part1_description

****

Victor Qin and Mohib Jafri
CS 141 Spring 2019
3/1/19

PA 3 Pt. 1 Description

****

In this PA we created a 1Hz clock and a timer along with their constituent testbenches.

Timer:
The timer is very similar in spirit to the universal binary counter. We implemented an async reset to catch any presses of reset (the rest is synchronous for simplicity's sake). In where we have reset, the most important block, it is nonblocking and will be asserted directly to out_reg at the end of the clock cycle. On the other hand, any other input will be stored in out_next, and out_next will only propagate to out_reg at the very end. This design allows for very clean designs and ease of establishing hierarchy.

To test timer, there were two primary test cases necessary to sufficiently prove we met design specs. First, a normal countdown. We load a value into init and set load high, then enable high and watch 1 sec intervals count down our value. Notice it also STOPS at 0. This confirms the core functionality. Next, we test different permutations of inputs to the system: all three en, rst, and load, then just en and load, then ~en (pause). This encompasses most if not all edge cases to look for.

1 Hz Clock:
The 1Hz clock runs by taking in the buffered clock provided by the starter code and counting based on it. On a clockedge or reset button press, the always block activates. It checks if reset is pressed - if so the internal counter r_count and the output clock clk_out both go low. After 50,000,000 - 1 clock cycles, the clk_out signal is flipped and r_count is reset (r_count = 0 serves as the first clock cycle). If neither of those states are reached, r_count is incremented. All signals are nonblocking, and because the logic is all in one if/else block there will only be one action executed at the end of the loop.

The testbench briefly resets the system and then feeds the clk_divider module a infinite clock running at 100 MHz.