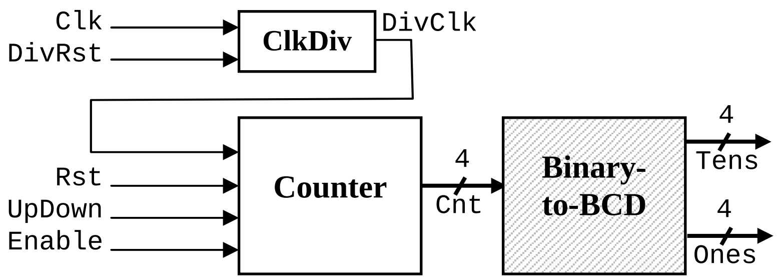
Lab 4: Binary to BCD Converter and Multiplexed BCD Display Driver

In your previous labs, you utilized a binary to 7-segment LED decoder to display a 4-bit binary number as a hexadecimal number. However, as the general public is not likely to be familiar with hexadecimal numbers, we may instead want to display the 4-bit number as a two digit decimal numbers (00 thru 15) using the two 7-segment LED displays. In this lab, you will design a *Binary to Binary Coded Decimal (BCD) Converter* and a *Multiplexed BCD Display Driver* to display the 4-bit output of your *Up/Down Counter* on the two 7-segment displays connected to the Spartan-3E FPGA board.

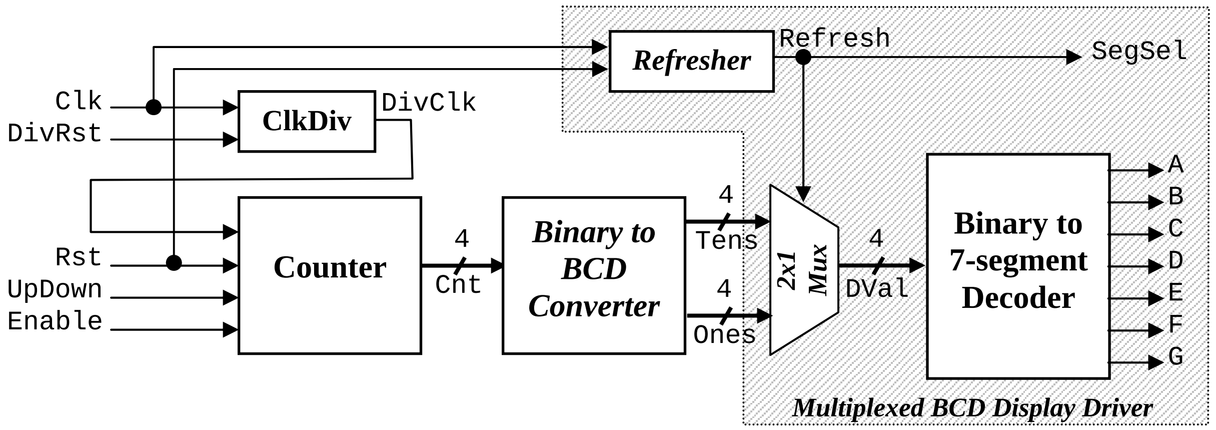
Binary-to-BCD Converter

In order to display the 4-bit binary number in decimal, you will first design a *Binary to BCD Converter*. A BCD number is a number that uses a 4-bit binary number to represent each decimal digit. For example, the binary number `1111` (equal to 15 in decimal), can be encoded as BCD number as `0001 0101`. As such, the *Binary to BCD Converter* has a 4-bit input, `Cnt`, and two 4-bit outputs, `Tens` and `Ones`, corresponding to the binary representation of the tens and ones digit of the decimal equivalent, as illustrated in the following figure.

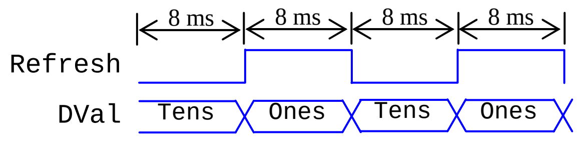


Multiplexed 2-digit BCD Display Controller

In designing the binary to 7-segment LED decoder in Lab 2, the `SegSel` output was used to control which 7-segment LED display would be utilized to display the 4-bit binary number. As such, we cannot simultaneously display a digit on both 7-segment LED displays. Instead, by repeatedly and continuously display a digit on each display faster than the human eye can respond, both displays will appear to be illuminated at the same time. In this lab, you will also design and build a *Multiplexed BCD Display Driver* to display the `Tens` and `Ones` outputs of the *Binary to BCD Converter* on the corresponding 7-segment LED displays. The *Multiplexed BCD Display Driver* builds upon your binary to 7-segment decoder by adding a refresher circuit to control when each 7-segment display will be illuminated and a multiplexer to select between the `Tens` and `Ones` output of the *Binary to BCD Converter*. The following provides an overview of the multiplexed BCD to 7-segment display driver.



