Digital Systems Electronics Laboratory 04

Group 13

s295391 Giorgio Zoccatelli
s295567 Vittorio Macripò

Due Date:	April 22, 2024
Delivery date:	March 21, 2024
Instructor:	Professor Guido Masera

Politecnico di Torino
Accademic Year 2023/24

Contents

1 Introduction

The aim of this laboratory activity is to get familiar with memory elements such as latches, flip-flops, registers and counters. The core of most exercises is based on the development of counters with different purposes ant techniques.

2 Gated SR latch

2.1 Design Entry

Following the description of the SR latch provided in the document, files *sr_latch.vhd* and *sr_latch_tb.vhd* are created. To verify the correct functioning of the implementation, we should take into account the following signals:

1. **CK clock**, which is scanned on 4 cycles and at each rising edge of the clock allows the action imposed by the set or reset commands on the output;
2. **R reset**, which if it takes the logical value '1' returns the output to '0';
3. **S set**, which if it takes the logical value '1' allows the transmission of the output signal;
4. **Q output**.

2.2 Functional Simulation

The testbench *sr_latch_tb.vhd* is designed to check the operation of the set and reset signals and their impact on the output signal.

2.3 Synthesis

As visible from the image below, the behavior of the RS latch is the same as we expected to be.

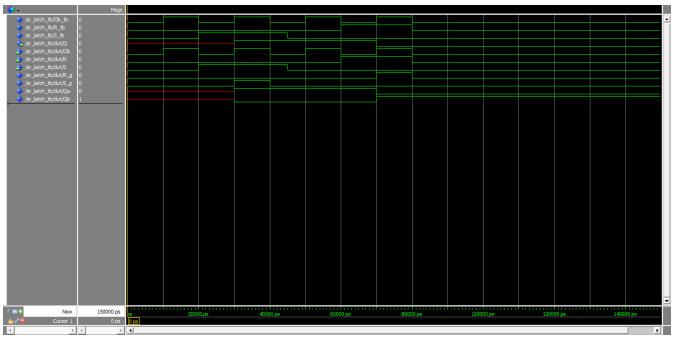

Figure 1: Waveform analysis of the SR latch

By observing the Figure 2, we can also verify that the implemented circuit reflects the expectations.

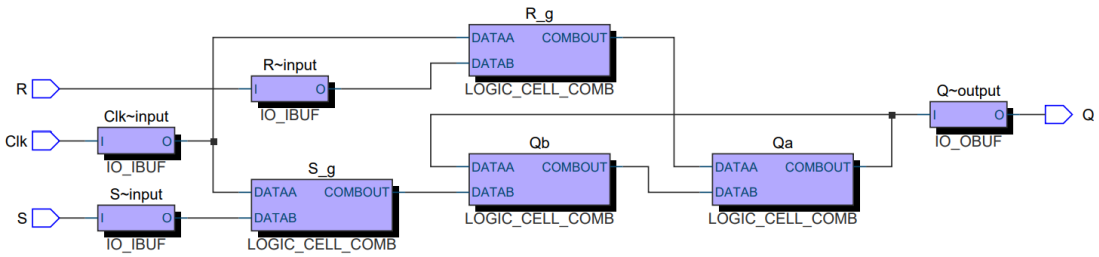

Figure 2: Rtl Map Viewer - SR latch

3 16-bit synchronous counter

3.1 Design Entry

A 16-bit synchronous counter allows you to count up to $2^n - 1$ by using the working principle of the 16 T-type flip-flops joined together via 15 AND gates. Thanks to the enable and reset signals we can also adjust the counting trend, stopping and restarting respectively.
As mentioned before, the circuit structure requires the description of the top-entity *counter.vhd* and of different components as in the case of *t_type_ff.vhd* as well as the description of the components dedicated to the interface with the board, such as *display_counter.vhd* and *regn.vhd*.

3.2 Functional Simulation

The description of the testbench (*counter_tb.vhd*) is relatively simple since it is enough to mark the count through the different clock cycles (**KEY0_tb**) and to manipulate the trend through the reset signals (**SW0_tb**) and enable (**SW1_tb**). Output Q is increased by 1 in each cycle. For optimization reasons, the implementation stops its count at 15, which can be represented in hexadecimal as F by using only one display (**HEX0**).

3.3 Synthesis

The counter behavior is correct, as we can see from the Figure 3. Note that the actual increase in count is always followed by a subsequent representation of the number on the HEX0 display during the next rising edge of the clock. The maximum frequency f_{max} has a value of 282,65 MHz and the total number of logical elements used is 22.

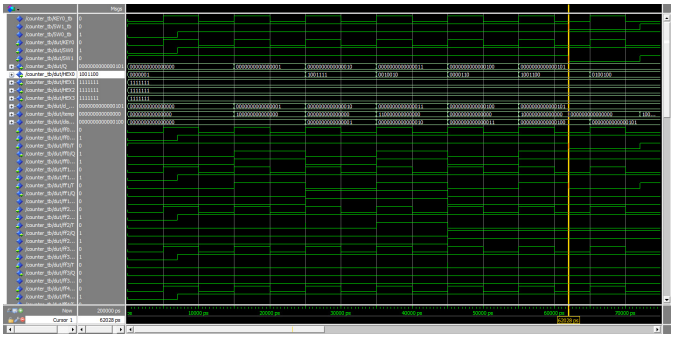

Figure 3: Waveform analysis of the 16-bit synchronous counter

3.4 4-bit synchronous counter

After the realization of the 4-bit version of the circuit (*counter.vhd*), we can look at the RTL Viewer and realize that the configuration obtained is the same as the one we were expecting without any significant difference. See Figure 4.

