Project 2

# Objectives

This project is designed to pull together procedural modeling and timing practices that were introduced within the course. A magnitude comparator, modeled after the 74HC/HCT85, was designed within a timing system in order to analyze the delays implemented.

# Design Process (modeling comparator and counter)

The main component of this project is a model following the HC85 chip. This is a four-bit magnitude comparator that uses both the magnitude of two 4-bit inputs, but it also uses two cascading inputs. Based on the documentation provided, I decided an if, else-if, else structure was optimal. The if-else structure compares the two 4-bit inputs. The special case, where the two inputs are equal, the cascading input comes into the picture. Within the equal input branch, a case structure was used to evaluate the cascading inputs. The figures below include several inputs and the output of the HC85 module in their correct operation. The test bench used to simulate the comparator used both a clock instance and an instance of the counter, detailed below, to provide an both four-bit inputs by using the eleven-bits of the counter’s output. The test bench then cycled through all possible comparisons; however, only certain values are detailed in the figures below. The module was verified by comparing the given inputs showcased by hand and verifying their validity.

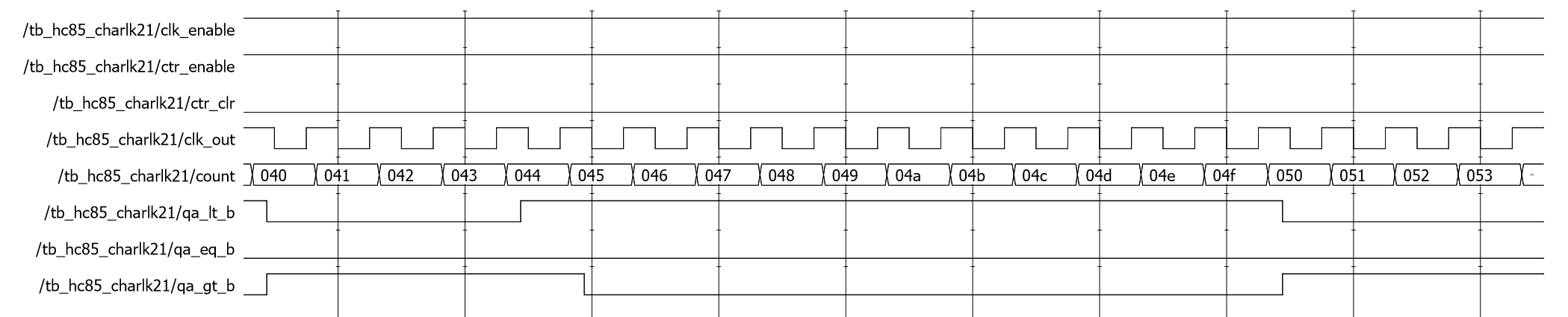


Figure 1. HC74 simulation of greater and less than examples with delays

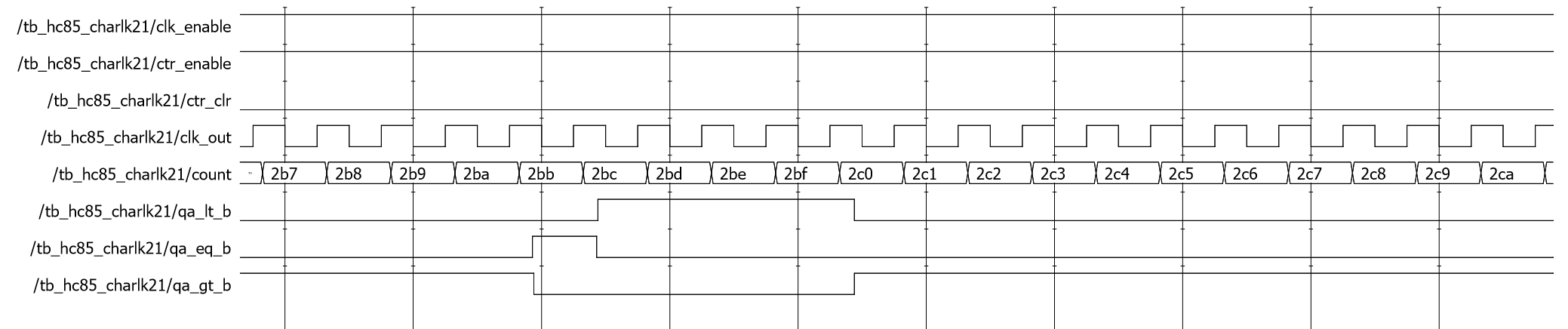


Figure 2. HC85 simulation of equal examples with delays

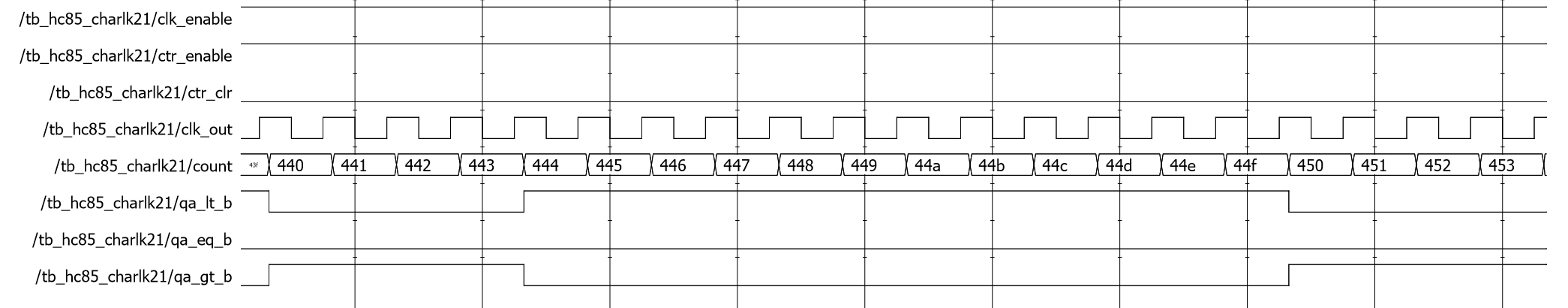


Figure 3. HC74 simulation of greater and less than examples without delays

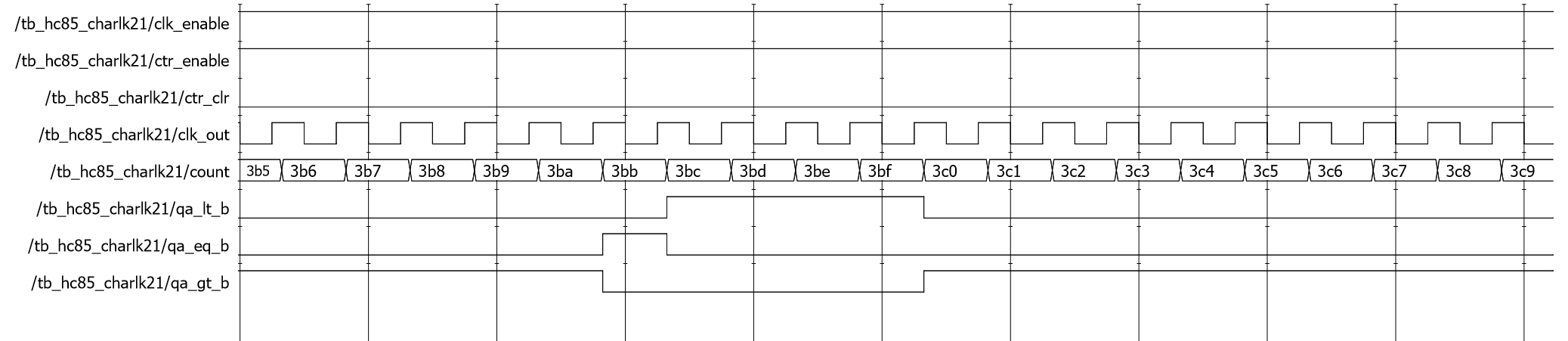


Figure 4. HC85 simulation of equal examples without delays

The 11-bit counter module was modelled on the 4-bit module provided in the project start files. Very little had to be modified to fit the requirements of the module. The most significant modification was the max value; this was changed to 2047, equivalent to (2^11)-1, compared to the previous (2^4)-1 for the four-bit counter. You may notice that the magnitude that the max value is raised to is directly correlated to the number of bits of the counter. Included below are two snapshots of the counter’s output. The test bench used to simulate the counter used a clock instance to provide and input to the counter, and then cycled through several states with the counter’s enable signal and its clear signal.

