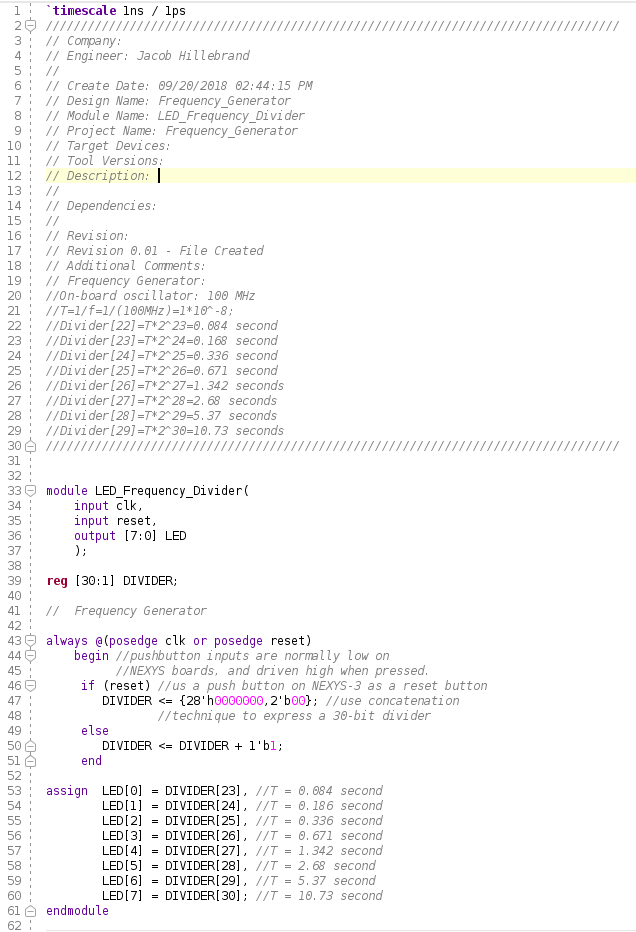
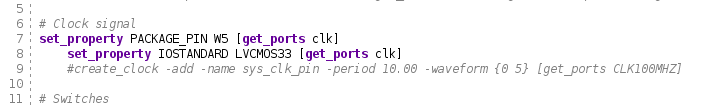
Lab 4 – Digital System Design Name: Jacob Hillebrand

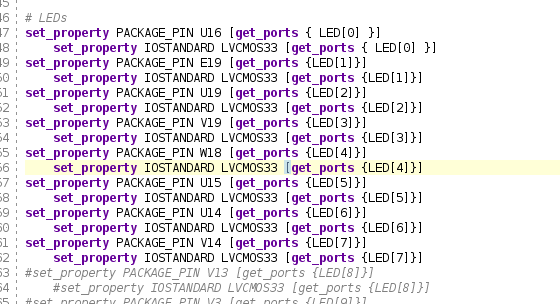
September 29, 2018

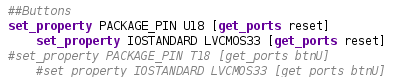
Lab 4 was a very lengthy lab, consisting of three separate subprojects. Each subproject was given a separate project in Vivado and used separate design files. The idea behind three separate subprojects was to use each of them to expand on previously explored labs, and provide additional functionality and insight to each. Additionally, the concept of a clock divider, which was heavily utilized in this lab, is slightly complex and much easier to understand when broken into smaller projects.

The first subproject was called LED Frequency Divider. This design was rather simple, and involved the small design and constraint files below.