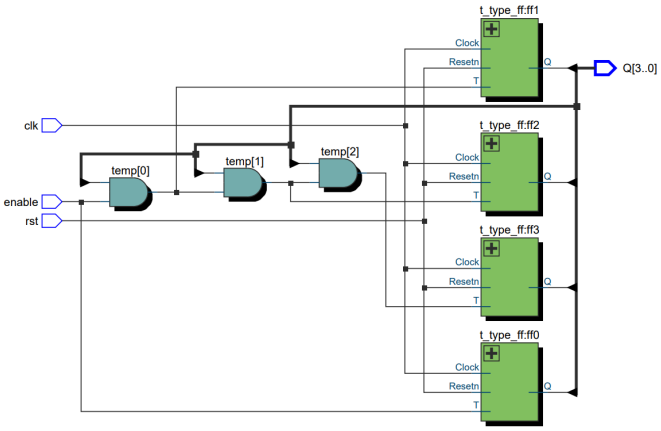

Figure 4: RTL Viewer - 4-bit synchronous counter

4 16-bit synchronous counter version 2

Referring to the previous section, we can simplify the structure of the code by using the statement $Q <= Q + 1$. Since the components in the top entity are always the same, only the code related to the implementation of the counter is reported, *counter.vhd*.
In this case, since it is a less complex circuit, the total number of logic elements is 16, while the maximum frequency f_{max} is 367.65 MHz.

4.1 Synthesis

The differences in this case between the previous implementation and the current one are substantial, as can be seen in Figure 5. In fact, the new circuit uses only a D-type fliplflop on which a sum block is iterated in a way that at each rising edge of the clock the Q value is increased by 1 through a loop. It is also verified that the complexity of the circuit itself is lower than the

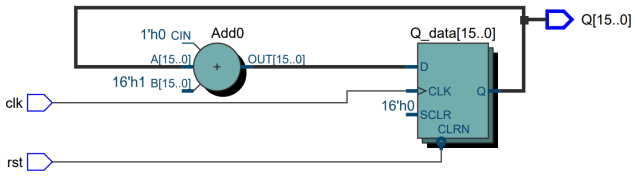

Figure 5: RTL Viewer - 16-bit synchronous counter version 2

previous one while maintaining the correct functioning, as can be seen from the waveform.

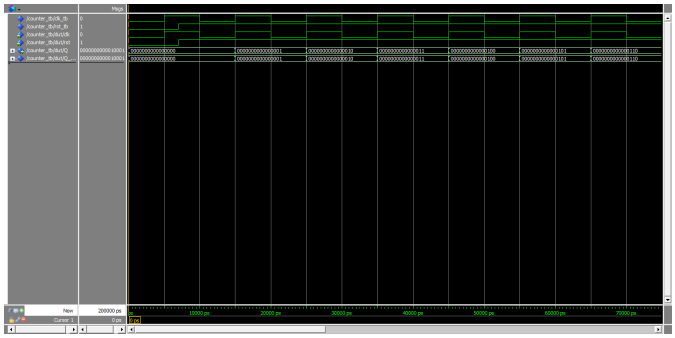

Figure 6: Waveform analysis of the 16-bit synchronous counter version 2

5 Flashing digits from 0 to 9

5.1 Design Entry

The aim of this exercise is to implement a counter that displays numbers from 0 to 9 driven by a 50 MHz clock signal. Among the different choices for the implementation of this circuit we chose a Johnson Counter. The circuit will work as represented in the following schematic.
For this circuit we will need:

- 5 D-type flip-flops to count from 0 to 9 implemented in *flipflop.vhd* .
- A register to store each clock cycle the value to display implemented in *regn.vhd*.
- An encoder to convert the output of the register into the corresponding decimal value displayed on HEX_0 implemented in *display_counter.vhd*.

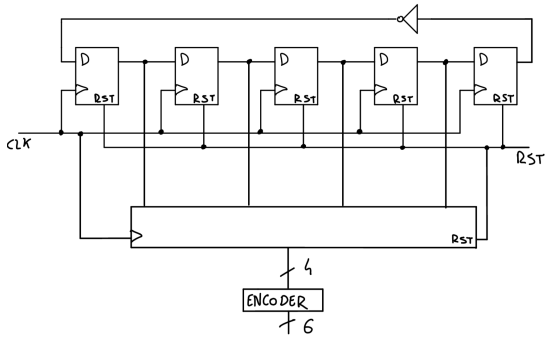

Figure 7: Circuit schematic

Notice that the clock signal will be provided by the DE1 board referring to it as *CLOCK_50* while the reset signal will be triggered by KEY_0.
The top entity is described in the file *johnson_counter.vhd*.

