1. Introduction and Objetives

In this work you should employ what you have learned in theory about memory elements and their application in shift register circuits.

Shift register arise from the frequent need of shifting one position all bits of a register, to the left and to the right. It is possible to have linear or circular shift according to the way in which the output and input of the register are connected.

2. Shift Register

a) Design, using type D Flip-Flops, a shift register *serial-in parallel-out* of 4 bits (you should have an input "Serial in", and 4 outputs, $Q_A Q_B Q_C Q_D$). Represent the logic diagram of the designed circuit:

Fill in the following table for the expected state of the outputs immediately after the rising edge of the “Clock” signal: (at the beginning the outputs are all at "0" and in the four following "Clock" rising edges, the input "Serial in" has the bits shown in the table):

Clock	Serial in	Q_A	Q_B	Q_C	Q_D
Initial State		0	0	0	0
$\uparrow t_1$	1				
$\uparrow t_2$	0				
$\uparrow t_3$	1				
$\uparrow t_4$	0				

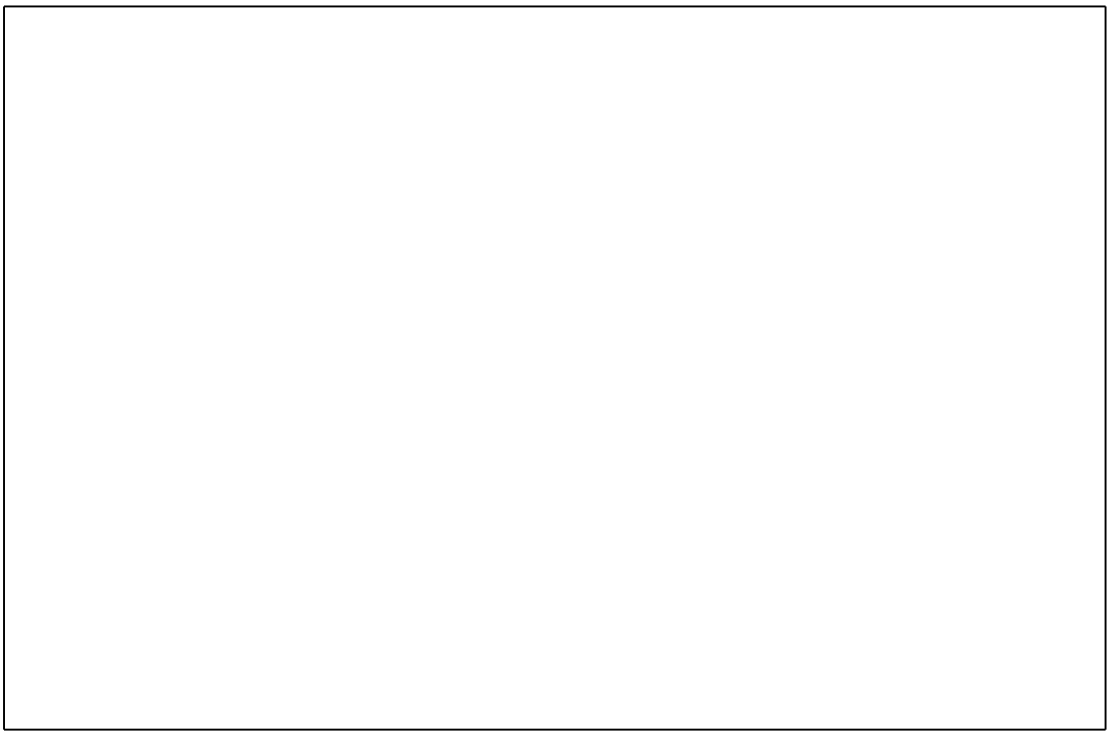
b) Do the simulation in Xilinx ISE, and check the outputs Q_A to Q_D. The “Serial in” input is the data input. The “Clock” input signal should vary periodically between "0" and "1".

Implement the same circuit in a Verilog module, with behavioral level description (not logic gate level). Simulate it.

3. Constant commutation

Keeping the previous circuit, it is now intended to control, by using a *switch*, the alternating functioning between a shift register and constant commutation (the outputs change at each clock cycle between all "1" and all "0"). In this sense, if the *switch* is at "1", the circuit should work as shift register, and if the *switch* is at "0", the circuito works in constant commutation. For that purpose, you should take into account that there are *Preset* and *Clear* inputs on the D-type *Flip-Flop* (see the truth table in the *datasheet*). Sugestion: using a type T *Flip-Flop* (implemented by using a type JK one), and two OR gates, allow to provide a solution to the problem.

Represent the total logic diagram (with the additional logic that implements the previously described behavior):



Note: You should append the *print screens* of the performed simulations, of the designed schematics, and also of the Verilog code.