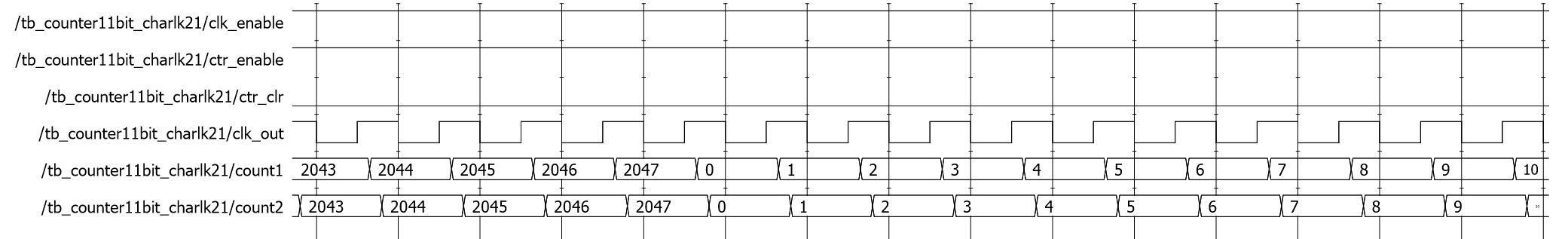


Figure 5. 11-bit counter waveform, max value and rollover

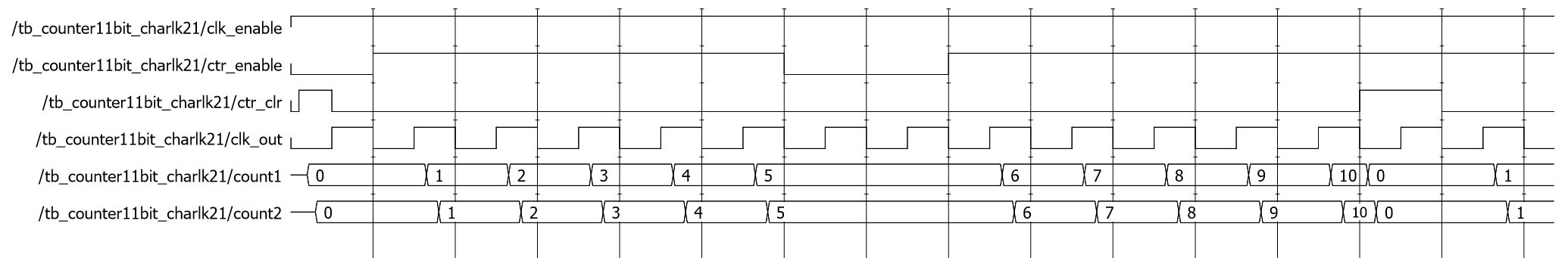


Figure 6. 11-bit counter waveform, clear and enable

The 14-bit register was the final module within the first level of the design. The register is required to load the 14-bit input into its output on the positive edge of a clock input. There a procedural approach was taken to monitor the clock input on the sensitivity list. When the clock triggered within the register the output of the register was loaded with the input. The test bench used to simulate the register was a simple counter and various input values that were used to load values into the register. Include below is a figure of the waveform of a 14-bit register. The implementation was correct if the values were loaded on the positive edge of the clock.

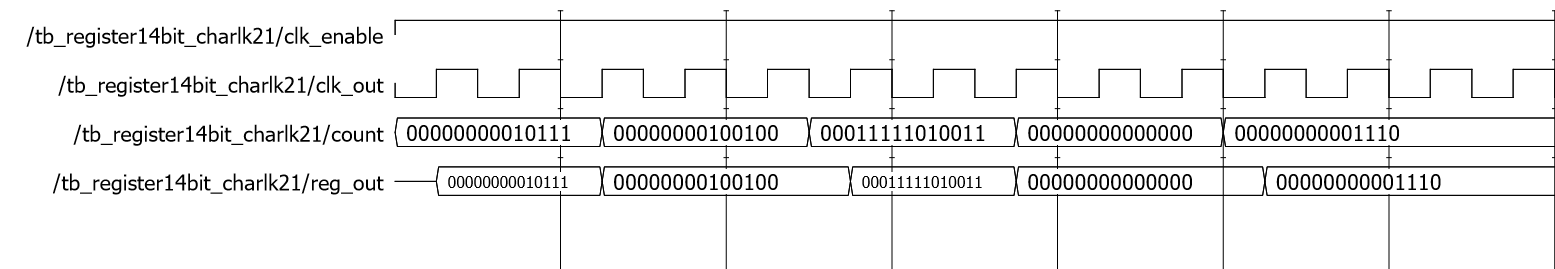


Figure 7. 14-bit register waveform

# Period Analysis

With the initial modules of the project designed, the pieces came together for two higher level modules that were tested within two different test benches. The test benches were designed to test the timing within the different lower level modules. The test bench for system one used a clock cycle of 100ns, longer than the sum of all components’ delays, to show the correct operation of the system. Figure 8 and 9 below shows two cases of correct operation. Both cases show the delay inherit within the system, but they show the system returning to a valid state by the next clock cycle.

