Lab 01
Jonathan Gusler
1/20/2019

Lab01 Pin	DE2 Pin
Clock	50MHz Clk
Resetn	SW[17]
LA	SW[16]
s	SW[15]
Data[7…0]	SW[7…0]
B[3…0]	LEDG[3…0]
B[3…0]	HEX0[6…0]
Done	LEDR[0]

Table 1: Appendix of program signals and their corresponding pins on the DE2. HEX0[6…0] is not used until the second half of the lab, at which point it replaces the usage of LEDG[3...0].

As an introductory lab, the primary goal of this lab is to introduce us the DE2-115 Intel FPGA. This is accomplished by creating a bit counter using step-by-step instructions from the Lab Professor as well as Intel's Quartus guide and the User Manual for the DE2.

The lab is broken into two parts. In the first part, the output is sent to four LEDs that represent binary representation of the result, the second part sends the output to a 7-segment display for decimal representation. A bit counter takes a binary number as input, in this case an 8-bit number, and returns the number of high bits. If the number input is 00001111, the output should be four, as four of the eight bits in the input are one (high).

The bit counter works as follows:

1) The user sets eight switches corresponding the 8-bit number, this is the Data signal
2) The Resetn switch is set low then high to reset the circuit to a clean state
3) The LA (load) switch is set high then low to load the 8-bit number into the shift register
4) The s (start) switch is set high to begin the process
5) The shift-register beings to shift right, and a counter is incremented every time the lowest bit's value is 1.
6) Before each shift operation, the value is checked to be zero or not.
7) If the value is zero, a Done signal is sent to an LED to light up and the output B[3…0] is sent to its four corresponding LEDs
8) This Done state is kept until the user turns s low.

The circuit schematics for both parts are shown in figures 1 and 2. Two timing diagrams, Figure 3 and Figure 4, show the signal values through the whole process using the lab given numbers $3B and $00 respectively. Figure 5 shows the process using an extra test value $FF to show additional proof that the circuit worked as desired.

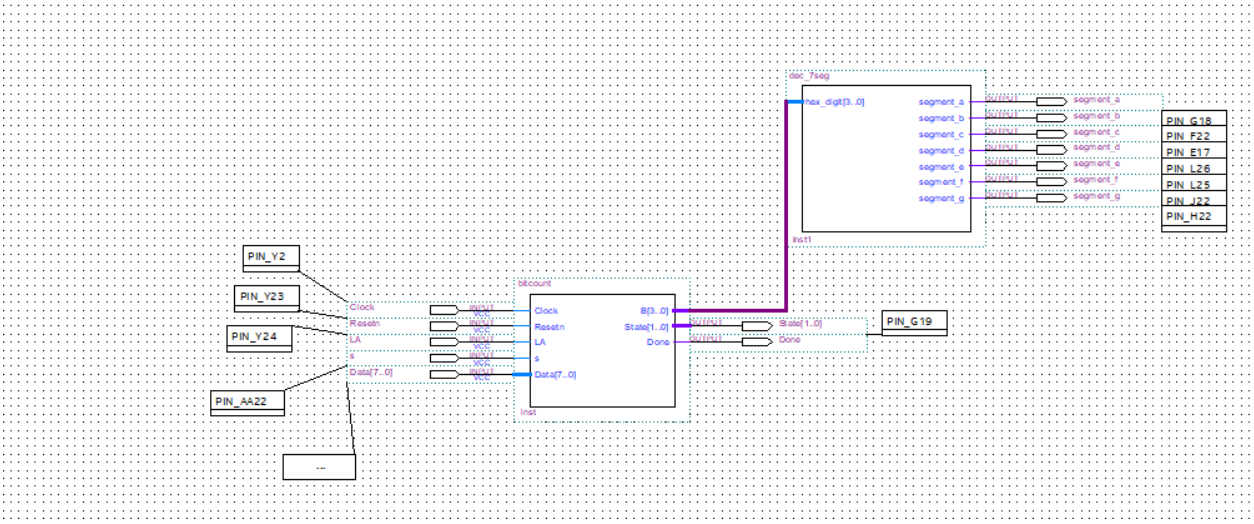

Figure 1: Schematic of the FPGA with the output going to the 7-segment display.

