Lab 7, Counters

These lab problems will make use of:

- “always” statements
- “register” datatypes (essentially, a latch)
- counters
- the system clock
- reuse of old modules

For each problem, in your journal, please describe:

- the problem you’re solving (in your own words)
- a description of your solution (in words)
- a description of your solution in block-diagram form (to support the words)
- a description of any novel structures in verilog that you need to use
- Any problems/bugs, etc that remain in your solution

Lab Problems

1. Using the system clock (@ about 100MHz, depending on board), create a counter that blinks one of the board’s LEDs at **two** second on, **one** second off resolution (so, a period of 3 seconds total). Blink a second LED at **1ms** on, **2ms** off.

2. Send the LED output from problem 1 to an oscilloscope (via a pmod pin) and check your work – to what accuracy are you creating the specified signal?

3. Use the system clock to make 8 LED’s count up the (binary) time in seconds since the board was powered on. In your journal, make it clear what the period of this display is (before it overflows and starts over). Note, you need to be able to describe this overflow by resetting the appropriate counter.

4. Using the (human) persistence of vision (POV), display the number 1234 on the 7-segment LED display. This isn’t as hard as it may seem if you reuse your work from the previous lab. Extra fun: how can you make one digit brighter than the rest?

Do 5 or 6

5. Again, using POV, map 6 switches to two 7-segment displays so that the binary number represented by the switches is displayed in base 16 on the display. For example, if the switch input is 01_0101, the 7-segment display should read 15 (0x15). If the switch input is 10_1111, the 7-segment display should display 2f (0x2f). If you didn’t build a hex display driver for the 7-segment display (lab 6) yet, this module could be useful https://github.com/ntmoore/phys332_fall21/blob/main/lab08_problem5.v

6. Using counters, POV, and switches to control the color output from the RGB LED’s such that: (a) The LED’s aren’t blindingly bright and (b) the color displayed is modified when the user toggles the switches. One way to do this is to have the max brightness of any

color be 50% duty and then have 3 banks of 4 switches map to the brightness of an individual color. Eg, if SW[3:1]=1111, then red would be at 50% brightness, SW[3:1]=1000 would be ~25% brightness, etc.

For each problem, please share a short youtube video that shows the module working on your board. Also, please turn in (scanned) copies of the work in your journals and listings or github links of/to the verilog code you used for each problem.