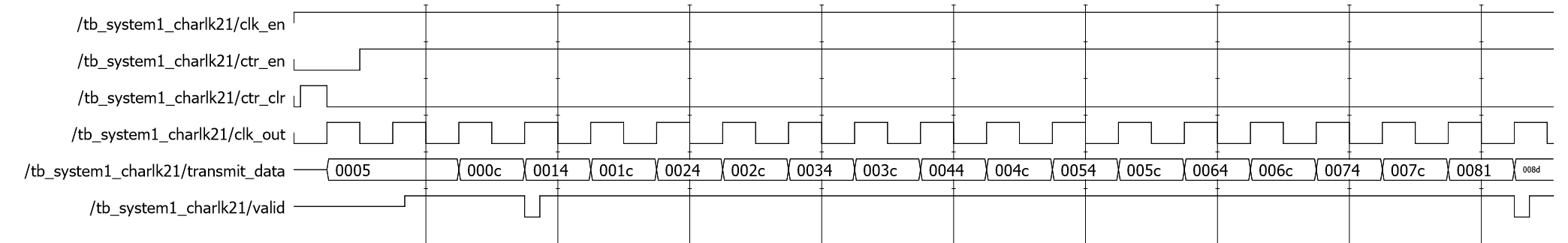


Figure 8. System 1 waveform, includes cycle with clear of counter

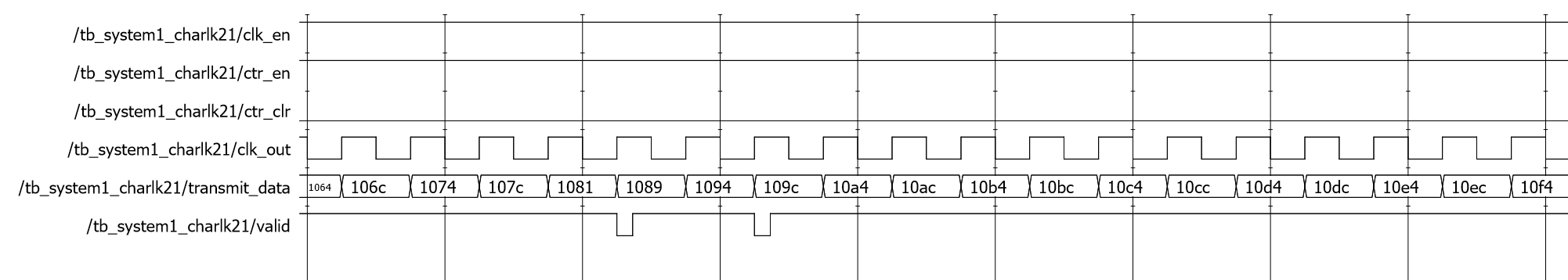


Figure 9. System One waveform, higher value input results

Once the system was validated using a arbitrarily large clock period, the system must go down! The next calculation was to determine a clock period that was shorter than was operable for a given section of the system. The longest possible delay for the system is within the transmit module, from the counter to the output of the comparator instance. The longest transition for the counter and the comparator is 15ns and 23ns respectively. This means that a period just shy of 48ns would prevent the system from operating properly, the comparator’s output goes out of synch with the counter when loaded into the register. Included in Figure 10 is a point in the second system where the output remains high past the near clock cycle, drops to be invalid, and never reaches a active state before the clock cycle reaches another rising edge. The clock cycle for this simulation is set to 37.5ns, but a clock cycle greater than 38ns would allow for proper functionality.

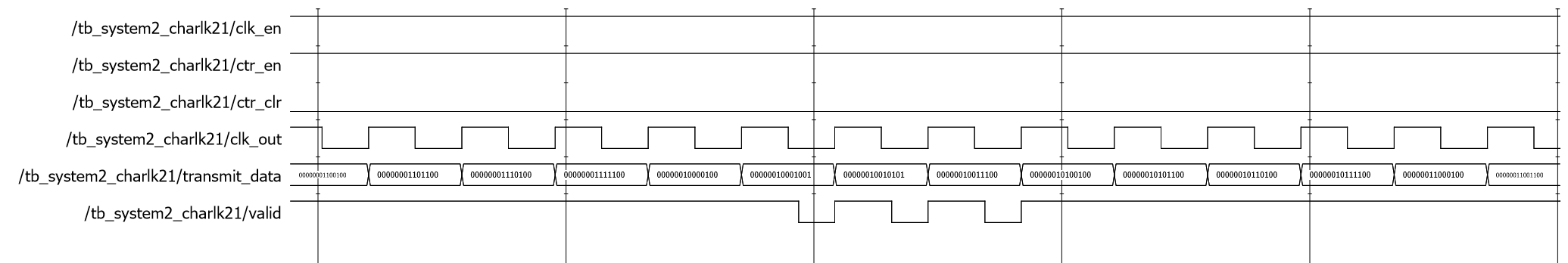


Figure 10. System Two waveform, failure of module by timing error

# Conclusion

In conclusion, the timing of a system has infinite importance. Delays in a circuit that are as small as 15ns can influence the functionality of a circuit – especially when components cascade. This showed up in as few as two modules, the counter and the comparator, and one could only imagine larger systems and the consideration that must be given to reducing delays. Another takeaway from the project was the syntax used to mimic hardware delays within modules. It was important to see the delays within the simulations, but important to learn that they lack synthesizability. To end on a single note – do not ignore delays.