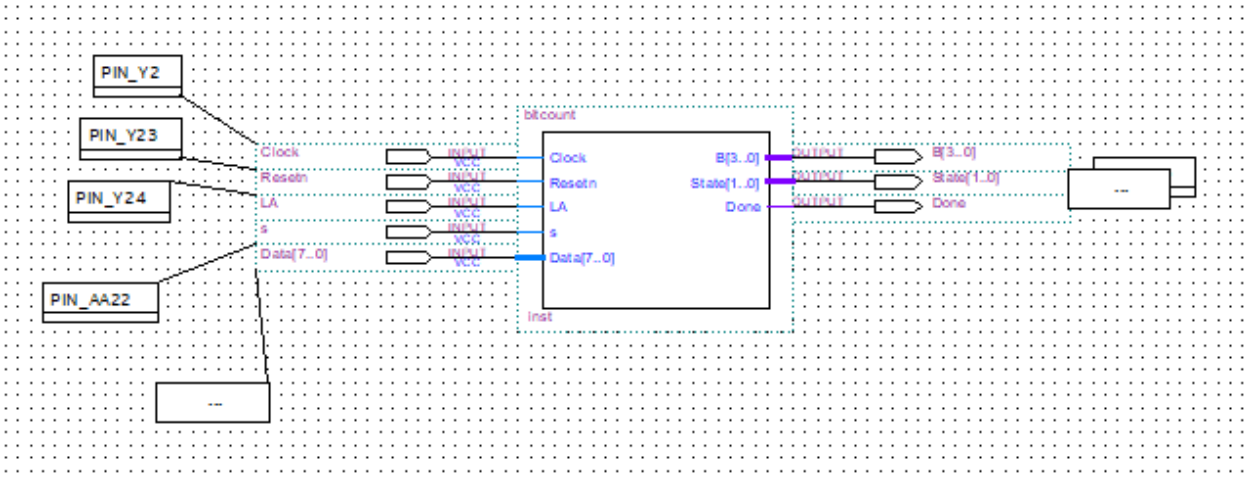

Figure 2: Schematic of the FPGA with the output going to the 4 LEDs.

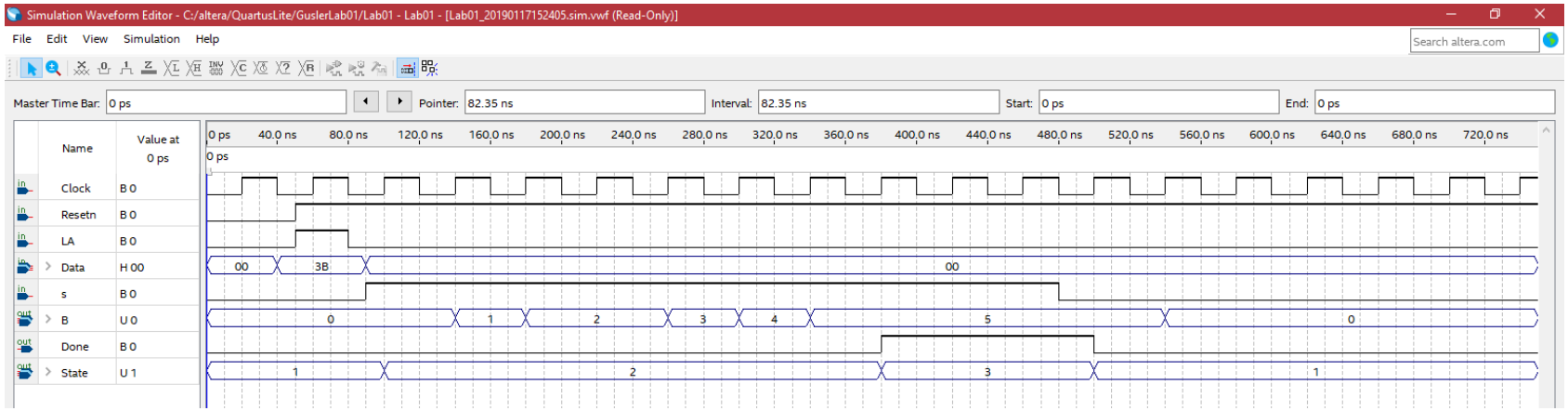

Figure 3: Waveform of Bit Counter with Data set to $3B.

