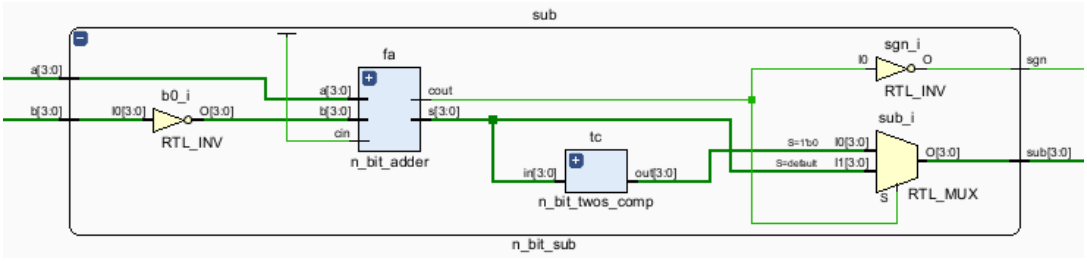

Figure 5: Schematic of the Subtractor

2.3.2 Simulation

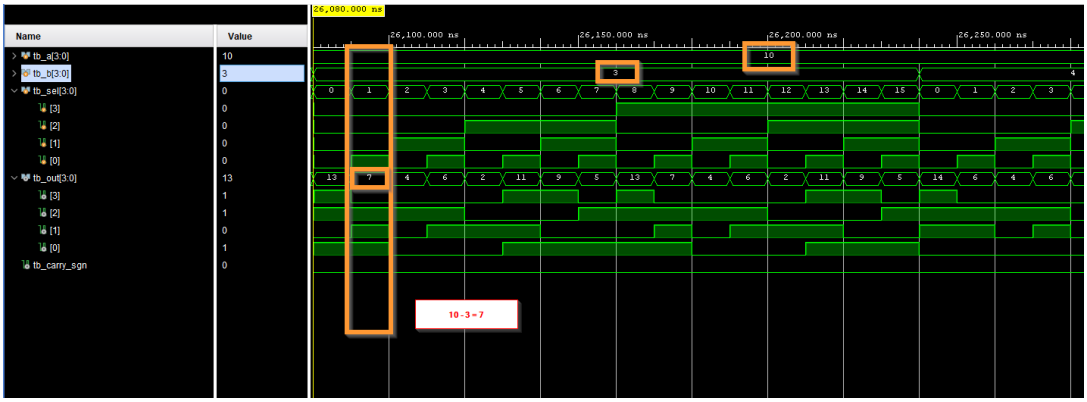

Figure 6: Simulation of the Subtractor

2.4 Binary Comparator

For the n-bit comparator, a comparator cell module was developed and used. The cell module took input bits "a" and "b" along with greater than, less than, and equal bits. These inputs were used to generate outputs for the cell that indicated the n-bit comparison, rather than just a single bit comparison.

