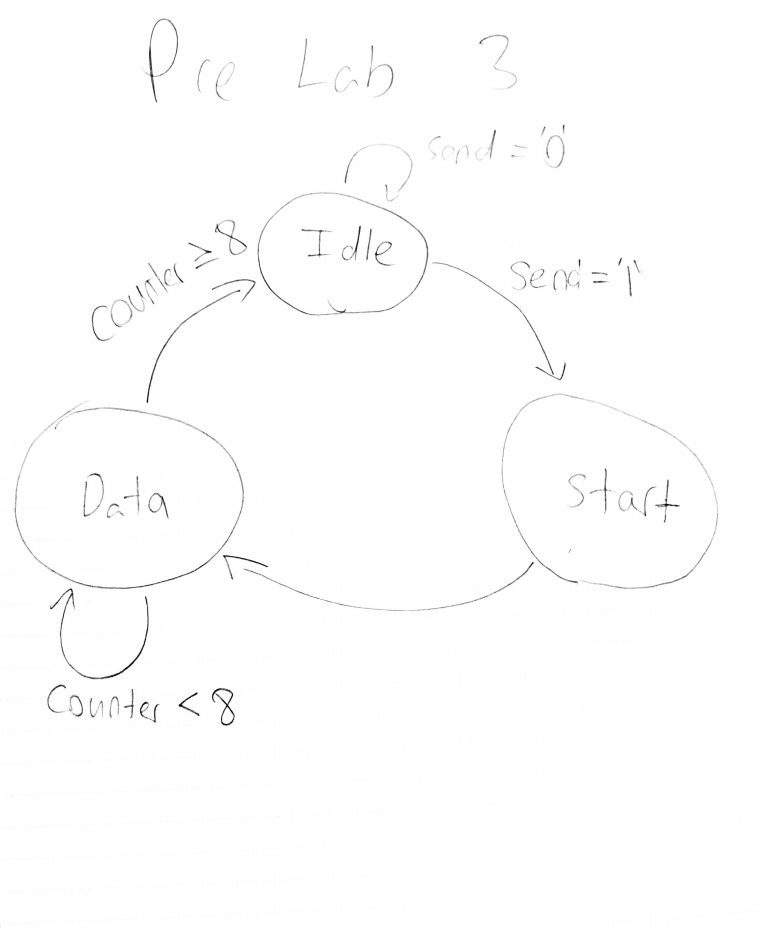
Embedded Systems 1

Lab Report 3

Gavin McKim

3/28/2019

Pre Lab:



Purpose:

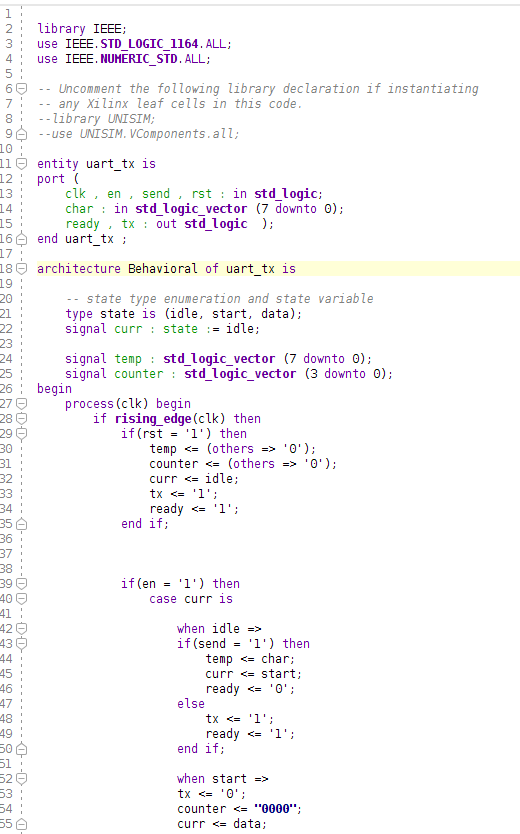
The main purpose of the lab was to introduce the idea of Finite State Machines(FSM). FSM are computational models that utilize “states”. The program performs certain actions depending on the state it is currently in. The state changes on clock ticks when certain conditions are met. The concept of FSM was demonstrated by creating a basic Universal Asynchronous Receiver Transmitter(UART). A UART uses FSM to transmit data between two machines.