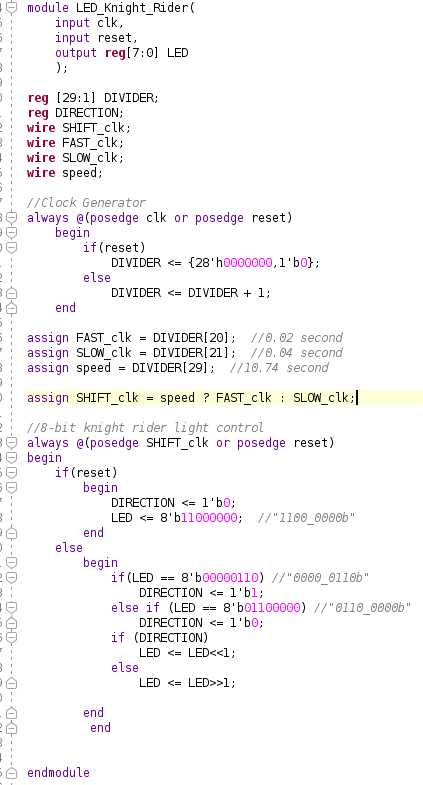






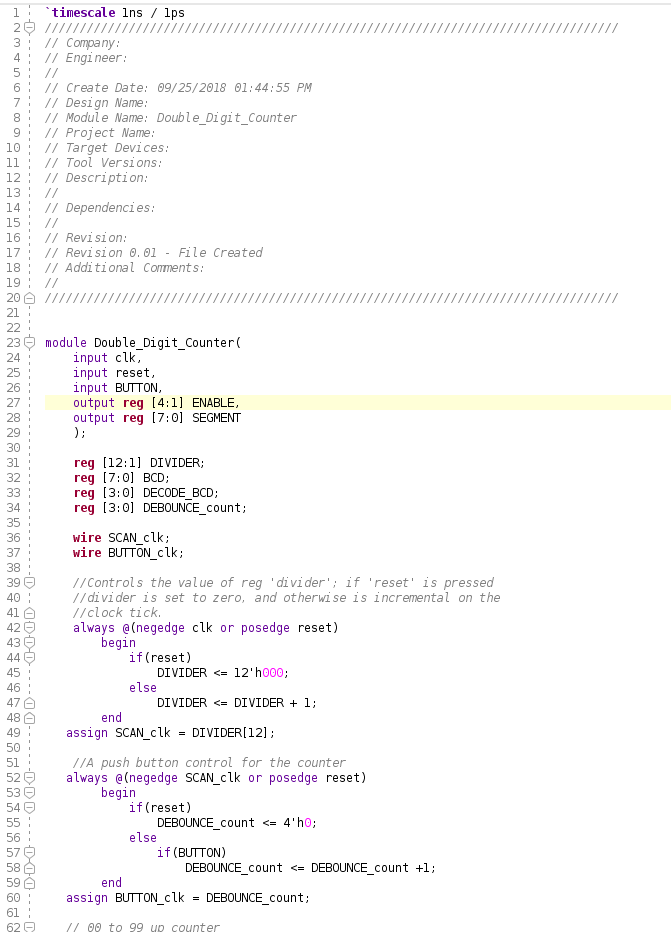
The design file simply took input from the clock on the FPGA, performed a division on it to reduce the number of cycles per second, and blinked up to 8 LEDs in increasing sequence. Each new LED lighting up was a display of the “tick” of the pseudo-clock that had been created with the divider, and allowed us to visually see this clock in action. The LEDs represented 8 bits, and they simply counted up from 1:256, incrementing with each clock tick.

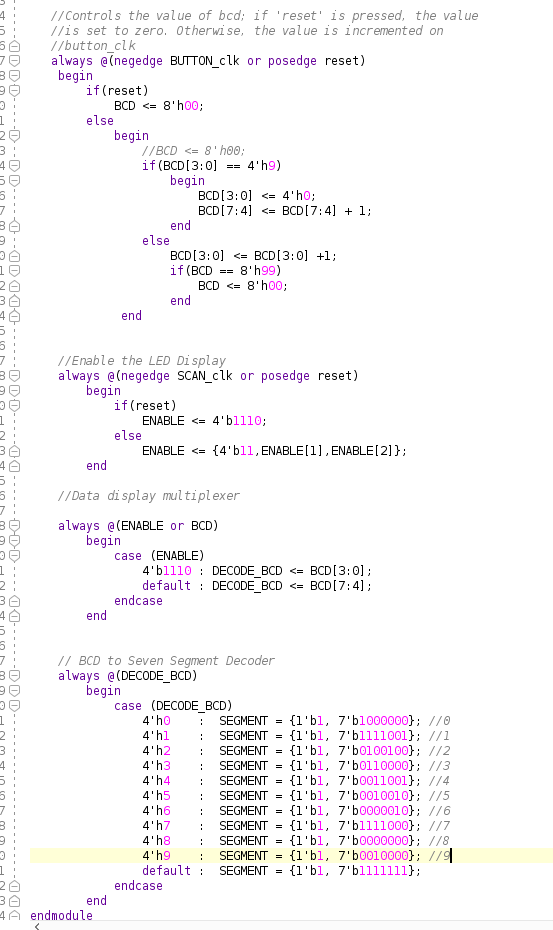
The second subproject was called Knight Rider LED display. This project used the same constraint file as the LED Frequency Divider, but had a slightly more complex design file, as shown below.