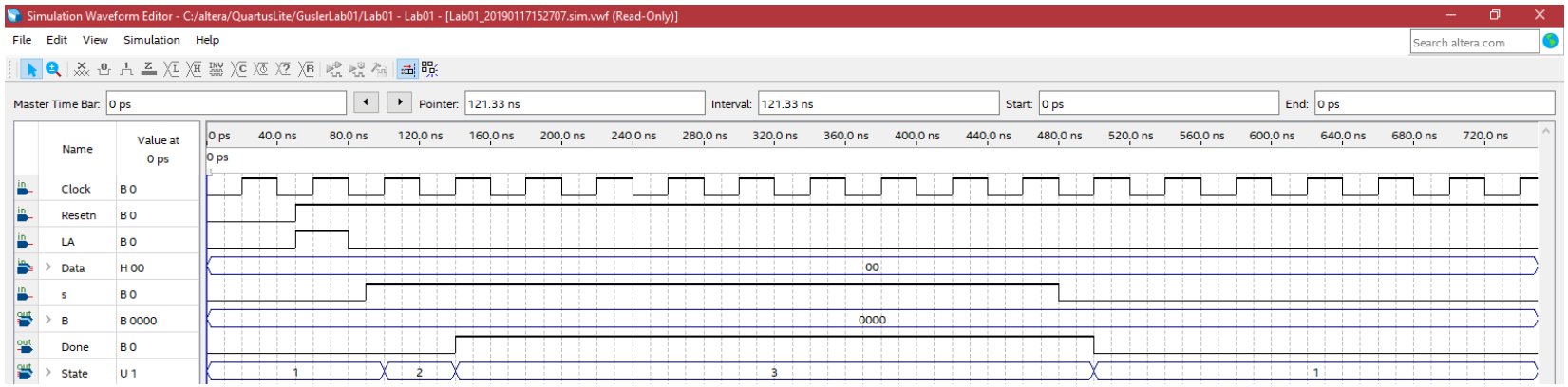

Figure 4: Waveform of Bit Counter with Data set to $00.

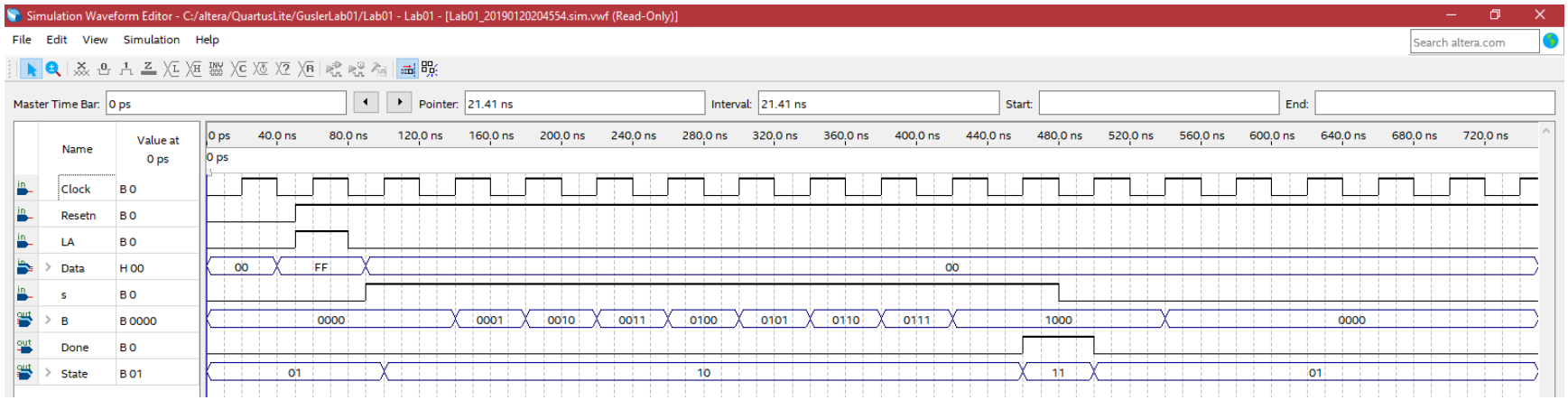

Figure 5: Waveform of Bit counter with data set to $FF.

