Report 03

Filippo Nevi - VR458510

October 2020

1 Design choices

1.1 Process style

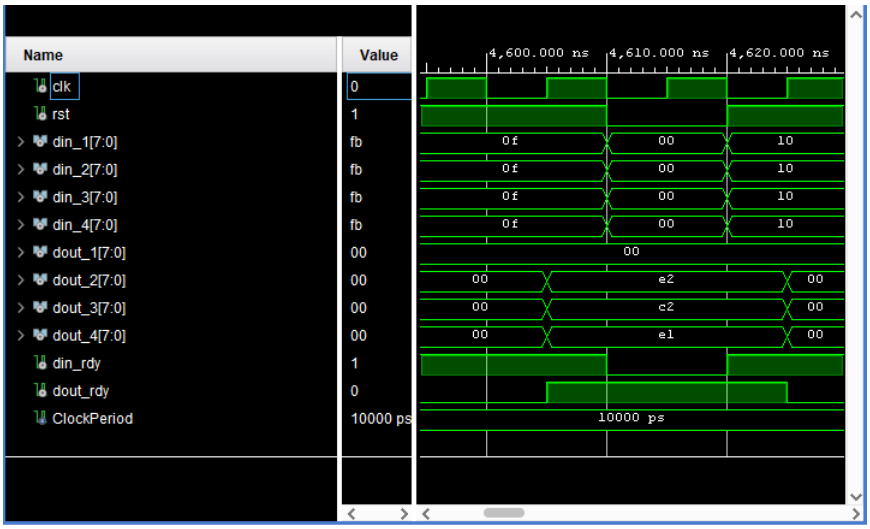
I've chosen to implement the Fixed-point Multiplier using the the **sequential synchronous** style because it represents a Final State Machine with (in my case) five states, so it needs a clock signal to synchronize the circuit.

1.2 Datatypes

I've only used `BIT`, `UNSIGNED` and `INTEGER` datatypes to model this circuit because these types are synthesizable, and since the fixed-point multiplication's algorithm is simply an integer multiplication with, at the end of it, the addition of the point at half of the product, I implemented this circuit without considering the point. The operands and result are converted in their rational form in the Tcl console when executing *stimuli.tcl*.

2 Simulation waveforms

2.1 Testbench



These waveforms show that all the four input signals **din_1**, **din_2**, **din_3**, **din_4** have the same hexadecimal value `0F`.

This integer multiplication is 0F0F·0F0F, and the result signals are

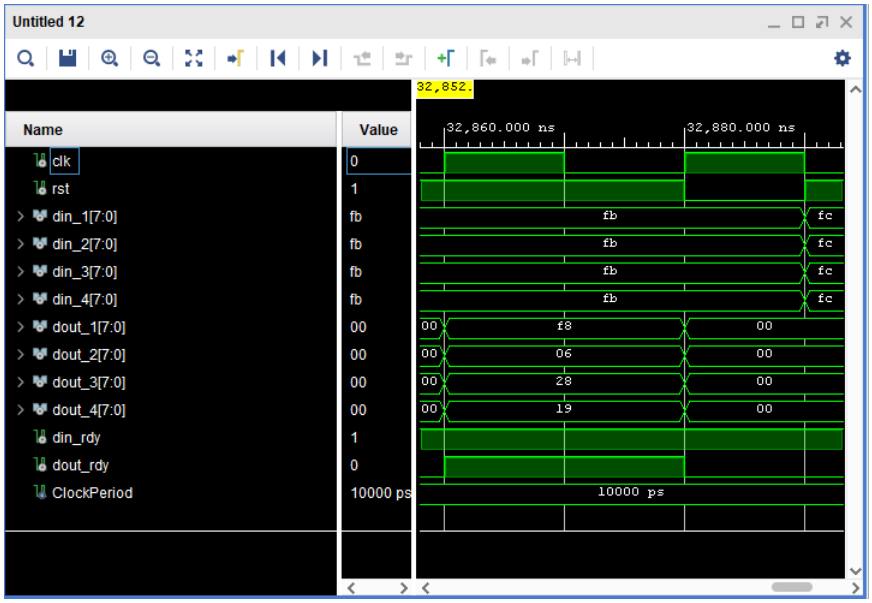
- dout_1 = 00
- dout_2 = E2
- dout_3 = C2
- dout_4 = E1

Converting the numbers in the fixed-point notation, we find out that the operation is:

$$15.05859375 \cdot 15.05859375 = 226.7612457275390625$$

We can also notice that the signal **dout_rdy** has positive value when the output signals are showing the results, while **din_rdy** and **rst** have negative value because the input signals are being reset.

2.2 Stimuli.tcl



In the image above, we can see that all the four input signals have the value `FB`. The multiplication is FBFB·FBFB, and the result signals are the values at time $32860ns$:

- dout_1 = F8
- dout_2 = 06
- dout_3 = 28
- dout_4 = 19

Converting the numbers in Fixed-point notation, the operation is:

$$251.98046875 \cdot 251.98046875 = 63494.15663146973$$