Management and analysis of physics dataset: FPGA Stopwatch (modulo 16)

Rocco Ardino	Alessandro Lambertini	Alice Pagano	Michele Puppin
1231629	1121181	1236916	1227474

Monday 13th January, 2020

1 Aim

The purpose of the assignment is to implement a 4-bit stopwatch, namely a counter, with the following functionalities:

- **START**: it enables the counting.
- **STOP**: it stops the counting.
- **RESET**: it resets the counting.
- **FREQUENCY SELECTOR**: it can change the frequency of the counting.
- **REVERSE SELECTOR**: it makes the stopwatch counting in reverse.

2 Implementation

The Arty7 board has 4 LEDs that could be used as a display counter ($0 \rightarrow 15$). Indeed, each LED is associated to a bit: an off LED corresponds to the bit state '0', while a blinking one corresponds to the bit state '1'. The time flow is regulated by the embedded clock of the board, which is used to implement the **counter**. Concerning the functionalities, **START**, **STOP** and **RESET** can be triggered by the embedded buttons of the board, while **FREQUENCY** and **REVERSE SELECTOR**s by the four switches. In particular, three switches are used for modulating the frequency of the counting and one is used for reversing it. The actual disposition of functions is illustrated in Figure 1.

Figure 1: Description.

```
p_cnt : process(clk,rst,sel_in) is
    begin
    if rst = '1' then
        counter <= (others => '0');
    end if;
    if rising_edge(clk) then
        counter <= counter +1;
    end if;
end process;
```

Figure 2

2.1 Counter

The code implementation is constituted by four main processes. The first two processes (`p_cnt` and `p_slw_cnt`) are used to implement the counter.

Since the speed of the embedded clock is too fast, first it has been used to increase the value of a vector of 28 elements (`counter`) each time the clock signal shows a rising edge, as in process `p_cnt`.

```
p_slw_cnt : process(clk,rst,frz,slow_clk,sel_in,state) is
    begin
    if rst = '1' then
        slow_counter <= (others => '0');
    end if;
    if rising_edge(clk) then
        slow_clk_p <= slow_clk;
        if state = '0' then
            if slow_clk = '1' and slow_clk_p = '0' then
                if sel_in(0) = '0' then
                    slow_counter <= slow_counter + 1;
                elsif sel_in(0) = '1' then
                    slow_counter <= slow_counter - 1;
                end if;
            end if;
        end if;
    end if;
end process;
```

Figure 3

Then it has been slowed down in process `p_swl_cnt` by taking the most significant bit

2.2 START, STOP and RESET

2.3 FREQUENCY and REVERSE SELECTOR

3 Simulation

```
speed : process(clk,rst,slow_clk,sel) is
    begin
    case sel is
        when "000" => slow_clk <= counter(27);
        when "001" => slow_clk <= counter(26);
        when "010" => slow_clk <= counter(25);
        when "011" => slow_clk <= counter(24);
        when "100" => slow_clk <= counter(23);
        when "101" => slow_clk <= counter(22);
        when "110" => slow_clk <= counter(21);
        when "111" => slow_clk <= counter(20);
        when others => null;
    end case;
end process;
```

Figure 4