2.4.1 Schematic

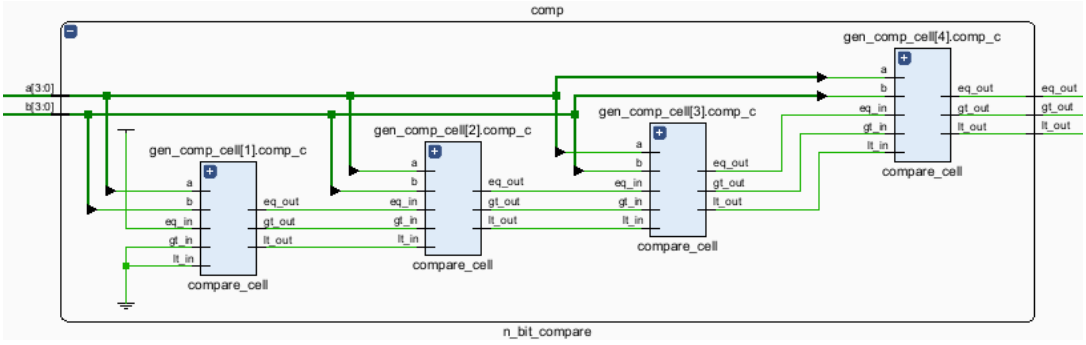

Figure 7: Schematic of the Comparator

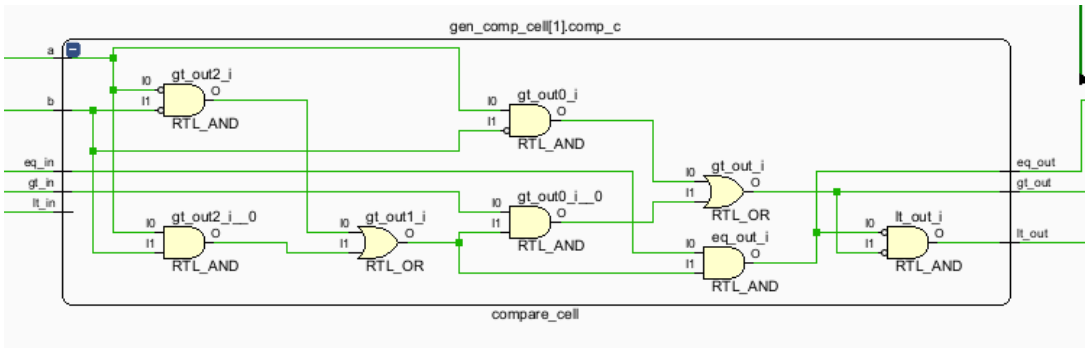

Figure 8: Schematic of the Comparator Cell

2.4.2 Simulation

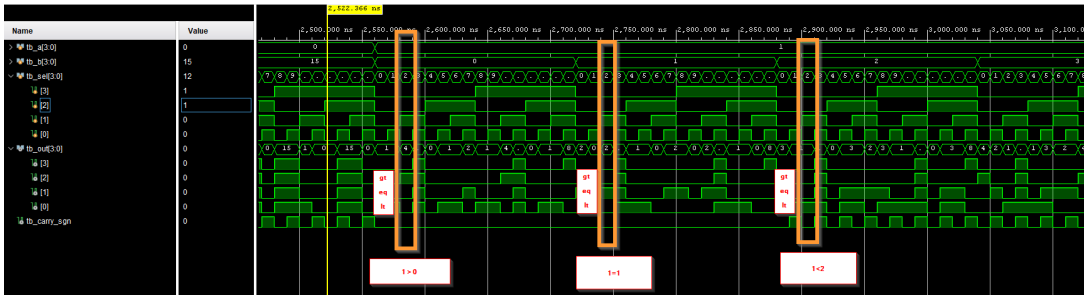

Figure 9: Simulation of the Comparator

2.5 Logical Operators

The ALU was designed to perform AND, OR and XOR functions.