5.2 Functional Simulation

The testbench of the design is reported in the file *johnson_counter.vhd*. In this testbench we have to set a clock period of 20 ns to respect the clock constraints and of course we will switch the clock signal every half period. We implemented the testbench so that we can see at least a complete cycle from 0 to 9 and then we can test the functionality of the reset.

5.3 Synthesis

In the synthesis we analyze two aspects: the simulation of our testbench on ModelSim and the time analysis of the delays inside the circuit. For the delays we can see by the following Figure that the worst case delay causes a maximum operative frequency $Fmax = 680.74$ MHz.

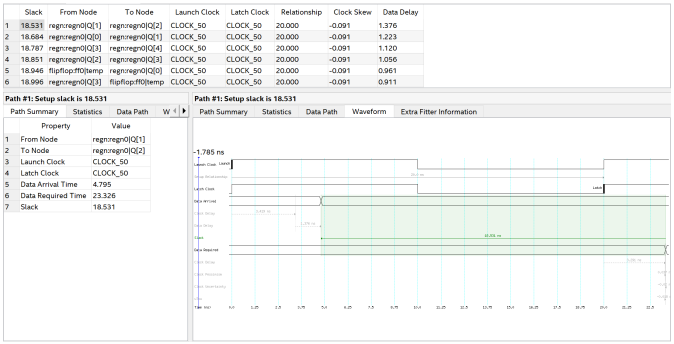

Figure 8: Time analysis

About the testbench simulation we can see in the following Figure (capturing only a portion of the total Wave analysis) the transition on the hexadecimal display of all decimal values from 0 to 9 and the resulting beginning of a new cycle from 0 to 2.
We can also have a look at the RTL viewer to see if the design implemented corresponds to the one drawn previously.

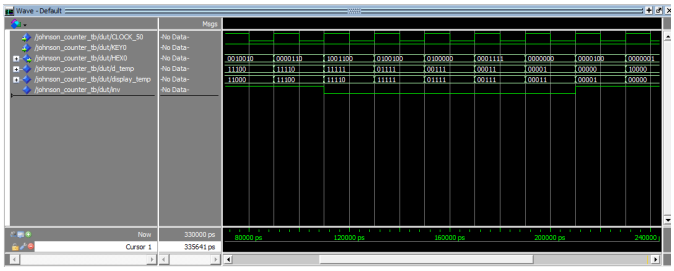

Figure 9: Time analysis

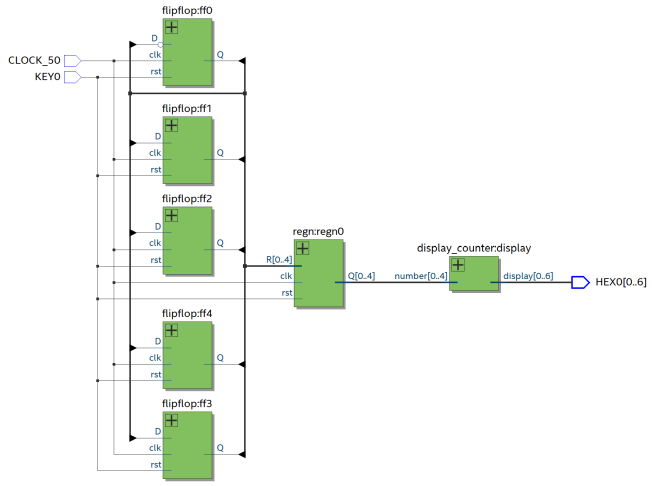

Figure 10: RTL Viewer - Flashing digits

6 Reaction timer

The aim of this exercise is to implement a reaction timer. Despite the amount of tries to implement the circuit correctly we didn't succeed in making it function properly, anyway we will describe the work done. First of all we want do describe in *delay_converter.vhd* the conversion from the $\mathbf{SW_{7\text{-}0}}$ to their corresponding contribution into the total delay (we decided to weigh each bit of a value of 1000 ms, so the maximum delay available will be of 8 s). After the delay is set, we send the value to *counter_delay.vhd* which starts counting until it reaches the delay we chosen previously and after this event it gives as an output a binary signal that enable the real counter beginning. For this second part of the exercise we reused the concept of the Johnson Counter: for each display **HEX*3-0*** we want to count repeatedly from 0 to 9, so we add again with the proper modifications the designs *flifflop.vhd*, *regn.vhd*, *display_counter.vhd* and *johnson_counter.vhd*. We have to make another observation: if we want to see the display updated every ms we need to flash the digit of each 7-segment display with a different frequency. In particular we implemented three frequency divider named *clock_divider_10.vhd*, *clock_divider_100.vhd* and *clock_divider_1000.vhd* to refresh the first three digit of the timer with the proper delay. We also tried to implement three other features: the turn on of $\mathbf{LEDR_0}$ when the count starts, the turn off of the same led when $\mathbf{KEY_3}$ is pressed and the reset of the circuit by pressing of $\mathbf{KEY_0}$. The top entity of the design is described in *reaction_timer.vhd* as for the testbench in *reaction_timer_tb.vhd*.