The first number tested is $3B. Seven clock cycles, seen in Figure 3, are needed to count the high bits. The second number is $00. One clock cycle, seen in Figure 4, is needed in this instance. Some bits take longer to count because there is at least one “0” between it and the previous “1” that was encountered. For an arbitrary 8-bit number, the number of clock cycles needed is eight minus the number of zeros before the first “1”, reading left to right. In the case of $00, the transition from S1 to S3 takes only one clock cycle because the first shift doesn’t occur until after the first time the value is read as not equal to zero; the shift doesn’t occur and doesn’t use up a clock cycle. Figures 6 and 7 show images of the FPGA using an input of $FF, Figure 6 showing the FPGA outputting to the 7-segment display and Figure 6 showing the output going to the LEDs.

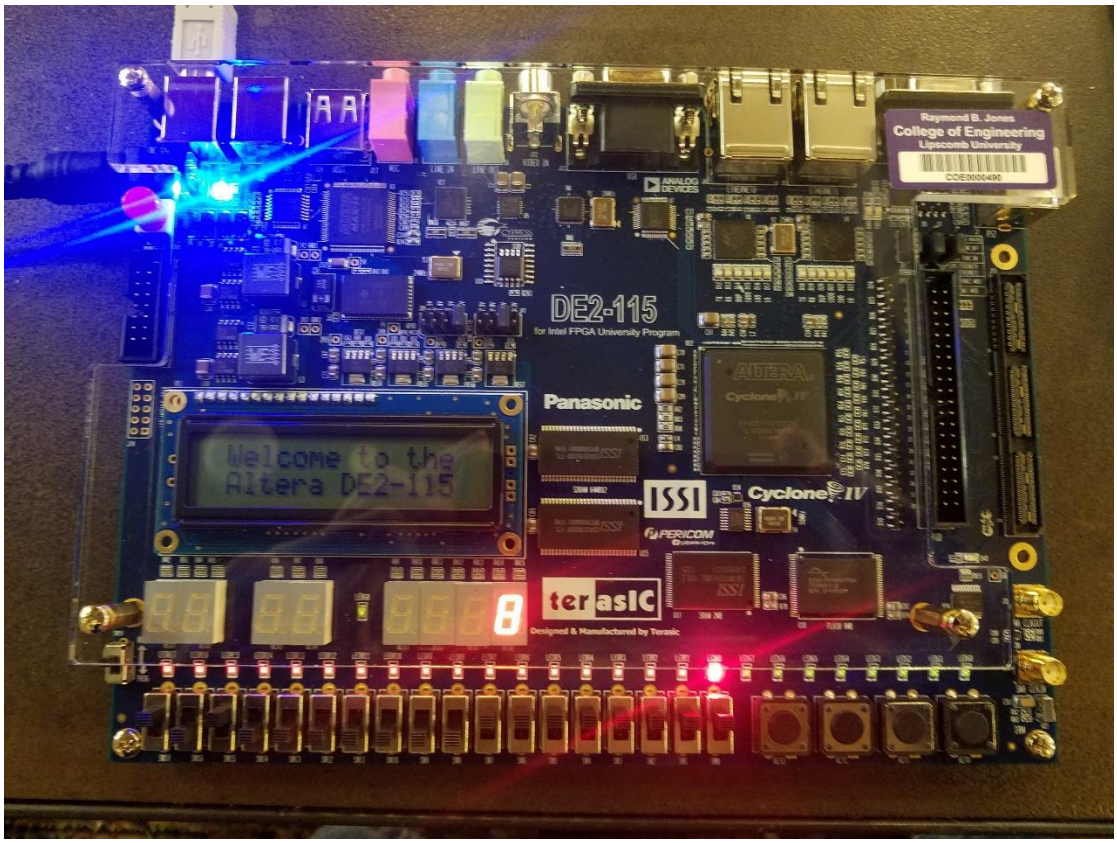

Figure 6: Image of FPGA showing Done signal, an input of $FF, seen in the eight left-most switches, and a correct output of 8 to the 7-segment display.