For the 7-segment LED display connected to the Spartan-3E FPGA board, each 7-segment LED display should be illuminated for 8 ms, as illustrated in the following timing diagram. Given the 50 MHz clock provided by the Spartan-3E FPGA board, the *Refresher* circuit should generate an oscillating output, `Refresh`, that remains `1` for 8 ms, `0` for 8 ms, and repeats. The `Refresh` output will be used to control the `SegSel` and as the select line to the multiplexer for selecting which BCD digit will be the input to your *Binary to 7-segment LED Decoder*. Please note that you will need to modify your *Binary to 7-segment LED Decoder* to remove the `SegSel` output, as the `SegSel` signal will be connected to the `Refresh` output of the *Refresher* circuit.



To final implementation will utilize the *Up/Down Counter* designed in Lab 3, but replaces the *Binary to 7-segment LED Decoder* the new your previous lab with the *Binary to BCD Converter* and *Multiplexed BCD Display Driver* components built in this lab. As such, when implemented on Spartan-3E FPGA Board, the overall top-level design will have the same inputs and output as the *Up/Down Counter* from Lab 3 and will use the same User Constraint File (.ucf).

Lab Procedure

1. Structurally design the *Binary to BCD Converter* using any of the following datapath components: adders, subtractors, incrementers, decrementers, multipliers, comparators, shifters, registers, multiplexers, decoders, encoders, and logic gates *(only when necessary)*. The above listed datapath components utilized within your design can be modeled behaviorally at a high-level of abstraction. *Please note that you do **not** need to utilize all components listed above, but rather you are restricted to those components.* Create a testbench to test the *Binary to BCD Converter* for correct functionality.

2. Structurally design the *Multiplexed BCD Display Driver* and *Refresher* sub-component using any of the above listed datapath components. *Again, please note that you do **not** need to utilize all components listed above, but rather you are restricted to those components.* Create a testbench to test the *Multiplexed BCD Display Driver* for correct functionality for one full 16 ms refresh period.

3. Create a top-level component that structurally connects your *4-bit Up/Down Counter*, the previously provided clock divider (*ClkDiv)*, the *Binary to BCD Converter*, and the *Multiplexed BCD Display Driver*. Download and test your design on the Spartan-3E FPGA board for correct functionality.

Demo *(you must demo the following aspects to the TA)*

1. Structural Verilog code for *Binary to BCD Converter* design and behavioral Verilog code for any datapath components utilized. Testbench and simulation waveforms demonstrating correct functionality of the *Binary to BCD Converter*.
2. Structural Verilog code for *Multiplexed BCD Display Driver* design and behavioral Verilog code for any additional datapath components not utilized within the *Binary to BCD Converter*. Testbench and simulation waveforms demonstrating correct functionality of the *Multiplexed BCD Display Driver* for one 16 ms refresh period.
3. Structural Verilog code for top-level component integrating *Binary to BCD Converter* and *Multiplexed BCD Display Driver* with your previous *4-bit Up/Down Counter*. Synthesis and implementation of your overall design to Spartan-3E FPGA Board demonstrating correct functionality.

Lab Report Requirements *(In addition to the standard lab report format)*

1. All structural and behavioral Verilog code for *Binary to BCD Converter*, *Multiplexed BCD Display Driver*, *4-bit Up/Down Counter*, modified *Binary to 7-segment Decoder*, top-level design component, and all datapath components utilized.
2. Simulation waveforms demonstrating correct functionality of the *Binary to BCD Converter*.

3. Schematic diagram for the *Binary to BCD Converter*, *Multiplexed BCD Display Driver*, and *Refresher* components clearly indicating the various datapath components utilized and their interconnections.
4. Each group should submit all of their structural and behavioral Verilog code to the appropriate dropbox on the D2L course website by Nov 9, 2007. Your online code submission will count for 5 points of your lab report score. Only one submission is required per group, but you must indentify yourself and your lab partner within comments at the top of ***all*** your Verilog files.