2.5.1 Simulation

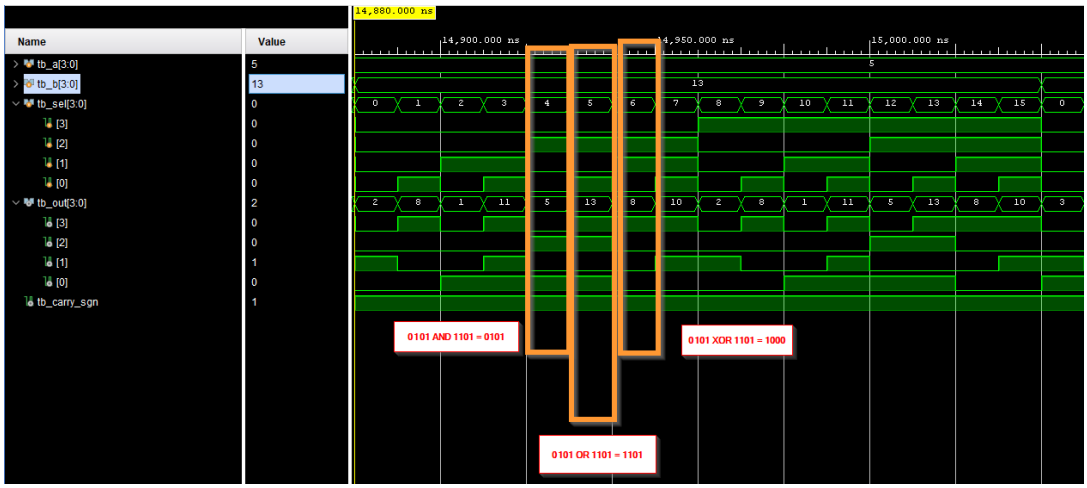

Figure 10: Simulation of AND, OR and XOR

2.6 ALU

The ALU was designed to perform the functions previously described. The ALU was an n-bit design, which made it versatile for handling data of varying bit-widths. For this project, the ALU was used for 4-bit functions.

2.6.1 Schematic

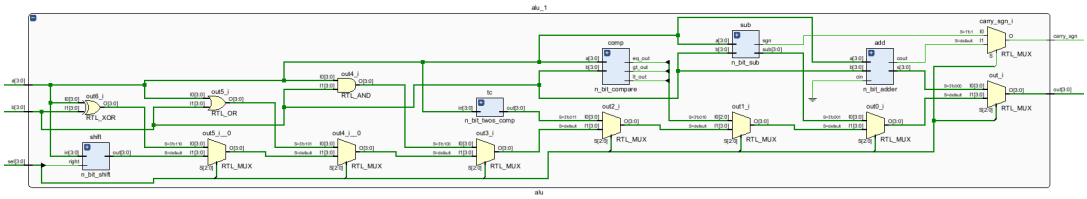

Figure 11: Schematic of the ALU

2.7 Wrapper for ALU

The wrapper of the ALU (ALU_Top) is used to capture the inputs and outputs of the ALU. As the design was implemented using 4 switches, 4 pushbuttons, and 4 LEDs, registers were employed to store the inputs and outputs. Edge detection was used on the pushbuttons (except pushbutton 0 which was an asynchronous reset), which were then used for storing registers.

2.7.1 Schematic

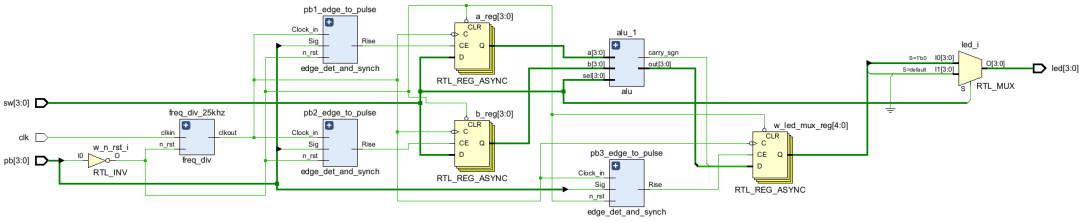

Figure 12: Schematic of the ALU_Top Module

2.8 Utilization

Utilization Post-Synthesis | **Post-Implementation**

Graph | **Table**

Resource	Utilization	Available	Utilization %
LUT	54	17600	0.31
FF	35	35200	0.10
IO	13	100	13.00
BUFG	1	32	3.13