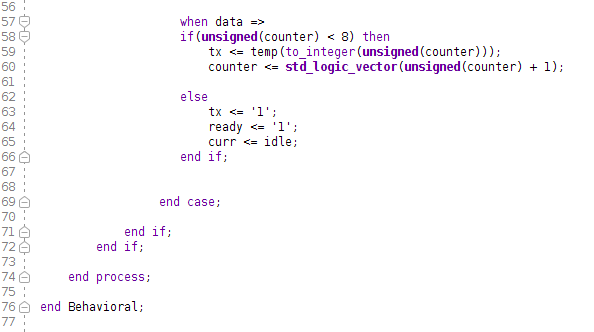
1. We Can Rebuild Him

Theory:

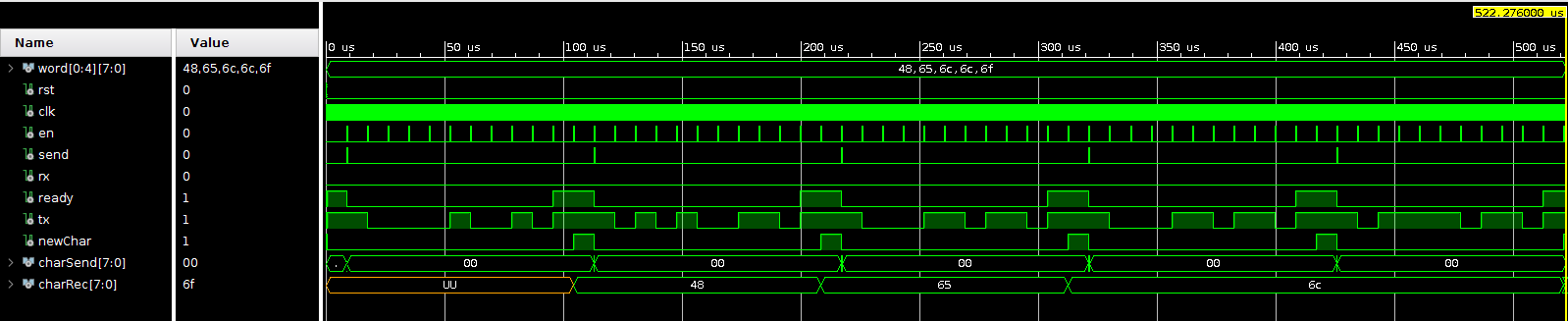
The first part of the lab involved completing the UART. The lab provided the final design of the UART and the receiver aspect of it. The first part had us create the transmitter side of the UART. This transmitter is triggered when the “line” in the UART goes low. When that happens the transmitter sends an 8-bit character one bit at a time. It does this using FSM. When the transmitter is triggered it goes in to a “start” state where it prepares to send the character. Then it transitions into the “data” state where it sends the character one bit at a time. When run with the provided test bench, it should show the correct hexadecimal values in charRec.

VHDL Code:





Simulation



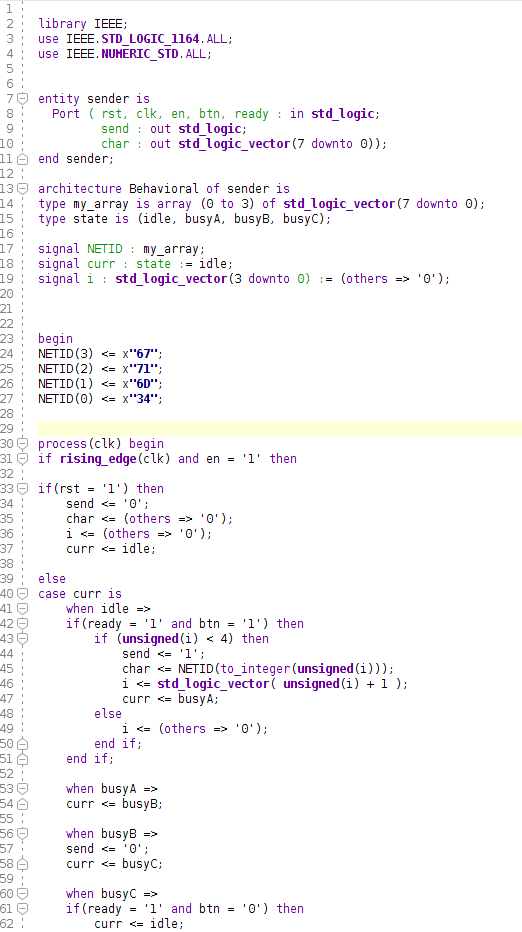
2. We Can Rebuild Him

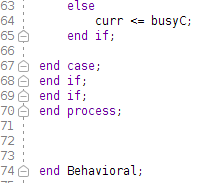
Theory

For the next part of the lab we create a program to test sending the computer information from the board. This program holds a four character array that contains my RUID(gqm4). The sender program sends this RUID utilizing a single FSM. It defaults in the idle state and when the button is pressed while the system is ready, it outputs one character of the RUID. Then it cycles through a few “busy” states. When the button is unpressed it goes from the last busy state back to idle. When it reaches the end of the RUID it starts over. This was the implemented in a top level design along with a clock divider, debouncers, and the uart. This top level design would be implemented onto a zybo board along with a PMOD JB. After using the consle to connect the devices, the Zybo’s buttons should control the program. When one button is pressed it should send only one character of my RUID until reaching the end at which point it starts over. If the reset button is pressed it should reset back to the first character in my RUID.

