Bounce counter for high-speed detection of switch bouncing

by: Alex Mazarakis (suvervised by William Brooks)

FPGA-based bounce counter for high-speed detection of switch bouncing as prescribed in the lab 6 (week of 2016-10-10 – 2016-10-14).

This code takes advantage of the core generation utility in Xilinx's suite to construct a digital clock multiplier (likely through a phase-locked loop) to boost the onboard clock from 8 MHz to 32 MHz.

```
"module multiplier(
            input clock,
            output multiple
   );

  timing instance_name
   (// Clock in ports
    .CLK_IN1(clock),      // 8Mhz clock
    // Clock out ports
    .CLK_OUT1(multiple));    // 32 Mhz output
endmodule"
```

The bounce counter itself is implemented as a simple, synchronous edge detector with incrementation.

Adjusting for operation of flip-flop on a single edge of the 32 MHz clock, and Nyquist sampling criterion, the system should be sensitive to bounces of width T = 62.5 ns or greater (F = 16 MHz).

```
“module bounce_counter(
        output [7:0] count, //the number of bounces observed
        input switch, //input from the switch
        input clock, //32 Mhz clock
        input rst //sigal to reset the count to 0
   );


        reg [8:0] transitions = 9'h4; // times switch transitions states
        reg level; //switch's initial position

        //reset the counter to initial position and check initial level
        always @(negedge clock)
        begin
                if (rst)
                begin
                        transitions <= 9'h0;
                        level <= switch;
                end else if (level != switch)
                begin
                        transitions <= transitions + 1;
                        level <= switch;
                end
        end

        assign count = transitions [8:1]; /** Implicit Division by 2 to account for both edges of a
bounce */

endmodule”
```

The exact wave form is not communicated to the end-user, but a non-zero count of bounces is produced consistent with sampling frequencies of modern microcontrollers.

As a secondary advantage, the FPGA based implementation is polling at 16 MHz (Responsive to 8 MHz or slower signals) making it slightly faster than the logic analyzer’s maximum of 5 MHz with a more robust and easily usable interface (e.g. not Acute LA Viewer.)