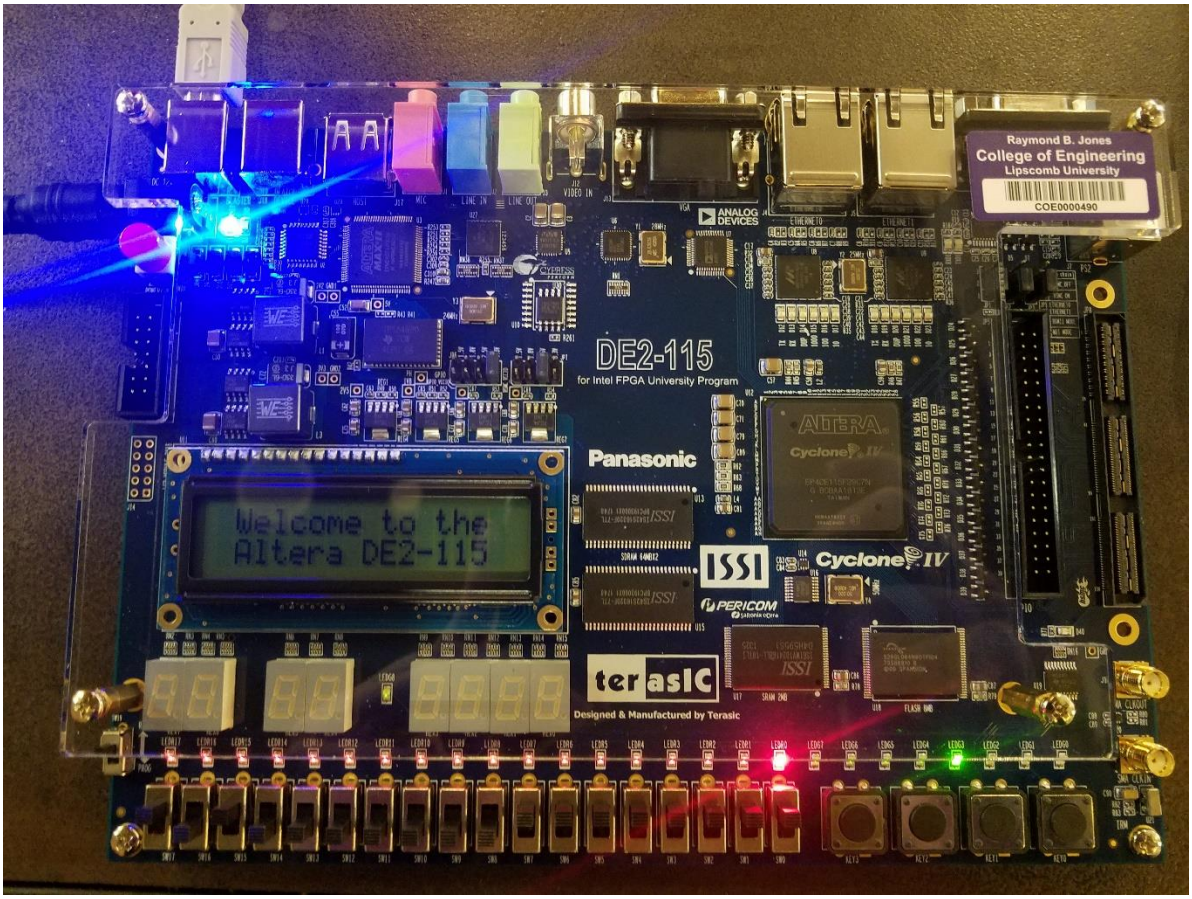

Figure 7: Image of FPGA showing Done signal, an input of $FF, seen in the eight left-most switches, and a correct output of 1000 (binary) to the bottom right green LEDs.