VHDL Code

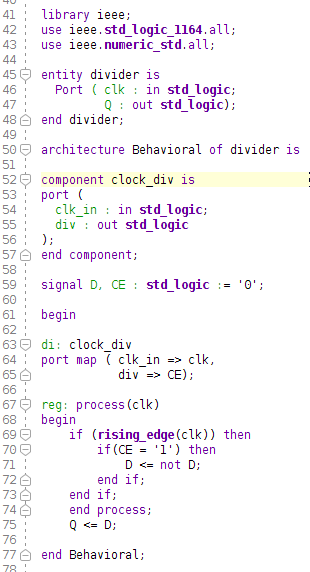
Sender



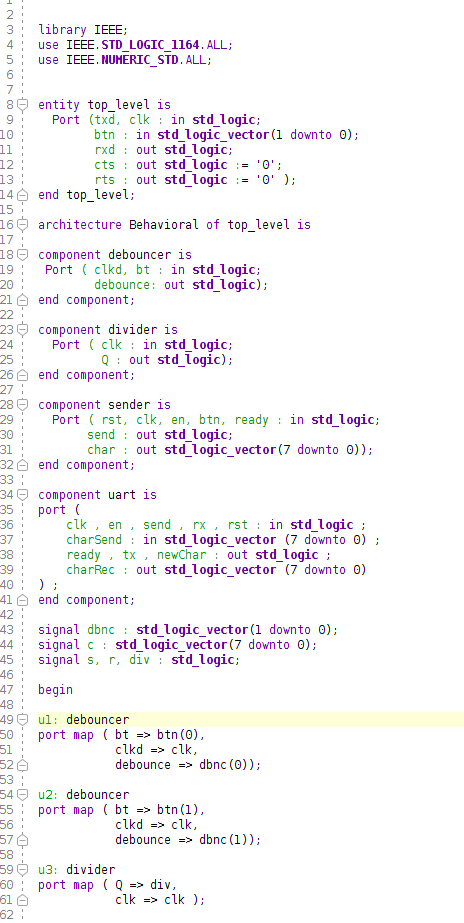


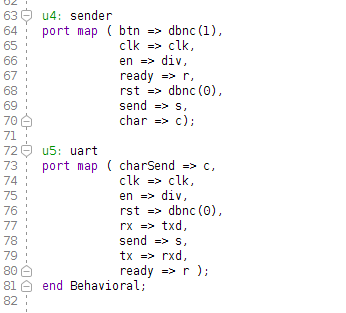
Clock Divider



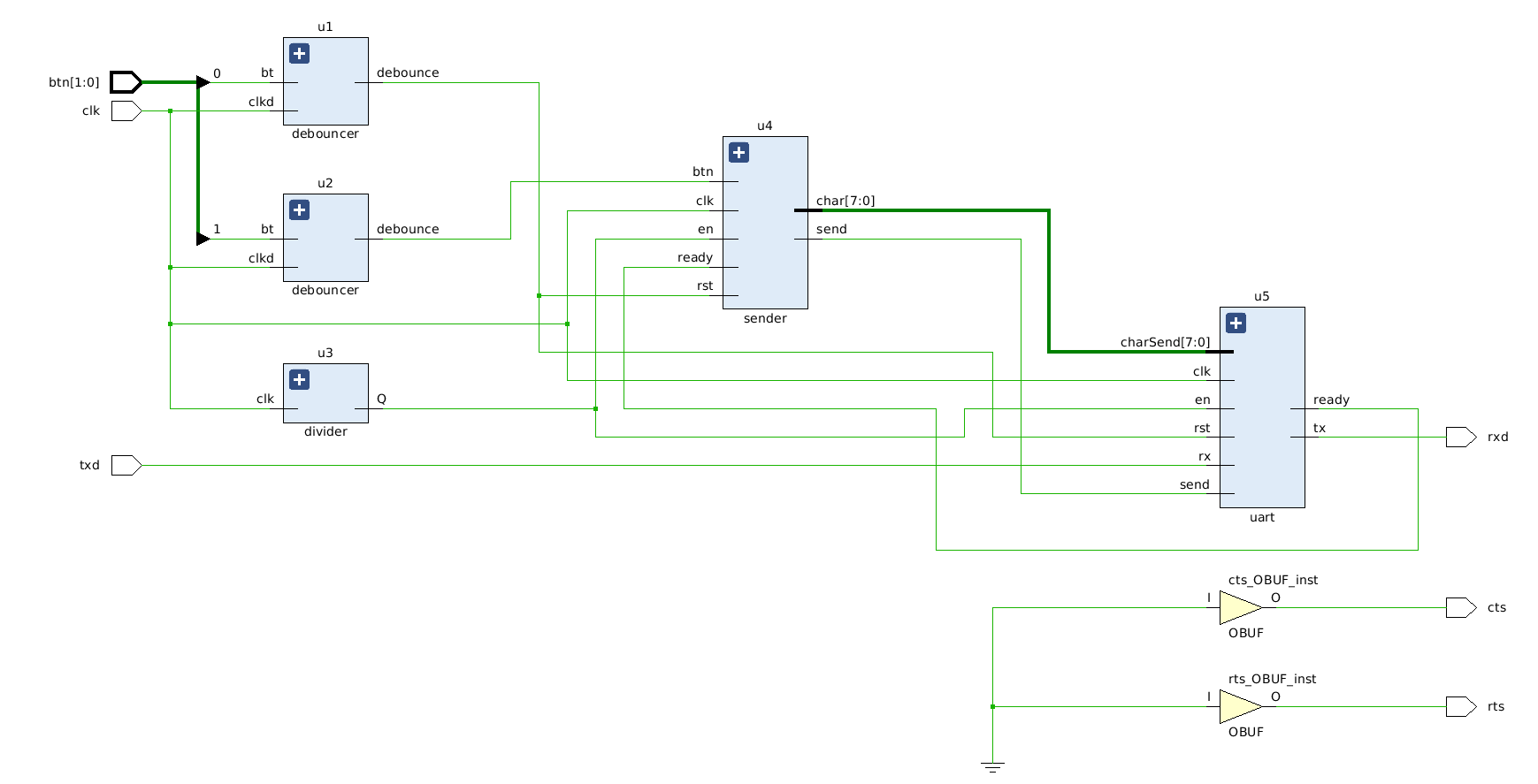


Top Level Design

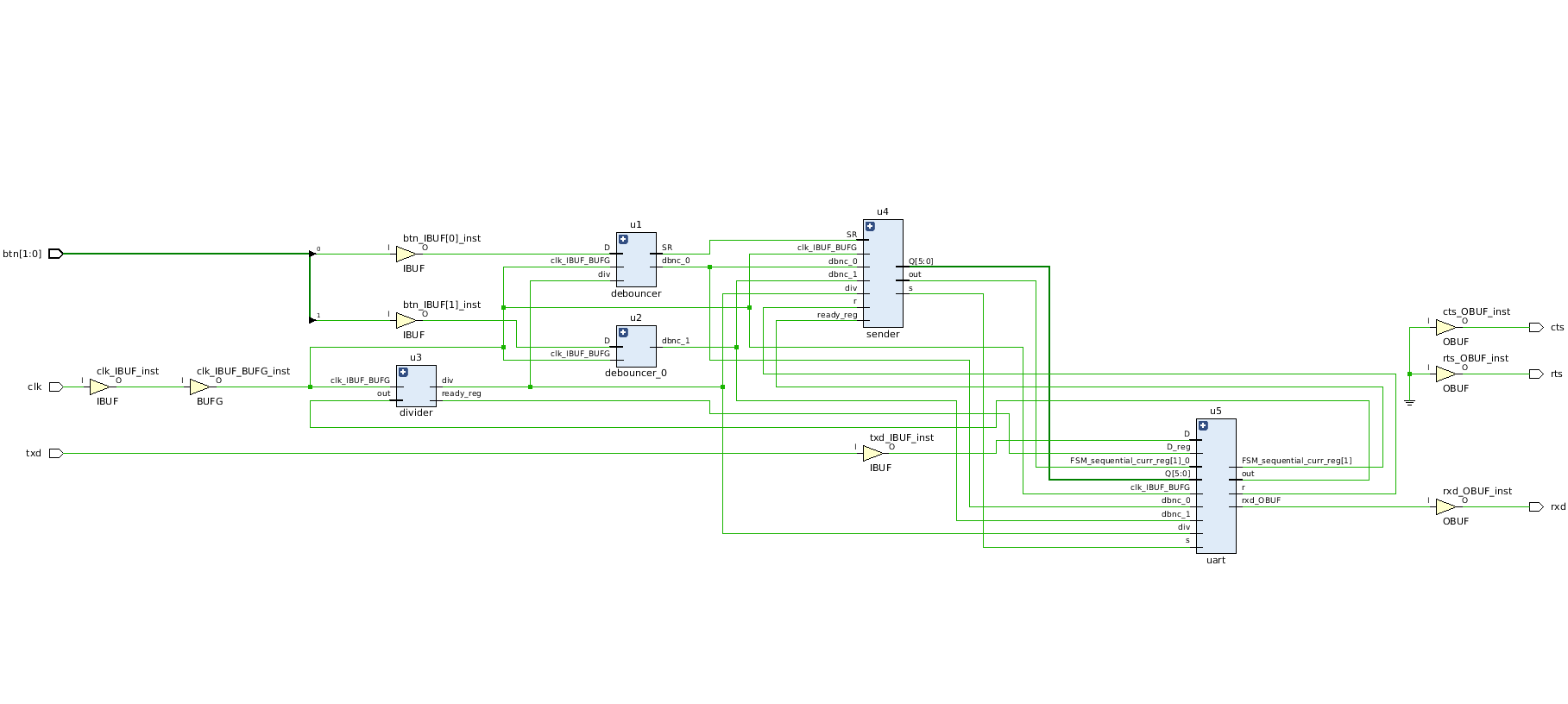




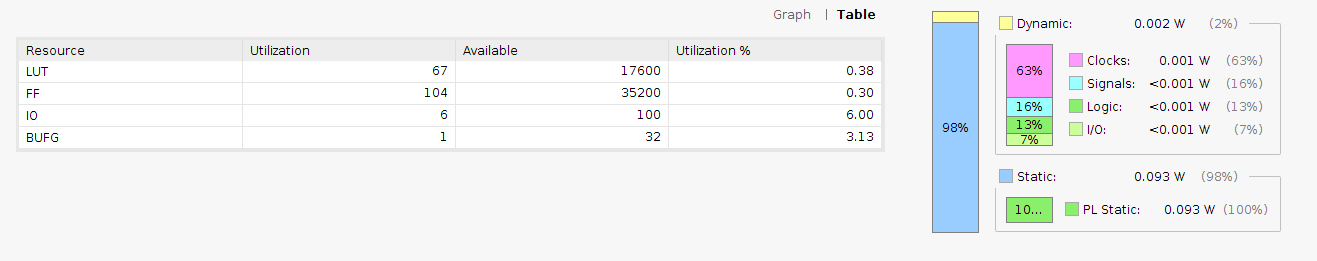
Elaborated Schematic



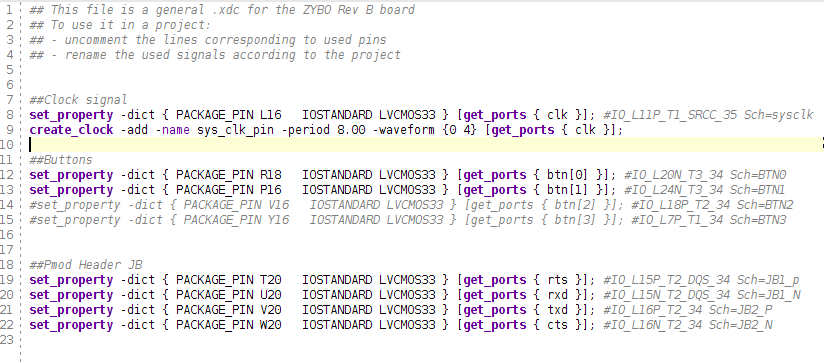
Synthesis Schematic



On-Chip Power and Utilization



Constraint File



Discussion

The lab was very useful in understanding Finite State Machines. This system was useful for understanding how FSM’s can be utilized and how they work. It also gave me an idea of how UART’s transfer information bit by bit. It also helped understand how to use a PMOD and write a constraint file for it. I feel as I completely understand FSM and how they work. However, there is still some confusion with UART’s. For example I created the extra credit echo program, but was unsure how it fit into the top level design. Also I am still uncomfortable with the debugging process. I had a bug with my top level implementation and it took me a very long time to figure out what was wrong.