Figure 13: Utilization Table of the ALU_Top Module

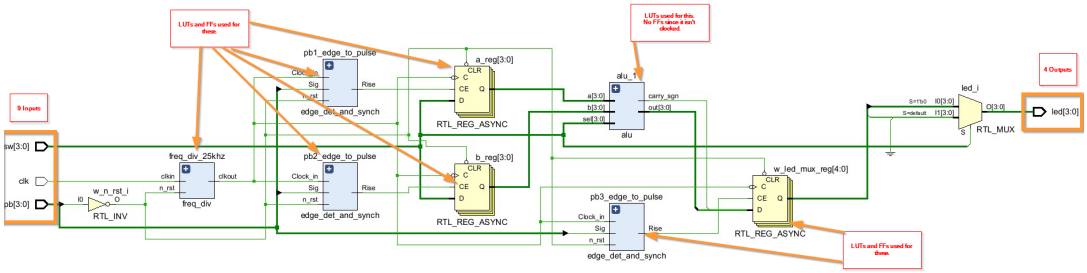

Figure 14: Correlation of Utilized Resources to Schematic Objects

2.9 Synthesis

Below is a figure showing the synthesized ALU design.

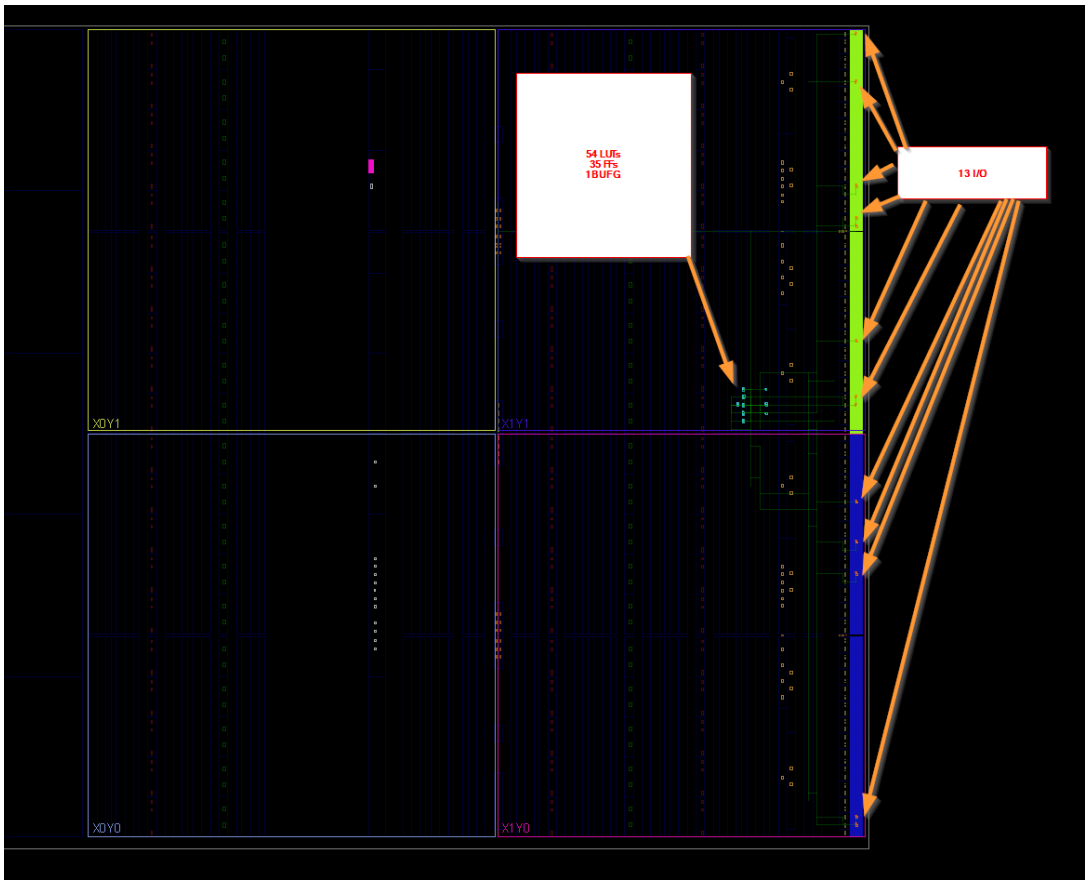

Figure 15: Synthesized Design