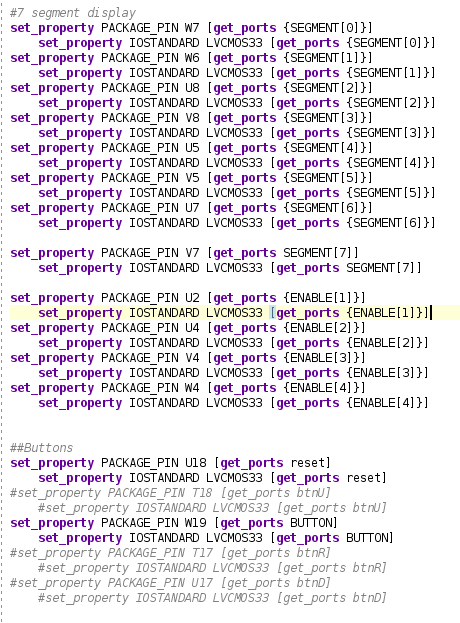
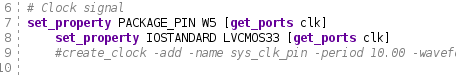


This design was intended to mimic the light display on the bumper of the car from the 1980’s series *Knight Rider*. The car had lights on the bumper that would light up only 1 segment at a time, and change which segment was on in rapid succession. This created an effect that made it look like the lights were bouncing back and forth between the sides of the car. To do this with our Baysis 3 board, we could utilize the same divider concept from the LED Frequency Divider project, but with some additional logic. The same 8 LEDs were used from the LED Frequency Divider project as well, and 2 LED’s were assigned to be on at any given time (with the LED register). As the clock “ticked”, the two “on” LEDs would be shifted either left or right, depending on whether they were currently set to be moving left across the LED row (as they had hit the right side) or right across the LED row (as they had hit the left side). The result was a “bouncing” effect of the LEDs rapidly moving from one side of the FPGA to the other. Additionally, two clock dividers were implemented, so the LEDs would move at a slow speed for ~5 seconds, then a fast speed for ~5, and repeat.

The third and final subproject was called Double\_Digit\_LED. This project was more complex than either of the previous two projects, and was governed by the design and constraint files below.







This subproject again implemented a clock divider like the previous two subprojects, but this time the divider served a slightly different purpose. The design was intended to be a counter from 0:99, and would increment whenever a button was pushed on the FPGA. The value of the count would be stored in the BCD registry, which could be set to increment with an input from BUTTON (the user pushing the button), and could be reset to 0 if the counter hit 99 OR if the user pressed the reset button. However due to the high frequency nature of the FPGA, when the user pushed the button to increment, the single push was often picked up as several pushes. This is known as signal bouncing, as the signal appears to bounce up and down when examined as a waveform. To compensate for this, we attempted to use the clock divider to only register user button pushes in sync with a divided clock cycle. Unfortunately, this was rather ineffective, and the result was a counter that would increment by 10, 20, or even 30 with every single push. We were introduced to the idea of using a finite state machine to “debounce” the input, and will be implementing it in a future design.

Aside from the bouncing of the Double Digit Counter, we were successfully able to complete all three subprojects. As we are still learning to use Vivado and the Baysis 3, there were still a few initial struggles trying to get the constraint files to work. Additionally, typos continue to be a major struggle for several of us. There were multiple times that I and others had to go back through our code, only to find we omitted a crucial chunk of code, or spelled variable names wrong. However, as we perform more labs, we start to understand the process a little better, and everything seems to go much more smoothly with each subsequent lab.