MP-1 Write-Up

Blake Vermeer
Kris Hall
Rohit Zambre

September 23, 2014

Date Performed: September 19, 2014
Instructors: Joseph Zambreno

1 Objective

In this lab we were introduced to the Xilinx tools and the lab environment setup. Also, we did some basic VHDL coding to manipulate a UART link.

2 Initial Test

After generating the bitfile, I uploaded the file to the Xilinx board. After running the program, LEDs 1 and 4 lit up and remained lit. I then connected to the Xilinx board over the serial link using minicom. The Xilinx board simply echoed back any data sent to it over the serial link.

3 Uppercase Conversion

4 Exploring the Design

5 Echo Delay

6 In-Flight Calculations

In this section, we modified the design so that it would echo back the sum of the previous two inputs if the last two characters entered we both 0 through 4. To do this we replaced the "echo" module with our own module called "num_adder". Below is a block diagram for the design:

The design is fairly simple. It primarily consists of a state machine and some supporting logic. The state machine is responsible for detecting new data, storing this data in registers, detecting when the UART is free, and detecting the occurrence of two consecutive numbers between 0-4 and outputting their sum. There is a submodule contained within this design (num_check) who's only job is to see if its two inputs are both 0-4 and return 1 if they are or 0 otherwise. Here is the state diagram for the state machine in "num_adder":

The state diagram is fairly straight forward. The module is normally waiting for new data to arrive. After new data arrives it waits for the UART module to be available to transmit the data. If the state machine received the signal from the "num_check" module then it proceeds to wait for the UART module to become available again and then transmits the second byte of data. The only slightly strange state is the DELAY state. This was necessary since the UART module has a one clock cycle delay between receiving new data to transmit and the time when it unsets the TX_busy_n flag to signal that the UART is busy.

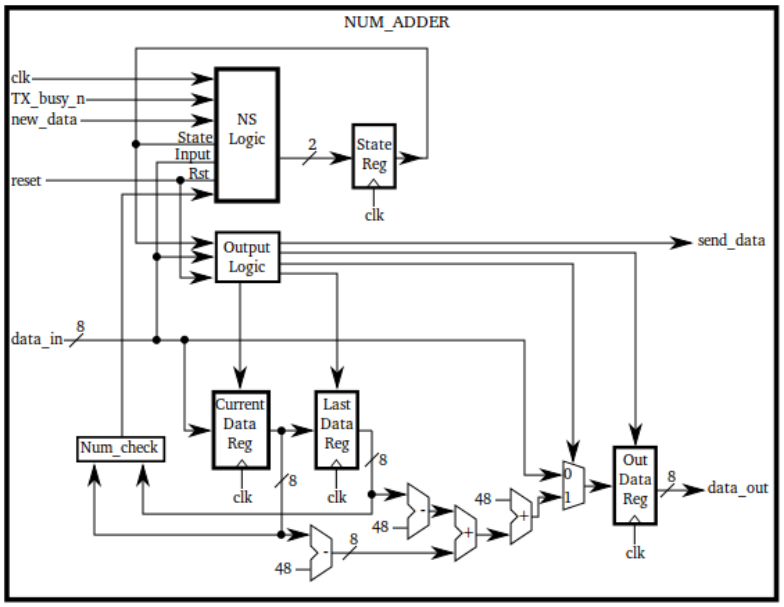

Figure 1: Part 6 - Block Diagram

7 Changing the Baud Rate

This section asked what changes would be necessary to change the baud rate of the module from 9600 to 38400. The well commented signals made that change fairly easy to find. In the "UART_driver" module you would change the "ck_div" input on the "UART_1" instance to x"011E". This is based on the formula to find the baud rate given below:

$$baud\_rate = F(clk)/(ck\_div * 3)$$

In our design, the base clock rate is 33MHz.

$$38400 = 33MHz/(ck\_div * 3)$$

$$\boldsymbol{ck\_div = 286}$$

8 Conclusion

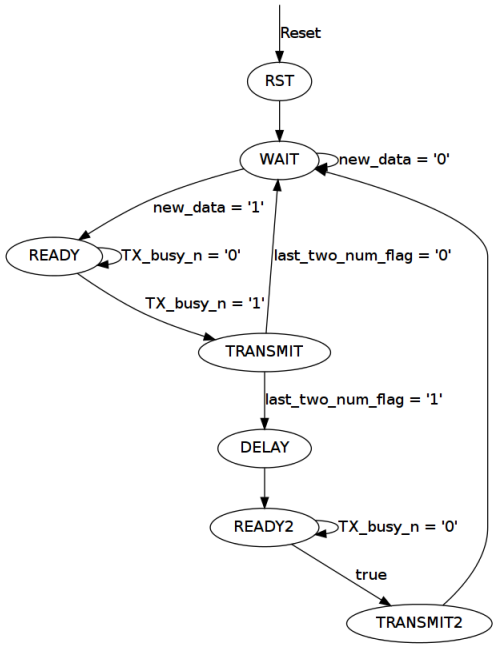

Figure 2: NUM_ADDER State Diagram